Analysis Framework for CBM-TRD Test Beam Data

Master Thesis

Philipp Munkes

Westfälische Wilhelms-Universität Münster
Institut für Kernphysik
AG Andronic

Erster Gutachter: Prof. Dr. Anton Andronic
Zweiter Gutachter: Prof. Dr. Christian Klein-Boesing

Münster, September 2020

If we have learned one thing
from the history of invention and discovery,
it is that, in the long run – and often in the short one –
the most daring prophecies seem laughably conservative...

ARTHUR C. CLARKE [Cla51]

Contents

Acronyms

ADC	Analog Digital Converter
AFCK	AMC FMC Carrier Kintex
AMC	Advanced MicroTCA Crate
ALICE	A Large Ion Collider Experiment
ASIC	Application Specific Integrated Circuit
CBM	Compressed Baryonic Matter
CERN	European Organization for Nuclear Research
CSA	Charge Sensitive Amplifier
DAQ	Data AcQuisition
DESY	Deutsches Elektronen SYnchrotron
DORIS	DOppel RIng Speicher
DPB	Data Processing Board
DSP	Digital Signal Processor
FAIR	Facility for Antiproton and Ion Research
FEB	Front End Board
FEE	Front-End-Electronic
FMC	FPGA Mezzanine Card
FNR	Forced Neighbor Readout
FPGA	Field Programmable Gate Array
GIF++	CERN Gamma Irradiation Facility
HERA	Hadron-Elektron-Ring-Anlage
HV	High Voltage
LEP	Large Electron Positron collider
LHC	Large Hadron Collider
LS	Long Shutdown
MWPC	Multi Wire Proportional Chamber
PETRA	Positron-Elektron-Tandem-Ring-Anlage
PKA	Primary Knock-on Atom
PRF	Pad Response Function
QA	Quality Assurance

QGP Quark-Gluon-Plasma

QCD Quantum Chromo Dynamics

ROB Read Out Board

ROC Read Out Chamber

SIS *Schwerionensynchrotron*

SPADIC Self triggered Pulse Amplification and Digitization asIC

SPS Super Proton Synchrotron

STR Self Triggered Readout

TRD Transition Radiation Detector

TSA Time Slice Archive

WA West Area

1 Introduction

A craftsman is only as good as their understanding of their tools. Crafting answers to fundamental questions about the universe requires a deep and comprehensive understanding of the tools used to investigate these questions.

The Compressed Baryonic Matter (CBM) experiment aspires to investigate rare particles and complex phenomena in low energy heavy ion collisions and study them in great detail. Some of these phenomena, like hypernuclei, strange dibaryons, and sub-threshold charm production, require enormous amounts of collisions, since they are very rare, while others like the Quantum Chromo Dynamics phase transition need large amounts of data, since they are very complex. The CBM experiment consists of a variety of detectors that need to work together in order to find answers to the questions asked of this experiment. These detectors need to be able to handle event rates of up to 10 MHz, which have not been achieved in heavy ion physics experiments before.

These rates are up to 2 orders of magnitude higher than existing heavy ion physics experiment [Abl+17] and require detectors designed specifically to handle them. The prototypes of the CBM Transition Radiation Detector (TRD) were therefore intensely tested at various in-beam-tests at different facilities. These tests produce large datasets, which need to be analyzed, ideally in a set of comparable analyses. Software tools for this purpose were developed and are detailed in this thesis.

Heavy ion collisions produce intense ionizing radiation, in particular large amounts of hadrons, like neutrons, that can severely degrade the detectors over time. Neutrons in particular can cause severe damage to materials [KP55; Mat82].

As even failures of small components can cause problems with the experimental setup, the High Voltage (HV) smoothing capacitors were tested for neutron radiation hardness. If these capacitors were to fail in the intense neutron radiation environment expected for the CBM experiment, the whole detector module on which they are installed would fail. It was therefore decided to investigate the radiation hardness of the HV-capacitors.

This thesis describes the irradiation of a number of capacitors considered for the CBM TRD and discusses their reliability under intense neutron radiation doses.

2 Background

2.1 The CBM Experiment

The Compressed Baryonic Matter (CBM) experiment is an upcoming fixed target experiment at the Facility for Antiproton and Ion Research (FAIR) in Darmstadt, see Figure 2.1. It is part of the *Schwerionensynchrotron* (SIS)100 accelerator complex and will be supplied with ion beams of high intensity at energies of up to 14 AGeV (GeV/Nucleon).

The CBM experiment aims to study multiple phenomena in low energy heavy ion collisions, while running at extremely high interaction rate of up to 10 MHz, see [BBE18].

The main task of the CBM experiment is the study of the Quantum Chromo Dynamics (QCD) phase diagram at low temperatures and high baryon densities. In past experiments, a new state of baryonic matter has been observed in high energy heavy ion collisions as early as 1995, see [Raf19, p. 28], at the WA85 (200 AGeV S-S collisions) and WA94 (200 AGeV S-W collisions) experiments. Later experiments around the year 2000 identified this state of matter as the Quark-Gluon-Plasma [HJ00], a state of matter in which quarks and gluons are not confined to individual nucleons, but free within a short lived fireball.

This state of matter is currently being studied by multiple experiments in the high energy/high temperature regime, most prominently by ALICE at CERN. An open question is whether this state of matter can also be observed at low temperatures and high baryon densities, what the order of the phase transition in this region of the QCD phase diagram is and if a critical point exists and where it lies.

In order to study collisions under these conditions, the CBM experiment consists of a wide variety of different detectors, see Figure 2.2. This group of detectors includes the TRD, whose main task is the identification of electrons/positrons against pions.

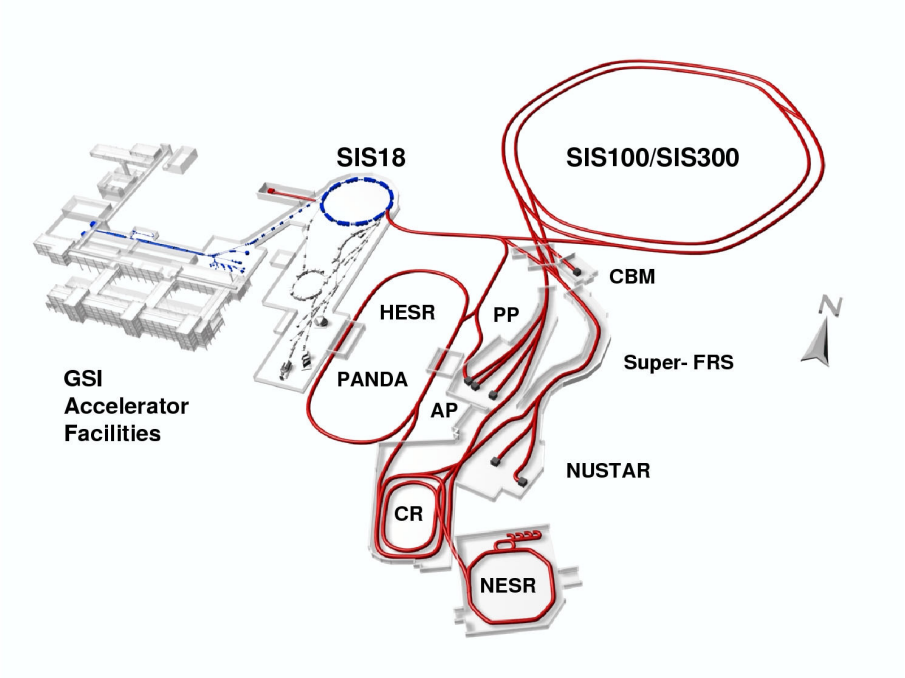


Figure 2.1: Schematic overview of the planned accelerator and experimental complex at FAIR and GSI (*Gesellschaft für Schwerionenforschung*). The SIS300 accelerator is planned for a future extension of the facility. [BBE18]

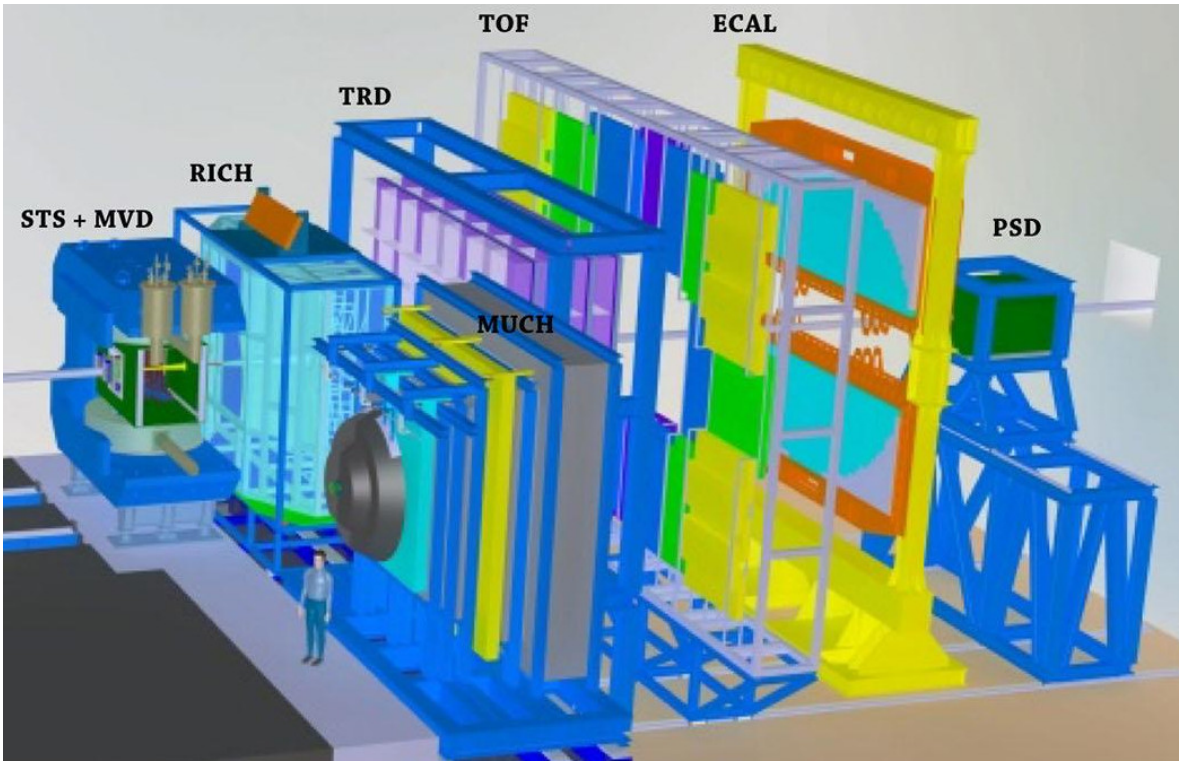


Figure 2.2: Render of the planned CBM experiment at FAIR. The ion beam from the accelerator enters into the experiment from the left side. [Abl+17]

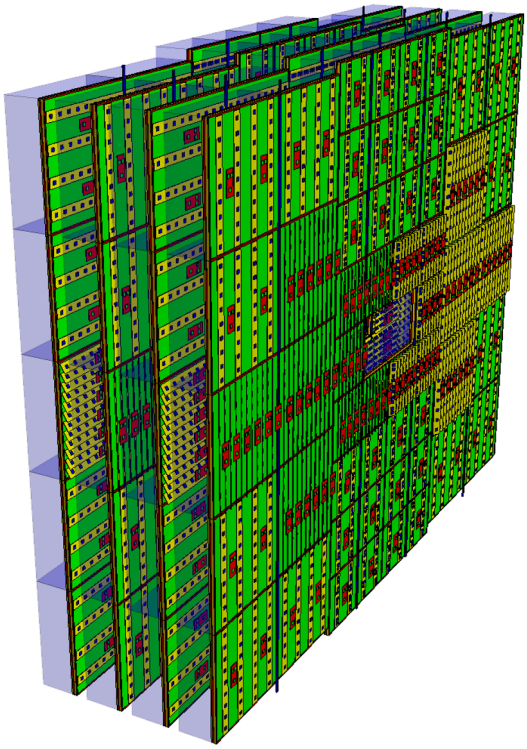


Figure 2.3: Render of the planned CBM-TRD at FAIR. This view is from the back of the detector and shows all four layers planned for the SIS100 setup. The readout chambers (green) are shown equipped with the front end electronics (yellow and red) and the radiators (transparent blue). In every second layer, each chamber is rotated by 90° compared to the first and third layer. [BBE18]

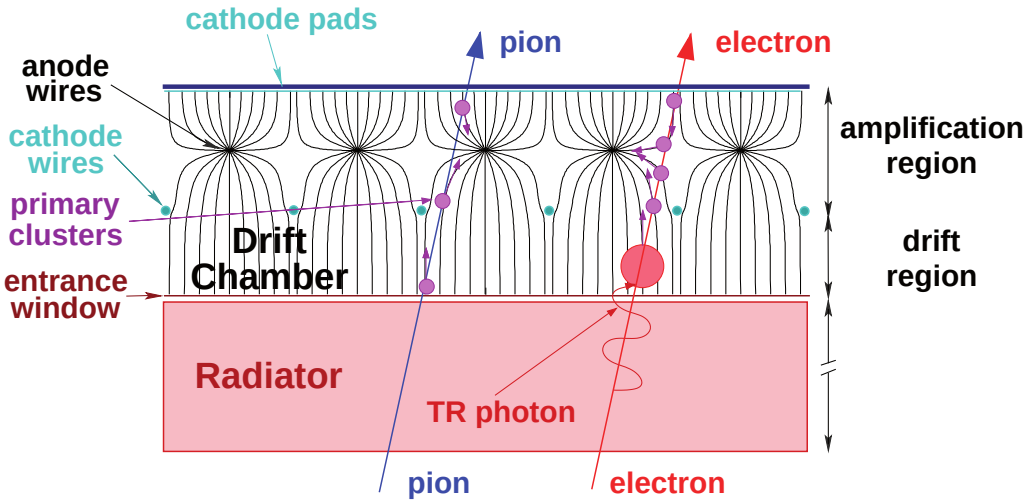


Figure 2.4: Schematic of an MWPC of type drift chamber with a radiator. A charged particle entering the detector may produce primary ionization clusters, the electrons of which will be accelerated towards the anode wires. This causes a charge avalanche that amplifies the signal. Electrons traversing the radiator may produce transition radiation photons, which can be detected when the photon is absorbed in the detector volume. [BBE18]

2.2 The CBM-TRD

Leptons carry information about both – the state of the fireball in a heavy ion collision, as they do not participate in the strong interaction, and the properties of various hadrons, like the J/Ψ. However, in heavy ion collisions, large amounts of pions, which are difficult to distinguish from electrons at energies above 1 GeV/c, are produced. Identifying electrons and pions and tracking charged particles is therefore one of the main tasks of the TRD [BBE18].

The TRD consists of four layers of 54 Read Out Chambers (ROCs) each, with the front end electronics including the SPADIC (Self triggered Pulse Amplification and Digitization asIC), and four layers of radiators, see Figure 2.3. The radiators consist of a passive material with many boundaries between two mediums with differing dielectric constants, while the ROCs are constructed as Multi Wire Proportional Chambers (MWPCs). These MWPCs are read out via the SPADIC Application Specific Integrated Circuit (ASIC), which is connected to the padplane at the back of the chambers. The data stream recorded by the Front-End-Electronics (FEEs) is then recorded via the Data AcQuisition (DAQ) chain and analyzed via a version of CBMROOT.

2.2.1 Multi Wire Proportional Chambers

A Multi Wire Proportional Chamber is a planar gas detector operated in the proportional amplification regime, meaning it is capable of not only determining the position of a charge deposition, but also of resolving the amount of energy loss in the detector. It is a commonly used detector for the instrumentation of large areas and consists of a plane of equally spaced anode wires between two closed cathode planes [Leo87; BRR08]. This detector can be modified into a drift chamber by adding an additional plane of cathode wires in place of one of the cathode planes, and moving one of the original cathode planes further away, as shown in Figure 2.4. The extension provides a longer path through the detector for the particles, increasing the amount of primary ionizations for a charged particle and improving the absorption of TR photons.

These chambers are filled with a mixture of a noble gas and an organic quench gas. In the case of the CBM-TRD, this is planned to be a mixture of (Xenon/CO_2 (85/15)) [BBE18]. Applying high voltage to the electrodes will result in an electric field similar to the one sketched in Figure 2.4, where charged particles are accelerated along the field lines towards the anode or cathode respectively.

A charged particle traversing the detector can produce a number of primary ion clusters. When the electrons from these primary ion clusters get close to the anode wires, they are accelerated due to the $1/r$ dependence of the electric field and produce new electron/ion pairs themselves. This is called a charge avalanche. This process is repeated multiple times for each electron, causing the creation of hundreds or thousands of electron–ion pairs for every primary ion. The detector can then be read out either directly via the signal on the wires or indirectly by measuring the induced mirror charge on the padplane, the latter of which is the method used in the CBM-TRD, see Figure 2.5.

