# An FPGA-based Sampling ADC for the Crystal Barrel Calorimeter

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

**Johannes Sebastian Müllers**

aus

Kirchen (Sieg)

Bonn, Juli 2019

Angefertigt mit Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Rheinischen Friedrich-Wilhelms-Universität Bonn.

Tag der Promotion: 26.11.2019

Erscheinungsjahr: 2019

1. Gutachterin: Prof. Dr. Ulrike Thoma
Helmholtz-Institut für Strahlen- und Kernphysik
Rheinische Friedrich-Wilhelms-Universität Bonn

2. Gutachter: Prof. Dr. Bernhard Ketzer
Helmholtz-Institut für Strahlen- und Kernphysik
Rheinische Friedrich-Wilhelms-Universität Bonn

# Abstract

The CBELSA/TAPS experiment in Bonn investigates the excitation spectra of protons and neutrons through meson-photoproduction. With its ability to polarize the target nucleons and the incident photon beam, the experiment has contributed significantly to a better understanding of the baryon excitation spectrum, and with that also to the understanding of the strong interaction in the non-perturbative regime. Since a recent upgrade, the experiment's main electromagnetic calorimeter, the Crystal Barrel, is read out by avalanche photodiodes. Their signal is digitized by integrating Fastbus ADCs, providing a value proportional to the energy deposited in the calorimeter crystals. As a result of long conversion and transfer times, those ADCs have become the limiting factor in the data acquisition, with possible readout rates of less than 2 kHz. Moreover, the possibility to identify pile-up, i.e. quickly-succeeding energy deposits that overlap in the integration window, is presently missing.

Already years ago, it has been investigated whether this readout system could be replaced by faster and more modern sampling ADCs, which offer access to the waveform representation for a more sophisticated analysis, but the investigations were limited to commercially available digitizers with high cost and low channel densities. In addition, the firmware (*operating system*) of such digitizers is usually closed-source, which does not allow for the implementation of experiment-specific algorithms. Due to the above reasons, the investigations had not led to a satisfactory solution, and the Fastbus ADCs are still in operation today.

In this thesis, the development and test of a custom (non-commercial) FPGA-based sampling ADC, which will replace the Fastbus ADC, has been driven forward. This so-called CB-SADC (Crystal Barrel Sampling Analog-to-Digital Converter) has been adapted from a prototype of the PANDA experiment in such a way that it is suited to operate within the specific conditions of the CBELSA/TAPS experiment.

Apart from the hardware development, the firmware for the FPGA, which processes the digitized data, was designed and tested extensively. Specific algorithms allow not only the determination of the deposited energy and the timestamps of each event, but also an event-wise baseline determination and the detection of pile-up events. An offline correction of pile-up events, which occur with significant frequency in the forward region of the calorimeter, leads to a higher data quality and improved efficiency and statistics.

Two prototype iterations of the CB-SADC have been produced and were tested thoroughly in the laboratory and in connection with the CBELSA/TAPS experiment. The results of those tests, and preliminary analyses of production beam times with 50 % of the calorimeter's forward half being read out by the CB-SADCs, showed an improvement of the data quality. The timestamp determination of the CB-SADCs in the energy regime below 10 MeV has provided data where the experiment's TDC (Time-to-Digital Converter) either has worse resolution or cannot provide timestamps at all. In addition, standalone tests in the laboratory and tests with the data acquisition system have confirmed a much higher readout speed compared to the integrating Fastbus ADCs. Based on the achieved results, it was finally decided to equip the whole calorimeter with the new CB-SADC readout.

At the time of finalizing this thesis, all CB-SADCs have been produced and are prepared for installation in the experimental hall. They will be running in parallel with the integrating Fastbus ADCs during the next production beam times, offering an opportunity to test the CB-SADC readout system as a whole. As soon as it is proven that the new readout works reliably, the limiting integrating Fastbus ADCs will be decommissioned and the CBELSA/TAPS experiment can start taking data with an increased readout rate.

# Outline

This thesis comprises ten chapters.

1. **Analog-To-Digital Conversion** introduces the technology of ADCs and summarizes the most popular architectures. The theoretical background to understanding the limitations with regard to noise and resolution is discussed.

2. **The CBELSA/TAPS Experiment** provides a brief introduction to the physics motivation of the experiment and an overview of the experimental setup in which the CB-SADC is being integrated.

3. **SADC Upgrade** motivates the development of the CB-SADC and shows how it can contribute to higher data quality and statistics for the CBELSA/TAPS experiment.

4. **A First Step: The 16-Channel PANDA-SADC Prototype** marks the *hands-on* beginning of this thesis. The first proof-of-concept is demonstrated with one of the earliest PANDA SADC prototypes.

5. **CB-SADC Hardware Development** describes the circuits on the CB-SADC board, and focuses in detail on the changes for the initial adaption of the PANDA-SADC and the succeeding revisions of the CB-SADC.

6. **Related Hardware Development** complements the previous chapter with the development of auxiliary hardware, such as the CB-SADC's power supply and the analog signal processing stage.

7. **Firmware Development** shows the detailed description of the CB-SADC's *operating system*, the firmware on the Field Programmable Gate Arrays (FPGAs).

8. **Software Development** provides an overview of the software development associated with the CB-SADC project, ranging from simple laboratory tools to the *local eventbuilder* for the integration into the CBELSA/TAPS experiment.

9. **Measurements Performed with the CB-SADC** shows the CB-SADC's capabilities in the laboratory and in the experimental environment, including the evaluation of preliminary analyses from the first production beam times with partial CB-SADC readout.

10. **Summary and Outlook** summarizes the progress made and the results obtained within this thesis and provides a brief outlook.

# Contents


**1. Analog-to-Digital Conversion** **1**
1.1. Introduction . . . 1
1.2. Principle of an ADC . . . 2
1.3. ADC Architectures . . . 3
1.3.1. Flash ADC . . . 3
1.3.2. Pipelined ADC . . . 6
1.3.3. Successive Approximation ADC . . . 10
1.3.4. Sigma-Delta ADC . . . 11
1.3.5. Counting and Integrating ADC . . . 13
1.4. ADC Performance . . . 13
1.4.1. Johnson–Nyquist Noise . . . 14
1.4.2. 1/f Noise . . . 15
1.4.3. Quantization Noise . . . 15
1.4.4. Input Referred Noise . . . 18
1.4.5. Aperture Jitter . . . 19
1.4.6. Aliasing . . . 21

**2. The CBELSA/TAPS Experiment** **23**
2.1. Physics Background . . . 23
2.1.1. Strong Interaction . . . 24
2.1.2. Baryon Spectroscopy . . . 25
2.2. ELSA . . . 29
2.2.1. Linear Accelerators . . . 29
2.2.2. Booster Synchrotron . . . 29
2.2.3. Stretcher Ring . . . 30
2.3. CBELSA/TAPS Experimental Setup Overview . . . 31
2.4. Bremsstrahlung Targets . . . 32
2.5. Tagging Hodoscope . . . 33
2.6. Photoproduction Targets . . . 33
2.7. Detector Systems . . . 34
2.7.1. Inner Detector . . . 34
2.7.2. Crystal Barrel Calorimeter . . . 35
2.7.3. Gas Čerenkov Detector . . . 35
2.7.4. MiniTAPS Calorimeter . . . 36
2.7.5. Gamma Intensity Monitor and Flux Monitor . . . 36
2.8. Readout Electronics . . . 37
2.8.1. Motivation for the APD Readout . . . 37
2.8.2. Concept of the APD Readout . . . 38

2.8.3. Front-End Electronics . . . 39
2.8.4. Slice Board . . . 41
2.8.5. Buffer/Timing Filter and Energy Sum . . . 42
2.8.6. Discriminator/TDC/Cluster Finder . . . 43
2.8.7. QDC Shaper . . . 44
2.8.8. QDC . . . 45
2.9. Trigger . . . 45
2.9.1. Event Qualification . . . 46
2.9.2. Trigger Timing . . . 46
2.9.3. Trigger Efficiency . . . 47
2.9.4. Trigger Electronics . . . 48
2.10. Data Acquisition . . . 50
2.10.1. DAQ Terminal . . . 50
2.10.2. Online Monitor . . . 50
2.11. Auxiliary Systems . . . 52
2.11.1. Light Pulser . . . 52
2.11.2. Slowcontrol . . . 52

**3. SADC Upgrade** **55**
3.1. Motivation . . . 55
3.2. Previous Studies . . . 57
3.2.1. Studies of the Energy Resolution of CsI(Tl) with High-speed ADCs . . . 57
3.2.2. Development of a VME Readout for the Struck Dl300 Flash-ADC . . . 57
3.2.3. Studies with the Sis 3320-250 Sampling-ADC . . . 57
3.2.4. Studies of Feature Extraction Methods with the SIS 3320-250 Sampling-ADC . . . 58
3.2.5. Qualitative Comparison of Commercially Available Sampling-ADCs . . . 58
3.3. Requirements and Explored Solutions . . . 58
3.3.1. Commercially available SADCs . . . 59
3.3.2. Custom SADC for the PANDA Electromagnetic Calorimeter . . . 59
3.4. Crystal Barrel SADC - Project Timeline . . . 61

**4. A First Step: The 16-Channel PANDA-SADC Prototype** **63**
4.1. Technical Specification . . . 63
4.2. Getting Started with Signal Processing . . . 64
4.3. Implementation of a Network Interface . . . 64
4.4. Examining the Transmitted Data . . . 65
4.5. Summary . . . 67

**5. CB-SADC Hardware Development** **69**
5.1. Concept . . . 70
5.2. Digitization . . . 72
5.2.1. Input Buffers and Anti-Aliasing . . . 72
5.2.2. Pipeline-ADC . . . 74
5.3. Clocking . . . 77
5.3.1. Jitter . . . 79

5.4. FPGA . . . 81
5.4.1. What is an FPGA? . . . 81
5.4.2. KINTEX-7 FPGA . . . 84
5.5. Programming and Arbitration . . . 84
5.5.1. Programming . . . 84
5.5.2. Arbitration . . . 85
5.6. Power Delivery Network . . . 86
5.6.1. Topology . . . 86
5.6.2. Layout Considerations . . . 87
5.6.3. Simulation . . . 88
5.7. Adaption and Changes of the Design . . . 91
5.7.1. Form Factor . . . 91
5.7.2. Power Supply . . . 92
5.7.3. Arbiter/PLL . . . 92
5.7.4. Trigger Connections . . . 92
5.7.5. $I^2C$ Slowcontrol . . . 95
5.7.6. Heatsink . . . 97
5.8. Revision Summary . . . 99
5.8.1. Revision 1.0 . . . 99
5.8.2. Revision 1.1 . . . 100
5.8.3. 125 MS/s Test . . . 101
5.8.4. Revision 1.2 (Production Revision) . . . 101

**6. Related Hardware Development 103**
6.1. Analog Input Card . . . 103
6.1.1. Signal Shaping . . . 104
6.1.2. Baseline Shifting . . . 106
6.1.3. Pole-Zero Cancellation . . . 108
6.2. Testboard . . . 115
6.2.1. Test/Analysis . . . 116
6.3. Power Supply . . . 118
6.3.1. Requirements . . . 118
6.3.2. Topology . . . 118
6.3.3. One-Channel Prototype . . . 120
6.3.4. Four-Channel Prototype . . . 122
6.3.5. Four-Channel Final Design . . . 126
6.4. CB-SADC Crate Controller . . . 127
6.4.1. First Concept: The SADC Controller . . . 128
6.4.2. Second Concept: The Backplane . . . 129
6.4.3. Future Concept: The CB-Sampling Analog-to-Digital Converter (SADC) Crate Controller . . . 132
6.5. Slowcontrol Extension . . . 134

**7. Firmware Development 135**
7.1. Overview of the Firmware . . . 136

7.2. Boot-up: Reset and Initialization . . . 137
7.2.1. Reset State Machine . . . 137
7.2.2. Phase Locked Loop Configuration . . . 139
7.2.3. ADC Initialization and Configuration . . . 139
7.3. ADC Data Interface . . . 141
7.3.1. Serialized Data Links . . . 141
7.3.2. Phase Calibration . . . 142
7.3.3. Bitslip . . . 146
7.3.4. De-Randomization . . . 146
7.3.5. Data Preprocessing . . . 148
7.4. Network . . . 153
7.4.1. Physical Layer: PCS/PMA IP Core . . . 155
7.4.2. Data Link Layer: Tri-Mode Ethernet MAC IP Core . . . 156
7.4.3. Network and Transport Layer: UDP/IP IP Core . . . 156
7.4.4. RX/TX Arbitration . . . 158
7.4.5. Reliability Protocol . . . 159
7.4.6. De-Randomizing Buffer . . . 160
7.5. Feature Extraction . . . 162
7.5.1. Baseline Tracker . . . 162
7.5.2. Integration . . . 163
7.5.3. Peak Finder . . . 164
7.5.4. Constant Fraction Discriminator . . . 164
7.5.5. Pile-Up Detection . . . 166
7.5.6. Packet Content . . . 169
7.5.7. Performance . . . 170
7.6. Sample Extraction . . . 171
7.6.1. Configuration . . . 171
7.6.2. Performance and Packet Content . . . 172
7.7. Trigger . . . 173
7.7.1. Self Trigger . . . 173
7.7.2. External Trigger . . . 173
7.7.3. Network Trigger . . . 173
7.7.4. Clock Trigger . . . 173
7.8. Further Firmware Development . . . 174
7.8.1. Backplane Firmware . . . 174
7.8.2. ELB-VME-VFB Slowcontrol/Trigger Firmware . . . 176

**8. Software Development** **177**
8.1. Qt Debugging Tool . . . 177
8.1.1. In Operation . . . 177
8.1.2. Code . . . 179
8.2. Python3 Tool Set . . . 179
8.3. Receiver and Configuration Tools . . . 180
8.4. Local Event Builder . . . 182
8.4.1. Waveform Compression . . . 183
8.4.2. Zero Suppression . . . 183

8.5. Slowcontrol . . . 183
8.5.1. Standalone Readout Tool . . . 183
8.5.2. Integration into the $I^2C$ Daemon . . . 184

**9. Measurements Performed with the CB-SADC 185**
9.1. The CB-SADC Rack . . . 185
9.1.1. CB-SADC-Server . . . 187
9.1.2. Network Infrastructure . . . 188
9.1.3. Sync Master and Slowcontrol . . . 190
9.2. Comparison of CB-SADC and QDC in the Laboratory . . . 190
9.2.1. Setup . . . 191
9.2.2. Measurements and Analysis . . . 192
9.3. Minimum Ionizing Particles @ CBELSA/TAPS . . . 194
9.3.1. Setup . . . 194
9.3.2. Analysis . . . 194
9.4. Readout of the Energy Sum . . . 197
9.5. Noise . . . 198
9.5.1. Analysis . . . 198
9.5.2. Discussion . . . 201
9.5.3. 312 kHz Noise Investigation . . . 201
9.6. Pile-up Detection and Recovery . . . 203
9.7. Pion Decay Analysis . . . 206
9.7.1. Setup . . . 206
9.7.2. Analysis . . . 206
9.7.3. Discussion . . . 209
9.8. Timing Capabilities . . . 210
9.8.1. Analysis . . . 210
9.8.2. Discussion . . . 212
9.9. Deadtime and Efficiency . . . 214
9.9.1. Readout Rates . . . 214

**10. Summary and Outlook 219**

**Appendices**

**A. Pictures 221**

**B. FPGA Firmware Development 229**
B.1. VHDL Coding . . . 229
B.2. Synthesis and Implementation . . . 231
B.3. Debugging . . . 232
B.4. Simulation . . . 233

**C. Configuration 235**
C.1. Configuration of Feature/Sample Extraction . . . 235
C.2. Configuration of LTM9009-14 ADCs . . . 235
C.3. Configuration and Control of the Reliability Protocol . . . 235

C.4. Configuration of Operating Parameters . . . . . 235

**D. Plots** **239**

**E. Code** **245**

**F. Other Figures** **259**

**Bibliography** **265**

**List of Figures** **279**

# 1. Analog-to-Digital Conversion

Since analog-to-digital conversion is at the heart of this thesis, a general introduction into the subject will be given in this chapter, before the aims of the CBELSA/TAPS experiment are discussed and the experimental setup is presented. A brief introduction will describe why analog-to-digital conversion is indispensable for computer-aided analysis of experimental data. Then, section 1.2 will present core principles behind Analog-to-Digital Converters (ADCs). Section 1.3 will provide an overview of the different types of ADCs and specify the domains in which they are used. Subsequently the most important aspects will be presented for each type. Last, possible noise and error sources and their origins are discussed in section 1.4 as a basis for investigations presented in the course of this thesis.

Many figures and notions in this chapter are taken from *The Data Conversion Handbook*, a book from Analog Devices engineer Walt Kester [1], an the related article *Which ADC Architecture Is Right for Your Application?* [2].

## 1.1. Introduction

A measurable quantity is presented to us in the macroscopic world usually as a continuous scalar or vector value. Our human senses allow us to qualitatively, and to some extent quantitatively, grasp those values. We are able to tell whether the milk for our coffee is *cold from the fridge*, has *room temperature*, or is *hot*[1]. In some cases like those, ways to easily measure and present those quantities by means of fully analog display are available: the temperature of milk may be determined with the help of the known temperature-dependent expansion of a liquid inside a thermometer, calibrated using the freezing and boiling point of water. The weight of coffee beans may be determined with a balance scale and gauged counterweights. The volume of coffee may be determined with gauged markings on a container.

In an imaginable scenario, 132 students would be able to track the temperature of 30 milk jugs in 8-hour shifts, with an interval of 2 min and a relative error of about 0.25 °C. The scenario could be crudely translated to the scope of this thesis as follows: The task is to measure the energy (heat) deposited in 1320 scintillator crystals (milk jugs), which would require intervals of less than 100 µs with relative errors of about 0.0015 °C. Without a doubt, one accepts that such an amount of data can only be processed by means of automated (electronic) data processing. The necessary steps for this are:

**Use a sensor to convert the measurable quantity to an electronic signal.** In our example, the thermometer would be replaced with a temperature-dependent resistance (NTC, PTC) and an Operational Amplifier (op-amp), which converts the temperature to a voltage or a current with a certain proportionality. Usually a linear proportionality is desired, but it always depends on the application. Regarding this thesis, it will be

[1] A comprehensible overview of thermoception is provided in [3].

shown that our primary quantity is the energy of photons deposited in a certain number of crystals, which, including an additional *analog* conversion through scintillation, will be converted to an electrical signal by Avalanche-Photodiodes (APDs).

**Convert the electrical signal to a digital signal.** This is what is called the analog-to-digital conversion and will be explained in detail in this section. As computing architectures work with the binary system (the states can only be 0 or 1), they are unable to deal with analog data, thus it must be converted. The conversion is inherently lossy, as it is discrete in time and amplitude.

**Process the digital signal.** Once a digital representation of the measured quantity is made available, it can be processed by a computer. Of course this processing is not limited to the architecture of a classical personal computer (i.e. CPU), but may be performed by architectures like micro-controllers, Field Programmable Gate Arrays (FPGAs), Application-Specific Integrated Circuits (ASICs), Graphical Processing Units, etc.

## 1.2. Principle of an ADC

The principle of converting an analog value to its digital representation is illustrated in Figures 1.1 and 1.2.

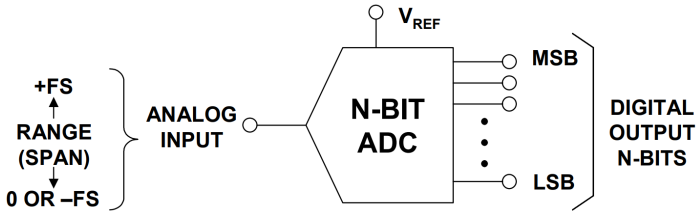



Figure 1.1.: Principle of an Analog-to-Digital Converter. [1]

On the left-hand side, the analog value is represented as a range from 0 (or $-FS$) to $+FS$. $FS$ stands for full scale and refers to the maximum voltage amplitude that can be handled by the circuit. In this example, we assume that the Full Scale Range (FSR) is a voltage of 10 V. An ADC can be devised to work with either unipolar (e.g. 0 V to 10 V) or bipolar (e.g. $-5$ V to 5 V) signals. Once the ADC with its fixed input range is chosen, the analog signal might have to be attenuated or amplified to match the input range.

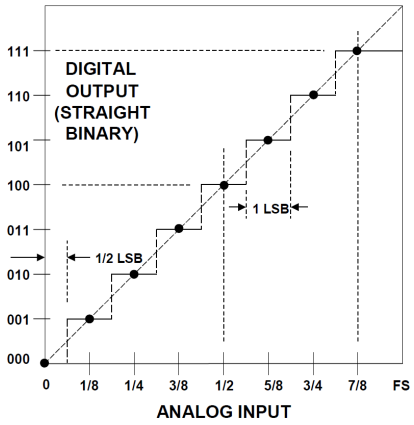
On the right-hand side, the digital signal is represented as N-bit, ranging from the Least Significant Bit (LSB) to Most Significant Bit (MSB). The number of bits determines the *vertical* resolution: the ADC can determine whether the amplitude of the analog signal is within one of $2^N$ intervals. Figure 1.2 illustrates this with 4-bit resolution and a FSR of 10 V: The vertical resolution is

$$\frac{10\,\mathrm{V}}{2^4\,\mathrm{div}} = 0.625\,\mathrm{V/div},$$

therefore every analog value between 0 V to 0.625 V will be assigned the binary value $0000_2$, every value between 0.625 V to 1.250 V will be assigned $0001_2$, and so on. In practice, there is usually an offset of $-0.5$ LSB, such that the 0 V level is in the middle (instead of the beginning) of the lowest *bin*.

| BASE 10 NUMBER | SCALE | +10V FS | BINARY | GRAY |
|---|---|---|---|---|
| +15 | +FS – 1LSB = +15/16 FS | 9.375 | 1 1 1 1 | 1 0 0 0 |
| +14 | +7/8 FS | 8.750 | 1 1 1 0 | 1 0 0 1 |
| +13 | +13/16 FS | 8.125 | 1 1 0 1 | 1 0 1 1 |
| +12 | +3/4 FS | 7.500 | 1 1 0 0 | 1 0 1 0 |
| +11 | +11/16 FS | 6.875 | 1 0 1 1 | 1 1 1 0 |
| +10 | +5/8 FS | 6.250 | 1 0 1 0 | 1 1 1 1 |
| +9 | +9/16 FS | 5.625 | 1 0 0 1 | 1 1 0 1 |
| +8 | +1/2 FS | 5.000 | 1 0 0 0 | 1 1 0 0 |
| +7 | +7/16 FS | 4.375 | 0 1 1 1 | 0 1 0 0 |
| +6 | +3/8 FS | 3.750 | 0 1 1 0 | 0 1 0 1 |
| +5 | +5/16 FS | 3.125 | 0 1 0 1 | 0 1 1 1 |
| +4 | +1/4 FS | 2.500 | 0 1 0 0 | 0 1 1 0 |
| +3 | +3/16 FS | 1.875 | 0 0 1 1 | 0 0 1 0 |
| +2 | +1/8 FS | 1.250 | 0 0 1 0 | 0 0 1 1 |
| +1 | 1LSB = +1/16 FS | 0.625 | 0 0 0 1 | 0 0 0 1 |
| 0 | 0 | 0.000 | 0 0 0 0 | 0 0 0 0 |

(a) Table with analog and corresponding digital values for a 4 bit ADC with a FSR of 0 V to 10 V. The Gray code is a modified binary code, where only one bit changes between transitions.




(b) Ideal dependence of digital and analog value for a 3-bit ADC.

Figure 1.2.: Example lookup table and graph for an ADC with −0.5 LSB offset. [1]

## 1.3. ADC Architectures

Quite a number of ADC architectures have made their way into today's applications. Five of the most common ones will be introduced in this section. The type of architecture is rarely chosen consciously by the application designer/engineer. In a typical application design flow, a parametric search based on the required sampling rate, (effective) resolution, size, power consumption, etc. usually yields a suitable solution without requiring knowledge about the *inside* of the ADC.

When it becomes necessary to optimize the system with respect to noise sources, the mechanisms leading to noise contributions have to be understood, and they are different for each architecture. Furthermore, each has its tradeoffs, which have to be considered as soon as the application falls into a regime that can be served by more than one architecture.

In Figure 1.3 the applications for ADCs are divided into four groups: Industrial Measurement, Voiceband/Audio, Data Acquisition, and High Speed; this thesis falls in the latter two categories. The three most commonly used ADC architectures are written in the respective ellipses. A fourth architecture is not mentioned: the bare Flash Analog-to-Digital Converter (FADC), which is the simplest one. It lies in the *Very High Speed* domain, beyond the limits of the scales: above 1 GHz and below 8 bit. All four architectures, as well as the outdated *Counting and Integrating ADC*, will be discussed in the following sections.

### 1.3.1. Flash ADC

FADCs are the most basic (and earliest Complementary Metal Oxide Semiconductor (CMOS)) type of ADC. It is still used today as a core digitizer in pipelined ADCs (cf. section 1.3.2), or in applications which require highest sampling rates (above 1 GHz).

The basic element of any modern ADC is a comparator, which itself can already be understood as a 1-bit ADC. It will be introduced in the next paragraph, followed by its implementation in the FADC architecture.

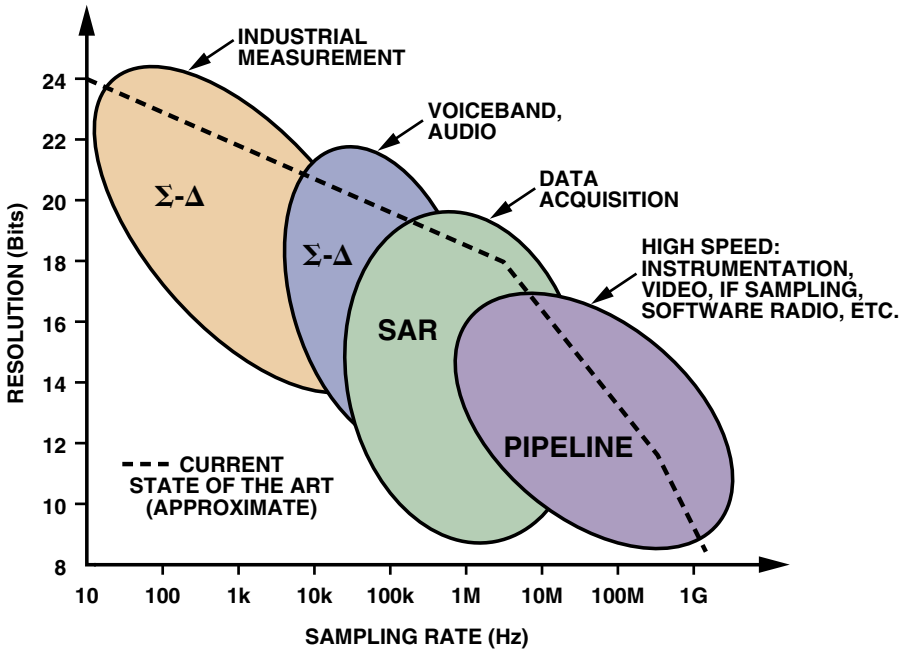



Figure 1.3.: ADC architectures and their field of usage, arranged by sampling rate and resolution: Sigma-Delta, Successive Approximation, Pipeline. Note that the *current state of the art* line refers to 2005. [2]

**Comparator** A bare comparator has two analog inputs and one digital (logic) output (see Fig. 1.4a). The two analog inputs are compared (hence the name); it is evaluated whether the difference $V_+ - V_-$ has a positive or negative value, and the state of the output reflects the outcome of the comparison with either a logic 1 or 0, respectively. In conclusion, a comparator can be regarded as a 1-bit ADC.

As an example, imagine the negative input being connected to a voltage of $V_- = 5\,\mathrm{V}$. If the positive input is $V_+ < 5\,\mathrm{V}$, the state of the logic output will be $V_{\mathrm{logic}} = 0$, if $V_+ > 5\,\mathrm{V}$, the state of the logic output will be $V_{\mathrm{logic}} = 1$. Comparing this with the table in Fig. 1.2a, this behavior is reflected in the MSB of the binary representation of the 4-bit ADC.

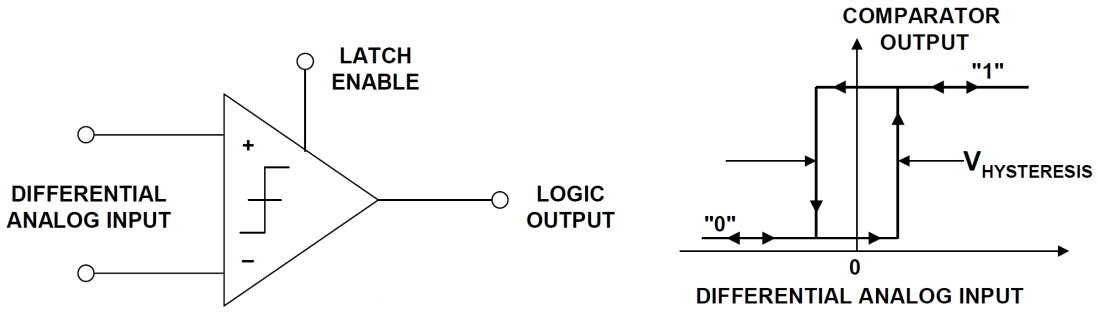
Once one enters the real (noisy) world, this ideal behavior is impaired: ff, in this example, the positive input would be $V_+ \approx 5\,\mathrm{V}$, the comparator would fluctuate between the two logic states due to electronic noise. This effect is called *transition noise* (cf. section 1.4.4). To compensate this to some degree, a comparator can have a fixed inherent[2] hysteresis behavior (see Fig. 1.4b). The logic state is switched from $V_{\mathrm{logic}} = 0 \rightarrow 1$ only if the positive input surpasses the negative input by at least $V_+ - V_- > \frac{V_{\mathrm{Hysteresis}}}{2} \neq 0$, and then switches back from $1 \rightarrow 0$ with the respective behavior, leaving up to $V_{\mathrm{Hysteresis}}$ allowed headroom for peak-to-peak noise on the input voltage difference $V_+ - V_-$, without resulting transition noise in $V_{\mathrm{logic}}$.

Further considerations have to be made regarding the digital logic that will process the

[2]For a hysteresis-free comparator, (variable) hysteresis can be achieved with a simple feedback resistor circuit.

comparator output. The state of the comparator must stay stable during this processing, which may be achieved using a built-in latch. If the latch signal is activated, $V_{\text{logic}}$ will be updated with the current value, and keep this value until the next latch activation. In between, changes in $V_+$ and $V_-$ will have no influence on $V_{\text{logic}}$.

The usage of comparators in an FADC will be discussed in the following paragraph.




(a) Schematic symbol of a comparator with analog inputs $V_+$ and $V_-$ and digital output $V_{\text{logic}}$.

(b) Hysteresis slope of a comparator.

Figure 1.4.: A comparator and its hysteresis slope. [1]

**Flash ADC** To extend the previously mentioned 1-bit ADC, consisting of just one comparator, to N-bit, one needs $2^N - 1$ comparators in total. This is shown in Fig. 1.5 as an example for a 3-bit FADC with seven comparators.

A resistor network is used to apply suitable thresholds to the comparators, such that the full scale is equally divided by $2^N$. Assume that a FSR voltage of 8 V is applied to the circuit as $V_{\text{REF}}$. Eight equal resistors of e.g. 1 kΩ limit the current flow to GND to 1 mA, with a voltage drop of 1 V on each resistor. Thus, the first comparator will switch from $0 \rightarrow 1$ at an analog input voltage of 1 V, the second at 2 V, and so on, which leads to a $2^N - 1$-bit thermometer code output. After a short time (allowing the comparator outputs to settle) the thermometer code is strobed to the priority encoder (using the aforementioned latch mechanism), which converts it to a 3-bit binary code.

The Flash ADC architecture allows very high speed conversion, as it is limited only by the switching speed of the comparators and the priority encoder. Transition times of less than 1 ns allow sampling rates in the GHz regime, unmatched by any other architecture. Since the number of required comparators increases exponentially with the number of bits, vertical resolutions of more than 8 bit (requiring 255 comparators) are rarely realized as cost, power consumption, and input capacitance scale accordingly (the latter limiting the input bandwidth).

At the time of writing this thesis, a (not exhaustive) search revealed the fastest available FADC has a sampling rate of 26 GS/s with 3-bit resolution (Analog Devices HmcAd5831Lp9Be). Very high sampling rates can also be achieved by interleaving multiple FADCs with the help of phase shifted strobe signals: if for example two FADCs are used, both digitize the same input signal, but the strobe signal of the second FADC is issued exactly one half cycle after the strobe signal of the first FADC.

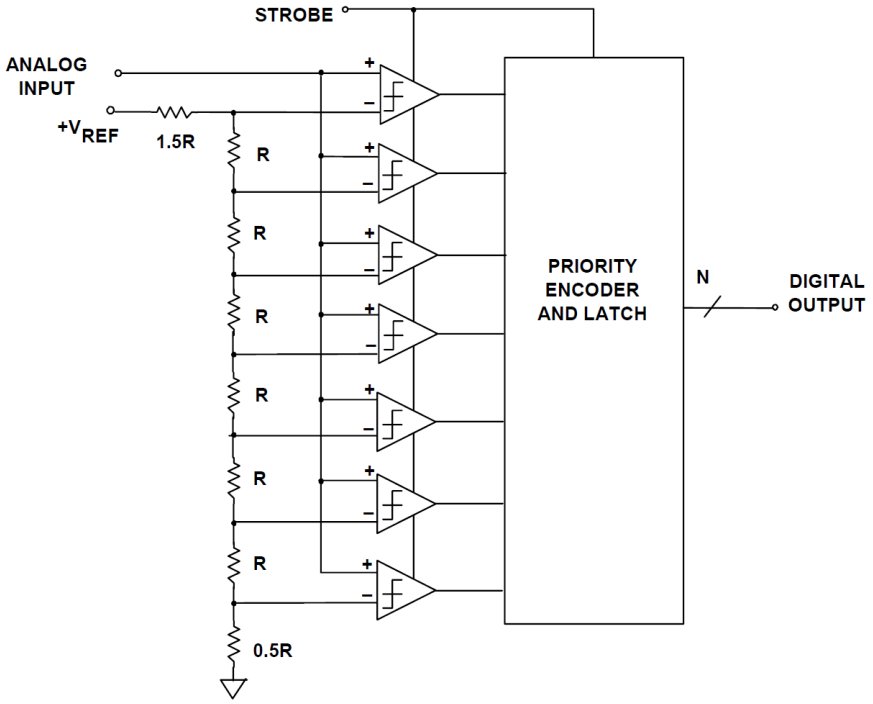



Figure 1.5.: A 3-bit FADC with resistor array and priority encoder for conversion from thermometer to binary code. Opposed to the explanation in the text, this circuit exhibits a $-0.5$ LSB shift of the scale, visible at the dimensioning of the first and last resistor. [1]

### 1.3.2. Pipelined ADC

To overcome the limitation of the $2^N$ scaling of comparators in FADCs, the Sub-Ranging ADC (SRA) has been devised in the 1950s. Basically, instead of one digitization stage with the full number of bits, it divides the digitization to multiple smaller units with less bits, but at the cost of speed. This architecture paved the way for the development of the pipelined architecture in the 1980s, which today is the architecture of choice for sampling rates of 10 MS/s up to 10 GS/s. The stages of the SRA are separated by synchronous elements, such that they behave like shift registers, allowing a higher processing speed, thus a higher sampling rate. Both architectures will be introduce in the following paragraphs.

**Subranging ADC** The basic principle behind the subranging architecture will be explained briefly with help of Figure 1.6, showing an example of a 6-bit SRA. The analog input signal amplitude is temporarily *stored* for digitization using capacitors; this mechanism is called Sample and Hold Amplifier (SHA). The SHA will stay in the *hold* state for one sampling clock cycle, which in return must not be shorter than the time needed by the remaining circuit to finish the digitization. Therefore the goal of the SHA is similar to that of the latch in the output path of the comparator.

A coarse digitization with a first 3-bit sub-ADC (usually FADC) provides the three MSB of the 6-bit digital value. This 3-bit value is converted back to an analog signal by a 3-

bit sub-Digital-to-Analog Converter (DAC) and subsequently subtracted from the original signal, leaving a residual signal, in amplitude smaller than the LSB of the first 3-bit stage. This residual signal is amplified with a gain of $G = 2^{N1} = 2^3 = 8$, and digitized again by a second 3-bit sub-ADC, providing the three LSB of the total 6 bit. After completion of both digitizations, the next sampling clock cycle may be asserted, releasing the hold-state of the SHA and strobing the output registers. The circuit is ready for the next conversion cycle.

To summarize: instead of using one 6-bit FADC, two 3-bit sub-ADCs are combined, which reduces the number of comparators from $2^6 - 1 = 63$ to $2 \cdot (2^3 - 1) = 14$. In general, for a N-bit architecture, a two-stage subranging ADC will required $2^N - 2^{\frac{N}{2}+1} + 1$ less comparators.

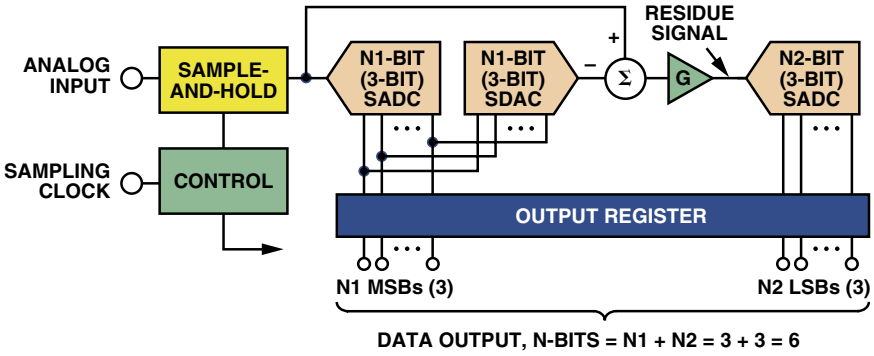
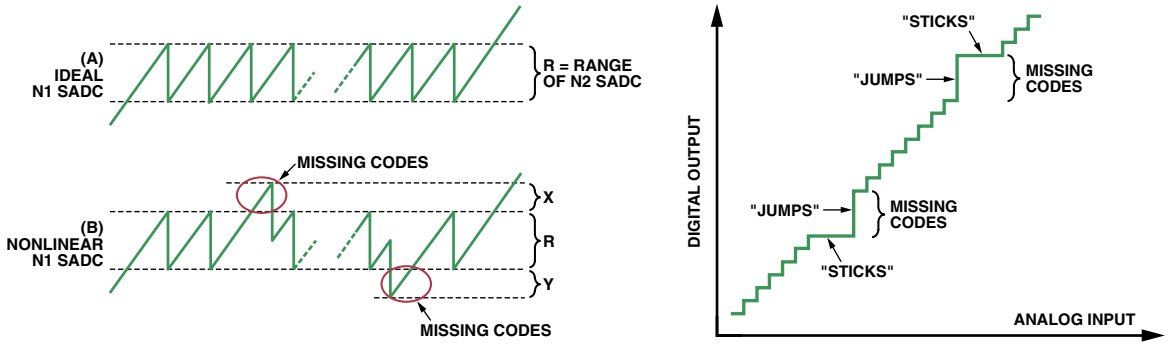



Figure 1.6.: A 6-bit subranging ADC, consisting of two 3-bit sub-ADCs (SADC) and one 3-bit sub-DAC (SDAC), with a subtracter (Σ) followed by an amplifier (G). [2]

**Error-Corrected Subranging ADC** An ideal SRA requires matching between sub-ADC and sub-DAC. This is feasible up to a resolution of 8 bit [2] in a two-stage SRA – after this point, non-linearities and imperfect gain matching become apparent, mainly due to temperature-dependent variations. In such a case it may happen that the residual signal is no longer within the range of the N2 sub-ADC at all times. The consequences are shown in Fig. 1.7. Figure 1.7a shows the residual signal as seen by the second FADC. With non-ideal matching, it can be out of its range $R$. This leads to the so-called *missing codes* (shown in the plot in Fig. 1.7b): the N2 FADC will be stuck, since values will be digitized as $111_2$ or $000_2$ as long as they overflow/underflow in the regions $X$ and $Y$, respectively.

As a perfect matching is not possible, this behavior can only be accounted for by extending the range of the second sub-ADC by 1 bit, together with applying an offset to the residual signal, such that there exists no region $Y$ below the range $R$ (cf. Fig. 1.7a). This is depicted in the scheme of a 6-bit error-corrected subranging ADC (devised in the mid 1960s) in Fig. 1.8. The MSB of the now 4-bit second sub-ADC is added to the 3 bit of the first sub-ADC.

**Pipelined ADC** The SRA has the advantage over the FADC of achieving higher resolution without the exponential demand for comparators. But it comes at a price: the processing speed is lower due to the additional time required by the DAC-subtraction mechanism, the second stage, and the error correction; not speaking of the additional time required if even more stages are implemented. This problem has been relieved with the invention of the pipelined architecture, which is depicted for a three (or more) stage SRA in Figure 1.9.




(a) Residual signal at the N2 sub-ADC in case of ideal and non-ideal matching of the N1 sub-ADC and sub-DAC.

(b) Consequences of missing codes in in the analog-to-digital conversion.

Figure 1.7.: Visualization of *missing code* errors in a non-error-corrected SRA, if a ramp signal is applied to the input. [2]

The stages, consisting of sub-ADC, sub-DAC, and subtraction circuit, are separated by a Track-and-Hold (T/H)-circuit, working in a similar manner as the SHA: the interim result is forwarded to the next stage and temporarily stored until the arrival of a sampling clock pulse. Thus, with every clock cycle, all interim results propagate to the respective following stages of the circuit, until after S+1 clock cycles a signal has passed all S stages (additional clock cycles are needed for the error correction logic as presented previously). This pipeline allows all stages and the error correction to be active at the same time, each working on a different sampling point and producing a digital output value with each clock cycle. This allows the pipeline architecture to operate at highest sampling rates, limited by the slowest stage in the chain (instead of the signal propagation through the whole chain).

The effect of the delay is shown in Figure 1.10: the digital output value reflects the analog input after 7 clock cycles (for the ADC LTM9009-14, used in this thesis, the delay is 6 clock cycles of 12.5 ns, a total of 75 ns).

In general, it is not possible to predict the underlying structure from the number of bits and the pipeline delay (which is stated in the datasheet of the ADC), as there are many degrees of freedom in choosing the number of stages, the bits per stage, the correction bits per stage, and the internal layout and delay of the error correction logic.

Although this approach has many advantages, there are two issues that the application engineer should be aware of:

1. The pipeline delay itself can be problematic, for example in applications where the ADC is part of a feedback loop, or timing is otherwise critical.

2. Not only is there an upper limit of the sampling rate, but also a lower limit due to an excessive voltage droop in the T/H circuit, which increases with the hold-time. This also makes pipeline ADCs unsuitable for one-shot applications (as opposed to continuous sampling applications).

A completely different approach, the Successive Approximation ADC, will be presented in the next section.

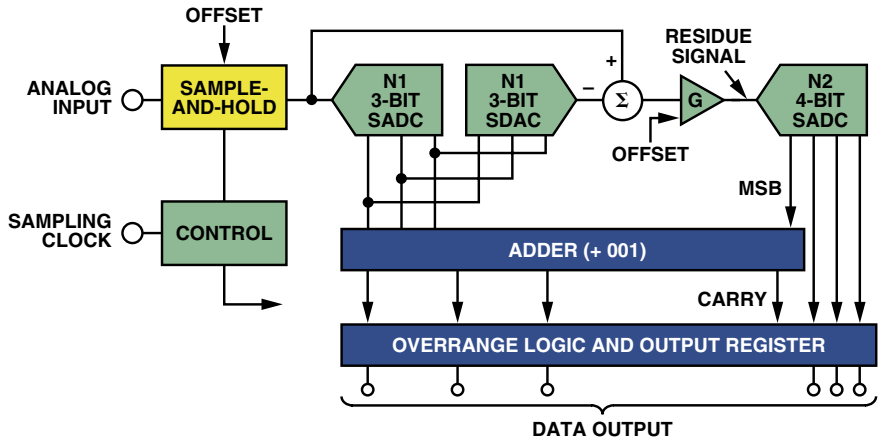



Figure 1.8.: Error-corrected subranging ADC. Compared to the basic subranging ADC, an offset is added to the residual signal and the range of the second sub-ADC is extended. That way a non-ideal matching can be compensated with the help of an adder and overrange logic. [2]

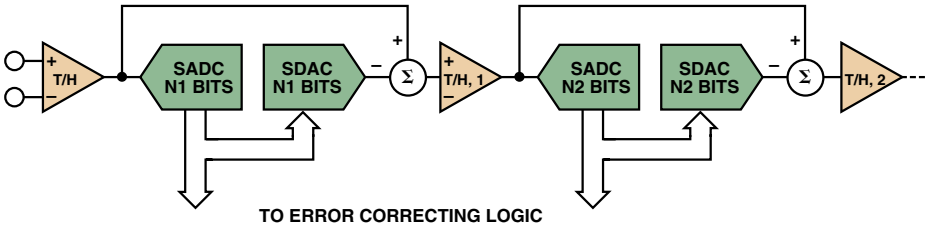



Figure 1.9.: Scheme of a 2+ stage pipeline architecture, with T/H circuits to separate the stages. The error correction is not depicted, but is similar to Figure 1.8. [2]

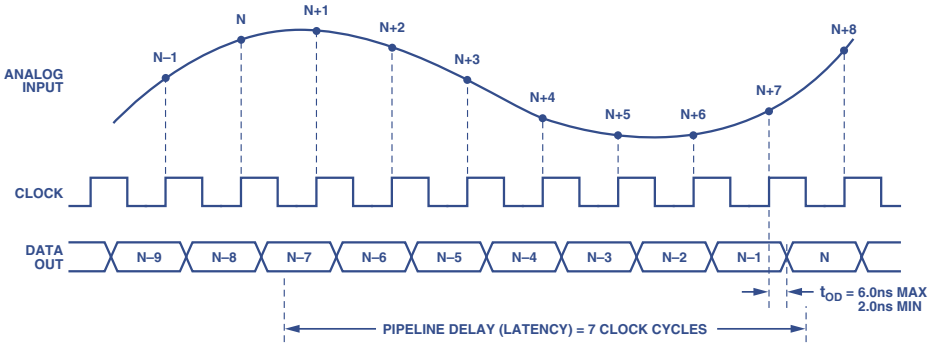



Figure 1.10.: Timing scheme of an ADC with pipeline architecture. The digital data is delayed by 7 clock cycles. [2]

### 1.3.3. Successive Approximation ADC

Although it has no distinct relevance for this thesis, the Successive-Approximation Register (SAR) ADC architecture deserves introduction in this section, as it is by far the most popular architecture in data acquisition. Its resolution is up to 20 bit, generally giving it a higher accuracy than the pipeline architecture, but at lower typical sampling rates of up to around 20 MS/s.

A block diagram of the architecture is shown in Figure 1.11. Like with other types, a Sample and Hold Amplifier (SHA) keeps the analog signal stable for the time of the conversion. With its single comparator, it has the characteristics of a 1-bit FADC; but at the lower input of the comparator, a DAC is placed instead of a fixed voltage source. The output voltage of the DAC is controlled by the successive-approximation register that is the name giver of this architecture. A dedicated timing logic allows not only continuous sampling, but also single-shot or burst modes.

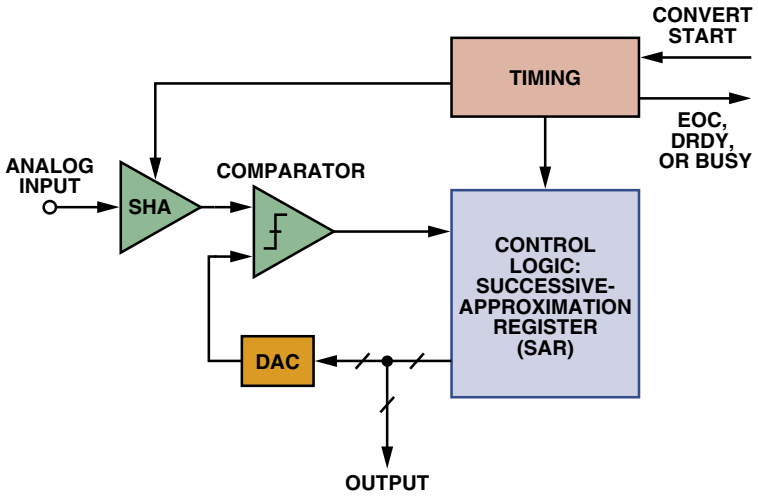



Figure 1.11.: Scheme of the SAR ADC architecture. The conversion has to be started with an external signal, and the SAR ADC signalizes after the approximation process has finished. [2]

The algorithm of the successive approximation works as follows: the DAC is set starting with a half-scale output voltage, which allows the comparator to determine the MSB of the analog input. Afterwards, the output of the DAC is set to either $\frac{1}{4}$ or $\frac{3}{4}$ of the full scale (depending on the MSB), and the next bit is determined, and so forth. This *divide et impera* algorithm is explained with an example in Figure 1.12.

It is evident that the accuracy and linearity of this architecture mostly depend on the characteristics of the DAC. With CMOS technology, precise switched-capacitor DACs are feasible to produce, granting a high degree of temperature stability; at the same time, the monolithic integration of more complex (timing) control logic and even multiplexers with analog switches are possible.

To overcome the resolution limit of the DAC, the $\Sigma$-$\Delta$ ADC architecture can be used, which will be explained in the next section.

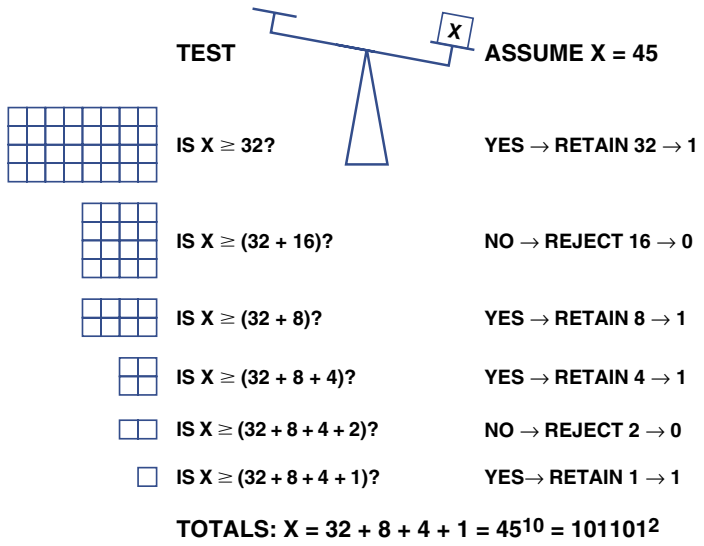



Figure 1.12.: Example of the algorithm of a 6-bit SAR ADC. [2] (corrected)

### 1.3.4. Sigma-Delta ADC

The Sigma-Delta ADC (Σ-Δ ADC) architecture is also not used directly within the scope of the SADC design presented in this thesis, but some of its techniques are emulated or adapted to improve the performance of the digital signal processing.

The architecture itself is dominant in high resolution applications (16 bit to 24 bit) with low bandwidth (up to 1 MHz), which comes down to mostly audio and industrial measurements. Although the idea came up already in the 1950s, it became widely available only in the late 1980s, when the implementation of digital filters in CMOS technology was market-ready.

The basic concept of this architecture consist of four steps: oversampling, noise shaping, digital filtering, and decimation. The effects of those steps is illustrated in Figure 1.13.

Figure 1.13 **A** shows the reference situation: an analog signal, digitized with an ADC working in continuous mode at a sampling frequency $f_S$. The rectangular area represents the signal, which is influenced by quantization noise, equally distributed over the whole bandwidth (green color). It is assumed that the analog signal is constrained through suitable filters to have a signal bandwidth of $\frac{f_S}{2}$ according to the Shannon-Nyquist sampling theorem (cf. section 1.4.6). In Figure 1.13 **B**, the signal is oversampled[3] by a factor of K (called the oversampling ratio). The quantization noise is still equally distributed, but now in a wider band, leading to a lower noise density per bandwidth. The noise is then removed by a digital filter in the frequencies above the signal bandwidth $\frac{f_S}{2}$. Afterwards, the sampling rate is digitally reduced to $f_S$. The Signal-to-Noise ratio (SNR) is increased by 3 dB with every doubling of K. For further explanation see also section 1.4.3, especially equation 1.17. Lastly, in Figure 1.13 **C**, the Σ-Δ modulator is introduced. The quantization noise is shaped in such a way by the modulator that a larger part lies within the region that is cut off by the digital filter.

[3] Oversampling refers to running an ADC with a higher sampling rate than required by the Shannon-Nyquist sampling theorem, e.g. if the signal bandwidth is 10 MHz, the sampling rate would be 20 MS/s, and an oversampling by a factor of $K = 4$ would lead to a sampling rate of 80 MS/s.

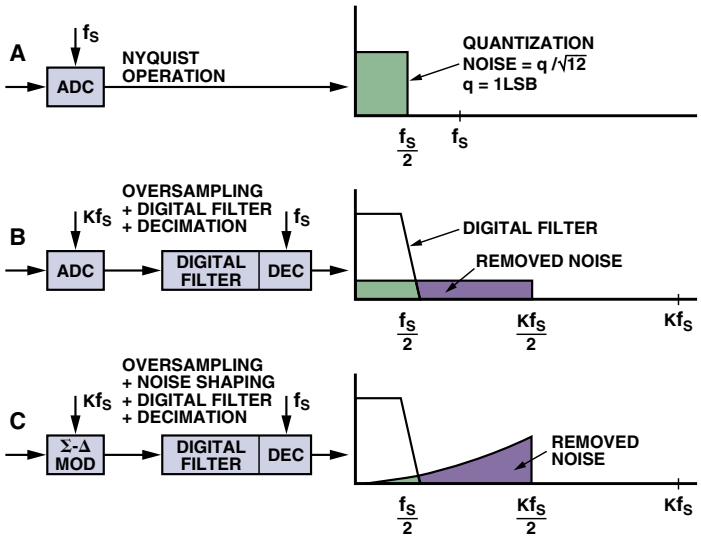



Figure 1.13.: Effects of oversampling, digital filtering, decimation, noise shaping on the noise spectrum of a Σ-Δ ADC. $f_S$ is the sampling rate, $K$ the oversampling ratio. [2]

The scheme of the modulator is shown in Figure 1.14. The modulator's integrator acts as a low-pass for the signal, but the following 1-bit comparator's quantization noise is effectively high-pass filtered due to the negative feedback (see [4] and section 2.3 of [5]). The modulator now generates a 1-bit data stream, with the density of logic 1 being proportional to the analog signal's amplitude). As the ADC and DAC in the modulator have a resolution of only 1 bit, they posses excellent differential linearity. Through the digital low-pass filter and the decimator, the resolution is increased from 1 bit to N bit, and the sampling rate is reduced. The influence of the quantization noise is greatly decreased, and can even be further reduced by implementing more integrators within the modulator circuit. This becomes increasingly difficult with three or more orders, as the phase shift introduced by the integrators reduces the stability of the circuit.

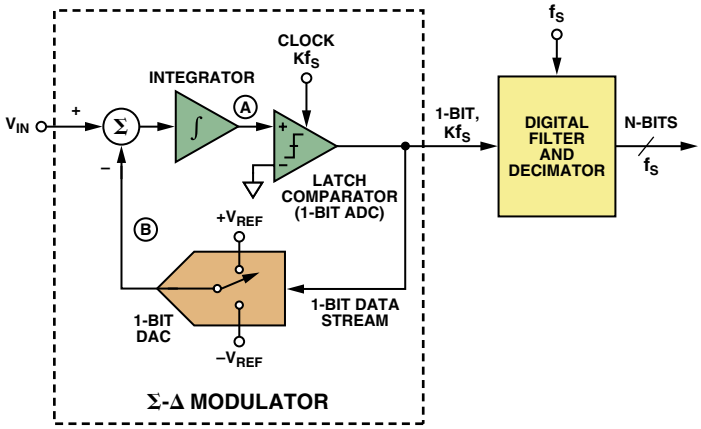



Figure 1.14.: Scheme of a first order Σ-Δ ADC with bipolar input. [2]

### 1.3.5. Counting and Integrating ADC

Counting and integrating ADCs, depending on the exact type also called single-slope, dual-slope, multi-slope, or Wilkinson ADC, are still used in modern measurement equipment like digital multimeters (DMM), as they provide a low-cost solution for high precision measurements in the regime below 1 kHz. The functionality will be explained with the help of Figure 1.15 for the case of a dual-slope integrating ADC: a capacitor is charged with a current proportional to the input signal $V_{\mathrm{in}}$ for a fixed integration time $T$. The capacitor is then disconnected from the input signal and connected to a reference potential (e.g. GND) such that it discharges linearly within the time $t_{\mathrm{x}}$. To measure the time, a clock signal is allowed to pass to a counter for the duration of $t_{\mathrm{x}}$. In summary, the number of counted clock pulses is proportional to the input voltage. The integrating character of this design is beneficial for averaging a slowly changing input signal. Furthermore, if the application allows to choose the integration time suitably long, line rejection of 50 Hz/60 Hz can be achieved.

Multi-Slope implementations of this architecture speed up the ramping time by using differently scaled currents to discharge the capacitor. Furthermore, reference voltages with opposing potentials may be used to compensate overshoot effects after fast ramping.

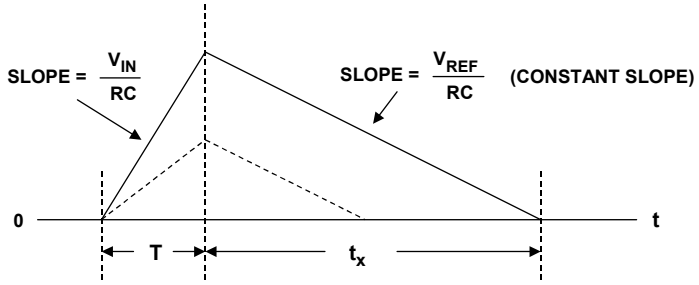



Figure 1.15.: Timing diagram of a measurement cycle in a dual-slope ADC for two different input voltages. [1]

## 1.4. ADC Performance

In this section, the most relevant sources of performance limitations in analog-to-digital conversion and analog signal processing are discussed. Naturally the focus is on parameters that are relevant for later discussion. For example, some parameters to characterize ADCs are very important for Software Defined Radio applications, where narrow carrier frequencies have to be separated from each other. Yet, for the digitization of a slow, broadband waveform as it will be the case in this thesis, they are not relevant. Hence, parameters like THD, SINAD, ENOB, SFDR, NPR[4], will not be covered in this thesis. A broader coverage including those topics can be found in [1].

Figure 1.16 shows a list of sources for noise, errors, and limitations, categorized into **Buffer**, **SHA**, and **Encoder**. The buffer, which is not necessarily part of the ADC, may also be representative for the noise introduced from the analog signal chain in front of the ADC. As

[4] Total Harmonic Distortion, Signal-to-Interference ratio including Noise and Distortion, Effective Number Of Bits, Spurious-Free Dynamic Range, Noise Power Ratio

a main source, the Johnson-Nyquist noise will be discussed in section 1.4.1, complemented by section 1.4.2 describing the characteristic 1/f noise that is dominant for low frequencies [6]. The quantization noise, an error due to the discrete nature (and thus a finite resolution) of an ADC, limits the maximum theoretical resolution of an ideal ADC, and will be discussed in section 1.4.3. Section 1.4.4 will summarize noise sources in ADCs and show the effects on the conversion and the obtainable SNR. The effect of jitter in the digitization clock on the amplitude resolution will be deduced in section 1.4.5. Lastly, section 1.4.6 will explain the effect of *Aliasing* in the digitization.

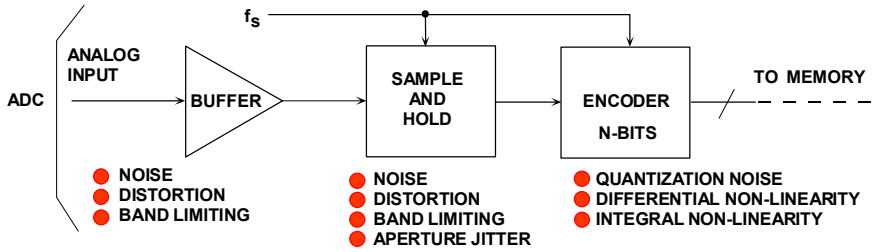



Figure 1.16.: Sources of noise, errors, and limitations in ADC circuits. [1]

### 1.4.1. Johnson–Nyquist Noise

Thermal excitation of charge carriers leads to a fluctuation of their velocity, resulting in the so-called *Johnson-Nyquist noise* (also called *Johnson noise* or *thermal noise*). The formula can be deduced from Planck's black-body-radiation formula: the spectral density of the power radiated by a black body in equilibrium is

$$\frac{dP}{df} = \frac{hf}{e^{hf/k_\mathrm{B}T} - 1} \tag{1.1}$$

where $k_\mathrm{B}$ is Boltzmann's constant and $h$ is Planck's constant. At room temperature, $k_\mathrm{B}T/h \approx 6 \times 10^{12}\,\mathrm{Hz}$, so the exponent is $\ll 1$ for all interesting frequencies and a series expansion yields

$$\frac{dP}{df} \approx \frac{hf}{1 + hf/k_\mathrm{B}T - 1} = k_\mathrm{B}T \tag{1.2}$$

The noise is distributed equally over the frequency spectrum, and it has a constant power spectral density (which is commonly referred to as *white noise*). Further deductions (section 3.4 in [6]), show that the noise voltage caused by a resistor with an impedance of $R$ is

$$\frac{dv_n^2}{df} = 4k_\mathrm{B}T \cdot R, \tag{1.3}$$

taking into account that in an electrical circuit the highest power transfer is reached when the source and load impedance are equal (the principle behind impedance matching).

For an RC circuit, as used in high-pass and low-pass filters, the contribution of the resistor cancels out due to the resulting filter's contribution, and the power spectral density scales with the inverse capacity:

$$\frac{dv_n^2}{df} = 4k_\mathrm{B}T \cdot \frac{1}{C}. \tag{1.4}$$

The total noise contributions within a bandwidth of $f_2 - f_1 = \Delta f$ are, respectively

$$v_n = \left(\int_{f_1}^{f_2} 4k_\mathrm{B}T \cdot R df\right)^{\frac{1}{2}} \quad \text{and} \quad \left(\int_{f_1}^{f_2} 4k_\mathrm{B}T \cdot \frac{1}{C} df\right)^{\frac{1}{2}} \tag{1.5}$$

$$= \sqrt{4k_\mathrm{B}T \cdot R \cdot \Delta f} \quad \text{and} \quad \sqrt{4k_\mathrm{B}T \cdot C^{-1} \cdot \Delta f} \tag{1.6}$$

### 1.4.2. 1/f Noise

While Johnson-Nyquist noise results from random fluctuations, 1/f noise stems from charges being affected by processes with specific time constants $\tau$. This can be, for example, a charge trap state in a semiconductor. Such processes, seen individually, are not always clearly understood regarding their origin and modeling [7]. Summed up they result in a 1/f behavior of the noise spectral density (sometimes also modeled as $1/\mathrm{f}^\alpha$ with $0.5 < \alpha < 2$). This makes it relevant only for low frequencies, as it will fall with $3\,\mathrm{dB\,oct}^{-1}$ below the Johnson-Nyquist noise floor. Figure 1.17 shows this characteristic 1/f corner.