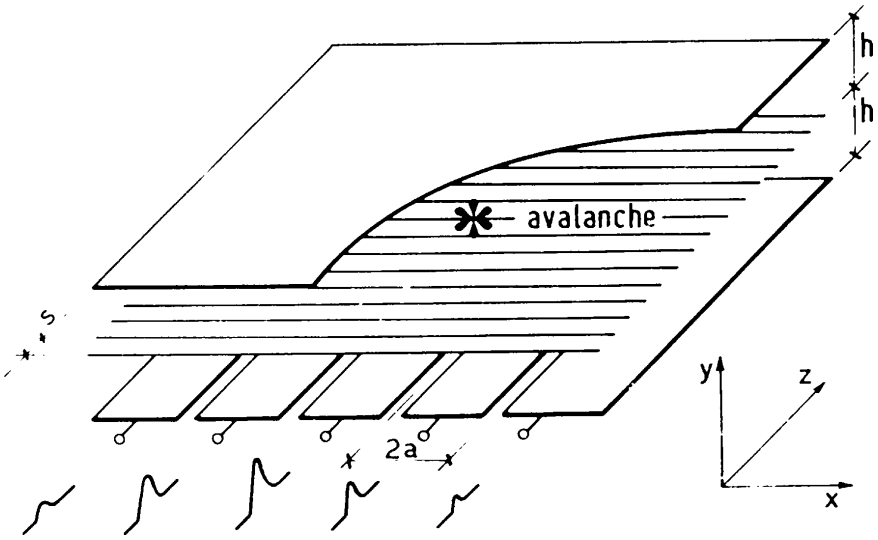


Figure 2.5: Sketch of an MWPC with cathode pad readout. One of the cathode planes is divided into pads, which are connected to the front end electronics. The closer a pad is to the center of the avalanche, the larger the fraction of the cluster's charge detected on this pad is, as is indicated by the height of the pulses below the pads. The total deposited energy of the cluster can be reconstructed by adding up the charge on all cathode pads. [BRR08]

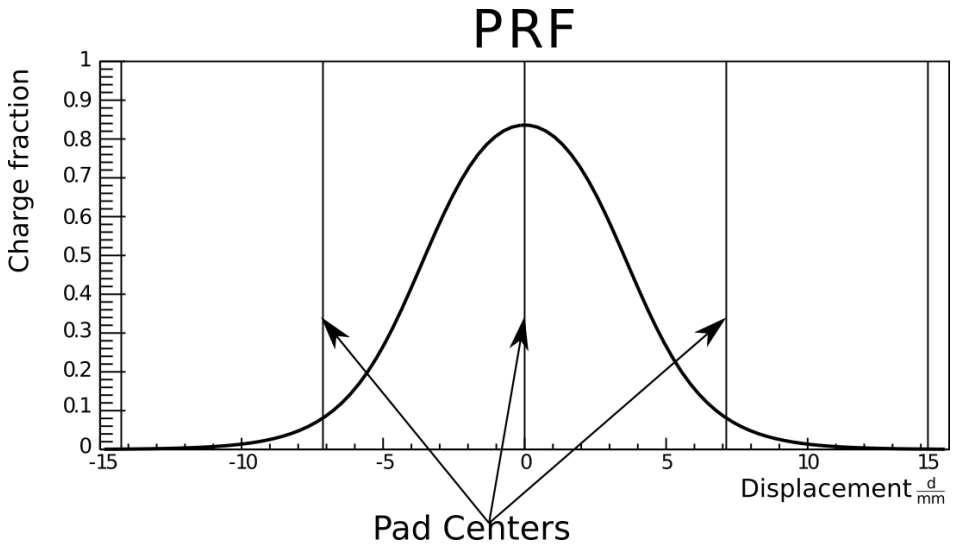


Figure 2.6: Sketch of the expected PRF for the MWPCs used for the In-Beam-Tests analyzed in this thesis, according to equation (2.2). The plot shows the charge fraction as a function of the displacement, i.e. the distance from the center of the charge avalanche. The parameters used are $K_3 = 0.38$ and $W = 7.125\,\text{mm}$. The vertical lines denote the sampling points for a charge avalanche directly above the central pad. [Mun16]

The induced charge cluster on the cathode pad plane is not detected as a point, but it is spread out over a larger area, which can be described via an empirical formula found by Gatti et al. [Gat+79]. This formula will be used in the form proposed by Mathieson [Mat88]:

$$\rho(d/h) = q_a \cdot \frac{\frac{\pi}{2} \cdot \left(1 - \frac{\sqrt{K_3}}{2}\right)}{4 \arctan\left(\sqrt{K_3}\right)} \cdot \frac{1 - \tanh^2\left(\frac{\pi}{2} \cdot \left(1 - \frac{\sqrt{K_3}}{2}\right) \frac{d}{h}\right)}{1 + K_3 \tanh^2\left(\frac{\pi}{2} \cdot \left(1 - \frac{\sqrt{K_3}}{2}\right) \frac{d}{h}\right)}, \tag{2.1}$$

with the geometric detector parameter K_3, the distance h between the padplane and the anode wires and the displacement d, which is the distance of the charged particle from the next pad center. The induced charge cluster cannot be sampled continuously, due to the discrete nature of the pads used for measurement, therefore equation (2.1) must be integrated over a single pad of width W [Ber14]:

$$\begin{aligned} \mathrm{PRF}(d/h) &= \int_{d/h - W/2}^{d/h + W/2} \rho(d'/h)\ d(d'/h)\ \mathrm{d}d' \\ \Rightarrow \mathrm{PRF}(d/h) &= -\frac{\arctan\left(\sqrt{K_3} \tanh\left(\pi\left(\sqrt{K_3} - 2\right) \cdot \frac{W - 2 \cdot d}{8 \cdot h}\right)\right)}{2 \arctan\left(\sqrt{K_3}\right)} \\ &\quad - \frac{\arctan\left(\sqrt{K_3} \tanh\left(\pi\left(\sqrt{K_3} - 2\right) \cdot \frac{W + 2 \cdot d}{8 \cdot h}\right)\right)}{2 \arctan\left(\sqrt{K_3}\right)} \end{aligned} \tag{2.2}$$

The resulting Pad Response Function (PRF) can then be plotted to receive Figure 2.6. This plot assumes $K_3 = 0.38$, a value expected for the geometric parameters of the CBM-TRD MWPC prototypes [Ber14] and consistent with previous attempts at reconstructing this measurement [Mun16].

2.2.2 The SPADIC

The SPADIC is an integrated readout ASIC with 32 channels developed specifically for the CBM-TRD [Arm13]. A prototype Front End Board (FEB) for the SPADIC 1.0 is displayed in Figure 2.7. A SPADIC features two groups of 16 channels, with each channel consisting of a Charge Sensitive Amplifier (CSA), a continuously running Analog Digital Converter (ADC), a Digital Signal Processor (DSP), and hit detection logic, see Figure 2.8. If a hit is detected on a channel, the sampled data stream is recorded and sent to the DAQ chain for analysis in the form of a hit message. The hit message can contain up to 32 samples, with the recorded samples being fully configurable. For example the SPADIC may be configured to only record up to sample 4 and skip samples 2 and 3. This message would therefore only contain the samples [0,1,4].

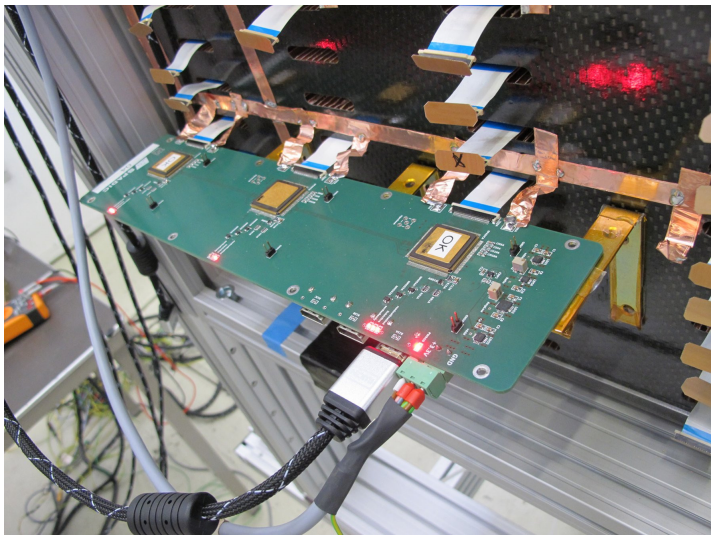


Figure 2.7: Photo of a SPADIC 1.0 FEB with three SPADICs on the back of an MWPC. [BBE18]

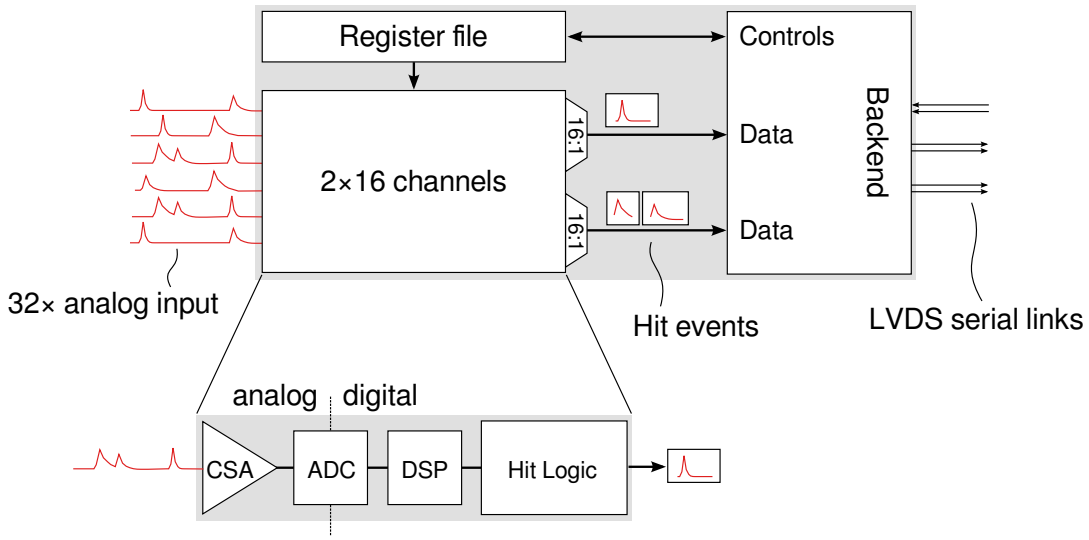


Figure 2.8: Conceptional block diagram of the SPADIC chip. The 'Backend' block is responsible for communication with the DAQ and implements the CBMNet interface in SPADIC 1.x and an E-link interface in SPADIC 2.x. [BBE18]

Charge Sensitive Amplifier The CSA of the SPADIC contains a shaper and amplifies a charge pulse on one of the pads inside the MWPC into an electric pulse signal shaped according to its step answer function [Arm13]:

$$V_{\text{shaper}}(t) = Q_i \frac{\text{const}}{\tau_s^2} t \; e^{-\frac{t}{\tau_s}}, \qquad (2.3)$$

with the induced charge Q_i and the time constant of the shaper, the *shaping time* τ_s. An overlay of pulses digitized by a SPADIC 1.x ASIC is displayed in Figure 2.9. The characteristic shape of the pulse is clearly visible.

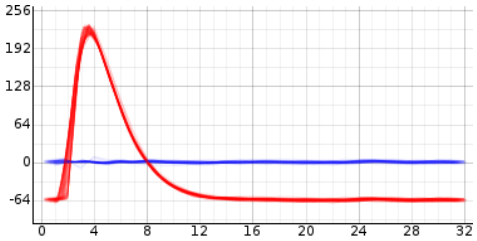


Figure 2.9: Digitized pulses generated by test injection into a SPADIC channel. The flat blue lines are the residuals of fits to the predicted pulse shape.[BBE18]

Self Triggered Readout The SPADIC was designed to run without an external trigger. This means that the ASIC needs to be able to trigger a readout based on the stream of ADC samples, a mechanism which is called Self Triggered Readout (STR). The implementation of this trigger differs between the SPADIC versions, but it can be configured to run in either a differential or an absolute mode with two threshold values *t0,t1*. All measurements in this thesis were recorded in the absolute mode, where three continuous samples *a, b, c* would need to satisfy the trigger condition with the threshold values *t0,t1*. For the SPADIC 1.1 this can be generally described via the following C++-Code:

```
bool g = (a > t0) && (b > t1)
bool h = (b > t0) && (c > t1)
bool t = !g && h
```

Should `t` be evaluated to be true, the trigger condition is fulfilled and the channel will record the configured set of samples and send a hit message with the trigger type 1. Hit type is an alternative way of labeling the trigger type and may be used synonymous. The SPADIC 2.0 trigger logic is similar to the described mechanism.

Forced Neighbor Readout Due to the shape of the PRF, see Figure 2.6, the SPADIC was designed to be able to trigger a readout on more than the channel that detected a hit. This mechanism is called the Forced Neighbor Readout (FNR), or colloquially neighbor triggering, and is fully configurable within a 16-channel group and capable of transmitting a FNR signal to a different channel group on up to three different FNR lanes. It allows the two trigger thresholds to be set to reasonably high values, to avoid triggering on noise, while allowing most of the cluster charge to be sampled. The SPADIC is therefore usually configured to read out at least the two neighboring pads to the triggering pad, i.e. the central three pads of a cluster.

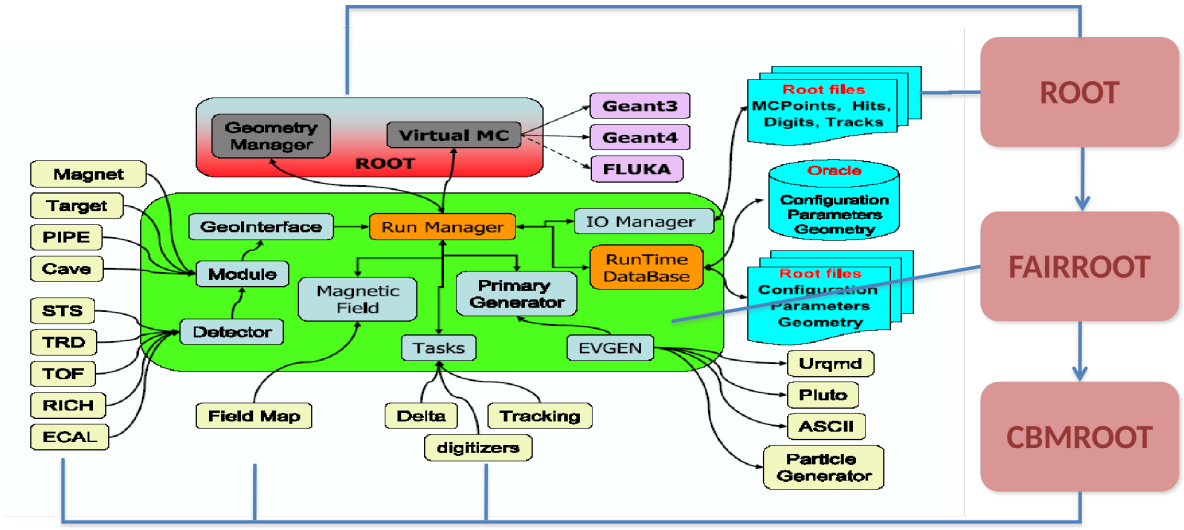


Figure 2.10: Sketch of the structure of the CBMROOT Framework, including the software packages it was split into since the start of development. The work in this thesis is based on the full CBMROOT framework. [AB06; Fri19]

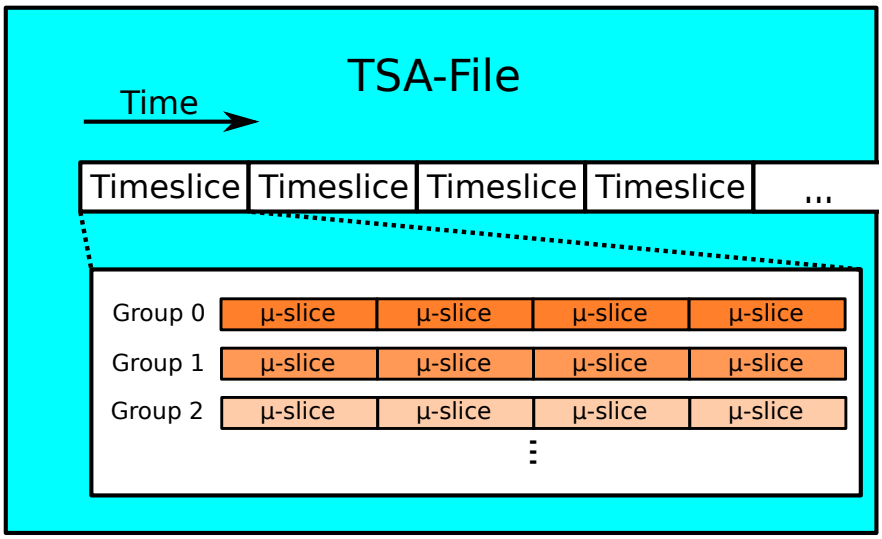


Figure 2.11: Sketch of the structure of a Time Slice Archive file. A TSA-file is split into timeslices, which contain the data from the full setup within a defined time period. The timeslices are split into micro-slices, which contain all information from a data source in the setup within a specific time period. The µ-slice bitstream is defined by the individual detector groups. In case of the TRD, data from 16 SPADIC channels is collected in a µ-slice.

A hit message can have both trigger types 1 and 2, which is referred to as trigger type 3. This occurs if a message is triggered via both the STR and the FNR mechanisms. The full set of trigger types is shown in Table 2.1:

Table 2.1: List of possible Trigger Types on SPADIC 1.x and 2.0

Trigger Types	Short Form	Long Form
0	DLM	Global Trigger
1	STR	Self Triggered Readout
2	FNR	Forced Neighbor Readout
3	SFR	Self triggered and Forced neighbor Readout

Parameters In this thesis, datasets from both SPADIC 1.1 and 2.0 are analyzed. These ASICs are similar, but have different configurations. A comprehensive breakdown of the differences can be found in [BBE18], the most important parameters will be reproduced in Table 2.2.

Table 2.2: Parameters of the SPADIC 1.1 and 2.0 ASICs examined in this thesis

	SPADIC 1.1	SPADIC 2.0
Shaping time	80 ns	240 ns
Sampling frequency (Design)	25 MHz	16 MHz
Sampling frequency (Typical)	16 MHz	
Protocol	CBMNET	E-link

Timestamps In order to reduce the size of an individual message from the SPADIC, the timestamp in a SPADIC message only has a width of a few bits. In case of the SPADIC 1.1 and 2.0, the timestamp is 12 bits wide and ticking with a frequency of 16 MHz. This timestamp wraps around every 256 µs and can therefore not be used to generate an absolute timestamp from the beginning of the run. The SPADIC sends epoch markers (in the form of epoch messages in the context of the SPADIC) whenever this overflow occurs and these can be used to reconstruct the full timestamp. Every channel group of 16 has its own clock and will therefore generate its own timestamp and epoch markers, with the synchronization between the channel groups being handled by the DAQ .

2.2.3 CBMROOT

This thesis is based on the CBMROOT framework [AB06], the general structure of which can be seen in Figure 2.10. CBMROOT provides tools for simulation and data analysis. These tools include an event loop for analysis tasks, which can provide functions to perform parts of the analyses at various stages of an analysis run.

The tools provided by CBMROOT also include the necessary infrastructure to unpack the TSA-files (Time Slice Archive) produced by the DAQ . The data from the detectors are recorded in a set of self contained timeslices that are then written into a Time Slice Archive (TSA), see Figure 2.11. These timeslices contain the full amount of data recorded by the DAQ in a defined timeframe. The timeslices are further organized into µ-Slices, which contain the data from an individual channel group for a fraction of the duration of the timeslice. All data analyses in this thesis are based on the analysis of data at the timeslice level, the substructure is provided for reference.

2.3 Interaction of Neutrons with Matter

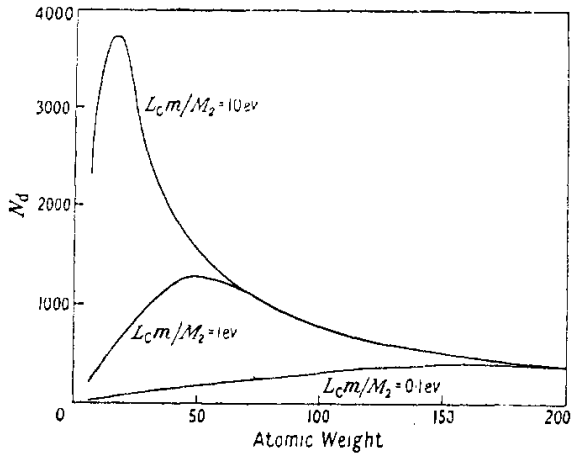


Figure 2.12: The average number of displaced atoms per primary knock-on produced by 2 MeV neutron bombardment as a function of atomic weight. The parameter $^{L_C m}/_{M_2}$ is a material constant defined in the source article [KP55].

The interaction of neutrons with matter has been studied for a long time [Dun+35] for multiple reasons. These reasons include the effect neutron radiation has on materials in nuclear power plants and particle physics experiments, where it causes the degradation of components. One of the primary mechanisms for neutron radiation damage in metals and ceramics is the displacement of atoms after a collision between a fast neutron ($E_{\text{kin}} > 1\,\text{MeV}$) and the Primary Knock-on Atom (PKA) [KP55; Mat82]. This PKA has an enormous amount of kinetic energy and will travel from their lattice spot through the crystal lattice while displacing a large number of lattice atoms, see Figure 2.12 for example.

Elastic scattering of neutrons can also be used to detect them with proportional counters filled with light gases. The nuclei of light gases like ^{3}He have comparable masses to the neutrons and large cross sections for the scattering of thermal neutrons, which is a reason these are used in commercial neutron detectors [Bur+97]. As the cross section for an interaction is large for thermal neutrons, these counters are usually surrounded by a moderating material like polyethylene that slows fast neutrons down before they enter the proportional counter.

3 Development of a New Analysis Framework

3.1 In-Beam-Test Activities and Analysis Framework

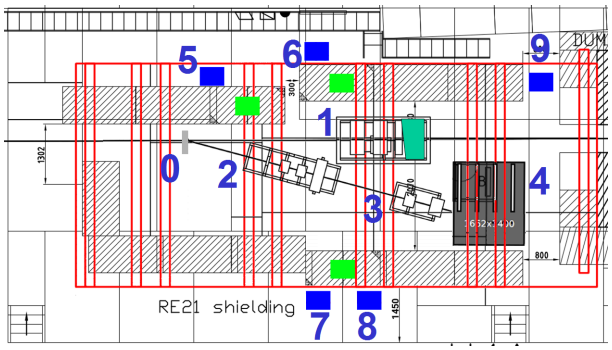
Different CBM-TRD prototypes have been tested at various facilities, including measurement campaigns at DESY, GIF++ and SPS. Described here is the development of a framework for the analysis of in-beam-test datasets, that is generalized in a way that the individual analyses are applicable to different campaigns. The development of this framework was started in preparation for the campaigns at Deutsches Elektronen SYnchrotron (DESY) II and CERN Gamma Irradiation Facility (GIF++) in 2017 and it was designed so that it would be usable for the analysis of data from the Super Proton Synchrotron (SPS) campaign in 2016, as well as other future and past campaigns. The potential generality in the analysis software between different campaigns enables comparisons between data from different campaigns.

This framework provides a set of tools for the analysis of multiple in-beam-test datasets on a common software stack and a set of basic analyses in order to asses data quality. A comparable set of analyses for the three mentioned campaigns will be constructed and discussed in this thesis.

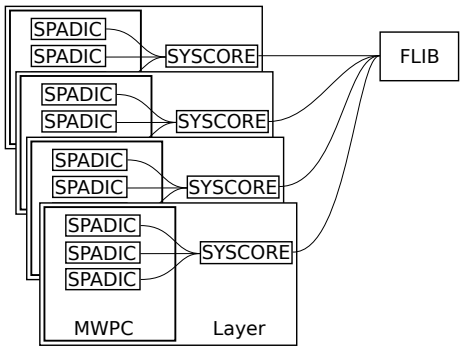
3.1.1 In-Beam-Test at SPS in 2016

The SPS is an accelerator at *CERN*. It started operating in 1976 and has since then continued to provide proton, antiproton and ion beams for fixed target experiments, both permanent installations and temporary installations for test beam campaigns. It is currently also used as the final pre-accelerator for the Large Hadron Collider and has been used as the pre-accelerator for the Large Electron Positron collider. Temporary experiments can use the beamline facilities at the North Area for detector development and other studies which require high energy particle beams.

In 2016 the CBM collaboration performed a test beam campaign at the H4 beamline for the TRD, Time-of-Flight and the muon detector prototypes, see Figure 3.1. This campaign was the first in which the TRD was operated in a 4-layer setup similar to the final setup at SIS100 and also the first in which a complex DAQ chain with multiple SYSCORE 3 Field Programmable Gate Array (FPGA) boards was used, for a total of twelve SPADIC 1.1 FEBs in the same experimental setup, see Figure 3.1b. This setup, with every second detector layer rotated by 90°, enables tracking along multiple detector layers in the X and Y direction for the first time, based only on CBM-TRD position measurements.



(a) Layout of the RE21 experimental area during the 2016 SPS CBM In-Beam-Test. 0:Pb-foil target, 4:Münster/Frankfurt TRD prototypes, 8: TRD gas analysis rack, 9: service and readout rack for Münster/Frankfurt TRD prototypes [ST17]



(b) Logical setup of the 2016 SPS CBM In-Beam-Test setup.

(c) Photo taken from the top of the target. The TRD setup has been rotated to a projective orientation relative to the target. (Picture: Cyrano Bergmann)

Figure 3.1: Photograph, floor plan and logical setup of the experimental setup at the H4 beamline at SPS in 2016

3.1.2 In-Beam-Test at DESY II in 2017

The DESY II Test Beam Facility is part of an accelerator complex operated by the DESY research center in Hamburg [Die+19]. This complex comprises of multiple particle accelerators and experimental setups, examples being DORIS, PETRA III and DESY II. DESY II is a circular synchrotron with a circumference of 292.8 m and electron/positron energies up to a maximum of 7 GeV [Die+19, p. 266]. It started operating in 1985 as a pre-accelerator for the HERA collider and continues to serve that role for the current DOppel RIng Speicher (DORIS) and Positron-Elektron-Tandem-Ring-Anlage (PETRA) III accelerators. At the DESY II Test Beam Facility, electron beams at momenta of up to 6 GeV are available for detector studies.

In 2017 the TRD prototypes were set up at the DESY II Test Beam Facility in Hamburg in order to study the Transition Radiation performance of different radiator configurations in combination with near to final electronics. The detectors were set up along the beam axis according to Figure 3.2 with 4 detector layers being operated with radiators, 2 layers of position reference MWPCs and 2 time reference scintillation detectors being available.

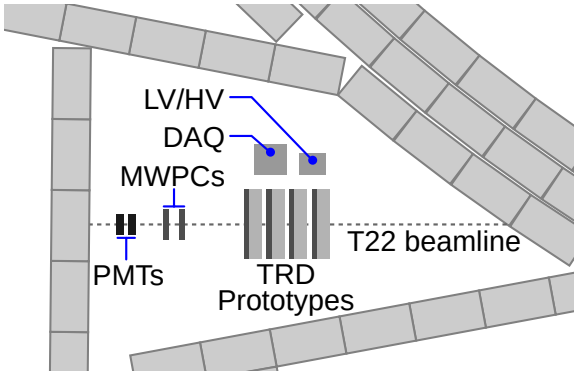
The DAQ was configured as seen in Figure 3.2b, with one of the reference MWPCs not being read out. Some of the main differences to the equipment at SPS were the use of SPADIC 2.0 FEBs and the replacement of the SYSCORE FPGA Board by the AMC FMC Carrier Kintex (AFCK) board. All layers, besides layer 4, were instrumented with at least one SPADIC 2.0 FEB with the front MWPC being outfitted with an additional SPADIC FEB for an ^{55}Fe source for the energy calibration.

3.1.3 In-Beam-Test at GIF++ in 2017

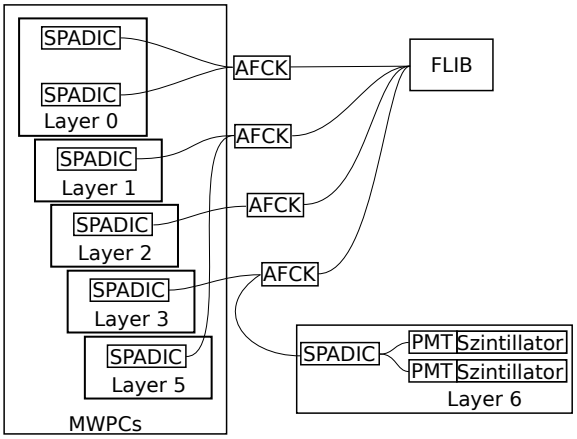
The GIF++ is a dedicated facility for the testing of particle detector prototypes [Gui15]. It is located at the H4 beamline of the SPS accelerator and features a 14 TBq ^{137}Cs source. This ^{137}Cs source can provide an intense photon flux, which can be combined with the high energy μ beam from the H4 beamline to study the behavior of detectors in environment with high flux rates of photons. The high intensity photon flux can further be used to age detectors, in order to study the long term performance of the design.

The CBM-TRD group used this facility in 2017 to test the high rate behavior of the detector and the DAQ . For the setup, a single MWPC prototype and the two scintillation detectors used previously at DESY II were operated at GIF++. The MWPC was placed in front of the γ source in a region of constant photon flux, while the scintillation detectors were placed in the shielded area beside the source, see Figure 3.3a. Five SPADIC 2.0 FEBs were used at GIF++, with four being placed next to each other in a row on the MWPC and an additional SPADIC 2.0 board being connected to the scintillation detectors.

This setup was then exposed to various γ flux intensities and the μ beam.



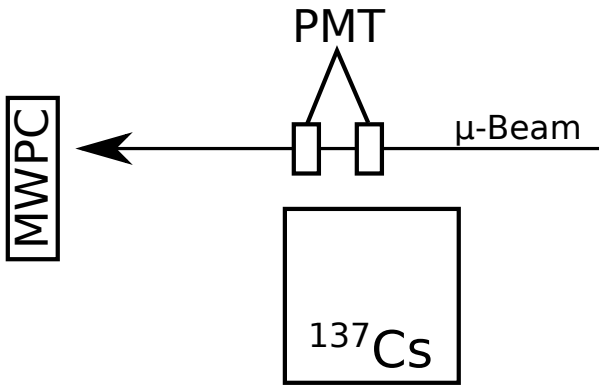
(a) Floorplan of the 2017 DESY In-Beam-Test setup.[BBE18; TS18]



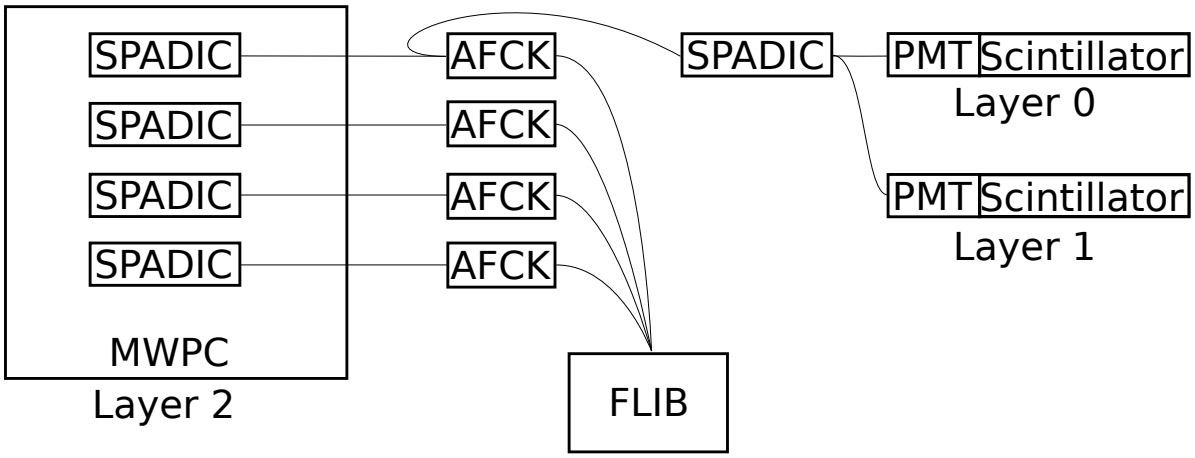
(b) Logical setup of the 2017 DESY II CBM In-Beam-Test setup

(c) Photograph of the setup with the 5 active MWPCs and the 2 Photo Multiplier Detectors. (Picture: Florian Roether)

Figure 3.2: Photograph, floor plan and logical setup of the Experimental Setup at the T22 beamline at DESY in 2017



(a) Floorplan of the 2017 GIF++ In-Beam-Test setup, the μ-Beam is not always available.



(b) Logical setup of the 2017 GIF++ In-Beam-Test setup.

(c) Photograph of the setup at GIF++ [ST17]

Figure 3.3: Photograph, floor plan and logical setup of the Experimental Setup at the GIF++ facility in 2017

3.2 Development of a New Analysis Framework

3.2.1 Design Goals and Structure

Previous Test Beam activities involved writing a tailored set of analyses for the specific circumstances of the experimental setup in CBMROOT [AB06]. In order to unify the analysis of the data taken at these different Test Beam activities, the author was motivated to develop a new framework in 2017 for the then upcoming in-beam-test campaigns at DESY II and GIF++.

This new framework should enable the analysis of past, such as the SPS 2016 one, and future campaigns, as well as customized laboratory setups. It should also provide a modular approach to the composition of a specific run of analysis. Extending the usability of an analysis to multiple in-beam-test campaigns requires a robust raw data unpacker, the introduction of a parameter handler and the replacement of previously hardcoded values by calls to this handler, as well as splitting the analysis into smaller more modular tasks, see Figure 3.4. The modular approach to analysis composition enables disabling unused parts of the analysis for performance, as well as replacing parts of an analysis pipeline.

Splitting the parameters from the analysis firstly required a formulation of the necessary parameter set for a generalized analysis and cataloging appropriate values for them:

- Mapping of SPADIC channel to detector pad
  - Complex due to the varied topologies of the Data connections, see e.g. Figures 3.1b, 3.2b and 3.3b
  - Uses the existing CbmTrdAddress infrastructure
- Provide appropriate numbers of active components
  - SPADICs per Data Processing Board (DPB)/Read Out Board (ROB)
  - Active DPBs
- Front-end parameters
  - Clock rates, shaping times of the analogue front end
- Geometric Parameters
  - Layers, Columns and Rows per Layer
  - Pad width and height

Furthermore, the framework includes a set of classes intended to serve as examples for users to construct their own analyses and provides a set of basic Quality Assurance (QA) tasks. The most important included classes and their intended use are listed here:

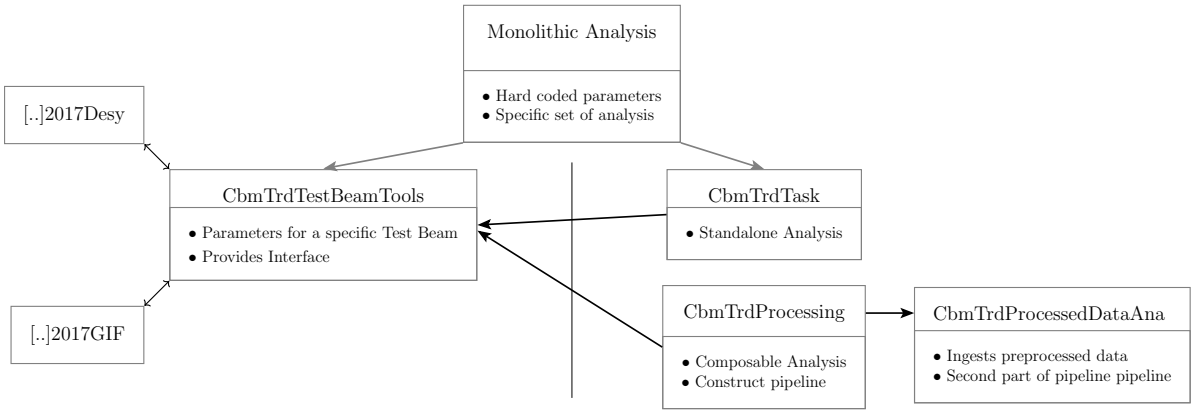


Figure 3.4: Schematic of the framework structure. The monolithic analysis for a specific Test Beam is split into a set of Test Beam Tools (left) and modular analyses (right). The Test Beam Tools can be exchanged freely to perform an analysis on a specific dataset. The modular analyses can either be standalone or they can be used to build processing pipelines for more complex analyses. Parameters can be queried via documented calls to the test Beam Tools.

CbmTrdQABase This is the base class for analyses in the in-beam-test analysis framework, it provides the base functionality. It fetches the necessary and available data array pointers, fRaw for CbmSpadicRawMessages, fDigi for CbmTrdDigi and fClusters for CbmTrdClusters, and the pointer fBT for the currently used instance of the CbmTrdTestBeamTools. It also initializes the histogram manager fHm and saves the included histograms after the analysis run has concluded in the output ROOT-file. It provides a basic analysis of timestamps of the incoming raw messages.

CbmTrdQAHit This is the first proper QA class that is derived from CbmTrdQABase. It provides a varied set of QA plots for CbmSpadicRawMessages that can be used to spot and diagnose phenomena at the level of an individual hit message with the current setup.

CbmTrdSimpleDigitizer This class performs a conversion from CbmSpadicRawMessage to CbmTrdDigi. It determines the full timestamp, the charge of the Digi, extracts various metadata and creates a new CbmTrdDigi object from the raw data. These CbmTrdDigi objects represent the smallest unit of information from the SPADIC and they can theoretically be processed by the existing facilities in CBMROOT, provided the setup is available both in the in-beam-test analysis framework and in the wider CBMROOT.

CbmTrdSimpleClusterizer This class performs a cluster search according to the algorithm developed by the author of this thesis in his Bachelors thesis, see [Mun16]. This algorithm provides basic functionality for a preliminary analysis of the data at a sufficient performance level for offline analysis. It outputs CbmTrdClusters, which are intended to consist of spatially and temporally close CbmTrdDigis that are expected to belong to the same physical hit.

The CbmTrdClusters can be classified into categories according to the criteria in Table 3.1:

Table 3.1: Classifications of CbmTrdClusters in the In-Beam-Test Framework

Classification	Description
kNormal	A normal well formed Cluster
kMissingSTR	A malformed Cluster without STR triggered digis
kMissingFNR	A malformed Cluster with at least one missing FNR triggered digi
kInvalidCharge	A Cluster with at least one negative charge value
kFragmented	Reserved: A Cluster that was created due to a noise induced retrigger
kEmpty	A Cluster without Digis

CbmTrdSimpleClusterAnalysis This class performs a simple analysis of various key parameters of the full set of CbmTrdClusters. The performed analyses are a set of hitmaps (called heatmaps in the code) of all reconstructed clusters in all populated detector layers, a set of reconstructed Pad-Response-Functions for the individual layers and a set of unfiltered and uncalibrated preliminary spectra.

In order to describe the full process of this framework, a modified version of the OCT19 release of CBMROOT will be used – the full patch set is available[1] – and an example analysis class will be constructed and discussed.

3.2.2 An Example Analysis Class

Constructing an analysis task in the in-beam-test analysis framework is simple, a full example is available in listings A.1 to A.3. It requires writing a C++ class which is derived from CbmTrdQABase. This class needs to declare its name to the called CbmTrdQABase constructor, declare a virtual destructor and implement the following two functions:

CreateHistograms() This function is called during initialization and used to create histograms and other data structures for the analysis. No datapoints are available at this point.

Exec (Option_t*) This function is reimplemented from FairTask and is called once per task for each timeslice in a TSA-file. The Option_t* pointer is a required parameter for the function by FairRoot, but not used in this example. All available datapoints can be used for analyses at this point.

[1] Link to the full patch set:https://uni-muenster.sciebo.de/s/4wxXZAOQucvVXhq

In this example class four analyses will be constructed and their efficiency will be evaluated w.r.t. the fraction of processed data. These four analyses will be an evaluation of the accumulated signal shape for a single channel and self triggered data, the amount of different trigger types on these channels, a reconstruction of the PRF on an individual detector layer and a hitmap of reconstructed clusters in an individual layer.

Trigger Types As outlined in Section 2.2.2, hit messages from the SPADIC are tagged with the corresponding trigger condition, either called synonymously hit type or trigger type, see Table 2.1. The ratio of these messages on specific channels should correspond to the amount of messages triggered via the FNR readout, with deviations helping to identify problems, like noisy channels or a misconfigured FNR-mechanism, on individual channels and their neighbors.