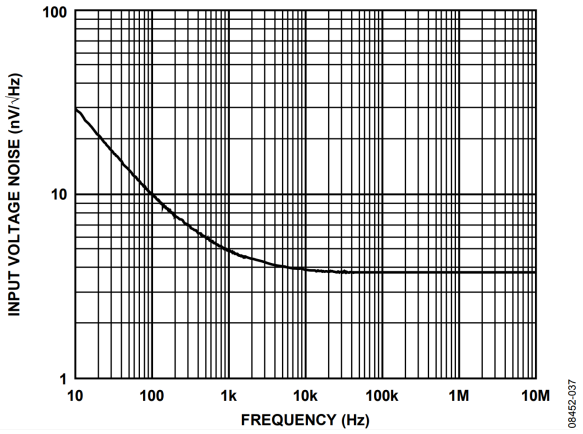
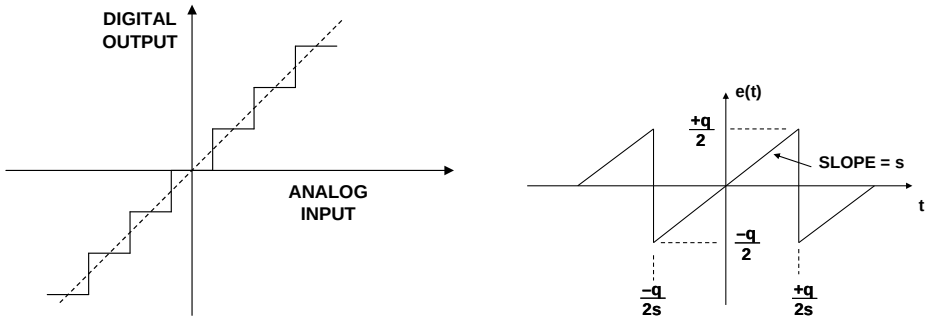



Figure 1.17.: Voltage noise spectral density of the Analog Devices Ada4940-2, showing the 1/f corner at roughly 1 kHz. [8]

### 1.4.3. Quantization Noise

An ADC converts a signal from a continuous to a discrete spectrum, which can only happen with finite accuracy. The error due to this quantization is called *quantization noise*. In Fig. 1.18 the error is illustrated with the help of an ADC's transfer function. Fig. 1.18a shows the transfer function of an ideal (quantized) ADC, together with the (dashed) correlation in a non-quantized system. Fig. 1.18b shows the difference of both, where $q$ is the quantization step and $s$ the arbitrary slope. Within the (repeating) window of $[\frac{-q}{2s}; \frac{q}{2s}]$, the function can be written as $e(t) = s \cdot t$, leading to an amplitude of up to $\frac{\pm q}{2}$. The frequency components of this sawtooth waveform have infinite bandwidth, but the harmonics are folded back to the Nyquist band (DC to $f_\mathrm{S}/2$) due to aliasing (cf. section 1.4.6). The Root Mean Square (RMS)




(a) Ideal transfer function with quantization steps.

(b) Quantization error (Input-Output of (a)).

Figure 1.18.: Effects of quantization. The transfer function, which ideally is linear, has discrete steps, leading to a deviation in the form of a sawtooth. [1]

quantization noise can now be calculated:

$$v_{\mathrm{QN,rms}} = \sqrt{\overline{e^2(t)}} \tag{1.7}$$

$$= \sqrt{\frac{s}{q}\int_{-q/2s}^{q/2s}(s\cdot t)^2 dt} \tag{1.8}$$

$$= \frac{q}{\sqrt{12}} \tag{1.9}$$

This noise is spread uniformly over the whole Nyquist bandwidth, assuming that it is uncorrelated to the input signal, which is approximately true for the digitization of broadband signals. If, on the other hand, a (narrowband) signal and the sampling frequency are harmonically related, the noise will be concentrated in the harmonics of the signal. This is more complex to analyze (see for example [9]), and will not be covered in this thesis, as it is not applicable. Figure 1.19 shows the effect in case of a close harmonic relation ($f_\mathrm{S}/f_\mathrm{a} = 32$) of the narrowband signal ($f_\mathrm{a}$) and the sampling frequency ($f_\mathrm{S}$) on the left, where the concentration of the quantization noise in the harmonics is obvious. On the right, the relation is $f_\mathrm{s}/f_\mathrm{a} = 4096/127 \approx 32.252$, and the noise is more uniformly distributed.

Knowing the RMS quantization noise, the theoretically achievable SNR can be calculated. Suppose the input signal of an N-bit ADC with quantization step $q$ is a full amplitude sine wave with arbitrary frequency $f$ (uncorrelated to the sampling frequency):

$$v(t) = A\cdot\sin(2\pi f t) \tag{1.10}$$

$$v(t) = \frac{q\cdot 2^\mathrm{N}}{2}\cdot\sin(2\pi f t). \tag{1.11}$$

As the RMS value of a sine wave with amplitude $A$ is $\frac{A}{\sqrt{2}}$, one obtains

$$v_{\mathrm{signal,RMS}} = \frac{q\cdot 2^\mathrm{N}}{2\sqrt{2}}. \tag{1.12}$$

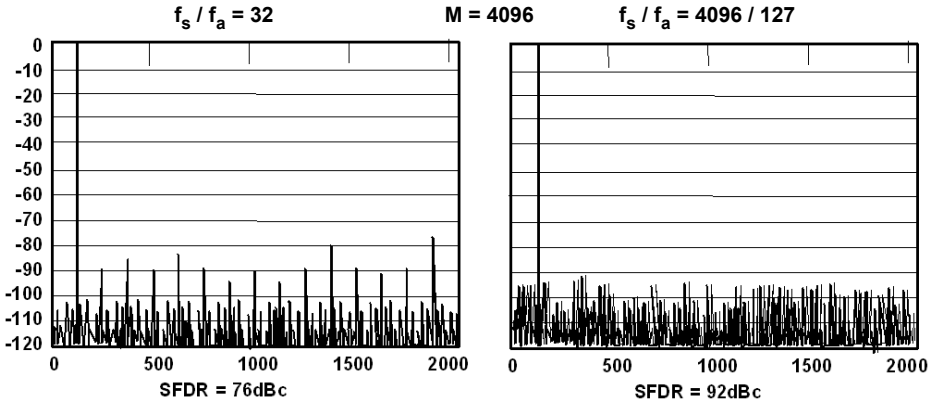



Figure 1.19.: Analysis of the Spurious-Free Dynamic Range, the ratio between the signal and the strongest noise or harmonic distortion (*spur*). The amplitude in dB is shown versus frequency (in arbitrary units). The data shows how the quantization noise affects the spectrum due to the correlation of a full amplitude narrowband signal related to the sampling clock. [1]

The theoretically achievable SNR, limited only by the quantization noise, can now defined as

$$\begin{aligned} \mathrm{SNR/dB} &= 20\log_{10}\left(\frac{v_{\mathrm{signal,RMS}}}{v_{\mathrm{noise,RMS}}}\right) && (1.13)\\ &= 20\log_{10}\left(\frac{q\cdot 2^{\mathrm{N}}}{2\sqrt{2}}/\frac{q}{\sqrt{12}}\right) && (1.14)\\ &= 20\log_{10}2^{\mathrm{N}} + 20\log_{10}\sqrt{\frac{3}{2}} && (1.15)\\ &= 6.02\mathrm{N} + 1.76. && (1.16) \end{aligned}$$

In conclusion, this formula describes the SNR of an ideal N-bit ADC for a full amplitude sine wave. In case of a broadband signal, this reduces to $\mathrm{SNR} = 6.02\mathrm{N}\,\mathrm{dB}$.

### Improving SNR - Process Gain

Depending on the application, this limit can be further improved by oversampling, as it was implied in section 1.3.4. Suppose the signal bandwidth is BW and the sampling frequency $f_{\mathrm{S}} = 2\cdot\mathrm{BW}$, fulfilling the Nyquist condition (see section 1.4.6). The quantization noise $\frac{q}{\sqrt{12}}$ will be completely contained in the sampled data due to aliasing. If the signal is now sampled at a higher sampling rate of $M\cdot f_{\mathrm{S}}$, the (more or less) uniformly distributed quantization noise will spread over the $M$-fold bandwidth and hence be reduced in amplitude by the same factor $M$. Removing the quantization noise outside of the signal bandwidth BW with digital filters (in case of infinitely high roll off) will increase the SNR by 3 dB for every doubling of $M$ [10]:

$$\mathrm{SNR/dB} = 6.02\mathrm{N} + 1.76 + 10\log_{10}\left(\frac{M\cdot f_{\mathrm{S}}}{2\cdot\mathrm{BW}}\right). \qquad (1.17)$$

### 1.4.4. Input Referred Noise

Quantization noise is usually not the limiting factor of a digitizing system. Instead, the theoretically possible SNR is degraded by intrinsic noise source, which appear like a noise source connected to the input of the ADC (hence called *input referred* noise). These noise sources include Johnson-Nyquist noise of the internal resistors and capacitors, but also the transition noise inherent in the FADC (a more detailed approach, treating the comparators' transition noise separately from the input referred noise, can be found in [11]). The effect can again be explained with the help of the transfer function. Figure 1.20 (left) shows how the discrete steps are smeared out by transition noise. This can be further smeared out by Differential Non-Linearity (DNL), occurring for example due to mismatching within internal resistor networks, shown as an overall effect on the right. To quantify the amount of noise, the

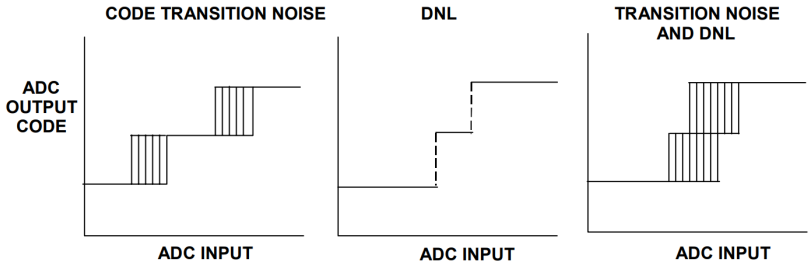



Figure 1.20.: Combination of transition noise and Differential Non-Linearity (DNL) in an ADC. The transition noise causes the quantization step width to smear out, while the DNL can lead to unequal step widths. [1]

ADC's input is connected to a stable, decoupled DC source (e.g. GND), or in case of an ADC with differential inputs, the inputs are shorted. In an ideal case, a single output code is visible, but due to the input referred noise, a Gaussian distribution is obtained (Figure 1.21a). Small deviations from the Gaussian distribution are due to the differential non-linearity. Larger deviations, or distributions that do not show a Gaussian distribution at all (Figure 1.21b), may indicate a poor design of the board layout.

The input referred noise can commonly be found in the datasheet of the digitizer in units of $\mathrm{LSB}_{\mathrm{RMS}}$. From this quantity, the *effective resolution* and the *noise-free code resolution*[5] of an ADC can be determined with

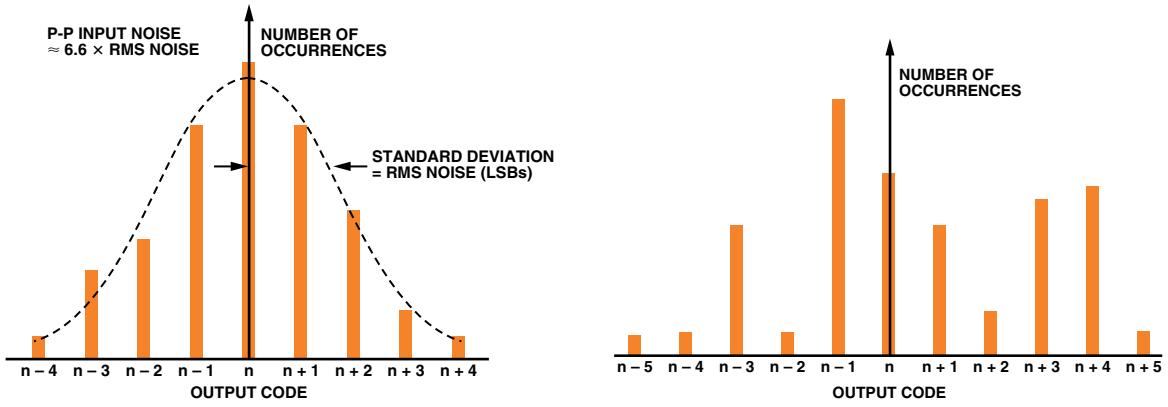
$$\text{Effective resolution/bit} = \log_2\left(\frac{2^{\mathrm{N}}}{\mathrm{noise}_{\mathrm{RMS}}}\right) \tag{1.18}$$

$$\text{Noise-free code resolution/bit} = \log_2\left(\frac{2^{\mathrm{N}}}{\mathrm{noise}_{\mathrm{pp}}}\right) \tag{1.19}$$

$$\approx \log_2\left(\frac{2^{\mathrm{N}}}{6.6 \cdot \mathrm{noise}_{\mathrm{RMS}}}\right) \tag{1.20}$$

$$\approx \text{Effective resolution} - 2.7 \tag{1.21}$$

[5] The conversion factor of 6.6 between RMS and peak-to-peak noise is a commonly used definition. At this value 99.9 % of all values of the Gaussian distribution are contained.




(a) Histogram with Gaussian distribution due to input referred noise, indicating also DNL effects.

(b) Histogram with non-Gaussian distribution, indicating more severe problems of the circuit.

Figure 1.21.: Histogram of ADC values with grounded/shorted input. An ideal (noise-free) ADC would show a single output code, N. [12]

where the noise is expressed in units of LSB [12]. While the noise-free code resolution describes how many counts (steps) the ADC can resolve distinctively (without any ambiguities due to noise), the effective resolution mostly has an advertising appeal. One has to pay attention which value is stated by the manufacturer.

### 1.4.5. Aperture Jitter

Most ADC architectures need an SHA or T/H circuit for temporary storage of the signal voltage on a capacitor until it is digitized/processed. This circuit is shown (simplified) in Fig. 1.22a. The *sample switch* connects the input signal to the *hold capacitor*, meaning that the capacitor voltage *tracks* the input signal. Timed by the sampling clock, the switch is opened, such that the capacitor will hold the current voltage for the time needed for digitization. The close-to-zero time required to open the switch is called aperture time. Fig. 1.22b shows the frequency-dependent effect of a jitter introduced in the aperture time:

For a signal with slope $\frac{dv}{dt}$, the jitter (uncertainty) in the time domain $\Delta t_{\mathrm{RMS}}$ will lead to an uncertainty in the signal amplitude of $\Delta v_{\mathrm{RMS}} = \frac{dv}{dt} \cdot \Delta t_{\mathrm{RMS}}$. To obtain the frequency dependency, suppose the input signal is a sine wave with frequency $f$:

$$v(t) = A \cdot \sin(2\pi f t) \tag{1.22}$$

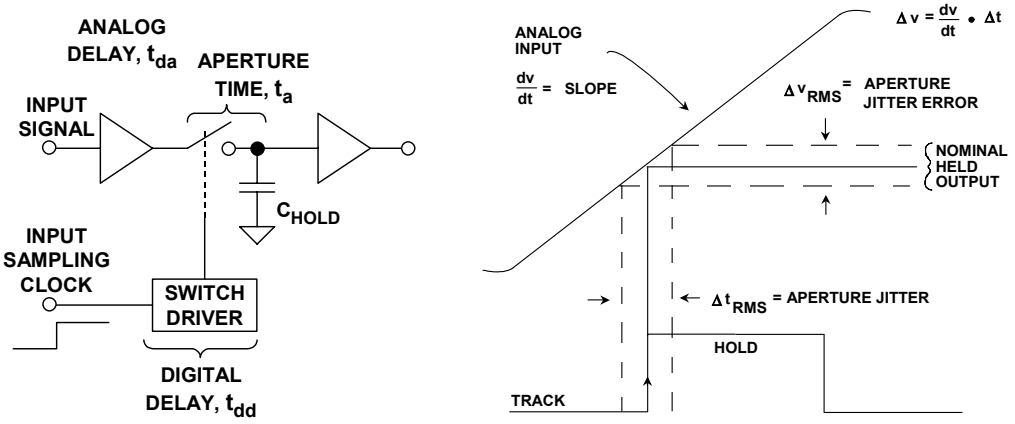
with slope and RMS value

$$\frac{dv(t)}{dt} = 2\pi f \cdot A \cdot \cos(2\pi f t) \tag{1.23}$$

$$\left.\frac{dv(t)}{dt}\right|_{\mathrm{RMS}} = \frac{2\pi f \cdot A}{\sqrt{2}}. \tag{1.24}$$

Multiplying this with the RMS of the aperture jitter, $\Delta t_{\mathrm{A,RMS}}$, the amplitude uncertainty is obtained [13]:

$$\Delta v_{\mathrm{RMS}} = \frac{2\pi f \cdot A \cdot \Delta t_{\mathrm{A,RMS}}}{\sqrt{2}}. \tag{1.25}$$




(a) Sample-And-Hold (SHA) Circuit. (b) Effect of aperture jitter.

Figure 1.22.: Simplified SHA circuit, showing the effect of aperture time jitter, which causes a voltage jitter depending on the signal slope. [1]

Analog to equation 1.13 one can now calculate the SNR limit due to the aperture jitter:

$$\mathrm{SNR(f)/dB} = 20\log_{10}\left(\frac{v_{\mathrm{signal,RMS}}}{v_{\mathrm{noise,RMS}}}\right) \tag{1.26}$$

$$= 20\log_{10}\left(\frac{A/\sqrt{2}}{\Delta v_{\mathrm{RMS}}}\right) \tag{1.27}$$

$$= 20\log_{10}\left(\frac{A/\sqrt{2}}{\frac{2\pi f\cdot A\cdot \Delta t_{\mathrm{A,RMS}}}{\sqrt{2}}}\right) \tag{1.28}$$

$$= 20\log_{10}\left(\frac{1}{2\pi f\cdot \Delta t_{\mathrm{A,RMS}}}\right) \tag{1.29}$$

This shows that the SNR worsens with higher signal frequency, which is shown in Fig. 1.23. The effect is not only due to the aperture time jitter of the ADC but also the jitter of the sampling clock $\Delta t_{\mathrm{S,RMS}}$. Since both effects are uncorrelated, they add quadratically:

$$\mathrm{SNR(f)[dB]} = 20\log_{10}\left(\frac{1}{2\pi f\cdot(\Delta t^2_{\mathrm{A,RMS}} + \Delta t^2_{\mathrm{S,RMS}})^{0.5}}\right) \tag{1.30}$$

Therefore, close attention has to be paid to the sampling clock. The clock sources, Phase-Locked Loops (PLLs), and clock distribution circuits need to have low jitter, and the signals need careful impedance matching and routing.

Although the effect of both jitters on the SNR is in principle the same, the influence in the power spectral density is quite different. This is explained in more depth in [14].

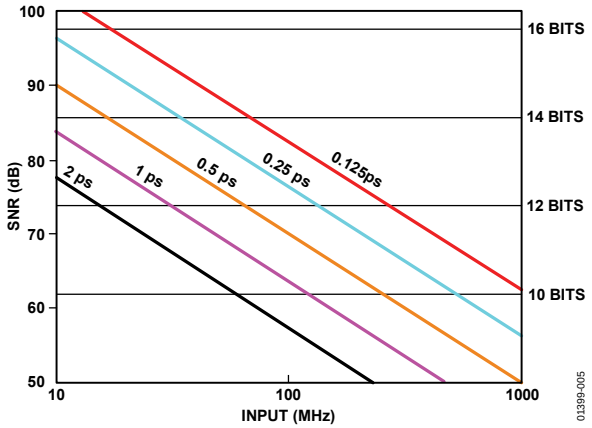


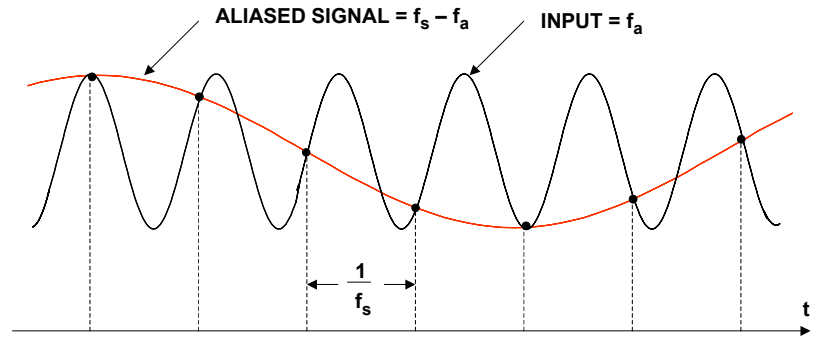

Figure 1.23.: Frequency-dependent degradation of the SNR for different $\Delta t_{\mathrm{A,RMS}}$ (colored). The right axis shows the limit due to quantization noise according to equation 1.13. [13]

### 1.4.6. Aliasing

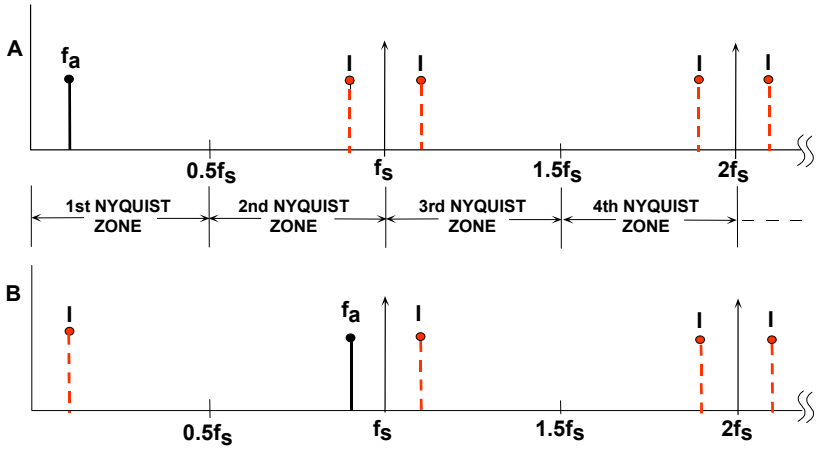
The Nyquist-Shannon Sampling Theorem [15, 16] states that for a given signal bandwidth $f_\mathrm{a}$ the sampling frequency has to be $f_\mathrm{S} > 2 \cdot f_\mathrm{a}$. If this *Nyquist condition* is not fulfilled, the signal cannot be reconstructed correctly from the discrete data, and instead a so-called *alias* or *image* with $f_\mathrm{Alias} = f_\mathrm{S} - f_\mathrm{a}$ will appear in the Nyquist band up to $0 < f < 0.5 f_\mathrm{S}$, also called *first Nyquist zone*. This is shown in Figure 1.24a. Figure 1.24b visualizes the effect in the frequency domain. Case **A** shows a signal with frequency $f_\mathrm{a}$, fulfilling the Nyquist condition. Theoretically its frequency could be reconstructed also at the position of the images, but the lowest frequency, which is in the 1st Nyquist zone, is always assumed. Case **B** shows that this leads to a wrong reconstruction, when $f_\mathrm{a}$ is not in the 1st Nyquist zone: the signal will be reconstructed at the position of the folded back image.

It is important to realize that this does not only apply to the signal, but also to any noise contribution outside of the first Nyquist zone, which is why a strong low-pass filter should be applied close to the highest frequency contribution of the useful signal. Such a filter is typically called *anti-aliasing filter*.

This concludes the introduction into ADC architectures and their noise and error sources.




(a) The sine with $f_a$ (black) is sampled at a frequency $f_s$ slightly below $2 \cdot f_a$. The *alias* appears at a frequency of $f_s - f_a$ (red), and the original waveform cannot be reconstructed.




(b) (A) $f_a$ lies in the 1st Nyquist zone and can be correctly reconstructed. Images of the signal with $f_a$ appear in all Nyquist zones. (B) If the signal does not meet the Nyquist criterion, one of its images will fold back into the first Nyquist zone and the signal will be wrongly reconstructed.

Figure 1.24.: Visualization of aliasing effects in time and frequency domain. A signal with frequency $f_a$ is sampled with $f_S$. [1]

# 2. The CBELSA/TAPS Experiment

The CBELSA/TAPS experiment is one of two baryon spectroscopy experiments at the electron accelerator facility ELSA in Bonn. Aiming at a better understanding of the strong interaction in the non-perturbative regime, its recent focus lies on the investigation of single and double polarization observables. Meson photoproduction experiments, using a polarized beam and a polarized target, are performed in order to obtain a better understanding of the spectrum and the properties of baryon resonances, which are the bound states of the strong interaction. The present chapter will start with a brief introduction into the physics motivation of the experiment and the experimental methods in section 2.1.
Since the CB-SADC that is developed within this thesis is the future readout of the experiment's main calorimeter, the Crystal Barrel, the experimental setup will also be presented in this chapter: First, in section 2.2, the accelerator facility ELSA will be presented. Then section 2.3 will guide through the remaining chapter, describing the different components of the CBELSA/TAPS experiment.

## 2.1. Physics Background

All known interactions in nature are described by the four fundamental forces: the **strong**, **electromagnetic**, **weak**, and **gravitational** force. The first three are described by quantum field theories and are part of the *Standard Model of Particle Physics*, while the description of the gravitational force relies on Einstein's *General Theory of Relativity*. A theory unifying all fundamental forces is yet to be conceived and accepted (the latter being the crucial point: every theory has to be tested by experiments), and may be regarded as the holy grail of modern physics.

The so-called *coupling strengths* $\alpha_X$ are a measure of the strength of the different *Standard Model* interactions. They are normalized and dimensionless, which allows a comparison between the three forces. They can be approximated to [17]

$$\begin{aligned} &\text{Strong interaction: } \alpha_s &&\approx 1, \\ &\text{El.magn. interaction: } \alpha_{em} &&\approx 1/137, \\ &\text{Weak interaction: } \alpha_w &&\approx 10^{-5}. \end{aligned}$$

Calculations of interaction probabilities depend on the coupling strengths of the given interaction. Complex interactions cannot be solved analytically and are approached with perturbation theories, which involve a power series (Taylor expansion) of the coupling strength. Clearly this is only feasible if the interaction is weak with $\alpha_X << 1$, since the power series will not converge otherwise.

While this may work well for the **Electromagnetic** and **Weak** interactions, it is problematic for the **Strong** interaction: the coupling constant does in fact vary significantly from $\alpha_s \approx 1$, depending on the momentum transfer, as shown in the next section.

### 2.1.1. Strong Interaction

The strong interaction is responsible for the formation of bound states of quarks (which are believed to be fundamental particles) and gluons (which are massless bosons, the mediators of the strong interaction). Those bound states are called **hadrons** and can be further classified as **baryons** (fermions, i.e. half-integer spin) or **mesons** (bosons, i.e. integer spin). The proton (consisting of two up and one down quark) and neutron (consisting of one up and two down quarks) are popular examples from the group of baryons, as they are the constituents of the atomic nuclei, which, together with electrons, make up our everyday surroundings.[1]

Similarly to the concept of *positive/negative charges*, to which the photon (the gauge boson of Quantum Electrodynamics) couples to, particles participating in the strong interaction are assigned *color charge*. The underlying theory is called **Quantum Chromodynamics (QCD)**. A single (anti-)quark carries a color charge of either (anti-)red, (anti-)green, or (anti-)blue. The force carrier of the strong interaction, the gluon, carries a color and anticolor charge at the same time. This allows a gluon-gluon interaction, as opposed to the photon, which does not carry an electromagnetic charge itself and hence cannot self-interact. This self-interaction is also responsible for the running coupling constant $\alpha_s$ of the strong interaction, and the so-called *confinement*: with increasing distance between two color-carriers (lower four-momentum transfer $Q$), $\alpha_s$ increases, which is shown in Figure 2.1. More and more energy is needed to further separate two quarks, and eventually the energy stored in the system surpasses the energy to create a quark-antiquark pair from the vacuum. They recombine with the separated quarks to form colorless objects again. As a result, *free* quarks are not observed in nature. In contrast, at decreasing distances or larger four-momentum transfer $Q$, the coupling constant decreases and the quarks can move almost freely, which is called *asymptotic freedom*. It is now evident that the strong interaction can be investigated by means of perturbation theories only at $Q >> 1\,\mathrm{GeV}$, where $\alpha_s << 1$. In this regime, experimental data is provided for example by the high energy experiments of the Large Hadron Collider at CERN. One way to experimentally access the challenging lower energy regime is **baryon spectroscopy**, or hadron spectroscopy in general.

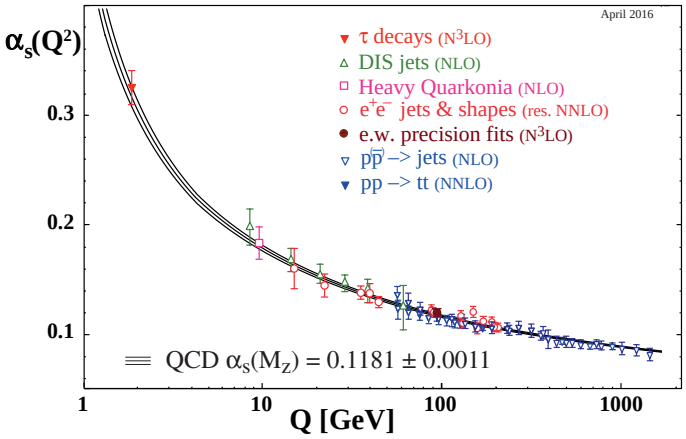



Figure 2.1.: Coupling strength $\alpha_s$ as a function of the energy scale Q. The data points have been extracted through QCD perturbation theory of different order. [18]

[1] While the proton and the bound neutron are stable, the free neutron has a mean lifetime of $\tau \approx 880\,\mathrm{s}$.

### 2.1.2. Baryon Spectroscopy

Spectroscopy is one of the key tools to understand composite systems like molecules, atoms, or nuclei. Figure 2.2 shows how it is used to detect the atomic compositions of an emission nebula, 3000 ly away from earth. Its gaseous content is continuously exposed to ionizing radiation from a nearby white dwarf. The excited atoms relax through radiation of photons in the visible spectrum. The spectrum shows clear structures, allowing to assign individual *peaks* to the transitions in the electron shells of the atoms. The composition of the nebula can be determined, and conclusions can be drawn about our galaxy's chemical evolution.

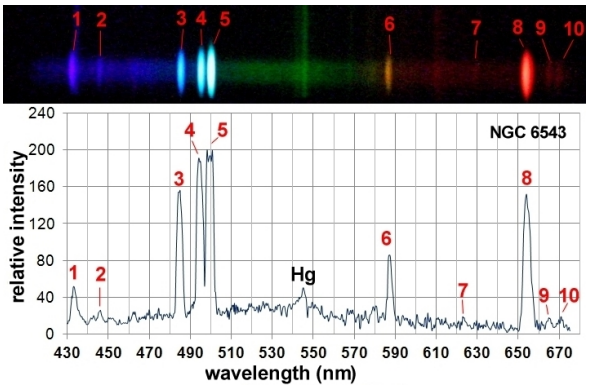



Figure 2.2.: Spectrum of the Cat's-Eye Nebula NGC 6543 with ten identified emission lines. The data was taken with a 20 cm reflector telescope and a DADOS spectrograph with an exposure time of 1 h. The upper part shows the photographic spectrum, the lower part the corresponding intensity plot. Peaks 1, 3, 8: hydrogen Balmer series; peaks 2, 6, 9: helium; peaks 4, 5: oxygen; peaks 7, 10: sulfur. [19]

Similar to the excitation of electrons in the atomic shell that allows to draw conclusions on the electromagnetic interaction and the atom's properties, the quark constituents of baryons (protons and neutrons) can be excited for us to learn about the strong interaction. The role of the white dwarf's ionizing radiation from the previous example is now taken by artificially created beams of pions/kaons, electrons, or photons. Instead of a gaseous nebula, a contained sample of nuclei is irradiated. In photoproduction experiments that investigate the excitation spectrum of the nucleon, typical photon energies range from a few 100 MeV to a few GeV, which is many orders of magnitude above what is needed for atomic spectroscopy. Figure 2.3 shows the total absorption cross-section for a photon impinging on a proton (in black). In contrast to Figure 2.2, where well-separated peaks are observed in the emission spectrum, it is difficult to extract valuable information from the total absorption spectrum. An exception is the very prominent peak of the $\Delta$-resonance, the first excitation of the nucleon, with a mass of $\sqrt{s} = 1232\,\text{MeV}$. The largest contribution to its decay is the $\gamma p \to p\pi^0$ decay channel with a probability of $\frac{2}{3}$; the decay channel $\gamma p \to n\pi^+$ with $\frac{1}{3}$ probability is not included in the figure. Further resonances can hardly be seen, except for minor enhancements to which several resonances contribute. If the different final states are separated (red, blue), peak-like structures become visible at the lower excitation energies, but towards higher energies the spectra are almost structureless again. Visible peaks usually stem from multiple overlapping and interfering resonances, and are therefore referred to as resonance regions.

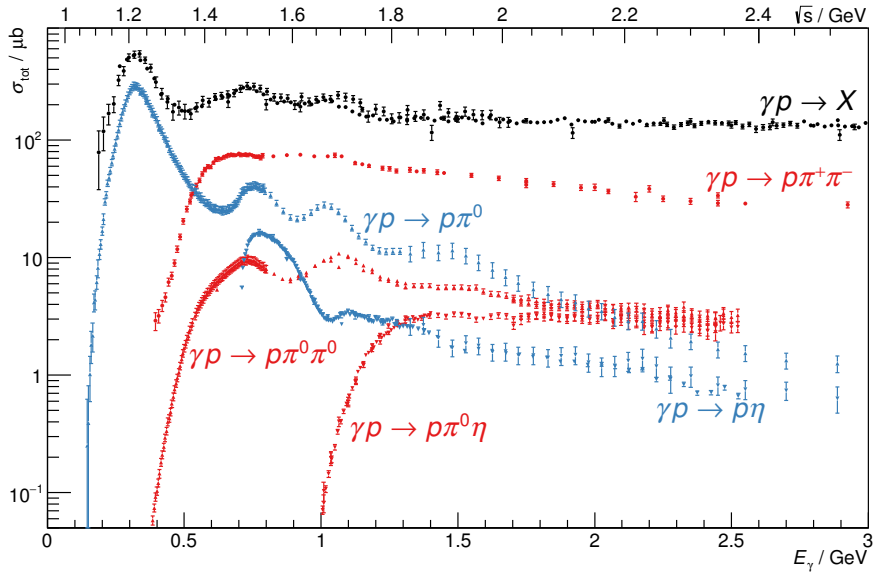



Figure 2.3.: Total photoabsorption cross-section of the proton and contributions from different meson photoproduction channels. The lower and upper x-axes show the energy of the impinging photon ($E_\gamma$) and the center-of-mass energy ($\sqrt{s}$), respectively. The total cross-sections were determined from several different experiments. [20]

To extract resonances and their properties such as mass, width, partial decay width, and the quantum numbers from the data, the measurement of total cross-sections alone is not sufficient. Due to their very short lifetime ($\approx 10^{-24}$ s), resonances are broad and therefore strongly overlapping. One experimental method is to filter the data with the polarization of the incident beam or target particle, or to measure the polarization of particles in the final state. This leads to asymmetries in the measured cross-section, which are expressed by the so-called single-/double-polarization observables. In the CBELSA/TAPS experiment, the photon beam (cf. section 2.4) and the target nuclei (cf. section 2.6) can indeed be polarized, allowing access to double-polarization observables. Furthermore, instead of a proton it is possible to use a neutron as target nucleus: while the $\Delta$*-resonances with isospin $I = 3/2$ are produced with the same strength off proton and neutron, this coupling might be significantly different for N*-resonances with $I = 1/2$. In the experiment, it is also possible to create multi-meson final states, providing access to additional interesting decay modes, including isospin selective ones. For example, isospin $I = 1/2$ and $I = 3/2$ baryon resonances can contribute to $\gamma p \to p\pi^0$, while $\gamma p \to p\eta$ or $\gamma p \to \Delta\eta$ are isospin-selective: the former one only allows $I = 1/2$ N*-resonances, the latter $I = 3/2$ $\Delta$*-resonances.

The combined data (also from other baryon spectroscopy experiments) can then be analyzed in a multi-channel partial wave analysis (PWA) to extract the resonances and their properties. The results from the partial wave analysis can be compared to expectations from theoretical models. Such models are usually simplified and do not reflect all properties of the full QCD theory, but they can be solved in the energy regime where $\alpha_s \approx 1$ makes perturbative solutions unfeasible. Two models will be shown in the following paragraphs.

**Constituent Quark Model** One popular approach to simplify the QCD theory is the fully relativistic constituent quark model based on the Bethe-Salpeter equation, by Löring, Metsch, and Petry [21]. In this model, the constituent quarks interact through instantons (residual interaction) and are considered to be confined in a linearly rising potential. In a later revision, an additional spin-flavor dependent interaction has been taken into account to refine the model [22]. The model successfully predicts many experimentally found resonances, albeit only requiring ten parameters for the calculation. A comparison of predictions and experimentally verified resonances is shown in Figure 2.4 for nucleons (baryons with isospin $I = 1/2$). One peculiarity is obvious: while prediction and experiment match more or less well, especially in the lower energy regime, many more states are predicted than measured for $M \gtrsim 2000\,\mathrm{MeV}$. This is referred to as the *missing resonances*, and two reasons can potentially explain this phenomenon [20]. First, in the past the majority of experimental data has been collected in $\pi N$ scattering, and many missing resonances are predicted not to couple to this channel. More data e.g. from $\gamma N$ scattering in this energy regime is required for a better understanding. Secondly, it could be that not all theoretically possible degrees of freedom of the baryon are realized in nature.

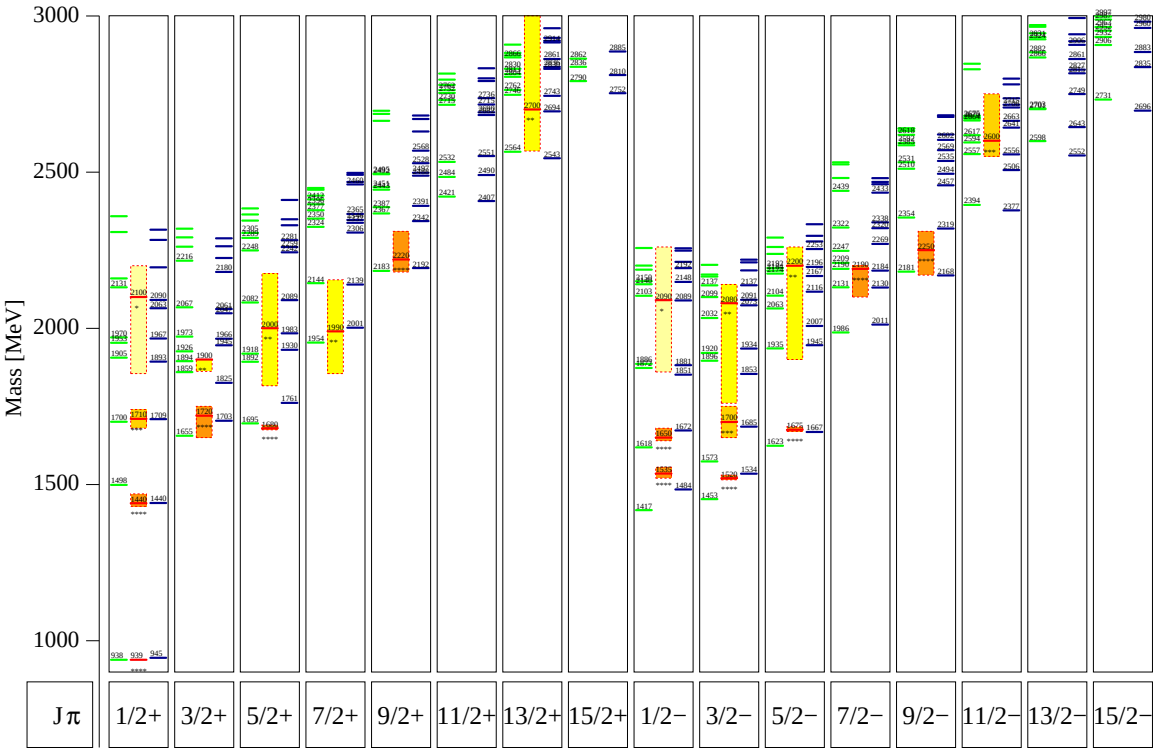



Figure 2.4.: Predictions and measurements of the nucleon's excitation spectrum. The y-axis shows the mass of the resonances (excitations) that are grouped by total angular momentum ($J$) and parity ($\pi$). The middle of each column shows the experimentally obtained values and uncertainties according to [23], while the markings left and right indicate the predicted position of resonances from two different models (left: [21], right [22] with additional spin-flavor interaction). [22]

**Lattice QCD Model** Another approach is Lattice QCD. With a much higher computing effort, the QDC theory can be solved numerically in a discretized space and time lattice for a constrained volume. The pion's mass is taken as a reference in this calculation, but the currently available computing power does not allow for a simulation with the correct mass; instead the simulations use higher (non-physical) masses and predictions are made by extrapolation towards the pion's true mass.

The obtained spectra (Figure 2.5) show a similarity to the data from the previously shown constituent quark model ($J = \frac{1}{2} \ldots \frac{7}{2}$ can be compared for the nucleons), and indeed the level counting is found to be consistent. Interestingly, also more states are predicted than observed here. Even though Lattice QCD starts from first principle, considerable approximations are made: the calculations are done at rather high pion mass, and the states calculated cannot decay. Therefore one might expect changes in the spectrum as soon as Lattice QCD calculations become more realistic. How large these changes will be remains to be seen.

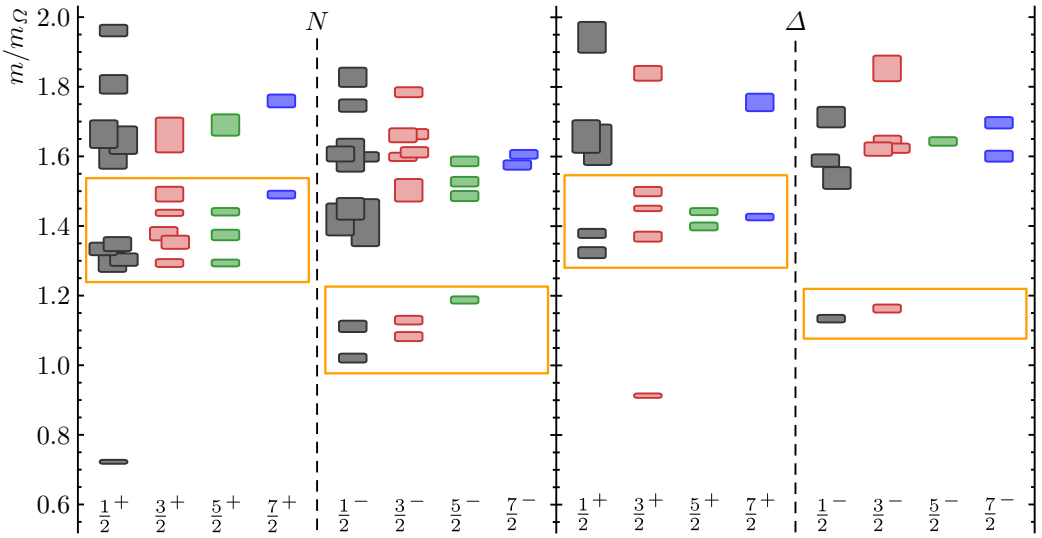



Figure 2.5.: Nucleon and Δ-baryon spectrum, calculated at a non-physical pion mass of 396 MeV. The x-axis shows different isospin/parity ($J^P$) configurations, the y-axis shows the mass in units of the Ω-baryon mass. The marked regions show the two lowest oscillator excitation bands of the constituent quark model. [20, 24]

**Conclusion** It was shown that the theoretical investigation of the strong interaction in the energy regime of baryon excitations is a challenging endeavor due to the large strong coupling constant. Discrepancies between the experimental data and the model predictions show the need for higher quality data and more statistics. It will be shown in section 3.1 that the SADC development presented in this thesis can contribute to an improvement of the CBELSA/TAPS experiment in both regards.

## 2.2. ELSA

The Electron Strechter Accelerator (ELSA) is a three-stage electron accelerator with a maximum energy of 3.5 GeV, which has been supplying the photon beam (through a Bremsstrahlung-process, see section 2.4) for the CBELSA/TAPS experiment for almost 20 years. Figure 2.6 shows an overview of the facility, including the two experimental sites, CBELSA/TAPS and BGO-OD. Technical details can be found in [25]. After a peak into the facility's future plans, the following sections will summarize the properties of the three accelerator stages.

**Future Plans** Within a Collaborative Research Center (SFB TR16, project D.2), studies and exploratory work have been conducted to allow an increase in energy and intensity of the ELSA facility [26]. It will be shown in section 3.1 that this thesis presents one building block to prepare the CBELSA/TAPS for a higher beam intensity.

### 2.2.1. Linear Accelerators

The first stage consists of a Linear Accelerator (LINAC) with up to 26 MeV beam energy. Historically two different accelerators were available:

**LINAC1** has been decommissioned in 2002, a replacement is being worked on [27, 28]. In the latest revision it was supplied with electrons from a thermal electron gun with an energy of 90 keV [29] and could supply a beam current of 800 mA at 20 MeV in pulsed operation. The beam from LINAC1 could either be fed to the next stage (Booster Synchrotron), or be used for material irradiation in a test area.

**LINAC2** has been built up following the need of a polarized electron beam for the investigation single/double polarization observables. It is able to supply longitudinally[2] polarized electrons with a pulsed beam current of up to 100 mA and a polarization degree of $\approx 83\,\%$. The polarized electrons are generated by irradiation of a GaAs photo-cathode with circularly polarized laser light [30] and enter LINAC2 with an energy of 50 keV. After the decommissioning of LINAC1, a thermal electron gun was installed at LINAC2 for the continued supply of unpolarized electrons.

In recent years, LINAC2 has been equipped with a new load lock system for improved operational availability. In addition, it has been shown that due to a new elaborate hydrogen cleaning system, it could operate with a current of up to 200 mA [31, 32].

### 2.2.2. Booster Synchrotron

The electron beam of the LINAC is injected into the Booster Synchrotron, which has a circumference of 69.6 m and runs at 50 Hz line frequency. The synchrotron was commissioned in 1967 with a designed energy of up to 2.5 GeV [33]. Nowadays it is used as a booster stage between the LINAC and the Stretcher Ring [34] with an extraction energy of 0.5 GeV to 1.6 GeV (usually operated at 1.2 GeV). The maximum design energy cannot be reached anymore, since one of the DORIS cavities has been moved to the Stretcher Ring.

[2] The electron's longitudinal polarization is flipped to a transversal polarization by an electrostatic deflector.

### 2.2.3. Stretcher Ring

The Stretcher Ring with a circumference of 164.4 m was commissioned in 1987/88 with a design energy of 0.5 GeV to 3.5 GeV (limited by the dipole magnet power supply and the RF generated acceleration voltage). ELSA is an accelerator of FoDo-type[3], supplemented by twelve sextupoles and two different types of resonators (DORIS, PETRA) for acceleration (see Figure 2.6 for details). It is nowadays operated in the so-called **Post-acceleration Mode**, which increases the energy of the electrons injected into the Stretcher Ring up to 3.5 GeV. The Post-acceleration mode was made possible by an upgrade of the control system in 1994, also allowing a nearly homogeneous filling of the Stretcher Ring, making the previously used (but energy limited) **Stretcher Mode** obsolete.

In a typical extraction cycle [35], ELSA is first filled with electrons by the synchrotron in at least three revolutions, since the ratio of the circulation periods[4] of Synchrotron and Stretcher ring is approximately $7:3$. Then, the electrons are accelerated with up to 6 GeV s$^{-1}$, typically[5] up to an energy of 3.2 GeV. Lastly, the beam is extracted to one of the two experiments CBELSA/TAPS and BGO-OD, or an area for detector tests. A typical extraction cycle takes roughly 8 s, with a duty factor of 80 % to 90 % to account for the time needed to fill the Stretcher Ring, and for the post-acceleration. More detailed information about the Stretcher Ring may be obtained from [36, 37] and numerous publications [38].

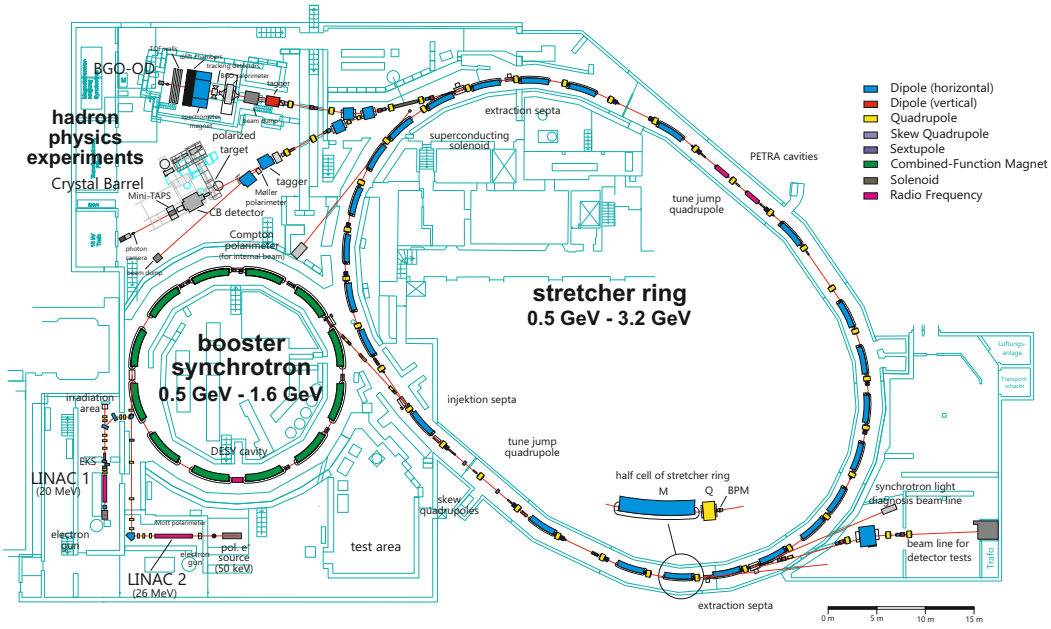



Figure 2.6.: Overview of the ELSA setup. Lower left: electron sources and linear accelerators, followed by Booster Synchrotron and the Stretcher Ring in the center. Upper left: experiment area with CBELSA/TAPS and BGO-OD experiment. [39]

[3] FoDo stands for: F = vertically focusing, horizontally defocussing quadrupole magent, D = vice versa, and o = space or dipole (deflection) magnet.

[4] Revolution time of the Stretcher Ring: 548 ns, revolution time of the Synchrotron: 232 ns.

[5] Acceleration up to 3.5 GeV comes with severe limitations and difficulties, which is why the accelerator is usually not operated above 3.2 GeV.

## 2.3. CBELSA/TAPS Experimental Setup Overview

Figure 2.7 shows an overview of the CBELSA/TAPS experiment. The electron beam extracted from ELSA hits one of the targets inside a goniometer tank, where unpolarized, linearly-polarized, or circularly-polarized photons can be created through bremsstrahlung (section 2.4). The electrons are then deflected from the photon beam, and a tagging hodoscope allows to determine the energy of the photons from the deflection angle of the electrons (section 2.5). The photons hit one of the photoproduction targets (section 2.6), and the reaction products are measured in the different detector systems (section 2.7) for calorimetry, timing, charge identification, and flux normalization.

The chapter concludes with details about the readout electronics (section 2.8), trigger system (section 2.9), data acquisition (2.10), and the light pulser and slowcontrol system (section 2.11).

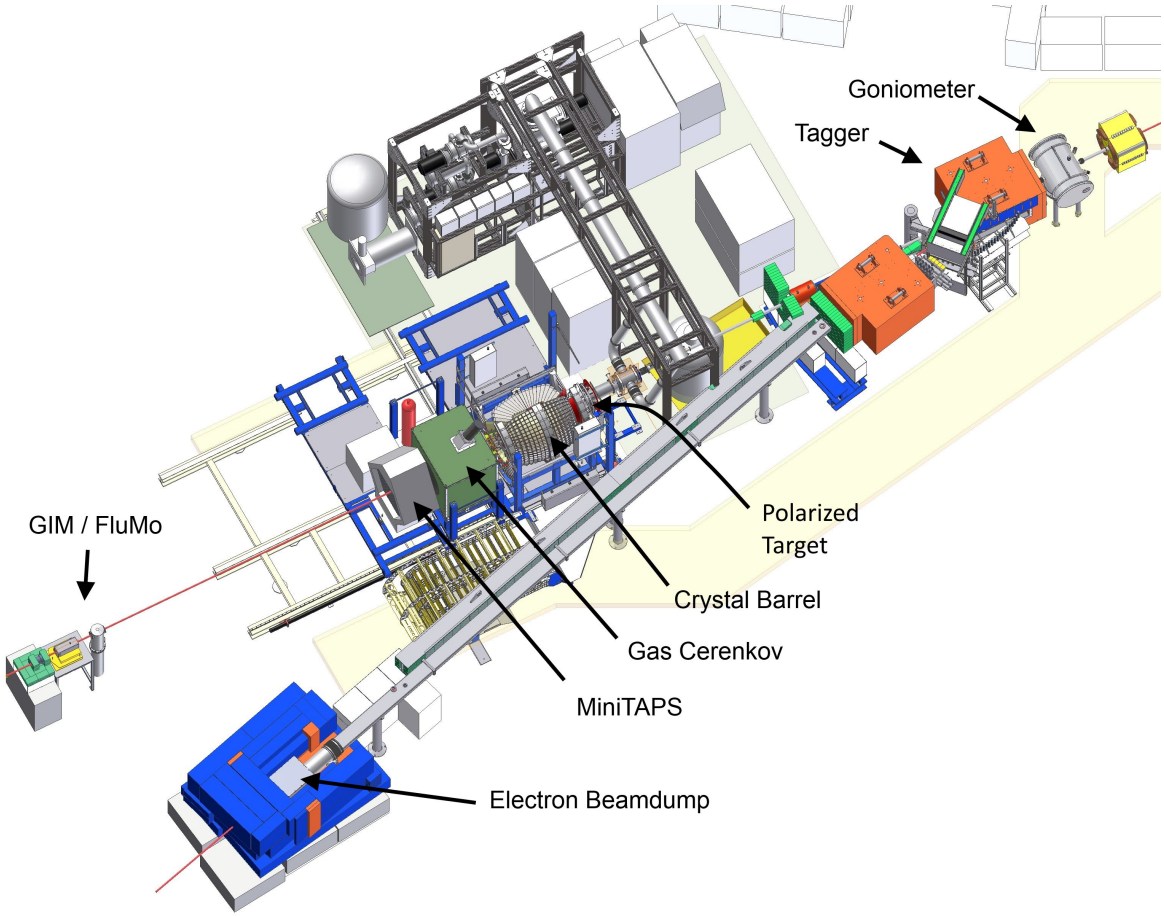



Figure 2.7.: Overview of the Crystal Barrel/TAPS experiment [40]. The electron beam is extracted from ELSA and enters on the right side of the drawing; it is diverted from the created photon beam and dumped in a block of iron (bottom left).

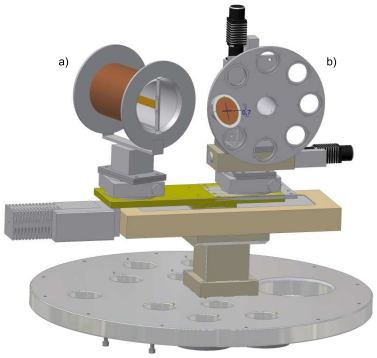
## 2.4. Bremsstrahlung Targets

After resonant extraction of the electron beam from the Stretcher Ring, the electrons impinge on one of the targets mounted in a five-axis goniometer (Figure 2.8). In the Coulomb field of the target nuclei, photons are created via bremsstrahlung. With amorphous copper radiators with (12, 50, 59 and 150) µm thicknesses[6], **unpolarized photons** [41] are created. Thicker radiators yield a higher photon rate at the expense of an increased multiple scattering probability. The energy spectrum of the photons approximately follows a $1/E_\gamma$ distribution. Additionally, linearly- and circularly-polarized photons can be created to access polarization observables as mentioned in section 2.1.2.

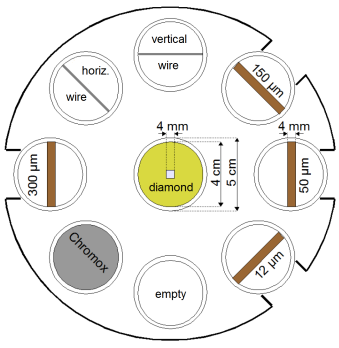
**Linearly-polarized photons** are created with coherent Bremsstrahlung from a diamond crystal radiator of 500 µm thickness using unpolarized electrons. In this process, the recoil from the impinging electron can be absorbed by the whole lattice. In addition to the non-coherent $1/E_\gamma$ spectrum, coherent Bremsstrahlung shows a peak that is cut off at one side due to kinematic limitations. The position of the peak depends on the orientation of the crystal lattice relative to the beam axis. With the goniometer, the diamond crystal can be tilted such that the position of this so-called *coherent edge* can be set to a demanded energy. The diamond radiator has been installed at the experiment in the scope of [42].

**Circularly-polarized photons** allow the measurement of helicity-dependent polarization observables. In the experiment they are created from longitudinally polarized electrons impinging on an amorphous radiator, a ferromagnetic VACOFLUX 50® foil. The system is depicted on the left in Figure 2.8. The foil is aligned by ±20° with respect to the beam axis and is surrounded by a coil to polarize it. It is now possible measure the Møller-scattering to calculate the polarization of the beam electrons scattered off the foil. Through the well-defined helicity transfer, the polarization of the photons can be determined [43].

The setup was developed and installed in the scope of [44] and was lately maintained and upgraded with new readout electronics [45].




(a) a) Møller-Polarimeter with VACOFLUX 50® foil and b) goniometer wheel. [40]




(b) Wheel with targets in eight outer/one inner tray. Currently 59 µm instead of 300 µm is mounted. [44]

Figure 2.8.: The goniometer with multiple translational and rotational axes, together with the radiator targets.

[6] The thickness corresponds to (0.84, 3.5, 4.13 and 10.5) $\times 10^{-3}$ $X_0$, where $X_0$ is the radiation length.

## 2.5. Tagging Hodoscope

All electrons are deflected from the photon beam line with the so-called Tagger magnet after they have traversed the goniometer tank. The more energy an electron loses due to Bremsstrahlung, the larger the angle of deflection in this magnet will be. This angle is measured with the Tagging Hodoscope to *tag* the created Bremsstrahlung-photons with an energy according to the relation $E_\gamma = E_{beam} - E_{tagger}$. The Tagging Hodoscope consists of two detector systems that are read out by Photo-Multiplier Tubes (PMTs): 96 overlapping plastic scintillators, covering an energy range of 2.1 % to 82.5 % of $E_{\text{beam}}$ (with a resolution of 0.1 % to 6.0 % of $E_\gamma$), as well as 480 scintillating fibers, covering a range of 16.6 % to 87.1 % of $E_{\text{beam}}$ (with a resolution of 0.1 % to 0.4 % of $E_{\text{beam}}$). [46]

The Tagging Hodoscope has recently been equipped with the same FPGA-based Time-to-Digital Converter (TDC) electronics as the Crystal Barrel Calorimeter (see section 2.8.6), but in this case with asynchronous (carry-chain) TDC firmware for best time resolution. Preliminary analyses have yielded a resolution of $\sigma_{\text{bars}} \approx 1\,\text{ns}$ and $\sigma_{\text{fibers}} \approx 1\,\text{ns}$ [35].

## 2.6. Photoproduction Targets

In the photoproduction target, the impinging bremsstrahlung photons can hit the target nucleons, ideally producing a resonant state. For photoproduction off an **unpolarized** (free) proton or a neutron, a target cell which can be filled with either $LH_2$ or $LD_2$ was developed [47]. For photoproduction off **polarized protons and neutrons** on the other hand, the Bonn Frozen-Spin Target with the target materials 1-Butanol or D-Butanol (perdeuterated) in a target cell of 2 cm length is used. It was first in operation in 1998 at the MaMi[7] accelerator facility [48] for the investigation of the Gerasimov-Drell-Hearn (GDH) sum rule [49] and was afterwards used in Bonn at the GDH experiment at ELSA for the same investigations at higher energies [50]. The Bonn Frozen-Spin Target can reach a polarization degree of up to 87.7 % at a magnetic field strength of 2.5 T at temperatures of around 60 mK [51]. To reach such a high degree of polarization at comparably low magnetic field strength and high temperature, the easier achievable high transverse-electric (TE) polarization of the target electrons is transferred to the nucleons via off-resonance microwave transitions.[8] This process is called dynamic nuclear polarization (DNP). Unfortunately, the target does not allow continuous operation: for initial polarization, it has to be inserted into a strong magnetic field to orient the spin. After removing the external field, the relaxation is stretched in the order of 1000 h with the help of an internal holding coil with 0.6 T. After a few days, the beam time has to be interrupted by the necessary re-polarization of the target material.

In 2011 the Frozen-Spin Target was dismounted for repairs. Coincidentally, a new continuous mode dilution refrigerator with a stronger internal coil is in development. It will allow continuous measurements without the need for re-polarization phases every few days [53].

In the mean time, the CBELSA/TAPS experiment relies on the Mainz-Dubna Frozen-Spin Target for the continuation of production beam times (see section 9.7), which operates at a temperature of around 27 mK with a holding coil strength of 0.45 T [51].

[7] MaMi: Mainzer Microtron

[8] To reach a direct polarization of the target nucleons in the same magnitude, a magnetic field strength of $\approx 15\,\text{T}$ at a temperature of $\approx 10\,\text{mK}$ would be necessary. [52]

Figure 2.9.: The Bonn Frozen-Spin Target with part of the holding structure. The target cell with the holding coil is situated at the very front. [40]

## 2.7. Detector Systems

Aiming at an investigation of reactions such as $\gamma p \to N^*/\Delta^* \to p\pi^0$, $p\pi^0\pi^0$, $p\pi^0\eta, \ldots$ or $\gamma p \to N^* \to p\eta, \ldots$ where the neutral mesons preferably decay into photons, a detector system is needed to measure the photons' energy, ideally over the full solid angle. In this section, the four detector components that measure the final state products of the de-excitation will be presented. The first detector surrounding the target is the **Inner Detector**, which is able to detect charged particles such as the proton in the final state. It is followed by the **Crystal Barrel Calorimeter**, ideally suited to measure the energy and direction of photons with almost $4\pi$ coverage, with additional charged particle identification for forward regions not visible to the inner detector. For narrow angles in the forward direction, the **MiniTAPS** calorimeter is responsible for the detection of photons and tagging of charged particles. Between the two calorimeters, a **Gas Čerenkov Detector** helps to detect electromagnetic background, which is then suppressed on the trigger level. At the end of the beam line, the **Gamma Intensity Monitor** and the **Flux Monitor** provide data for flux normalization.

### 2.7.1. Inner Detector

The detector for charged particle identification, also abbreviated ChaPI, or simply called the Inner Detector, is a cylindrical detector consisting of 513 fibers in three layers. It surrounds the photoproduction target, allowing for a charged particle identification in the CBELSA/-TAPS experiment since the start of data taking in 1999 [54, 55], with an initial polar angle coverage of $24° < \Theta < 166°$ [56]. Due to the arrangement of the layers (one parallel to the beam axis, two contorted), the penetration point can be reconstructed with a hit in two of three layers. A fast readout with 16-fold PMTs allows for the integration into the trigger system. Modifications regarding the mounting structure of the detector, which were necessary due to new photoproduction targets, took place in 2006 [56] and 2017 [57]. Since 2017, the Inner Detector is read out with the new FPGA-based TDC electronics of the Crystal Barrel Calorimeter (see section 2.8.6). Due to mechanical constraints of the currently-used Mainz-Dubna Frozen-Spin Target, the position has been shifted downstream, resulting in a polar angle coverage of $13.7° < \Theta < 155°$ [58].

### 2.7.2. Crystal Barrel Calorimeter

The main detector of the CBELSA/TAPS experiment is the Crystal Barrel Calorimeter (shown in Figure 2.10), which (together with the MiniTAPS calorimeter) covers almost the full solid angle. It was build to be used at the LEAR accelerator starting 1989 [59], where it was used as a spectrometer to study the products of $\overline{p}p$ and $\overline{p}d$ annihilations. It originally consists of 1380 CsI(Tl) scintillator crystals, which are arranged in 30 slices in forward and backward direction, each holding 23 crystals. With 30 cm length (corresponding to 16.1 radiation lengths), the CsI(Tl) crystals are able to contain almost 99 % of the energy of a 2 GeV photon. Although the spatial granularity is only 6°×6° (only 6°×12° for three rings closest to the beam in forward and backward direction), the angular resolution due to the showering of the electromagnetic cascade is as good as 1.2° [60]. The properties of the CsI(Tl) scintillator crystals, as well as the granularity, lead to an energy resolution [59] of

$$\frac{\sigma_E}{E} \approx \frac{2.5\,\%}{\sqrt[4]{E}}, \tag{2.1}$$

which has been confirmed in measurements after the recent APD upgrade [61].

For the setup in Bonn, where it is used since 1999 in different configurations, the two most backward rings of crystals eventually had to be removed due to mechanical constraints of the photoproduction target and the Inner Detector. With 23 crystals in the forward slices and 21 crystals in the backward slices, the overall coverage in the azimuthal angle is now $\theta \approx 12.2°$ to 156°.

**Forward Plug Veto** The three most forward rings lack azimuthal coverage by the Inner Detector. Alternative methods for fast detection of charged particles have been investigated [62, 63], leading to the installation of overlapping plastic scintillator plates in front of the calorimeter crystals, read out by PMTs [64]. Apart from providing the charge tagging, the so-called Forward Plug Veto can be used as a trigger veto to suppress electromagnetic background in the forward direction (electrons, positrons, charged pions), but for this usually only the Gas Čerenkov Detector is used (see next section).

**APD Upgrade** During the last years, the Crystal Barrel Calorimeter underwent extensive maintenance and upgrade tasks to improve the trigger efficiency for reactions off a neutron target. The previous PIN photo-diode readout has been replaces by an APD readout, and new readout electronics have been installed. This will be discussed in more detail in section 2.8. The Sampling-ADC presented in this thesis is part of the new readout electronics.

### 2.7.3. Gas Čerenkov Detector

In the forward angle $\theta < 12.8°$, electromagnetic background can be suppressed with the Gas Čerenkov Detector. Electrons and positrons of energies >17.4 MeV, stemming from Compton scattering or pair creation processes in the target cell, create Čerenkov light in the $CO_2$ gas of the detector. The detector is read out with a PMT and is usually used as a veto detector in the trigger system, reaching a background suppression efficiency of 99.97(5) % [66].

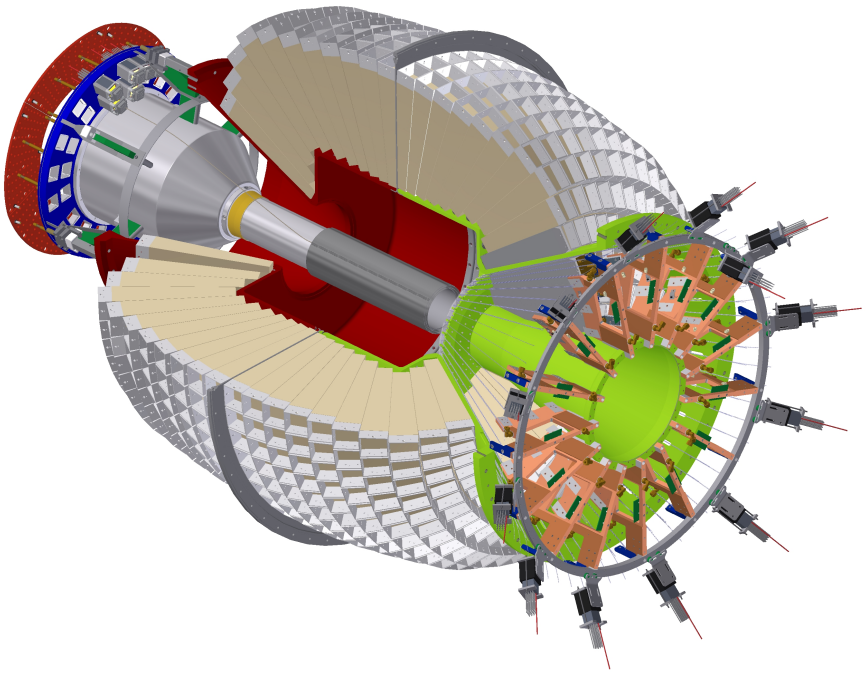


Figure 2.10.: Sectional drawing of the Crystal Barrel Calorimeter. The photon beam enters from the top left side, which shows the holding structure for the Inner Detector (dark gray in the center of the calorimeter) with the PMT readouts for the fibers. The right side shows the holding structure containing the three most forward types of crystals, including the PMT readout for the plastic scintillator plates, which are mounted in front of the crystals. [65]

### 2.7.4. MiniTAPS Calorimeter

The Crystal Barrel Calorimeter is complemented in the angular range $\theta = 12°$ to $1°$ by the MiniTAPS Calorimeter[9]. It consists of 216 $BaF_2$ scintillator crystals, which are suitable for the higher-rate forward region due to their short signal decay [67, 68]. Barium fluoride has two decay components, a very fast one with 600 ps and a much slower with 630 ns. The slower decay component is removed for the inner region of crystals by means of an optical band-pass filter to reduce pile-up events in this region with highest hit rates [69, 70].

### 2.7.5. Gamma Intensity Monitor and Flux Monitor

Two detectors at the end of the photon beam line provide data for the flux normalization of the cross-section. The Gamma Intensity Monitor, consisting of a $4 \times 4$ matrix of $PbF_2$ crystals, measures the Čerenkov light that is created by electromagnetic showers [71]. Due to saturation effects at count rates above 10 MHz, a second detector called the Flux Monitor, which is based on plastic scintillators, has been installed in addition [72].

[9] MiniTAPS is adapted from the original Two Arm Photon Spectrometer (TAPS) calorimeter.

## 2.8. Readout Electronics

The readout electronics of the Crystal Barrel Calorimeter and other detectors have been extensively upgraded in the last years within a sub-project of the Deutsche Forschungsgemeinschaft funded "Collaborative Research Centre 16 - Subnuclear Structure of Matter":

### SFB/CRC TR16 D.3: Timing and Tracking for the Crystal Barrel Detector

> "In order to enhance the trigger capabilities of the Crystal Barrel experiment, a new Avalanche-Photo-Diode readout, which will provide a good time resolution while maintaining the excellent energy resolution of the calorimeter, has been developed and will be installed in 2013. The future energy readout is planned to be realized with modern sampling ADCs to further enhance the rate capability. Additionally, a prototype Time-Projection-Chamber was developed, which can be integrated into the Crystal Barrel setup to provide tracking information for charged final state particles." [73]

Three projects are mentioned in this abstract: the APD readout (which is described in [61] and [74]), the SADC project (which is described in this thesis) and the Time Projection Chamber (which is described in [75]). Coinciding with the end of the funding period in 2016, the Crystal Barrel Detector has indeed been fully upgraded with the new front-end readout electronics and has gone through an isolated commissioning phase. The successful commissioning in conjunction with the complete CBELSA/TAPS experiment followed in 2017.

In this section, a motivation and an overview of the new read-out electronics will be provided, with a focus on the components whose characteristics influence the SADC development or performance. A more detailed insight of the development of the readout electronics is given in chapter 5 of [61].

### 2.8.1. Motivation for the APD Readout

The previous readout, consisting of Positive-Intrinsic-Negative (PIN)-photodiodes, was not able to generate a signal fast enough to include the Crystal Barrel Calorimeter signals into the first-level trigger decision due to its insufficient Signal-to-Noise ratio (SNR). This caused increased dead time and decreased trigger efficiency in specific scenarios (details follow in section 2.9). It was attempted to generate a fast timing signal with the PIN-photodiodes, but it was found that it is not possible without unacceptable limitations [76, 77] (e.g. increased trigger threshold levels). Two alternatives have been investigated as an addition to or a replacement for the PIN-diode readout:

- readout with APDs [77, 78]
- readout with Silicon Photo Multipliers (SiPMs) [79, 80]

An APD is a single photodiode that, due to doping, exhibits a region of high field strength, leading to avalanche-like multiplication of electron-hole pairs. SiPMs are arrays of small APD "pixels" operating in Geiger-mode. At the time, the APDs showed a better SNR, and it was possible to generate a timing signal with a latency of less than 130 ns for energy deposits of 10 MeV [77, 81]; and it was decided to commence the upgrade with APDs.

### 2.8.2. Concept of the APD Readout

Figure 2.11 provides an overview of the signal chain, originating from the APDs and ending in the data storage of different digitizers. An overview of all components will be given, with details being provided in the following paragraphs and sections as indicated in the figure.

Each of the 1320 CsI(Tl) crystals is equipped with two **APDs**, glued to the end face of the crystal, with a dedicated **Preamplifier** circuit. The preamplified signals are added and transmitted over several meters of cable by the **Linedriver**, with an intermediate stage to consolidate the signals of each slice of the Crystal Barrel Calorimeter (**Slice Board**, not shown). The **Buffer/Splitter** receives the signals and distributes them to three branches. The first two branches, **SADC** and **Charge-to-Digital Converter (QDC)**, measure the signal's amplitude or integral, which are proportional to the energy deposited in the CsI(Tl) crystals. The first branch was added specifically for the SADC upgrade described in this thesis, allowing the SADCs to run in parallel to the existing QDC readout. The third branch shows the **TDC** that creates timestamps for all registered signals, and provides fast information about energy deposits and their clustering in the calorimeter; this information is necessary for the integration into the trigger decision. Data from all three branches is processed by so-called Local Event Builders (LEvBs) and finally stored to disk.

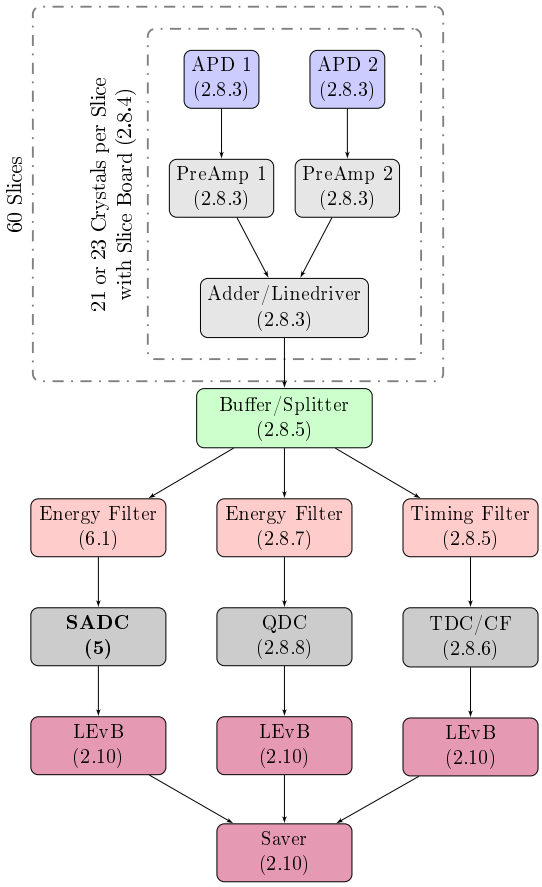



Figure 2.11.: Overview of the readout chain of the Crystal Barrel Calorimeter with the new APD front end, showing also the SADC that is described in this thesis.

### 2.8.3. Front-End Electronics

This section describes the part of the new electronics situated directly behind the CsI(Tl) crystal, namely APDs, preamplifier, and linedriver. A photograph of the assembly is shown in Figure 2.14. The readout was developed within the dissertation of C. Honisch [61].

**Avalanche Photodiodes** For the upgrade, every CsI(Tl) crystal has been equipped with two type S11048(X3) APDs from HAMAMATSU. They are categorized as "short-wavelength type" [82], or more general "reverse type": they have a $p^+$-$\pi$-p-$n^+$-doping[10], with the photons impinging on the $p^+$ layer and the high-field multiplication region shortly behind the entrance window (see Figure 2.12). This leads to a higher SNR when compared to the "reach-through type" (with a structure of $p^+$-p-$\pi$-$n^+$), as the dark current mostly undergoes hole multiplication instead of electron multiplication [83].

The newly-assembled crystals, including the electronics mentioned in the following paragraph, were extensively tested and characterized within the dissertation of M. Urban [84].

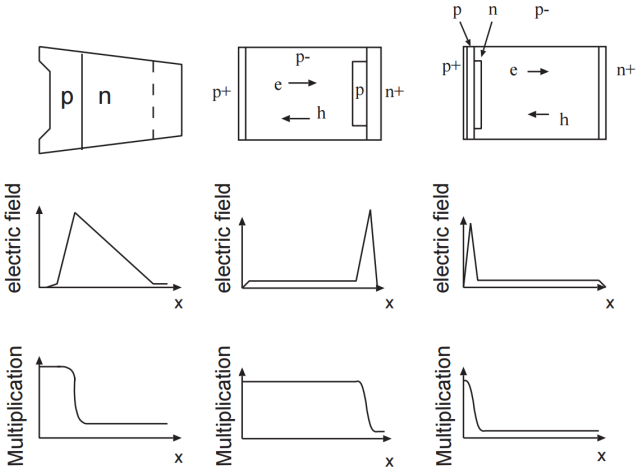



Figure 2.12.: Internal structures of (a) beveled edge (b) reach-through (c) reverse-type APDs. The photons impinge from the left. The early electron multiplication after the entrance window lead to a good SNR for the reverse-type APDs. [83]

**Preamplifier and Linedriver** The low-noise charge preamplifier SP917[11] has been developed at the University of Basel, with synergistic effects due to the previous/parallel developments of preamplifiers for the PANDA experiment [85]. It has been produced in revision SP917E for the CBELSA/TAPS experiment with a measured bandwidth of 13.5 MHz and a high-pass filter with a time constant of 54 µs [86]. It has been chosen such as a compromise of SNR and pile-up probability: longer time constants would yield a larger signal, since a larger proportion of the incoming signal is stored on the capacitor before it discharges. This results in a better SNR. But, if the time constant is chosen to be too long, the probability for pile-up gets higher and may lead to an overdriven amplifier for large signals and high hit rates.

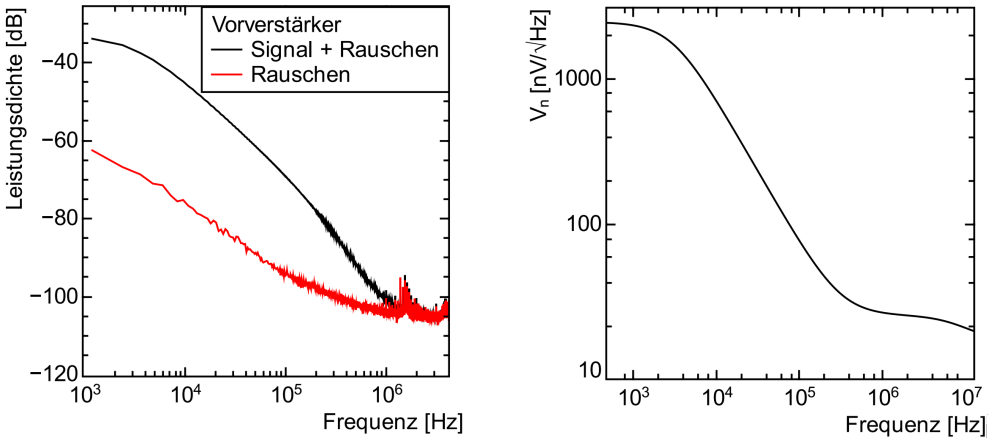
[10] $\pi$ stands for the p-light-absorption region.
[11] Schematic Plan with project number 917

Each APD signal has a fully independent preamplifier circuit. An adder, which is controlled externally via an $I^2C$ bus, allows to turn off each of the two APD signals if one APD or preamplifier fails and/or exhibits too much noise. This lowers the probability of failure per crystal, and theoretically lowers the noise level by $\sqrt{2}$ (cf. [61] section 5.4.5.1) if both APD signals are added. It was shown that this method is unfortunately not reliable, as the used op-amp [87] allows large signals to pass the disable-circuitry through protective diodes.

The preamplified, added signal is fed to a symmetrical line driver, which can drive up to 4 V on 100 Ω terminated twisted-pair transmission lines. It must be noted that the signal is not truly differential: to achieve lower power consumption in the idle state, the common mode voltage is set close to the quiescent level of the preamplified signal. This will later justify the need for a baseline-shifting circuit for the CB-SADC (see section 6.1.2).

**Power Spectral Density** For this thesis, a relevant information is the power spectral density of this configuration, which has been measured with and without scintillation signal in [61]. Figure 2.13 shows not only this measurement, but furthermore a simulation with the SPICE framework, done by M. Steinacher (University of Basel).

The measurement implicates that there is no prominent signal contribution above 1.5 MHz. The simulation shows a spectral noise density of 25 nV $\sqrt{\text{Hz}^{-1}}$ at this frequency. The frequency limit will become relevant for the determination of the input filter configuration of the SADC analog input stage in section 6.1.1, and furthermore for considerations regarding the sampling frequency and the possible decimation of the sampled data, whereas the spectral noise density can be taken as a rough limit for noise contributions of other components.

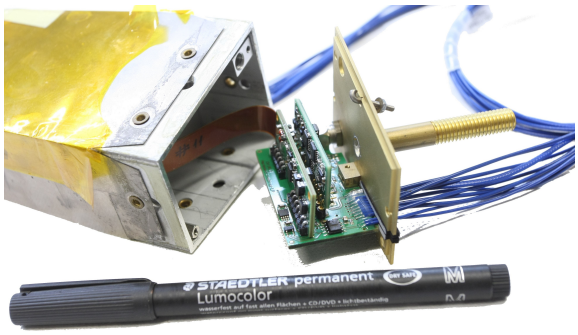



(a) Power spectral density of the preamplifier for noise and signal+noise (30 MeV to 50 MeV deposited). The signal contribution is negligible above 1.5 MHz.

(b) Simulated spectral noise density of the preamplifier circuit with connected APD, showing ≈ 25 nV $\sqrt{\text{Hz}^{-1}}$ at 1.5 MHz.

Figure 2.13.: Investigations of the noise density of the APD front-end signals, comparing measurement with simulation. [61]

(a) CsI(Tl) crystal in its housing, with the APD front-end readout inserted. The SAMTEC connector is plugged into the Slice Board.

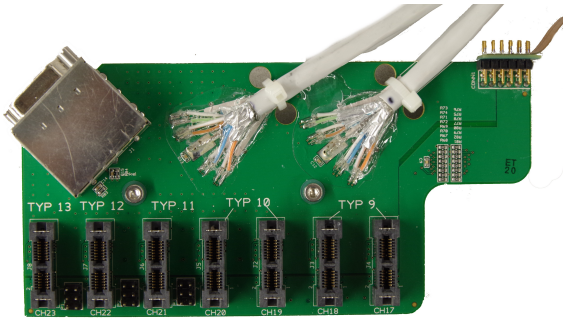



(b) Housing opened, exposing the APD front-end readout. The flexible connection is attached to the two APDs. The two upright boards are the Preamplifier and the High Voltage Regulation. The Linedriver is part of the main board.

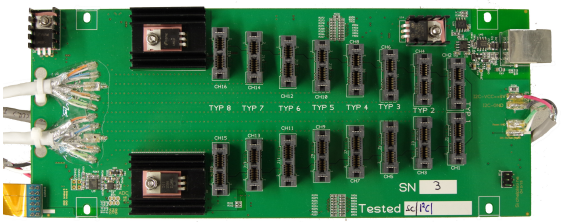
Figure 2.14.: Front-end Electronics attached to the CsI(Tl) crystal, developed in [61].

### 2.8.4. Slice Board

Sixty so-called Slice Boards (referring to the orange-like slice structure of the Crystal Barrel Calorimeter) provide grouping of the electronic signals for 21 or 23 crystals of each slice, allowing for sturdy and neat cabling of the Crystal Barrel Calorimeter. The front-end readouts of the crystals are connected to the Slice Boards by a custom SAMTEC connector, carrying the analog (APD) signal, low voltage, and slowcontrol ($I^2C$) signals in a flexible cable assembly (see Figure 2.14). The Slice Board, shown in Figure 2.15, distributes low voltage and slowcontrol signals from the central components to the front-end readouts, and consolidates up to 23 analog signals into a sturdy MRJ21 cable (without any buffering circuits). Sensors on the Slice Board allow monitoring of voltages and temperatures and are, together with the slowcontrol signals of the front end, transmitted by a special circuit via CAT5 cables to the central slowcontrol system (cf. section 2.11.2 and [88]). Distribution of the high voltage for the APDs is done by a separate small board for every slice.




(a) First board with plugs for seven crystals and an MRJ21 plug to connect to the BuffTi (left).




(b) Second board with plugs for 16 crystals, and slowcontrol and power connections (right).

Figure 2.15.: Slice Board, due to mechanical constraints split in two parts. The two gray cables and the flex cable connect the two parts. Developed by C. Honisch.

### 2.8.5. Buffer/Timing Filter and Energy Sum

The CsI(Tl) signal has to be distributed to three further stages (cf. Figure 2.11): The energy branches (QDC and CB-SADC) and the timing branch (TDC/Cluster Finder). To achieve this distribution, the signal is actively split into three paths within the so-called BuffTi modules (Buffer/Timing Filter) that are mounted in six upward-facing Nuclear Instrumentation Module (NIM) crates left and right of the Crystal Barrel Calorimeter.

As an interconnect between systems with different channel densities, one has to find the least common multiple between the cabling systems of Slice Board (24), QDC shaper (8), SADC (16) and TDC (24), which is indeed 48. This results in two Slice Boards being connected to one BuffTi module, which then delivers the signal to six QDC shaper cables, three SADC cables, and two TDC cables. Note that in this grouping some channels will inevitably not carry a signal (as only 21 or 23 crystals are connected to the Slice Board), e.g. for the forward half of the Crystal Barrel Calorimeter, two slices carry 46 signals, which will be distributed to $3 \cdot 16 = 48$ SADC channels, thus two channels will stay without signal.

The input signals are buffered on the BuffTi module by THS4130 op-amps with a bandwidth of 150 MHz, and a voltage noise contribution of less than $2\,\mathrm{nV}\,\sqrt{\mathrm{Hz}}^{-1}$ for frequencies above 100 Hz [89]. The output drivers are summarized in the following:

**QDC** Two MRJ21 connectors distribute the signals to adapters which allow to feed the signal through the QDC Shapers to the QDC readout via the previous cabling system (three 2x17 pinheader connectors for eight signals each).

**SADC** Three MRJ21 connectors (carrying only 16 differential signals each) connect to the SADC setup. The signals are buffered with TI OPA2613 op-amps in unity gain configuration with a bandwidth of 230 MHz, and a voltage noise contribution of less than $2\,\mathrm{nV}\,\sqrt{\mathrm{Hz}}^{-1}$ for frequencies above 1 kHz [90].

**TDC** Two MRJ21 connectors are fed with a high-pass filtered, amplified signal, which is used for the TDC/Clusterfinder readout. The high-pass filter yields a fast rising signal, necessary for the time-critical processing for the trigger decision. The relation of pulse height to deposited energy still exists, but the accuracy is degraded.

**Fast Energy Sum** Two extension boards in the BuffTi module create analog summation signals of the high-pass filtered TDC output signals, the so-called fast energy sum, resulting in one signal for each of the 60 slices. This signal is planned to be used for an energy sum trigger, for a scenario where one wants to have a trigger condition depending on the overall energy deposition in the Crystal Barrel Calorimeter (e.g. 100 MeV deposited in the whole barrel). According to preliminary investigations, this will allow to suppress events stemming from the photoproduction of the high-cross-section $\Delta$-resonance already on the trigger level, leading to a higher lifetime and less storage space requirement.

For diagnostic purposes, a copy of this signal is fed to the SADC connector (this is possible due to the fact that max. 23 of 24 channels on the slice board carry a signal). The analog input filters of the CB-SADCs are modified to accommodate the faster rise time of this signal. Investigations regarding functionality and calibration of the Fast Energy Sum have been conducted in [91]. The final revision of the Fast Energy Sum is planned to be installed in the BuffTi modules and implemented into the trigger system in 2019 [92].

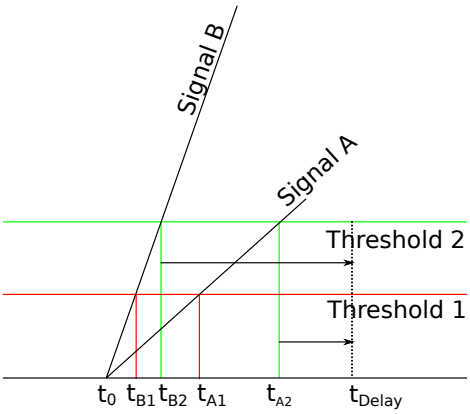
### 2.8.6. Discriminator/TDC/Cluster Finder

The Crystal Barrel Discriminator, which was developed by C. Honisch based on the ELB-VME-VFB [93] (a versatile SPARTAN-6 based data processing platform for the VME standard), is responsible for the timestamp determination of the Crystal Barrel Calorimeter's energy deposits, and furthermore for generating the trigger signal based on the number of detected energy deposit clusters. It features $4 \times 23$ analog input channels with dual threshold discriminators. The discriminators convert the analog signals that have passed the BuffTi's high-pass filter (see previous section) to a digital signal, indicating that a certain threshold is crossed; this principle is called *leading edge discrimination*. The position of the threshold crossing relative to the absolute position of the signal depends on the amplitude, which is referred to as *time walk*. Thus, a second threshold is used for compensation (see Fig. 2.16a). The discriminated signals are processed by the SPARTAN-6 FPGA for determination of the timestamp and the recognition of the hit pattern in the Crystal Barrel Calorimeter [94]. Both will be explained in the following paragraphs.

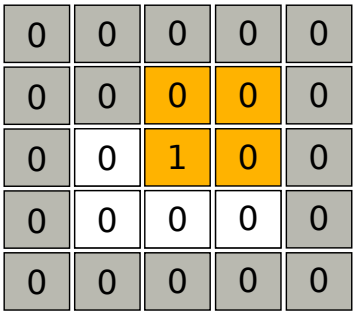
**TDC** A fully synchronous[12] multi-hit TDC is implemented in FPGA logic. The discriminated signals are received by the FPGA with a sampling rate of up to 800 MS/s, allowing a quantization of 1.25 ns. This fully synchronous firmware is developed in the scope of the dissertation of P. Klassen [94]. Alternatively, it is possible to implement asynchronous carry-chains, where the quantization is contingent on the transmission delay inside of, and between, the logic elements in the FPGA. With this technique, picosecond resolution is possible, but it requires very careful design and manual constraining and placement of the logic elements. The drawback is that with any change in the firmware the routing inside the FPGA is likely to change, requiring a tedious re-calibration of the system. Hence, whenever its resolution is sufficient, a synchronous system is strongly preferred, as it behaves deterministically. While the Crystal Barrel Discriminator is used with its fully synchronous firmware as a TDC for the Crystal Barrel Calorimeter, it is used for many more detectors (e.g. the tagging hodoscope, the Forward Plug Veto) in the CBELSA/TAPS experiment with the asynchronous carry-chain firmware, since a higher resolution is beneficial in those cases.

**Cluster Finder** The trigger decisions for the CBELSA/TAPS experiment mainly rely on counting the number of energy deposits from the final state photons in the calorimeters (also called *multiplicity*). Due to the nature of the scintillation process in the CsI(Tl) crystals, as well as their geometry, the electromagnetic shower generated by one photon usually spreads over multiple crystals. It is the task of the Cluster Finder to identify all such hit patterns and count the correct number of energy deposits, fast enough to be taken into account for the trigger decision. The correct counting of clusters is not trivial, as one has to find a method to separate touching and overlapping clusters stemming from different particles. It was shown in simulations [95] that the pattern presented in Figure 2.16 never underestimates the number of clusters, and overestimates it only in less than a few percent of all events (e.g. in 3.5 % of the simulated cases, three instead of two clusters were counted [61]). The implementation of this algorithm in an FPGA is feasible and indeed allowed the integration of the Crystal Barrel Detector into the first-level trigger.

[12] In the context of FPGAs, *synchronous* refers to the usage of clocked flip-flops in the logical design.




(a) Dual-threshold discrimination. The correct starting point $t_0$ can be extrapolated depending on the time between the threshold crossings $t_2 - t_1$.




(b) Cluster finding algorithm. If the pattern (white) touches the corner of a cluster (orange), it is counted (visualized by the 1).

Figure 2.16.: Visualization of the dual-threshold technique and the cluster finder algorithm. The L-shaped pattern is aligned such that the longer side goes along the beam axis. *Courtesy of Peter Klassen.*

### 2.8.7. QDC Shaper

The analog signal from the BuffTis passes through the QDC Shapers before being digitized in the QDCs. The QDC Shaper is a band-pass filter, which shapes the more than 100 µs long, exponentially decaying preamplified CsI(Tl) signal to an almost Gaussian-like signal with a Full Width Half Maximum (FWHM) of approximately 4 µs, giving it a suitable length for the digitization in the QDC (see next section).

The QDC Shaper has been developed at the University of Zürich in 1989 [96] and consists of three high-pass filters, two of which have an adjustable pole-zero cancellation (cf. section 6.1.3), and three low-pass filters. It has a low-gain and a high-gain output, as two different digitizers had been used previously; today, only one output is used. The pedestal or baseline of the output, which is the quiescent level of the signal, can be adjusted to a suitable value. Intuitively, one would expect this level to be adjusted to zero; in reality this leads to problems, as the baseline is not stable: first of all, electronic circuits usually exhibit temperature dependencies, which in this case will lead to fluctuations and even drifts of the baseline. Secondly, it may happen that a signal undershoots, for example due to mismatched or non-existing pole-zero cancellation in an AC-coupling. Both cases may lead to values below the zero level, which cannot be adequately digitized by the uni-polar QDC. Therefore, the baseline is set to a level above zero, and its value has to be subtracted in the analysis.

After the APD upgrade of the Crystal Barrel Calorimeter, the pole-zero cancellation, baseline, and the gain have been carefully readjusted with the help of the new light pulser system. It was shown in [97] that indeed the QDC Shapers are still suitable to be used after the APD upgrade, with a changed feedback time constant of 54 µs. The peaking time of the filter is not far from the ideal (2.9 µs and 2.3 µs respectively) and the dynamic range of the pole-zero cancellation allows ideal cancellation, although being close to the limit.

### 2.8.8. QDC

The shaped CsI(Tl) signals are passed on to the LeCroy 1885F ADC [98], a 96-channel 12-bit Fastbus ADC, which, due to its integrating character (cf. section 1.3.5) is also called a QDC. The test, the installation and the readout of this QDC are described in [99]. Changes in the readout system are documented in [100].

The functional principle of this QDC is as follows: the shaped CsI(Tl) signals are divided into three parts with a ratio of $8:1:1$[13], allowing the QDC to work in an $8:1$ dual-range setup.[14] The first two paths charge two capacitors for the time of an externally applied gate, which is started by the trigger system. This gate is set to a length of 6 µs.[15] Afterwards, the capacitor's voltage is digitized. Usually it is automatically decided with the help of comparators which of the two capacitors will be used for the digitization (auto-range), but the QDC can also be fixed to work in either low or high range.

The QDC does not feature a dedicated ADC for each of the channels. Instead, all 96 channels are multiplexed onto a single 12-bit ADC (Datel Adc-521mc [101]), which is a Digitally Corrected SRA (see section 1.3.2) with two 7-bit FADCs. It is crucial for this thesis to inspect the dead time of this QDC, as it will compete with the new SADC readout: after the integration gate, a *Measure Pause Interval* of 10 µs allows to inhibit the conversion, which has been facilitated as a *Fast Reset* in the old two-level trigger scheme of the CBELSA/TAPS experiment. The conversion itself takes 265 µs for all 96 channels, followed by approximately 150 µs to 180 µs transfer time[16] from the Fastbus via a Sis 4100 Ngf [103] sequencer to the VME CPU's memory. Via the LEvB running on the VME CPU, the QDC is connected to the experiment's data acquisition.

Summed up this limits the theoretically possible readout rate to less than 2 kHz, making the QDC the limiting component of the Crystal Barrel data acquisition. It will be shown that one benefit of the SADC readout presented in this thesis is to surpass this limitation.

## 2.9. Trigger

The CBELSA/TAPS experiment is dependent on a central trigger system to start acquisition and recording of data from all detector components. This stands in contrast to *triggerless* experiments like PANDA, where data is continuously sent from all detector readouts to a central event-building unit that qualifies all information just-in-time. The three criteria invoked for the event qualification will be shown in section 2.9.1. The timing constraints that led to the previous two-level trigger are explained in section 2.9.2. The efficiency of the trigger, especially comparing the situation before and after the inclusion of the Crystal Barrel Calorimeter into the first-level trigger, will be discussed in section 2.9.3. The trigger electronics and signals will be explained in some more detail in section 2.9.4, which allows a better understanding of the CB-SADC's integration into the readout chain (discussed in section 6.4).

[13] The last $\frac{1}{10}$ is ungated and used for internal pedestal correction.
[14] As the factor of 8 corresponds to 3 bit, the LeCroy 1885F ADC is commonly referred to as a 15-bit QDC.
[15] It has been shown in [99] that this length is indeed possible, although the QDC is specified only up to a gate length of 2 µs.
[16] Depending on sources, the transfer time lies within that interval. [99, 100, 102]

### 2.9.1. Event Qualification

It is important to carefully select the algorithms/criteria which trigger the experiment, such that the dataset is of high quality regarding the physical processes that are meant to be investigated. Unwanted *background* events should be excluded from being recorded to storage, decreasing the dead time and required hard disk space. Decisions in the CBELSA/TAPS experiment are made based on three criteria:

**Energy deposition:** It is demanded that the energy deposited in the calorimeters surpasses a threshold. One has to distinguish between single-crystal thresholds, cluster thresholds and threshold for the sum of the whole calorimeter (the new *Fast Energy Sum*).

**Multiplicity:** After consolidating neighboring hits in the crystals, the number of clusters can be determined and taken into account. This is done for both calorimeters, the Crystal Barrel Calorimeter and MiniTAPS, independently.

**Charge:** A hit which is identified as a charged particle can serve as a trigger, for example in the *Inner Detector* that may detect the recoil target proton, or as a trigger veto: the veto signals of the Forward Plug Veto and the Čerenkov may be used to suppress electromagnetic background, which is boosted in the forward direction.

### 2.9.2. Trigger Timing

The generation of a trigger signal has to occur within a certain time window after the event, since the data retention of the readout electronics is limited. Some systems can buffer the data with internal (digital or analog) buffers, but sometimes a delay has to be introduced by long cabling, for example in case of the QDCs.

In the previous setup, this limit was approximated to $\approx 300\,\text{ns}$ [61]. Most components regarded in the trigger decision, namely the Inner Detector, MiniTAPS, the Gas Čerenkov Detector, the Tagging Hodoscope and the former PMT readout in the three most forward rings of the Crystal Barrel Calorimeter (called Forward Plug) were able to meet this limit. The Crystal Barrel Calorimeter, on the other hand, could not deliver the multiplicity information as quickly as that. Responsible for this was the old Cluster Finder, the predecessor of the new one discussed in section 2.8.6. In its first version, called Fast Cluster Encoder (FACE), it needed around 100 µs for the determination of the hit pattern [104]. This was improved by two orders of magnitude with a cellular logic Cluster Finder to $t_{CL} = (n+1)\cdot 0.8\,\mu\text{s}$ for the determination of $n$ clusters [105]. Still, it did not meet the criterion of 300 ns, therefore a two-level trigger scheme was used, with the Cluster Finder being the only component of the second level; this is visualized in Figure 2.17. The second level was invoked in cases where detectors in the first level triggered, but the demanded multiplicity was not obtained without information from the Crystal Barrel Calorimeter. In cases where including the FACE did not result in the demanded multiplicity, a so-called *Fast Reset* was issued to inhibit the acquisition of the QDCs; this reduced the dead time greatly due to the QDC's considerable conversion and transfer time.

The old trigger scheme is described in more detail in chapter 3 of [106].

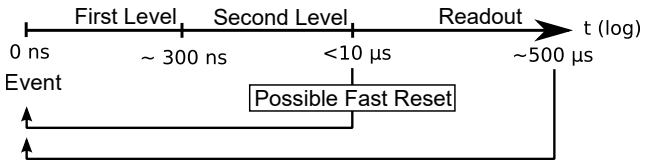



Figure 2.17.: Two-level trigger before the APD upgrade. All components, except for the Crystal Barrel, delivered a trigger signal within $\approx 300\,\text{ns}$. Its old cluster finder (FACE) was allowed up to $10\,\mu\text{s}$ to determine the multiplicity. If no clusters were found, a *Fast Reset* was issued to inhibit digitization in the QDCs. [61]

### 2.9.3. Trigger Efficiency

In the new setup, the second-level trigger is omitted as the new Cluster Finder in combination with the APD readout can deliver a much faster trigger and multiplicity signal, and hence the Crystal Barrel Calorimeter is fully integrated into the first level of the trigger. Before this improvement, the CBELSA/TAPS experiment could recognize reactions without charged particles in the final state (like for example $\gamma n \to \pi^0 n \to \gamma\gamma n$) only with limited efficiency. This is shown in Figure 2.18. The further the trajectory of the $\pi^0$ is in the forward direction, the higher the probability that at least one of the decay photons can be detected by the Forward Plug or MiniTAPS, yielding full trigger efficiency. If the trajectory is in the backward direction, the recoil neutron will go in the forward direction and could be detected with a limited efficiency of $\approx 25\,\%$ by the Forward Plug or MiniTAPS. In between, it is kinematically less and less likely that any of the particles reach a detector that is part of the first trigger level. The full line demonstrates how the trigger efficiency for neutral final states improves, now that the full Crystal Barrel Calorimeter is integrated in the first-level trigger: close to $100\,\%$ are reached for almost the complete angular coverage.

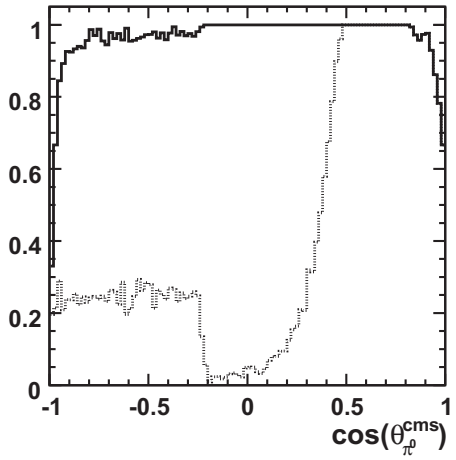



Figure 2.18.: Trigger efficiency for the reaction $\gamma n \to \pi^0 n$ with an incident photon energy of 2 GeV, depending on the polar angle of the neutral pion's trajectory in the Center-of-Mass System. The dashed line shows the situation without the Crystal Barrel Calorimeter in the first-level trigger (before the APD upgrade). [107]

### 2.9.4. Trigger Electronics

A custom trigger and synchronization system has been developed for the CBELSA/TAPS experiment within [100], with later modifications and enhancements [108]. A UML diagram in Figure 2.19a shows the sequence of this trigger system, which will be explained in the following. The header shows the four stages of the trigger, from right to left:

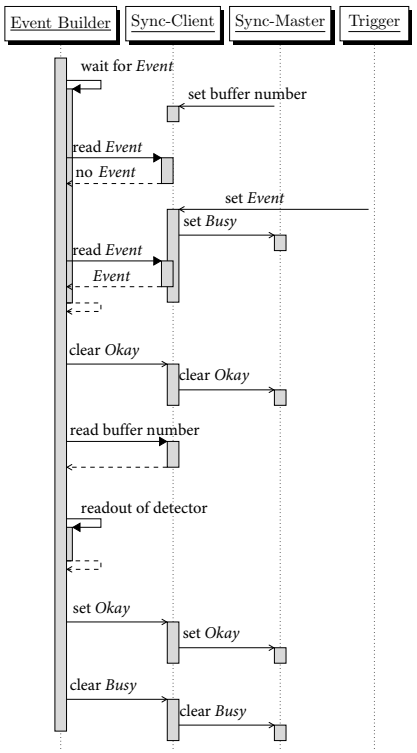
**Trigger** The trigger signals from the detectors are collected in a central trigger unit that generates the experiment's trigger signal based on the programmed event qualifier. This trigger signal is commonly referred to as the *Event* signal.

**Sync Master** The distribution of the *Event* signal and the synchronization of all readout systems are achieved with FPGA-based VME boards called the *Sync Master*. They receive the *Event* signal and the *Sysreset* signal from the *trigger*, and distribute it via a CAT5 connector (using LVPECL) to the *Sync Clients*. In return, they receive the *Busy* and *OK* signal on the same connector, indicating the state of the Sync Client. In addition, the *Sync Master* distributes the *Sync Bus*, which is explained later in this section. Figure 2.19b shows a schematic representation of the Sync Master VME board.

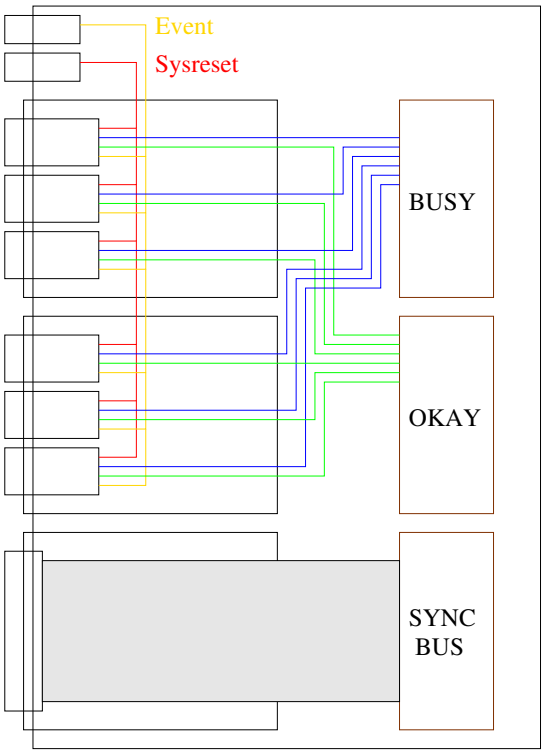
**Sync Client** Each detector readout system of the CBELSA/TAPS experiment is controlled by a Sync Client (also using a FPGA-based VME board). The *Sync Client* communicates with the LEvB via the VME interface.

**Local Event Builder (LEvB)** A daemon running on a VME CPU is responsible for collecting the data from the detector readout. All Local Event Builders send their data to the main Event Builder, the *Event Saver (evs)*.

The *Busy* signal is set by each *Sync Client* as soon as it receives the *Event* signal. This blocks the *trigger* from generating further *Event* signals. When the event is processed by the associated LEvB, the *OK* signal is set and the *Busy* signal is reset. As soon as this is true for all *Sync Clients*, a global *Sysreset* signal is issued, resetting all components to a well-defined state and opening the *trigger* gate for new events. In principle, this is sufficient for a synchronous trigger system, if the implementation in software and hardware is correct and there are no problems in handling the signals. As this cannot always be guaranteed, considering reflections and crosstalk in faulty or non-ideal connections, a 5-bit counter is transferred from the *Sync Master* to all *Sync Clients*. This counter is referred to as *Buffer Number*. It is incremented with every event, after it reaches 31, it goes back to 0. If one of the detectors misses an *Event* signal for any reason, or receives an odd *Event* signal, the counters will get out of sync, and the *Event Saver* will stop the readout. This *Buffer Number* is distributed on the so-called *Sync Bus*, a 34-pin flat ribbon cable, with Emitter-coupled logic (ECL) signal standard. The pin assignment of the *Sync Bus* is shown in Table 2.1. Apart from the *Buffer Number*, two more important functions are realized with the sync bus. First, a 3-bit *Event ID* can be used to classify the event, such that the readout electronics may react differently to different kind of events. This is being used, for example, for the so-called *Scaler Events*, which are not related to the experiment trigger, but are used to regularly read out internal counters in the LEvBs. Second, with the 5-bit *Address* bus, it is possible to activate and deactivate individual LEvBs. The implementation of this *Sync Bus* for the CB-SADC will be explained in sections 6.4 (hardware) and 7.8.2 (firmware).




(a) Trigger sequence in the chain from the trigger to the LEvB. The issuing of the Sysreset by the trigger is not shown, it follows immediately after the buffer number is set.




(b) Schematic of the VME *Sync Master* module, which receives the Event and Sysrest signals from the trigger system and distributes them as Low Voltage Positive Emitter-coupled Logic (LVPECL) signals via CAT5 connectors. The returned *Busy* and *OK* signals are transferred via the VME bus to the *trigger* LEvB, which also controls the *Sync Bus*.

Figure 2.19.: The trigger and synchronization system as it was developed within [100] and [108]. Figures from [108].

| Bit | 15-10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Mode 1 | free | Addr Sel 0 | | Buf-Nr Bit 4 | Buf-Nr Bit 3 | Buf-Nr Bit 2 | Buf-Nr Bit 1 | Buf-Nr Bit 0 | Event ID | Event ID | Event ID |
| Mode 2 | free | Addr Sel 1 | Cmd-Strobe | Addr Bit 4 | Addr Bit 3 | Addr Bit 2 | Addr Bit 1 | Addr Bit 0 | Event ID | Event ID | Event ID |

Table 2.1.: Signals of the 16 bit Sync Bus, which is transferred on a flat ribbon cable with ECL signals. The pin assignment changes when indicated by bit 9.

## 2.10. Data Acquisition

The current Data Acquisition (DAQ) of the CBELSA/TAPS experiment has, alongside the trigger electronics, been developed within [100] and [108]. An introduction into the DAQ system regarding the CB-SADC development is important, since its integration into the DAQ will be described in section 8.4. Structure and functionality of the components will be explained with the help of Figure 2.20.

**The LEvB daemons** are shown on the left-hand side, running on the respective VME CPU and processing the data from the associated ADCs and TDCs.

**The *DAQd*** (DAQ Daemon) runs on the `control2` server and is connected to the LEvBs through the *Control Network* (green). It can start and stop the LEvB daemons and can configure them following a set of `xml` files. Through the `control2` server it is also possible to access the control system of the ELSA accelerator.

**The *baevb* daemon** runs on the `cbvxt1`. It is responsible for communication with the trigger module and the Sync Master module, which can activate the Sync Clients of the detector's readout systems, as described in the previous section. The same VME CPU is also responsible for slowcontrol and light pulser.

**The *evs* daemon** (from *event saver*) receives the processed event data from all LEvBs via the *Data Network* (red). It checks the integrity and synchronicity of all packets, and sends the assembled data stream in the `ZEBRA` format [100, 109] via a dedicated 10 Gbit/s link to the storage system `cbdstorage1`. A different container format, `ROOT Tree`, is planned to succeed the `ZEBRA` format. On important advantage will a better and selective compressibility [35].

Two important user applications of the DAQ system will be explained in the following.

### 2.10.1. DAQ Terminal

The DAQ Terminal (*DAQt*) is the Graphical User Interface (GUI) that allows control over the DAQ Daemon (*DAQd*), or more generally speaking, over the whole readout system during a beam time. The *DAQt* can, in principle, be opened on any machine within the CB Network. The GUI allows the user to select the `xml` configuration files for all LEvBs, start and stop the run, and observe parameters and log messages from all components. If one of the LEvB daemons is in a faulty state, it can conveniently be restarted.

### 2.10.2. Online Monitor

A daemon called *explorad* is running on the `cbmonitor` workstation and fetches a fraction of the processed event data from the *saver3* during runtime. This allows to display graphical representations of the detector readout data in the so-called Online Monitor that was developed within [110]. The Online Monitor allows for just-in-time assessment of the data quality. The graphs are generated using the `ROOT`[17] environment, which is used extensively in the experiment's data analysis. The SADC presented in this thesis has been integrated in the Online Monitor within [111].

[17] https://root.cern.ch/

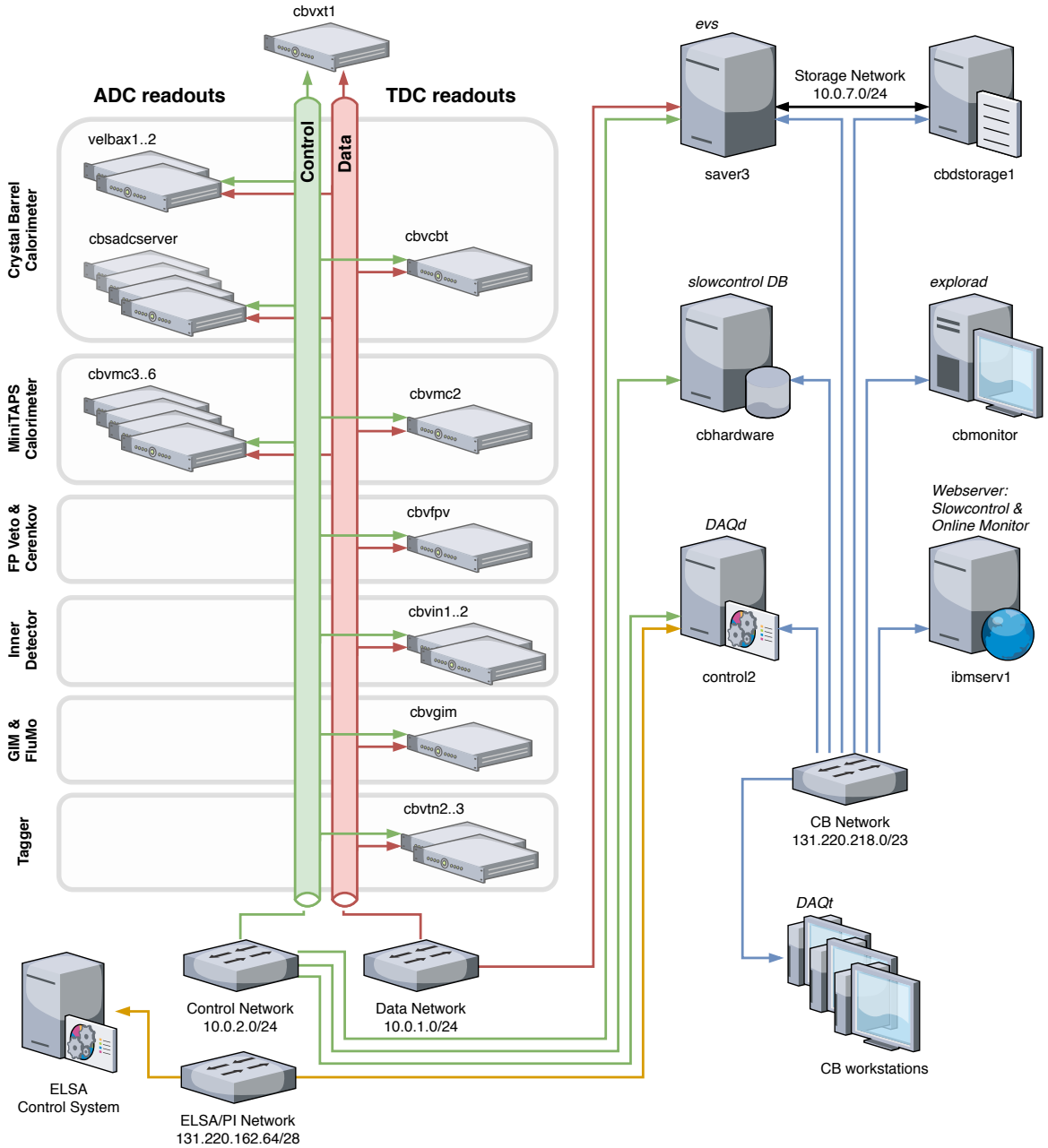



Figure 2.20.: Schematic of the CBELSA/TAPS experiment's network architecture. The left-hand side shows the VME CPUs responsible for the ADC (energy information) and TDC (timing information) readouts of the detectors. The right-hand side shows the involved servers, workstations and clients. The names of daemons are in *italic*. For this thesis, the `cbsadcserver` has been integrated into this scheme. Not shown: in the first implementation, the *cbsadc* LEvBs have had their own storage network and file server.

## 2.11. Auxiliary Systems

Two important auxiliary system, which neither fall in the *detector* nor the *readout* category, will be briefly introduced: the **light pulser**, used for calibration of the low and high range of the QDC, and the **slowcontrol**, which monitors and controls environmental and operational parameters in the experiment.

### 2.11.1. Light Pulser

For detector diagnosis and calibration of the dual-range QDC (see section 2.8.8), a xenon-flashbulb-based light pulser was installed at the CBELSA/TAPS experiment in the scope of [112, 113] and later maintained and upgraded within [114] and [115]. The light pulser was connected with fibers to the individual CsI(Tl)-crystals, such that the light flashes were then detected by the PIN photodiode.

For the APD upgrade, a method has been investigated for quasi-continuous gain monitoring using the light pulser system. This is necessary because of the temperature dependence of the APD's gain; and the light pulser can be used for an offline-correction of the remaining gain drift [116]. The method requires a high pulse repetition rate at different intensities to allow for measurements with sufficient statistics within the spill breaks of ELSA. This was not possible with the xenon flashbulb, having a pulse repetition rate of roughly 7 Hz. Thus, a new light pulser based on high-power Light Emitting Diodes (LEDs) has been developed and integrated into the experiment by M. Urban [84, 116], using the same fibers as the old light pulser. The LED wavelength and pulse form are optimized to mimic the scintillation light as close as possible, matching the spectral sensitivity of the APDs.

### 2.11.2. Slowcontrol

A centralized monitoring and controlling system allows to obtain an overview of the environmental and operational parameters of many components involved in the experiment. This system is referred to as the *slowcontrol*, as the data transmission happens very slowly compared to the high speed readout systems of the detectors. Examples for typical applications are:

- monitoring temperatures of crates and detector components,
- monitoring voltages and currents of power supplies,
- controlling the APD gain through high voltage setting,
- turning on/off power supplies,
- changing the position of the goniometer to select different radiators.

Various bus systems (like Inter-IC Communication ($I^2C$), Onewire, Profibus) are used for different components. All of those slowcontrol sub-systems store their data in an SQL-database [117] hosted on the `cbhardware` server that can be accessed via the web server `ibmserv` at https://slowcontrol.cb.uni-bonn.de/slowcontrol/ (cf. Fig. 2.20).

Latest changes to the slowcontrol include the integration of two new subsystems in light of the APD upgrade: a network of **Onewire** temperature sensors has been installed on the

mechanical frame of the Crystal Barrel Calorimeter to evaluate the new oil-cooling system [118]. Furthermore, a new FPGA-based **I²C** readout has been added to the slowcontrol [88] and was integrated into the web interface [119]. It was mainly designed for communication with the new front-end APD electronics. Due to the modularity and scalability of this new system it has also been used to integrate the CB-SADC (see sections 5.7.5 and 8.5) itself, and furthermore the light pulser and the BuffTi. The I²C protocol will be explained briefly in the following paragraph.

**I²C** is a synchronous open-drain bus consisting of two bidirectional signals: SDA (serial data) and SCL (serial clock). All communication is started by a *master*, which can transmit or request data to/from the *slaves*. Figure 2.21 shows an example of possible topologies and slaves. The master can be a microcontroller, processor, or FPGA. The bus can be split by switches, which allow to resolve address conflicts. Moreover, the bus can be buffered to overcome the limit of the wire capacitance (400 pF). Not shown in the scheme is the possibility to galvanically isolate bus segments, which is used if electronics with different ground levels have to be connected.

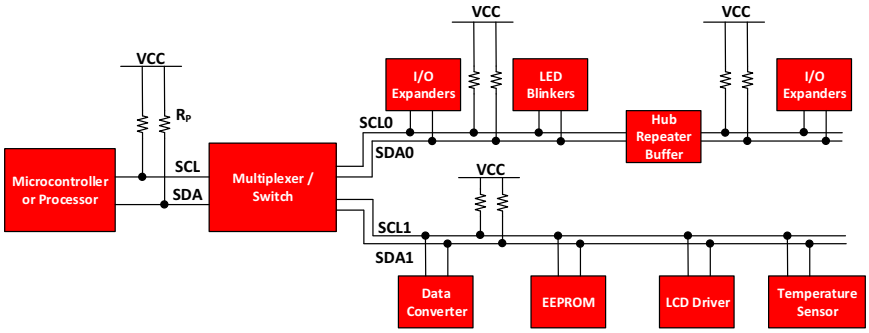



Figure 2.21.: Example of an I²C bus. [120]

Figure 2.22 shows a typical I²C transmission: the slave is addressed with the first byte of every transmission, consisting of the 7-bit address and one bit indicating whether it is a read or write operation. The next byte(s) represent the data written from the master to the slave. The address call, as well as every byte, has to be acknowledged by the slave, which will, for the duration of one bit, take control over the SDA signal.

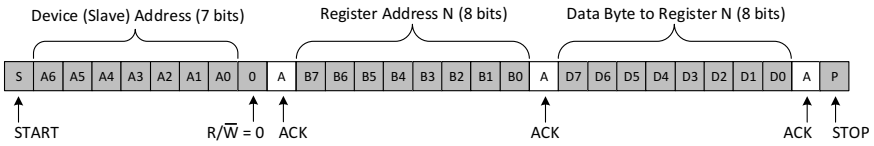



Figure 2.22.: Example of an I²C transmission. A master writes two bytes to a slave. [120]

I²C is usually used to interconnect components within one device. In the CBELSA/TAPS experiment, it is used across detector components, which surpasses the capacitance limit of I²C by far. This is why the individual I²C components in the experiment are equipped with special driver circuits (Texas Instruments Pca9600) that allow the transmission of the I²C signals over more than 100 m using CAT5 cables [88, 121].

# 3. SADC Upgrade

This chapter will present the preparatory considerations for this thesis, as well as a project timeline. It will start with the motivation for an FPGA-based SADC in section 3.1. In section 3.2, previous studies of the CBELSA/TAPS Collaboration, which were evaluated for this thesis, will be summarized. The requirements for the SADC upgrade, and an assessment of commercial SADC solutions, are then presented in section 3.3. Finally, an overview of the resulting timeline of the CB-SADC project will be provided in section 3.4.

## 3.1. Motivation

In sections 2.8.7 and 2.8.8, the current energy readout for the Crystal Barrel Calorimeter, the **QDC**, was presented. Although the system has proven to be reliable in the almost 20 years it has been operational and has met the requirements in regard to energy resolution, there are reasons advocating a new readout system. First of all, with the nature of the sampling technique and the processing of the data with an FPGA, access can be gained to previously unavailable methods to obtain **higher quality data** and **increased readout rate**:

**Event-based Pedestal Subtraction** All signals from the calorimeter crystals have a none-zero pedestal or baseline (cf. section 2.8.7). For the data analysis, the value of the pedestal needs to be known and subtracted from the measured signal, as this cannot be done by the QDC itself. Therefore, before every run, the baseline is determined using a so-called clock trigger (i.e. a trigger with a fixed frequency, uncorrelated to the experiment trigger). The drawbacks of this method are the influence of possible detector signals (the accelerator is not turned off for this measurement) and furthermore the drifts during one run due to rate dependencies and temperature changes. With an SADC, the determination of the baseline becomes possible for every pulse separately, just by *looking back* before the pulse, when the SADC is triggered. It is even conceivable to have a more elaborate algorithm that tracks the baseline over the duration of a run, and that detects if the pulse does not start from the baseline but from the tail of another pulse.

**Pile-Up Detection and Recovery** The rate of events in the Crystal Barrel Calorimeter depends on the configuration of electron beam energy, bremsstrahlung radiator, and target. At a certain rate it will become more and more likely that two pulses overlap. The QDC has no means of identifying or even recovering such events, instead it will integrate the pulse and hence provide a wrong energy information, which in the best case will be discarded in the kinematic analysis. This is quite problematic, as it can lead to angular dependent effects that are difficult to account for in the simulations of the detector acceptance. With access to a sampled representation of the pulse, it is quite well possible to at least detect such pulses (cf. chapter 6.2 in [122]), and in case

of a detection to provide and save the sampled representation to disk for later analysis. The question whether or not a complete recovery will be possible within the processing means of the FPGAs on the SADC board (as opposed to a recovery in the "offline" data analysis) might be answered in [111].

**Sub-Threshold Timestamps** The TDC of the Crystal Barrel Calorimeter can deliver timestamps only down to a certain threshold, which can be more than 10 MeV in the forward direction. The CB-SADC is not limited by such a threshold, and is expected to deliver timestamps down to the 1 MeV region.

**Increased Readout Rate** As pointed out in section 2.8.8, a readout cycle of the current digitizer, the QDC, takes almost 500 µs, limiting the theoretical readout rate to about 2 kHz. It is expected, and will be shown in a limited setup in section 9.9, that the new SADC will be a magnitude faster. This will immediately allow the CBELSA/TAPS experiment to improve the *statistics per beam time* ratio.

**Pulse Shape Analysis** A theoretically accessible information, which is not being exploited at the moment, is the discrimination of particles by the response of the CsI(Tl) scintillation light. This has been demonstrated in [123–125]; yet first attempts for the Crystal Barrel Calorimeter with the help of SADCs did not yield a sufficient discrimination power [126]. It seems that due to the analog signal shaping employed for the Crystal Barrel Calorimeter, parts of the characteristic scintillation light response cannot be reconstructed. Still, from the digitizer point of view, the way for more investigations will be paved, and further attempts will likely be presented in [111, 127].

Apart from the listed benefits for the data quality and quantity, additional more **technical aspects** are also in favor of a new readout:

**Maintainability** It goes without saying that at an age of more than 20 years for the QDCs and almost 30 years for the signal shapers, the maintainability is ever decreasing. It is questionable if the QDC could still be serviced by LeCroy at all, given that their regarding division is not existing anymore. Thus, only minor repairs can be expected to be conducted locally. The shaper, in principle, consists of standard discrete components that can be exchanged easily. Still, a number of failures have been observed, and can be expected to become more frequent in the future.

**Ease of Use** During preparations for the commissioning of the new Crystal Barrel Calorimeter setup with APD readout, the current shaper/QDC setup has been tuned in regard to baseline setting and pole-zero cancellation. Many man hours (in fact weeks) were spent on this manual and tedious procedure, which required adjustment of potentiometers with screwdrivers while watching the signal on an oscilloscope. This whole procedure is digitally controlled in the new SADC setup, therefore it is possible to automate it with a script. Once the procedure is developed, tested and programmed, a new tuning of the parameters will be a matter of minutes.

**Ressources Consumption** Not only does the old readout consume a lot of space (four 19-inch racks), it also has a high power consumption for the electronics and the cooling. From a financial and economical point of view, an optimization is desirable. The new solution will indeed fit into only two NIM-crates and consume less than 1 kW of power.

## 3.2. Previous Studies

One of the first steps of the CB-SADC project was the examination of previously conducted studies in the regarding field. Four theses from members of the Crystal Barrel collaboration have carved the way towards a new SADC readout and will be summarized in the following sections.

### 3.2.1. Studies of the Energy Resolution of CsI(Tl) with High-speed ADCs

Christian Funke has investigated the readout of CsI(Tl) signals with commercially available SADCs and QDCs in a laboratory setup [128]. He compared the LeCroy 4300b Fera (11 bit QDC) with the Struck Dl401 FADC (8 bit at 100 MS/s) and used a single CsI(Tl) crystal of the Crystal Barrel Calorimeter for measurements using cosmic radiation and light pulser signals.

As part of his measurements, he artificially reduced the data from 8 bit to 6 bit to get results comparable to Struck DL300 Flash-ADC (cf. next section), and he decimated the sampling rate from 100 MS/s down to 5 MS/s. He concluded that, in his setup, the SADC had a worse energy resolution than the QDC, and that the extensive amount of data produced by the SADC is extremely difficult to handle. As he identified the ability to separate piled-up pulses as the only real advantage of the SADC, he recommended to use SADCs only in the forward region of the Crystal Barrel Calorimeter.