Signal Shapes Hit Messages from the SPADIC include samples of the individual electronic signal recorded by the individual channel. This representation of the cumulative signal shapes can be used to identify problems with the individual channel like noise or insufficient trigger thresholds.

As both of these analyses are closely linked, they will be discussed at the same time. Firstly, histograms for these measurements need to be created, which is specified in listing 3.1. As a histogram for each individual channel on every possible SPADIC in the setup will be created, the number of Data Processing Boards (called ROBs in the framework for historic reasons) and the amount of possible SPADICs per ROB needs to be queried. In order to keep the analysis as simple as possible, the logical channel groups of 16 channels, or half a SPADIC will be used instead of the full 32 channels of the ASIC. One SPADIC FEB with 32 channels will therefore appear as 2 „Half_SPADICs“ with 16 channels each.

Firstly a name which enumerates the channel group via a call to the `GetSpadicName()` function is generated and saved in the `SpadicN` variable. This is then combined with the channel number in order to generate the full histogram name and add the new histogram to the histogram manager fHm.

For the signal shapes a 2D histogram will be used to present the cumulative signal shape w.r.t. ADC value and sample number. Binning has been chosen for the histogram to provide the highest amount of detail contained in the signal shape, which is 32 samples of 9 bit integer values from -256 to 255. The bin borders are set off by 0.5 to ensure that an entry will always be filled into the correct bin. The histogram is then constructed and added to fHm. Afterwards, the histogram pointer is retrieved from the histogram manager, with the histogram's name, in order to set the axis titles. The hit types can be presented as a simple 1D histogram with one bin for each of the 4 possible trigger conditions, see Table 2.1, which is constructed and configured analogous to the signal shape histogram.

These histograms can now be used to record the results of the analysis in the `Exec()` function, see listing 3.2. For the analysis, all raw message objects that are available via the fRaw pointer will be accessed. As the entries are stored in a `TClonesArray`, they can be retrieved via the

Listing 3.1: Creating histograms for Signal Shapes and Trigger Types in CreateHistograms() in CbmTrdExampleAnalysis.cxx

```
8   for (Int_t dpb = 0; dpb < fBT->GetNrRobs(); dpb++) {
9     for (Int_t spadic = 0; spadic < fBT->GetNrSpadics() * 2; spadic++) {
10      for (Int_t channel = 0; channel < 16; channel++) {
11        TString spadicN = GetSpadicName(dpb, spadic, "DPB", kHalfSpadic);
12        TString histN = "Self_triggered_SignalShape_" + spadicN +
13                        "_Channel_" + std::to_string(channel);
14
15        fHm->Add(histN.Data(), new TH2I(histN.Data(), histN.Data(), 32,
16                    -0.5,31.5, 512, -256.5, 255.5));
17        fHm->H2(histN.Data())->GetXaxis()->SetTitle("Sample Nr");
18        fHm->H2(histN.Data())->GetYaxis()->SetTitle("ADC Value");
19        histN = "Trigger_type_for_Hitmessages_" + spadicN + "_Channel_" +
20                std::to_string(channel);
21        fHm->Add(histN.Data(),
22                 new TH1D(histN.Data(), histN.Data(), 4, -0.5, 3.5));
23        fHm->H1(histN.Data())->GetXaxis()->SetTitle("Trigger Type");
24      }
25    }
26  }
```

Listing 3.2: Filling histograms for Signal Shapes and Trigger Types in Exec(Option_t*) in CbmTrdExampleAnalysis.cxx

```
79  long NrRawMessages = fRaw->GetEntriesFast();
80  for (int iRaw = 0; iRaw < NrRawMessages; iRaw++) {
81    CbmSpadicRawMessage *currentRaw =
82        static_cast<CbmSpadicRawMessage *>(fRaw->At(iRaw));
83    if (currentRaw->GetHit() != true)
84      continue;
85    // Count only Hit Messages
86    int dpb = fBT->GetRobID(currentRaw);
87    int spadic = fBT->GetSpadicID(currentRaw);
88    int channel = currentRaw->GetChannelID();
89    TString spadicN = GetSpadicName(dpb, spadic, "DPB", kHalfSpadic);
90    TString histN = "Trigger_type_for_Hitmessages_" + spadicN +
91                    "_Channel_" + std::to_string(channel);
92    fHm->H1(histN.Data())->Fill(currentRaw->GetTriggerType());
93    if (currentRaw->GetTriggerType() != 1)
94      continue;
95    histN = "Self_triggered_SignalShape_" + spadicN + "_Channel_" +
96            std::to_string(channel);
97    TH2 *HistPtr = fHm->H2(histN.Data());
98    int NrSamples = currentRaw->GetNrSamples();
99    int *Samples = currentRaw->GetSamples();
100   for (int iSample = 0; iSample < NrSamples; iSample++) {
101     HistPtr->Fill(iSample, Samples[iSample]);
102   }
103 }
```

`At()`-function with their index. Any raw message that is not a *Hit* message is then skipped, as only those contain the data of interest. It is similarly easy to limit analyses to the other message types (*Epoch*, *Info* and *Overflow* being the most relevant) or to use a combination of these types. The histogram name for the trigger type analysis is then constructed via the helper functions and filled with the trigger type after being fetched from the histogram manager For all *Hit* message with trigger type 1 (STR), the corresponding histogram is also fetched and filed with the samples for the signal shape.

Analyses in the framework do not need to be constrained to the raw messages, which is what will be discussed next. The following analyses will be focused on cluster analysis, a hitmap of the distribution of clusters within a detector layer and the PRF will specifically be reconstructed.

As these analyses are performed on `CbmTrdClusters`, these need to be reconstructed first. This firstly requires condensation of the information from the raw messages into `CbmTrdDigis` before running a clusterizer on them. Since the framework provides classes for both of these functions, both need to be run before any analysis class that uses them. The required order can easily be seen in listing A.1, lines 39 to 46.

Cluster Hitmap Evaluating the distribution of clusters in the detector layers is a useful tool in order to detect hotspots and, in particular, to confirm the locations of beam spots and radioactive sources. This can simply be done by querying the 2D position of the cluster's center and filling it into a histogram of appropriate dimensions. Since this is meant to be a simple analysis, a filter for well formed clusters, as defined in Table 3.1, will be applied, without performing any more complicated filtering or reconstruction.

Creating the histograms for each individual layer is very straightforward and can easily be seen in listing A.3, lines 29 to 43 and lines 54 to 60. For consistency with the existing analyses in the framework, hitmaps are called heatmaps in the code. Since this histogram is intended to provide a geometric representation of the distribution of clusters on the padplane, the geometric parameters of the specific layer need to be queried, in particular the number of pad columns/rows and the width and height of a pad. Layers are not automatically rotated to switch columns and rows, this needs to be done separately if the orientation is relevant to the particular analysis being performed.

In listing 3.3, the necessary code to perform this hitmap analysis is printed. The code starts with a performance optimization, since the first layer is always layer 0, all plots w.r.t. layers can easily be stored in an array and be queried using the layer number as an index. To evaluate the efficiency, a variable to store the amount of used clusters is created and initialized to 0. The code then loops over all clusters and filters out any not fitting the definition of a well formed cluster by calling the `ClassifyCluster()` function of the `CbmTrdTestBeamTools` and rejecting any clusters not evaluated to `kNormal`. Afterwards the cluster is counted for the efficiency evaluation and the 2D position of the cluster within a layer is determined, which is

Listing 3.3: Filling a Cluster hitmap in the Exec() function in CbmTrdExampleAnalysis.cxx

```
105   std::vector<TH2 *> Heatmaps;
106   for (Int_t layer = 0; layer < fBT->GetNrLayers(); layer++) {
107     TString histN = "Cluster_Heatmap_Layer_" + std::to_string(layer);
108     Heatmaps.push_back(fHm->H2(histN.Data()));
109   }
110
111   int UsedClusters = 0;
112   uint NrClusters = fClusters->GetEntriesFast();
113   for (uint iCluster = 0; iCluster < NrClusters; ++iCluster) {
114     CbmTrdCluster *CurrentCluster =
115         static_cast<CbmTrdCluster *>(fClusters->At(iCluster));
116     if (CurrentCluster) {
117       if (fBT->ClassifyCluster(CurrentCluster) !=
118           CbmTrdTestBeamTools::CbmTrdClusterClassification::kNormal)
119         continue;
120       UsedClusters++;
121       int layer = fBT->GetLayerID(CurrentCluster);
122       double center = fBT->GetCentralColumnID(CurrentCluster);
123       double displacement = fBT->GetColumnDisplacement(CurrentCluster);
124       double padWidth = fBT->GetPadWidth(layer);
125       double xPos = padWidth * (center + displacement);
126       center = fBT->GetCentralRowID(CurrentCluster);
127       displacement = fBT->GetRowDisplacement(CurrentCluster);
128       double padHeight = fBT->GetPadHeight(layer);
129       double yPos = padHeight * (center + displacement);
130       Heatmaps.at(layer)->Fill(xPos, yPos);
131     }
132   }
133
134   FillHist(fHm->H1("Efficiency Heatmap"), UsedClusters, NrClusters);
135   UsedClusters = 0;
```

done by getting the position on a pad level and then adding the displacement from the pad center. Finally the `FillHist()` helper function is called to fill the efficiency histogram.

Pad-Response-Function The reconstruction of a PRF is a valuable measurement in order to check the performance of the electronic systems. As the PRF is a function of the ROC's geometry, any reconstruction should be similar to the theoretical curve, so deficiencies in the detection and reconstruction chain should show up as artifacts in this particular plot. As the histogram creation is straightforward, we will omit this code here and instead refer to listing A.3, lines 45 to 52 and lines 62 to 67.

The code to reconstruct the PRF is shown in listing 3.4. Like the code in listing 3.3, it iterates over all clusters and only analyses the well formed ones, but in addition, the PRF can also be filtered to only include clusters of a certain width, which in the case of listing 3.4 is all clusters with a width of at least 3 pads. Firstly the center of the cluster along the

Listing 3.4: Filling a PRF histogram in the Exec() function in CbmTrdExampleAnalysis.cxx

```
137   for (uint iCluster = 0; iCluster < NrClusters; ++iCluster) {
138     CbmTrdCluster *CurrentCluster =
139         static_cast<CbmTrdCluster *>(fClusters->At(iCluster));
140     if (CurrentCluster) {
141       if (fBT->ClassifyCluster(CurrentCluster) !=
142           CbmTrdTestBeamTools::CbmTrdClusterClassification::kNormal)
143         continue;
144       if (fBT->GetColumnWidth(CurrentCluster) >= 3) {
145         UsedClusters++;
146         int layer = fBT->GetLayerID(CurrentCluster);
147         double displacement = fBT->GetColumnDisplacement(CurrentCluster);
148         double center = fBT->GetCentralColumnID(CurrentCluster);
149         center += displacement;
150         TString histN = "PRF_Layer_" + std::to_string(layer);
151         TH2 *currentHist = fHm->H2(histN.Data());
152         const std::vector<int> & Digis = CurrentCluster->GetDigis();
153         float Chg = fBT->GetCharge(CurrentCluster) * 1e4;
154         for (uint i = 0; i < Digis.size(); i++) {
155           CbmTrdDigi *Digi = static_cast<CbmTrdDigi *>(
156                                fDigis->At(Digis[i]));
157           float colId = CbmTrdAddress::GetColumnId(fBT->GetAddress(Digi));
158           currentHist->Fill(colId - (center),
159                             (Digi->GetCharge() / Chg) * 100.);
160         }
161       }
162     }
163   }
164
165   FillHist(fHm->H1("Efficiency PRF"), UsedClusters, NrClusters);
```

row is calculated analogously to listing 3.3. As the cluster is a lightweight object, it mostly contains a list of CbmTrdDigis in the cluster. This list is made up of the individual indices of the CbmTrdDigis in the TClonesArray where they are stored. Plotting the PRF requires the cluster charge, which needs to be multiplied by 10^4 due to changes to the CbmTrdDigis implemented since the creation of this framework. Plotting the PRF can now be performed by iterating over all the CbmTrdDigis in the cluster and filling the histogram with the distance from the cluster's center and the charge ratio scaled to a %-value.

All the resulting histograms are stored in the output ROOT-file reported at conclusion of the macro in a folder within this file with the name of the analysis class, in this case CbmTrdExampleAnalysis.

3.3 Results/Performance

In order to demonstrate this setup, the described code will be run on data from the three previously described in-beam-tests.

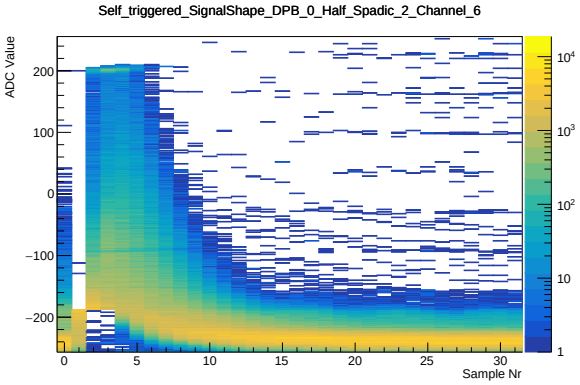
3.3.1 SPS 2016

The chosen TSA-file for the SPS 2016 test beam is from run 188, a 32 minute run recorded on the 10th of October 2016 from 05:53 to 06:25, and is 8 GB in size. A sample histogram from each of the previously described analyses from the example class has been chosen and prepared in Figure 3.5.

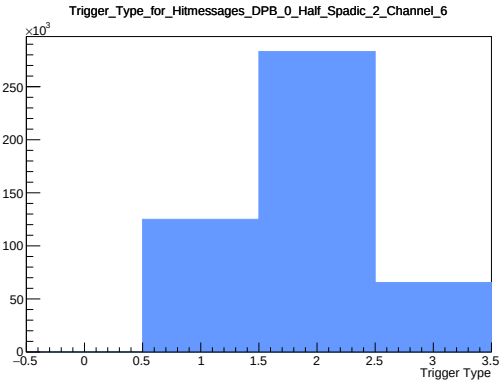
Signal Shape The cumulative signal shape in Figure 3.5a shows the characteristic pulse response of the SPADIC 1.1. The SPADIC 1.1 features a design shaping time of 80 ns and a sampling frequency of 16 MHz in this run. This sampling frequency corresponds to a time between samples of 62.5 ns. The short shaping time compared to the sampling frequency is clearly observable in the sharp pulse between samples 0 to 15. A sharp band of samples at ADC-Values of around +200 implies clipping in the front end electronics, which could potentially distort high charge values, requiring particular reconstruction techniques. The dominant baseline around -240 shows that the signal returns to baseline after the pulse, which indicates that the channel is running as expected. Since this analysis is evaluating self triggered *Hit* messages, the efficiency of this analysis is $e = \frac{125017}{473542}\frac{\text{STR}}{\text{total}} = 26.4\%$, which can be evaluated via Figure 3.5b.

Trigger Types Due to the configuration of the SPADIC 's neighbor trigger logic during the run, a ratio of STR to FNR of about 1:2 is expected. Figure 3.5b, where the distribution of trigger types for the same channel as Figure 3.5a is plotted, is consistent with this expectation. Of note here are the trigger type 3 messages which need to be counted towards both FNR and STR messages for the ratio, as they are part of both groups. Calculating the ratio of STR to FNR hit messages provides a value of $e = \frac{190467}{348525}\frac{\text{STR}}{\text{FNR}} = 54.7\%$, which is close to the expected value. This implies that the channel is neither overly noisy itself nor next to a noisy channel. The other possibility of this channel being both noisy and next to a noisy channel cannot be fully disregarded, but the low amount of hit messages with the combined trigger type 3 suggests that this hypothesis is unlikely.

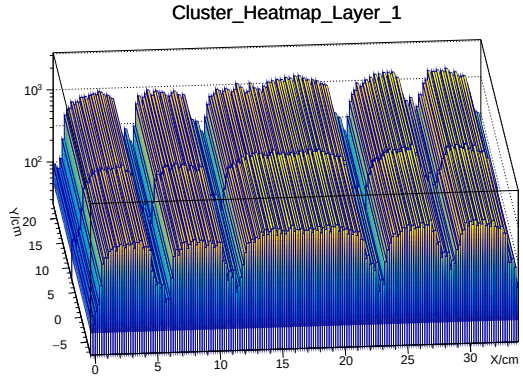
Cluster Hitmap The cluster hitmap in Figure 3.5c shows five distinct groups of active channels, while the expectation would be three: one for each SPADIC FEB. The gap between the two groups on the left and the two groups on the right corresponds to where the expected boundary between the two channel groups lies. Since the center group does not feature this gap, it could imply a configuration error of these particular FEBs similar to the one described in [Mun16, pp. 22, 23], or a timestamp reconstruction error due to missed epoch messages on one of the



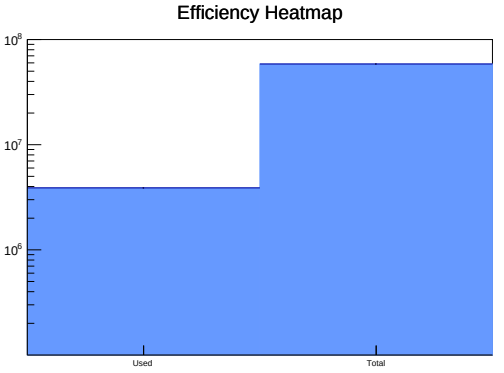
(a) Cumulative Signal Shape of all self triggered hits for a single channel, displayed are the individual ADC values against the corresponding sample number.



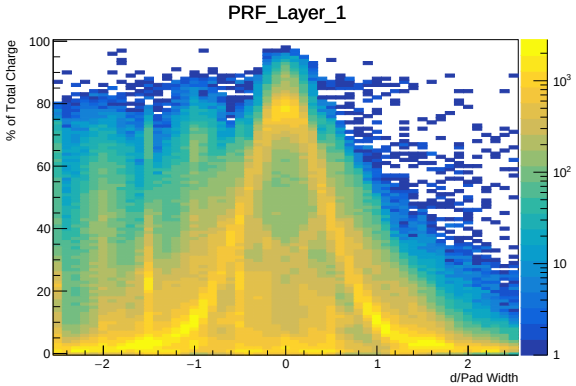
(b) Trigger Type distribution for the channel in Figure 3.5a



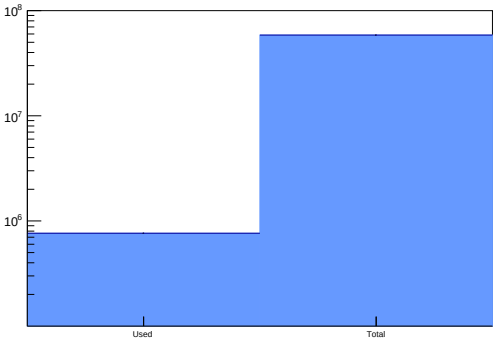
(c) 2D Cluster Hitmap on a single layer of the experimental setup, only the active detector area is displayed here



(d) Efficiency of clusters used for hitmaps in the same run as Figure 3.5c



(e) Plot of the reconstructed PRF on a single layer, displayed are the charge percentages of the cluster charge contained in a Digi against the distance of this Digi from the cluster's center



(f) Efficiency of clusters used for PRF in the same run as Figure 3.5e

Figure 3.5: Set of histograms for an analysis run on SPS 2016 Data. The chosen TSA-file is from a 32 minute run recorded on the 10th of October 2016 from 05:53 to 06:25 and is 8 GB in size.

channel groups. Confirming either of these hypotheses would require an in-depth analysis on the raw message stream. Besides these unexpected gaps the hitmap is flat without spikes, which implies that the padplane of this particular layer is illuminated evenly and that there are no overly noisy channels. However, examining the efficiency via Figure 3.5d shows that only a small fraction of clusters were deemed suitable for inclusion in this analysis, specifically 6.6%. Since the only criterion for inclusion was being well formed according to the criteria in Table 3.1, this means that the vast majority of found clusters are unsuitable or malformed. This would be a suitable starting point for a more advanced analysis into the particularities of a SPADIC 1.x based system, but this is beyond the scope of this thesis.

Pad Response Function Lastly, the PRF in Figure 3.5c shows the characteristic curve according to Mathieson with heavy artifacts. In particular, two distinct categories of artifacts can be observed, these are firstly clusters where the central pad does not contain the highest charge fraction and secondly clusters in which the individual pads seem to be locked at discrete displacements which are a positive multiple of $d = x \times (-0.5)$. Since SPADIC 1.1 data have not been analyzed in depth yet in published works, all hypotheses stated here should be treated as such.

A potential hypothesis for the first category may be an abundance of messages in which the signal of the channel with the highest charge fraction did not fall below the first trigger threshhold and therefore did not trigger a *Hit* message. This could probably result from a retrigger during the recording of a message, specifically if one of the originally FNR triggered channels fulfills the trigger condition after the main channel. As the SPADIC 1.1 doesn't feature a dead time before a retrigger can occur [Arm13], this can likely occur on clusters off-center from the central pad.

The second category is only observed on this SPADIC 1.1 dataset and neither in Figures 3.6 and 3.7 nor in Spadic 1.0 data, see figure 5.12 from [Mun16, p. 31]. It seems to be a phenomenon specific to this SPADIC revision or this experimental setup. The discrete nature of these peaks along the displacement axis combined with the wide spread along the charge fraction axis makes this phenomenon particularly difficult to explain. Further research into SPADIC 1.1 datasets would be required to form a consistent hypothesis.

Examining the efficiency via Figure 3.5f shows that only a small fraction of clusters were deemed suitable for inclusion in this analysis, specifically 1.3%, which is only a fourth of the efficiency of the heatmap Figure 3.5d. Since the only criterion for inclusion in addition to being well formed according to the criteria in Table 3.1 was a width of at least 3 columns, the available data suggests that the majority of „well formed“ clusters seems to be constructed of three Digis on two pads. This could explain the second category of artifacts, but the cause of this phenomenon still needs to be investigated. A hypothesis for the cause could be that a message is emitted twice on the same channel with the same timestamp, but without further research this cannot be confirmed.

3.3.2 DESY 2017

For the DESY 2017 test-beam a 2 GB TSA-file from run 122 , recorded on the 13th of September 2017 over a period of three hours and 20 minutes, has been selected. This TSA-file was chosen at random from the 213 TSA-files, of up to 2 GB in size, comprising the full run data and is not meant to provide a comprehensive portion of the full data. A sample histogram from each of the previously described analyses from the example class has been prepared in Figure 3.6.

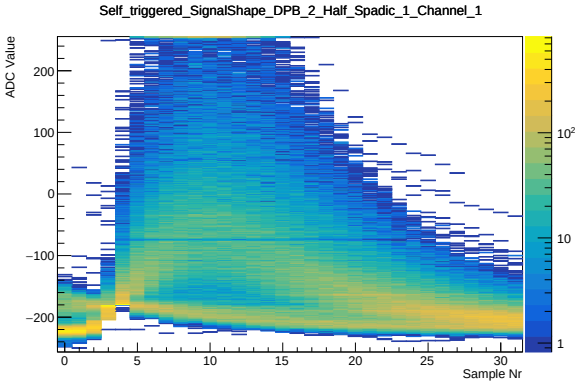
Signal Shape The histogram of the cumulative pulse shape in Figure 3.6a shows two distinct signal components. The more prominent one is the narrow band of samples located around ADC=-200 at sample 7 for example. This set of signals has been observed in previously published works, see [Mey19] for example, and it is caused by a particularity of the SPADIC 2.0 trigger when operated with 2 different absolute threshholds (Thr1 and Thr2). If a channel records the following sequence of samples s_i:

$$s_0 \geq \text{Thr2} \wedge \text{Thr1} < s_1 < \text{Thr2} \wedge s_2 > \text{Thr2},$$

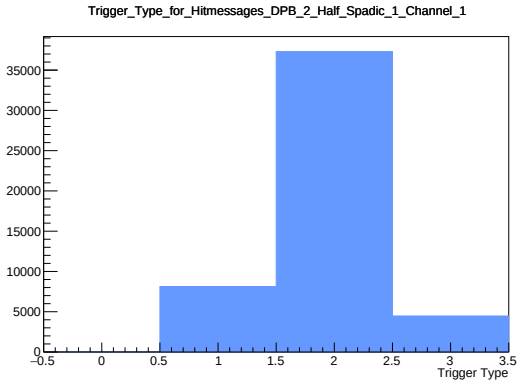
which can happen during a return to baseline, due to the long shaping time of $\tau_s \approx 240\,\text{ns}$ compared to the sampling period of 62.5 ns, an STR hit message is triggered. This behaviour has been changed for future SPADIC versions beginning with the SPADIC 2.1.

A second less prominent band of signals is observable above this and it follows the typical signal shape for a SPADIC 2.0, peaking around sample number 10. As only low amounts of clipping are observed near to 250 ADC-values, it can be inferred that the signal is recorded with little distortion. Near sample 32, the signal is returning to the baseline around -200 ADC-Values, the system seems to be operating in a stable hit frequency regime. Since this analysis is evaluating self triggered *Hit* messages, the efficiency of this analysis is $e = \frac{8132}{49904}\frac{\text{STR}}{\text{total}} = 16.3\%$, which can be evaluated via Figure 3.5b.

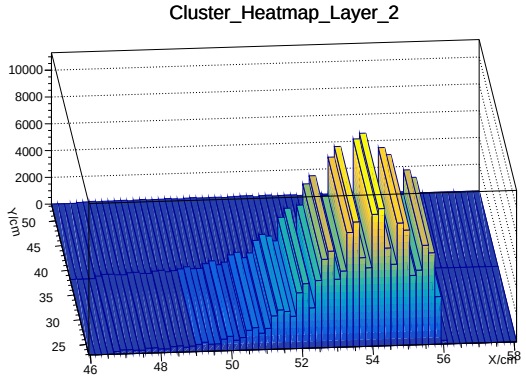
Trigger Types Due to the configuration of the SPADIC 's neighbor trigger system during the run, a ratio of STR to FNR of about 1:2, or 50%, is expected, when three adjacent channels trigger with the same frequency, like on an evenly illuminated padplane. Figure 3.6b, in which the distribution of trigger types for the same channel as Figure 3.6a is plotted, is not consistent with this expectation, since the amount of FNR messages vastly outstrips STR messages. Examining the fraction of STR compared to FNR hit messages on this channel provides a ratio of $e = \frac{12604}{49904}\frac{\text{STR}}{\text{FNR}} = 25.3\%$. This can be explained with either a noisy neighbor channel or the beamspot on the padplane, where the examined channel is not located in the center of this region. Examining Figure 3.6c, a smooth hotspot is visible around X=54 cm. The examined channel is located at X=52 cm, which is on the side of the beamspot. This suggests that the excess of FNR messages on this particular channel is not caused by a noisy neighbor channel, but rather by the intensity profile falling off at the edges of the beamspot.



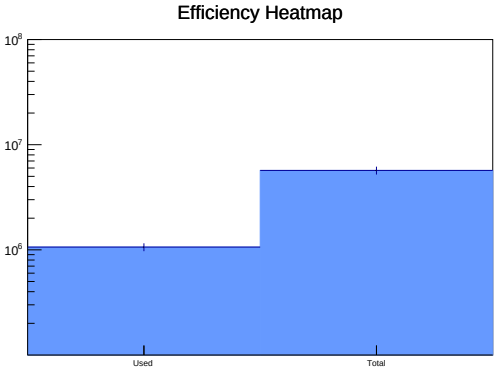
(a) Cumulative Signal Shape of all self triggered hits for a single channel, displayed are the individual ADC values against the corresponding sample number.



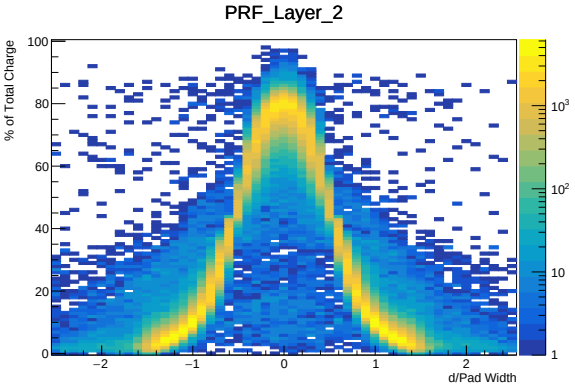
(b) Trigger Type distribution for the channel in Figure 3.6a



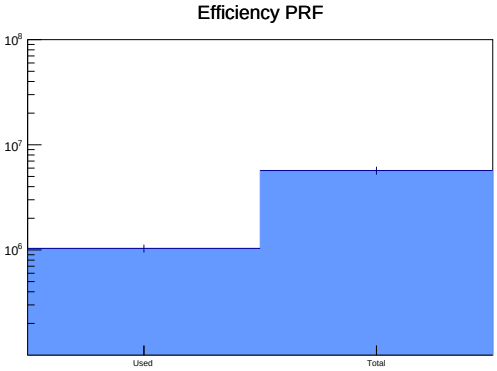
(c) 2D Cluster Hitmap on a single layer of the experimental setup, the histogram is cropped to the active detector area.



(d) Efficiency histogram of all clusters used for hitmaps on all layers in the same run as Figure 3.6c



(e) Plot of the reconstructed PRF on a single layer, displayed are the charge percentages of the cluster charge contained in a Digi against the distance of this Digi from the cluster's center.



(f) Efficiency of clusters used for PRF in the same run as Figure 3.6e

Figure 3.6: Set of histograms for an analysis run on DESY 2017 Data. The chosen TSA-file is from a run recorded on the 13th of September 2017 over a period of three hours and 20 minutes and is 2 GB in size.

Cluster Hitmap The electron beam for the DESY setup is expected to be relatively narrow in the plane of the detector, due to the collimator placed at the entrance of the cave. Examining Figure 3.6c shows a narrow hotspot in the front row of pads. Since the histogram is binned in such a way that 4 bins cover each of the active pads, the substructure visible in the beamspot can be identified to be identical to the pad structure. This implies that there is a bias towards the pad center either during the reconstruction or during the data taking. Since the PRF in Figure 3.6e shows no obvious bias towards any pad center, the occurrence during data taking is deemed more likely.

A hypothesis for how this bias could emerge could be that lower energy clusters can trigger a readout if they hit the center of the pad, but not if the hit is towards the edges of the pads. This could easily be studied by adding a filter for the cluster energy and observing if the substructure diminishes at higher energies.

Compared to the SPS 2016 beamtime, the efficiency of the hitmap analysis is greatly improved, with an efficiency of 18.7%, see Figure 3.6d. This implies that the changes made between the SPADIC 1.1 and 2.0 have greatly improved the fraction of usable data, which should also result in an improvement of the efficiency of the PRF.

Pad Response Function Examining the PRF in Figure 3.6e shows a vast improvement in data quality. Even with the exact same preprocessing as in Figure 3.5e, most of the artifacts observed there are not seen here and the vast majority of clusters are reconstructed close to the line expected according to Mathieson. A small amount of clusters is observed where the central pad has less than the majority of the charge, but these clusters can be filtered out fairly straightforwardly. The improved data quality shows most prominently when the efficiency of this PRF reconstruction is measured, see Figure 3.6f. Compared to Figure 3.5f, the amount of used clusters is vastly improved, from 1.3% in the SPS data to 18.2% on this DESY 2017 dataset. This improvement suggests that the hypothesis that the fraction of usable data has improved from the SPS 2016 dataset is likely true. The reconstructed PRF in Figure 3.6e could already be used without further post-processing to repeat the previous analysis and determination of the K_3 parameter from [Mun16].

3.3.3 GIF 2017

For the GIF++ 2017 test beam, a 2 GB TSA-file from run 32, recorded on the 10th of October 2017 over a five minute and 20 second period, has been selected. This TSA-file was selected from the middle of the run to avoid startup problems and the potential issues from a premature shutdown of the ^{137}Cs-source. The ^{137}Cs-source was operated with an attenuator setting of A3B1C2, meaning that the event rate in the detector was reduced by a factor of roughly 220 from the full rate for this run [Pfe+16, p. 7]. Since the GIF++ source has an activity of 13.9 TBq, this reduced flux would still lead to a very high load in the detector. The specific load in the detector cannot be calculated without a full simulation of the setup, due to the complex interactions of the radiation with the surrounding material and within the detector volume. For this specific run only a subset of 20 samples from a maximum of 32 in a hit message was recorded.

Signal Shape Inspecting the cumulative signal shape in Figure 3.7a shows a few noteworthy features. Firstly, there is a band of samples around -160 ADC-values, which is right around the two identical threshold values of -160. These could occur due to the high constant load placed on the detector by the ^{137}Cs-source, but this hypothesis would need further investigation.

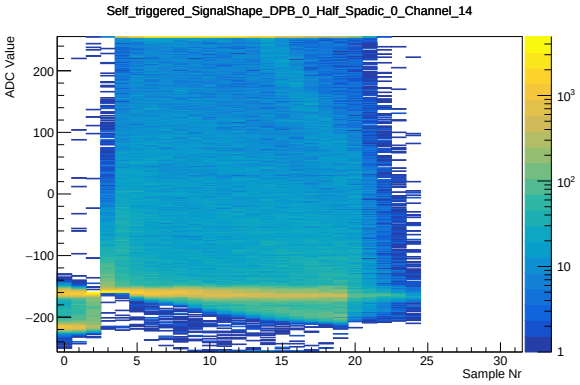
Secondly, there is significant clipping at +255 ADC-values. This is probably the result of a misconfiguration of the detector's gas gain, possibly due to an overly high voltage being applied to the electrodes.

Furthermore, there are some entries at sample numbers of 20 to 24, despite only 20 samples being recorded. This is the result of the signal reconstruction algorithm developed by J. Beckhoff in his masters thesis [Bec18], which is used in the SPADIC 2.0 unpacker to rejoin a class of signal fragmentation.

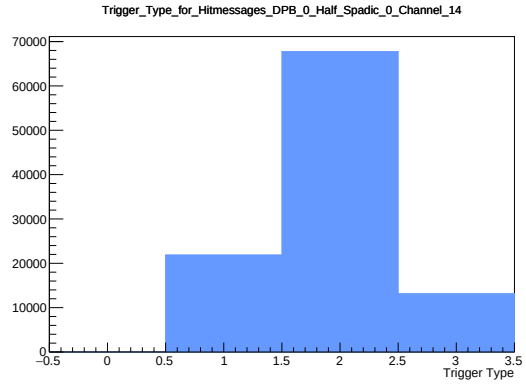
The typical signal shape is not easily visible, probably due to the clipping and the band of samples around -160.

According to the criteria used previously, the efficiency of this analysis is $e = \frac{21893}{102785}\frac{\text{STR}}{\text{total}} = 21.3\%$, which can be evaluated via Figure 3.7b.

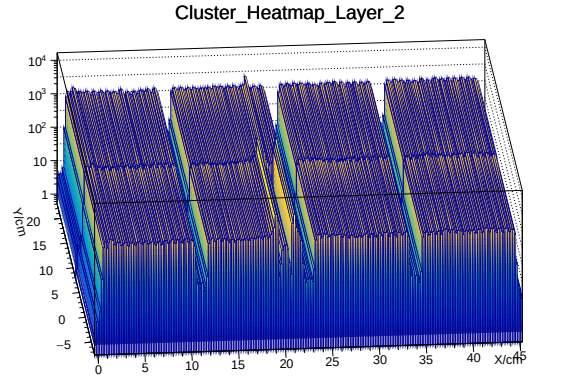
Trigger Types Analogously to the other datasets, a ratio of STR to FNR of 1:2 would be expected, but Figure 3.7b, where the distribution of Trigger Types for the same channel as Figure 3.7a is plotted, is not consistent with this expectation. The amount of FNR messages exceeds the expected amount with a ratio of $e = \frac{35042}{80892}\frac{\text{STR}}{\text{FNR}} = 43.3\%$ slightly, which suggests that one or both of the neighboring pads is a little more likely to trigger on a hit. Examining Figure 3.7c shows no obvious hotspot of clusters in the leftmost block of pads, which is where this particular channel is connected.



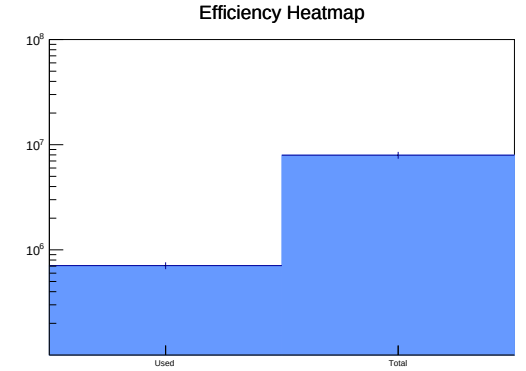
(a) Cumulative Signal Shape of all self triggered hits for a single channel, displayed are the individual ADC values against (?) the corresponding sample number



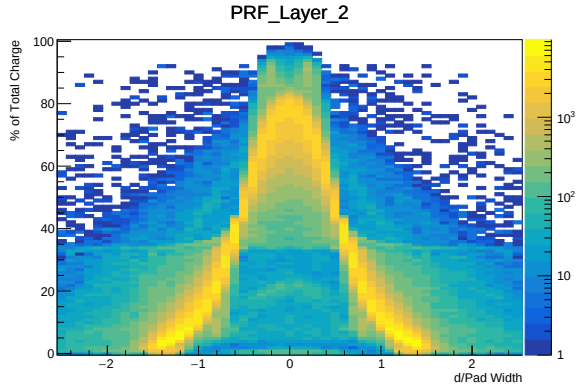
(b) Trigger Type distribution for the channel in Figure 3.7a



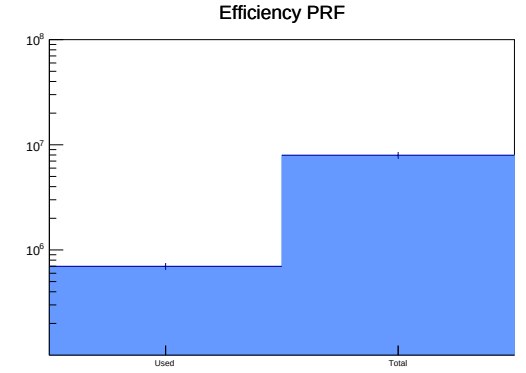
(c) 2D Cluster Hitmap on a single layer of the experimental setup, only the active detector area is displayed here



(d) Efficiency of clusters used for hitmaps in the same run as Figure 3.7c



(e) Plot of the reconstructed PRF on a single layer, displayed are the charge percentages of the cluster charge contained in a Digi against the distance of this Digi from the cluster's center.