### 3.2.2. Development of a VME Readout for the Struck Dl300 Flash-ADC

The old Jet-Drift-Chamber of the CB-LEAR experiment was read out by the Struck Dl300 FADC system, with a non-linear resolution of 6 bit (effectively 8 bit) and a sampling rate of up to 100 MS/s. The development of a new and faster readout for the Crystal Barrel Calorimeter was the goal of Torge Szczepanek's doctoral thesis [129]. He devised a readout that uses a special FPGA plugin-card for the VME CPU, allowing a readout rate of more than 1 kHz. The system was developed to be used in a dual-range setup only for the Crystal Barrel Forward Plug, and was known to have worse resolution for small pulses compared to the QDC readout. During the time of the development, the CBELSA/TAPS experiment was moved to a different area within the experimental hall, so the system has never been tested under beam conditions, nor has it been installed at the new experiment area at all.

### 3.2.3. Studies with the Sis 3320-250 Sampling-ADC

In his diploma thesis [126], Steffen Schaepe has made extensive investigations regarding the implementation of a Sampling ADC readout for the Crystal Barrel Calorimeter, following the two previously presented studies. He has used an Sis 3320-250 ADC with a sampling rate of 200 MS/s, which was connected to a second output of the Crystal Barrel Shaper, in parallel to the QDC readout. For the three most forward rings, which were read out with PMTs, the signal was additionally investigated without shaping. Testing the readout in conjunction with the complete CBELSA/TAPS experiment allowed high rate data taking resulting in high statistics, compared to laboratory setups that uses cosmic radiation. Moreover, the experimental environment allowed more realistic studies.

Investigations of the extracted time resolution of the calorimeter signals have been compared with the time resolution of the CATCH TDCs, which were commonly used in the experiment, yielding a comparable resolution. The energy resolution of the single-range readout, on the other hand, was not as good as the dual-range QDC readout, most probably because the analog signal filters where not optimized, and digital filtering was not implemented. A pulse shape discrimination was attempted, but the analysis of the data did not allow an event-based discrimination of protons and photons.

### 3.2.4. Studies of Feature Extraction Methods with the SIS 3320-250 Sampling-ADC

Philipp Mahlberg has investigated methods for the determination of the energy information from sampled pulses with a prototype of the new APD readout, using again the Sis 3320-250 [130]. He concluded that in general the APD readout is less noisy when compared to the PIN-diode readout, but pointed out that the determination of low energetic crystal signals (<10 MeV) will still be difficult without further analog filtering.

### 3.2.5. Qualitative Comparison of Commercially Available Sampling-ADCs

After and during the studies performed with the Sis 3320-250 ADC, three other commercial solutions were investigated [131]. First, an Sis S3316-125-16, for which Sis together with KVI[1] developed a specially prepared firmware for measurement of photons in scintillators, second a 14-bit SADC from Caen, and third the Wiener Avm16 that is also shown on the left in Figure 3.2. It was concluded at the time that the offered firmwares for the on-board FPGAs did not fulfill the requirements of the experiment. They were never open source, and the configuration possibilities were not flexible enough. This collided with the idea to implement *feature extraction* algorithms to be used in the experiment. In addition, all solutions had a very poor channel density, when compared to the solution that is developed within this thesis. This typically leads to a higher cost per channel, since infrastructure (like FPGAs, power supply, ...) is shared between a smaller number of channels, and furthermore to growing rack space requirements.

## 3.3. Requirements and Explored Solutions

After examination of the aforementioned studies and extensive discussions, possible solutions were elaborated. First of all, the minimal requirements for the Crystal Barrel SADC have to be specified. Naturally the project has evolved during this thesis, and so have the insights and also the constraints. The technical requirements can be summarized as follows:

**Bandwidth** An analog bandwidth of at least 1.5 MHz is required according to Figure 2.13, resulting in a sampling frequency of at least 3 MS/s according to the Nyquist–Shannon sampling theorem (section 1.4.6). At first sight, this might seem very low compared to what is commercially available. But a higher sampling rate than necessary can be used to oversample the signal, which allows to improve the SNR of the SADC (cf. section 1.4.3).

[1] Kernfysisch Versneller Instituut (Groningen, NL)

**Resolution** The actually required resolution is not easy to determine from the current setup, as it is difficult to compare the resolution of the QDC with the resolution of an SADC. The designed energy range is determined to range from below 1 MeV to 2.5 GeV. To *observe* a pulse with 1 MeV, a noise level of 0.3 MeV is a reasonable assumption, which would lead to dynamic range of $\frac{2.5\,\mathrm{GeV}}{0.3\,\mathrm{MeV}} \approx 8300$ ($\hat{=}$78.4 dB). This can be achieved by a measured variable with a noise-free code resolution of 14 bit (without oversampling), resulting in 153 keV/LSB (cf. section 1.4.4).

**Firmware** The firmware should either completely, or at least to some level of modularity, be under control of the user. As an example, it is planned to implement a pile-up recovery mechanism, which is certainly not provided by closed-source firmwares.

**Rack Space** At most, the equivalent rack space of 4 NIM or VME crates should be used, which is the free space in the rack on the movable carriage of the Crystal Barrel Calorimeter. If more space is required, a fixed rack has to be used, and complicated cabling that employs a cable chain is necessary.

**Cost** Limits to the costs are difficult to determine and have, in fact, not been formally set for this project. Presumably an amount of more than 100 k€ has to be expected, whereas moving closer to the 0.5 M€ region would bear no proportion.

**Timeframe** Since many studies have been conducted on the topic, the aim within this thesis is to finally develop, test, and implement an SADC system following the APD upgrade. Thus, at the end of this thesis, the system should be either fully or at least partly installed and in operation.

### 3.3.1. Commercially available SADCs

Investigating commercial solutions, one quickly realizes that companies focus on high sampling rates, which are necessary for fast signals from e.g. plastic scintillators or PMTs, but not for a slower CsI(Tl) signal with preamplifier characteristics as in the case of the Crystal Barrel Calorimeter. As the cost and size clearly scale with the sampling rate, this can be regarded as a negative point. The highest density solution from CAEN seems to be the 740 DIGITIZER FAMILY, with 64 channels in a VME module, which is a reasonable density for our endeavors. Unfortunately it is only a 12-bit architecture, which could be too low for a single-range setup, and a closed-source firmware. SIS on the other hand offers only up to 32 channels in a VME module, but technically suitable with 16-bit resolution and 125 MS/s. The firmware seems well developed and maintained, yet it is to be expected that this solution will be very expensive. Due to the very high cost, and almost no flexibility regarding in-house development of the firmware, a commercial solution has not been truly considered as an alternative. An option, which finally provided the basis for this thesis, will be presented in the next section.

### 3.3.2. Custom SADC for the PANDA Electromagnetic Calorimeter

After the previously summarized studies, the SADC plans gained momentum in connection with the development of an SADC for the PANDA experiment [132]. Before discussing the SADC itself, a brief introduction of the PANDA experiment will be provided.

**PANDA at FAIR** Proton Antiproton Annihilation in Darmstadt (PANDA) is a hadron spectroscopy experiment that will be installed at the High Energy Storage Ring of the Facility for Antiproton and Ion Research (FAIR). One of its goals is the search for exotic states in the charmonium mass region.

The PANDA-SADC is being designed for the Electromagnetic Calorimeter (EMC), which is shown in Figure 3.1. It consists of 11 360 $PbWO_4$ (PWO-II) scintillator crystals in the Barrel, plus 3856 in the Forward and 528 in the Backward Endcap [133]. The Barrel, the Backward Endcap and most of the Forward Endcap, will be equipped with an APD readout, similar to the Crystal Barrel Calorimeter. For the inner region of the Forward Endcap, Vacuum Photo Tetrodes (VPTTs) will be used to account for higher rates of up to 500 kHz, associated with higher radiation doses.

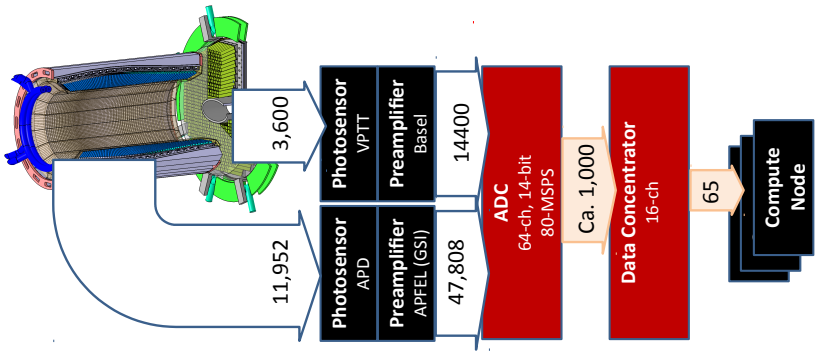



Figure 3.1.: The readout scheme of the PANDA EMC, consisting of more than 15,000 $PbWO_4$ scintillator crystals. The Barrel is shown in blue, the Forward Endcap in green (the Backward Endcap is not shown). All crystals will be read out with a dual-range setup. The mentioned numbers and partitioning were preliminary and have been adjusted during the planning phase of the PANDA experiment. [134]

The requirements of the digitizer for the PANDA EMC have been determined based on simulations, taking into account the maximum deposited energy, possible electromagnetic interactions, and mechanical restrictions (list not intended to be exhaustive). According to the Technical Design Report (TDR) [135] they can be summarized as follows:

- dynamic range of 12 000 (12 GeV maximum photon energy deposition, down to a noise level of 1 MeV), compared to only up to 7500 for the Crystal Barrel Calorimeter,
- sampling rate between 40 MS/s and 80 MS/s,
- time resolution below 1 ns,
- smallest possible form factor with low power requirements,
- radiation hardness (due to placement close to the detector).

The TDR mentions a first prototype [136] for the Advanced Telecommunications Computing Architecture (ATCA) standard, which was planned to be the reference for the following development commencing in Uppsala (Sweden). The progress between 2008 and 2013 is

shown in Figure 3.2. A first prototype had been sold to W-IE-NE-R and was not used further for PANDA. The second prototype (2011) was used for the first tests in the light of this thesis, which will be discussed in chapter 4. The third prototype (2012) already featured 64 channels and used the ADC chip which is still being used for the SADC in this thesis. One of the biggest changes for the fourth prototype (2013) was moving from the XILINX VIRTEX 5/6 to the more modern XILINX KINTEX 7 FPGA. The design sources of this last prototype have been made available by the developer Pawel Marciniewski [137] for custom modification within this thesis. Adaption and modification of this design will be presented in chapter 5.

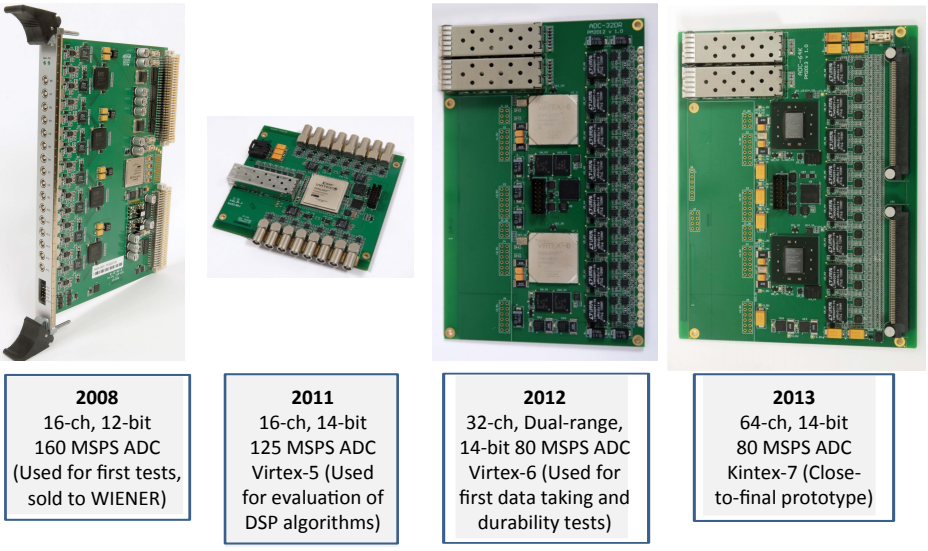



Figure 3.2.: History of the PANDA SADC development. [134]

## 3.4. Crystal Barrel SADC - Project Timeline

This section intends to provide a historically accurate timeline of the project, which is not necessarily reflected in the order of the chapters in this thesis. The list is sorted according to the start date of the related task. It includes six sub-projects of bachelor and master students that were connected to the SADC development; naturally, their results will also be shown throughout this thesis. At the end, an overview of the open tasks will be shown.

**Jan'14 to May'14** Setup of the 16-channel PANDA-SADC prototype for tests, subsequently development and test of first feature extraction algorithms (chapter 4).

**Jun'14 to Oct'14** Adaption and customization of the 64-channel PANDA-SADC schematics and PCB design; resulting in the first CB-SADC prototype (chapter 5).

**Nov'14 to Dec'15** Development of low-noise switching power supply (section 6.3).

**Nov'14 to Oct'15** Development of a slowcontrol module, which was the predecessor of the backplane (master thesis of G. Urff [138], section 6.4.1).

**Mar'15 (ongoing)** Firmware development (chapter 7).

**Oct'15 to Nov'16** Test of feature extraction algorithms (master thesis of J. Schultes, section 9.2).

**Nov'15 to Oct'16** Development of the analog input card (Bachelor thesis of T. Poller [139]). After the thesis was finished, work on the project was continued by T. Poller. This time allowed the additional integration of shaping and an $I^2C$ controllable pole-zero cancellation. The result is described in section 6.1.

**Aug'16 to Oct '16** Improvement and bug-fixing of the SADC and power supply design; resulting in the second CB-SADC prototype (sections 5 and 6.3).

**Sep'16 to Nov'16** Development of a backplane for slowcontrol/trigger distribution (section 6.4.2).

**Oct'16** The first CB-SADC prototype was running in a "parasitic" setup. It was not fully connected to the DAQ, but running in a self-triggered or non-synchronized-triggered mode. Only 23 of the 1320 CsI(Tl) where read out. Results are discussed in 9.3.

**Dec'16** Arrival and test of seven CB-SADC prototypes (revision 1.1).

**Apr'17 and Sep'17** Test beam times with partial CB-SADC readout.

**Oct'17 to Nov'18** Setup of a test procedure for quality assurance of the mass-produced SADCs (master thesis Y.-C. Wang [140], section 6.2).

**Dec'17** Commissioning beam time with partial CB-SADC readout, now concentrated to the forward region of the Crystal Barrel Calorimeter to investigate pile-up detection.

**Apr'18** First production beam time with partial CB-SADC readout (section 9.7).

**Aug'18** Arrival of first CB-SADC (production revision 1.2), followed by laboratory tests.

**Oct'18** Second production beam time with partial CB-SADC readout.

**Oct'18** Arrival and test of the remaining 25 CB-SADCs (production revision 1.2).

**Oct'18 to Oct'19** Development of a CB-SADC *Crate Controller* as a successor of the Backplane (master thesis of J. Knaust [141], section 6.4.3).

**Apr'19 to Jul'19** Development of improved algorithms for FPGA-based baseline determination (bachelor thesis of L. Zywietz-Rolon [142]) and investigations of particle identification capabilities (bachelor thesis of M. Gräf [127]).

**May'19** Arrival and test of the final revision of the Analog Input Card.

**Future** Some tasks for the complete installation at the CBELSA/TAPS experiment are still open. It is expected that they can be completed within 2019.

- Production, test, and installation of the CB-SADC Crate Controller
- Signal test of the assembled CB-SADCs with Analog Input Card
- Installation of all components, including cabling, at the experimental site
- Setup of the final `cbsadcserver`

# 4. A First Step: The 16-Channel PANDA-SADC Prototype

This chapter will summarize the progress and findings of working with the 16-channel PANDA-SADC prototype. It has been developed in 2011 by Pawel Marciniewski [132] as an experimental desktop platform for the development of Digital Signal Processing (DSP) algorithms (cf. Fig. 3.2). Likewise it was the first opportunity to get familiar with the FPGA firmware development also in the CB-SADC project.

Although the hardware is not of particular interest, some technical specifications will be summarized in section 4.1. The firmware development, leading to a first display of digitized waveforms, as well as a leading edge discriminator, will then be shown in section 4.2. Last, in section 4.3 the successful implementation of an Ethernet interface will be shown. Most explanations regarding hardware and firmware will not go into too much depth, instead the text will refer to the chapters where equivalent steps were taken with the 64-channel SADC.

## 4.1. Technical Specification

The 16-channel PANDA-SADC prototype is based on four 4-channel ADCs (LTC2175) with 14 bit at 125 MS/s that are connected to a Xilinx Virtex-5 FPGA (LX50T) for data processing. Unipolar single-ended signals can be connected with Lemo connectors and are symmetrized with either an operational amplifier (LTC6404) or (optionally) with a transformer. The board furthermore features the possibility to introduce a test pulse in the signal path, and to shift the baseline (both features were not explored within this thesis). After being processed by the FPGA, the data can be transferred off the board with a network interface.

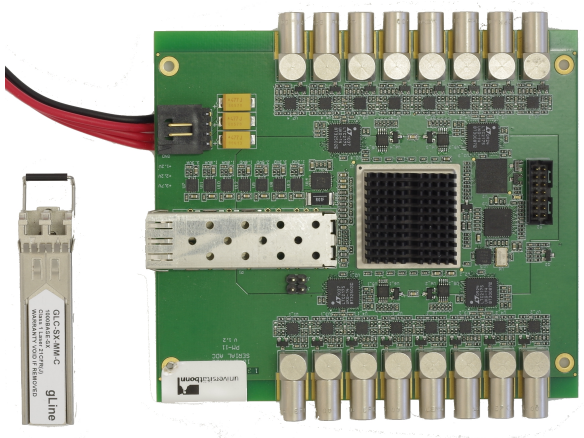


Figure 4.1.: 16-Channel PANDA-SADC with a gigabit fiber SFP (not plugged in).

## 4.2. Getting Started with Signal Processing

A basic firmware was supplied together with the prototype [132], giving a considerable head start for the initial operation. This firmware contained the logic to initialize the on-board clock manager (cf. section 7.2.2), and to de-serialize the data from the ADC chips (cf. section 7.3.1). This results in a continuous data stream of 16×14 bit at a rate of 125 MS/s, yielding a total raw data rate of 3.5 Gbyte/s. This data can now be manipulated by any means of data processing possible on the FPGA. Exploring this was a goal of this prototype.

Since a network interface had not been implemented yet, the data from the FPGA has been observed through a debugging interface (JTAG). An Integrated Logic Analyzer (ILA) was inserted into the firmware, allowing the inspection of signals in the FPGA through the software Xilinx Chipscope (cf. appendix B.3). With this it was possible to access the data stream and effectively view the sampled waveforms, which is shown in Fig. 4.2. Apart from the raw data, this also allows to observe the outcome of implemented signal processing that is the aim of this investigation. The first "extracted features", which are also visible in Fig. 4.2, where an *activity flag* and a *leading edge discriminator*. The leading edge discriminator allowed to implement a simple self-triggering mechanism, which has proven to be very helpful for laboratory setups that did not allow the connection of an external trigger signal. In addition (but not shown here) a constant fraction discriminator and an integrator were implemented.

Overall, this first *hands-on* experience has built a good basis for the future implementation of feature extraction algorithms, which will be shown in section 7.5.

## 4.3. Implementation of a Network Interface

The most important step to be achieved with the 16-channel PANDA-SADC prototype was implementing an interface to transport the processed data off the board using the Small Form-factor Pluggable (SFP) port. Possible protocols for this data transfer are (amongst others) the well-established standards Ethernet, Fibre Channel, PCI Express, or proprietary protocols like Aurora (Xilinx). It was chosen to focus on Ethernet, since it was expected that also in the experiment setup the data from the SADCs will be sent to a computer using regular Ethernet switches.[1]

**Implementation and Test Setup** The implementation of a network interface in the FPGA will be explained in more detail in section 7.4.1 (an overview is provided in Figure 7.16 on page 155) and will only be summarized here. The implementation consists of the transceiver (*physical layer*) and the media access controller (MAC, *data link layer*). The transceiver was implemented using the Ethernet 1000BASE-X PCS/PMA IP Core [143], followed by the Virtex-5 Embedded Tri-Mode Ethernat MAC Wrapper (*TEMAC*) [144]. The most notable difference in this case is that the Virtex-5 architecture has embedded, dedicated logic for the *TEMAC*, whereas the Kintex-7 will need a so-called *softcore* implementation (costing logic resources in the FPGA).

[1] Opposed to that, an intermediate stage will be introduced in the PANDA SADC readout: an FPGA-based *Data Concentrator* (cf. Fig. 3.1) will be used to further reduce/compress the data, before it is forwarded to *Compute Nodes* for storage/analysis. This three-stage design is necessary due to the concept of the free running readout (no central trigger).

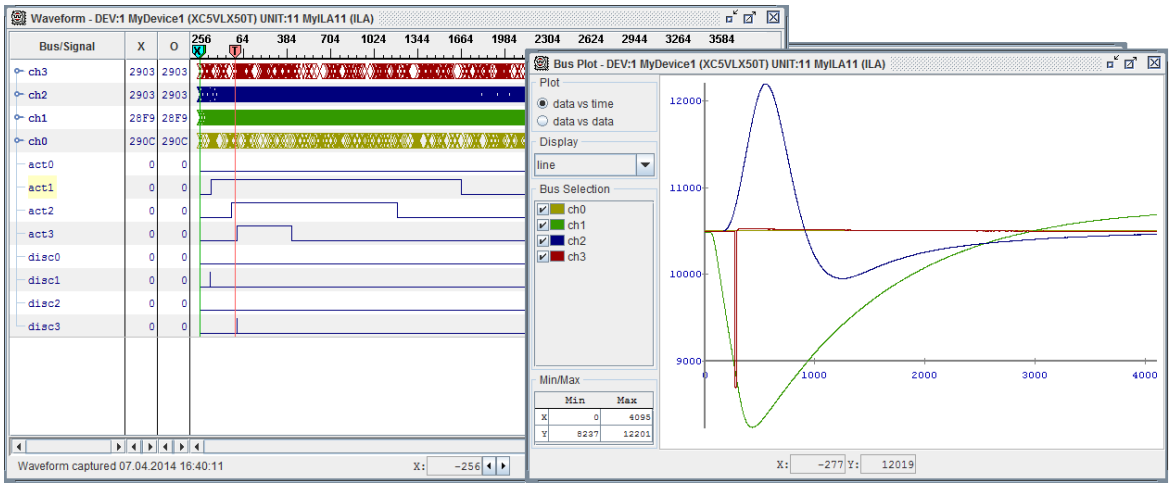



Figure 4.2.: Screenshot of the first pulses that were observed in a laboratory setup with Chipscope, using the 16-channel PANDA-SADC prototype. The snippet shows the digital data in the *Waveform* window and graphs in the *Bus Plot* window. An arbitrary pulse from a waveform generator was connected to three channels after passing different shaping options (none of which are actually preferable). The signal on `ch1` shows a long overshoot of the tail due to mismatched pole-zero cancellation, `ch2` has almost bipolar characteristics. The very short signal on channel `ch3` has been used as a trigger pulse. Apart from the waveforms, one can see the activity flags (`actX`), coarsely indicating the signal duration, and the leading edge discriminator (`discX`) that recognizing the negative pulses.

To test for a successful implementation, it was chosen to use a XILINX SP605 evaluation board with XILINX SPARTAN-6 FPGA as a receiving end (see Fig. 4.3). Using the evaluation board instead of directly connecting to a computer allowed an investigation of the connection on the lowest level: after initial difficulties, so-called *loopback testing* helped to identify problematic parameters. In a loopback test the data is returned to the sender after specific stages of the transmitter or receiver (cf. Fig. 2-26 *Loopback Testing Overview* in [145]). In a direct connection with a computer, those tests could not have been performed: parts of the communication are handled by the network controller, and only limited information is available to the operating system.

## 4.4. Examining the Transmitted Data

After the connection was established successfully, the SADC was disconnected from the evaluation board and instead connected to the Ethernet port of a regular computer. The SADC was connected to a signal generator producing a rising pulse, similar to the blue trace shown in Figure 4.2, on channel 14. The SADC was configured to self-trigger on this channel with the help of the implemented leading edge discriminator. Upon triggering the SADC sends a so-called *Feature packet*, containing event information and extracted features from all channels, and a *Sample packet*, containing a *snapshot* of the waveform with a length of 512 samples. Both encapsulated in an Ethernet Type 2 Frame as shown in Figure 4.4. Note that

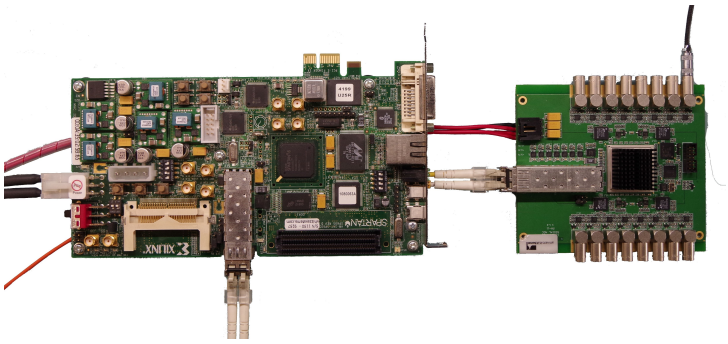
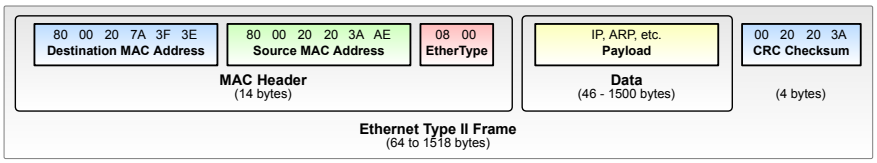


Figure 4.3.: XILINX SP605 evaluation board connected to the 16-Channel PANDA-SADC with a fiber for loopback tests of the network interface. A test signal is connected with a LEMO cable (top right) to investigate the signal processing.

the structure of the networks packets, as it will be presented in this section, is different from the one for the final firmware (shown in section 7.5.6), as the requirements and accessible information have evolved.




Source: `https://commons.wikimedia.org/wiki/File:Ethernet_Type_II_Frame_format.svg`

Figure 4.4.: Content of a Ethernet Type 2 Frame, which has been used as convention for the first tests. *Payload* refers to the actual *user data* (feature packet, sample packet). The CRC checksum is not visible to the user, as the error correction is handled by the Network Interface Card.

**Feature Packet** The software WIRESHARK was used to capture incoming packets and analyze the raw packet content. An example for a feature packet is shown in Fig. 4.5. The content can be *decoded* with the help of the following information:

- **Protocol Header (14 byte):** Consists of the 6-byte destination and source MAC address, and a 2-byte identifier for the packet type (see MAC Header in Fig. 4.4). `0x6559`, translating to *raw frame relay*, was chosen arbitrarily.
- **Packet Header (4 byte):** `0xCBCBCBCB` is used as delimiter for the packet.
- **Event Counter (4 byte):** A counter that is increased with every trigger event.
- **Event Timestamp (8 byte):** A timestamp that is stored upon triggering, relative to the startup time of the FPGA. It consists of 4 byte counting seconds and 4 byte counting nanoseconds.
- **CFD Timestamp (16×2 byte):** The zero-crossing time of a constant fraction discriminator is determined for every channel, relative to the event timestamp.

- **Integral (16×2 byte):** An integral of specified length for every channel.
- **Packet Trailer (4 byte):** `0xCBCBCBCB` is used as delimiter for the packet.

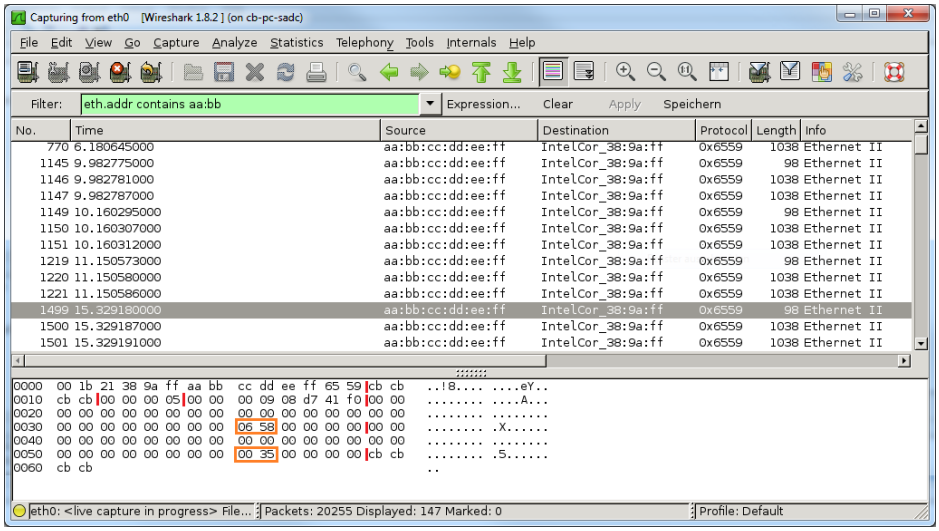



Figure 4.5.: A typical feature packet. The top part of the window lists all received packets, the bottom part shows the packet content in hexadecimal decoding and ASCII decoding (the latter does not make sense to investigate, since the data is not ASCII encoded, but binary). The packet was send to the computer with MAC address `0x00:1b:21:38:9a:ff` and send from the SADC with MAC address `0xaa:bb:cc:dd:ee:ff`. The event counter shows that it is the fifth event. The event timestamp `0x0000000908d741f0` corresponds to 9.148 324 848 s. The following channel-wise *features* are all zero except for channel 14, where a CFD timestamp (`0x0658`≙1.624 µs) and an integral value (`0x0035`≙35) have been determined.

**Sample Packet** Likewise, an example for a *Sample packet* is shown in Fig. 4.6. The protocol header, packet header and packet trailer are the same as for the *Feature packet*. Between packet header and trailer, a snapshot of the waveform with 512 2-byte sample points is transmitted. The position of the snapshot relative to the internal self-trigger is chosen such that the pulse is fully contained in it.

## 4.5. Summary

Working with the 16-channel PANDA-SADC prototype has been a valuable experience for the start of the CB-SADC project. The hardware was ready-to-use and the existing firmware allowed a quick start. First feature extraction algorithms for the determination of timestamps and integrals were successfully implemented in the firmware and could be verified using integrated logic analyzers. A network interface has been implemented, covering the physical layer and the data link layer to transmit Ethernet Type 2 frames. Still missing was the

*network* and *transport* layer, necessary for example for UDP/IP transmissions. The transfer of network packets containing extracted features and raw waveform snippets was investigated with a network packet analyzer, since no software has been developed at the time to interpret and visualize the data. The firmware was used as a starting point for the later development for the 64-channel CB-SADC described in chapter 7.

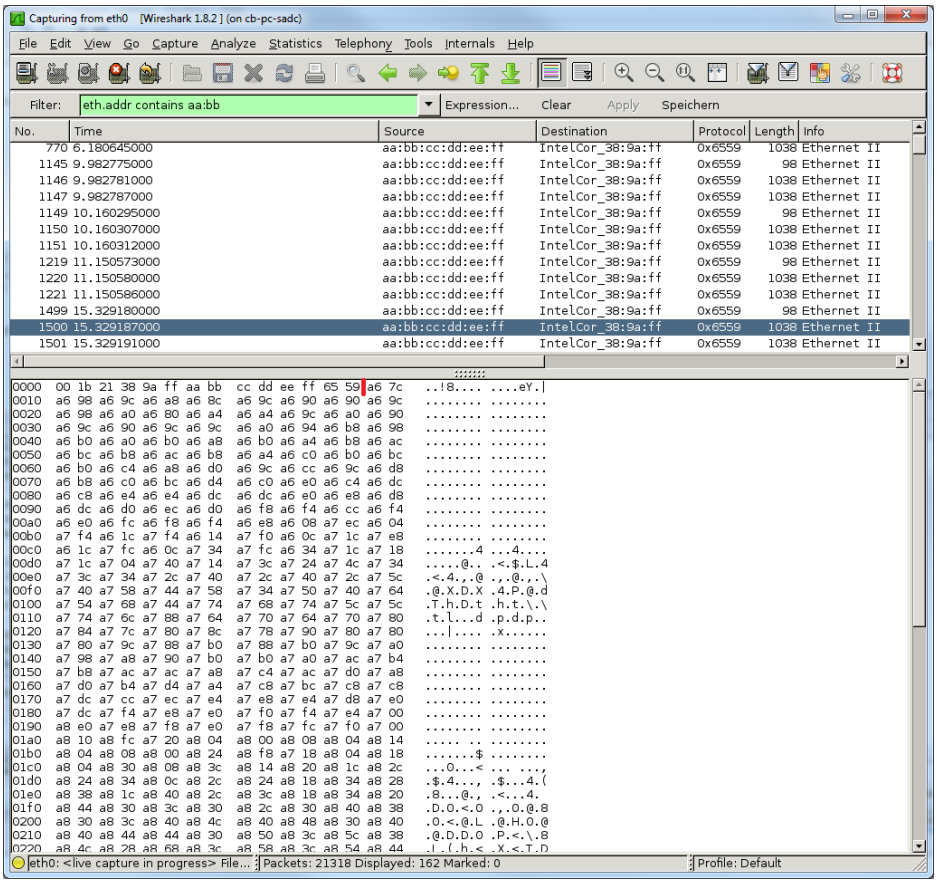



Figure 4.6.: A typical sample packet. The top part of the window lists all received packets, the bottom part shows the packet content. The protocol header is the same as in Fig. 4.5. The *payload* contains a waveform snippet with 512 samples of 2 byte. One can even "see" the rising pulse shape in the raw data by observing the 2 byte data samples, starting with `0xA67C`, eventually progressing to `0xA7**` in the second third of the screenshot, to `0xA8**` in the last third. The falling part of the pulse is not visible in the screenshot.

# 5. CB-SADC Hardware Development

This chapter will cover the hardware development of the CB-SADC itself, while the auxiliary hardware (*power supply, analog input, backplane, ...*) will be discussed in chapter 6. As stated in section 3.3.2, the design of the CB-SADC was derived from one of the PANDA-SADC prototypes, which is visualized in Figure 5.1. The first 64-channel PANDA-SADC, `ADC-32DR v1.0`, featured Xilinx Virtex-6 FPGAs. Optimizations in cost, power consumption, and connectivity have led to the Xilinx Kintex-7 based `ADC-64K v1.0` [137, 146]. This version was the branch point at which the development of the CB-SADC started: the design sources were used to develop `ADC-64K-CB v1.0` in 2014, followed by the revised version `ADC-64K-CB v1.1` in 2016. In parallel, the PANDA-SADC has also been improved up to version `ADC-64K v3.1` in 2016. On various occasions there have been synergies between the two developments. Whenever the review of the schematics or the testing in the laboratory have revealed a bug on either side, it has been evaluated in a mutual effort, presumably leading to benefits for both parties.

Section 5.1 will present the design concept of the CB-SADC and guide through the remaining sections of this chapter.

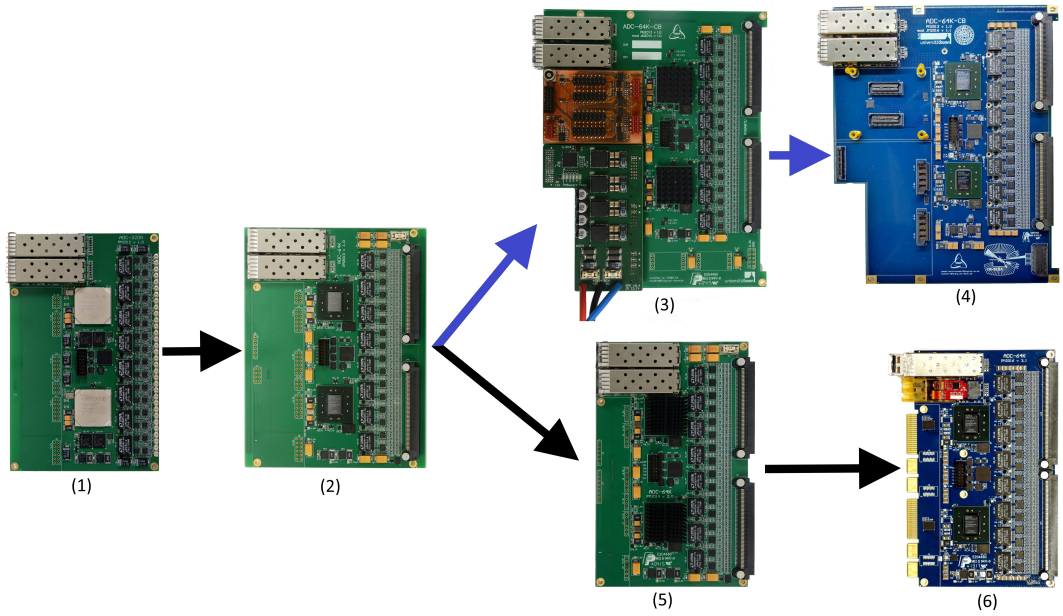



Figure 5.1.: Development of the 64-Channel SADC Prototypes, the black arrows indicate the PANDA branch, the blue arrows the CB-SADC branch: (1) `ADC-32DR v1.0` (2) `ADC-64K v1.0` (3) `ADC-64K-CB v1.0` (4) `ADC-64K-CB v1.1` (5) `ADC-64K v2.0` (6) `ADC-64K v3.1`. The CB-SADC production revision, `ADC-64K-CB v1.2`, is not shown.

## 5.1. Concept

Since the CB-SADC is based on the PANDA-SADC, its core concept should be argued based on the development within the PANDA framework. The design decisions for the PANDA-SADC project were driven by three core requirements:

- **High channel density:** The installation at the PANDA experiment takes place in a constrained space, requiring high-density digital and analog components on the board. At the writing of this thesis, the chosen ADC chip is still the only one offering eight channels each at a this sampling rate and footprint. Further decrease in size would only be possible with ASIC (Application-Specific Integrated Circuit) development.
- **Low power:** Due to the constrained space at the PANDA experiment, the cooling is equally challenging. The ADC chips, the input buffer op-amps, and the FPGA of an SADC can have very high power demands and are therefore usually responsible for most of the produced heat. Thus, they were selected from the lowest power options that were available on the market.
- **Radiation tolerance:** Since the PANDA-SADCs are placed close to the detector, they are subject to considerable radiation. A dedicated circuit on the board allows the PANDA-SADC to self-recover in a number of radiation damage scenarios.

More details about the design requirements can be found in the PANDA TDR [135].

The requirements have led to the concept presented in Figure 5.2. The components that are named in the concept can be identified in the photograph of the CB-SADC in Figure 5.3. The rightmost block represents the analog input stage, where the signals from the Crystal Barrel Calorimeter are connected to the board. The purpose of this circuit is to buffer and shape those signals, which are then connected to the digitization stage. Here, the analog signals are converted to digital signals by eight eight-channel sampling ADC chips with 80 MS/s and 14 bit resolution. The two KINTEX-7 FPGAs receive the digitized and serialized[1] data from the ADCs. They are the core of the board and can be configured through their firmware to process the ADC data and extract desired information (*feature extraction*) or waveform snippets, which are then sent through a network interface to a further processing stage like a computer. Between the FPGAs, a PLL circuit produces clocks of different frequencies, which are necessary for the operation of the ADC chips and the FPGAs themselves. An intricate arbitration and programming circuit, consisting of the elements *MUX, ARBITER, CFG PROM*, allows the FPGAs to reconfigure each other and to reprogram the PLL, in the case of radiation damage.

**Outline** The path from the analog data up to the de-serializing stage within the FPGAs will be discussed in section 5.2. In section 5.3 the clock generation and distribution will be investigated. A brief introduction into the used FPGA will be provided in section 5.4. The programming of the FPGAs and a special arbitration circuit will be presented in section 5.5. The power distribution network will be analyzed in section 5.6. Last, section 5.7 summarizes the changes which make the CB-SADC distinct from the PANDA-SADC, and section 5.8 provides an overview of the prototype revisions of the CB-SADC.

[1] Instead of a parallel connection of 14 bit at 80 Mbit/s, data is transmitted with 2 bit at 640 Mbit/s (including padding) for each ADC channel (cf. section 7.3.1).

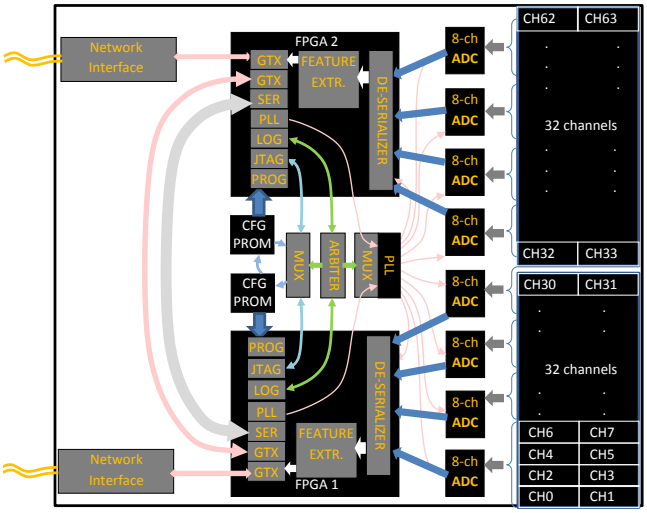



Figure 5.2.: Data flow of the SADC. For good comparison, the elements are arranged similarly to the actual board from Figure 5.3. [147] (mod.)

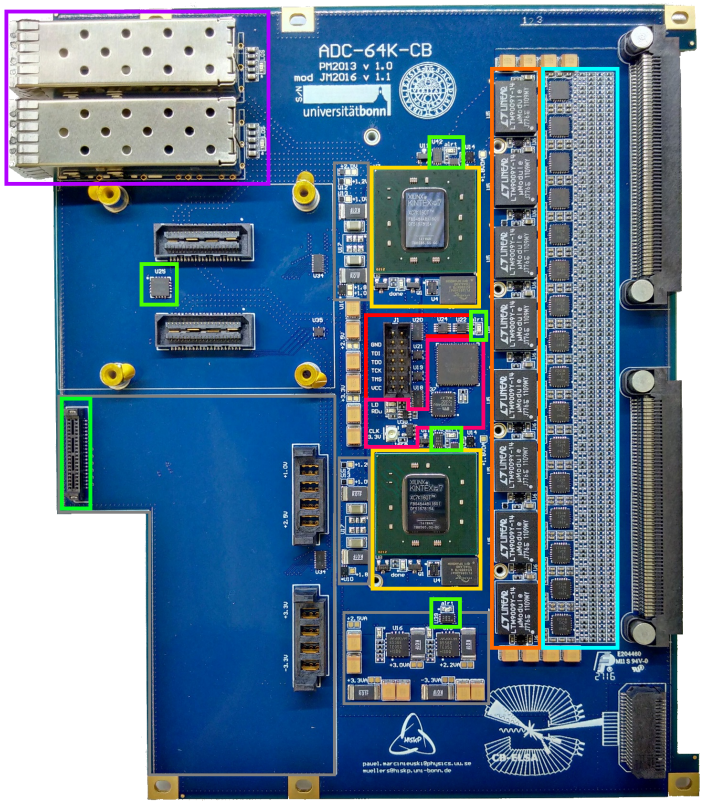



Figure 5.3.: Picture of the second CB-SADC prototype, with colored borders to group elements: input buffers and optional shaping, octal ADC ICs, FPGAs with memory, clock distribution, programming and arbitration, slowcontrol, SFP interface, power distribution. Below the SFPs and at the bottom right an extension board can be mounted to access slowcontrol, trigger, and programming signals.

## 5.2. Digitization

The digitization stage, consisting of the input buffers and the pipeline-ADCs, is located on the right-hand side of Figure 5.3 and 5.2. This stage is a crucial step, as the data is crossing from the analog to the digital domain. Careful consideration of the Printed Circuit Board (PCB) layout is necessary to prevent any crosstalk of fast digital signals to the analog signals. In section 5.6 it will be shown how the power distribution network also obeys a strict separation of the domains for the same reason.

At this stage, the analog signal is already optimized in regard to pulse length and SNR by the Analog Input Card (cf. section 6.1.1). This allows to modify the signal filtering, or shaping, by exchanging the Analog Input Card, and keeping the CB-SADC's configuration universal.The following sections will show the limited on-board analog signal processing and the digitization stage.

### 5.2.1. Input Buffers and Anti-Aliasing

The layout of the input buffer circuit, consisting of op-amps, resistors, and capacitors, was not modified with respect to the PANDA-SADC, where it is being used as a complex shaping stage with a Sallen-Key filter topology [148]. For the CB-SADC, the circuit is mainly used as a unity-gain buffer to decouple the signal driving circuit from the pipeline-ADC's inputs. For this the values of the resistors and capacitors have been modified.

The op-amp ADA4940-2 from ANALOG DEVICES was chosen due to the high channel density (two fully differential channels in a 3 mm × 3 mm case) and the low standby current of 1.25 mA per channel. It is supplied with a 3.0 V power supply and capable of almost rail-to-rail output swing (0.1 V to 2.9 V), with a −3 dB bandwidth of 25 MHz at 2 $V_{pp}$ up to 260 MHz at 0.1 $V_{pp}$ [8]. Between op-amp and the pipeline-ADCs, a first order passive low-pass filter is placed. The complete input buffer schematic is shown in Figure 5.4.

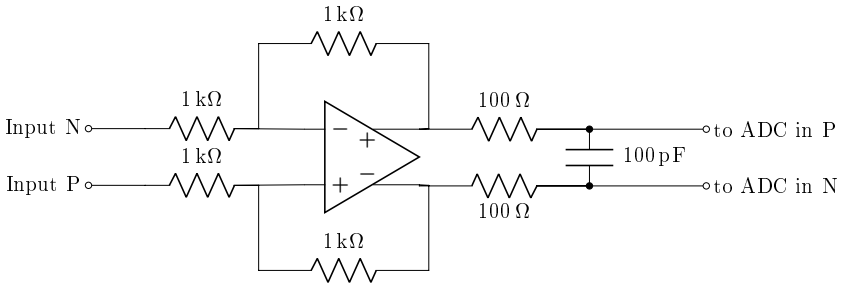



Figure 5.4.: Schematic of the input buffer on the CB-SADC, with 1 kΩ input resistors and 1 kΩ feedback resistors (gain = 1), and anti-aliasing filter at the output.

The low-pass filter improves the SNR and also acts as a weak anti-aliasing filter to suppress frequencies above the Nyquist limit $\frac{f_s}{2} = 40$ MHz (see section 1.4.6). With a steepness of $-6\,\mathrm{dB\,oct^{-1}}$ ($-20\,\mathrm{dB\,dec^{-1}}$), it is far from the ideal *brick wall* filter, which is a filter with infinite steepness. Keeping this in mind, the choice of the filter frequency is now a compromise between bandwidth and SNR. Although the bandwidth of the Crystal Barrel Calorimeter signals does not exceed 1.5 MHz (cf. Figure 2.13a), the additional measurement of the fast energy sum with spare SADC channels already presents a higher-bandwidth application. A

simulation by C. Honisch has yielded that the shape of the fast energy sum signal is not affected too much if the low-pass is at a frequency not much below 8 MHz. Thus, the −3 dB filter frequency is set to

$$f = \frac{1}{2\pi \cdot 2 \cdot 100\,\Omega \cdot 100\,\mathrm{pF}} \approx 7.96\,\mathrm{MHz}$$

In combination with the separate shaping stage (cf. section 6.1.1), the overall attenuation is −67.8 dB at $\frac{f_s}{2} = 40\,\mathrm{MHz}$, amounting to a sufficient suppression of signal contributions above the Nyquist limit. In the next paragraph the noise contribution of the op-amp and the surrounding resistors will be calculated.

**Noise** The op-amp's noise contribution [8] after this first order low-pass is

$$\begin{aligned} U_{\mathrm{ADA}} &= 3.9\,\mathrm{nV_{RMS}}\,\sqrt{\mathrm{Hz}}^{-1} \cdot \left( \int_{0\,\mathrm{MHz}}^{260\,\mathrm{MHz}} \frac{1}{1 + \left(\frac{f}{7.96\,\mathrm{MHz}}\right)^2} df \right)^{0.5} \\ &= 3.9\,\mathrm{nV_{RMS}}\,\sqrt{\mathrm{Hz}}^{-1} \cdot \sqrt{12.26\,\mathrm{MHz}} \\ &\approx 13.7\,\mu\mathrm{V}. \end{aligned}$$

Additional noise contributions stem from the resistors and the capacitor. For one 1 kΩ input resistor, the noise contribution for the same bandwidth at a temperature of 30 °C is, according to equation 1.5:

$$\begin{aligned} \sqrt{v_n^2} \cdot \sqrt{\Delta f} &= \sqrt{4k_{\mathrm{B}} \cdot 30\,^{\circ}\mathrm{C} \cdot 1\,\mathrm{k\Omega}} \cdot \sqrt{12.26\,\mathrm{MHz}} \\ &\approx 14.3\,\mu\mathrm{V}, \end{aligned}$$

which is comparable to the op-amp's contribution. Thus, the noise from the resistors is dominating in this part of the circuit. The noise contributions of the individual components can now all be calculated and added (quadratically), but this becomes tedious, especially when taking into account the additional shaping stage. A simulation of the noise density and the total noise has been done with the SPICE frameworks TINA and LTSPICE. The simulation models for resistors and capacitors are integrated in the framework, whereas the model for the ADA4940-2 has been obtained from ANALOG DEVICES[2]. The simulations will be shown in section 6.1.1, together with the shaping stage on the Analog Input Card.

The noise contribution of an op-amp circuit can be lowered by two factors. First, the thermal resistor noise could be reduced by choosing lower value resistors, reducing by a factor of two will lower the noise by a factor of $\sqrt{2}$. This will lead to a higher power consumption of the analog front end (cf. section 6.1) and the feedback circuit, since the power consumption scales inversely with the resistances. It will be shown later that the total noise seen by the CB-SADC is sufficiently low, and further optimization is not worth the additional power consumption. Secondly, the filter frequency could be lowered further, but this would deteriorate the SADC's capability to measure higher frequency components which are needed for the measurement of the fast energy sum.

[2] The imported `.cir` netlist provided by ANALOG DEVICES (revision B 2/2012) states that the voltage/current noise are not fully specified.

### 5.2.2. Pipeline-ADC

Following the demands of section 3.3.2, the only ADC series fulfilling the given requirements was the LTM9009-14 by LINEAR TECHNOLOGIES. The density of eight channels in a 11.25 mm × 9 mm case allowed the packed design shown in Figure 5.3 (orange region). Although the LTM9009-14 is available with a sampling rate of up to 125 MS/s, the smallest 80 MS/s version was chosen, as the sampling rate is sufficient, and naturally price and power requirements scale with the sampling rate. The LTM9009-14 is a pipelined architecture as described in section 1.3.2. As shown in Figure 5.5, it consists of eight 14-bit ADC cores (cf. Figure 1.8 and 1.9 on page 9) with the sampling clock being supplied by the *encode input*. The SHA is shown in more detail in Figure F.3 on page 261, where the internal capacitances and resistances, which are the load for the input buffers, are quantified.

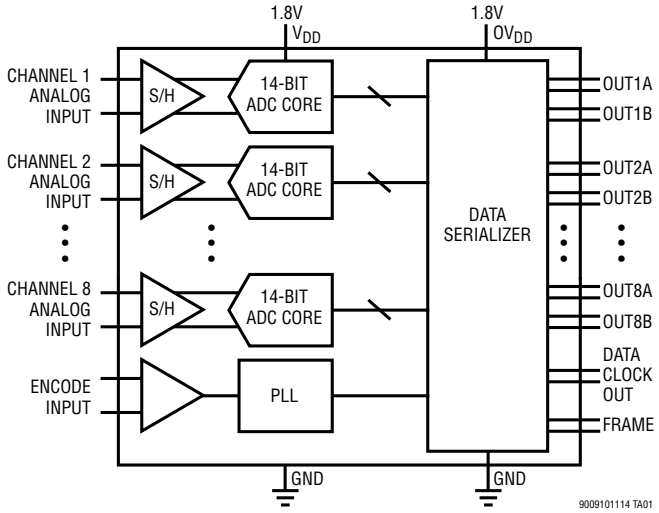



Figure 5.5.: Functional block diagram of the LTM9009-14, showing the ANALOG INPUTs (connected to the input buffers) and the ENCODE INPUT (connected to the digitizing clock) on the left, and the digital OUT#A/B (connected to the FPGA) on the right. The DATA SERIALIZER reduces the width of the digital signals (at the expense of higher transmission speed). The FRAME output indicates sample boundaries for de-serialization. [149]

The LTM9009-14 uses fully differential signaling for the analog inputs and the digital outputs, which yields better data integrity and allows higher bandwidths. Analog and digital parts of the Integrated Circuit (IC) have their own power supply (denoted as $V_{\mathrm{DD}}$ and $OV_{\mathrm{DD}}$), which are also physically well separated on the footprint. This allows to separate the copper traces on the PCB, and crosstalk from the fast switching digital domain to the sensitive analog domain can be reduced.

In the next paragraphs, the resolution and the linearity will be evaluated on a theoretical level (based on the statements in the datasheets), while the respective measurements will be presented in section 9.2. The LTM9009-14 can be operated with a full scale range of $1\,\mathrm{V_{pp}}$ or $2\,\mathrm{V_{pp}}$ (the latter being the default in the scope of this thesis), which can be selected by an external resistor. The digital interface (`OUT1A`, `OUT1B`, ...), which is connected to the FPGA, will be described in section 7.3.1.

**Resolution** The vertical resolution of the LTM9009 is 14 bit, leading to a maximum theoretical SNR of 86.4 dB according to equation 1.13. With a maximum input voltage range of $A_{IN^+} - A_{IN^-} = 2\,V_{pp}$ the step size is

$$1\,\mathrm{LSB} = \frac{2\,V_{pp}}{2^{14}} \tag{5.1}$$

$$\approx 122\,\mu V. \tag{5.2}$$

However, the ADC is not free of intrinsic noise (cf. section 1.4). The input referred noise can be observed by measuring the digital output with a short-circuited analog input, shown in Figure 5.6a. For an ideal ADC, a single output value would be present (in case of this differential 14-bit ADC, the center value is $2^{13} - 1 = 8191$). The measurement shows a seemingly Gaussian distribution with a center value of 8188. The deviation of the center value by 4 LSB can be explained by the specified offset error of $\pm 3\,\mathrm{mV} \widehat{=} 25\,\mathrm{LSB}$. The transition noise is specified in the datasheet to be $1.2\,\mathrm{LSB}_{RMS}$ (corresponding to a FWHM of roughly 2.8 LSB). Due to the measurement method this includes any influence of quantization noise and can be translated to a Direct Current (DC) noise level of about

$$\sigma_{\mathrm{trnoise}} = 1.2\,\mathrm{LSB}_{RMS} \tag{5.3}$$

$$= 146\,\mu V_{RMS} \tag{5.4}$$

at $2\,V_{pp}$ full scale. The corresponding SNR can now be calculated to be

$$\mathrm{SNR} = 20\log_{10}\left(\frac{A_{\mathrm{signal,RMS}}}{\sigma_{\mathrm{trnoise}}}\right) \tag{5.5}$$

$$\approx 20\log_{10}\left(\frac{2\,V/\sqrt{2}}{146\,\mu V_{RMS}}\right) \tag{5.6}$$

$$\approx 79.7\,\mathrm{dB}. \tag{5.7}$$

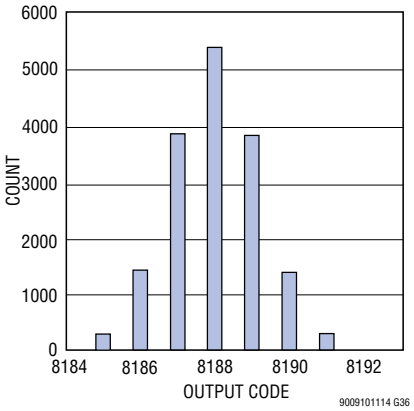
The datasheet states a lower DC SNR of $\approx 73.4$ dB at $-1$ dBFS signal level (see Fig. 5.6b), which makes a discrepancy of almost 6 dB [149]. Unfortunately the discrepancy cannot be explained, as the datasheet does not disclose more details about the measurements. The curve in Fig. 5.6b shows nicely how the SNR degrades with higher input frequency, which can be explained by the influence of the aperture time jitter of $0.15\,\mathrm{ps}_{RMS}$. For a 1 MHz input signal at full scale, according to eq. (1.25) (page 19), this accounts for a noise level of

$$\Delta v_{\mathrm{RMS}} = \frac{2\pi f \cdot A \cdot \Delta t_{\mathrm{A,RMS}}}{\sqrt{2}} \tag{5.8}$$

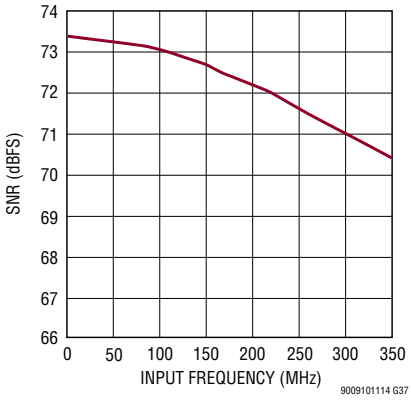
$$= \frac{2\pi \cdot 1\,\mathrm{MHz} \cdot 2\,V \cdot 0.15\,\mathrm{ps}_{RMS}}{\sqrt{2}} \tag{5.9}$$

$$\approx 1.3\,\mu V, \tag{5.10}$$

which is negligible compared to the transition noise $\sigma_{\mathrm{trnoise}}$, since in the scope of this thesis, the broadband CsI(Tl) signal will not have frequency components much above that. But, as stated in section 1.4.5, the jitter of the sampling clock has to be considered in the same way, and is added quadratically to the aperture time jitter. It will be measured, calculated and discussed in section 5.3 whether this will affect the performance.



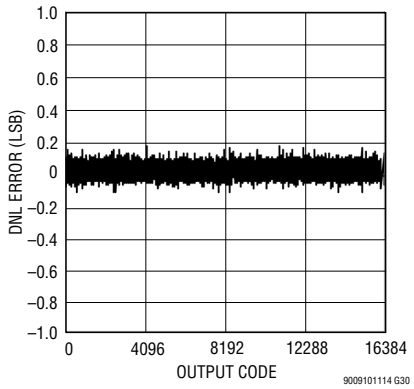

(a) ADC output with shorted input, transition noise $\sigma_{\mathrm{trnoise}} = 1.2\,\mathrm{LSB}$.



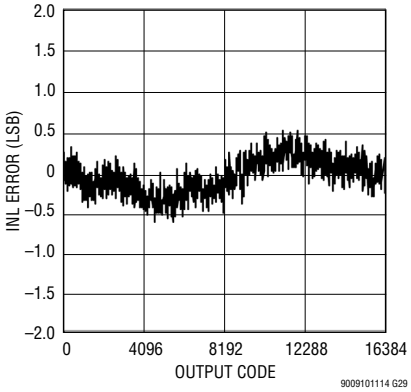

(b) SNR versus input frequency, showing aperture time jitter effects.

Figure 5.6.: Noise performance of the LTM9009-14 according to the datasheet.[149]

**Linearity** It is important to know and understand the linearity of the ADC, which is expressed in Differential Non-Linearity (DNL) and Integral Non-Linearity (INL). The DNL describes the deviation of the step size (see equation 5.1), which is stated to be typically $\pm 0.3\,\mathrm{LSB}$. This is negligible, even more so in applications where the data is averaged during signal processing, as is the case here. In contrast, the INL, which is defined as the deviation of the transfer function from the ideal transfer function, can be crucial. Figure 5.7 shows both types of non-linearity, and a clear structure is visible in Figure 5.7b. The INL is specified in the datasheet to be below 1 LSB (typically), which, in the worst case, would amount to an energy deviation of $\approx 150\,\mathrm{keV}$ over the whole range. This value does not seem critical, and could (in principle) even be calibrated and compensated.




(a) Differential Non-Linearity.




(b) Integral Non-Linearity.

Figure 5.7.: Linearity characteristics of the LTM9009-14 according to the datasheet, showing the deviation from the ideal step size (DNL) and from the ideal transfer function (INL). [149]

## 5.3. Clocking

Both the ADCs and the FPGAs fabric require external clock signals for operation. In trivial applications, this is usually achieved with a quartz crystal. In the case of the CB-SADC however, the demands for the clocking architecture are more complex:

1. Eight clocks for the eight LTM9009-14 ADCs with $< 1$ ps jitter (estimate based on equation 5.8) and 80 MHz are required for the digitization, as is explained in section 1.3.2. Those clocks will be called `CLK_ADC`.

2. Two clocks for the DSP/feature extraction fabric of the FPGAs which will handle the processing of the sampled data of the ADCs. Like for the ADCs the clock has 80 MHz, and is called `CLK_DSP`.

3. Two differential clocks with 100 MHz are needed for the network fabric of the FPGAs (called `CLK_MGT` after XILINX's name for the fabric).

4. A fixed phase relation between the generated clocks is required for defined timing determination.

5. A fixed phase relation to a source clock is required for a defined timing relation between all SADCs, if they will all be locked to the same source clock.

6. The reference source clocks should be selectable from multiple sources.

The requirements are fulfilled by a custom clock management circuit as shown in Fig. 5.8. The circuit consists of two components, which can also be identified on the PCB in Fig. 5.10a: A four-channel clock multiplexer (MICREL SY89546) for the selection of the source clock, and a low-noise clock jitter cleaner with an internal Voltage-Controlled Oscillator (VCO) and twelve configurable clock outputs (TEXAS INSTRUMENTS LMK04806).

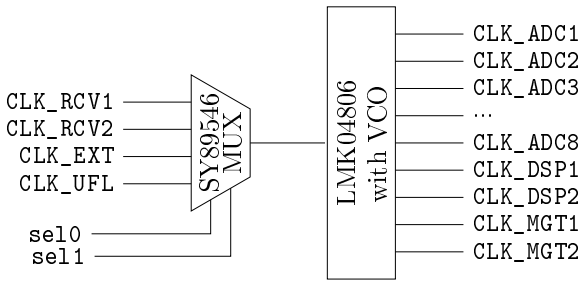



Figure 5.8.: Clocking scheme of the CB-SADC, with four selectable reference clock sources, and twelve configurable output clocks.

On the left-hand side of Figure 5.8, four different clock sources are shown, which can be selected through a DIP switch on the back side of the FPGA (see Figure 5.10b). The first two clock sources (`RCV`) are received from the FPGAs themselves, as they are able to source a clock either through the connected 100 MHz MEMS oscillator that is required for boot-up, or a recovered clock from the SFP's Receiver path; the latter allows a phase synchronization to the Ethernet switch. The third clock source `CLK_EXT` is routed to the auxiliary connector

at the back of the CB-SADC, allowing a synchronization through a connected backplane (see section 6.4). The last clock source is a `U.FL` connector on the board (see Figure 5.10a), which allows to test the clock circuit easily in a laboratory setup.

On the right-hand side of Figure 5.8, the LMK04806 with the demanded $8+2+2$ output clocks is shown. This IC consists of two separated internal Phase-Locked Loop (PLL) stages called PLL1 and PLL2. The PLL1 stage is a *clock jitter cleaner*, which is neither being used in the original PANDA-SADC nor the CB-SADC, since further jitter cleaning is not worthwhile, given the expected quality of the source clocks and the requirements of the circuits clocked by the LMK04806. The PLL2 stage is shown in Figure 5.9 configured in 0-Delay Single Loop mode, and is responsible for the generation of twelve clocks with up to six different frequencies from the internal VCO. The input `OSCIn` is connected to the output of the multiplexer SY89546. Through a feedback loop and a phase detector, the internal VCO is configured to run at 2400 MHz. Each of the six blocks following the VCO can be configured with a specific integer divider to obtain the output frequencies. In case of the CB-SADC, the required frequencies are obtained with dividers of 30 (for `CLK_ADC` and `CLK_DSP`) and 24 (for `CLK_MGT`). Since the feedback (loopback) can be placed after those blocks at the clock output, it is possible to match the phase of the outgoing clock to the source clock; this mode is called *0-Delay*. A more detailed overview of the configuration and the internal circuit can be found in Figure F.5 and Figure F.4 (page 262 f.).

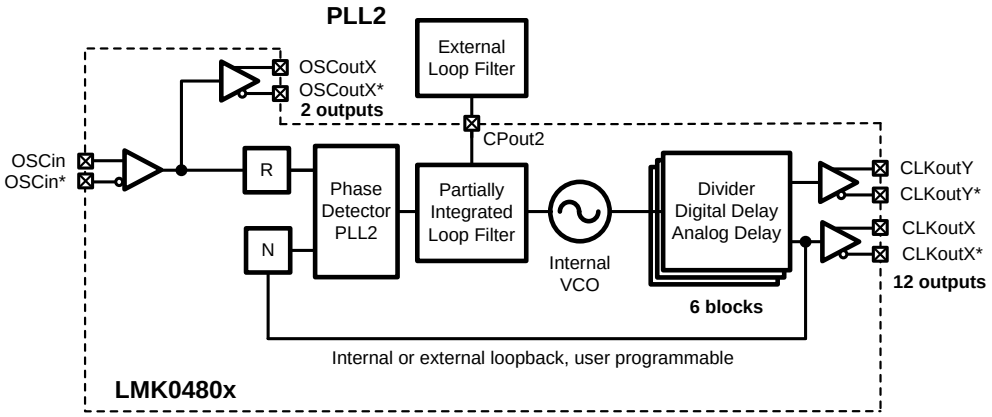


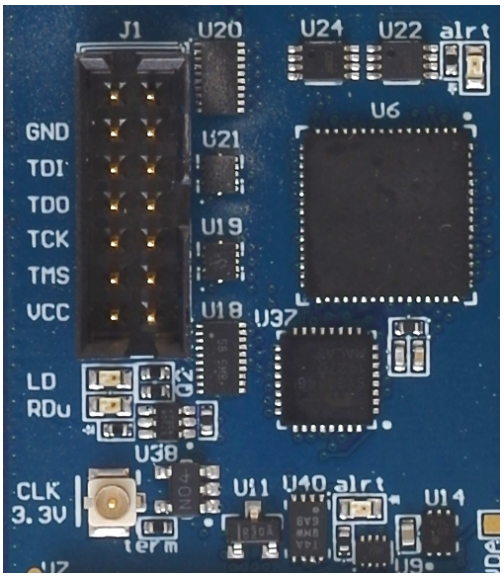

Figure 5.9.: Functional block diagram of the LMK04806 in 0-Delay Single Loop mode, where the first PLL is deactivated. The output of the muxer SY89546 is connected to the OSCin differential input. R and N are dividers to match the desired frequency of the Phase Detector (100 MHz) and the VCO (2400 MHz) to the reference clock and output clock. The digital or analog delay functionality is not used. [150]

.

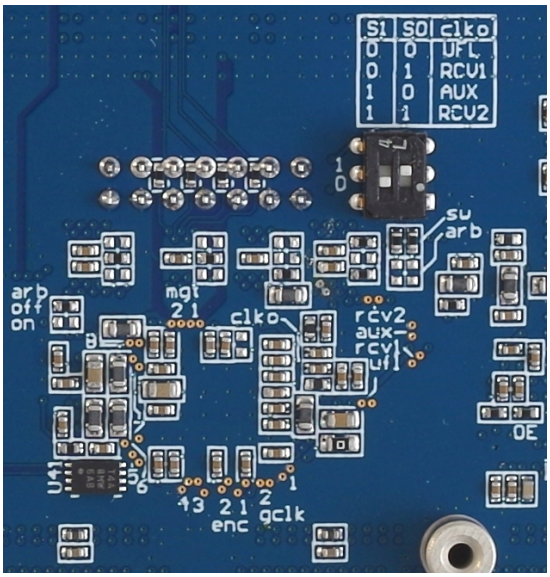
**Configuration** The LMK04806 is configured through an independent digital bus called `µwire`, connected to both FPGAs. Through an arbitration circuit, either FPGA can take control over this bus (cf. section 5.5.2). The configuration is extensive and allows fine tuning in several ways, e.g. optimization of the internal phase detector, and setting of the integrated loop filter. An overview of the configuration of the PLL2 can be found in Figure F.5 (page 262). The configuration is implemented in the firmware (see section 7.2.2).

### 5.3.1. Jitter

It is not advisable to use FPGAs as direct clock sources for ADCs due to an output jitter of $\approx 100\,\mathrm{ps_{RMS}}$. Nevertheless, with a PLL like the LMK04806, the FPGA may be used as a reference for the ADC clocks. It will now be shown that a sufficiently low jitter is obtained. For the CB-SADC adaption of the PANDA-SADC design, the connection of the voltage supplies for the LMK04806 has been optimized as recommended in [150]. Several ferrite beads (inductances) and capacitors were added to decouple the supplies of PLL1, PLL2, and the different clock domains. Furthermore, measurement points (see Figure 5.10b) were added to the CB-SADC around the LMK04806 by removing the tenting from the vias. Those allow to conveniently measure the clock outputs with the help of an active differential probe.




(a) PLL LMK04806 (U6) and muxer SY89546 (U37), U.FL clock input (bottom left), and clock status LEDs (`LD`, `RdU`) to indicate programming and phase lock. Also visible is the arbitration circuit with U18 to U24, and the JTAG programming port J1.




(b) Probe points for the measurement of the clock signals, distributed along the back side of the TEXAS INSTRUMENTS LMK04806. Also shown is the DIP switch for the selection of the SY89546's source clock (source `CLV_RCV1` is selected).

Figure 5.10.: The CB-SADC's clock distribution circuit on the top- and bottom-layer (magenta region in Fig. 5.3). The circuit and layout have been extensively modified with respect to the original PANDA-SADC design.

To analyze the jitter, the clock signals were measured at various points using a KEYSIGHT MSO9000 oscilloscope with the KEYSIGHT N2750A active differential probe. Exemplary measurements of all twelve clock outputs are shown in Fig. D.1 (page 239). Unfortunately the clock signals all exhibit reflections, which hints at a problem with impedance matching. Figure 5.11 shows a measurement of the clock `CLK_ADC2` (digitization clock of the second LTM9009-14 ADC), on the left close to the LMK04806, and on the right close to the LTM9009-14. According to the analysis routines of the oscilloscope, the jitter at the LTM9009-14 is around $3.5\,\mathrm{ps_{RMS}}$ and has increased by 30 % along the traces, although the

PCB manufacturing process was optimized to yield 100 Ω characteristic impedance, matching the termination resistance. The measured jitter has never exceeded 5 $ps_{RMS}$, which according to eq. 1.25 leads to a voltage noise of 44 $\mu V_{RMS}$ for a full scale sine signal with 1 MHz. This worst-case estimate is well below the 146 $\mu V_{RMS}$ input referred noise.

(a) Measurement of the clock output `CLK_ADC2` close to the LMK04806, $\sigma_{t,period} = 2.7\,ps_{RMS}$.

(b) Measurement of the clock output `CLK_ADC2` close to the LTM9009-14, $\sigma_{t,period} = 3.5\,ps_{RMS}$.

Figure 5.11.: Comparison of the clock signal and jitter at the source (LMK04806) and destination (LTM9009-14), measured with the KEYSIGHT N2750A active differential probe. The oscilloscope screenshot shows a full clock period (12.5 ns) and exhibits reflections. The period was calculated by the oscilloscope's measurement routines over several thousand samples, which is displayed as a histogram. The width of this histogram, $\sigma_{t,period}$, is the jitter. The jitter degrades from 2.7 $ps_{RMS}$ at the source to 3.5 $ps_{RMS}$ at the ADC, presumably due to non-ideal impedance matching. During the measurement a brick wall filter at 1 GHz was activated.

It is interesting to also investigate the clock outputs from the FPGAs, since they are used as source clocks for the LMK04806. Figure 5.12 shows the clocks `CLK_RCV1` and `CLK_RCV2`, which originate from 100 MHz MEMS oscillators that are routed through the first and second FPGA, respectively. On the first look, the clock waveforms seem much cleaner and with less reflections effects compared to the clocks generated by the LMK04806, indicating a better impedance matching of the signals. Taking a closer look, the histograms of the measured clock period show broad, non-Gaussian distributions, which seem to be superpositions of at least two or three distinct modes (frequencies) that the MEMS oscillator allows. The standard deviation is calculated by the oscilloscope to be almost 40 $ps_{RMS}$.

Hence, it is shown that the clock jitter cleaner has improved the jitter by a factor of at least ten and produces stable output clocks. It can be summarized that the clock distribution circuit works well considering the demands of the application. Improvements could be made by closer inspection of the impedance matching, e.g. the measurement of the actual trace impedance on the PCB; but this requires special equipment which was not available at the time. Furthermore, it might be worth to systematically tune and test the LMK04806's configuration and try different frequencies for the *phase detector* and also different *loop filter* parameters.

(a) Clock `CLK_RCV1` with a period of 10 ns.

(b) Clock `CLK_RCV2` with a period of 10 ns.

Figure 5.12.: Jitter measurement of the FPGAs output clocks, measured close to the input of the MICREL SY89546 clock multiplexer. The content and method is similar to Fig. 5.11. The standard deviation of the distribution is now 40 $ps_{RMS}$, a factor ten above Fig. 5.11.

## 5.4. FPGA

In this section the core component of the PANDA-SADC and CB-SADC, the FPGA, will be introduced. After an introduction into FPGAs, the specific choice made for the PANDA-SADC will be briefly reviewed to see if it is suitable for operation at the CBELSA/TAPS experiment.

### 5.4.1. What is an FPGA?

FPGAs (Field Programmable Gate Arrays) are integrated circuits which can be configured by the customer after manufacturing (*"field programmable"*). This is in contrast to CPUs (Central Processing Units), GPUs (Graphic Processing Units) and ASICs (Application Specific Integrated Circuits). The four architectures differ in flexibility, efficiency and cost (visualized in Fig. 5.13), which will be briefly evaluated.

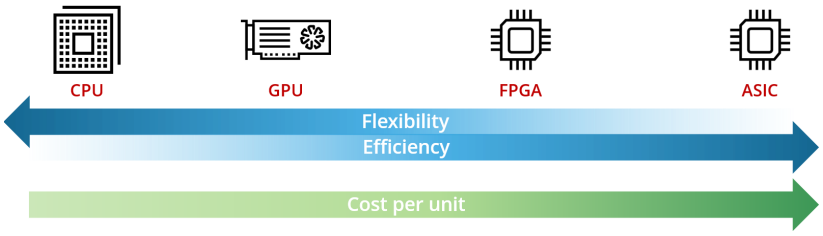



Figure 5.13.: Comparison of CPU, GPU, FPGA, ASIC.
Source: `https://hackernoon.com/a-gentle-introduction-to-hardware-accelerated-data-processing-81ac79c2105`

**CPUs** are the typical architecture used in personal computers and servers. They support a specific set of instructions, with which they can run compiled software in a procedural way. Although they can run at higher clock frequencies than FPGAs, they lack the ability for high parallelization, making them less suitable in many use cases. With multi-core CPUs and simultaneous threading, processes can run in parallel, but this is not comparable in all aspects to true parallelization.

**GPUs** are the core elements of graphic cards, whose evolution was driven mainly by the wider marked of computer gaming. They gained popularity in computing applications due to their ability to solve matrix and vector operations efficiently and with much higher parallelization when compared to CPUs.

**ASICs** are manufactured for one very specific use case. They reach unparalleled performance due to their density and customization, at the expense of very high initial development efforts and costs. They cannot only incorporate digital, but also analog circuits. But, once produced, they are limited to their designed functionality, and the addition of features or corrections of bugs usually results in a new, costly production.

**FPGAs** are ideally suited for the demands of a parallel signal processing application as on the CB-SADC. Their main logic resource are the so-called Configurable Logic Blocks (CLBs) which in general consist of lookup tables, muxers, and flip-flops. From those elements it is possible to design a multitude of combinatorial and sequential circuits. A simplified example of a CLB is shown in Figure 5.14. A part of the actual CLB of the KINTEX-7 is shown in Figure F.6 on page 263.

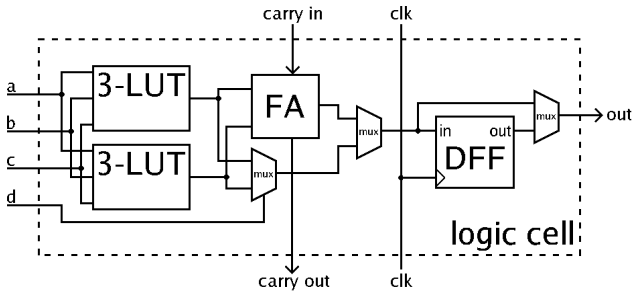



Figure 5.14.: Simplified example of a CLB, consisting of two 3 bit lookup tables, a full adder, and a D-flip-flop. The content of the lookup tables is programmed as part of the firmware and is static. All other signals are variable during operation.
Source: `https://en.wikipedia.org/wiki/Logic_block#/media/File:FPGA_cell_example.png`

After being supplied with power, the FPGA has no functionality, it is *blank*. It has to be programmed with a firmware such that its elements can interact with each other: the CLBs are connected by a reconfigurable interconnect network (shown in Fig. 5.15), its configuration basically dictates the behavior of the FPGA. This network allows to route signals to and from the physical pins of the FPGA, which are situated in the so-called Input Ouput Blocks (IOBs), and other specific elements inside the FPGA. Two such elements are show in the figure: Block RAM (BRAM) and DSPs.

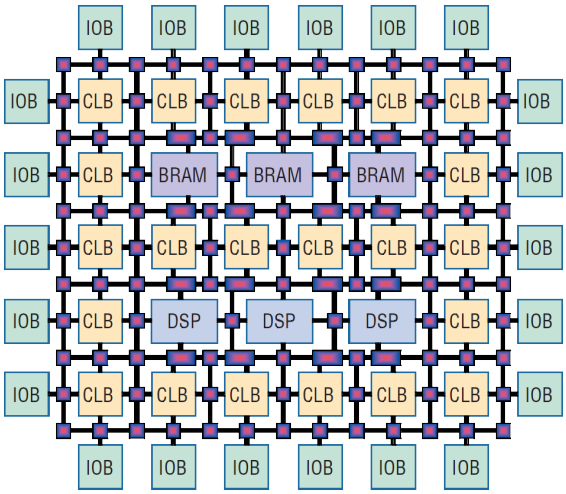



Figure 5.15.: Core element inside an FPGA. The CLBs can be connected to each other and to the IOBs with the reconfigurable interconnect (black lines and red boxes). Also shown are two special elements: DSPs and Block RAM (BRAM). [151]

The BRAM is a contiguous area of memory cells, which typically allow the implementation of a two-port memory (e.g. reading and writing at the same time, or reading from two addresses simultaneously). It stands in contrast to the implementation of memory cells with flip-flops in the CLBs, which is refereed to as Distributed RAM (DRAM).

The DSP building block allows the implementation of resource-saving data processing. In principle, all (mathematical) operations can be implemented using CLBs; but, with increasing complexity the required area in the FPGA grows, and the possible clock speed decreases due to the required propagation time through the interconnect network. The DSPs can take over complex operations due to the integrated dedicated multiplier, accumulator, pre-adder, arithmetic unit, and more (see Fig. 5.16).

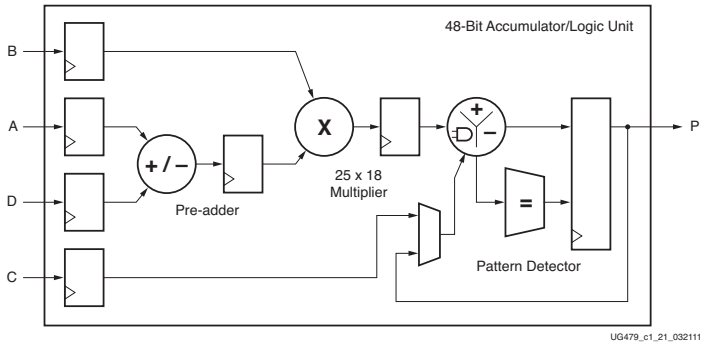



Figure 5.16.: Block diagram of a so-called `DSP48E1` slice used in the KINTEX-7. The possible functionality (operations) are visualized. [152]

### 5.4.2. Kintex-7 FPGA

The yellow regions in Figure 5.3 show the two KINTEX-7 FPGAs with the adjacent Read-only Memory (ROM) which stores the firmware. During the PANDA-SADC development, the KINTEX-7 has succeeded the VIRTEX-6 architecture due to its better pricing, yet with sufficient performance. The version placed on the board is the `XC7K160T-1FBG484C`, with a BGA footprint of $22 \times 22$ solder balls. Unfortunately this is the highest configuration level with this footprint, leaving no room for an upgrade without redesign.

In the scope of this thesis, the important parameters of the FPGA are the integrated 600 `DSP48E1` slices and the 11 700 kbyte BRAM [153]. With one FPGA being responsible for 32 ADC channels, this leaves 18 slices per channel, which can be dedicated to FIR filters (see section 7.3.5) or other algorithms. The remaining 24 DSP slices may be used for operations that are not parallelized.

The 11 700 kbyte block Random Access Memory (RAM) plays an important role for the data retention. While for the QDC readout the analog data is delayed with regard to the trigger signal by analog delay cables, for the CB-SADC this delay will be purely digital on the FPGA. With a data rate of

$$32 \cdot 14\,\text{bit} \cdot 80\,\text{MHz} = 4.48\,\text{Gbyte/s} \tag{5.11}$$

the memory is sufficient for a maximum retention of

$$\frac{11\,700\,\text{kbyte}}{4.48\,\text{Gbyte/s}} \approx 2.61\,\text{ms}. \tag{5.12}$$

This is more than enough for the usual delay between trigger and signal ($< 1\,\mu\text{s}$). However, the memory is also required for buffers in the network implementation and furthermore for debugging purposes (like the implementation of Integrated Logic Analyzers (ILAs) as shown in section 4.2).

The choice of the FPGA model, which was made by the PANDA-SADC designer [137], seems reasonable for this application, keeping in mind that for the high channel count in the PANDA experiment the cost of the board was a critical point. For the CB-SADC it would certainly have been nice to work with one of the more expensive models, as for example the largest KINTEX-7 model has 1920 DSP slices instead of 600. This would allow for more intricate digital filter design. Essentially, it was out of the question to redesign this part of the board, since the change of footprint would have required a complete rerouting and rearrangement of most of the components on the board.

## 5.5. Programming and Arbitration

This section describes the programming of the FPGAs through the JTAG interface and from the onboard ROM and the special *arbitration* circuit.

### 5.5.1. Programming

After power-up, the FPGAs are blank and hence not operational, as their *programming* is volatile. Two methods are used to write the firmware to the FPGA:

**JTAG** The JTAG interface, which was discussed in section 4.2 and is shown in Fig. 5.10a, is not only used for debugging, but also for programming of the firmware. With a suitable programmer, which can be attached directly to a computer via the USB port, the process of writing the firmware (*bitfile*) takes a few seconds. After a power cycle the device will be blank again and has to be re-programmed.

**ROM** A Read-only Memory (ROM) is attached directly to each of the FPGAs, as shown in Figure 5.3 on the bottom right of the FPGA. After power-up, the FPGA automatically checks for the presence of this memory on dedicated pins, and tries to load a stored firmware. The ROM is usually used to keep the most recent *stable* firmware, allowing a quick reset of the board. Loading the firmware from the ROM takes less than 1 s.

### 5.5.2. Arbitration

For the PANDA-SADC radiation hardness is a crucial demand of the design, since the boards are placed close to the detector. The FPGA's programming might be affected by ionizing radiation, which in this case is categorized into having transient effects (device not permanently damaged) or permanent effects. A subset of transient effects are Single Event Upsets (SEUs). In case of a SEU, the state of a memory cell (block RAM, flip-flop, configuration memory) is inverted, and the state or programming becomes invalid. The result is virtually unpredictable, it can be that the operation is not affected at all, or it can be catastrophic. Due to a broad usage of Kintex-7 in particle physics, extensive studies regarding the effects on this FPGA can be found (e.g. [154]).

Xilinx offers a solution called *Soft Error Mitigation* [155], which allows to monitor and correct certain kinds of SEUs; still, the device's operational state cannot always be recovered. Measurements and simulations for the PANDA setup predict the Mean Time Between Failure to be 17.8 s for automatically correctable SEUs and 529 s for SEU occurrences that require a complete reconfiguration of the FPGA [156]. Obviously this cannot be recovered by hand during the experiments' runtime.

A solution has been elaborated that allows the PANDA-SADC to recover itself in most situations of this kind. Figure 5.2 shows that there are several connections between the two FPGAs, controlled by the so-called *Arbiter*. The *Arbiter* allows an FPGA to take control not only over the configuration (JTAG) of the other FPGA, but also the programming of the Lmk04806 (PLL) and the selection of the source clock for the Sy89546 clock muxer. This allows one FPGA to take control over the board and fix the programming of the second FPGA, which may be affected by an SEU.

This mechanism does not seem necessary for the installation at the CBELSA/TAPS experiment, as the CB-SADCs will be installed in an area with comparably low radiation background. The arbitration circuit on the CB-SADC is still fully functional, but by default deactivated: upon closer inspection of the left side of Figure 5.10b, a resistor can be identified that is soldered to the `arb off` position, which deactivates the automatic arbitration. In this configuration, only one of the two FPGAs has control, e.g. over programming the PLL (Lmk04806). Furthermore, just below the DIP switch in the same figure, two more resistors are soldered to the `sw` (instead of `arb`) position to configure the source clock selection to depend on the DIP switch instead of the arbiter's selection. By soldering the resistors to different positions, the arbitration circuit can be partly/fully activated as originally intended.

## 5.6. Power Delivery Network

The power delivery network of the CB-SADC comprises the conversion and distribution of different voltages to all components. Its analysis and understanding are important to guarantee a stable operation of the CB-SADC. Under all operating conditions, all circuits need to be supplied with a voltage within their defined acceptance range. At the same time, the efficiency and heat of the regulators has to be optimized.

Section 5.6.1 will sketch the topology of the power distribution, followed by layout considerations in section 5.6.2, which have caused some optimizations in the CB-SADC adaption compared to the PANDA-SADC.

### 5.6.1. Topology

The CB-SADCs are supplied with power through a NIM-crate, which can deliver up to six voltage rails: ±6 V, ±12 V, and ±24 V. The circuits on the SADC, on the other hand, require voltages in the range of 1.0 V to 3.3 V (and −3.3 V for the analog shaping card). Those voltages have to be transformed by regulators on the CB-SADC board.

DC power conversion can be achieved by two topologies: **linear regulators** and **switching regulators**. Linear regulators *burn* the excessive voltage, from the outside they behave like a voltage-controlled current-limiting resistor. Their implementation is trivial, as they only require capacitances on the input and output voltage rails, and furthermore one or two feedback resistors whose value will determine the output voltage. The efficiency is determined by the ratio of the output to input voltage rail:

$$\eta_{\text{lin}} = \frac{P_{\text{out}}}{P_{\text{total}}} \tag{5.13}$$

$$= \frac{U_{\text{out}} * I_{\text{load}}}{U_{\text{in}} * I_{\text{load}}} \tag{5.14}$$

$$= \frac{U_{\text{out}}}{U_{\text{in}}}, \tag{5.15}$$

and the power dissipation depends on the voltage difference

$$P_{\text{PD}} = I_{\text{load}} * (U_{\text{in}} - U_{\text{out}}) \tag{5.16}$$

$$= I_{\text{load}} * U_{\text{in}} * (1 - \eta). \tag{5.17}$$

Hence, large currents and voltage drops lead to a high power dissipation, which requires careful thermal layout of the PCB, or the usage of heatsinks.

An alternative for such situations are switching regulators, which can reach efficiencies of ≈ 90 %, largely independent from voltage drop and load. As they are not employed on the CB-SADC board itself, but only on the pluggable power supply, switching regulators will be explained in more detail in section 6.3.

Figure 5.17 shows a simplified hierarchical overview of the power delivery network, starting with the NIM crate power supply on the top. The FPGAs, ADCs, op-amps, and the LMK04806 are shown as they are responsible for most of the power demand. The linear regulator MAXIM INTEGRATED MAX8556 has been chosen for the PANDA-SADC because of its ripple rejection of 40 dB at 100 kHz and the ability to supply up to 4 A. The TOREX XC6222 on the other hand were chosen mostly because of their comparably small size, which allows them to be placed within a very high density area in the design.

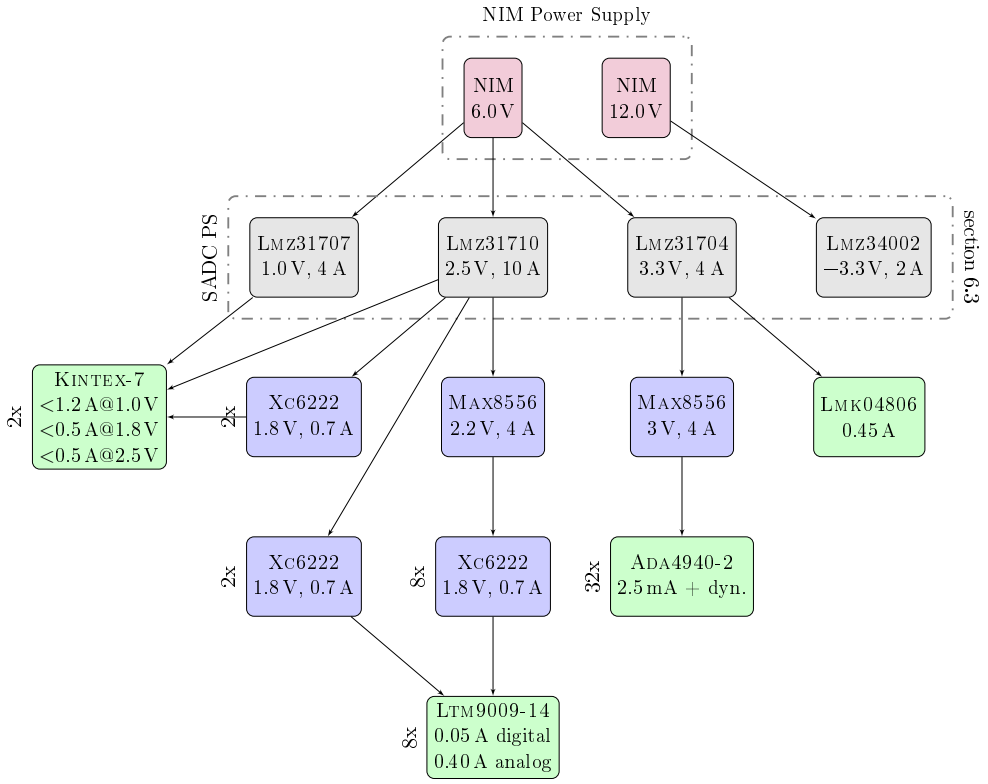



Figure 5.17.: Power Delivery Network of the CB-SADC. Red: voltages provided by the NIM-crate. Gray: switching regulators on the pluggable CB-SADC power supply. Blue: linear regulators on the CB-SADC board. Green: ICs with the highest current demands. The current demand was either measured with the prototype (in case of the FPGA, where it depends heavily on the firmware) or determined from datasheets. For the regulators the maximum output current is stated.

### 5.6.2. Layout Considerations

While signals on a PCB are connected with *traces* with copper thicknesses of 18 µm and width down to 100 µm, the power delivery network usually consists of so-called *polygons*. This technique makes use of the space on the PCB layer which is not occupied by traces, and fills it with a coherent copper area, connected to the power net. Compared to a trace with fixed width, this grants the highest possible conductance. Figure 5.18 shows how this technique can be exploited.

In a high-density design it is sometimes not possible to generously allocate space to the polygon areas, and a balance has to be found between competing nets. The goal is to obey the *minimum supply voltage* limit of the IC which is connected to the power net, as the supply voltage will decrease depending on the resistance of the copper trace or polygon and the current demand of the IC according to Ohm's law:

$$\Delta U = R_{\text{copper}} \cdot I \tag{5.18}$$

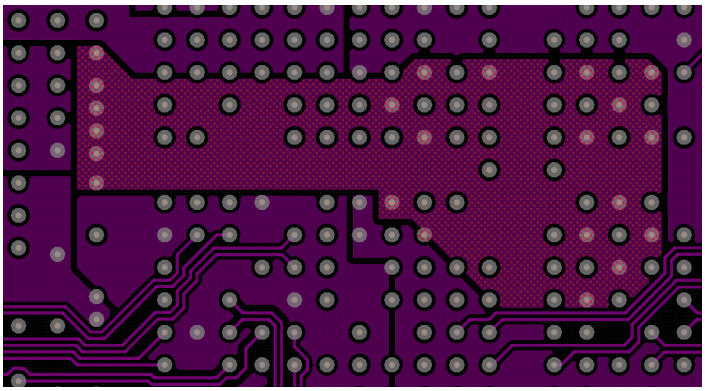


Figure 5.18.: Example for routing with traces and polygons. The snippet shows an area of 16 mm × 11 mm on one of the inner layers of the PCB, beneath one of the FPGAs, to which most of the vias connect. The checkered copper polygon is used to distribute the 1 V FPGA core voltage (`1.0VDU2`), originating from a shunt resistor connected through the six vias on the left, to the FPGA through 16 vias. At the bottom, some differentially routed traces (a pair of traces routed tightly coupled together) can be seen, disrupting the polygons.

### 5.6.3. Simulation

The software in which the PANDA-SADC and CB-SADC PCB layout were created (Altium Designer) features a tool called Pdn Analyzer. This tool can simulate the current flow (and hence voltage drop) across the layout, simultaneously for all nets, allowing to identify and correct possibly problematic routing before the production. The simulation is based on physical parameters specific for the production, such as the layer stackup (copper thickness of the layers), the thickness of the via plating (the galvanically applied conductive coating of the drilled holes, connecting the layers), and the ambient temperature.

To run the simulation, parameters have to be assigned on a schematic level: *nets*, *sources* (power supplies) and *loads* (ICs, LEDs,etc.). Such a definition is shown exemplarily in Figure 5.19 for one of the more than 20 nets. The important parameters for the definition of the *sources* are the voltage, the current source capability, the output resistance, and the thermal limitations. The have been extracted from the datasheets for the switching regulators on the power supply (Lmz31710 [157]) and various linear regulators on the CB-SADC. The definition of the *loads* requires knowledge of the approximate current requirements and the acceptable supply voltage range. The supply voltage range and the static current requirements of most ICs can be extracted from their datasheets, which has been done for the ADCs (Ltm9009-14 [149]), the PLL (Lmk04806 [150]), the muxer (Sy89546 [158]), and the op-amps (Ada4940-2 [8]), being the decisive power consumers.

For the FPGAs, on the other hand, the static current requirement only makes up for a fraction of the total current. The larger dynamic current drain depends heavily on the firmware design and the connected periphery. Furthermore, even the figures for the static current are often only stated as a worst case number, e.g. at the temperature limit of 85 °C, which overestimates the operating conditions.

**FPGA Current Estimation** There are limited possibilities to estimate the current consumption for the FPGA with the help of Xilinx Vivado, which can analyze the currently active firmware design. Figure F.7 (page 264) shows a coarse calculation of the power, which is even broken down into various parts of the FPGA. In principle, this method requires the definition of the internal activity caused by external stimuli that cannot be predicted. In the case of the CB-SADC firmware, for example, one cannot predict the experiment's trigger rate, which in turn influences the rate of network packets being sent, which then defines how often the block RAM is filled and emptied. Clearly this can only be approximated, which has been attempted, knowing from an early prototype firmware that the approximate current for the core voltage of 1 V is 1.2 A. Figure F.8a shows how for this example the activity has been defined by means of a *Toggle Rate* (percentage of time the state is changed) of the respective group of components, and Figure F.8b shows the according calculated currents for the various voltage rails of the FPGA, summing up to $\approx$ 1.2 A (this approach is somewhat heuristic). In later firmwares, where the data rate has been decimated from 80 MS/s to 20 MS/s (see section 7.3.5), the current drain for the 1 V rail has decreased to 0.8 A, proving the strong dependency on the individual firmware.

**Results of the Simulation** The simulation has been executed, taking into account all nets, and all except a few ICs with negligible current consumption (below 2 mA). For the FPGAs, the current demand of the FPGA's core and bank voltages has been measured with a recent firmware (revision `1902201109`) and has been multiplied with a safety factor of two, for more demanding future firmware designs. The results will be discussed as an example for the 1 V net, as it supplies the FPGAs directly and is not too complex. Figure 5.19 shows the simulated circuit including the calculated FPGA core voltage, propagated from the power supply connector through shunt resistors to the FPGA's pins. The FPGAs are split into two independent loads, one represents the core voltage supply, one the supply of the integrated BRAM (Block RAM). The same information is presented in Table 5.1, and a visualization that can aid the improvement of the design is shown in Figure 5.20.

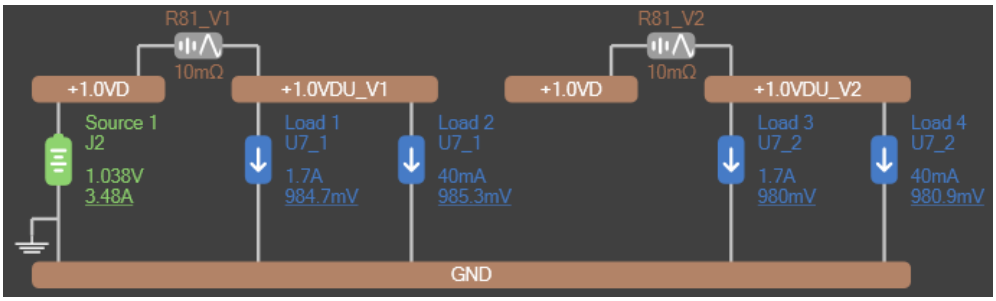


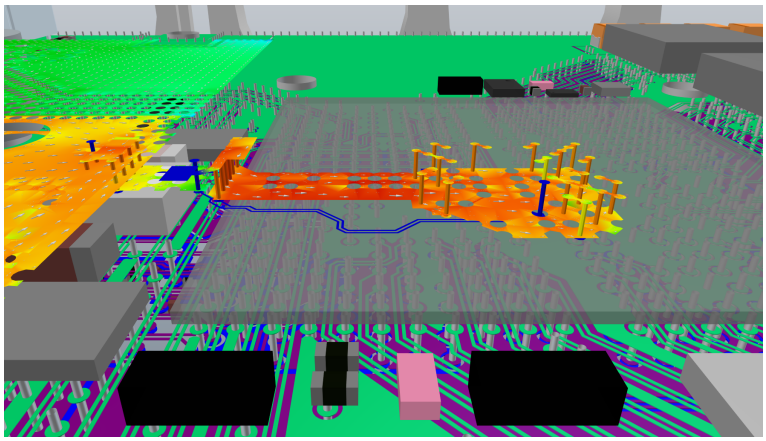

Figure 5.19.: Example of the definition of a circuit for simulation in Pdn Analyzer. The source `J2` of the net `1.0VD` is shown on the left-hand side and is the plug leading to the CB-SADC power supply. The net extends to two independent branches, separated through shunt resistors (`R81_V*`) with an impedance of 10 mΩ. The loads `U7_*` are the FPGAs, split up into the block ram voltage supply with 40 mA and the core voltage supply, with an assumed current consumption of 1.7 A. The underline voltages already show the result of the simulation, a clear voltage drop from the source voltage of 1.038 V can be seen.

The voltage drops by roughly 50 mV, of which 17 mV can be attributed to the 10 mΩ shunt resistor, and the remainder to ohmic loss in the routing. The minimum required voltage of 970 mV [159] is barely fulfilled, the margin is in the 1 % regime. If the current demand increases, the primary voltage will have to be increased; but it must still obey the (absolute) maximum input voltage in case the FPGA is empty and draws (almost) no current.

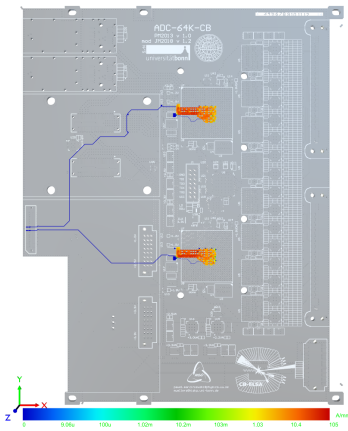
| Sink | Current/A | $V_{min}$/V | $V_{max}$/V | $V_{sim}$/V | Margin/% |
|---|---|---|---|---|---|
| FPGA1 (core) | 1.70 | 0.970 | 1.030 | 984.72 | 1.52 |
| FPGA1 (BRAM) | 0.04 | 0.970 | 1.030 | 985.32 | 1.58 |
| FPGA2 (core) | 1.70 | 0.970 | 1.030 | 980.05 | 1.04 |
| FPGA2 (BRAM) | 0.04 | 0.970 | 1.030 | 980.89 | 1.12 |

Table 5.1.: Results of the PDN ANALYZER simulation for the FPGA's current drain on the 1 V power rail. The margin shows that there is not too much reserve for a higher current demand of the FPGAs.

**Conclusion** The results for all nets show that all criteria are fulfilled and hence the power delivery network has been adequately designed. The majority of the power delivery network has been designed by P. Marciniewski [137] for the PANDA-SADC, but several modifications were made, e.g. for the investigated `1.0VDU2` polygon the voltage drop could be reduced, the complete power net routing around the PLL LMK04806 and muxer SY89546 was redesigned and decoupled with ferrite beads, and several decoupling capacitors were added or moved.



(a) Composite of the board layout (copper traces, vias, and polygons), the components, and the false-color current density visualisation of the `1.0VDU2` polygon. The small arrows show the simulated direction of the current flow.



(b) Board overview. `1.0VDU1` (bot.) and `1.0VDU2` (top) are highlighted.

Figure 5.20.: Example of the visualization of the PDN ANALYZER results. The false-color area shows the current density of the FPGA core voltage supply nets `1.0VDU*`, in this snippet ranging between 1 A/mm$^2$ to 100 A/mm$^2$. The shown polygon is the same as highlighted in Figure 5.18. The blue (low current density) traces are connected to a measurement circuit (cf. Figure 5.24).

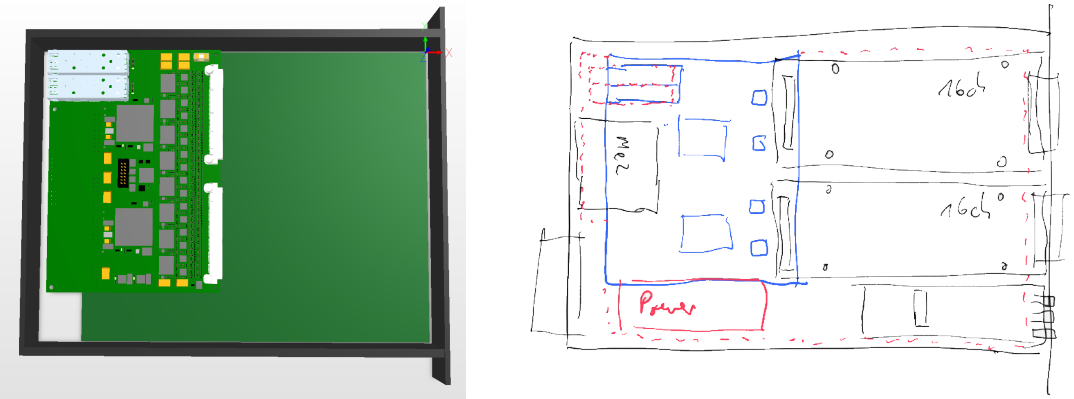
## 5.7. Adaption and Changes of the Design

It would not have been possible to develop the CB-SADC from scratch within the scope of this thesis. Instead, due to the existing core design of the PANDA-SADC in revision `ADC-64K v1.0`, it was possible to concentrate on improving the design and make specific modifications necessary for the CBELSA/TAPS experiment [160]. Some of those changes were already mentioned throughout chapter 5 and will be summarized again in this section. First, the changes in the form factor will be recalled in section 5.7.1. Next, the differences in the power supply will be shown in section 5.7.2. The modifications in the clock distribution will be summarized in section 5.7.3. The addition of trigger connections necessary for the CBELSA/TAPS experiment will be presented in section 5.7.4. Lastly, the slowcontrol features and the development of a heatsink will be presented in section 5.7.5 and 5.7.6.

### 5.7.1. Form Factor

The custom form factor of the PANDA-SADC was optimized in size to fit into a custom made holding structure that also provides cooling. For the CB-SADC it was decided to stick to one of the more popular form factors, such that existing infrastructure can be used. Popular candidates were NIM, VME or ATCA, from which NIM was chosen, as it is very simple to implement. The benefit of VME would have been the ability to transfer the digitized data using the VME bus, but the data throughput would not have been sufficient. The ATCA standard allows much higher data rates, but the implementation is more complex, and no ATCA hardware is being used in the CBELSA/TAPS experiment so far.

Figure 5.21 shows the very first ideas of how the PANDA-SADC could be adapted. The final assembly in the NIM cassette can be seen in Figure A.6 (page 225).




(a) Composite CAD Figure, showing the PANDA-SADC in comparison to a NIM cassette.

(b) Conceptual drawing of the PANDA-SADC inside a NIM cassette, including analog shaping cards (*16ch*), two extensions for slowcontrol/trigger signals (*Mez* and bot. right) and a power supply (*power*). It was expected to have a design with 32 instead of 64 channels.

Figure 5.21.: Early ideas for the CB-SADC adaption of the PANDA-SADC design, as presented on the CBELSA/TAPS Collaboration meeting in March 2014. *Courtesy of Christoph Schmidt.*

### 5.7.2. Power Supply

For the PANDA-SADC the plans for the power supply have evolved in the last years. At the time of the adaption, a 12 V external DC supply was meant to be connected alongside the signals on the front connectors. A switching power supply with air coils (due to the placement in a strong magnetic field) would then transform this to 1 V, 2.5 V, and 3.3 V. Those plans have been changed in the mean time, as the challenges coming with radiation hard design of a power supply suggested a centralized low voltage distribution.

For the CB-SADC the task was straightforward. A switching power supply has been designed to convert the voltages provided by the NIM power supply to the on-board low voltages (cf. Fig. 5.17). Since the development of this power supply and the CB-SADC was going on in parallel, and to keep development costs low, it was decided not to integrate this power supply on the board, but instead to make it pluggable. The development will be shown in section 6.3.

### 5.7.3. Arbiter/PLL

The arbitration circuit, which was introduced in section 5.5.2, is of high importance for the operation in high radiation areas. However, it is not necessary for the installation at the CBELSA/TAPS experiment, as the CB-SADCs will be installed in an area with low radiation background. The arbitration circuit on the CB-SADC is still fully functional, but the circuit has been modified such that there are three options now: *Arbitration active*, *FPGA1 in control*, *FPGA2 in control*. Setting one FPGA to have fixed control simplifies the initialization routines at power-up.

Upon closer inspection of the left side of Fig. 5.10b, a resistor can be identified in the `arb off` position, and some further resistors (not labeled) indeed fix the control to *FPGA1*. Hence, only this FPGA can take control over the programming of the PLL LMK04806.

The selection of the source clock, which originally is also controlled by the arbitration circuit, can now also be selected independently with a DIP switch. In the original PANDA-SADC design, there existed only two source clocks, namely `RCV1` and `RCV2`, and whichever FPGA was active would also be selected as a source clock for the SY89546 clock muxer and hence the LMK04806 clock distribution circuit. For the CB-SADC, the two-channel muxer was exchanged for a four-channel muxer to include two more clock sources (cf. Fig. 5.8).

### 5.7.4. Trigger Connections

In the original design, the PANDA-SADC can connect to the outside world only through the SFP connectors (and the JTAG connector, for programming purposes). It was not foreseen to connect, for example, external trigger signals, since in the PANDA experiment the PANDA-SADC is running in a trigger-less readout: the sampled ADC data is being processed by the FPGAs all the time, and the decision whether or not the event could have been interesting is taken by the SADC itself. A later stage that is called the *Data Concentrator* then takes care of the assembly of the information from all connected SADCs and revises all events.

For the CB-SADC, it is necessary that the firmware running on the FPGAs can communicate with the trigger/sync system as explained in section 2.9.4. This requires at least **seven** incoming (*Event, Sysreset, 5-bit Buffer Number*) and **one** outgoing (*Busy*) signal. Note that the *OK* signal is not taken into account, as on this level it is redundant to the *Busy* signal.

Although roughly 40 FPGA pins were left unconnected in the PANDA-SADC design, it was challenging to select those that could be routed in the right direction due to dense routing and power polygon placement. It was possible to select 16 signals per FPGA. Up to revision `ADC-64K-CB v1.1` those were connected to bi-directional level translators (Texas Instruments Txb0108). The level translators serve two purposes: first, they translate the FPGA bank voltage of 1.8 V to a more common logic level of 3.3 V; second, they protect the FPGA from wrongly applied voltages (e.g. accidentally applied over-voltage only destroys the level translator, but not the FPGA). For the production revision `ADC-64K-CB v1.2` the routing has been changed. Both will be discussed in the next paragraph.

**Routing** The routing of the signals is shown in Figure 5.23a. For each FPGA, the 16 signals are connected to two 8-channel Txb0108 level translators, and from there to the *Extension Connectors* on the left-hand side. Additionally, half of the signals are also connected to a third Txb0108 level translator and routed to the *Front Connector* on the bottom right. Each of the level translators can be deactivated with a DIP-switch located at the back of the CB-SADC (see Figure 5.22), which allows, for example, to route half of the connections to the *Extension Connectors*, the other half to the *Front Connector*. It has proven to be difficult to transmit such single-ended signals without interferences to adjacent traces (shown in Figure 7.25). These problems could only be faced by limiting the slew rate of the signal source, but this workaround has not been investigated regarding its applicability under all circumstances. It seemed advisable to choose one of the four following solutions for the final version of the CB-SADC for better reliability of the trigger connection:

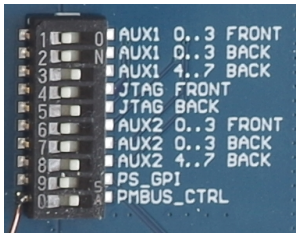



Figure 5.22.: Dip-switch on CB-SADC.

1. Improve the routing such that the traces are sufficiently shielded from each other.
2. Tie every other trace to GND for shielding.
3. Use differential signals on the path from FPGAs to the connectors.
4. Use differential signals with so-called cross-point switches.

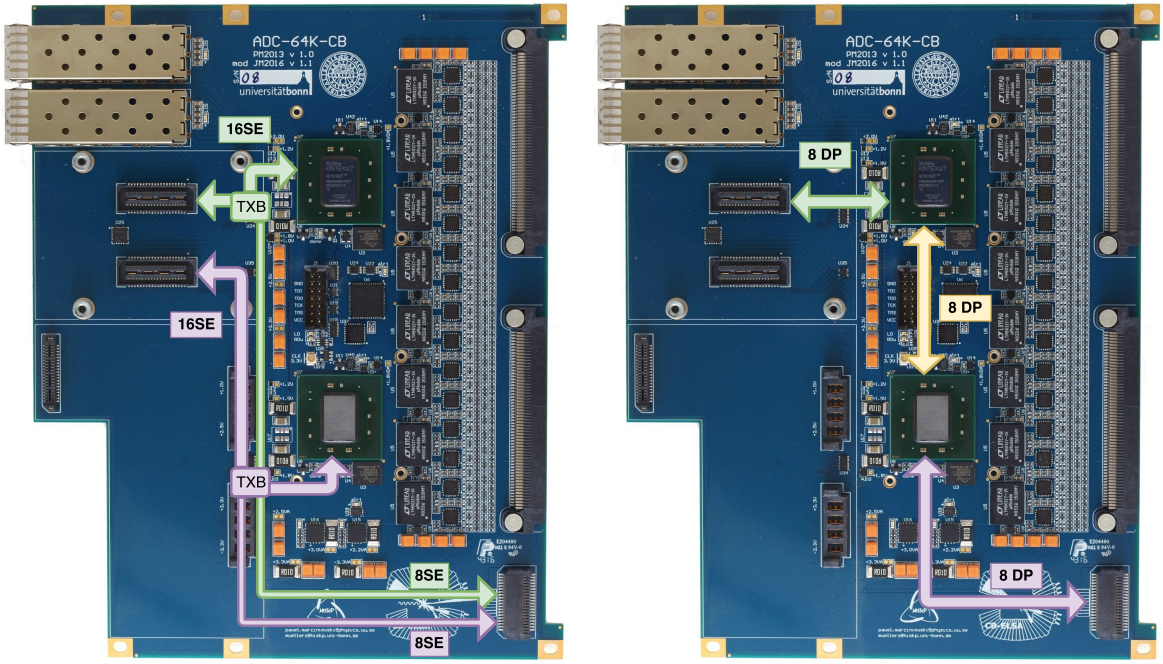
The **first** solution is unrealistic to implement, as the space available for routing is constrained on the board. The **second** solution could work, but would halve the number of signals. The **third** solution uses differential routing, in which the tightly coupled differential pair does not emit disturbances in the far field, i.e. to the next differential pair. This solution would not allow to switch the routing between the *Extension Connectors* and the *Front Connector* anymore with the Txb0108, and would also halve the number of signals. The **fourth** solution is the most complex, but also the most flexible solution, but it requires some effort for implementation and would have to be evaluated with a prototype. The cross-point switches would allow to route signals dynamically from the FPGAs to the connectors, or even from FPGA to FPGA, or connector to connector; with one drawback: the direction of any connection has to be predetermined. Due to the complexity and inflexibility regarding the direction, this solution did not seem to be desirable.

Taking into consideration that there is an additional and up to now unused bus of eight differential pairs between the two FPGAs, the third option seemed to be promising when

the layout is changed as shown in Figure 5.23b: from the upper FPGA, all differential pairs are routed to the *Extension Connector*, from the lower FPGA, all are routed to the *Front Connector*. The bus between the FPGAs is used to transmit the information to each other.

This allows the greatest flexibility regarding the routing, with the only drawback being that the FPGA pins are directly connected and in this sense unprotected against overvoltage. With this solution it is in principle possible to operate the eight differential pairs again as 16 single-ended signals, with the same caveat as in the original design (requirement for limited slew rate due to crosstalk issues). This change has been implemented for the production revision `ADC-64K-CB v1.2` and leads to an incompatibility with the prototype revisions `ADC-64K-CB v1.1` and `ADC-64K-CB v1.0`. This can be resolved only by modifying the firmware and the electronics interfacing to the *Extension Connectors* and the *Front Connector*.

**Conclusion** The design change has allowed fast and reliable connections from the trigger electronics to the CB-SADCs; and the numerous workarounds to deal with crosstalk problems on an electronic level and in the firmware can now be dismissed.




(a) Trigger connections in revision `ADC-64K-CB v1.1`. 16 single-ended signals from each FPGA are connected to level translators (TXB0108). From there, up to 16 each can be routed to the extension connector, and up to 8 each can be routed to the Front Connector.

(b) Proposed alternative trigger connections for the production revision `ADC-64K-CB v1.2`. Eight differential pairs are routed from each FPGA to one of the connectors, and the pre-existing connection with eight differential pairs between the FPGAs is used for data exchange.

Figure 5.23.: Routing changes for the external signals (trigger, slowcontrol, programming) on the CB-SADC. The signal can be access at the *Extension Connectors* (top left below the SFP metal cages) and the *Front Connector* (bottom right).

### 5.7.5. I²C Slowcontrol

The concept and necessity of a *slowcontrol* has been discussed in section 2.11.2. In terms of the CB-SADC it is interesting to monitor different temperatures, voltages, and currents on the board, which is not only useful for runtime-monitoring, but also during the testing procedures. In addition, the power supply (section 6.3) and the analog input card (section 6.1) will feature circuits that are controlled in their behavior via this slowcontrol.

The original PANDA-SADC design did not feature any externally accessible sensors or circuits. Although there are two temperature sensors and interfaces for configuration of the LTM9009-14 ADC chips and the LMK04806 PLL chip, they are only connected directly the FPGAs and use the Serial Peripheral Interface (SPI) protocol, hence they are only accessible through the network interface and require the FPGAs and the network to be operational. To overcome this limitation, an independent I²C-based bus has been added to the design. The I²C bus is easy to implement, as it consists only of two signals, serial data (`SDA`) and serial clock (`SCL`) (cf. section 2.11.2).

The next two paragraphs will show how I²C is used for the monitoring of temperature and voltage rails (voltage and current of supply voltages), followed by an explanation of the bus structure and sensor placement on the CB-SADC.

**Temperature Monitoring** It is important to observe the temperatures on the PCB, since an uncontrolled environment may lead to overheating and finally destruction of the board, including a possible fire hazard. Apart from that, the temperature monitoring has proven to be very helpful for the development phase, for example in the development and assessment of the heatsink for the CB-SADC (see section 5.7.6).

Four temperature sensors (ATMEL AT30TS750A) have been placed across the board, close to hot-spots that were identified with an infrared camera. With a resolution of 0.0625 °C, subtle temperature differences could be noticed in different test scenarios. The temperature sensor offers a so-called *alert* output, which can change its state according to a programmable temperature threshold. For each of the four sensors, an LED has been connected to this output, which was helpful for detecting overheating when the CB-SADC was operated in the laboratory with limited air-flow cooling (the LED can be seen for example in Figure 5.10a on page 79, top right corner). The threshold was programmed to 50 °C, leaving ample headroom to critical temperatures.[3]

**Voltage Rail Monitoring** The monitoring of voltages and currents across the CB-SADC allows to assess whether all regulators are operating within tolerance. Moreover, they provide an indication when the programming of an FPGA has failed, which will be reflected in a less-than-expected increase of the current. The monitoring of the voltages is controlled completely by the power supply, which features a 16-channel monitoring circuit (see section 6.3). Certain voltages on the CB-SADC are routed directly to the power supply connector for measurement, whereas the currents are measured through shunt resistors with a TEXAS INSTRUMENTS INA197/8. This amplifier is necessary since the voltage drop on the shunt resistor should be as low as possible, but still the full scale of the measurement circuit should be exploited for best resolution.

[3] The lowest maximum operating temperature for any component is 70 °C for the LTM9009-14 ADCs.

**Layout** The layout will be explained with the help of Figure 5.24. The $I^2C$ bus can be connected either through the Extension Connector or the Front Connector. Through a four-channel $I^2C$ switch (Texas Instruments Pca9546) the bus is split in three branches.

**The upper branch** connects to two of the temperature sensors, one for the upper FPGA (T_FPGA2), one for monitoring of the Lmk04806 PLL circuit (T_LMK), and to the FPGA itself (FPGA2). Through its firmware the FPGA can provide information to the slowcontrol as an $I^2C$ slave. The branch also connects to one of the analog input cards (BLV) (chap. 6.1).

**The lower branch** is identical except that the second temperature sensor is placed close to the linear regulators for the analog voltage of the digitization circuits (T_LDO).

**The power supply branch** connects to the monitoring circuit on the power supply and is responsible for measuring the Current & Voltages on the CB-SADC.

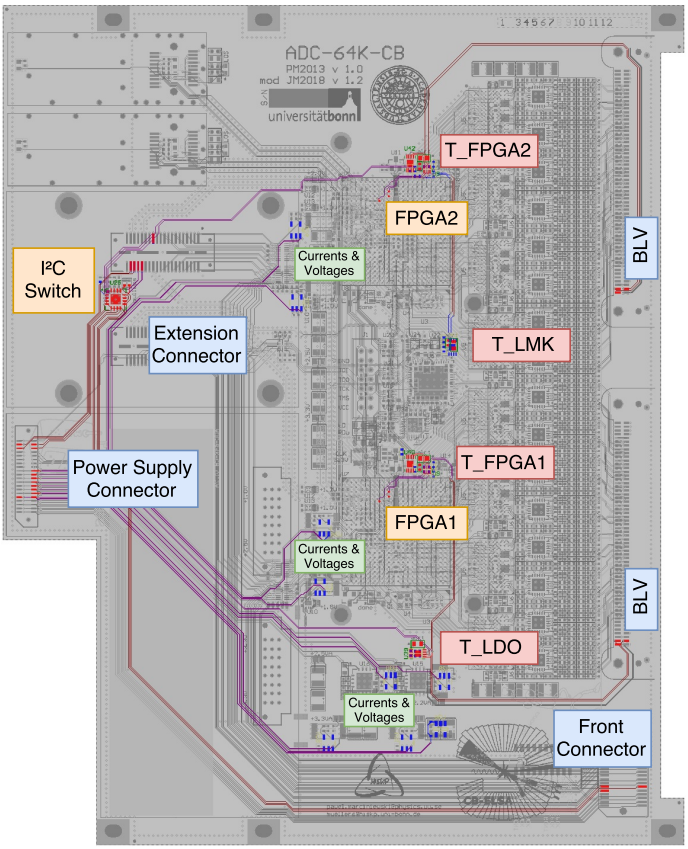



Figure 5.24.: Overview of the slowcontrol routing and the connected devices. The blue boxes indicate handover interfaces for the $I^2C$ signal. The red boxes are placed next to the temperature sensors. The areas close to the green boxes contain the shunt resistors and amplifiers for the current measurement. The yellow boxes indicate the $I^2C$ switch and the connection to the FPGAs. The different colors of the traces indicate routing on different layers.

### 5.7.6. Heatsink

The CB-SADC consumes roughly 20 W, which is dissipated as heat. Since most of the heat dissipation is concentrated to small areas, a heatsink is necessary for a more efficient cooling supported by the NIM crate's air ventilation. The simplest solution is mounting commercially available heatsinks with heat conducting tape, which has been done for the first prototype, `ADC-64K-CB v1.0`. This did not seem ideal, as it was not possible to equip the smaller ICs like the op-amps and the clock muxer, and furthermore the PLL LMK04806 (which sits between the FPGAs that are *shielding* it from the airflow) has proven to be a critical hot-spot. This can also be observed in Fig. 5.25 (only the left FPGA is programmed).

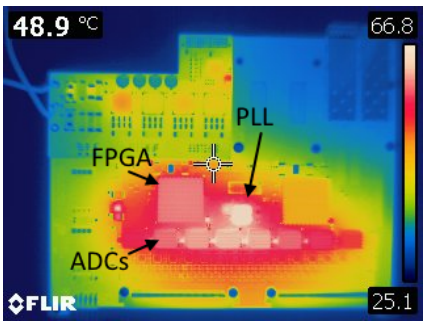



Figure 5.25.: Infrared pseudocolor image of the `ADC-64K-CB v1.0`.

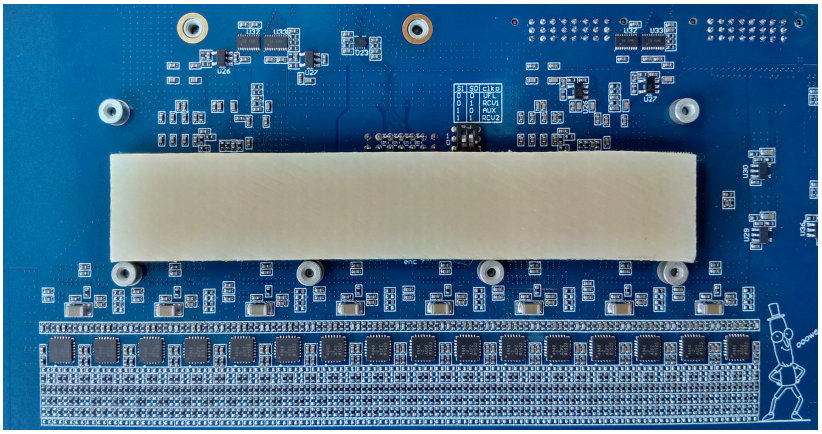
**Development** The development goal is to reach temperatures of at most 60 °C, leaving 10 K headroom to the ADC's maximum operating temperature. Custom heatsinks were designed using the CAD software AUTODESK INVENTOR and were produced in the institute's workshop. Two revisions are shown in Figure 5.26. On the bottom side they have a specific height profile, which has been extracted from the position and height of the components on the CB-SADC. This allows thermal coupling with rather thin heat conducting pads (thickness 0.5 mm, allowing to compensate for slight deviations in the height). In the first version (5.26a), the 3D body was extruded for every component, but this is not ideal in terms of machine time during production. In the second version (5.26c), the 16 extrusions for the ADA4940-2 op-amps have been consolidated into one, and the extrusions for the LTM9009-14 ADCs were removed as they were the tallest components. Moreover, an extrusion for the SY89546 clock multiplexer was added and the total area has been increased. Those modifications have lead to an average temperature improvement of $> 4\,°C$.

The heatsink is mounted to the CB-SADC with six M2 screws that were not part of the original PANDA-SADC design. They can be identified on Fig. A.6 (p. 225). Eight additional holes in the heatsink above LEDs (4x temperature warning, 2x bootup, 2x regulator failure) are filled with a light guide to improve the LED's visibility.

**Coupling to NIM Cassette** Although the heatsink has been optimized in a few revisions, the temperature of the board as indicated by the sensors sometimes surpassed 60 °C. This is partly due to the non-uniform airflow provided by the NIM crate, using three fans to distribute the airflow to twelve NIM slots. As an example, an average temperature difference of 5.9 °C has been observed between slot 17/18 and 19/20 (cf. Figure 5.27b). To lower the temperatures further, a heat conductive pad was applied at the back side of the CB-SADC, which can thermally couple the PCB to the back cover of the NIM cassette [161] (cf. Figure 5.27a). To bridge the distance of roughly 4 mm, a so-called gap-pad of type BERGQUIST GP2500S20-0.160-02-0816 with a thermal conductivity of $2.4\,\mathrm{W\,m^{-1}\,K^{-1}}$ has been used. As shown in Figure 5.27c this has lead to improvements in the range of 5 °C to 15 °C depending on the sensor and NIM slot. In total it can be concluded that the thermal management of the CB-SADC is sufficient for long-term operation with temperatures even well below 50 °C.

(a) V1, bot. (b) V1, top. (c) V2d, bot. (d) V2d, top. (e) V2d, lined up to be equipped with heat conducting pads.

Figure 5.26.: Two revisions of the heatsink for the CB-SADC.



(a) Location of the pad on the back of the PCB. The size of the pad is 200 mm × 10 mm × 4 mm. It is surrounded by the six nuts which hold the heatsink on the top side of the PCB.

| # | ID | FPGA1 | FPGA2 | LMK | MAX |
|---|---|---|---|---|---|
| 1 | d57597 | 53.812 | 50.125 | 56.188 | 52.750 |
| 2 | d50b60 | 55.250 | 47.438 | 58.875 | 52.688 |
| 3 | d52ee1 | 48.562 | 45.125 | 51.500 | 47.000 |
| 4 | 000000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 5 | d5394a | 47.312 | 44.062 | 48.938 | 43.000 |
| 6 | d51057 | 53.062 | 50.000 | 56.438 | 52.500 |
| 7 | d52064 | 49.625 | 47.188 | 53.312 | 47.875 |
| 8 | 000000 | 0.000 | 0.000 | 0.000 | 0.000 |

(b) Without pad.

| # | ID | FPGA1 | FPGA2 | LMK | MAX |
|---|---|---|---|---|---|
| 1 | d57597 | 43.062 | 41.375 | 43.750 | 45.375 |
| 2 | d50b60 | 43.250 | 41.125 | 43.438 | 46.375 |
| 3 | d52ee1 | 41.812 | 39.875 | 42.625 | 43.500 |
| 4 | 000000 | 0.000 | 0.000 | 0.000 | 0.000 |
| 5 | d5394a | 38.812 | 38.375 | 39.875 | 39.062 |
| 6 | d51057 | 42.938 | 41.750 | 43.312 | 44.750 |
| 7 | d52064 | 41.812 | 40.938 | 43.250 | 43.250 |
| 8 | 000000 | 0.000 | 0.000 | 0.000 | 0.000 |

(c) With pad.