(f) Efficiency of clusters used for PRF in the same run as Figure 3.7e

Figure 3.7: Set of histograms for an analysis run on GIF++ 2017 Data. The chosen TSA-file is from run 32 recorded on the 10th of October 2017 over a period of five minute and 20 second and is 2 GB in size.

Cluster Hitmap In Figure 3.7c, the Cluster hitmap for this particular dataset is shown. Four distinct blocks are visible here, corresponding to the four SPADICs connected to this detector, with the characteristic dips at the edges of the active area between FEBs. Within these blocks, the distribution of clusters is mostly flat, with the exception of the center left block in which two sharp peaks are visible in the front row. These spikes are likely the result of one or more noisy channels and should be excluded from in-depth analysis.

Examining the efficiency via Figure 3.7d shows that only a small fraction of clusters were deemed suitable for inclusion in this analysis, specifically 8.9%. This is worse than the DESY results of 18.7% in Figure 3.6d, but slightly better than the SPS results of 6.6% from Figure 3.5d. It is likely the result of the high load placed on the detector, which would also stress the Front-End-Electronic and could potentially cause issues.

Pad Response Function The reconstructed PRF for this particular run is displayed in Figure 3.7e. In comparison to the PRF for the DESY run in Figure 3.6e, this histogram contains more unexpected entries. In particular, there is a larger fraction of clusters with the central Digi not carrying the most charge and correspondingly a higher amount of clusters with most of the charge on one of the outer pads. Beside these broader areas in the histogram, there are four smaller peaks at (x,y) coordinates of $(\pm 0.8, \approx 5\%)$ and $(\pm 0.2, \approx 90\%)$, which might be the result of samples being lost due to the high load placed on the SPADIC .

The efficiency of this analysis can be calculated to 8.8% from Figure 3.7f and is very close to the efficiency of the heatmap at 8.9%. Compared to the efficiencies in the DESY dataset at 18.2% (PRF) and 18.7% (heatmap), this is slightly improved at first glance. However since the GIF++ PRF includes more artifacts, as was described in the previous paragraph, the efficiency of this analysis after these clusters have been filtered out may turn out to be lower.

4 Capacitor Irradiation Testing

4.1 Considerations

The basis for a successful experiment is reliable hardware, even in small components. As spatial and spectral resolutions depend heavily upon a stable high voltage potential, the MWPCs are planned to be equipped with filtering capacitors, see [BBE18]. The planned type of capacitors was chosen to be ceramic HV 2n2 M 3kV E capacitors, a capacitor type already in use in the A Large Ion Collider Experiment (ALICE) TRD ROCs, due to it being readily available and cost effective.

During LHC Run 1, several modules showed unexpectedly high leakage currents under applied HV, which would eventually be traced to the filtering capacitors. While the HV filter boards were modified for all Super-Modules newly installed during Long Shutdown (LS)1, the capacitors already installed continued to fail during Run 2. During LS2, nearly all of these capacitors were removed from the extracted SMs in order to resolve leakage issues.

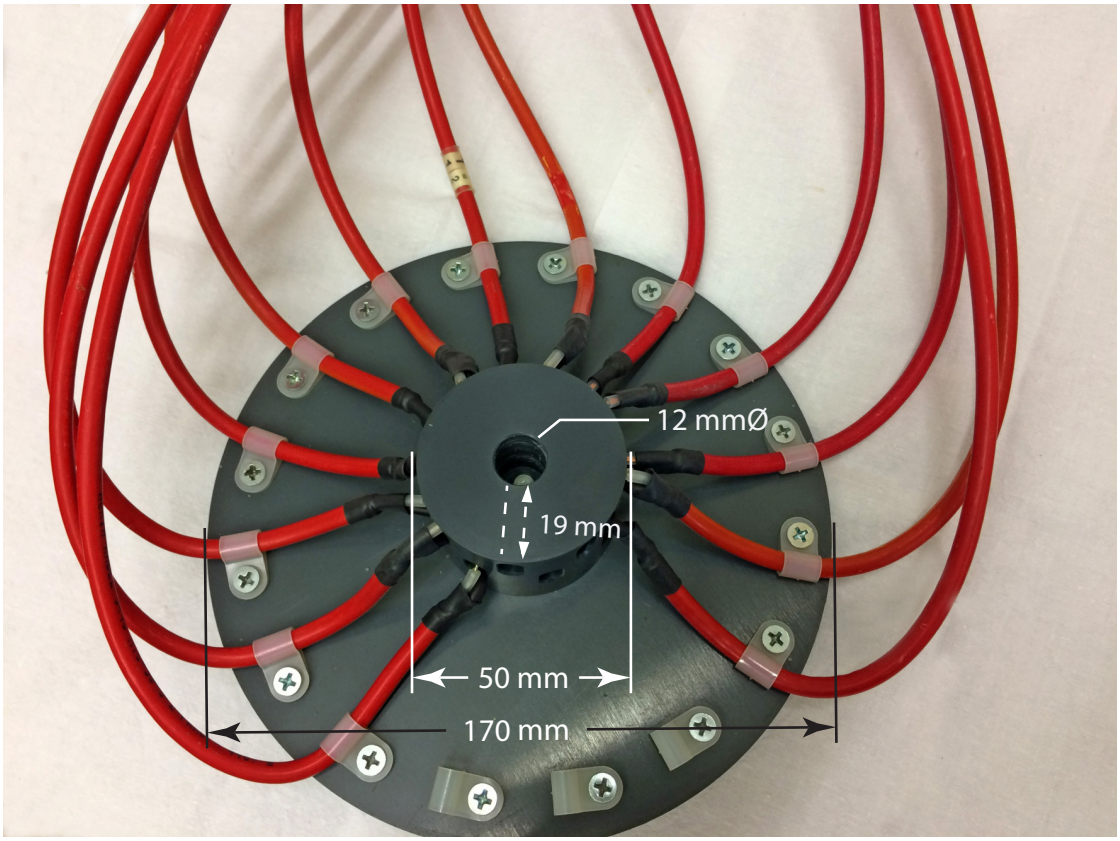
As the CBM-TRD intends to use these capacitors for HV filtering in a significantly more intense radiation environment, the radiation dependence of the failure rate was decided to be newly investigated during the HV filter board design process for the CBM-TRD.

4.2 Experimental Setup

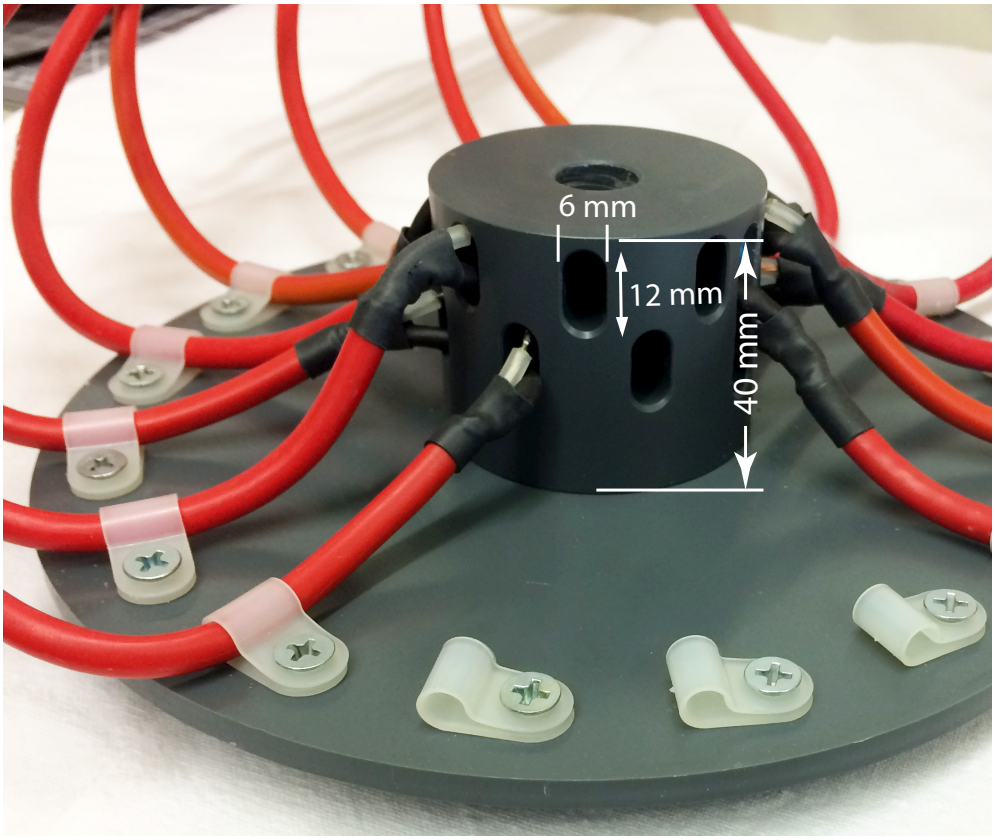
To assess the failure rate due to radiation damage in a short amount of time, an intense radiation source was needed. A Radium-Beryllium source was selected for this purpose, as significant amounts of neutron flux are expected under the experimental conditions of CBM [BBE18].

In order to provide a consistent dosage for the capacitors, a capacitor holder was produced by the mechanical workshop at IKP Münster, see Figure 4.1, to fix the position of the capacitors relative to the neutron source. This holder can accommodate up to 14 ceramic HV disk capacitors of type 2n2 M 3kV E and an additional KEMET PHE450/F450 foil capacitor at the same time.

The failure point of the individual capacitors was determined via monitoring the leakage current of all capacitors. A capacitor would have failed this test if its HV current at 2 kV surpassed 50 nA for more than 1 hour during the test, or if it tripped the over-current protection. High voltage of 2 kV was therefore applied to all capacitors via an ISEG HV EDS F 025n power supply for the total duration of the test. In order to identify false positives due to ambient



(a) Topside view of capacitor holder, with dimensions.



(b) Sideways view of capacitor holder, with dimensions.

Figure 4.1: Photographs showing the dimensions of the capacitor holder for the neutron irradiation study. The capacitors are attached to HV cables and fixed in place via brackets. Eleven capacitors are ceramic HV 2n2 M 3kV E capacitors, one capacitor is a KEMET PHE450/F450 foil capacitor. (Pictures: Norbert Heine)

(a) Front view of the bunker constructed for the irradiation assembly. The red HV cables are visible at the entry point into the bunker.

(b) Backside view of the bunker constructed for the irradiation assembly. The ambient condition sensor is circled in blue.

Figure 4.2: Views of the radiation protection bunker constructed for the irradiation assembly. The bunker is constructed from powdered paraffin, enriched radiation protection concrete and lead plating on top.

conditions like high humidity and thereby increased leakage on all capacitors, humidity and temperature were logged for the duration of the test. The ambient condition sensor was placed close to the irradiation assembly, see Figure 4.2b, while still being shielded from the radiation.

The irradiation assembly, consisting of the capacitor holder, with the capacitors and the neutron source attached, was placed in a newly constructed radiation protection bunker in the institute's isotope lab, see Figure 4.2. The radiation safety of this bunker was evaluated by measuring the neutron dose rate in publicly accessible locations on the closest five sides of the room. It was found to be indistinguishable from background radiation.

After the neutron source was introduced to the irradiation assembly, the high voltage was turned on and the test started on the 14th of May 2019 at 10:38:28 and run continuously until the 12th of July 2019 at 09:18:37, for a total of 58 days, 22 hours, 40 minutes and 9 seconds or 1414.67 h.

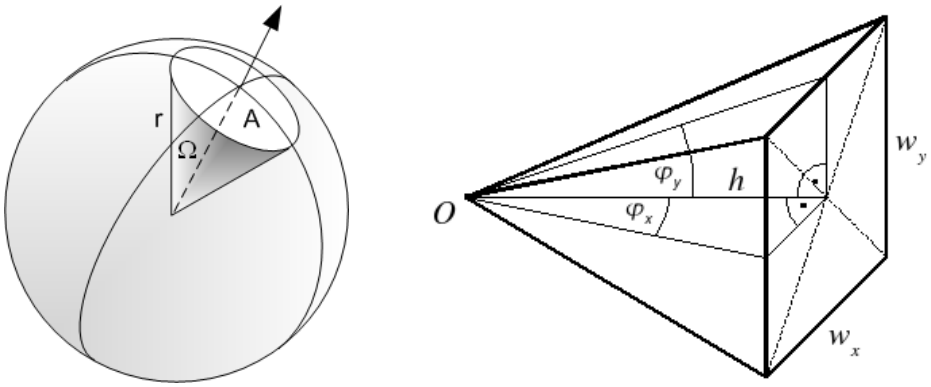
4.3 Dosage Calculations

As no thermoluminescent dosimeter was employed during the test, dosage calculations had to be performed using the dose rate and the duration of the experiment. A Berthold LB 123 UMo dosimeter with an LB 6411 - Neutron Probe was used to determine the dose rate. As its calibration factor in the spectrum of a RaBe source is close to 1, see [Bur+97], the measured dose rate could be used without further conversion. This neutron probe employs a proton recoil counting tube inside a polyethylene moderator and is calibrated to a dose rate of 1.27 $\frac{\mu Sv/h}{cps}$ (counts-per-second), according to the public data sheet [Tec]. As the detector is counting particles from a radioactive source and a calibration factor from counting rate r to the dose is provided, the error can be estimated to be $\Delta r = \sqrt{r}$. Since no averaging time for the counting rate is provided by the manufacturer in public documentation, the averaging time is estimated to be around 10 s for the purpose of this thesis.

The dosage rate was measured to 40 µSv/h at a distance of 10 cm±2.5 cm detector-edge-to-source, with no provided error. This results in a counting rate of $r = 315\frac{1}{10s}$ and an estimated counting rate error of $\Delta r = \sqrt{315}\frac{1}{10s} = 17.75\frac{1}{10s}$, resulting in a dosage of (40±2.25) µSv/h. As the neutron probe is calibrated for low energy neutrons and the RaBe source also produces a spectrum of these, the biological weighing factor of 10 can simply be divided out, resulting in a dosage rate of (4±0.225) µGy/h.

As the source is not embedded within the detector, it only captures a part of the total radiation dose corresponding to the fraction of the solid angle it covers. The solid angle this detector covers with regards to the neutron source is modeled as a cone with the base located at the center of the detector, see Figure 4.3a, at a distance of $h = (350 \pm 25)$ mm and a diameter of the detector's active volume of $d = 250$ mm. The solid angle covered by the detector can be calculated via:

$$\Omega = 2\pi \left(1 - \cos\frac{\omega}{2}\right) \tag{4.1}$$



(a) Measurements for the solid angle of a cone. [Com07]

(b) Measurements for the solid angle of a pyramid. [Com08]

Figure 4.3: Schematic view of the parameters used for solid angle calculation

with ω being the cone angle. Using equation (4.1), the solid angle covered by the detector can be calculated to:

h/mm	Δh/mm	d/mm	ω	$\Delta\omega$	Ω/sr	$\Delta\Omega$/sr
350	25	250	0.686	0.045	0.366	0.048

The detector therefore covers (2.91±0.4)% of the full solid angle and the isotropic radiation field and therefore the RaBe source produces a dose of (137±23)µGy/h over the full solid angle.

Due to their rectangular cross-section, both capacitor types can be modeled as pyramids in order to calculate the absorbed radiation. The covered solid angle by the capacitors can be calculated via the Osterom-Strackee Formula, see [VS83]:

$$\Omega = 4 \arctan \frac{w_x \cdot w_y}{2h \cdot \sqrt{4h^2 + w_x^2 + w_y^2}} \tag{4.2}$$

The definition of the variables can be seen in Figure 4.3b.

The cylindrical ceramic 2n2 M 3kV E capacitors have a diameter of (11 ± 0.2) mm and are placed as close to the source as possible, meaning they are placed directly at the inner wall of their respective slot in the holder. The inner cylinder for the source has a radius of (6 ± 0.1) mm and the wall between the capacitor slot and the source slot has a thickness of (2.0 ± 0.1) mm. The cuboid foil capacitor is likewise placed at the inner wall of its slot in the holder. Since both capacitors have rectangular cross-sections, they are modeled as pyramids for the purpose of determining the covered solid angle. The base of these pyramids is the cross-section in the center of the capacitor, for the ceramic disk capacitors this is at a distance of 1/2 of the diameter $w_y = 11$mm from the inner wall, while for the foil capacitor it is set at 1/2 of its height of $t = 12.5$mm.

According to (4.2), the ceramic and foil capacitors therefore cover solid angles relative to the source of:

Table 4.1: Dimensions and solid angles covered by the capacitors

Type	h/mm	Δh/mm	w_x/mm	Δw_x/mm	w_y/mm	Δw_y/mm	Ω/sr	ΔΩ/sr
Ceramic	13.5	0.2	3.5	0.5	11	0.2	0.194	0.001
Foil	14.3	0.2	18	0.2	6.5	0.5	0.476	0.002

As an isotropic radiation field is assumed, the absorbed radiation can be calculated to be proportional to the fraction of the covered solid angle. The full solid angle of 4π comprises of a dose field of (137±23)µGy/h, resulting in the calculated dosage rates Dr_i in Table 4.2.

Table 4.2: Calculation of the dosage

Type	Ω/sr	ΔΩ/sr	$\frac{\Omega}{4\pi}$/%	$\frac{\Delta\Omega}{4\pi}$/%	Dr_i/µGy/h	Δ Dr_i/µGy/h
Ceramic	0.194	0.001	1.54	0.01	2.1	0.4
Foil	0.476	0.002	3.79	0.02	5.2	0.9

Multiplying the dosage rates with the test duration of 1414.67 h results in total dosages for the ceramic capacitors of $D_{\mathrm{ceramic}} = (3.0\pm0.5)$mGy and a total dosage of $D_{\mathrm{foil}} = (7.4\pm1.2)$mGy for the foil capacitor.

4.4 Equivalent Lifetime Calculation

With the calculated total dose experienced by the capacitors, an estimation can be made for the equivalent lifetime in the full SIS 100 CBM-TRD setup. As studies of the radiation environment in the CBM TRD have been completed, this thesis will take area dosage rates from the TDR, see [BBE18, p. 52], and estimate the equivalent lifetime of the capacitors in this environment. By examining Figure 4.4, we can estimate the area dose for the central part of the detector to be around 20Gy/m²/(2 months) and below 10Gy/m²/(2 months) in the largest part of the detector. The dosage experienced by the capacitors therefore needs to be divided by the cross-section $A_i = w_{x,i} \cdot w_{y,i}$ of the capacitor, resulting in the equivalent area dose Da_i in Table 4.3.