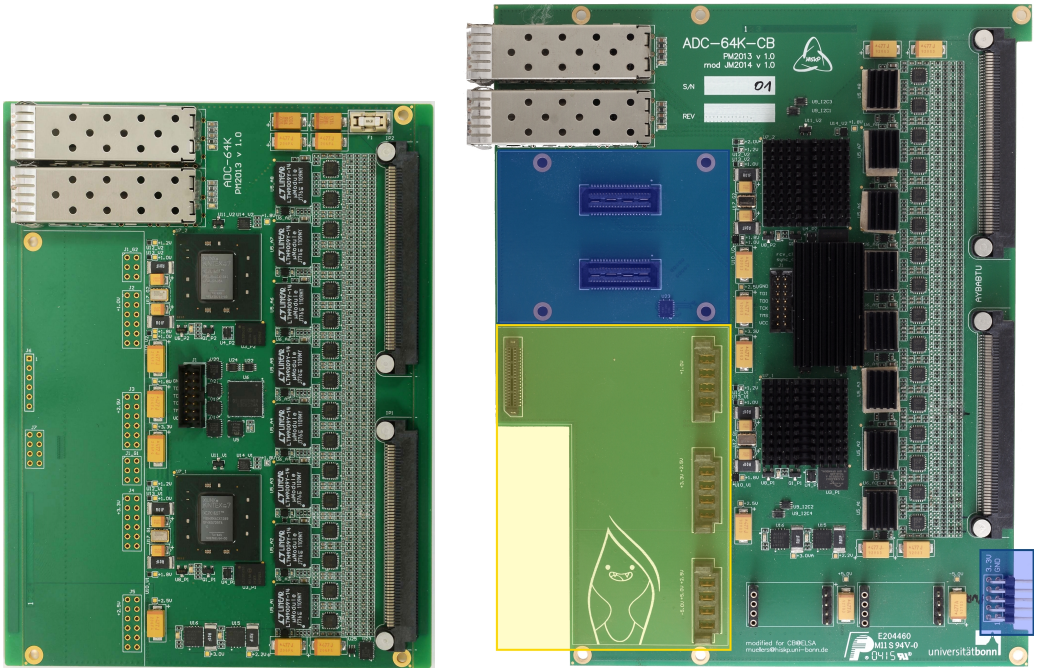
Figure 5.27.: Effect of mounting a thick conductive pad on the back of the CB-SADC. The sensor readouts show the unique ID of six CB-SADCs (in adjacent NIM slots), followed by the four temperature sensors as shown in Fig. 5.24 (MAX→ LDO).

## 5.8. Revision Summary

Two CB-SADC prototype revisions and the final revision have been produced within the scope of this thesis. In the following sections, the differences between those three revisions will be summarized. Additionally, a special revision of the CB-SADC has been produced, which was equipped with 125 MS/s instead of 80 MS/s digitizers. The outcome of the tests will also be presented briefly.

### 5.8.1. Revision 1.0

In the first revision that was produced in March 2015, the main changes were the form factor (NIM), the new pluggable slot for the power supply, and the addition of the Crystal Barrel trigger and slowcontrol signals. Those changes are visualized in Figure 5.28. The $I^2C$ bus has been added already in this revision and was connected to the FPGAs and the connector towards the analog front end, but no sensors were integrated on the CB-SADC at the time. Luckily no bugs prevented initial operation, and a smaller issue in the arbitration circuit (which was present also in the PANDA-SADC) could be fixed with additional wiring. Currently this prototype is being used in a detector setup for teaching purposes ("*Student's experiment*") [162].




(a) Original PANDA-SADC revision `ADC-64K v1.0`.

(b) CB-SADC revision `ADC-64K-CB v1.0`, equipped with small heatsinks.

Figure 5.28.: Comparison of the original PANDA-SADC prototype (`ADC-64K v1.0`) and the first revision of the CB-SADC (`ADC-64K-CB v1.0`). The *Extension Connector* for trigger and slowcontrol signals, as well as the *Front Connector* with status signals, are highlighted in blue; the new power supply connector in yellow.

### 5.8.2. Revision 1.1

The first revision (`ADC-64K-CB v1.0`) was produced mainly for a change of form factor and the testing of the SADC in regard to the firmware. The focus for the revision `ADC-64K-CB v1.1` was on specific improvements for the CB-SADC and early operation at the CBELSA/TAPS experiment, and almost all changes/improvements described in section 5.7 can be attributed to this revision. Photographs throughout chapter 5 were almost exclusively from this revision, e.g. Figure 5.3 on page 71.

The following list summarizes the changes:

**Analog Signal:** An option to allow a dual-range setup was added, using only every second input channel, but connecting it to two ADC inputs. This option is mandatory for the PANDA-SADC, but will not be used for the Crystal Barrel Calorimeter (the corresponding resistors are not assembled). Still, it might prove to be useful if the CB-SADC will ever find a use in the readout of another detector, for which the dynamic range in a single-range setup is not sufficient.

**Clocking:** The clock multiplexer was exchanged to allow four instead of two clock sources, adding the possibility to connect a clock in the laboratory (with a U.FL socket) or through the *Extension Connector*. The power delivery network of the PLL (LMK04806) was improved for better decoupling and jitter performance. Status LEDs were added to get optical feedback from the programming of the LMK04806, helping to identify problems with invalid configurations. Testpoints were added on the back of the PCB to measure the performance of the clock signals with a differential oscilloscope probe.

**Connectivity:** The small connector at the front, only used for status signals, was replaced by a high density SAMTEC MEC8 connector ("*Front Connector*") to carry also trigger, slowcontrol, and programming signals. Level converters (TXB0108) were added between the FPGAs and the *Extension Connector* and *Front Connector*, changing the voltage level to 3.3 V and protecting the FPGAs from overvoltage. An unused high speed serial interface (*GTX*) was routed from each FPGA to the extension connectors for future purposes.

**Power:** All tantalum capacitors on the CB-SADC were replaced by either Multi Layer Ceramic Capacitors or tantalum polymer capacitors to prevent degradation (since the liquid electrolyte diffuses after some time).

**Mechanical:** Six M2 nuts were added to the PCB to allow for mounting of the custom heatsink.

The revision 1.1 was produced in a batch of seven boards and shipped in October 2016. Two minor bugs (a network LED indicating *no connection* instead of *connection*, reference voltage of a DIP switch) had to be fixed after the production.

Six of the board have been installed in the experimental hall and were used for most of the measurements presented in chapter 9. The seventh board was equipped with 125 MS/s ADCs (see next section) and has remained in the laboratory for firmware tests.

### 5.8.3. 125 MS/s Test

One of the seven boards of the revision `ADC-64K-CB v1.1` is equipped with Linear Technologies Ltm9011-14 ADCs instead of Ltm9009-14, which can run up to a sampling rate of 125 MS/s instead of 80 MS/s (thus allowing an analog signal with a bandwidth of up to 62.5 MHz according to the Nyquist limit). Although this sampling rate is not necessary for the readout of the Crystal Barrel Calorimeter, the test results were interesting concerning the possible application in other detectors, which may profit from a bandwidth of more than 40 MHz (for example the MiniTAPS calorimeter).

Furthermore, this test assembly provided the first test results for the development of the latest PANDA-SADC revision, which was designed to have an assembly option for the 125 MS/s Ltm9011-14. It has been shown that the assembly with Ltm9011-14 was successful. Modifications in the firmware and the programming of the clock output frequencies of the Lmk04806 have indeed allowed for taking data with 125 MS/s. Unfortunately, one point has not been considered carefully enough and became apparent only during the tests: the current drawn by the Ltm9011-14's analog circuit is typically 582 mA, compared to 395 mA for the Ltm9009-14. For eight ICs this adds up to more than 4.6 A, which is above the 4 A specification of the used linear regulator Max8556. Since the actual current limitation circuit inside the Max8556 allows a current of at least 5 A [163], and the careful thermal design was able to handle the resulting increase in temperature, this has not been a problem during tests in the lab. Moreover, when operated at 80 MS/s instead of 125 MS/s (which is possible), the current is again below 4 A, such that after testing, this prototype has been used alongside the six other boards for regular operation.

For the PANDA-SADC, the problem was solved by adding a second Max8556 linear regulator to the board, and splitting the current into two rails.

### 5.8.4. Revision 1.2 (Production Revision)

For the production revision, the changes described in Figure 5.23 (page 94) have been implemented to allow a safer transmission of the trigger signals. First loop-back tests have shown that the transmission of the signals works as expected. Apart from that, changes were minuscule to avoid a further prototype iteration:

**Front Connector:** Changed from Samtec MEC8 to the higher density connector MEC6, after it was clear that the *back*plane will be moved to the *front* of the CB-SADC, requiring more signals to be made accessible (this will become clear in section 6.4.3).

**Linear Regulators:** Different regulators were chosen at multiple places to have a footprint with improved solderability, and for better margin reserved for the current demand of the high speed interfaces (*GTX*) in the FPGA.

In addition, some minor and mostly cosmetic problems that appeared in the previous revision have been fixed.

# 6. Related Hardware Development

The CB-SADC project has been accompanied by four smaller projects that were developed, assembled, tested, and used within the scope of this thesis. In this section, all four projects will be presented and their task in connection with the CB-SADC will be made clear.

The Analog Input Card, which is plugged in the CB-SADC from the front, is responsible for the analog processing of the signals from the Crystal Barrel Calorimeter and will be introduced in section 6.1. Section 6.2 will present a test adapter for quality assurance of the Analog Input Card, together with a preview of the test results. Section 6.3 will show the development and evolution of the CB-SADC's power supply from a first prototype to the final version. Section 6.4 will present the evolution of the CB-SADC Crate Controller, which connects the CB-SADCs to the trigger and slowcontrol signals of the CBELSA/TAPS experiment, and allows access to the FPGA's programming interface. In section 6.5 a slowcontrol extension developed for stand-alone tests of the CB-SADC will be discussed.

## 6.1. Analog Input Card

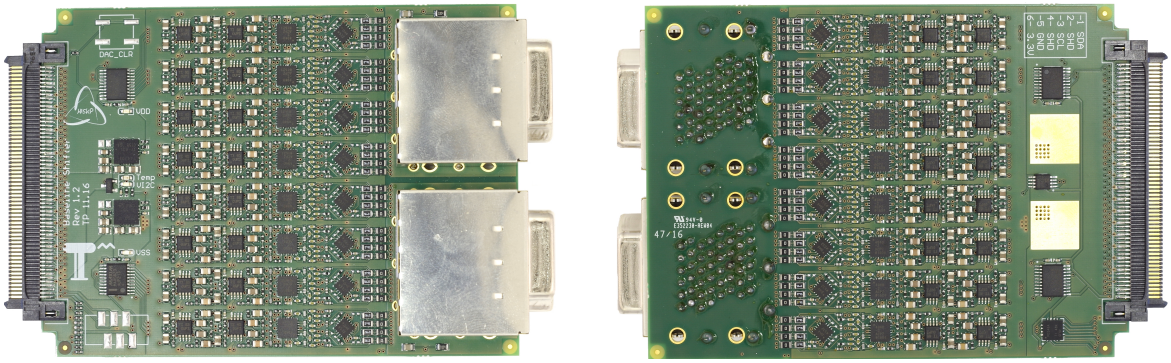
The Analog Input Card is responsible for *shaping* the preamplified signals from the Crystal Barrel Calorimeter, before they are digitized by the CB-SADC. The goals of shaping are **shortening the pulse** to reduce pile-up, and **increasing the SNR** for a better signal quality. At the beginning of the CB-SADC's development, it has been decided that such a shaping stage will be separated from the main PCB of the CB-SADC, using the Samtec Erm/f8 connector. Several points have led to the decision:

1. The shaping architecture was not fixed yet and it was unknown whether the shaping stage, as it was on the PANDA-SADC, could be used for implementing all desired shaping options. Alternatively, an option was to consider having fully digital shaping by means of Digital Signal Processing (DSP) algorithms implemented in the FPGAs.

2. The input connectors (Mrj21) could not be determined before the development of the BuffTi (section 2.8.5) with matching connectors was completed (2015).

3. The circuits that allow baseline shifting, or even a variable pole-zero cancellation, were not yet developed and could not be integrated into the (first) CB-SADC prototype.

In November 2015, the circuit, simulation, and layout of a first prototype of the Analog Input Card have been completed within the bachelor thesis of T. Poller [139]. His thesis included the baseline adjustment and was later complemented by the shaping stage and a circuit for a variable pole-zero cancellation. The shaping stage will be summarized in section 6.1.1. The $I^2C$-controllable baseline shifter and pole-zero cancellation will be shown in section 6.1.2 and 6.1.3, respectively. Figure 6.1 shows the resulting assembled PCB, which has been produced 14 times and was used for prototype tests and the beam times in 2018/19.



(a) Top side, ERM/F8 (left) and MRJ21 (right). (b) Bottom side.

Figure 6.1.: Analog Input Card for the CB-SADC. Between the ERM/F8 and the MRJ21 connector one can identify on both sides: DACs for baseline setting and positive/negative voltage regulator (1st column, top only), low-pass filter for the baseline adder circuit (2nd/3rd column), op-amps with band-pass filter (4th column), potentiometers for pole-zero cancellation (5th column).

### 6.1.1. Signal Shaping

Signal shaping refers to band-pass filtering of the analog signal. The aim of the shaping is to shorten the pulse to achieve pile-up reduction, and to increase the SNR. The Analog Input Card is equipped with a passive high-pass filter at 160 kHz responsible for shortening the signal, followed by an active low-pass filter at 160 kHz and a passive low-pass filter at 7.2 MHz to increase the SNR. The architecture is shown in Fig. 6.2). In the next paragraphs the aims and filter parameters will be discussed in more detail.

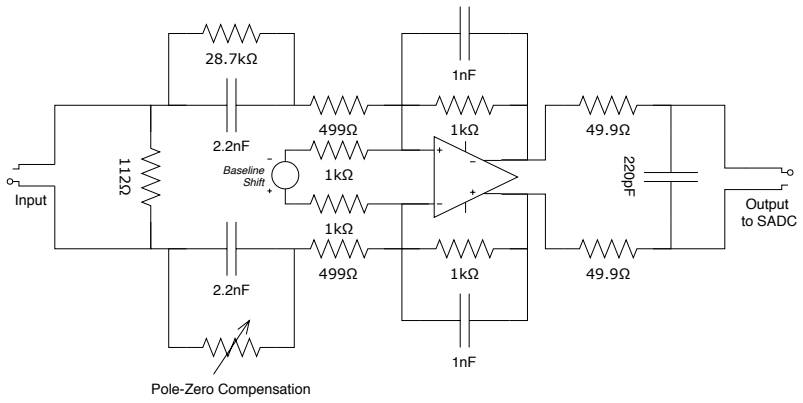
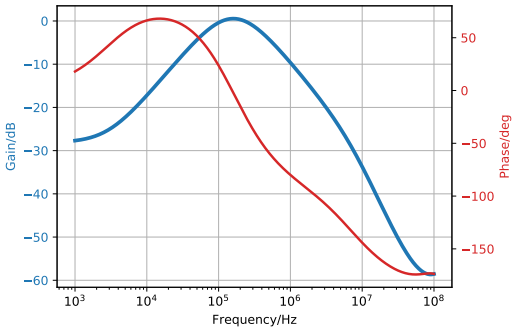


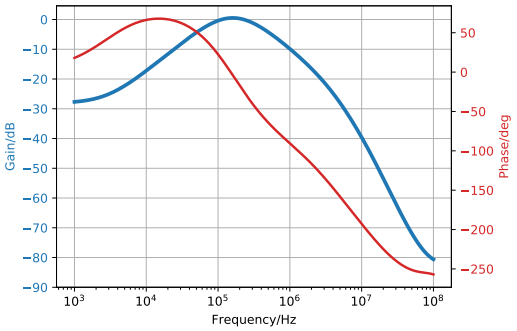

Figure 6.2.: Shaping circuit on the Analog Input Card, left to right: termination resistor, high-pass with pole-zero cancellation, input resistors with baseline adder, ADA4940-2 op-amp with active low pass, output series resistor with passive low pass. The detailed circuit for baseline shifting can be found in [139], and for the pole-zero cancellation in Figure 6.10.

**Improving SNR:** It was shown in Figure 2.13a (page 40) that the signal contribution is negligible above 1.5 MHz. Following the argument in section 1.4.6, it is advisable to have a low-pass filter close to this frequency. It was shown in [97] that for a certain filter (1st order high-pass, 4th order Bessel low-pass, noiseless) the maximum SNR is achieved with a step response peaking time of 1.5 µs, which corresponds to a CsI(Tl) signal peaking time of 2.3 µs and a shaping time of 0.64 µs. Simulations with the Analog Input Card's filter architecture have yielded a center frequency of 160 kHz for the band-pass (passive high-pass, active low-pass) as a good compromise to obtaining similar performance [164]. The filter frequency of the passive low-pass is set to 7.25 MHz, followed by a passive low-pass on the SADC board (see section 5.2.1) with 8.0 MHz. Bode-plots of the filter chain can be seen in Fig. 6.3a and 6.3b. With the filter on the CB-SADC, a suppression of −67.8 dB is reached at the Nyquist limit of 40 MHz. The noise contribution of the Analog Input Card has been determined with a simulation framework (see Figure F.1 on page 259). Figures 6.3c and 6.3d show the noise density of the circuit, without and with the buffer and low-pass filter on the CB-SADC. The resulting total noise contribution of the whole filter chain in the range of 50 Hz to 1 GHz is

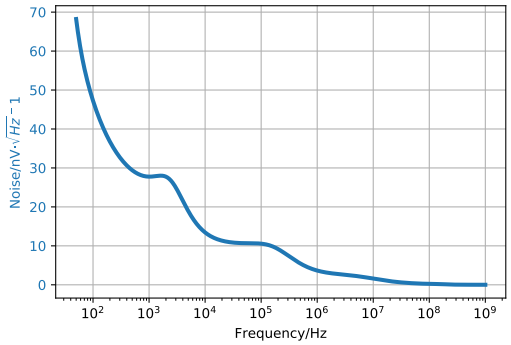
$$U_{\text{noise,RMS}} \approx 37.4\,\mu\text{V}. \tag{6.1}$$



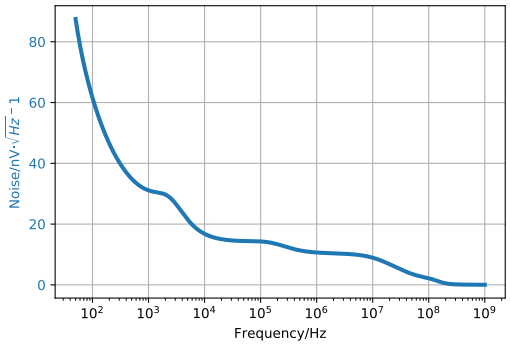

(a) Filter on Analog Input Card.




(b) Filter on Analog Input Card and CB-SADC.




(c) Filter on Analog Input Card.




(d) Filter on Analog Input Card and CB-SADC.

Figure 6.3.: Bode plot and noise density after shaping. The Bode plot shows the frequency-dependent gain and phase shift of the filter. The frequency-dependent noise density can be integrated into the total noise contribution.

**Pile-up reduction:** The preamplifier's signal has an exponential decay time of 54 µs (see section 2.8.3). Hence the signal will drop below 1 % of its amplitude after roughly 250 µs. A second signal appearing within this window will sit on top of the falling slope, and it will be more difficult to analyze its amplitude compared to the situation where it sits on the flat baseline. With the AC-coupling (high-pass filter), this slow exponential tail can be cut off, which is shown in Figure 6.4. The effective window, in which a second signal would have to be considered as pileup, is now one order of magnitude shorter. In principle it is also possible to cut off a purely exponential tail by means of digital algorithms (*moving window deconvolution*) implemented in the FPGA. An investigation, where data has been recorded without AC-coupling during a beam time, has revealed that the tail cannot be described well with a purely exponential function (just one time constant), which has lead to unsatisfactory results with an attempted moving window convolution. This issue will be discussed in more detail in the dissertation of J. Schultes [111]. It might be that a more thorough investigation will lead to a more sophisticated *digital* shaping and a removal of the analog high-pass in the future.

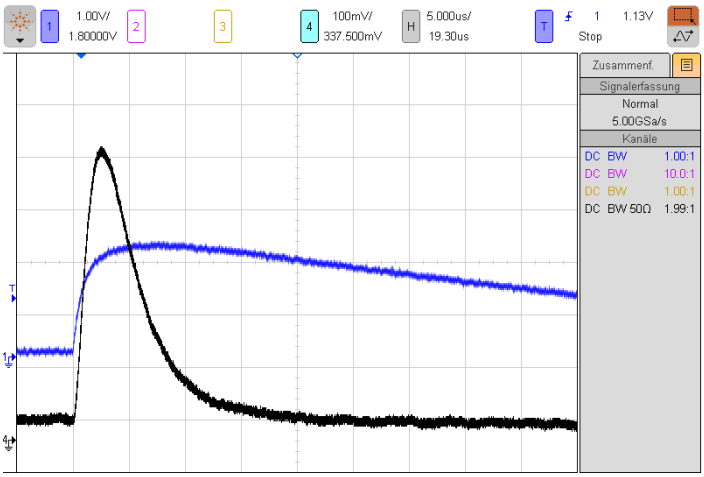



Figure 6.4.: Effect of band-pass shaping. The blue signal is a recorded CsI(Tl) preamplifier pulse from an arbitrary waveform generator, with an exponential tail of $\tau \approx 54$ µs. The black signal is the output of the Analog Input Card with its band-pass filter. The time per division is 5 µs.

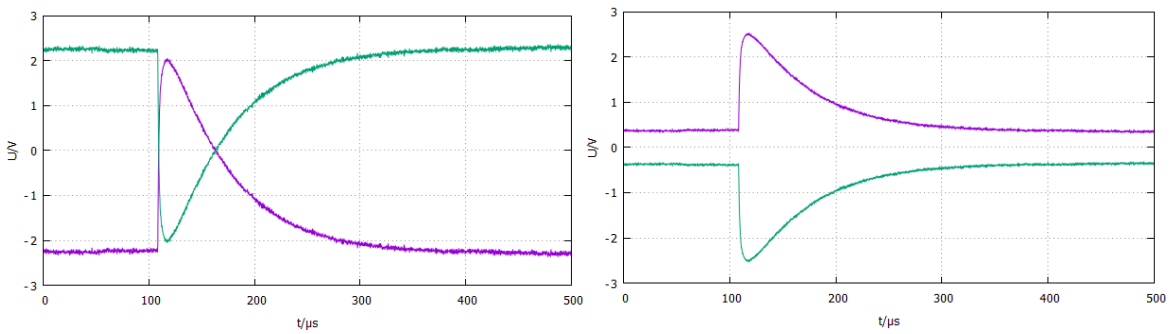
### 6.1.2. Baseline Shifting

To save power in the driver's circuits of the line driver and the BuffTi module (see sections 2.8.3 and 2.8.5), it was decided to refrain from using a truly differential signal transmission. In a truly differential transmission, the positive and negative signal's dynamic range is symmetric around the same voltage level, which is shown exemplary in Fig. 6.5a. For most of the time the detector signal is close to the baseline, which would lead to a continuous current flow through the termination resistors on the Analog Input Card. This is an additional load for the driver circuit, and also leads to heat produced on the Analog Input Card.

The signal levels for the transmission have been shifted such that the baseline is close to 0 V for both the positive and the negative signal (see Fig. 6.5b), which solves this problem.

As the ADC chip on the CB-SADC has a truly differential input, the signals will only fill half of the dynamic range in this configuration. To take advantage of the full dynamic range, it is necessary to shift the signals after the termination. It has been investigated in [139] how this can be achieved. The simplest (passive) method is adding a DC bias to the signal. This can only be achieved in combination with a capacitive coupling (high-pass) and can furthermore lead to a substantial output impedance. Instead, a classical adder-circuit with an op-amp was implemented, leading to a very low and frequency-independent output impedance.

To allow remotely-controlled adjustment of the baseline shifting, it is not sourced by a fixed voltage, but rather an 8-bit DAC, which is connected to the CB-SADC's $I^2C$ bus. The DAC output voltage is low-pass filtered (two orders), such that as little noise as possible is added to the signal. Further details regarding the implementation can be found in [139].



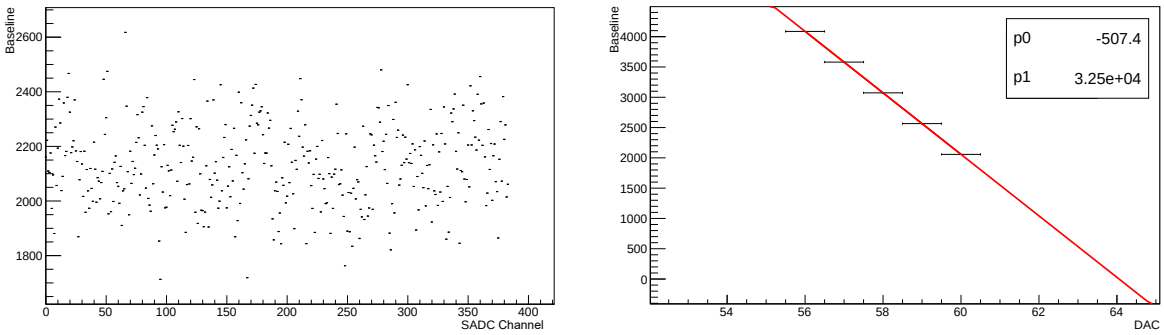

(a) Truly differential transmission. (b) Shifted differential transmission.

Figure 6.5.: Differential transmission of the CsI(Tl) preamplifier signal. With a 100 Ω termination between the differential signals, the transmission as shown in (a) will lead to a substantial power dissipation, hence in the experiment the signals are shifted as shown in (b). On the Analog Input Card the signals have to be shifted back to situation (a) to use the full range of the fully differential ADCs. [139]

**Automated Analysis and Adjustment** The behavior of the baseline shifting circuit has been investigated within M. Biel's bachelor thesis [165]. The goal of the thesis was the development of an automated baseline determination based on the DAC's setting, which in return allows to determine the ideal DAC setting for achieving a desired baseline value.

For a first analysis, which was conducted with six CB-SADCs installed in the experimental hall, all DACs have been set to the same value. This has has led to a distribution of baselines for all 384 channels with a span of roughly 900 on a 16-bit scale (see Figure 6.6a).

Figure 6.6b shows the linear correlation between DAC setting and baseline value. It can be deduced that with an ideal setting the baseline distribution should be containable to a span of roughly 500, already improving the previous setting. It is clear that the dynamic range of the DAC cannot be fully exploited, since only DAC settings between 0 and $\approx 63$ can be used; larger values will yield a negative baseline which cannot be measured by the SADC. This is shown in Figure 6.6b. In the final revision this shortcoming was resolved by replacing the DAC by its 12-bit version, allowing a much finer setting of the baselines.




(a) *Width* (span) of the baseline distribution with the same DAC setting for all channels.

(b) Correlation between DAC setting and baseline level, shown for one channel.

Figure 6.6.: Investigation of the baseline distribution and dynamic range. The y-axis is expressed in units of amplitude, ranging from 0 to $2^{16} - 1$. [165]

### 6.1.3. Pole-Zero Cancellation

When investigating the signal after band-pass shaping closely, it can be noticed that a weak but very long undershoot appears after the signal (cf. Figure 6.11). This is caused by the response of the high-pass in combination with the exponential decay of the unshaped CsI(Tl) preamplifier signal. In this section it will be shown how this is dealt with on the Analog Input Card. The ideal step-response of the high-pass filter on the Analog Input Card is an exponential decay with a decay time of

$$\tau_{\text{diff}} = R_{\text{diff}} \cdot C_{\text{diff}} \approx 1.1\,\mu\text{s}. \tag{6.2}$$

However, a typical charge preamplifier as used in the Crystal Barrel Detector (see section 2.8.3) does not produce a step function. Since it consists of an integrator circuit as shown in Fig. 6.7a, instead, it exhibits an exponential decay with a time constant of

$$\tau_{\text{preamp}} = R_{\text{f}} \cdot C_{\text{f}} \approx 54\,\mu\text{s} \tag{6.3}$$

As neither the CsI(Tl) scintillating light nor the APD's signal are a step function, this is again idealized, and in reality the signal is convoluted, for example, with the decay time of the scintillation mechanism.

Due to the much shorter $\tau_{\text{diff}} \approx 1.1\,\mu\text{s}$, the preamplifier signal will still supply a negative trend after the high-pass filter's own decay has diminished, which leads to an undershoot. This undershoot can be compensated by adding a fraction of the preamplifier signal to the shaped signal with a resistor $R_{\text{pz}}$ in parallel to $C_{\text{diff}}$. The effect is shown in Figure 6.7b.

Mathematically, this introduces a *zero* which compensates the *pole* of the preamplifier with

$$\tau_{\text{preamp}} = \tau_{\text{pz}} \tag{6.4}$$

$$R_{\text{f}} \cdot C_{\text{f}} = R_{\text{pz}} \cdot C_{\text{diff}}, \tag{6.5}$$

which can be expressed as the transfer function

$$G(s) = \frac{1 + s \cdot \tau_{\text{pz}}}{1 + s \cdot \tau_{\text{preamp}}} = 1 + \frac{s \cdot (\tau_{\text{pz}} - \tau_{\text{preamp}})}{1 + s \cdot \tau_{\text{preamp}}}. \tag{6.6}$$

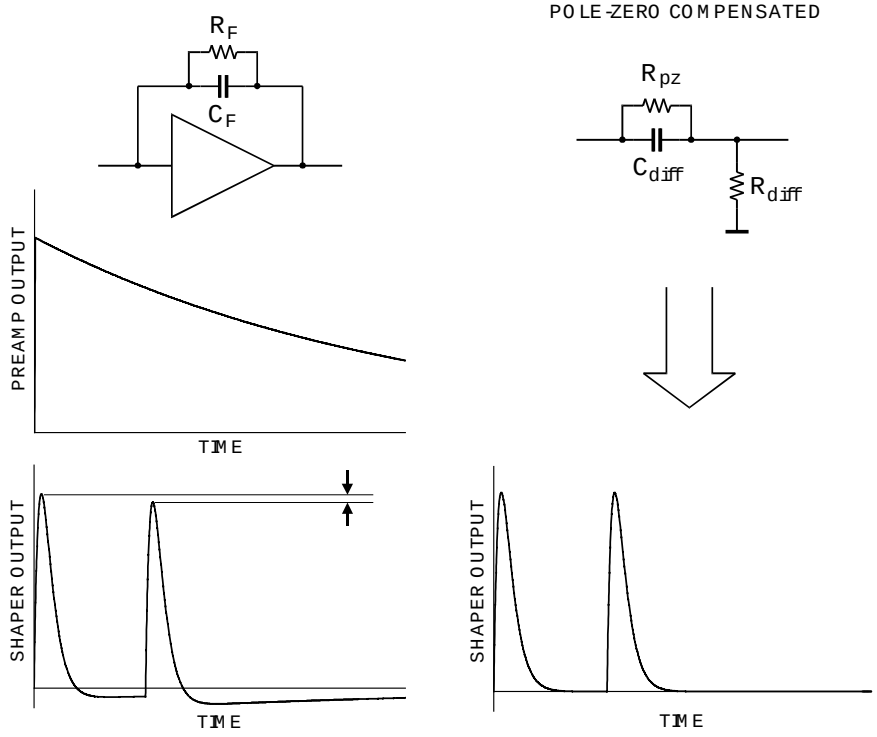
From the formula, it immediately follows that $G(s) = 1$ for $\tau_{\text{pz}} = \tau_{\text{preamp}}$. For the high-pass filter of the Analog Input Card this would result in a pole-zero cancellation resistor of

$$R_{\text{pz}} = \frac{\tau_{\text{preamp}}}{C_{\text{diff}}} \tag{6.7}$$

$$= \frac{54\,\mu\text{s}}{2.2\,\text{nF}} \tag{6.8}$$

$$\approx 24.5\,\text{k}\Omega. \tag{6.9}$$

A simulation with a recorded preamplifier pulse has yielded a seemingly better cancellation with $R_{\text{pz}} \approx 28.7\,\text{k}\Omega$. This may be due to the convolution with the decay time constant of the CsI(Tl) signal, or the variation of the preamplifier's decay time constant (at some point it was suspected that there is a larger variance in the decay time due to production of the preamplifier boards in two separate batches, but it was not conclusively investigated). It seems advisable to implement a solution with a variable resistor, such that with varying conditions all channels produce waveforms of similar shape. In particular, the widths of the pulses should not vary, as they might become relevant, e.g. for time-over-threshold measurements or particle identification.




(a) An integrating preamplifier, leading to a long exponential tail. After shaping with a high-pass, an undershoot occurs.

(b) The high-pass (differentiatior), consisting of $C_{\text{diff}}$ and $R_{\text{diff}}$, is complemented by a resistor $R_{\text{pz}}$ to compensate undershoot.

Figure 6.7.: The effect of compensating the *pole* of the preamplifier's integrator with a *zero* in the shaping circuit. Figures taken from [6].

**Implementation of the Cancellation** In the former Crystal Barrel Shaper, a possible variation is dealt with by the implementation of a mechanical potentiometer (used as a rheostat), which allows fine-tuning. Evidently, a mechanical potentiometer has two severe drawbacks:

- space occupancy (roughly $1\,\mathrm{cm} \times 1\,\mathrm{cm}$ per channel)
- on-site operation necessary.

Since the CB-SADC's high channel density with 64 channels, distributed to four MRJ21 connectors, leaves very little space on the front panel of the NIM cassette, mechanical potentiometers have not been considered as an option. Instead, the possibility to exchange the mechanical potentiometers by digital potentiometers/rheostats has been explored, see for example [166]. Such digitally controllable potentiometers allow a fast remote operation, allowing automated measurements as for the baseline shifting.

**Selection of the Digital Potentiometer** The selection criteria for the digital potentiometer were the following:

- occupies less than $6\,\mathrm{mm} \times 6\,\mathrm{mm}$ for two channels
- bandwidth of at least $\approx 10\,\mathrm{kHz}$ such that the exponential slope with $\tau_{\mathrm{preamp}} \approx 54\,\mu\mathrm{s}$ can pass through unattenuated
- setting via the $\mathrm{I}^2\mathrm{C}$ bus together with the baseline shifter circuit.

A promising candidate was (and is) the ANALOG DEVICES AD5142A [167] with a footprint of $3\,\mathrm{mm} \times 3\,\mathrm{mm}$, two 8 bit potentiometers, a full scale resistance of $10\,\mathrm{k}\Omega$ or $100\,\mathrm{k}\Omega$, and a $-3\,\mathrm{dB}$ bandwidth of 430 kHz (for the $100\,\mathrm{k}\Omega$ variant). The functional block diagram of this potentiometer is shown in Figure 6.8a, indicating the $\mathrm{I}^2\mathrm{C}$ interface, an internal memory, and the two independent potentiometer registers (called RDAC).

**Integration** Figure 6.8b visualizes that each of the two potentiometers in fact consists of two rheostats with resistive elements that are switched by CMOS gates. Since the two rheostat registers can be controlled independently, the idea came up to also use each rheostat independently in the circuit, thus effectively having a four-channel rheostat. Unfortunately, the two rheostats still share a common wiper port which does not allow this implementation.

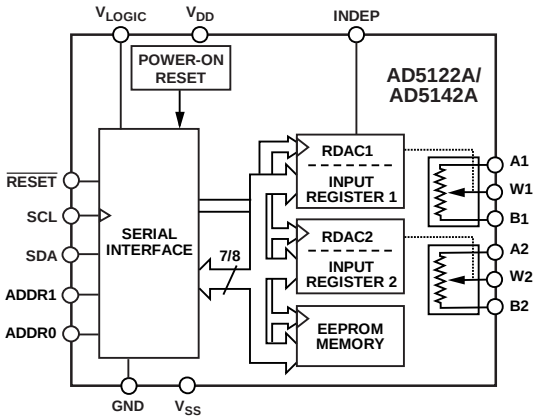
Nevertheless, the application can still benefit from this architecture, as the rheostats could be operated either in parallel or in series. Since the parallel operation of both rheostats is technically possible, but not explicitly advertised in the datasheet, it was carefully tested. To decide which of both is more suitable, one has to consider the benefits of each solution.

Putting two rheostats in **series** allows an extended range of resistance with linear steps over the whole range, which will be either $200\,\mathrm{k}\Omega$ or $20\,\mathrm{k}\Omega$. Knowing that the resistance should be around $25\,\mathrm{k}\Omega$, the operation of a series configuration with a total of $20\,\mathrm{k}\Omega$, in series with a third fixed resistor of $15\,\mathrm{k}\Omega$ would yield a viable range with a step size of $\frac{10\,\mathrm{k}\Omega}{2^8-1} \approx 39\,\Omega$:

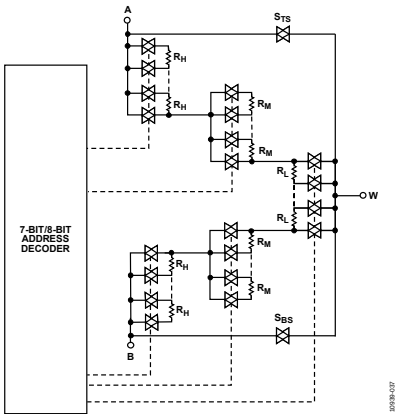
$$R = R_{\mathrm{fixed}} + R_{\mathrm{poti1}} + R_{\mathrm{poti2}} \tag{6.10}$$

$$= 15\,\mathrm{k}\Omega + 10\,\mathrm{k}\Omega \cdot \frac{RDAC1}{255} + 10\,\mathrm{k}\Omega \cdot \frac{RDAC2}{255} \tag{6.11}$$

$$\approx 15\,\mathrm{k}\Omega + 39.2\,\Omega \cdot (RDAC1 + RDAC2) \tag{6.12}$$




(a) Functional block diagram, showing the two independent RDAC registers, programmed through the $I^2C$ serial interface. Their content dictates the potentiometer setting.




(b) Internal circuit of the RDAC implementation, showing the independent resistor chains from terminal A to W(iper) and B to W.

Figure 6.8.: Block diagrams of the Analog Devices Ad5142a. [167]

In a **parallel** configuration, the range of the rheostats will be either 5 kΩ or 50 kΩ, from which the first one has been excluded out of precaution due to the limited dynamic range. The 50 kΩ step size would be $\frac{50\,\mathrm{k\Omega}}{2^8-1} \approx 196\,\Omega$ when both rheostats are configured to the same value, which is shown in Fig. 6.9a. Additionally, is possible to select a coarse range with one of the RDAC registers and make a fine adjustment with the other, which leads to a non-linear effective step size that can be finer than the 39 Ω of the series configuration (e.g. with a configuration of RDAC1=63, changing RDAC2=199 to RDAC2=200 yields a step of 23 Ω). This is visualized in Fig. 6.9b and can be calculated with the formula

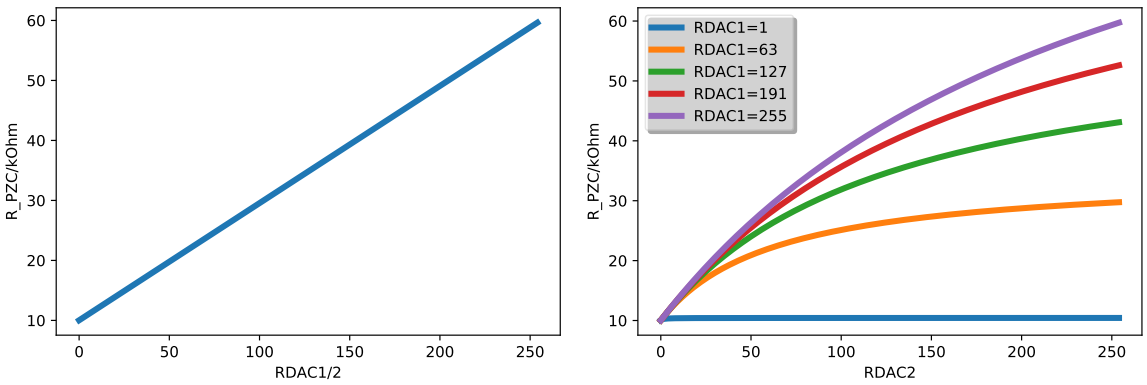
$$R = R_{\text{fixed}} + \frac{1}{\frac{1}{R_{\text{poti1}}} + \frac{1}{R_{\text{poti2}}}} \tag{6.13}$$

$$= 10\,\mathrm{k\Omega} + \frac{1}{\frac{1}{100\,\mathrm{k\Omega}\cdot\frac{RDAC1}{255}} + \frac{1}{100\,\mathrm{k\Omega}\cdot\frac{RDAC2}{255}}} \tag{6.14}$$

$$\approx 10\,\mathrm{k\Omega} + \frac{392\,\Omega}{\frac{1}{RDAC1} + \frac{1}{RDAC2}}. \tag{6.15}$$

Thus, with careful configuration, the parallel implementation yields a wider range and can still have an even smaller step size compared to the serial configuration. The resulting circuit is shown in Figure 6.10 with a series resistance of 10 kΩ (which might eventually be decreased or even omitted for the production version).

Unfortunately only one of the differential paths has this variable resistance, while the other has a fixed value of 28.7 kΩ. Differentially, an ideal cancellation can be achieved. What should not be neglected is the deterioration in the common mode rejection. Due to the asymmetric paths, a 50 Hz hum was identified in the experiment in varying strengths depending on the rheostat setting. Before the final production of the Analog Input Card, the fixed resistor was chosen to be as close as possible to the ideal rheostat setting (25.5 kΩ).




(a) Rheostats programmed to the same value.

(b) Rheostats programmed independently.

Figure 6.9.: Achievable pole-zero cancellation resistance in a parallel configuration, depending on the rheostat settings. With both rheostats programmed at the same value, the dependence is linear and a range of 10 kΩ to 60 kΩ can be obtained. The behavior is similar to a serial configuration with a constant step size. Setting both rheostats to different values allows a finer adjustment when the range is carefully chosen.

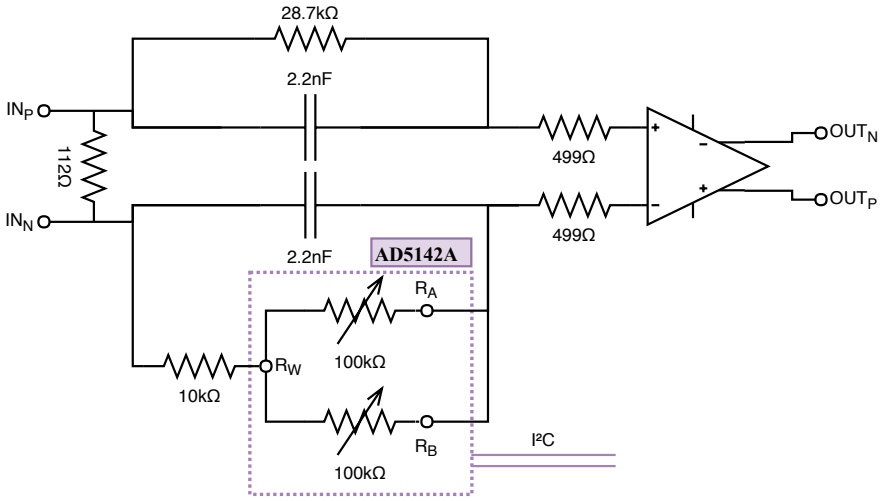
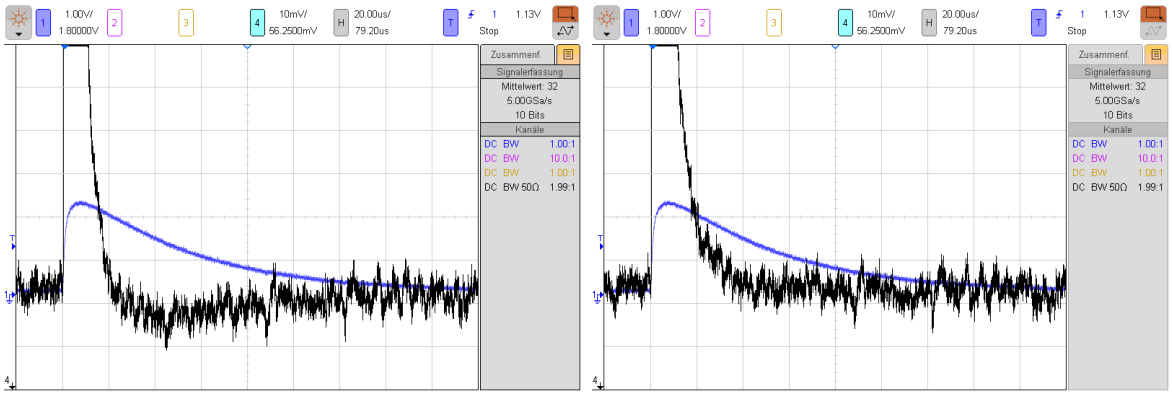



Figure 6.10.: Circuit of the digitally adjustable pole-zero cancellation, compare Figure 6.2 for context. In the negative path, the two rheostats are connected in parallel, together with a 10 kΩ series resistor. In the positive path, a fixed resistor with 28.7 kΩ was chosen in the prototype. It has been replaced by 25.5 kΩ, which reflects the most probable value and results in a better cancellation. The improved symmetry yields a better common mode rejection of −50 dB in the worst case (before: −45 dB in the best case).

Figure 6.11 shows the effect of the circuit on the pole-zero cancellation in a test setup in the lab. Indeed the undershoot is visible and can be compensated by a suitable setting. The next paragraph will show the first attempts to optimize the rheostat setting in conjunction with the CBELSA/TAPS experiment.



(a) Weak cancellation, undershoot clearly visible. (b) Undershoot compensated.

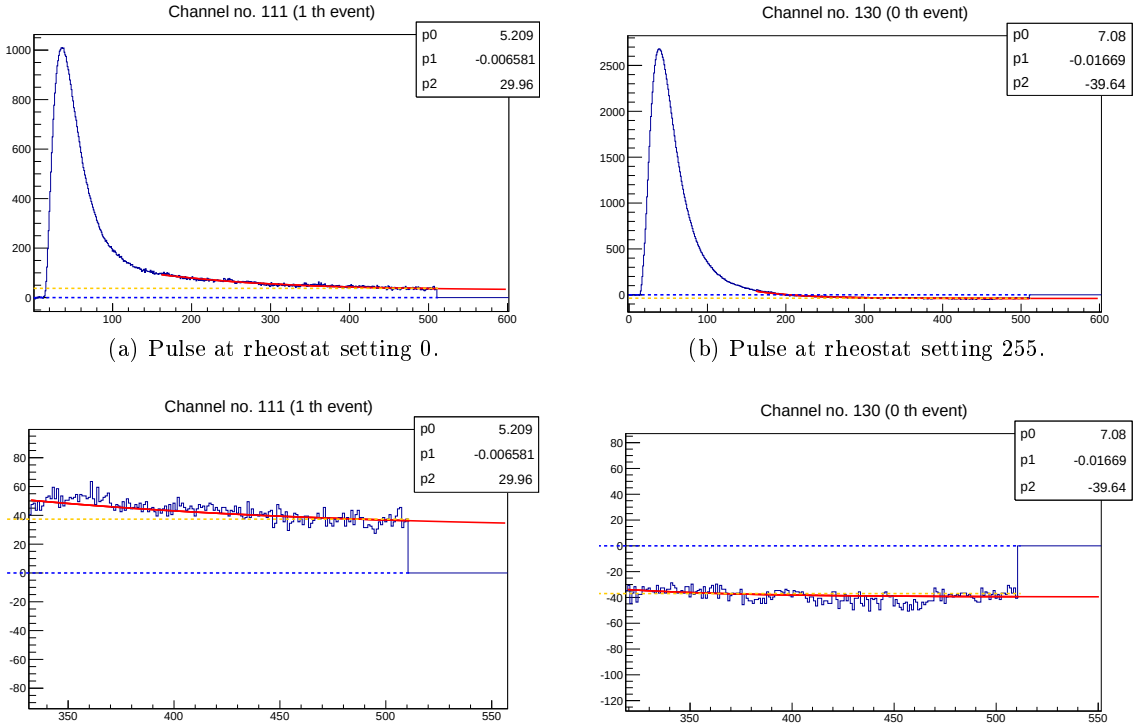
Figure 6.11.: Effect of the pole-zero cancellation, with the preamplifier signal in blue (recorded pulse from a measurement of cosmic particles at the Crystal Barrel Calorimeter, played back on an arbitrary waveform generator) and the signal after shaping with the Analog Input Card in black. Compared to Fig. 6.4, the time/division is increased by a factor of four (20 µs), and the amplitude/division for the shaped signal is decreased by a factor of ten (10 mV).

**Automated Adjustment Procedure** Following the automated baseline adjustment, M. Biel [165] has also devised an automated analysis procedure to determine the ideal setting of the rheostats for the pole-zero cancellation. The methods have first been evaluated with *light pulser* runs, where a defined intensity of light from an LED is injected in to the CsI(Tl) crystal. This allows to quickly generate the necessary statistics even for different intensities. For a more ideal setting it is necessary to investigate either pulses during a beam time, or alternatively fall back to cosmic muons, as they have a more similar shape to the pulses created by photons (compared to the *light pulser*). In any case, the necessary statistics for the adjustment can be collected in a matter of hours.

Multiple analysis methods have been investigated, all based on the observation of the exponential tail of the pulse. In a simpler method, the last samples in the sampling window have been averaged and compared to the baseline that is determined before the start of the pulse. The ideal cancellation would be if the difference of both is zero, or to be more exact, a value close to zero, depending on the height of the observed pulse and the position of the exponential decay. Another method was investigated, which involved an exponential fit to the tail, such that the base level of this fit could be compared to the baseline. This method revealed that a simple exponential function does not describe the tail perfectly, which is attributed to the convolution with the characteristics of the scintillation process and the APDs.

The methods are visualized in Figure 6.12 by showing the two extreme rheostat settings. The measurements were, at the time, averaged over all channels. Both methods were unfortunately (but unsurprisingly, due to the mentioned drawbacks) not conclusive: the first method has yielded an ideal rheostat setting with an average value of 78 to 79, the second method 66.

The methods were re-evaluated with a sampling window of increased length after a modification of the CB-SADC's firmware. This was done for a number of settings with the first method during the October 2018 beam time. The sampling window was chosen to be four times longer, giving a much better indication of the baseline level after the pulse. It was revealed that the pole-zero cancellation was in fact still under-compensated (the undershoot was still visible), and the method has now yielded ideal values much closer to the result of the second method. The ideal setting was now also determined channel-wise, since due to fluctuations of the resistances, the ideal settings might be different from channel to channel. The optimized settings have been applied in the beam time in spring 2019.




(a) Pulse at rheostat setting 0.

(b) Pulse at rheostat setting 255.

(c) Tail at rheostat setting 0.

(d) Tail at rheostat setting 255.

Figure 6.12.: Investigation of the under-/overcompensation in a sampling window of regular length (25.6 µs), comparing the baseline (blue dashed) to the averaged signal level at the end of the tail (yellow dashed) and an exponential fit (red). The level of cancellation can be grasped on a quantitative level, but it is evident from (c) that the exponential tail is still falling with considerable steepness at the end of the sampling window. The sampling window ends after 512 samples, explaining the discontinuity at the end of the window. [165]

## 6.2. Testboard

Within her master thesis, Y.-C. Wang [140] has developed and tested a PCB to aid the mass testing of the final CB-SADC boards in conjunction with the Analog Input Card. This so-called *Testboard* will be presented in this section, including a PYTHON framework to perform tests and analyses. Before this testing procedure was developed, all CB-SADCs and Analog Input Cards have been tested *by hand*, involving a number of consecutive steps with different auxiliary equipment and software. The tests included (list not exhaustive)

1. Identification of short circuits on both boards before initial power-up
2. Measurement of voltages on test points of both boards
3. Readout of all $I^2C$ voltage, current, and temperature sensors
4. Measurement of the noise level
5. Test of the shaping characteristics and the gain
6. Test of the baseline shifting circuit
7. Test of the pole-zero cancellation circuit

The first two steps, which involve probing the board with a multimeter, are not worth to be automated, since they will take only a few minutes per CB-SADC. The third step could be automated more easily, at least if there is access to the CBELSA/TAPS experiment's slowcontrol, as will be shown in section 8.5. In reality, the last four steps have proven to be quite tedious: first of all, to test the channels individually, it was necessary to plug a test signal to each of the 64 inputs. Secondly, the effects of shaping, baseline shifting, and pole-zero cancellation had to be evaluated from a live display (cf. Figures 8.1 and 8.2 on page 178), which only allowed a qualitative judgment.

For this reason a PCB was developed, which can connect a test signal to one of the 64 channels of a CB-SADC by the means of digitally controllable switches (de-multiplexers). The board is shown in Figure 6.13. On the left-hand side, the assembly consists of two board layers, which are connected mechanically with spacer sleeves and screws, and electrically with SAMTEC FJH flat flexible cables. The main board, together with the four elevated smaller boards, forms connectors that are modeled after the plug of the MRJ21 cable, which is used to connect the signals from the Crystal Barrel Calorimeter (more exactly, from the BuffTi) to the Analog Input Card in the CB-SADC. This allows to plug the Testboard directly into the assembled CB-SADC, saving the time for cabling (cf. Figure A.9 on page 227).

A recorded CsI(Tl) pulse from the experiment is fed to the Testboard from an arbitrary waveform generator and converted into a differential signal. The conversion takes place on a separate PCB which is not shown here. It is plugged in on the top right of the Testboard, leaving room for future improvements (for example, the integration of the signal generation, to be independent from an arbitrary waveform generator). The signal is then distributed through a number of four-channel demultiplexers (ANALOG DEVICES ADG1409) that are controlled by $I^2C$ I/O registers. An ARDUINO board is used as an $I^2C$ master and connects to the Testboard, and through the Testboard also directly to the CB-SADC.

The foreseen test procedure and examples of the preliminary analysis will be presented in the next section.

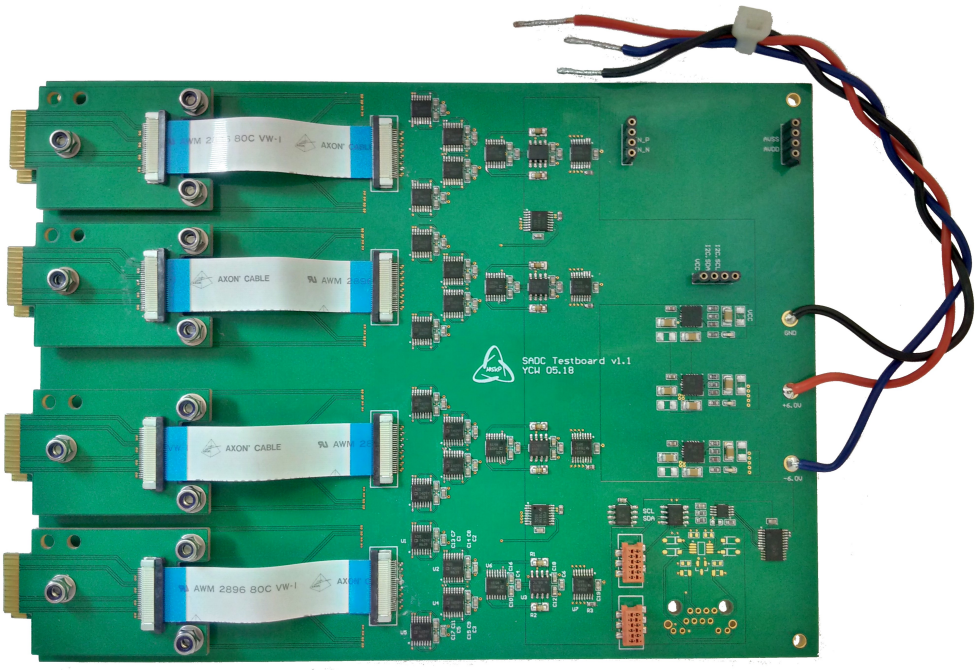
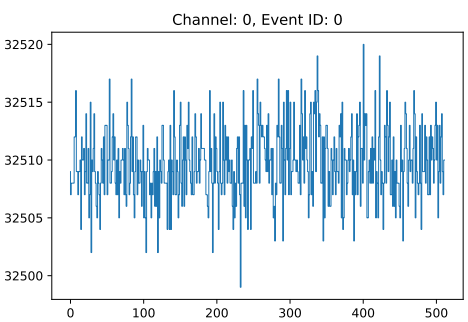



Figure 6.13.: CB-SADC Testboard developed within [140]. The left side shows the MRJ21-like connectors, followed by flat-flex cables and the demultiplexer circuits. The lower right region shows $I^2C$ connectors and components, the middle right linear regulators for power supply of the circuits. On the top right a separate board can be plugged in to supply a differential test signal.

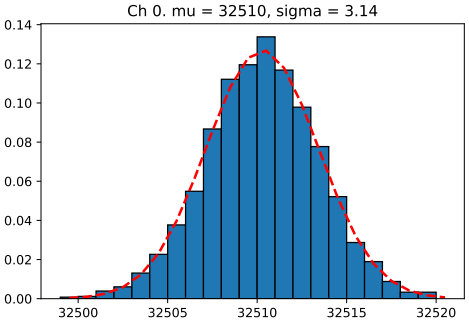
### 6.2.1. Test/Analysis

The whole test procedure can be controlled through a PYTHON script, and the necessary analyses are performed with the `matplotlib` and `numpy` libraries. The script is able to configure the whole setup: the CB-SADC firmware through UDP network packets, and the sensor readout, pole-zero cancellation parameters, and the baseline-shifting parameters through $I^2C$ with the ARDUINO $I^2C$ master. Raw data is received from the CB-SADC through network packets (similar to the readout in the experiment setup), and after analysis, a test report is generated automatically. To provide an example, plots from two of the analyses are presented in the following: Figure 6.14 shows a part of the outcome of the automated analysis of the baseline shifting circuit (cf. section 6.1.2), also yielding information about the noise of the individual channels. For every channel, the width and position of the baseline is analyzed, and the dependence of the position from the DAC setting is visualized. Figure 6.15 shows parts of the analysis to test the pole-zero cancellation circuit (cf. section 6.1.3). A test signal is recorded with different settings of the rheostat, resulting in different levels of under- or overcompensation, which may be quantified by the ratio of pulse amplitude to the baseline level in a specified interval after the pulse. Figure 6.15c shows that indeed this quantifier is suitable to test if the circuit it working, as the ratios are clearly distinct.

A first test with a prototype CB-SADC and Analog Input Card did indeed identify problems that were undetected in manual testing. More details are presented in [140].

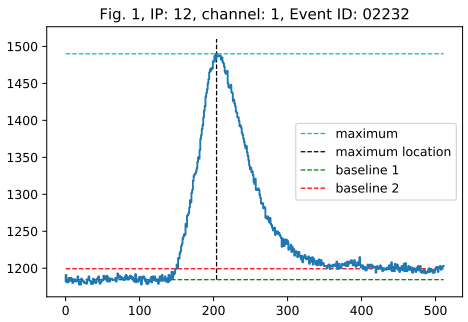



(a) Baseline of one analyzed channel, the deviation from a flat line is the noise.

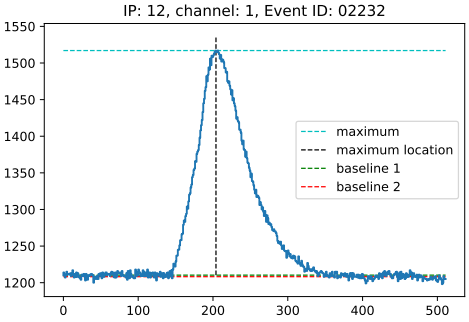



(b) Projection of the data points with a Gaussian fit. $\boldsymbol{\mu = 32510}$ is the position of the baseline, $\boldsymbol{\sigma = 3.14}$ is the width (the RMS noise).

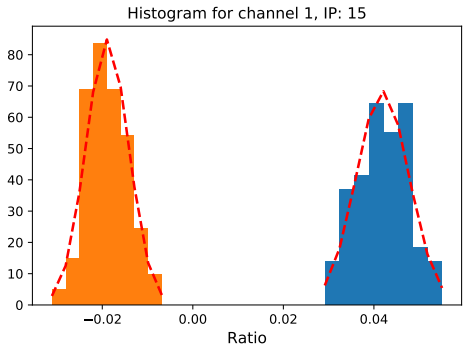
Figure 6.14.: Analysis of the baseline with PYTHON scripts written in the scope of [140]. The data was recorded in a laboratory test setup.



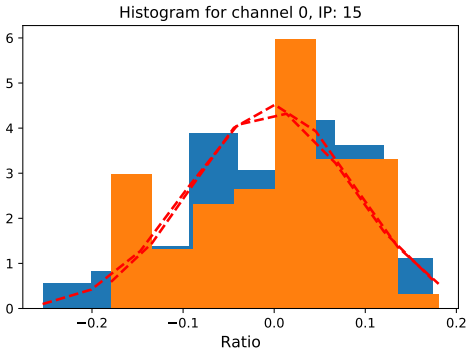

(a) Rheostat setting 0.




(b) Rheostat setting 255.




(c) Ratio of a regular channel, both cases are distinct.




(d) Ratio of a channel with defective potentiometer or not signal.

Figure 6.15.: Analysis of the pole-zero cancellation circuit with `python` scripts written in the scope of [140]. Two waveforms (from a *light pulser* run) are shown with extreme rheostat settings, the difference in the tail is clearly visible. The histograms show the ratio between maximum of the pulse and amplitude of the tail (last 100 samples), for both settings in blue and orange.

## 6.3. Power Supply

This section describes the development of the modular power supply for the CB-SADC. Like the PANDA-SADC, the board is not powered by a single voltage rail, but four different voltages. It had been decided not to integrate a power supply on the CB-SADC's PCB. Comparable to the choice made in the case of the Analog Input Card, this allowed parallel testing and development. Furthermore, it allows for the exchange of the power supply in case of failure, without exchanging the whole SADC.

Section 6.3.1 will summarize the essential requirements. In section 6.3.2 the used *switching* topology will be introduced. Sections 6.3.3 and 6.3.4 will show the development of and insights from the prototypes. The final version will then be presented in section 6.3.5.

### 6.3.1. Requirements

The information stated in section 5.6 (more specifically from Figure 5.17) directly leads to the requirements for the four voltage rails that have to be supplied to the CB-SADC.

- 1.0 V with at least 2.4 A (as determined experimentally). With a more complex firmware the current demand might increase, therefore a safety margin of 1.6 A will be added.
- 2.5 V with at least 8 A as measured or extracted from datasheets.
- 3.3 V with roughly 1 A required by the CB-SADC and additionally 1 A required by the Analog Input Cards.
- −3.3 V with 1 A required by the Analog Input Cards.

Apart from those minimum currents, the voltage should be relatively *clean* by means of residual ripple or any kind of noise. But, since all critical voltage rails, like for example the supply of the analog domain of the LTM9009-14 ADC chips, are further regulated by especially low-noise linear regulators, this is not a very strict demand.

Lastly, the power supply should integrate well into the slowcontrol system (see sections 2.11.2 and 5.7.5), which includes not only voltage monitoring, but also the possibility for remote power cycles to recover the CB-SADC from possible unresponsive conditions.

### 6.3.2. Topology

In section 5.6.1 the topology of the power delivery network of the CB-SADC has been discussed. Since the voltage differences between the four supply rails and the actual voltages required on the board are well below 1 V, and low noise was important because of the proximity to analog circuits, linear regulators were used. For the power supply, on the other hand, the four rails have to be transformed from the NIM-supplied rails of 6 V and 12 V. Assuming the 1.0 V rail could have a current demand of up to 4 A, linear regulation from 6 V would produce 20 W heat dissipation, which amounts roughly to the total power demand of the CB-SADC. Therefore, a switching topology has to be used, which can typically reach efficiencies around 90 %.

The most commonly used topology is the buck-regulator, which is explained in Figure 6.16.