Table 4.3: Equivalent Area Dose for both capacitors

Type	A_i/mm²	ΔA_i/mm²	D_i/mGy	ΔD_i/mGy	Da_i/Gy/m²	ΔDa_i/Gy/m²
Ceramic	38.5	5.5	3.0	0.5	78	17
Foil	117	9	7.4	1.2	63	12

The equivalent area doses experienced by the capacitors can be divided by the simulated area dose rates from [BBE18] in Figure 4.4, which results in estimates of the expected lifetime

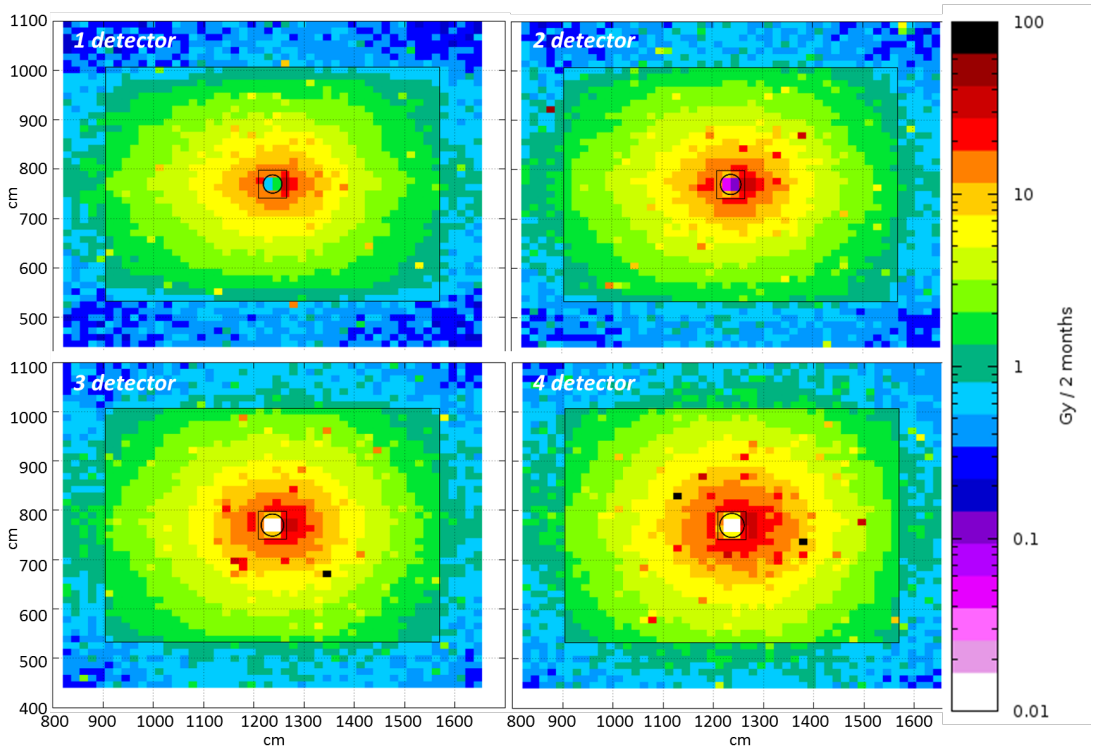


Figure 4.4: The total ionizing radiation dose per square meter expected in the four CBM TRD layers. This plot is taken from [BBE18, p. 52]. The calculation was performed with FLUKA [Bat+07] and corresponds to two months of Au + Au collisions at 10 AGeV with an interaction rate of 5 MHz.

Table 4.4: Estimated lifetime of capacitor types

Type	$\frac{t_{\mathrm{i,Life,central}}}{\mathrm{months}}$	$\frac{\Delta t_{\mathrm{i,Life,central}}}{\mathrm{months}}$	$\frac{t_{\mathrm{i,Life,peripheral}}}{\mathrm{months}}$	$\frac{\Delta t_{\mathrm{i,Life,peripheral}}}{\mathrm{months}}$
Ceramic	7.8	1.7	15.6	3.4
Foil	6.3	1.2	12.6	2.3

under continuous beam in the SIS100 setup. These estimates have been tabulated in Table 4.4 for reference.

For the ceramic disk capacitors, this results in an equivalent lifetime of (8 ± 2) months in the central part of the detector and (16 ± 3) months in the larger part of the detector.

For the foil capacitor, this results in an equivalent lifetime of (6 ± 1) months in the central part of the detector and (13 ± 2) months in the larger part of the detector.

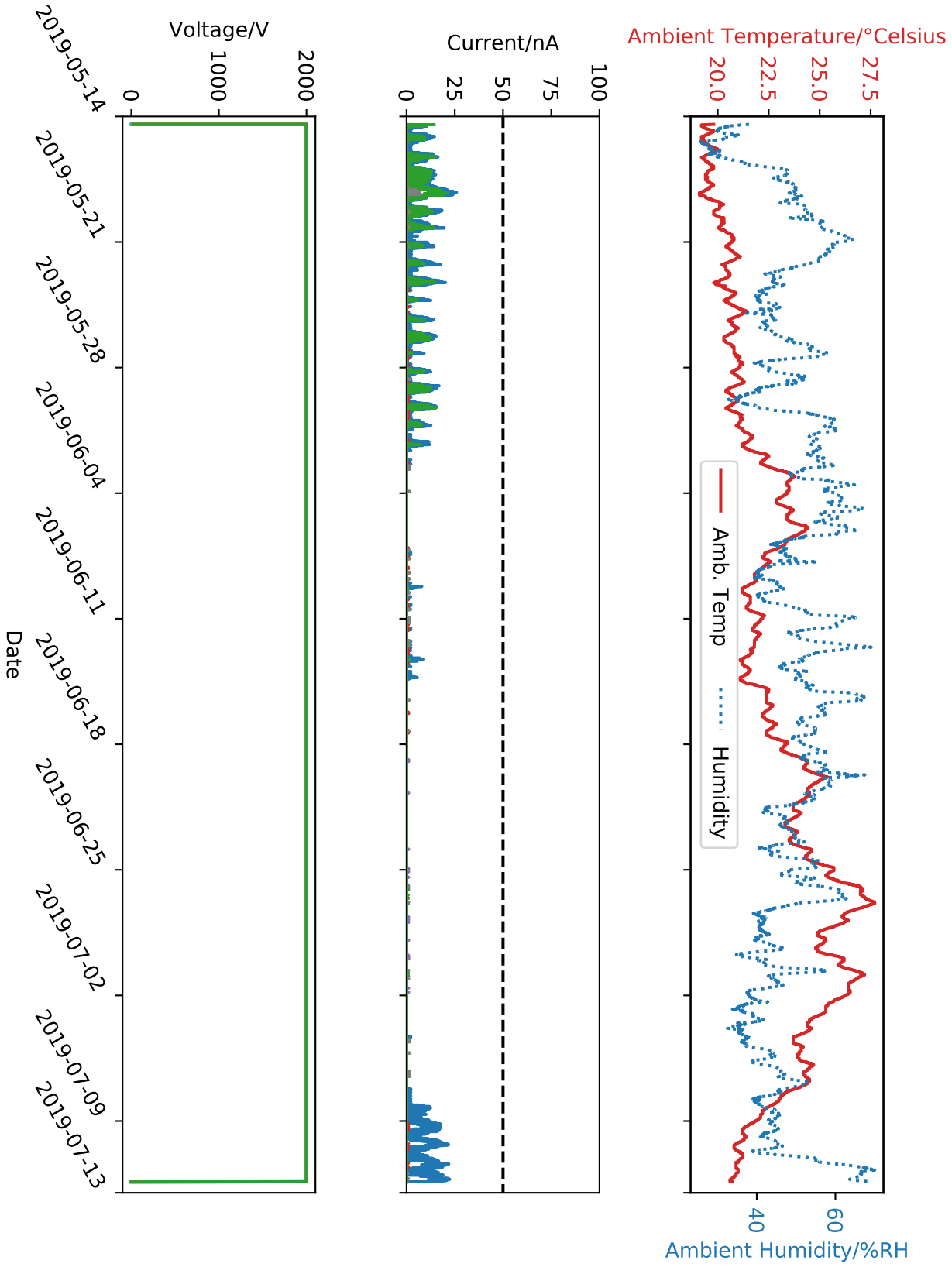


Figure 4.5: Plot of ambient conditions and HV data for all 12 capacitors for the duration of the test. The test period begins 2019-05-14 10:38:28 and ends 2019-07-12 09:18:37, for a total duration of 1414.67 h. The dashed line in the current diagram displays the failure point. No legend for current and high voltage is provided, as none of the 12 tested capacitors failed the test.

4.5 Discussion

The equivalent lifetime figures for the capacitors need to be combined with the amount of tested capacitors in order to calculate the expected total fraction of capacitors failing the tested radiation load. We will be estimating this fraction via the binomial distribution (4.3).

$$f(k,n,p) = \binom{n}{k} p^k (1-p)^{n-k} \overset{k=0}{\Longrightarrow} (1-p)^n \tag{4.3}$$

$$\Rightarrow \alpha = (1-p)^n \overset{\ln}{\Rightarrow} \ln(\alpha) = n \cdot \ln(1-p) \Leftrightarrow \frac{\ln(\alpha)}{n} = \ln(1-p)$$

$$\overset{\exp}{\Longrightarrow} p = 1 - \exp\left(\frac{\ln(\alpha)}{n}\right) \tag{4.4}$$

An examination of Figure 4.5 for any capacitors that failed the test shows no failures within this sample of $n_{\text{ceramic}} = 11$ and $n_{\text{foil}} = 1$. Calculating the reliability can therefore be done by taking (4.4) and inserting the expected parameters. To estimate the expected maximum failure probability with a confidence interval of 95%, $\alpha = 1 - 95\% = 0.05$ and n equal to the sample size are used.

A sample size of $n = 11$ therefore results in an maximum failure probability of $p < 23.8\%$ for the ceramic capacitors, meaning that with a confidence of 95% more than 76.2% of the ceramic capacitors should last longer than (8 ± 2) months of continuous beam in the central region of the detector and longer than (16 ± 3) months in the larger part of the detector.

Redoing this calculation for the foil capacitor with a sample size of $n = 1$ contrastingly results in a maximum failure probability of $p < 95\%$ for the ceramic capacitors, meaning that with a confidence of 95%, more than 5% of the foil capacitors should last longer than (6 ± 1) months of continuous beam in the central region of the detector and longer than (13 ± 2) months in the larger part of the detector.

The reliability of the ceramic capacitors with regard to radiation damage seems sufficient for the SIS100 CBM-TRD, which is planned to be operated intermittently with yearly shutdowns with maintenance access. Assuming yearly beam time of 3 months, the tested dosage is equivalent to more than 2 1/2 years of operation with high intensity beams. Increasing the expected survival rate to $1 - p > 95\%$ would require the testing of more than 58 capacitors under similar conditions. As the constructed setup can simultaneously hold and monitor 12 ceramic capacitors, this would require an additional 47 tested capacitors or 4 fully populated test setups.

The tested foil capacitor cannot be evaluated reliably, as the sample size is too small for any robust failure probability estimations. If these were to be further evaluated for the CBM-TRD, a new capacitor holder would need to be constructed, as the existing one only holds a single foil capacitor.

5 Conclusion and Outlook

In this thesis, a new framework for the analysis of data from various in-beam-tests was presented and discussed.

The experimental setups, with which the data was taken, were described in Sections 3.1.1 to 3.1.3. These setups had different configurations, which included different FEEs at SPS in 2016 (SPADIC 1.1) and at DESY II and GIF++ in 2017 (SPADIC 2.0). The geometric configuration was different in all of these setups, with the SPS setup having four layers with three SPADIC 1.1 FEBs on a SYSCORE DPB each, the DESY II setup having seven layers with zero to two active SPADIC 2.0 FEBs on AFCK DPBs, and the GIF++ setup having up to five SPADIC 2.0 FEBs on four AFCK DPBs on a single layer and two reference scintillation detectors.

The design goals and general structure of the framework is discussed in Section 3.2.1. The framework presented in this thesis is intended to provide tools to aid in the analysis of data from different in-beam-test campaigns and a set of example classes, which can also be used for a preliminary evaluation of the data. This required a new architecture for the analysis tasks, capable of handling the different electronics configurations. A parameter handler to manage the differences in the electronics configrations was introduced and new analysis tasks depending on it were developed. A schematic structure of the framework is provided in Figure 3.4.

The most important analysis classes/tasks included in the framework are also discussed in Section 3.2.1. The first discussed class is CbmTrdQABase, the base class for all analysis tasks in the framework. Two analysis tasks which provide a basic set of analyses are also discussed, in the CbmTrdQAHit and the CbmTrdSimpleClusterAnalysis classes. Finally two classes that provide processing for other tasks in the framework are discussed: the CbmTrdSimpleDigitizer, which creates CbmTrdDigis from the CbmSpadicRawMessages from the unpacker, and the CbmTrdSimpleClusterizer, a simple cluster search class based on prior work of the author [Mun16].

The base components of the framework were then used to create a new example analysis from scratch. This analysis class consists of four different analyses, which provide a comprehensive introduction into the functionality and the use of the framework. This class is discussed in Section 3.2.2 with the source code printed in listings A.1 to A.3 and a patch available here[1]. The source code includes the basic class structure and including an inheritance from CbmTrdQABase,

[1] Link to the full patch set against the OCT19 release of CBMROOT:
https://uni-muenster.sciebo.de/s/4wxXZAOQucvVXhq

the required constructor and the required functions for an analysis class `CreateHistograms()` and `Exec(Option_t*)`.

A set of four analyses touching on both raw data and cluster analysis was then motivated and described. Each of these required the creation of the necessary histograms, which was detailed only for the signal shape and the trigger type analyses in listing 3.1.

The first set of analyses was an analysis of the trigger type distribution and the cumulative signal shape on an individual channel for every possible channel in the setup. This required two histograms, one for the signal shape and one for the trigger types, see listing 3.1, for each of the 16 channels from both channel groups on every SPADIC on every DPB, the amount of which varies between the different setups. In listing 3.2, the process of querying and analyzing CbmSpadicRawMessage objects was detailed and discussed.

Furthermore, the process of analyzing different properties of CbmTrdCluster objects was discussed in listings 3.3 and 3.4, with the creation of a hitmap of the clusters on a detector layer and a reconstruction of the Pad Response Function on a detector layer.

This newly created analysis was then used on data from the various setups.

In the case of the SPS 2016 in-beam-test, the results are discussed in Section 3.3.1. The signal shape in Figure 3.5a showed the expected shape for a SPADIC 1.1, with the distribution of trigger types in Figure 3.5b being compatible with an even distribution of hit messages in the region of the examined channel. The 2D cluster hitmap in Figure 3.5c showed areas of strongly reduced intensity between channel groups, even on the same FEB. This was hypothesized to be either the result of a misconfiguration or a desynchronization between the channel groups. The PRF in Figure 3.5e showed several categories of artifacts that could not be fully explained by the available analyses. In addition, the efficiency of both of these analyses was discussed, as it was quite low at e_{hitmap} ≈6.6% and e_{PRF} ≈1.3%, and a potential route for further refinement of these analyses on this particular dataset was suggested.

For the DESY 2017 in-beam-test, the results are discussed in Section 3.3.2. In the histogram for the signal shape in Figure 3.6a, a secondary signal component, besides the primary one, was observed and explained due to a particularity of the SPADIC 2.0 trigger. When examining the distribution of trigger types in Figure 3.6b, an explanation of the imbalance of trigger types to the expectations could be found, by examining the channels position in the cluster hitmap in Figure 3.6c. The channel was found to be located on the side of the clearly visible beamspot, which explained the imbalance. The PRF in Figure 3.6e was very clear and could already be used to repeat a complex analysis from [Mun16]. The increased efficiencies of the hitmap and PRF analyses of e_{hitmap} ≈18.7% and e_{PRF} ≈18.2% suggest an improvement in the underlying data quality.

The GIF++ 2017 in-beam-test data were also examined in Section 3.3.3. For this dataset, the examined channel showed a few noteworthy features in the signal shape, see Figure 3.7a, that were discussed. The clipping and the constant band were explained by a mismatch of the gain and the high activity of the source. The results of the signal reconstruction algorithm

developed by J. Beckhoff in his masters thesis [Bec18] were clearly visible. Examining the trigger type distribution in Figure 3.7b showed a ratio compatible with the expectation of an evenly illuminated padplane, which was confirmed by examining Figure 3.7c. The hitmap showed a flat distribution of clusters, with the expected dips between the SPADICs. Like the DESY data in Figure 3.6e, the PRF in Figure 3.7e showed no strong artifacts. The efficiencies for the hitmap and PRF analyses were lower compared to the DESY data to $e_{\text{hitmap}} \approx 8.9\%$ and $e_{\text{PRF}} \approx 8.8\%$. This was explained by the high load placed on the detector by the high activity of the source.

Finally, a test of capacitors with regard to radiation hardness was detailed and discussed in chapter 4. This includes a description of the setup with the capacitor holder shown in Figure 4.1 and the bunker for the whole irradiation setup in Figure 4.2. The capacitors were exposed to intense neutron radiation from a RaBe source for a period of nearly two months and monitored for failures via the current draw, see Figure 4.5. The neutron dosage experienced by the capacitors was determined and the equivalent lifetime in the final CBM experiment was calculated. The ceramic capacitors were determined to have been tested to a dose equivalent to (8 ± 2) months of continuous Au+Au beam at 10 AGeV and 5 MHz in the center of the detector and (16 ± 3) months in the outer part, see Table 4.4. Since no failures occurred, according to Figure 4.5, at a sample size of eleven capacitors, the failure rate for the tested dose/lifetime was calculated via equation (4.4) to be below 23.8%, with a confidence interval of 95%.

This thesis detailed the development and usage of a framework for the analysis of data from various in-beam-tests and evaluated the performance of a newly constructed example class on data from three campaigns. The results were examined and possibilities to refine and specialize the provided class were outlined w.r.t. the individual dataset. Furthermore, an experimental setup to test ceramic capacitors considered for the use in the permanent SIS100 setup was detailed and discussed. An equivalent lifetime under a continuous high intensity beam was calculated and the maximum expected failure rate under these conditions was determined.

The development and the use of a framework for the analysis of in-beam-test data for the CBM-TRD were discussed. As the framework can be customized for other setups than the described ones, i.e. measurements in the lab or prior or later in-beam-tests, it can be used to track the performance of the CBM-TRD prototypes and the electronics over their development. For the discussed datasets from SPS, DESY II and GIF++, four simple analyses were shown and potential avenues to refine these analyses were discussed.

The test of the capacitors under neutron radiation hinted that the tested capcacitors might be suitable for use in the full experiment. Further tests are needed, as the failure rate under full CBM conditions could only be estimated to be below 23.8% for a period of (8 ± 2) months at the highest intensities. As this is longer than the planned 3 months of beamtime between shutdowns with maintenance access, the capacitors seem suitable, but further tests are needed, as discussed in Section 4.5.

List of Figures

List of Tables

Listings

Bibliography

[AB06] Mohammad Al-turany and Denis Bertini. “CbmRoot : Simulation and Analysis framework for CBM Experiment”. In: *Computing in High Energy and Nuclear Physics (CHEP-2006), Volume 1 of MACMILLAN Advanced Research Series*. 2006, pp. 170, 171.

[Abl+17] T. Ablyazimov, A. Abuhoza, et al. “Challenges in QCD matter physics –The scientific programme of the Compressed Baryonic Matter experiment at FAIR”. In: *The European Physical Journal A* 53.3 (2017-03), p. 60. ISSN: 1434-601X. DOI: 10.1140/epja/i2017-12248-y. URL: https://doi.org/10.1140/epja/i2017-12248-y.

[Arm13] T. Armbruster. “SPADIC - a Self-Triggered Detector Readout ASIC with Multi-Channel Amplification and Digitization”. PhD thesis. ZITI, Heidelberg University, 2013. URL: https://www-alt.gsi.de/documents/DOC-2013-Jun-21.html.

[Bat+07] G. Battistoni et al. “The FLUKA code: Description and benchmarking”. In: *AIP Conference Proceeding* 896 (2007-09), pp. 31–49.

[BBE18] *The Transition Radiation Detector of the CBM Experiment at FAIR : Technical Design Report for the CBM Transition Radiation Detector (TRD)*. Tech. rep. FAIR Technical Design Report. Darmstadt, 2018, 165 p. DOI: 10.15120/GSI-2018-01091. URL: https://repository.gsi.de/record/217478.

[Bec18] Johannes Beckhoff. “Automated Test Stand Setup and Signal Reconstruction for the CBM-TRD”. MA thesis. WWU Münster, Institut für Kernphysik, 2018-08.

[Ber14] Cyrano S.H. Bergmann. “Development, Simulation and Test of Transition Radiation Detector Prototypes for the Compressed Baryonic Matter Experiment at the Facility for Antiproton and Ion Research”. PhD thesis. Westfälische Wilhelms-Universität Münster, 2014. URL: http://www.uni-muenster.de/imperia/md/content/physik_kp/agwessels/thesis_db/ag_wessels/bergmann_2014_dissertation.pdf.

[BRR08] W. Blum, W. Riegler, and L. Rolandi. *Particle Detection with Drift Chambers*. 2nd ed. Springer, 2008. ISBN: 978-3-540-76683-4. URL: http://mipt.jinr.ru/xdocs/blum.pdf.

[Bur+97] B. Burgkhardt, G. Fieg, et al. “The neutron fluence and H*(10) response of the new LB 6411 REM counter”. In: *Radiation Protection Dosimetry* 70 (1997-04). DOI: 10.1093/oxfordjournals.rpd.a031977.

[Cla51] Arthur C. Clarke. *The exploration of space / by Arthur C. Clarke*. English. Temple Press London, 1951.

[Com07] Haade – Wikimedia Commons. *Solid Angle.png*. 2007-02. URL: https://commons.wikimedia.org/wiki/File:Solid_Angle.png.

[Com08] Wikipedagoge – Wikimedia Commons. *SolidAngle Pyramid.png*. 2008-03. URL: https://commons.wikimedia.org/wiki/File:SolidAngle_Pyramid.png.

[Die+19] R. Diener, J. Dreyling-Eschweiler, et al. “The DESY II test beam facility”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 922 (2019-04), pp. 265–286. ISSN: 0168-9002. DOI: 10.1016/j.nima.2018.11.133. URL: http://dx.doi.org/10.1016/j.nima.2018.11.133.

[Dun+35] J. R. Dunning, G. B. Pegram, et al. “Interaction of Neutrons with Matter”. In: *Phys. Rev.* 48 (3 1935-08), pp. 265–280. DOI: 10.1103/PhysRev.48.265. URL: https://link.aps.org/doi/10.1103/PhysRev.48.265.

[Fri19] Volker Friese. “CBM software - an overview”. In: *CBM Software School*. 2019-09.

[Gat+79] E. Gatti, A. Longoni, et al. “Optimum Geometry for Strip Cathodes on Grids in MWPC for Avalanche Localization Along the Anode Wires”. In: *Nucl. Instr. Meth.* 163 (1979), pp. 83–92. DOI: http://dx.doi.org/10.1016/0029-554X(79)90035-1. URL: http://www.sciencedirect.com/science/article/pii/0029554X79900351.

[Gui15] R. Guida. “GIF++: The new CERN Irradiation Facility to test large-area detectors for HL-LHC”. In: *2015 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*. 2015, pp. 1–4.

[HJ00] U. Heinz and M. Jacob. “Evidence for a New State of Matter: An Assessment of the Results from the CERN Lead Beam Programme”. In: *arXiv:nucl-th/0002042* (2000-02). URL: http://arxiv.org/abs/nucl-th/0002042.

[KP55] G H Kinchin and R S Pease. “The Displacement of Atoms in Solids by Radiation”. In: *Reports on Progress in Physics* 18.1 (1955-01), pp. 1–51. DOI: 10.1088/0034-4885/18/1/301. URL: https://doi.org/10.1088%2F0034-4885%2F18%2F1%2F301.

[Leo87] W. Leo. *Techniques for Nuclear and Particle Physics Experiments*. Springer-Verlag, 1987.

[Mat82] Hj. Matzke. “Radiation damage in crystalline insulators, oxides and ceramic nuclear fuels”. In: *Radiation Effects* 64.1-4 (1982), pp. 3–33. DOI: 10.1080/00337578208222984. eprint: https://doi.org/10.1080/00337578208222984. URL: https://doi.org/10.1080/00337578208222984.

[Mat88] E. Mathieson. “Cathode Charge Distributions in Multiwire Chambers”. In: *Nucl. Instr. Meth. A* 270 (1988), pp. 602–603. URL: http://www.jlab.org/Hall-D/detector/fdc/ref/mathieson.pdf.

[Mey19] Adrian Meyer-Ahrens. “Electron Detection Efficiency of the CBM-TRD Prototypes in Testbeams at DESY”. MA thesis. Westfälische Wilhelms-Universität Münster, Institut für Kernphysik, 2019-03.

[Mun16] Philipp Munkes. “Ereignis-Rekonstruktion für CBM TRD-Testdaten”. bscthesis. Westfälische Wilhelms-Universität, 2016-07.

[Pfe+16] Dorothea Pfeiffer, Georgi Gorine, et al. *The radiation field in the Gamma Irradiation Facility GIF++ at CERN*. 2016. arXiv: 1611.00299 [physics.ins-det].

[Raf19] Johann Rafelski. *Discovery of Quark-Gluon-Plasma: Strangeness Diaries*. 2019. arXiv: 1911.00831 [hep-ph].

[ST17] Ilya Selyuzhenkov and Alberica Toia, eds. *CBM Progress Report 2016*. Literaturangaben. Darmstadt: GSI, 2017. ISBN: 978-3-9815227-4-7. URL: https://repository.gsi.de/record/201318.

[Tec] Berthold Technologies. *Neutron Probe LB 6411 for Measurement of the Ambient Dose Equivalent for Neutrons*.

[TS18] *CBM Progress Report 2017*. Tech. rep. CBM Progress Report 2017. Darmstadt, 2018. DOI: 10.15120/GSI-2018-00485. URL: https://repository.gsi.de/record/209729.

[VS83] A. Van Oosterom and J. Strackee. “The Solid Angle of a Plane Triangle”. In: *IEEE Transactions on Biomedical Engineering* BME-30.2 (1983), pp. 125–126.

A Source Code for Example Analysis

A.1 Macro

Listing A.1: readTsaExample.C

```
enum kSetupID { kSPS = 0, kDESY = 1, kGIF = 2 };
void readTsaExample(TString inFile = "/opt/CBM/Daten/121_cern-fex.tsa",
                    int Setup = 0) {
  // --- Specify number of events to be produced.
  // --- -1 means run until the end of the input file.
  Int_t nEvents = -1;
  TString outFile = inFile;
  outFile.ReplaceAll(".tsa", ".root");
  // --- Set log output levels
  //  FairLogger::GetLogger()->SetLogScreenLevel("ERROR");
  //  FairLogger::GetLogger()->SetLogVerbosityLevel("LOW");
  // --- Set debug level
  gDebug = 0;

  CbmFlibFileSourceNew *source = new CbmFlibFileSourceNew();

  source->SetFileName(inFile);
  CbmTSUnpackSpadic *spadic_unpacker = new CbmTSUnpackSpadic();
  CbmTSUnpackSpadic20 *spadic_unpacker20 = new CbmTSUnpackSpadic20();

  switch (Setup) {
  case kSPS:
    source->AddUnpacker(spadic_unpacker, 0x40);
    CbmTrdTestBeamTools::Instance(new CbmTrdTestBeamTools);
    break;
  case kDESY:
    source->AddUnpacker(spadic_unpacker20, 0x10);
    CbmTrdTestBeamTools::Instance(new CbmTrdTestBeamTools2017DESY);
    break;
  case kGIF:
    source->AddUnpacker(spadic_unpacker20, 0x10);
    CbmTrdTestBeamTools::Instance(new CbmTrdTestBeamTools2017GIF);
    break;
```

```
34    }
35    // --- Run
36    FairRunOnline *run = new FairRunOnline(source);
37    run->SetOutputFile(outFile);
38    FairTask *digitize = new CbmTrdSimpleDigitizer();
39    run->AddTask(digitize);
40
41    FairTask *Clusterrize = new CbmTrdSimpleClusterizer();
42    run->AddTask(Clusterrize);
43
44    FairTask *ClusterAnalysis = new CbmTrdExampleAnalysis();
45    run->AddTask(ClusterAnalysis);
46    run->Init();
47    run->Run(nEvents, 0); // run until end of input file
48 }
```

A.2 Header

Listing A.2: CbmTrdExampleAnalysis.h

```
1  #ifndef FLES_READER_TASKS_CBMTRDEXAMPLEANALYSIS_H_
2  #define FLES_READER_TASKS_CBMTRDEXAMPLEANALYSIS_H_
3
4  #include <CbmTrdQABase.h>
5
6  class CbmTrdExampleAnalysis : public CbmTrdQABase {
7  public:
8    CbmTrdExampleAnalysis() : CbmTrdQABase("CbmTrdExampleAnalysis"){};
9    virtual ~CbmTrdExampleAnalysis(){};
10
11   virtual void CreateHistograms();
12   virtual void Exec(Option_t *opt);
13
14   ClassDef(CbmTrdExampleAnalysis, 1);
15 };
16
17 #endif /* FLES_READER_TASKS_CBMTRDEXAMPLEANALYSIS_H_ */
```

A.3 Sourcefile

Listing A.3: CbmTrdExampleAnalysis.cxx

```
#include <CbmTrdExampleAnalysis.h>

const auto kHalfSpadic = CbmTrdTestBeamTools::kSpadicSize::kHalfSpadic;

ClassImp(CbmTrdExampleAnalysis);

void CbmTrdExampleAnalysis::CreateHistograms() {
  for (Int_t dpb = 0; dpb < fBT->GetNrRobs(); dpb++) {
    for (Int_t spadic = 0; spadic < fBT->GetNrSpadics() * 2; spadic++) {
      for (Int_t channel = 0; channel < 16; channel++) {
        TString spadicN = GetSpadicName(dpb, spadic, "DPB", kHalfSpadic);
        TString histN = "Self_triggered_SignalShape_" + spadicN +
                        "_Channel_" + std::to_string(channel);

        fHm->Add(histN.Data(), new TH2I(histN.Data(), histN.Data(), 32,
                      -0.5,31.5, 512, -256.5, 255.5));
        fHm->H2(histN.Data())->GetXaxis()->SetTitle("Sample Nr");
        fHm->H2(histN.Data())->GetYaxis()->SetTitle("ADC Value");
        histN = "Trigger_type_for_Hitmessages_" + spadicN + "_Channel_" +
                std::to_string(channel);
        fHm->Add(histN.Data(),
                 new TH1D(histN.Data(), histN.Data(), 4, -0.5, 3.5));
        fHm->H1(histN.Data())->GetXaxis()->SetTitle("Trigger Type");
      }
    }
  }

  // Create Histograms for the cluster heatmap for individual layers
  for (int layer = 0; layer < fBT->GetNrLayers(); layer++) {
    TString histN = "Cluster_Heatmap_Layer_" + std::to_string(layer);
    int nrOfRows = fBT->GetNrRows(layer);
    int rowNr = fBT->GetNrColumns(layer);
    float padWidth = fBT->GetPadWidth(layer);
    float padHeight = fBT->GetPadHeight(layer);
    fHm->Add(histN.Data(),
             new TH2I(histN.Data(), histN.Data(), rowNr * 4,
               -0.5 * padWidth,(rowNr - 0.5) * (padWidth),
               nrOfRows * 1, -0.5 * padHeight,
               (nrOfRows - 0.5) * padHeight));
    TH1 *currentHist = fHm->H2(histN.Data());
    currentHist->GetXaxis()->SetTitle("X/cm");
```

```
42      currentHist->GetYaxis()->SetTitle("Y/cm");
43    }
44    // Create Histograms for the Pad Respons Function
45    for (int layer = 0; layer < fBT->GetNrLayers(); layer++) {
46      TString histN = "PRF_Layer_" + std::to_string(layer);
47      fHm->Add(histN.Data(), new TH2I(histN.Data(), histN.Data(), 51, -2.55,
48                       2.55, 101, -0.5, 100.5));
49      TH1 *currentHist = fHm->H2(histN.Data());
50      currentHist->GetXaxis()->SetTitle("d/Pad Width");
51      currentHist->GetYaxis()->SetTitle("% of Total Charge");
52    }
53    // Create Histograms for the efficiency determination
54    const char *binNames[2] = {"Used", "Total"};
55    TString histN = "Efficiency Heatmap";
56    fHm->Add(histN.Data(), new TH1D(histN.Data(), histN.Data(),
57                     2, -0.5, 1.5));
58    TH1* currentHist = fHm->H1(histN.Data());
59    for (int i = 1; i <= 2; i++)
60      currentHist->GetXaxis()->SetBinLabel(i, binNames[i - 1]);
61
62    histN = "Efficiency PRF";
63    fHm->Add(histN.Data(), new TH1D(histN.Data(), histN.Data(),
64                     2, -0.5, 1.5));
65    currentHist = fHm->H1(histN.Data());
66    for (int i = 1; i <= 2; i++)
67      currentHist->GetXaxis()->SetBinLabel(i, binNames[i - 1]);
68  }
69
70
71
72
73  void CbmTrdExampleAnalysis::Exec(Option_t *) {
74    // Create anonymous fill function
75    auto FillHist = [&](TH1 *h, int used, int total) {
76      h->Fill("Used", used);
77      h->Fill("Total", total);
78    };
79    long NrRawMessages = fRaw->GetEntriesFast();
80    for (int iRaw = 0; iRaw < NrRawMessages; iRaw++) {
81      CbmSpadicRawMessage *currentRaw =
82          static_cast<CbmSpadicRawMessage *>(fRaw->At(iRaw));
83      if (currentRaw->GetHit() != true)
84        continue;
85      // Count only Hit Messages
86      int dpb = fBT->GetRobID(currentRaw);
```

```
87      int spadic = fBT->GetSpadicID(currentRaw);
88      int channel = currentRaw->GetChannelID();
89      TString spadicN = GetSpadicName(dpb, spadic, "DPB", kHalfSpadic);
90      TString histN = "Trigger_type_for_Hitmessages_" + spadicN +
91                      "_Channel_" + std::to_string(channel);
92      fHm->H1(histN.Data())->Fill(currentRaw->GetTriggerType());
93      if (currentRaw->GetTriggerType() != 1)
94        continue;
95      histN = "Self_triggered_SignalShape_" + spadicN + "_Channel_" +
96              std::to_string(channel);
97      TH2 *HistPtr = fHm->H2(histN.Data());
98      int NrSamples = currentRaw->GetNrSamples();
99      int *Samples = currentRaw->GetSamples();
100     for (int iSample = 0; iSample < NrSamples; iSample++) {
101       HistPtr->Fill(iSample, Samples[iSample]);
102     }
103   }
104
105   std::vector<TH2 *> Heatmaps;
106   for (Int_t layer = 0; layer < fBT->GetNrLayers(); layer++) {
107     TString histN = "Cluster_Heatmap_Layer_" + std::to_string(layer);
108     Heatmaps.push_back(fHm->H2(histN.Data()));
109   }
110
111   int UsedClusters = 0;
112   uint NrClusters = fClusters->GetEntriesFast();
113   for (uint iCluster = 0; iCluster < NrClusters; ++iCluster) {
114     CbmTrdCluster *CurrentCluster =
115         static_cast<CbmTrdCluster *>(fClusters->At(iCluster));
116     if (CurrentCluster) {
117       if (fBT->ClassifyCluster(CurrentCluster) !=
118           CbmTrdTestBeamTools::CbmTrdClusterClassification::kNormal)
119         continue;
120       UsedClusters++;
121       int layer = fBT->GetLayerID(CurrentCluster);
122       double center = fBT->GetCentralColumnID(CurrentCluster);
123       double displacement = fBT->GetColumnDisplacement(CurrentCluster);
124       double padWidth = fBT->GetPadWidth(layer);
125       double xPos = padWidth * (center + displacement);
126       center = fBT->GetCentralRowID(CurrentCluster);
127       displacement = fBT->GetRowDisplacement(CurrentCluster);
128       double padHeight = fBT->GetPadHeight(layer);
129       double yPos = padHeight * (center + displacement);
130       Heatmaps.at(layer)->Fill(xPos, yPos);
131     }
```

```
132   }
133
134   FillHist(fHm->H1("Efficiency Heatmap"), UsedClusters, NrClusters);
135   UsedClusters = 0;
136
137   for (uint iCluster = 0; iCluster < NrClusters; ++iCluster) {
138     CbmTrdCluster *CurrentCluster =
139         static_cast<CbmTrdCluster *>(fClusters->At(iCluster));
140     if (CurrentCluster) {
141       if (fBT->ClassifyCluster(CurrentCluster) !=
142           CbmTrdTestBeamTools::CbmTrdClusterClassification::kNormal)
143         continue;
144       if (fBT->GetColumnWidth(CurrentCluster) >= 3) {
145         UsedClusters++;
146         int layer = fBT->GetLayerID(CurrentCluster);
147         double displacement = fBT->GetColumnDisplacement(CurrentCluster);
148         double center = fBT->GetCentralColumnID(CurrentCluster);
149         center += displacement;
150         TString histN = "PRF_Layer_" + std::to_string(layer);
151         TH2 *currentHist = fHm->H2(histN.Data());
152         const std::vector<int> & Digis = CurrentCluster->GetDigis();
153         float Chg = fBT->GetCharge(CurrentCluster) * 1e4;
154         for (uint i = 0; i < Digis.size(); i++) {
155           CbmTrdDigi *Digi = static_cast<CbmTrdDigi *>(
156                                  fDigis->At(Digis[i]));
157           float colId = CbmTrdAddress::GetColumnId(fBT->GetAddress(Digi));
158           currentHist->Fill(colId - (center),
159                             (Digi->GetCharge() / Chg) * 100.);
160         }
161       }
162     }
163   }
164
165   FillHist(fHm->H1("Efficiency PRF"), UsedClusters, NrClusters);
166 }
```

Acknowledgements

I would like to thank Prof. Dr. Anton Andronic and Prof. Dr. Christian Klein-Bösing for the opportunity to work on the measurement campaigns described in this thesis, the described software framework and to write this thesis about it. Their advice proved helpful whenever I sought it. I would also like to thank Norbert Heine, the team at the *Feinmechanische Werkstatt* and Prof. Dr. Alfons Khoukaz for their assistance during the neutron irradiation study.

I want to give my special thanks to Philipp Kähler, Adrian Meyer-Ahrens and Ruben Weber for all the help in form of suggestions, constructive criticism, discussions of plots and proof reading. I thank also the additional proof readers Hannes Morgenweck, Patrick Schneider and my sister Stella Munkes, and also to my parents Katharina and Dr. Ulrich Munkes.

Declaration of Academic Integrity

I hereby confirm that this thesis on *Analysis framework for CBM TRD Test Beam Data* is solely my own work and that I have used no sources or aids other than the ones stated. All passages in my thesis for which other sources, including electronic media, have been used, be it direct quotes or content references, have been acknowledged as such and the sources cited.

_____

(date and signature of student)

I agree to have my thesis checked in order to rule out potential similarities with other works and to have my thesis stored in a database for this purpose.

_____

(date and signature of student)