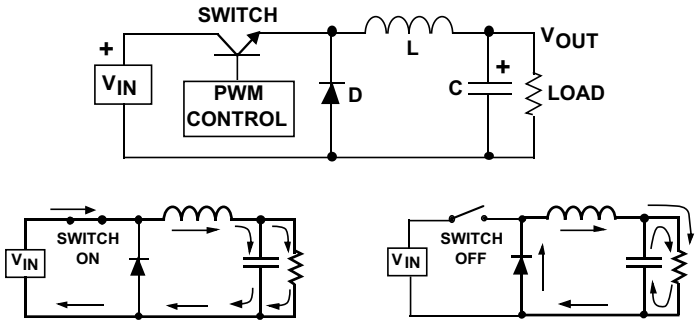



Figure 6.16.: A simple, non-isolating switching power supply with buck topology. When the switch is turned on, an increasing current flows through the coil which stores energy in its magnetic field. At the same time, the capacitor is charged and the output voltage will ramp up at the load. When the switch is turned off, the current through the coil will slowly decrease and the left side of the coil becomes negative, hence at some point the diode will start conducting, closing the circuit. During the off-cycle, the energy stored in the capacitor contributes to the current going into the load. Coil and capacitor form a second-order low pass that rectifies the square wave output from the switch: the duty cycle of the Pulse-Width Modulation (PWM) is proportional to $V_{OUT}$.

It is possible to build such switching power supplies with different levels of integration, depending on the demands of the application.

- One extreme is building the power supply completely from discrete parts, choosing switch, diode, inductor, and capacitor, and also creating the circuit which regulates the frequency and duty cycle of the Pulse-Width Modulation (PWM). This is tedious and even more challenging for more complex architectures, but it leaves room for lowest level customization, matching, and optimization, depending on the application.

- At the common level of integration, at least the PWM control circuit is supplied through an IC, and the user can set the output voltage in a similar fashion as for linear regulators (with a feedback resistor, or even digitally with buses like $I^2C$).

- At higher integration levels, the switch or even the diode and/or inductor are molded in a package with the IC to control PWM. This is the least demanding regarding PCB design and part selection, and can save valuable time.

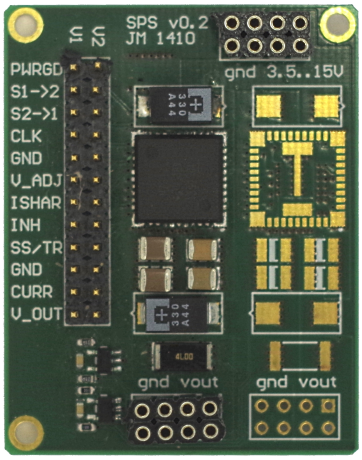
Depending on the level of integration, the design of a switching power supply can be a challenging task itself. For the design of the CB-SADC's power supply, the highest possible integration level was chosen, requiring only a few additional parts. The Texas Instruments Lmz31710 was chosen as a suitable solution for this, especially since the manufacturer also promotes its very low electromagnetic emission due to a molded shielding. In the next sections, the prototype developments based on this regulator will be discussed.

### 6.3.3. One-Channel Prototype

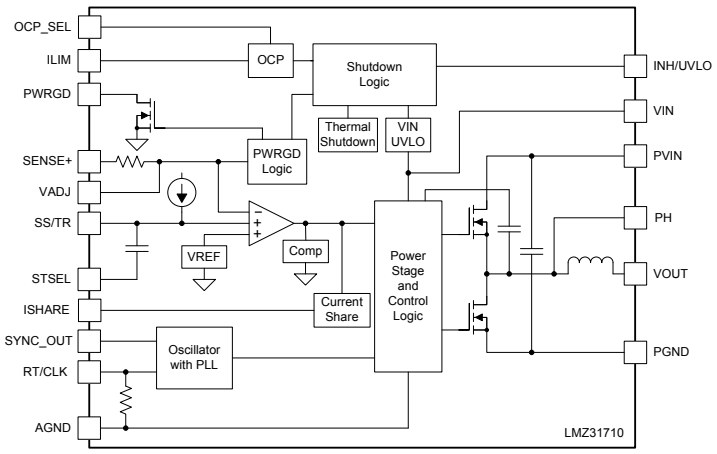
To investigate the operation and performance of the Texas Instruments Lmz31710, and to estimate the *size per channel* for a power supply based on this architecture, a one-channel prototype has been developed, which can be seen in Figure 6.17a (in fact the board can host two channels, which was foreseen to optionally test the current sharing functionality and the synchronization of multiple regulators). Right next to it, the internal schematic of the Lmz31710 is shown. It can be seen that, compared to the simple buck-architecture as shown in Fig. 6.16, the diode is replaced by a second switch (the switches are depicted as field effect transistors (FET) instead of bipolar transistors in this case). Both connect to the also integrated inductor (coil) in front of the `VOUT` terminal. This architecture benefits from lower losses in the second switch compared to the diode, but complexity is added in the control logic, as it has to be ensured that both switches never conduct at the same time.

The left-hand side of both figures shows the variety of control possibilities for this power supply IC, of which only two shall be explained. The `VADJ` pin allows to set the PWM duty cycle and hence the output voltage, by simply connecting a resistor with a specific value from this terminal to GND. In a similar manner the switching frequency, which is controlled by an internal oscillator, can be adjusted by connecting a resistor to the `CLK` pin. The oscillator can also be controlled by applying a clock signal to this pin directly, which allows to operate multiple power supplies synchronously (or shifted in phase). This will indeed be used in the four-channel prototype (see next section).

The next paragraph will show test measurements of the prototype, identifying the so-called residual ripple.




(a) Assembled Prototye with Lmz31710 in the center, input capacitor above, output capacitors and shunt resistor below; control inputs on the left, bottom left voltage/current measurement ICs.




(b) Internal schematic of the Lmz31710 switching regulator. The left side contains control logic (e.g. current limiting, voltage adjustment, current sharing, PWM frequency) which can be identified also on the connector of the PCB. The right side shows the two integrated high-power switches and the integrated inductance. [157]

Figure 6.17.: Prototype of the CB-SADC power supply with Lmz31710 switching regulator. One of two channels is assembled.

**Power Test and Residual Ripple** The prototype has been set up and configured with an input voltage of 6 V and an output voltage of 1 V. The output was connected to a load of 400 mΩ (high power resistor), resulting in a current of 2.5 A. During operation, the switching frequency and the output voltage were observed with an oscilloscope.

Figure 6.18 shows that the AC-coupled[1] output voltage (`v_out`) exhibits a residual ripple of almost 20 $mV_{PP}$ or 3.2 $mV_{RMS}$. A correlation to the 240 kHz switching (PWM) clock (`clk`) is clearly visible, with the larger high frequency spikes in `v_out`, most likely being correlated to the internal duty cycle of the switches. This ripple can be problematic for sensitive circuits like the ADCs and should be minimized if possible. A good result has been achieved by introducing an LC low-pass filter, consisting of a ferrite chip bead[2] and further capacitors. Some care was taken in the selection of the ferrite chip bead, since the LC circuit might show unwanted behavior like the so-called *tank oscillations* [168], leading to adverse effects. A suitable ferrite chip bead is the HI3312X101R-10 from Laird-Signal Integrity Products, which can sustain up to 10 A with a DC resistance of only 4 mΩ. Together with a bulk capacitance of 330 µF (plus the bulk capacitances on the CB-SADC), the output voltage (`v_out filter`) now has a residual ripple of less than 0.5 $mV_{RMS}$. As a result, this ferrite bead has been included into the design of the four-channel prototype.

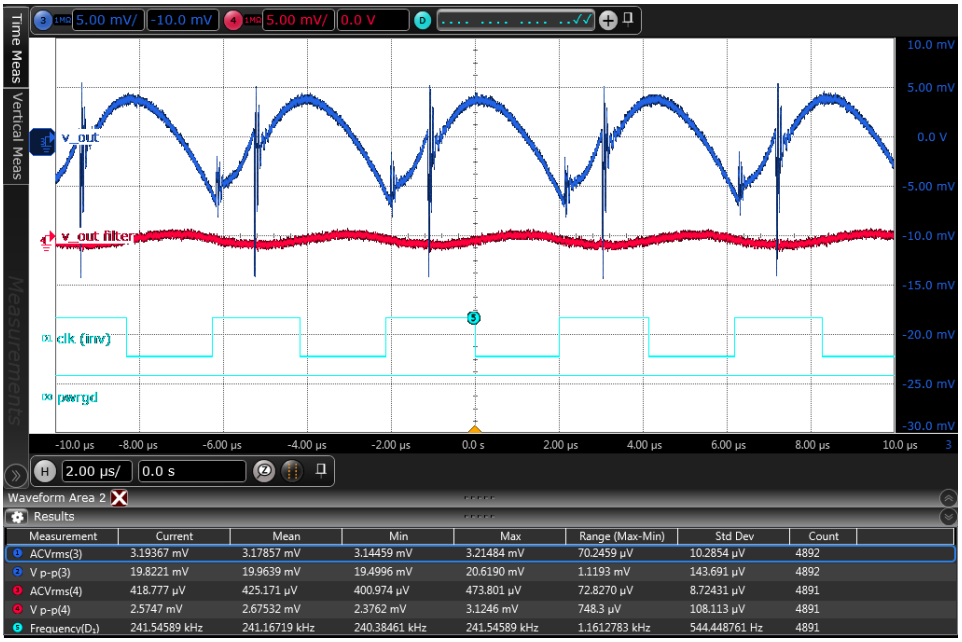



Figure 6.18.: Measurement from a test-setup of the one-channel prototype power supply, with a switching frequency of 240 kHz. The blue and red signal show the residual ripple of the output voltage, without and with additional ferrite chip bead, respectively, with 5 mV per division each. The time per division is 2 µs. The measurements displayed at the bottom quantify the improvement (e.g. `Mean` for `Vp-p(3)` and `Vp-p(4)`).

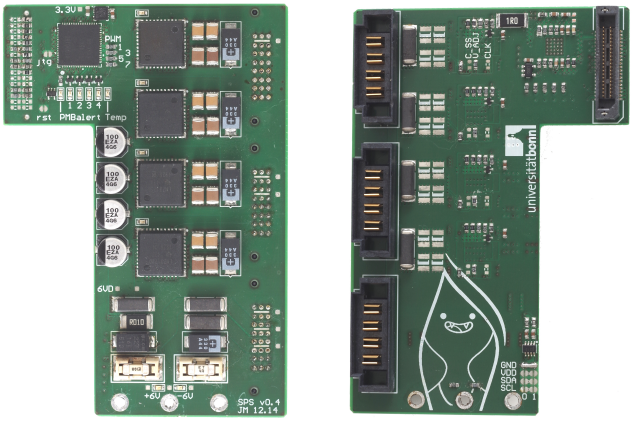
[1] AC-coupling removes the DC voltage and only shows remaining AC content.

[2] A ferrite chip bead is a surface mounted inductor with low Q factor, consisting of a printed coil surrounded by NiZn ferrite material. It is used especially for EMI (Electromagnetic Interference) filtering.

### 6.3.4. Four-Channel Prototype

After the successful test of the one-channel prototype, a four-channel design was prepared for the first CB-SADC prototype (`ADC-64K-CB v1.0`). This revision was not yet foreseen to provide ±3.3 V for the Analog Input Card, which is why it lacks the negative voltage regulator. The fourth channel was instead a second 2.5 V supply to split the analog and digital supply rail on the power supply level, allowing to test whether the two rails influence each other. The assembled prototype is shown and described in Fig. 6.19.

While the design of the four power supply channels could mostly be copied from the one-channel prototype, another challenge had to be faced: the one-channel prototype allows manual control and monitoring of the output voltage and current, which is not sufficient for the operation with the CB-SADC. Instead, it is necessary to control and monitor the power supply in a more automated way. It should be possible to remotely turn the CB-SADCs on/off to recover them from a failure. When the power is turned on, it can be important in which order the voltage rails are activated (especially for FPGAs), which requires the ability to control the single supplies in a sequential manner. Additionally, it is helpful to not only measure the output voltages and currents on the power supply, but other voltages on the CB-SADC as well. This all was achieved with a so-called Power Supply Sequencer and will be discussed in the next paragraph. The paragraphs that follow will then show a test under load, and the results of the optimization of the switching frequency.




(a) Top side. (b) Bottom side.

Figure 6.19.: Prototype of the four-channel power supply. On the top side, the Power Supply Sequencer can be seen on the top left, accompanied by a small regulator to supply an *always-on* control voltage. The sequencer's ADC inputs are connected to an array of resistor dividers on the left, on top of the power supply connector show on the left of Fig. 5.24 (page 96). Below the Sequencer, LEDs show the status of the power rails and the temperature. On the left side of the LMZ31710 regulators, four 100 µF aluminum polymer capacitors buffer the input voltage from the NIM power supply. Individual ceramic capacitors are put additionally right of every regulator due to their lower equivalent series resistance, together with a tantalum polymer bulk capacitance of 330 µF. At the bottom, ferrite beads, two fuses and the solder connection for the NIM supply are visible.

**Control and Monitor** The Texas Instruments Ucd90160, a *16-Rail Power Supply Sequencer and Monitor with ACPI Support* [169], was chosen to be the brain of the power supply. It is connected to the $I^2C$ slowcontrol bus of the CB-SADC, but uses an extension of the $I^2C$ protocol called Power Management Bus (PMBus). It extends the bus by two signals, `Control` (write) and `Alert` (read). The `Control` signal is usually used to turn a power supply on/off, while the `Alert` signal is connected to all (open-drain) alert pins on the whole bus (for example used to detect over-temperature, over-current, or similar conditions.)

The capabilities of the Sequencer can be best explained by looking at the functional block diagram in Figure 6.20.

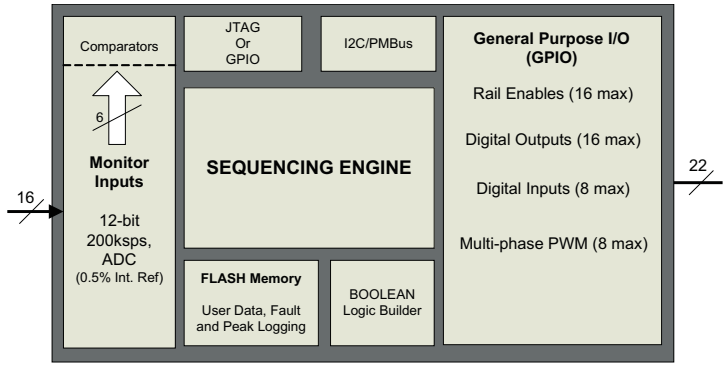



Figure 6.20.: Functional block diagram of the Ucd90160. [169] (mod.)

The left side represents the ability to measure up to 16 analog voltages, which was discussed already in section 5.7.5 and was indicated by the purple traces in Figure 5.24 on page 96. The right sides shows 22 GPIO connections, which can be used to control the power supply:

- **Rail Enable** is used to turn each Lmz31710 on/off at a specific time, this can be used for *sequencing* the rails and to turn on/off the whole CB-SADC remotely.
- **Digital Input/Output** are used to control LEDs (output), and to observe the *power good* signal from the Lmz31710 (input), indicating a valid state of the regulator.
- **Multi-phase PWM** refers to a centralized switching clock distribution, which is used to operate all four Lmz31710 at optimal and phase synchronous frequencies.

The manufacturer provides a software called Fusion Digital Power™ [170], which allows the complete configuration and monitoring of the Ucd90160 with the help of a USB-to-$I^2C$ adapter connected to a Windows workstation. The adapter is integrated into a development board [171] that can also be seen in Figure 6.21. All power supplies have been initially programmed and tested using this solution; but in the experiment, the monitoring has to be automated through the CBELSA/TAPS experiment's slowcontrol system. In [138] a first standalone slowcontrol solution was programmed by G. Urff, implementing for example all commands to access the 16 ADC measurements, to configure the multi-phase PWM, and to control the rails [172]. Within that thesis, as a first application of this standalone slowcontrol, a measurement was conducted to find the ideal switching clock frequencies for the four power rails, based on the achieved regulator efficiency. The integration of the sequencer's readout into the experiment's slowcontrol libraries and database is presently prepared, but still has to be tested and finalized.

**Load Test** To ensure a safe operation of the power supply, it was tested extensively under load before connecting it to the CB-SADC prototype. For the purpose of this test, a small adapter board was developed (see Figure A.7 on page 226), which exhibits the same connections to the power supply as the CB-SADC, and allows to connect loads and measurement equipment. The complete setup can be seen in Fig. 6.21. The rails were successfully tested with load currents of at least 5 A for all rails. During the test procedure, the temperature increase on the board was monitored with an integrated $I^2C$ temperature sensor and with an infrared camera. The temperature reached roughly 70 °C (on the desk, without any air flow), thanks to the high efficiency of the regulators. Temperatures will be considerably lower since the realistic load is only $\approx 50$ % of this test load, and that in the CB-SADC the power supply will be cooled by the NIM crate's airflow (see next paragraph). In addition to monitoring the temperature, the output voltages and residual ripple have been monitored carefully with an oscilloscope.

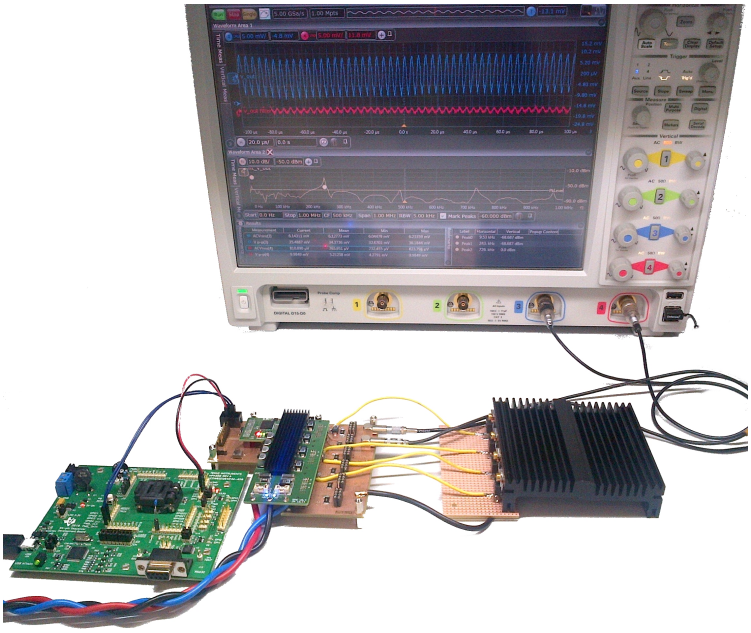


Figure 6.21.: Test of the four-channel prototype under load. On the left, the development board [171] used for $I^2C$ communication is connected to the test adapter with the power supply prototype plugged in. On the right, four high power resistors in `TO220` packages with a common large heatsink provide the test loads, connected by the yellow wires. The oscilloscope is connected to the power supply's output voltage with LEMO connectors to minimize electromagnetic interference. The display shows a measurement similar to Fig. 6.18.

After extensive testing of the sequencer's functionality and the behavior of the power supply under heavy load, it was concluded that the design of the four-channel power supply was working as expected. Also during later operation in conjunction with the CB-SADC (`ADC-64K-CB v1.0`), nothing gave rise to suspect otherwise. Despite the positive outcome, the design had to be iterated once more to include a negative power rail for the Analog Input Card. This final revision will be discussed in section 6.3.5.

**Switching Frequency Optimization** In [138], G. Urff has investigated the ideal switching frequencies for the different output voltages. According to the LMZ31710 datasheet, the recommended switching frequencies range from 200 kHz up to 1.2 MHz (see Fig. 6.22).

| SYNCHRONIZATION FREQUENCY (kHz) | PVIN = 12 V $V_{OUT}$ RANGE (V) | | PVIN = 5 V $V_{OUT}$ RANGE (V) | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| 200 | 0.6 | 1.3 | 0.6 | 1.5 |
| 300 | 0.8 | 2.0 | 0.6 | 4.3 |
| 400 | 1.1 | 2.5 | 0.6 | 4.3 |
| 500 | 1.4 | 3.4 | 0.6 | 4.3 |
| 600 | 1.6 | 5.0 | 0.7 | 4.3 |
| 700 | 1.9 | tbd | 0.8 | 4.3 |
| 800 | 2.1 | tbd | 0.9 | 4.3 |
| 900 | 2.4 | tbd | 1.0 | 4.3 |
| 1000 | 2.7 | tbd | 1.1 | 4.3 |
| 1100 | 2.9 | tbd | 1.3 | 4.3 |
| 1200 | 3.2 | tbd | 1.4 | 4.3 |

Figure 6.22.: Recommended switching frequencies for the LMZ31710. [157]

For the investigation, G.U. has connected the CB-SADC (`ADC-64K-CB v1.0`) to a 12 V power supply and measured the primary side current with the help of the $I^2C$ current sensor (INA219) on the four-channel power supply. By reprogramming the switching frequencies provided by the UCD90160 sequencer, he has optimized the efficiency rail by rail, and ended up with a total power consumption of 21.58 W instead of 21.95 W. Although the difference is only $\approx 400$ mW (less than 2 %), the optimized heat dissipation on the power supply should have an effect on the operating temperature. He has compared the heating up of the power supply with the recommended versus the optimized switching frequency (see Figure 6.23). Surprisingly this small improvement has lead to a 1.5 °C lower equilibrium temperature.

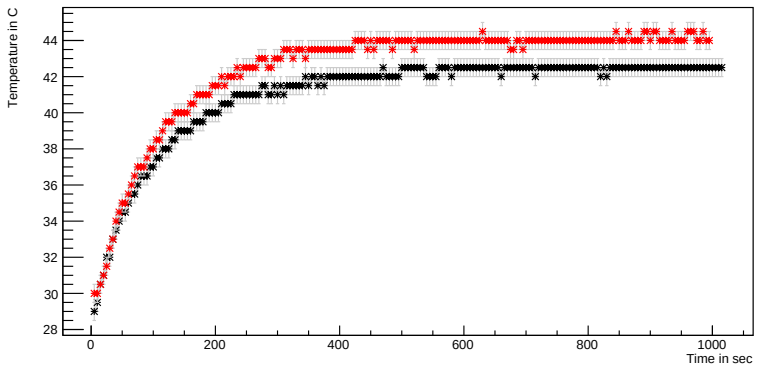


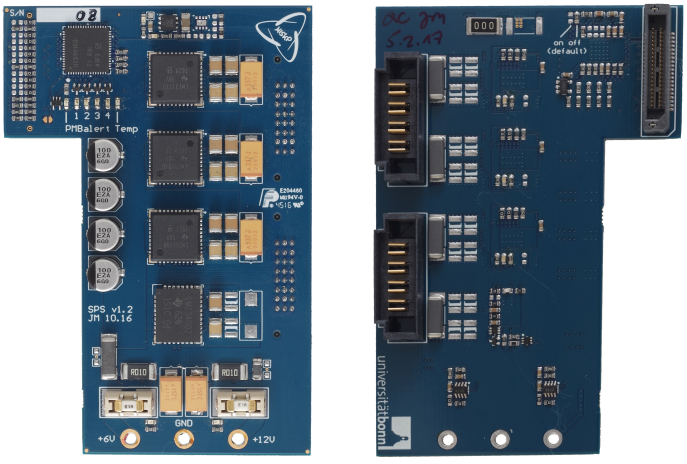

Figure 6.23.: Temperature increase on the CB-SADC power supply after power on. The black line represents a set of switching frequencies as recommended in the datasheet. The red line results from iterative efficiency optimization of the switching frequencies. The temperature difference is at least 1.5 °C in equilibrium. The CB-SADC with power supply was mounted in a NIM crate. [138]

It should be noted that this measurement will have to be repeated for the production revision due to two reasons: first, this was tested with the first four-channel prototype with dual 2.5 V rail and missing −3.3 V rail. Second, the primary voltage was 12 V for all regulators, as opposed to 6 V for the positive regulators in the production revision.

### 6.3.5. Four-Channel Final Design

With the development of the second prototype revision of the CB-SADC (`ADC-64K-CB v1.1`) it was decided to also introduce some changes in the power delivery network, also requiring changes in the power supply. Most importantly, the operating voltages for the Analog Input Card are now supplied. In the previous concept, two extension slots were placed on the `ADC-64K-CB v1.0`, which can be seen at the bottom of Figure 5.28b on page 99. Those slots were connected directly to the ±6 V NIM supply and were intended to linearly pre-regulate the voltage down to ±4.5 V, followed by further linear regulation on the Analog Input Card to ±3.5 V with currents of approximately 600 mA [139]. At the time it seemed the most flexible solution, but the additional regulator board and the linear voltage drops of 2.5 V were not ideal. As a result, it was decided to also use switching regulators for the pre-regulated voltages, which were decreased to ±3.3 V with further linear regulation to ±2.7 V on the Analog Input Card. Since the 3.3 V rail was already present, it was only necessary to add the negative supply. An IC matching the LMZ31710 series is the LMZ34002, which is also highly integrated, but uses an inverting buck-boost topology [173]. For a higher safety margin, the LMZ34002 is operated from the 12 V NIM supply, which leads to a maximum safe operating current of ≈ 1.8 A (efficiency ≈ 78 %), compared to ≈ 1.2 A at 6 V NIM supply [174]. The two separated 2.5 V rails were connected on the CB-SADC, as they could be supplied by a single regulator and the consolidation of analog and digital domain were shown not to be an issue. This allowed the integration of the new regulator without increasing the size of the power supply.

Further changes were the option to assemble a small switching regulator for the control voltage supply, in case the whole power supply is operated only from the 12 V NIM supply (which is now, in principle, possible). One of the three high-current connectors was omitted, as it was not required anymore. This also makes mixing of the incompatible revisions impossible. Figure 6.24 shows the final revision of the CB-SADC's power supply.




(a) Top side. (b) Bottom side.

Figure 6.24.: Final revision of the four-channel power supply. The lowest of the four regulators is the new LMZ34002 inverting regulator. At the top, the non-assembled switching regulator option for the control voltage can be seen.

## 6.4. CB-SADC Crate Controller

Each CB-SADC requires connections to the following systems for the successful integration into the CBELSA/TAPS experiment:

- **trigger/sync** with a connection to the four trigger signals (*Event, Sysreset, Busy, OK*) and the ten *Sync Bus* signals as described in section 2.9.4, such that data from the CB-SADC can be synchronously acquired by the experiment's DAQ;
- **slowcontrol** with an $I^2C$-over-CAT5 interface as described in section 2.11.2 (for the integration into the experiment's slowcontrol system), such that the CB-SADCs can be turned on/off remotely and their operating parameters can be monitored;
- **JTAG** such that the CB-SADCs can be programmed with new firmwares and that their behavior can be investigated with in-system debugging.

The so-called Crate Controller of the CB-SADC project consolidates multiple SADCs into one *virtual* endpoint for each of those system. Conceptually it would have been possible that each of the 24 SADCs embodies an individual *slowcontrol device* or *sync client*. The problem with this concept is the limitation of both systems: In case of the **slowcontrol system**, there are currently 30 unoccupied channels, theoretically enough to connect 24 CB-SADCs. It is not desirable to push this to the limit, as a greater number of channels was planned as a reserve for future additions to the system. In case of the **sync system** the situation is worse. In the current implementation, a complete VME board as shown in Fig. 2.19b (page 49) is required to integrate up to nine new devices into the system. This would have implied high costs.[3] Hence, grouping many CB-SADCs to resemble one *slowcontrol/sync/programming* endpoint is beneficial. Apart from those functions, the Crate Controller can also handle the **phase synchronization of the digitization clocks** by supplying the same clock to all CB-SADCs, such that their PLL (LMK04806) can lock its phase onto it (see section 5.7.3). The connection scheme is visualized in Figure 6.25. In section 6.4.1 the first concept of a Crate Controller will be shown. Section 6.4.2 shows the current solution, the *Backplane*, which was in operation during all beam times so far. Finally, section 6.4.3 will present ideas for a future/final revision of the Crate Controller, since some difficulties had to be faced with the current solution.

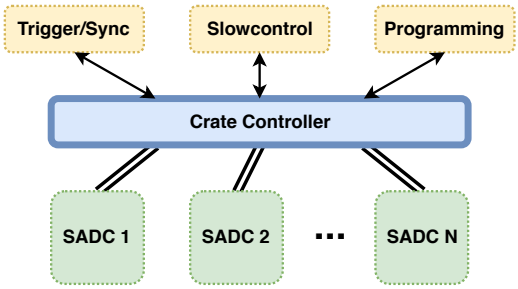
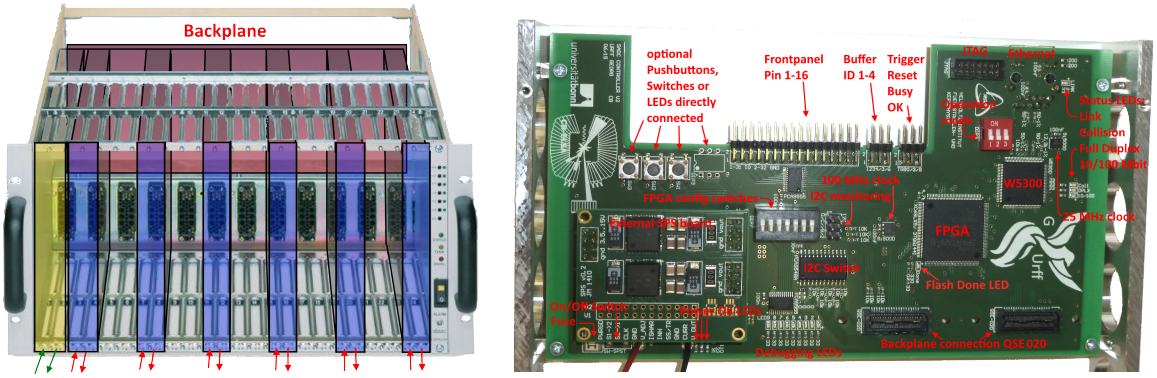



Figure 6.25.: Scheme of the Crate Controller, handling the distribution of trigger/sync, slowcontrol, and programming interface, to N SADCs.

[3] A system for more cost efficient integration of new sync clients is currently in development.

### 6.4.1. First Concept: The SADC Controller

A first concept has been developed within the master thesis of G. Urff [138]. At the time it was considered to distribute the 24 required CB-SADCs across four NIM crates, since this was the reserved space for the installation and it seemed to relieve the thermal management. This led to the decision to connect all CB-SADCs with a backplane, but put the necessary circuitry into a separate NIM module. The concept is visualized in Fig. 6.26a.




(a) NIM crate with SADC Controller (yellow), backplane (red) and SADCs (blue).

(b) Assembled PCB mounted inside a NIM cassette.

Figure 6.26.: Concept and prototype of the CB-SADC Controller. The controller was meant to be interfaced with a passive backplane in the back of the NIM crate, to which all CB-SADCs interface. The trigger/sync, slowcontrol, and JTAG signals are plugged in at the front of the SADC Controller. [138]

The prototype of the SADC Controller is shown in Figure 6.26b. It was developed with the XILINX SPARTAN-6 FPGA as a core component, which contains the $I^2C$ core and all logic circuits relevant for the trigger/sync distribution. It is complemented by the WIZNET W5300, a special IC to add a Transmission Control Protocol (TCP) stack, allowing an easy implementation of a network terminal (`Telnet`) on the FPGA. Two cable connectors (Fig. 6.26b bottom right) carry all signals that should be distributed to the six SADCs, and are meant to connect to the (passive) backplane.

A firmware for the on-board XILINX SPARTAN-6 was written and the Controller has been tested in the laboratory in conjunction with the CB-SADC (`ADC-64K-CB v1.0`). It was shown that the `Telnet` implementation was working well, and the $I^2C$ capabilities have been used to test the SADC's slowcontrol sensors and the power supply (cf. Fig. 6.23 on page 125).

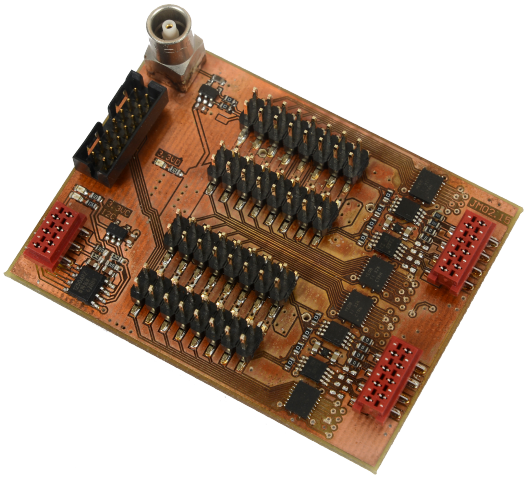
**Discarding the First Concept** was decided since the improved thermal management allows to fully equip two NIM crates, and freeing up the reserved rack space might ease future additions to the readout electronics. As this does not leave a free NIM slot for the SADC Controller, it requires all circuits to be moved to a different board that interconnects all CB-SADCs. In the next section, a first solution called the Backplane will be presented.

### 6.4.2. Second Concept: The Backplane

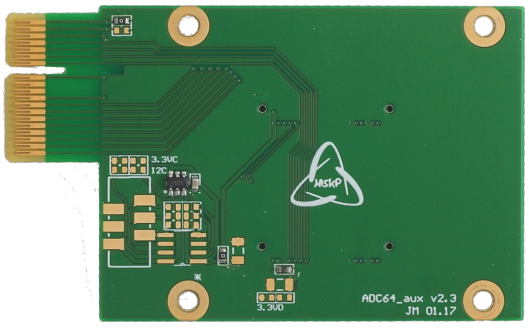
This section will present the development of the Backplane, which is suitable to connect up to four CB-SADCs. The prototypes have been used to investigate the distribution of the trigger/sync signals through level converters and FPGA logic, and furthermore to test the daisy-chain of four CB-SADCs on the JTAG bus. During various test beam times, and also in the production beam times in 2018 and 2019, two prototypes of revision 1.1 have been used successfully during data taking, each connecting to three CB-SADCs.

Before the design of the Backplane prototypes is described, the handover point between Backplane and CB-SADC, the Extension Board, will be introduced.

**The Extension Board** is mounted with two connectors on a slot towards the back of the CB-SADC (cf. Figure 5.3 on page 71). It connects to the $I^2C$ and JTAG bus, to 16 single-ended TTL signals (GPIO), and to one high speed serial interface of each FPGA. Additionally it is possible to connect a clock signal, which provides the possibility to lock the PLL (LMK04806) of all CB-SADCs to one externally provided clock. Figure 6.27a shows a prototype of the Extension Board for extensive debugging of those connections. It was the first project milled on a prototyping machine in the laboratory with a structure size of only 100 µm. Since no problems surfaced in first tests, parts of the circuitry where re-used for the Backplane, and a new revision of the Extension Card with direct connection of all signals to a `PCI`-like connector interfacing with the Backplane was developed (Fig. 6.27b).



(a) First revision, milled on the LPKF PROTOMAT S63. Red connector on the left for $I^2C$, two other red connectors for trigger/sync with LVPECL converters. Clock input with LEMO connector, JTAG plug, GPIOs on pin-headers.




(b) Current revision with `PCI`-like connector on the left, which attaches to the backplane. Not assembled: Circuit for direct $I^2C$ connection. The new revision can be screwed down through mounting holes.

Figure 6.27.: Two revisions of the extension card, which is used to connect the CB-SADCs to the external signals of slowcontrol ($I^2C$), trigger/sync, JTAG, clock.

**Backplane Revision 1.0** The first revision of the Backplane is shown in Fig. 6.29a and 6.29b. The FPGA, ROM, and other supplementary circuits, have been copied from the SADC Controller. The **I²C** circuit comprises an I²C-over-CAT5 receiver, a switch, and an I/O register. Through four channels of the switch, the CB-SADCs can be addressed individually. The fifth channel connects to the I/O register, through which the power supply's PMBus signals (`Control`/`Alert`) are written/read. A clock distribution based on the MICREL SY89113U clock muxer was integrated to test a phase synchronization of the CB-SADCs.

**Backplane Revision 1.1** For this improved revision, mounting holes were added to the board, such that the Backplane can be mechanically fixed to the back of the NIM crate. This was necessary since the retention force of the connectors was not enough to reliably hold the Backplane in place. In addition, the I²C capabilities were enhanced by a second I/O register to control the clock muxer, and a few bugs in the layout have been fixed. A computer drawing of the assembled PCB can be seen in Fig. 6.29c. Two boards have been assembled and were successfully used for test and production beam times in the CBELSA/TAPS experiment, with each board being connected to three CB-SADCs.

**JTAG Bus** The JTAG bus is used for in-system debugging and programming. It consists of four signals: `TDI` (data in), `TDO` (data out), `TCK` (clock), `TMS` (mode select). Typically all devices on the bus are in a daisy chain, and the `TDO` signal of one device is routed to the `TDI` signal of the next device, and `TCK` and `TMS` are routed along them. This concept does not apply for the CB-SADCs, where the `TDI`/`TDO` signals ofr the FPGAs are daisy chained, but `TCK` and `TMS` are distributed in a star distribution.

When multiple CB-SADCs are connected to the same JTAG bus on the backplane, the propagation delay of the `TDI`/`TDO` loop will deteriorate the phase relation to `TCK`/`TMS`. Since the loop through one CB-SADC is $\approx 400\,\text{mm}$ when entering from the Extension Board[4], and the propagation speed on a PCB is roughly $0.5 \cdot c_0$, the drift is about 2.7 ns. At a typical clock of 15 MHz, this is a phase shift of about 14.4°. To circumvent this, two different series of buffers have been used (SN74LVC and SN74AUP, see Fig. 6.28). At similar operating conditions, SN74LVC has a propagation delay of 1.5 ns to 4.1 ns, and SN74AUP 2.5 ns to 6.5 ns. Although this does not exactly compensate the drift, it is evident that it will relieve the problem.

At the moment it is tested only for up to four CB-SADCs. Two further solutions are possible, should problems occur despite the buffering. First, the JTAG clock speed can be lowered, and secondly, instead of a daisy chain, a JTAG switch/multiplexer can be devised.

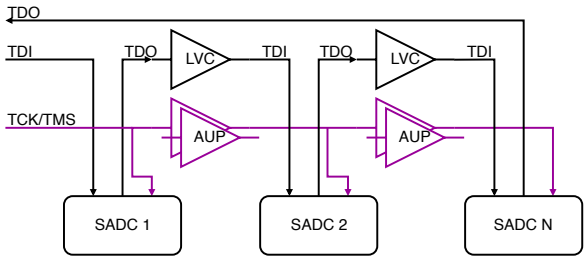
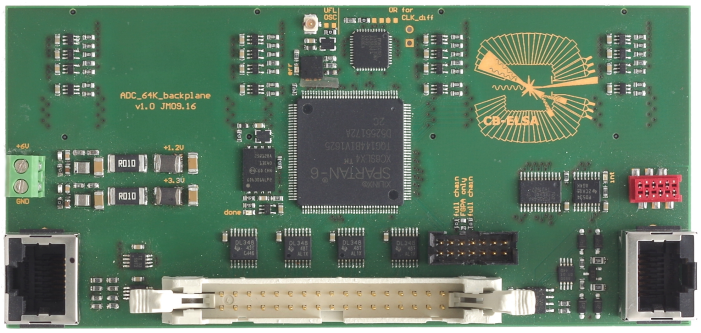
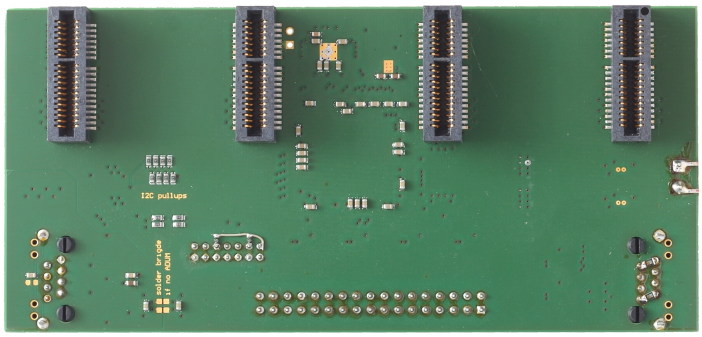



Figure 6.28.: Scheme of the buffers used for the JTAG signals.

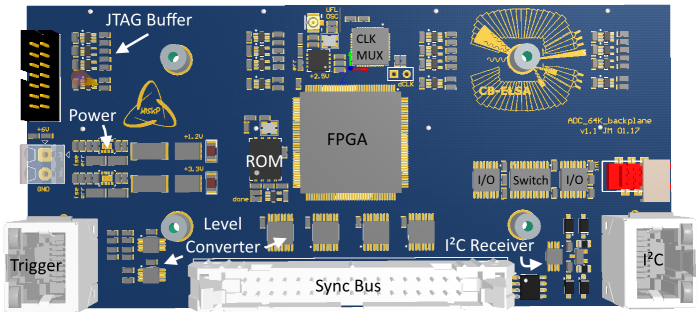
[4]When using the Front Connector, it is even roughly twice that length.



(a) Revision `1.0`, top side with most components.



(b) Revision `1.0`, bottom side with connectors to the CB-SADC extension cards.




(c) Revision `1.1`. Compared to the first prototype, the JTAG connector was moved, and mounting sockets for mechanical fixation were added. Since this prototype is prsently fixed in the crate, a Computer-aided Design drawing is shown instead of a photograph.

Figure 6.29.: Prototype revision `1.0` and `1.1` of the backplane. The SPARTAN-6 in the middle is surrounded by clock muxer/distribution (top), flash ROM (left), level translators (bottom) and JTAG port (bottom right/top left). The CAT5 connectors are used for the trigger system (left) and the slowcontrol system (right). The sync bus is received on the flat ribbon connector at the bottom.

### 6.4.3. Future Concept: The CB-SADC Crate Controller

During the development and tests of the Backplane prototypes, several problems and drawbacks could be identified. Some of them were, in principle, evident already in the conceptional phase of the Backplane, but others have emerged only during/after the installation in the hall and the continued operation.

- The most critical problem has been the installation of the prototypes at the back of the CB-SADCs, situated on top of the NIM power supply in the back of the crate. In this position everything is extremely **difficult to service**, it is virtually impossible to access the Backplane and its connections without removing all network cables from the CB-SADCs and un-mounting other hardware which complicates the access (network switches unrelated to the CB-SADC readout are mounted in this position).

- A second important issue is the distribution of the **JTAG** signals. Although the method described on page 130 has worked well even with bus speeds in the MHz regime, it is questionable and difficult to evaluate whether it will still work with twelve instead of four CB-SADCs. It is likely that the daisy chain has to be split, which in the current implementation would require to use a higher amount of USB programming adapters for the JTAG bus. This solution seems to be non elegant at best.

- A third issue is, in general, the tedious **programming procedure** for the CB-SADCs and the Backplane itself. Currently, a network connection has to be established between a computer running Xilinx Vivado and a network-to-USB converter, to which a JTAG programmer is attached. From a separate computer the programming procedure can then be issued through the Xilinx Vivado GUI (or alternatively a Tool Command Language script). The work flow is far too complicated to allow a safe, easy, and reproducible execution without special knowledge. A far simpler process is necessary, which allows to control the (re-)programming procedure through the DAQ daemon. This requires an interface from a computer in the `control` network to the JTAG programmers. This problem was partly relieved by connecting the programmers to one of the VME CPU of the experiment, running the Xilinx Vivado GUI remotely. Clearly, it is desirable to have a dedicated system taking care of this.

- A last issue is the usability of the debugging interface, which will be shown in section 7.8.1. The problem is tightly related to the previously mentioned point, since like the programming procedure it has to rely on the JTAG connection and the workstation connected to it. It would be valuable to have an interface which can be accessed independently, for example through a web page or a Telnet connection.

To sum up, the situation does not favor a solution as previously planned (a *Backplane* stretching over twelve CB-SADCs in the back of the NIM crate). In the following paragraph, a revised concept will be presented that aims to solve all mentioned problems. The new *CB-SADC Crate Controller* is being developed and tested in the scope of the master thesis of J. Knaust [141].

**Concept** To resolve the first issue it was first planned to move the necessary board to the front face of the NIM crate, plugging it into extensions that reach out from the bottom of each CB-SADC. It was discussed in detail in section 5.7.4 and shown in Fig. 5.23 (page 94) that indeed the CB-SADC is well prepared in its final revision to allow access to all necessary signals not only from the back, but also from the front. The space in this region is limited, as it is constrained by the MRJ21 connectors from above. Unfortunately, a first mechanical model has shown that the tolerances for a fixed solution like this are crucial, and that destructive forces on the solder connections can not be excluded.

As a solution, the crate controller will now be mounted below the NIM crate, where 1U rack space is available. The CB-SADCs will then be connected to the crate controller with SAMTEC FEDP cables. This also allows for easier maintenance of the system, since every CB-SADC can be accessed without having to remove the crate controller.

It is planned to employ a XILINX ZYNQ SoC (System-on-Chip), which is a combination of an FPGA and an ARM processor. This SoC will make it considerably easier to implement network functionalities as it will be running a full LINUX distribution, the basis for resolving most of the mentioned issues. Current developments show that a JTAG multiplexer (TEXAS INSTRUMENTS SCANSTA112) can be used to connect transparently to all attached CB-SADCs. The SoC can then connect to the JTAG multiplexer and offer connection to it through Ethernet using the XILINX Virtual Cable (XVC) technology, in principle providing the same experience as the currently used network-to-USB converter with the USB JTAG programmers. Alternatively it can even take the role of a JTAG controller itself, such that it can independently issue the complete programming procedure after a bitfile is provided to the SoC (e.g. through a mapped network folder). In the same way a debugging and monitoring interface could easily be provided, maybe even with a web page being hosted by the SoC. For easier implementation of the SoC it is planned to employ a module as offered by the company ENCLUSTRA.

Figure 6.30 shows the pre-production version of the crate controller, with twelve SAMTEC FCDP-16 high-density connectors on the front, and the ENCLUSTRA ZYNQ module in the middle. The module was already tested together with an evaluation platform, and was successfully operated with a first prototype of the crate controller.

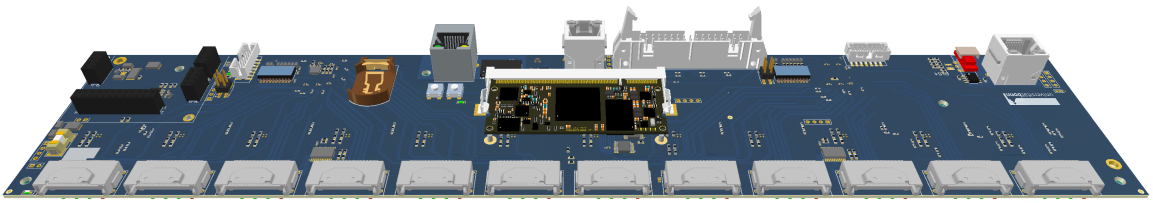


Figure 6.30.: Concept of the future CB-SADC Crate Controller. Twelve SAMTEC FCDP-16 connectors on the front carry the signals for trigger/sync, slowcontrol, and programming. Below the connectors, LEDs can show the status of each CB-SADC. The three network sockets are used for the trigger signal, the slowcontrol, and for network access of the ZYNQ module. The pinheader socket receives the sync bus. All connections will be made available on the front face.

## 6.5. Slowcontrol Extension

Before the CB-SADCs could be integrated into the Crystal Barrel slowcontrol that is installed in the experimental hall, it had to be tested in the laboratory. The $I^2C$ slowcontrol system, as developed and described in [88], is based on the transmission of unidirectional $I^2C$ signals on CAT5 cables with an increased driving strength, using the Texas Instruments Pca9600 (see section 2.11.2).

Figure 6.31 shows how the bidirectional `SCL` signal is split into two unidirectional signals (`Tx, Rx`) which are transmitted with a higher current capability over a CAT5 cable.

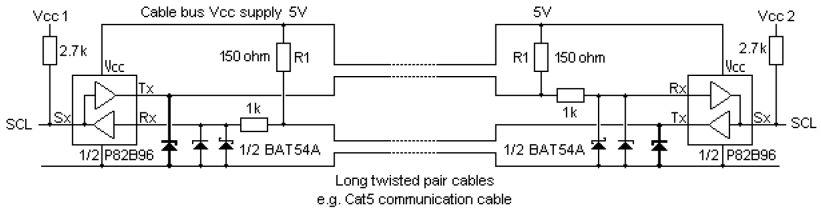
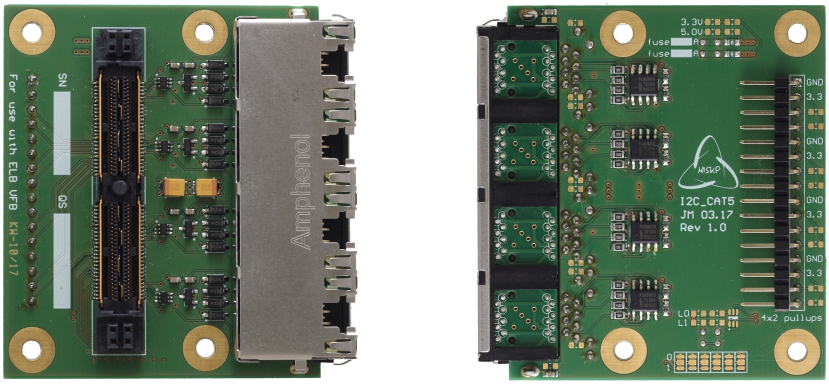



Figure 6.31.: Schematic of driving $I^2C$ over a CAT5 cable, using the Pca9600 (or P82B96). The voltages of the unidirectional signals `Tx, Rx` are limited by diodes, allowing ground level differences between the two end points. Only the `SCL` path is shown.

To use and test this system in conjunction with the CB-SADC, which has the according receiver circuit integrated on the Backplane, the circuit that is used in the experimental hall was replicated. A suitable PCB has been developed within this thesis. The assembled PCB can be plugged onto the extension slot of the Elb-Vme-Vfb module [93]. The assembled PCB is shown in Figure 6.32 and can be seen (plugged onto the Elb-Vme-Vfb module) in operation with the CB-SADC rack in Fig. 9.2. It allows to connect up to four $I^2C$ buses with CAT5 connectors and furthermore offers eight direct connections to FPGA pins on the Elb-Vme-Vfb module that can be used for four more $I^2C$ buses.




(a) Bottom side, showing the extension connector, network connector, protection diodes, and LED drivers.

(b) Top side, showing the $I^2C$ drivers and the pin header for additional signals.

Figure 6.32.: Slowcontrol extension card for the Elb-Vme-Vfb module.

# 7. Firmware Development

In addition to the hardware itself, which was described in the previous chapters, the development of the firmware is the second important pillar of the CB-SADC project.

Firmware, in this context, refers to the description of the FPGA logic, which is discussed in the paragraph below. Without the firmware, the CB-SADC is just a *brick*: the network connection, the decoding, processing and analysis of the data, the configuration, ... everything is controlled by the firmware. Through its continuous development, the CB-SADC will be further improved, making it a more powerful digitizer with every new algorithm, and improving its reliability.

**Describing FPGA Firmware** is often compared to *programming* in the sense of *programming a microcontroller* or *programming software* (running in the scope of an operating system on a processor). Although there are certain similarities, one has to distinguish between *hardware description* and *programming*. One way to look at it is the comparison of the *compiled* code:

- For **hardware description** of an FPGA, a Bitstream/Bitfile is produced which dictates how the logic elements, lookup tables, memory blocks, input/output registers and so on, are interconnected and initialized (c.f. section 5.4).
- For **programming**, a code written in machine code (`assembly`) is produced, which can be interpreted by a processor in a procedural fashion (step by step).

The creation of an FPGA Bitstream involves several steps which are shown in Figure 7.1. All steps can be prepared and executed from within a very powerful IDE called Xilinx Vivado. The design flow, including remarks about the Very High Speed Integrated Circuit Hardware Description Language (VHDL) and the steps to verify a firmware are discussed in more detail in appendix B.

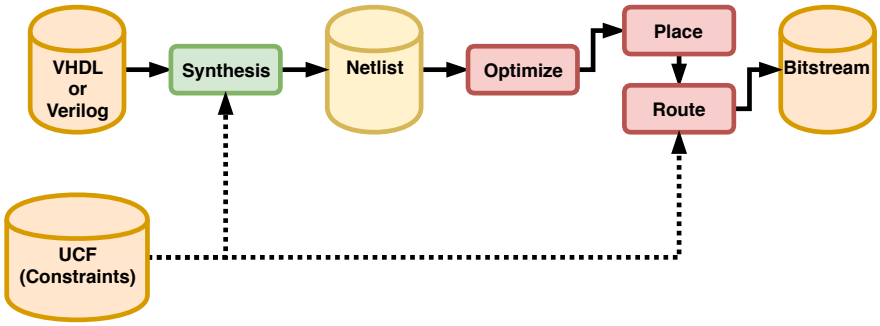



Figure 7.1.: Design flow from writing the VHDL code to the firmware (Bitstream). The *source code* written in VHDL is synthesized into a *netlist*, from which the *bitstream* is created, taking account of specific physical *constraints* of the FPGA.

## 7.1. Overview of the Firmware

This section will provide an overview of the structure of the firmware. The firmware is organized in modules with defined interfaces, and each module will be represented in its own section in this thesis. This is visualized in the module overview in Figure 7.2.

First, section 7.2 will describe how the boot-up and initialization routines ensure a reproducible correct operation of the whole firmware. Section 7.3 will show how the raw digitized data from the ADC ICs is processed by the FPGA to prepare it for the feature extraction and sample extraction. In section 7.4, the network implementation, which allows the CB-SADC to send and receive data via the User Data Protocol (UDP), will be covered. The development of feature extraction and sample extraction can be found in sections 7.5 and 7.6 respectively, followed by the explanation of the different trigger possibilities in section 7.7. Finally, section 7.8 will show firmware developments for two associated FPGA projects, the Backplane and the slowcontrol. Configuration options of the firmware, which are mentioned throughout those sections, are summarized in appendix C.

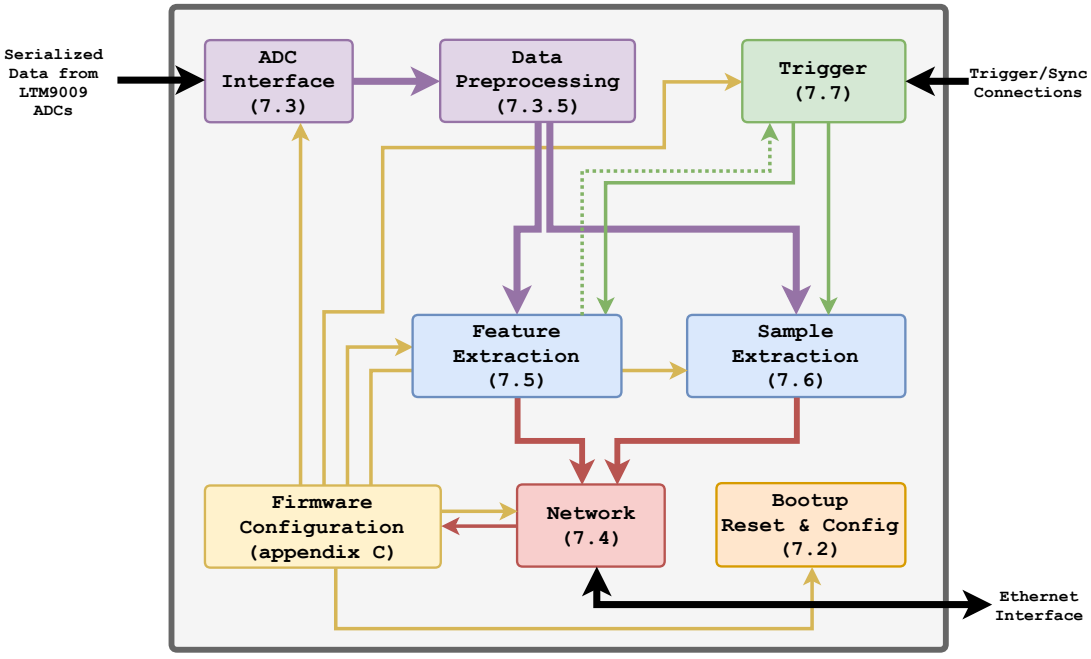



Figure 7.2.: Structure of the firmware modules. The according sections are denoted. The purple arrows represent the continuous streams of ADC data. The red arrows are AXI4-STREAMS (a proprietary data bus from XILINX). The trigger signals and the internal trigger generation (dashed) are green. All configuration signals are yellow. Note that the connections from the *Bootup Reset & Config* module are not shown as they affect all other modules. Black arrows represent external physical interfaces to the FPGA.

**GIT** The firmware is currently hosted and shared in a restricted access GITLAB environment (`https://agthoma.hiskp.uni-bonn.de/gitlab/CB/vhdl/CB-SADC64/`).

## 7.2. Boot-up: Reset and Initialization

This section will describe the parts of the firmware which are relevant for initialization when the CB-SADC is powered up and the firmware is loaded from the attached memory, or when it is reconfigured during operation. A reset state machine, described in the following section 7.2.1, guarantees a deterministic behavior of all modules. Some of the modules need time for initialization or configuration and may depend on attached hardware like the ADC chips (LTM9009-14) or the Phase-Locked Loop (PLL) (LMK04806). It has to be ensured that other modules, which may have a dependency on those, do not prematurely start their own initialization. Section 7.2.2 will briefly describe the configuration of the PLL (LMK04806) (see section 5.3) to provide proper clock frequencies to the FPGA and the ADC ICs. Similarly, section 7.2.3 shows the configuration of the ADC ICs themselves.

### 7.2.1. Reset State Machine

It has to be ensured that each module of the firmware starts its operation only after the necessary conditions are fulfilled. As long as this is not the case (e.g. because not all power rails are ramped up, the network is not initialized yet, other required routines have not been executed yet, ...) the module has to be held in a reset state. Two different reset implementation strategies will briefly be discussed before the reset sequence is presented.

**Asynchronous vs. Synchronous Reset** First of all, one has to distinguish between *synchronous* and *asynchronous* resets. While asynchronous resets are more common for FPGA designs that are evaluated for the design and production of ASICs, for pure FPGA design a synchronous reset is more controllable and preferable. An **asynchronous reset** is not related to a clock domain, usually because it stems from a signal that is generated outside of the FPGA. Since it is asynchronous, it does not pass any clocked registers (flip-flops), and can propagate through the FPGA fabric quite fast. Due to its nature, it may cause meta-stability in flip-flops, as the arrival of the reset signal relative to the clock edge cannot be predicted. In addition, for larger designs the asynchronous reset may arrive at different primitives in different clock cycles, leading to non-deterministic behavior. Even worse, this can change from one firmware to the next one due to different implementation results. For a **synchronous reset** design, all reset signals originate from, and are received by, clocked registers (flip-flops). This way the reset signal may not be as fast as the asynchronous reset, but it can be taken into account for the timing analysis, if desired. Due to the easier and more controllable handling, and no specific need for asynchronous reset signals, it was chosen to use synchronous reset for the CB-SADC firmware wherever possible.

**State Machine** A (finite) state machine consists of a finite number of states, one of which is active at a time. Transitions between the states are induced by external stimuli. This method has been used for a module that is responsible for the order of resetting the other modules of the CB-SADC's firmware after power-up. In each state it is defined which reset signals are active (releasing one reset signal in every state, until in the last state all are released). The stimulus for the next state is usually an indicator that the part of the firmware whose reset signal was released has finished its initialization.

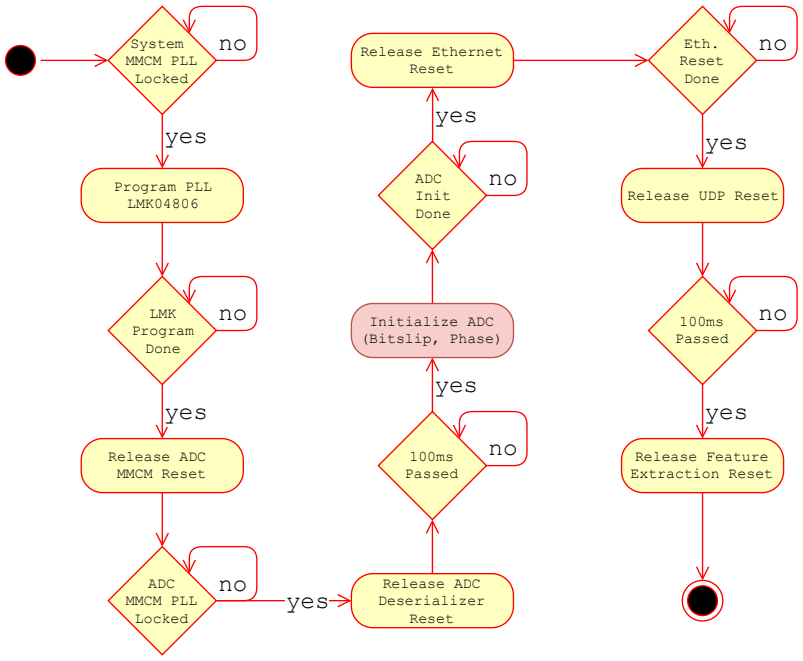



Figure 7.3.: Reset state machine of the CB-SADC firmware. The highlighted ADC initialization is handled in a dedicated state machine, shown in section 7.2.3.

- The initial stimulus is supplied by a special FPGA primitive, a mixed-mode clock manager (MMCM) that generates the clocks necessary for the startup of the firmware. It issues a `locked` signal after an internal PLL locks onto the *startup* clock supplied by an external MEMS oscillator. This indicates that the FPGA has entered an operational state. The system clock is necessary for all following states.

- In the next state, the LMK04806 `program` command is issued (see following section).

- After its programming is finished, the LMK04806 supplies the ADC and DSP clocks (cf. section 5.3), hence the *reset* signal of a second MMCM (*ADC Clock Manager*) responsible for their handling is released.

- After the MMCM's PLL is locked, the ADC de-serializer reset is released. Since is lacks an indication of being in a working state, the next state is entered only after 100 ms.

- Next the ADCs chips are configured through the µwire interface (cf. section 7.2.3), and the bitslip and phase calibration are executed (cf. section 7.3.2).

- After the ADC initialization is done, the `reset` signal of the Ethernet core is released. After the Ethernet core issues a `reset done` signal, the UDP core `reset` signal is released. The next state is entered after 100 ms.

- Finally the `reset` of the feature extraction is released and the state machine reaches its final state (from which it can only exit after a power cycle).

### 7.2.2. Phase Locked Loop Configuration

The Texas Instruments Lmk04806 is configured through a so-called µwire interface. The VHDL code for this was adapted from an existing firmware code for the 16-channel PANDA-SADC prototype, which has used a similar IC. However, the content of the programming registers has been carefully examined and some changes in firmware and hardware have been introduced. The content of the programming registers can be found in listing E.1 on page 245, and parts of the configuration are visualized in Fig. F.5 on page 262.

### 7.2.3. ADC Initialization and Configuration

The Linear Technologies Ltm9009-14 octal ADC is configured through an SPI interface, which was adapted from an existing firmware. In the original PANDA firmware, only a simple boot-up configuration was implemented, whereas for the CB-SADC firmware a more complex state machine was written. This state machine is necessary for two initialization algorithms of the Ltm9009-14, which will be presented in sections 7.3.2 (*Bitslip*) and 7.3.3 *Phase Calibration*. The initialization of the ADCs was introduced as part of the reset state machine in Figure 7.3. This section will now present the initialization state machine (see Figure 7.4) and explain what happens in each step.

`adc_reset` All internal registers of the Ltm9009-14 are reset and the IC is momentarily put in sleep mode with reduced power consumption. This ensures reproducible conditions after the firmware is reprogrammed (since the ADC ICs are not necessarily powered down in this process and will keep their previous configuration).

`adc_init` The Ltm9009-14 is initialized with the following setting:

- Transmission of the serialized data with 16 bit in two Low Voltage Differential Signaling (LVDS) lanes with 320 MHz DDR (see section 7.3.1).
- Driving strength of the LVDS lanes is set to 1.75 mA[1].
- Internal termination of the LVDS lanes is turned off[2].

`adc_pattern_on` A fixed 16-bit pattern of `0x3330` is programmed into the Ltm9009-14, this pattern is send to the FPGA for phase calibration instead of the actual digitized value.

`adc_calibrate` The phase calibration is started (see section 7.3.2).

`adc_bitslip` The bitslip calibration is started (see section 7.3.3).

`adc_pattern_off` After successful calibrations, the pattern is disabled, and actual digitized data is transmitted by the Ltm9009-14.

`adc_wake` The Ltm9009-14 is woken up (if it was sleeping) and is configured to use `XOR` randomization of the data (see section 7.3.4).

[1] This has been changed with regard to the original firmware. It is the lowest value possible and was verified to work without problems. In total, this setting reduces the CB-SADC's power consumption by 640 mW.

[2] This also was modified and saves power; it is necessary only for long transmission paths without proper termination at the end of the path.

After successful execution of the state machine, the `init done` signal is issued and evaluated by the previously introduced reset state machine (*"ADC Init done?"*). Apart from the automatic execution of the state machine at boot-up, it is possible to trigger it with a specific network packet, or through a Virtual Input/Output (VIO) when the firmware is debugged through the JTAG interface. This command is called `re-initialize`. Additionally, it is possible to execute each step separately, which can be of great help during debugging.

**Sleep Mode** It is possible to issue an `adc_sleep` command, which will reduce the power consumption of all eight LTM9009-14 ICs on the CB-SADC by 6.4 W, not including the power savings from the reduced losses in the linear and switching regulators. It is recommended to implement this during the operation at the CBELSA/TAPS experiment (in between the beam times, in case the CB-SADCs are not completely turned off), as reduced power will lead to reduced heat, which in return will improve the life expectation of all components on the CB-SADC.

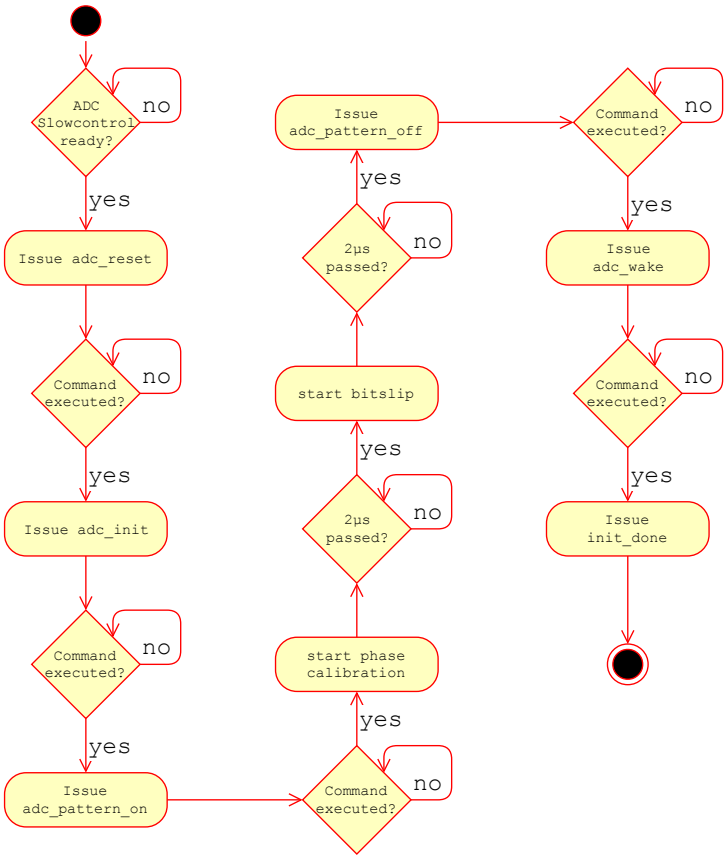



Figure 7.4.: Configuration state machine for the initialization of the LTM9009-14, with states as describe in the text. This state machine is executed as part of the previously described reset state machine.

## 7.3. ADC Data Interface

In this section, all modules directly or indirectly handling the digitized ADC data on the KINTEX-7 FPGA will be presented (except for the feature extraction and sample extraction, which can be found in sections 7.5 and 7.6). The VHDL module handling the interface between the LTM9009-14 ADCs and the FPGA will be shown in section 7.3.1. It was adapted from an existing VHDL project for the PANDA-SADC with minimal changes. The phase calibration and bitslip mechanism, which were written for the CB-SADC, will be explained in more detail in sections 7.3.2 and 7.3.3. Section 7.3.4 will explain why de-randomization is used for the transfer of the digitized data. Section 7.3.5 will introduce a module which carries out initial processing of the data, before it is fed to the feature extraction and sample extraction.

### 7.3.1. Serialized Data Links

The digitized data, which has a resolution of 14 bit at a sampling rate of 80 MS/s, cannot be transferred in a *parallel* fashion from all the LTM9009-14 ADCs to the KINTEX-7 FPGAs: for all 32 channels (per FPGA) this would require $32 \cdot 14 = 448$ LVDS signal pairs, far more than there are signal pins available on the KINTEX-7 FPGA. For this reason, the data is *serialized* by the LTM9009-14, meaning that all bits are transferred in series on one (or more) lanes at a higher frequency. This is described with the help of Figure 7.5. The LTM9009-14 is configured such that it transmits the serialized data (16 bit per sample, two zeros are added to the 14-bit values as *padding*) on two lanes (`OUT`). Therefore they have a serial data rate of

$$\mathrm{DR} = 80\,\mathrm{MHz} \cdot \frac{16\,\mathrm{bit}}{2} \tag{7.1}$$

$$= 640\,\mathrm{Mbit/s} \tag{7.2}$$

which results in a bit period of

$$t_{\mathrm{SER}} = \frac{1}{\mathrm{DR}} \tag{7.3}$$

$$= 1.5625\,\mathrm{ns} \tag{7.4}$$

To capture this data on the FPGA in a synchronous fashion, a serial clock (`DCO`) is transmitted by the LTM9009-14 along the two lanes. It is running only with 320 MHz, since data is captured on the rising and the falling edge of the clock signal (this is called Double Data Rate (DDR)). In the FPGA the data now needs to be *de-serialized*, meaning the data on the 2-bit lanes running at 640 Mbit/s needs to be captured and de-multiplexed to a 16-bit bus running at 80 Mbit/s. Two difficulties can emerge in such a process:

1. The clock transition of `DCO` needs to happen in the center of the data bit of the `OUT#` lanes. In section 7.3.2 it will be explained why, and how this is ensured.

2. The de-serializer does not know the sample boundaries in the lanes, meaning it cannot distinguish where the transmission of a new sample starts. Section 7.3.3 will show how this is resolved.

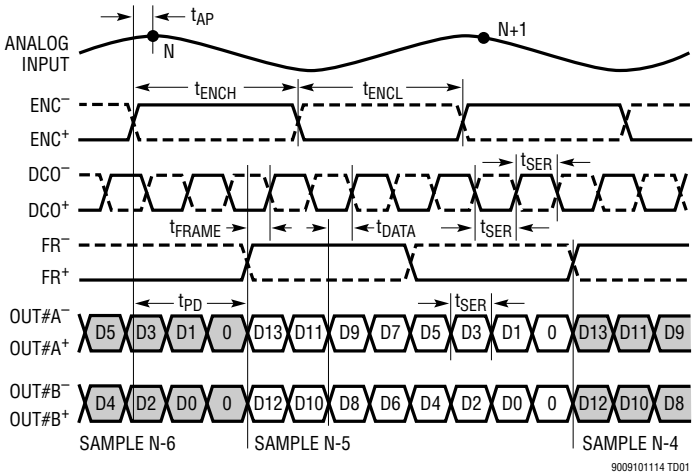



Figure 7.5.: Serialization of the Linear Technologies Ltm9009-14. The first row shows an analog waveform with two sampling points, `N` and `N+1`. The row below shows the (differential) sampling clock `ENC` (supplied by the PLL (Lmk04806)) with a frequency of 80 MHz. The last two rows show the 14-bit serialized data (`D13` down to `D0`) split to two lanes `OUT#A` and `OUT#B` (since the data is transmitted as 16 bit, it is padded with a zero for each lane). `DCO` is the clock which is transmitted alongside the serialized data towards the FPGA (note that it is DDR, `OUT#` is sampled on the falling and rising edge of `DCO`). `FR` is the so-called frame signal, which indicates the boundaries of a new sampling point with its rising edge. It can be used for a bitslip mechanism. [149]

### 7.3.2. Phase Calibration

In a synchronous data transmission it needs to be ensured that the clock crossing (clock `DCO` changes 0→1 or 1→0) happens in the middle of the data bit `OUT#A/B`. In other words, clock and data need to have a phase shift of 90°, ideally. This is indicated in Figure 7.5 by the interval labeled $t_{DATA}$. The reason for this is that the data, when it changes from one bit to the next, does not settle instantaneously. Several factors are responsible for smearing out the transitions. The most notable are the slew rate limitation of the driver circuit, and reflections caused by a non-ideal impedance matching. This leads to a *Sample Window* which is effectively smaller than the *Bit Period* $t_{SER}$. This is shown in Fig. 7.6.

It will be attempted in the following to approximate the size of the *Sample Window* by adding up the most prominent sources of uncertainty. This is described in some detail in [176], which even provides a spreadsheet to calculate the receiver skew margin (RSKM); however, it is not exactly applicable due to different implementation approaches for the CB-SADC.

- The source of `DCO` and `OUT#` is the Ltm9009-14 IC. In the datasheet [149], $t_{DATA}$ is specified as $0.35\,t_{SER}$ to $0.65\,t_{SER}$ ($0.5\,t_{SER}$ corresponds to the ideal 90°). This can be translated to a clock uncertainty of 0.468 75 ns. Note that this is specified over the whole temperature range of 0 °C to 70 °C, hence a smaller uncertainty can be expected under typical operating conditions.

- The ADC's datasheet [149] furthermore specifies a `DCO` cycle-to-cycle jitter of 60 ps.

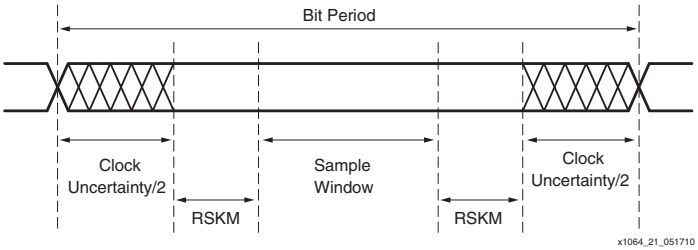



Figure 7.6.: Definition of the sample window. The bit period $t_{\mathrm{SER}} = 1.5625\,\mathrm{ns}$ is theoretically reduced by a clock uncertainty of up to $0.468\,75\,\mathrm{ns} + 60\,\mathrm{ps} \approx 0.53\,\mathrm{ns}$. The receiver skew margin consists of the FPGA's total system jitter and the sampling error, which add up to $71\,\mathrm{ps} + 0.4\,\mathrm{ns} \approx 0.47\,\mathrm{ns}$. If the clock crossing happens outside of the sample window of $t_{\mathrm{SW}} \approx 0.56\,\mathrm{ns}$, it might be that the data will be corrupted. [175]

- When the `DCO` clock is received and routed through the FPGA fabric, it is subjected to several effects contributing to an additional jitter, which makes up a part of the receiver skew margin. The *Timing Summary Report* in Xilinx Vivado states a total system jitter for the `DCO` clock network of 71 ps for one firmware that was investigated exemplarily. This number is calculated by Xilinx Vivado after the implementation (place/route) and therefore depends on the individual firmware.
- The sampling error at the receiver pins of the Kintex-7 FPGA is specified in the datasheet [159] as $t_{\mathrm{SAMP\_BUFIO}} = 0.4\,\mathrm{ns}$ and makes up most of the RSKM.

Those uncertainties/jitters sum up to roughly 1 ns (under the assumption that they are uncorrelated), leaving a *Sample Window* of only roughly one third of the bit period:

$$t_{\mathrm{SW}} = t_{\mathrm{SER}} - 0.53\,\mathrm{ns} - 0.47\,\mathrm{ns} \tag{7.5}$$

$$\approx 0.56\,\mathrm{ns}. \tag{7.6}$$

It has to be noted that this is only a rough approximation: on the one hand, the list probably does not include all error sources. On the other hand, some numbers are *worst case* scenarios, and it might also be that some are in fact not uncorrelated. Moreover, especially for the first point, it is not clear from the datasheet if the mentioned range refers to a static phase shift or possible short term fluctuation.

A source of a clearly *static* phase shift is the routing on the PCB. At a propagation speed of roughly $0.5 \cdot c_0$, a length mismatch between the clock and data lanes will cause a skew of $6.75\,\mathrm{ps\,mm^{-1}}$. As length mismatches of more than 15 mm are present, an additional static skew of more than 0.1 ns will be present, leading to phase shifts of more than

$$\frac{15\,\mathrm{mm} \cdot 6.75\,\mathrm{ps\,mm^{-1}}}{1.5625\,\mathrm{ns}} \cdot 360° \approx 20°. \tag{7.7}$$

The routing inside the FPGA fabric through different primitives (input buffers, clock buffers, delay primitives, ...) can additionally cause a substantial (static) skew.

Since the static skew was difficult to determine, it was decided to conceive an algorithm that restores the ideal phase between `DCO` and `OUT#`. It will be presented in the following paragraph.

**Phase Calibration Algorithm** In [175] and [176], methods are presented for a dynamic phase correction during de-serialization. For the CB-SADC firmware, the speed is low enough and the margin high enough to attempt a static calibration instead. In contrast to a dynamic calibration, this should save valuable resources in the FPGA. The static calibration is executed once at boot-up as part of the configuration state machine (Fig. 7.4).

All phase calibration methods are based on the so-called IDELAYE2 primitive, which is instantiated between the signal input pins and the de-serializer primitive. It allows to delay a signal in 31 steps (taps) of 78 ps, hence almost 2.5 ns in total. As this spans across more than a 1-bit period ($t_{\mathrm{SER}} = 1.5625\,\mathrm{ns}$), it is suitable to correct any phase mismatch.

As a first step in the calibration, the ADC (LTM9009-14) is programmed to send a fixed pattern on the serialized data lanes instead of the actual digitized signal (cf. section 7.2.3). Ideally, a pattern is chosen that has a bit transition with every clock cycle of the serialized lane, such that it has maximum sensitivity for a phase mismatch. Taking a closer look again at Figure 7.5, it is evident that this can be achieved with `D0, D1` = `1`, `D2, D3` = `0`, `D4, D5` = `1`, and so on. This results in a 14-bit pattern of `0b11001100110011`, or in hexadecimal representation, including the padding with 0 to get a 16-bit value: `0xCCCC`.

Now the state machine shown in Figure 7.7 can be examined. The pattern has been programmed as part of the ADC's initialization before. When the calibration is started, the IDELAYE2 primitive is reset to tap 0. Empirically observed, at this tap setting one is already within the *Sample Window*, but not necessarily at an ideal position. After some settling time, the de-serialized pattern is stored in an intermediate variable, with which it shall be compared in the next steps. Now the tap value is incremented so much that the sampling edge (falling/rising edge of the `DCO` clock) lies outside of the *Sample Window*, which will lead to unstable/*scrambled* data, and the comparison with the stored value will fail.

At this point, the ideal tap setting can be calculated. Empirically it was found out that the *Sample Window* does in fact stretch over up to 16 taps, which equals $t_{\mathrm{SW}} = 16{\cdot}78\,\mathrm{ps} \approx 1.25\,\mathrm{ns}$, more than twice compared to the previously approximated worst case calculation. Hence, when the unstable situation has been detected, the tap setting should be decreased by $\frac{16}{2}$ to be in the middle of the *Sample Window*, ideally. Depending on the tap setting at which this happens, it could happen that the ideal setting is actually before the initial setting of tap 0. In such a case the delay can also be increased further, such that the sampling takes place in the next bit. To achieve this, the tap setting is first increased by 4 taps to skip the unstable region (and crossing to the next bit), and then increased by $\frac{16}{2}$ taps. This situation has not been observed. Luckily this saves the additional work in correction of timestamps: if the data would be shifted by a complete bit period, this might have to be taken into account in the determination of timestamps in the feature extraction algorithms.

Due to its complexity, the algorithm was simulated extensively with a test bench (cf. appendix B.4) before it was implemented into the firmware. The simulation code can be found in Listing E.2 on page 247. The test bench simulates the serial link with the `OUT#` data lanes and their corresponding `DCO` clock. Random skews in the expected order of magnitude are added to the data lanes and the clock, and a cycle-to-cycle jitter as previously described is introduced. This made it possible to observe the phase calibration in the simulation, allowing to fine tune it before the implementation into the firmware. The behavior during the simulation can be observed in Figure 7.8 (together with the bitslip mechanism). The static phase calibration has proven to work reliably on the CB-SADC.

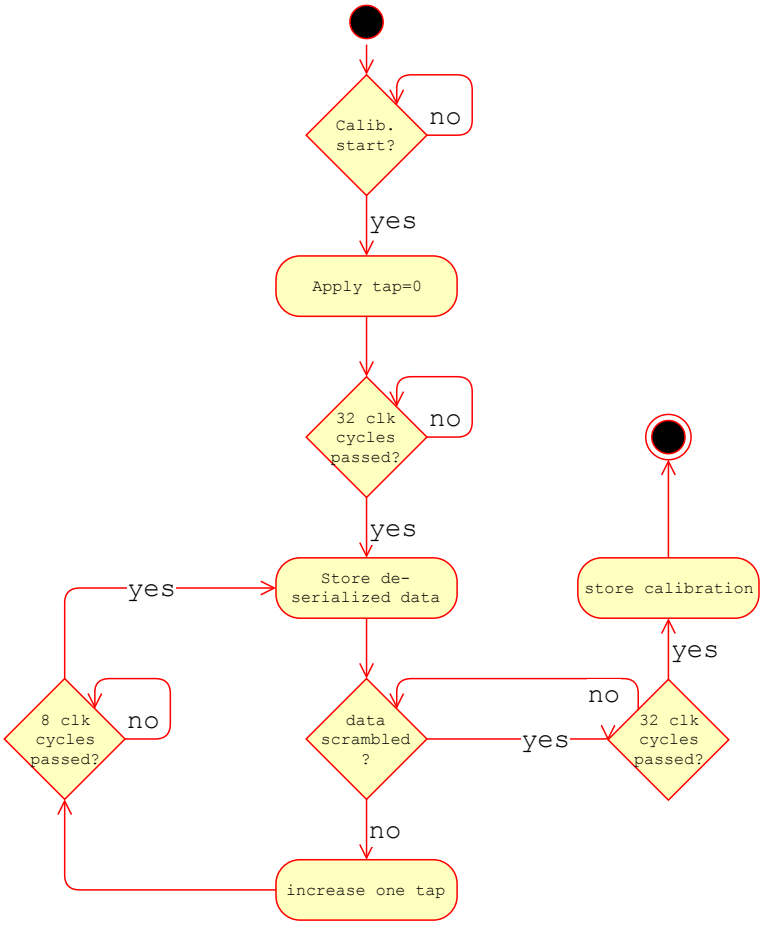



Figure 7.7.: Phase calibration state machine of the CB-SADC firmware.

### 7.3.3. Bitslip

The necessity and the principle of the bitslip operation will be explained with the help of Figure 7.8. When the ADC data is de-serialized, it should ideally appear as a 14-bit (16-bit) vector with the content `D13, D12, D11, ..., D2, D1, D0, 0, 0`. But for the de-serializer it is arbitrary where it starts to analyze the serial data stream and therefore where to make the boundary from one sample to the next. If there is, for example, an offset of 2 `DCO` cycles, the vector may be `D9, D8, D7, ..., D2, D1, D0, 0, 0, D13, D12, D11, D10` instead. The simplest solution to overcome this is to observe the so-called frame signal (see `FR+/-` in Figure 7.5). By definition, the frame signal is high when the MSB is transmitted on the `OUT#` lanes, and low when the LSB is transmitted. Therefore, if it is de-serialized along the data, its content will be `0xF0` (`0b11110000`) in the case of a correct alignment. A simple bitslip mechanism now has to check the content of the frame signal and issues a bitslip command if it is different from `0xF0`. Every time this bitslip command is issued, the de-serialized data will be shifted by one bit.

One frame signal is transmitted for every LTM9009-14 (once for eight channels). Theoretically, the phase calibration might lead to a situation where one or more of the lanes of those eight channels are skewed by one bit due to a high tap setting. For this reason, the frame signal based bitslip algorithm that was present in the PANDA laboratory firmware was omitted. The new algorithm does not rely on the frame signal, instead the pattern that was programmed for the phase calibration is used for a per-lane bitslip (artificially programming a *frame* signal on every lane). This can be observed together with the phase calibration in Figure 7.8. Since it is not possible to tell the sample boundaries in the case of a programmed pattern of `0xCCCC`, the pattern was in fact changed to `0x3330`. This still leaves a changing pattern with almost every bit boundary, but at the same time allows an unambiguous determination of the sample boundary for the bitslip algorithm.

### 7.3.4. De-Randomization

After the phase is calibrated and the sample boundaries are corrected by the bitslip mechanism, the digitized and de-serialized data is now stable and in the correct order. The content, however, still has to be de-scrambled in another manner.

The LTM9009-14 supports a special data-scrambling option called *randomization*. It will be shown with the help of an example how the randomization works and what its benefit is. With no analog signal present, the LTM9009-14 will measure a voltage close to 0 V, placing it exactly at mid-scale (since the LTM9009-14 can measure positive and negative signals). At exactly mid-scale, the 14-bit binary representation will cross from `0b01111111111111` to `0b10000000000000`. Statistically speaking, similar situations will occur with high probability in any situation where the baseline is (coincidentally) close to an ADC value of $2^N$. This is not a problem per se, but it can become a problem on an electrical level, at points where the data is transmitted in parallel and non-differential. This can

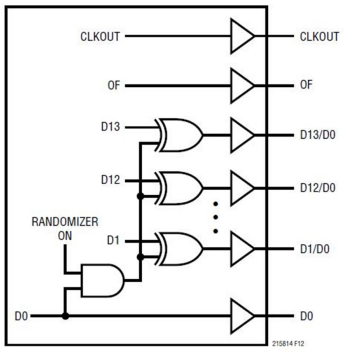



Figure 7.9.: Randomization circuit.
Source: `http://www.analog.com/en/technical-articles/digital-output-randomizer-for-high-speed-adcs.html`

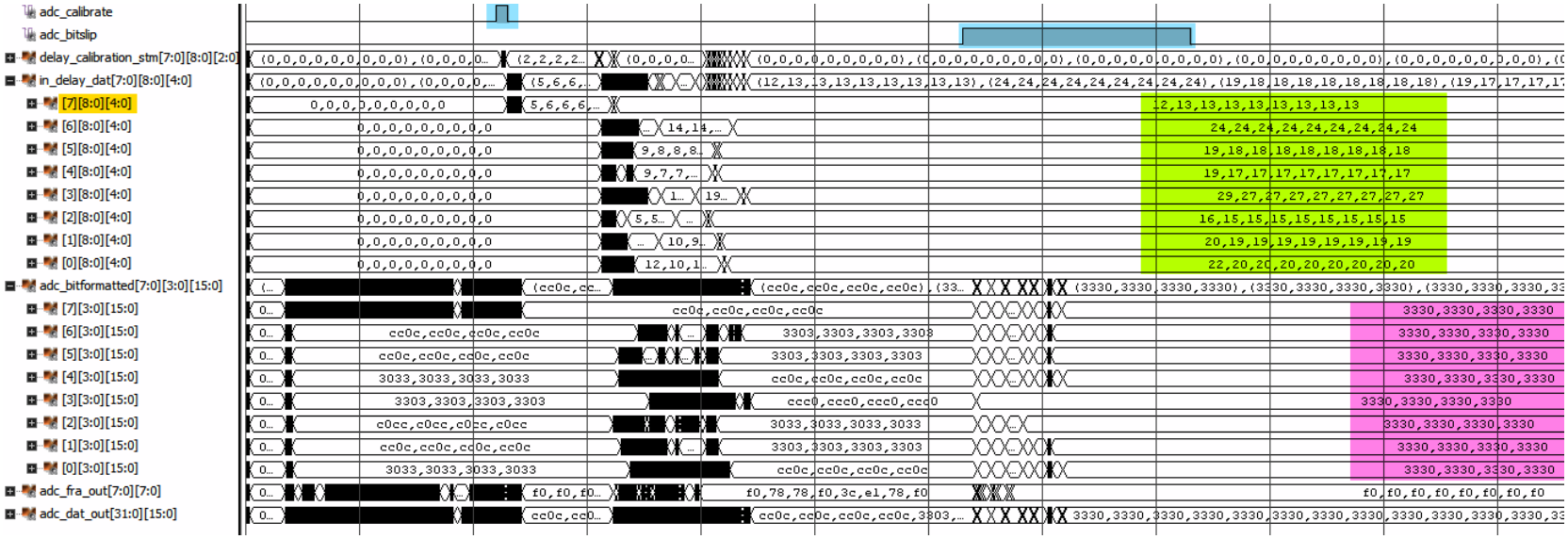



Figure 7.8.: Simulation of the phase calibration and bitslip mechanism with the XILINX VIVADO simulator. The left column shows the name of different signals or arrays of signals, e.g. `in_delay_dat[7][8:0][4:0]` (yellow) is an array of nine signals (`[8:0]` means the array goes from 8 down to 0) with each signal having 5 bit (`[4:0]`). Those nine signals represent the tap settings for four channels (two serial lines per channel) and the frame signal `FR`. The right column shows the value of the signals in decimal or hexadecimal representation, the crossings indicate a change of value (black blocks are accumulations of changes). The first two rows show the control signals that start the phase alignment (`adc_calibrate`) and the bitslip (`adc_bitslip`) in blue. The signals `in_delay_dat` represent the tap setting for all 32 channels of the FPGA, and it can be observed that they increment shortly after `adc_calibrate` is issued. After successful phase calibration, indeed the taps differ from channel to channel, and sometimes even within a channel from lane to lane, which can be seen by observing the numbers in the region marked green. Below, the de-serialized data lanes (`adc_bitformatted`) can be observed, each row showing the values of four channels. They are programmed with a pattern of `0x3330`, but in all cases the bitslip is wrong at the beginning and the pattern is shifted. At the end it can be seen in the magenta region that all lanes indeed show `0x3330`, indicating a successful bitslip: on all lanes the serialized data stream is de-serialized with correct sample boundaries, yielding the programmed and expected pattern.

be (presumably) inside of the LTM9009-14 (before the serialization), and inside of the FPGA (after de-serialization). What can happen in this case is that a considerable electromagnetic field is emitted from up to 13 signals (as the 14 bit are transmitted in parallel) that change at the same time (they are in this case the so-called *aggressor*), which in turn can induced a voltage on any neighboring signal (*victim*). It has been observed that in such a case, for a short time, the signal can be `0b11111111111111` or `0b00000000000000`, as the highest bit (*victim*) is affected by the *aggressor's* emanating electromagnetic field. Such a behavior is of course unwanted, and luckily the manufacturer offers a solution to overcome it:

**XOR Algorithm** The data is modified inside of the LTM9009-14 before serialization with a simple `XOR` algorithm. The uppermost 13 bit are each `XOR`'d with the last bit, which itself stays unmodified. This is shown on a level of logic elements in Figure 7.9. The effect of this transformation will be shown with the situation mentioned previously.

`0b01111111111111` → `0b10000000000001`
`0b10000000000000` → `0b10000000000000`

The data can be deciphered by applying the same operation once more. It is clear that unwanted transitions of this kind will still take place, the only difference is that now they occur more distributed at random values, and not so prominent anymore at the $2^N$ boundaries. On a timescale of a few samples, this might help to avoid interference with neighboring circuits.

### 7.3.5. Data Preprocessing

The ADC data is now de-serialized, phase-aligned, sample-boundary-aligned, and de-randomized. In the next step, it is possible to activate a **Decimation** and/or **FIR Filter** module before the data is handed over to the **Feature Extraction** and the **Sample Sender**.

#### Decimation

Decimation refers to a reduction of the sampling rate: each decimation will reduce it by a factor of two. The data from the LTM9009-14 has a sampling rate of 80 MS/s, hence a Nyquist bandwidth of up to 40 MHz. It was shown in Figure 2.13a (page 40) that the signal does not have frequency components distinguishable from noise above 1.5 MHz. Consequently, a three-fold decimation, resulting in a sampling rate of 10 MS/s and a Nyquist bandwidth of 5 MHz, should still be sufficient to reconstruct the signal; but to leave more headroom (for example for the sampling of the energy sum signal with the CB-SADC, which has higher frequency components), the decimation was always limited to 20 MS/s within the scope of this thesis. It was shown in section 1.4.3 that a decimation by a factor of $M = 2^2 = 4$ can increase the SNR by up to $M \cdot 3\,\mathrm{dB} = 12\,\mathrm{dB}$, but this is only the case if the noise in the higher frequency regimes is removed accordingly. Even without taking this benefit to the full extent, the decimation still has two other valuable advantages:

1. The network traffic caused by sending full waveform samples is reduced by the same factor of $M$, since for the same length of a sampling window, less data has to be sent.

2. The DSP clock network inside the FPGA can run with frequency lower by a factor of M, since all algorithms are executed at the sampling clock frequency. This leads to a faster implementation process, as the placement and routing is relieved, and longer paths are allowed inside the FPGA. It may be that at some point the firmware will be so complex that it might not even be possible to implement it without the decimation.

**Decimation Algorithm** The algorithm for decimation is trivial to implement in an FPGA. A number of $M$ samples is summed up continuously, much like the implementation of a simple moving average filter. The sample is not divided by $M$, instead the resolution is increased by $\log_2(M)$ bit. The value is now processed further in the FPGA with the new decimated clock frequency $f_{\mathrm{DSP}} = f_{\mathrm{sampling}}/M$. This algorithm is also known as *Integrate and Dump* or *Accumulate and Dump* filter.

### FIR Filter (Digital Signal Processing)

Instead or together with the decimation, a very powerful signal processing algorithm can be used: a so-called Finite Impulse Response (FIR) filter, allowing very efficient implementations of low-pass filters, as an example. The name of this kind of filter is derived from the fact that the impulse response (meaning the output signal, if the input signal is an infinitely short pulse) is finite in length, so its value is guaranteed to return to the baseline/zero after a defined time. This stands in contrast to Infinite Impulse Response filters, which due to their feedback might even exhibit uncontrollable oscillations, much like analog filters with feedback circuits do, if there is no sufficient phase reserve.

The filter is characterized by a number of coefficients $a(k)$. For a filter with $N$ coefficients, the filtered signal at sample number $n$, $y(n)$, is calculated from the original sample $x(n)$ as

$$y(n) = \sum_{k=0}^{N-1} a(k)x(n-k) \quad n = 0, 1, \ldots \tag{7.8}$$

Figure 7.10 shows a symbolic representation of this FIR filter. For the calculation of the current filter output $y(n)$, the previous $N-1$ data samples need to be retained in storage elements ($Z^{-1}$). This is typically achieved with a chain of flip-flops (a so-called *shift register*). Each of the data samples $x(n)$ to $x(n-(N-1))$ is now multiplied with a specific coefficient a(k), and all products are summed up to the filter output $y(n)$. Comparing this method with the previously described decimation yields that this is in fact also a FIR filter with $N = 4$ and all coefficients $a(k) = 1$, with the only difference that $y(n)$ is then processed at one quarter of the original sampling rate.

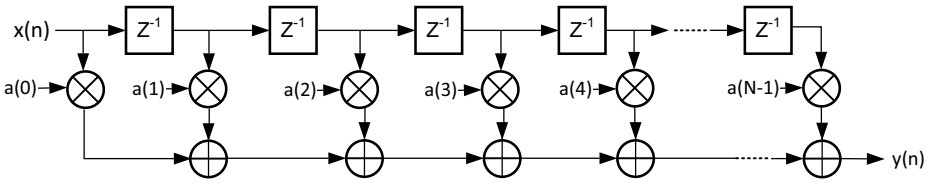



Figure 7.10.: FIR filter implementation with $N$ taps (coefficients), requiring $N-1$ storage cells ($Z^{-1}$), $N$ multipliers, and $N-1$ adders. [177]

**First Test** It has been attempted to implement an FIR filter in the CB-SADC firmware, with the goal in mind to cut away high frequency noise (i.e. the implementation of a low-pass). The coefficients $a(k)$ have been determined with the help of a PARKS-MCCLELLAN filter design algorithm, implemented and configurable through a web applet [178]. The specification of the filter will be explained with the help of Figure 7.11. The algorithm's performance is determined by five input parameters:

**Pass-Band Boundary** $\Omega_p$ is the end of the lower frequency band that should contain all frequency components of the original signal that have a sufficient SNR.

**Pass-Band Ripple** $\delta_p$ specifies the allowed deviation from unity gain in the pass band. Ideally the deviation is close to zero.

**Stop-Band Boundary** $\Omega_s$ is the beginning of the band that follows after the transition band, in which the attenuation drops steeply (compared to analog filters of reasonable complexity).

**Stop-Band Ripple** $\delta_s$ Same as Pass-Band Ripple, but for the stop band.

**Stop-Band Attenuation** $\Omega$ specifies the minimum attenuation that should be reached in the stop band.

**Number of Coefficients** implicitly also specifies the number of required adders and multipliers.

It is evident that there is a certain balance between those parameters: the number of required coefficients scales with higher stop band attenuation $\Omega$, and inversely with pass and stop band ripple $\delta_p$ and $\delta_s$, as well as the width of the transition band $\Omega_s - \Omega_p$. The compromises have to be chosen carefully. Luckily for broadband applications, the pass and stop band ripple are not as critical, such that it is possible to chose a smaller transition band width and a higher stop band attenuation $\Omega$.

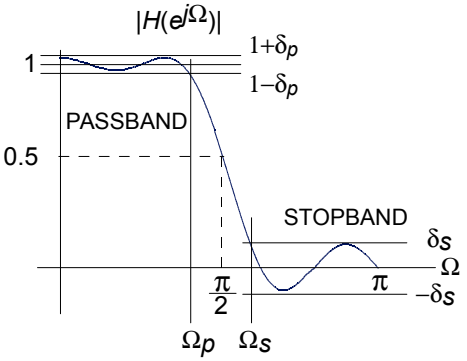



Figure 7.11.: Transfer function of a half-band FIR filter. The x-axis shows the frequency, the y-axis the attenuation. The pass band contains the useful signal, followed by the transition band with a steep decrease of the gain, until the stop band is reached. [177]

When implemented in an FPGA, the so-called DSP slices are used for the implementation, since they contain dedicated logic for the implementation of adders and multipliers (cf. Fig. 5.16 on page 83). The KINTEX-7 on the CB-SADCs contains 600 DSP slices, and usually one slice is required per coefficient. This implies that up to 18 slices can be used per channel ($600/32 = 18.75$). It will be shown in section 7.5.5 that two slices will probably be required for another algorithm, possibly leaving only 16 for the FIR filter.

At the time a low-pass has been implemented with 18 slices, using the XILINX FIR compiler [177]. The previously-mentioned decimation to a sampling rate of $f_S = 20\,\mathrm{MS/s}$ (which is implemented with "regular" logic (CLBs, cf. Fig. 5.14), not costing any DSP resources) has been left active, since the FIR filter's effectiveness scales with $\frac{\Omega_p}{f_S}$. The pass band boundary was set to $\Omega_p = 1\,\mathrm{MHz}$, and the stop band boundary to $\Omega_s = 4\,\mathrm{MHz}$, leaving a transition band of 3 MHz width. Limited by the number of DSP slices, the achieved pass band ripple was $\delta_p = 0.06\,\mathrm{dB}$, with a stop band attenuation of $\delta_p = -46.35\,\mathrm{dB}$. Figure 7.12 shows the according transfer function. It gives a good feeling of how powerful the achievable FIR filter can be, even though the resources per channel are not abundant.

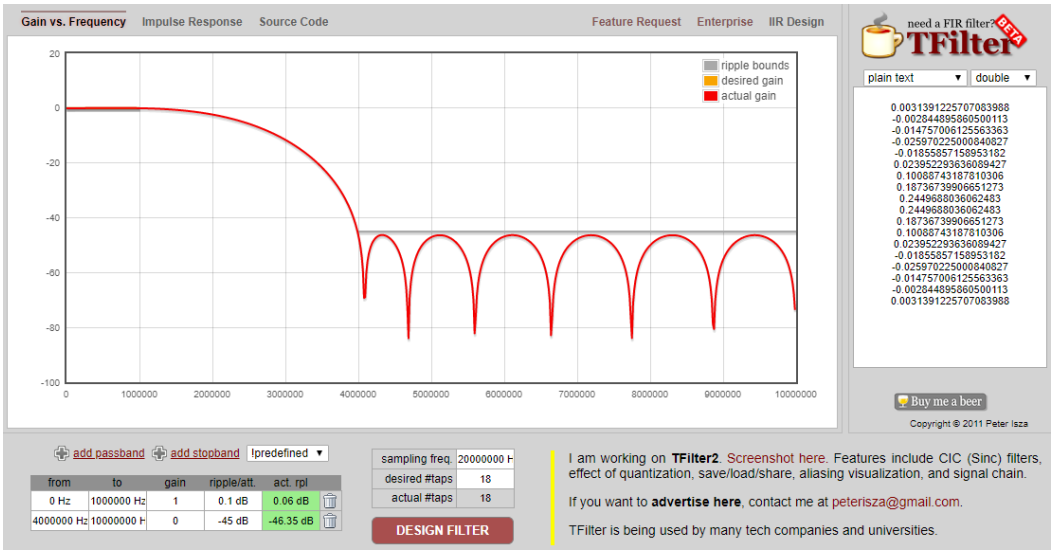



Figure 7.12.: Transfer function of the low-pass FIR filter as calculated by TFILTER [178] for the CB-SADC firmware. With the available 18 DSP slices and a previous decimation to 20 MS/s, it was possible to achieve a suppression of −46.35 dB for frequencies above 4 MHz.

**Outlook** During the beam times the FIR filter was left deactivated. This decision has been made due to a problem that was not understood at the time: although the pass band, by definition, has an attenuation of 0 dB, the numerical values were not as expected. This issue has to be investigated further before the FIR filter can be implemented safely in a production beam time. It is planned to set up a proper DSP simulation framework with MATHWORKS MATLAB and SIMULINK, which in conjunction with the XILINX SYSTEM GENERATOR FOR DSP plugin are capable to produce code for immediate implementation into the FPGA. With this, it should be possible to obtain a better understanding of the filter's effect.

### Buffering and SMA Filter

As an alternative to the FIR filter, a simple moving average (SMA) filter has been implemented. It was mentioned previously that this can be expressed also by equation 7.8, with $a(k) = \frac{1}{N}$. With a moving average window of $N$, the filtered signal $y(n)$ can be written as

$$y(n) = \frac{1}{N} \sum_{k=0}^{N-1} x(n-k) \quad n = 0, 1, \ldots \tag{7.9}$$

This again can be rephrased to

$$y(n) = \frac{1}{N}(y(n-1) - x(n-N) + x(n)). \tag{7.10}$$

Therefore, effectively, the filter does not require $N$ coefficients, but rather two, if the result of the previous sample $y(n-1)$ is stored in a shift register in addition. Thanks to the simplicity of this filter, it can be implemented easily with one constraint: if $N = 2^n$ it is possible to achieve a division by a simple bit shift operation. An integer division is otherwise only possible with more complex algorithms, usually involving DSP resources. Because of this, the simple moving average filters in this thesis all have a windows length of $N = 2^n \quad ; n \in \mathbb{N}$.

For the implementation, a behavioral description of a classical buffer has been written, invoking either distributed RAM (flip-flops) or block RAM as storage cells (depending on the depth of the buffer). This manual adjustment of the depth parameter, overriding XILINX VIVADO's implementation automatism, allows for a flexible use of the storage resources on the FPGA. A number of such buffers is concatenated, so that SMA filters of different depths can share the same storage elements. This is shown in Figure 7.13.

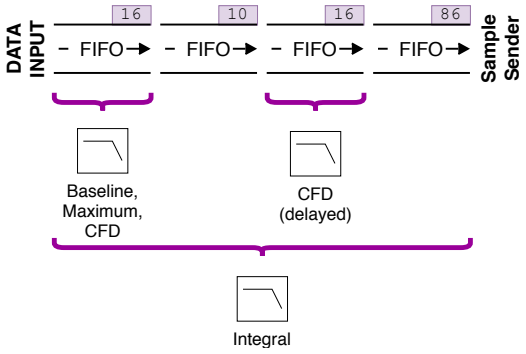



Figure 7.13.: Four concatenated buffers, using First In, First Out (FIFO) storage with distributed RAM or block RAM. The number at the top right indicates the depth in samples. Below, three different Simple Moving Average (SMA) filters are depicted, which are used as low passes and share common buffer resources. The complete buffer chain has a length of 128 samples when the data is decimated to 20 MS/s. The delayed data is also used to feed the sample sender, allowing it to access parts of the waveform before the trigger time.

The SMA filters will be used in various modules of the feature extraction (section 7.5) and conclude the Data Preprocessing section. In the next section, the network implementation will be introduced.

## 7.4. Network

The digitized and processed data needs to be transferred from the CB-SADCs to a computer system (LEvB, see section 2.9.4). It was already discussed for the 16-channel PANDA-SADC prototype in section 4.3 that Ethernet is the carrier of choice for the data, and the argument is still the same with the 64-channel CB-SADC. For the 16-channel PANDA-SADC prototype, the Ethernet implementation included only the exchange of MAC packets via 1 Gbit/s fiber connection, which was sufficient for first tests. This needs to be extended to fulfill the following requirements:

- The connection should be possible through both copper cable and fiber. This allows a versatile operation independent from the available infrastructure, especially during the development phase.
- Regular network hardware has to be compatible and needs to be able to route packets from $N$ CB-SADCs to $M$ LEvBs.
- A loss of data packets cannot be tolerated, all data must arrive at the LEvB.

Figure 7.14 shows an overview of the network module architecture in the CB-SADC firmware, including references to the sections in which they are explained. The most notable differences to the firmware for the 16-channel PANDA-SADC prototype is the implementation of UDP/IP (section 7.4.3) and the development of a custom reliability protocol (section 7.4.5). All modules will be explained in the following sections after two short paragraphs introducing the AXI4-STREAM standard and the OSI model.

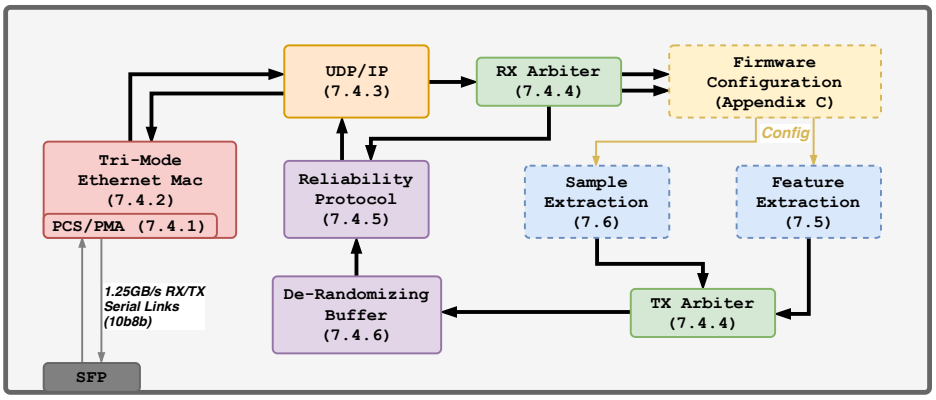



Figure 7.14.: Overview of all modules inside the FPGA connected to the network directly or indirectly. The (physical) SFP module is connected to the FPGA, using two high-speed serial links (RX/TX). The Tri-Mode Ethernet Mac (TEMAC) implements the *physical layer* and *data link layers*. The UDP/IP module implements the *network* and *transport layers*. In the RX (receiver) path, the RX Arbiter sorts the incoming packets to different configuration modules. In the TX (transmitter) path, network packets from the sample extraction and feature extraction are arbitrated and stored in a de-randomization buffer to minimize the dead time. Afterwards, the custom reliability protocol ensures the successful transmission of all packets. All thick black arrows represent AXI4-STREAMs.

**AXI4-Stream** For the connections between all network modules the proprietary XILINX AXI4-STREAM standard is used. This powerful, yet simple protocol will also be used throughout other modules of the firmware, wherever possible. It will be explained with the help of Figure 7.15: the data is transmitted in an 8 bit wide bus called `TDATA` with a clock `ACLK` of 125 MHz (which is equivalent to 1 Gbit/s). Two control signals are transmitted in parallel to the data: `TVALID` to indicate the presence of valid data on the data bus, and `TLAST` to indicate the last byte of the stream (the end of the transmission). The receiving side indicates whether it can process the current transmission (byte) with the `TREADY` signal. The AXI4-STREAM standard is documented in [179].

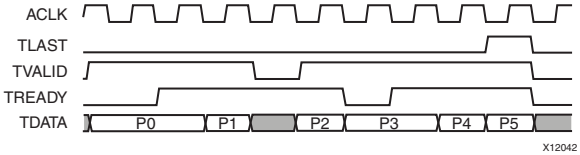



Figure 7.15.: Transmission of six bytes with AXI4-STREAM. A byte is transmitted successfully when `TVALID` and `TREADY` are high in the regarding clock cycle. [179]

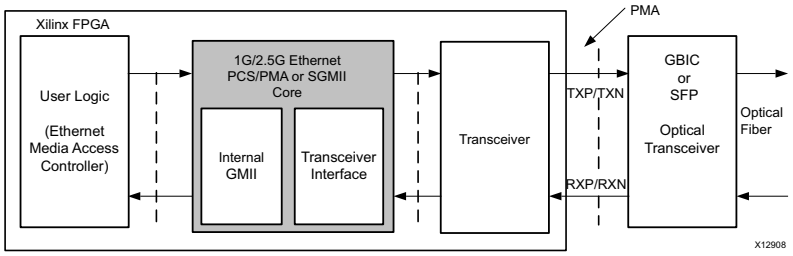
**OSI Model** The OSI (Open Systems Interconnection) model [180] describes the different layers of network communication, with the goal to find a standardized description, where different applications, protocols, or technologies on different layers are interchangeable. The model is suitable to guide through the implementation of the Ethernet interface in the CB-SADC firmware and it helps to understand the tasks of the different modules.

| | Layer Name | Description | CB-SADC Implementation | Section |
|---|---|---|---|---|
| 7 | Application | Create the Data | Feature/Sample Extraction | 7.5/7.6 |
| 6 | Presentation | Format the Data | Packet Assembly | 7.5.6/7.6.2 |
| 5 | Session | Establish Connections | *not implemented* | - |
| 4 | Transport | End-to-End Connection/Reliability | UDP/Custom Reliability/Arbiter | 7.4.3/7.4.4/7.4.5 |
| 3 | Network | Logical Addressing | IP | 7.4.3 |
| 2 | Data Link | Physical Addressing | MAC | 7.4.2 |
| 1 | Physical | Raw Data on Physical Medium | 1000BASE-X/T | 7.4.1 |

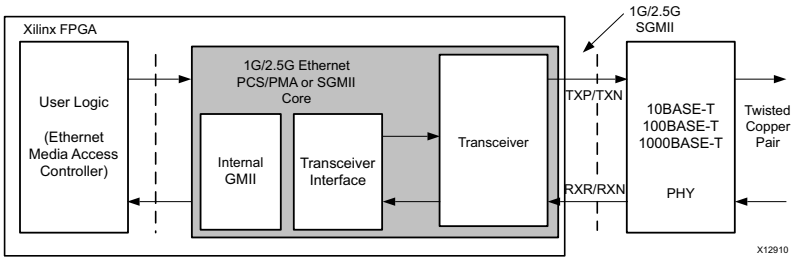
Table 7.1.: List of the seven layers of the OSI model. Layer 1 describes the physical medium used for transportation, which in case of the CB-SADC can be either 1000BASE-X (Gigabit Ethernet over fiber) or 1000BASE-T (Gigabit Ethernet over copper cable). It is implemented in the firmware with the **PCS/PMA IP Core**. Layer 2 implements a hardware address (*MAC address*) and a packet structure for data transmission, using the **TEMAC IP Core**. Layer 3 implements an IP address, well known from local area networks (LAN) and the internet. Layer 4 implements the UDP protocol that allows to direct data based on *ports*. Both are implemented with the **Opencores UDP/IP IP Core**. Since UDP lacks reliability, it is complemented by a **Custom Reliability Protocol**. Layer 5 is not implemented, since it is not required in this application. Layer 6 and 7 are can be associated with the feature extraction and sample extraction.

### 7.4.1. Physical Layer: PCS/PMA IP Core

The lowest layer of a network implementation, according to the OSI model [180], is the physical layer. The physical layer comprises PCS (Physical Coding Sublayer), PMA (Physical Medium Attachment Sublayer), and PMD (Physical Medium Dependent Sublayer). Physically, only the PCS and PMA are implemented in the FPGA, while the PMD is part of the network module plugged into the CB-SADC's SFP slot. The interface towards the SFP slot is a high speed serial link (to two differential pin pairs of the KINTEX-7 FPGA). It can be either PMA or the so-called SGMII (Serial Gigabit Media Independent Interface), depending on the individual SFP module used. The physical layer is implemented in the FPGA using the XILINX PCS/PMA IP Core [143]. It can be selected whether a network connection with **1000BASE-T** (copper cable) or **1000BASE-X** (fiber) should be implemented, resulting in different interfaces towards the SFP slot. This is visualized in Figure 7.16.




(a) 1000BASE-X (fiber) implementation.




(b) 1000BASE-T (copper) implementation.

Figure 7.16.: Different schematic of a fiber and copper network implementation. The left module is the TEMAC core (explained in the following section), the right side represents the (physical) module plugged into the SFP cage. [143]

**Modification for 100 MHz Clock** It was necessary to extract the complete PCS/PMA IP Core and modify the source files for compatibility with the digitizing clock and the network clock. Both are supplied by the Phase-Locked Loop (PLL) (LMK04806) as described in section 5.3, and are derived from the same Voltage-Controlled Oscillator (VCO) frequency, which for the LMK0480X series can be in the range of 1840 MHz to 3072 MHz (see Fig. **??**). The digitizing clock is fixed at 80 MHz, while the standard network clock is 125 MHz. Their least common multiple is 2000 MHz, this requires the use of the LMK04803. Since it is desirable to maintain as much compatibility with the PANDA-SADC as possible, it was attempted

| PART NUMBER | VCO FREQUENCY |
|---|---|
| LMK04803 | 1840 to 2030 MHz |
| LMK04805 | 2148 to 2370 MHz |
| LMK04806 | 2370 to 2600 MHz |
| LMK04808 | 2750 to 3072 MHz |

to solve the problem using the LMK04806. This was possible by modifying the transceiver, which is instantiated by the PCS/PMA IP Core. In the FPGA fabric, a so-called Channel PLL (CPLL) (Figure 7.17) is used to generate stable reference frequencies for the network fabric (RX/TX data path clocks), whose parameters need to be modified to match the VCO frequency. The frequency of the VCO of the CPLL depends on the reference clock `PLLClkin` and the CPLL's internal dividers N1, N2, M:

$$f_{\mathrm{PLLClkout}} = f_{\mathrm{PLLClkin}} \cdot \frac{\mathrm{N1} \cdot \mathrm{N2}}{\mathrm{M}} \tag{7.11}$$

$$= 125\,\mathrm{MHz} \cdot \frac{5 \cdot 4}{1} = 2.5\,\mathrm{GHz} \tag{7.12}$$

By modifying the divider N2 (internal parameter name: `CPLL_FBDIV`) from 4→5, it is possible to operate the CPLL with $f_{\mathrm{PLLClkin}} = 100\,\mathrm{MHz}$, and the LMK04806 can be configured as shown in Figure F.4 on page 261.

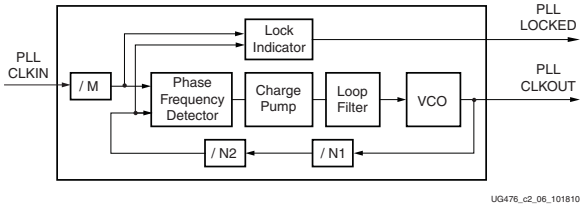



Figure 7.17.: Synthesis of the network clock frequency $f_{\mathrm{PLLClkout}} = 125\,\mathrm{MHz}$, which is used across the all network related modules in the CB-SADC firmware. The input frequency $f_{\mathrm{PLLClkin}} = 100\,\mathrm{MHz}$ is directly supplied by the LMK04806. [145]

### 7.4.2. Data Link Layer: Tri-Mode Ethernet MAC IP Core

The TEMAC IP Core is responsible for the transmission of data on the physical link that is established by the PCS/PMA IP Core. The structure of the transmitted data is a so-called Ethernet Type II frame, whose content is shown in Table 7.2. On the Physical layer it is extended by an 8 byte *Preamble/Start of Frame Delimiter* at the beginning, and a forced Inter-Packet Gap (between two frames) of 12 byte at the end.

This frame structure is already sufficient to exchange data in a point-to-point network, as demonstrated for the first implementation with the 16-channel PANDA-SADC in section 4.3. The payload can be up to 1500 byte in a regular network environment, but it is possible to transmit much larger frames using the so-called *Jumbo Frames*, which the TEMAC IP Core in principle supports. This payload could contain the feature extraction or sample extraction data directly (as it was the case in section 4.3), or it may encapsulate the frame structure of the higher OSI layer implementations, as shown in the following section.

### 7.4.3. Network and Transport Layer: UDP/IP IP Core

Higher level protocols are necessary for proper routing of network packets, when for example multiple CB-SADCs are connected to one or more Local Event Builders (LEvBs) through a network switch. In this section, the third OSI layer using User Data Protocol (UDP)/Internet Protocol (IP) will be presented and their implementation discussed.

| Preamble/SFD | MAC Dest. | MAC Src. | EtherType | Payload | FCS | IPG |
|---|---|---|---|---|---|---|
| 8 byte | 6 byte | 6 byte | 2 byte | 46 byte to 1500 byte | 4 byte | 12 byte |

Table 7.2.: Content of an Ethernet Type II frame, consisting of the destination and source MAC addresses, followed by the EtherType, payload, and frame check sequence (FCS). The EtherType depends on the content of the payload. In this implementation it is either `0x0800` to indicate an IPv4 frame, or `0x0806` for an ARP request (explained in section 7.4.3). The payload usually encapsulates the frame structure of the higher level protocols (such as IPv4), but can in principle also carry raw data. The frame check sequence is used to detect errors in the transmission, usually using CRC (cyclic redundancy check). If the checksum calculated in the receiver differs from the transmitted checksum, the frame will be discarded (without any action taken to request a re-transmission of the frame on this level). In total the Ethernet Type II frame adds an overhead of 38 byte.

**Internet Protocol** In a pure MAC environment, the MAC address is the only source/target *identifier* to exchange packets. Using IP, each device is additionally assigned an IP address, consisting of four octets which are represented decimally (e.g. `192.168.0.10`). The network traffic is now directed based on this IP address. It is the task of the **Address Resolution Protocol** (ARP) to resolve those IP addresses: if the recipient of a network packet (and hence the MAC address) is unknown, a packet with the content "who has `192.168.0.10`?" will be broadcast to all devices in the network that can be reached by the network switch. The owner of this address will now respond with "`192.168.0.10` is at `XX:XX:XX:XX:XX:XX`" (indicating the MAC address). This allows an intermediate device (e.g. a network switch) to be aware of the MAC address belonging to a certain IP address, and more importantly, to which (physical) *port* it is connected. In a non-switched network, all packets would be sent to all connected devices, leading to an ever growing network traffic.

The Internet Protocol is commonly extended on the Transport Layer by either the TCP or UDP. A first choice would usually be the TCP, given that it inherently guarantees the successful transmission of each packet through an intricate session management and handshaking. Historically, the protocol was designed to this extent with a bad end-to-end connection quality in mind; this is is why it is still successfully used for, for example, mobile data connections, where a very poor signal quality and frequent change of base stations is challenging. This is at the same time the reason not to use it in the CB-SADC firmware: the CB-SADCs and the LEvB will be connected in a very reliable way, such that a lost packet will occur with diminishing probability under normal operation conditions. Furthermore, TCP is known to be quite resource consuming when implemented in FPGAs due to its abundant feature set. It is not feasible to use it with the Kintex-7 FPGA of the CB-SADC. As a result, the UDP has been implemented and will be introduced in the following paragraph.

**User Data Protocol** The much slimmer UDP extends the Internet Protocol by *ports*, which range from 0 to 65 535. Each transmission has a source and a destination port, which allows to route the incoming packets (now called *datagrams*) to different processes/modules, without the need for further user-defined identifiers in the payload. Due to this, the connection is no longer regarded as *host-to-host*, but instead as *process-to-process*. Together with the IP, an

additional 28 byte of protocol overhead is added. Taking the Ethernet Type II frame into account, this sums up to a total protocol overhead of 64 byte.

The simplicity of the UDP comes at a price: it is essential to find a way to guarantee a reliable transmission of data. This will be discussed in section 7.4.5.

**Implementation with OpenCores IP Core** It has not been attempted to write a UDP/IP implementation from scratch in the scope of this thesis, as it is quite a demanding task on its own. This decision was strongly encouraged by the existence of an open-source IP Core [181]. So far the implementation has worked without problems and there was no need for any modifications.

**Jumbo Frames** The additional headers of UDP (8 byte) and IP (12 byte) reduce the Maximum Transmission Unit (MTU) from 1500 byte to 1480 byte (cf. *Payload* in Table 7.2). This limit can be increased to 8980 byte by using jumbo frames. The TEMAC and the UDP/IP core support jumbo frames, and so does the used network switch and network interface card in the LEvB. It leaves room for further optimization, as it reduces dead time from the protocol overhead and from the round-trip-time (in case of two way communication). With the current operation modes and scenarios that were tested with the CB-SADC, this was not necessary and has not been tested yet.

### 7.4.4. RX/TX Arbitration

The interface of the UDP/IP IP core consists of two AXI4-STREAMs, one to send data (TX) and one to receive data. Internally, the firmware has multiple modules that expect to receive or transmit, which are also depicted in Figure 7.14. In both instances this requires a module to switch or arbitrate the packets.

**RX Path** The received UDP packets are evaluated by the RX switch module. Depending on the UDP destination port, the packets are forwarded to one of four modules, as shown in Table 7.3. Three ports are associated with configuration routines (see appendix C) and one directly leads to the module for the reliability protocol (section 7.4.5).

| Port | Module |
|---|---|
| 4096 | Configuration of feature and sample extraction |
| 4097 | Configuration of LTM9009-14 ADCs |
| 4098 | UDP reliability protocol |
| 4099 | Configuration of operating parameters |

Table 7.3.: UDP Ports used for the routing of incoming packets in the firmware.

**TX Path** Packets can be either transmitted by the feature extraction or the sample extraction, and for the future it is planned to allow a read-back of the configuration as a third source. This will ensure that all configuration packets have reached the CB-SADC and that it is configured as intended. The arbitration between those sources is achieved with an AXI4-STREAM Interconnect IP Core [182].

### 7.4.5. Reliability Protocol

It was decided to attempt a rather simple approach to guarantee a reliable packet transmission: a *burst* of 32 UDP packets is sent from the CB-SADC to the LEvB, which in return has to acknowledge this burst. If not all 32 packets were received, the LEvB can request them individually to be sent again. It will be explained in detail in the next paragraphs how this reliability protocol (also called *burst* mode/mechanism) is realized.

#### Implementation

As a foundation for the *Burst* mechanism, 32 AXI4-STREAM compliant FIFO buffers have been implemented, each capable to hold a network packet up to an MTU of 2 kbyte. The incoming packets, originating either from the feature extraction or sample extraction, are stored in those buffers sequentially, using an AXI4-STREAM Interconnect IP Core [182], which arbitrates with a round robin algorithm[3]. As soon as all buffers are filled[4], they are emptied towards the UDP/IP core. A so-called *Burst Header* is appended automatically to the payload as shown in Table 7.4.

| 63-56 | 55-48 | 47-40 | 39-32 | 31-24 | 23-16 | 15-8 | 7-0 |
|---|---|---|---|---|---|---|---|
| 0xCB | 0xCB | 0xCB | 0xCB | BUF_COUNT | TYPE | BUF_NR | BUF_TOTAL |

Table 7.4.: Content of the 8-byte *Burst Header*, ordered from bit 63 down to 0.

The *Burst Header* starts with four consecutive bytes containing `0xCB`, which serve no special purposes. They have been proven helpful as a human-readable delimiter, and can be temporarily replaced to transmit additional data. The fifth byte (`BUF_COUNT`) is a counter, increasing with every new *Burst* (hence every 32 packets). The field `TYPE` is used to distinguish the origin of packets in the firmware, which currently can be either the feature extraction or sample extraction. `BUF_NR` is the position of the packet within the burst (0..31), while `BUF_TOTAL` specifies the burst size (32 packets in the current implementation). This allows to change the burst size dynamically without reprogramming the LEvB. Furthermore, it was introduced in case the need arises to send an incomplete burst, in order to not delay the data too much (for example with a very slow event rate).

The communication (or handshaking) between CB-SADC and LEvB will be explained with the help of the state machine shown in Figure 7.18, and the three possible *modes* of the burst mechanism (cf. appendix C) are explained in the following:

- `CMD_RST`: The firmware starts in the *Reset* state, where no transmissions are possible. At the same time, a reset is issued to the feature extraction and sample extraction modules to ensure clean starting conditions.

- `CMD_PUSH`: In *Push* mode, the reliability protocol is disabled. After packets in the FIFOs buffers are sent, the buffers are cleared immediately and can be filled with new packets from feature extraction or sample extraction. This mode allows the highest transmission rate, but the UDP packets might get lost anywhere on the way.

[3] Due to this, the packets are not necessarily sent in the order that they are created by the CB-SADC.
[4] It will be shown in section 7.4.6 that this condition can be optimized and omitted.

- `CMD_ACK`: In *Acknowledge* (ACK) mode, the FIFO buffers retain all packets after the burst of 32 packets has been sent. After the LEvB has received and verified the integrity of the whole burst, a *clear buffers* (`CMD_CLEAR_BUFFER`) command is sent to the CB-SADC, and the buffers can accept new data. If at least one packet from a burst is received, the LEvB expects to receive the remaining packets within a configurable time. When this timeout is reached, it will issue a `CMD_RESEND_BUFFER` command to the CB-SADC, requesting the missing packets that are still in stored in the FIFOs.

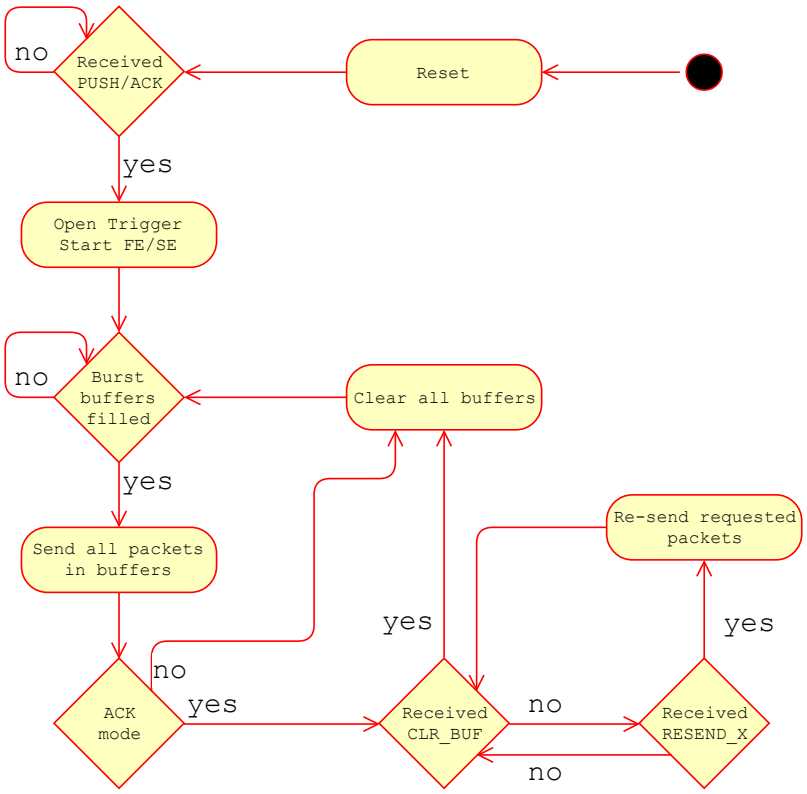



Figure 7.18.: State machine of the Reliability Protocol. The state machine is set back to the beginning immediately if a RESET command is received.

### 7.4.6. De-Randomizing Buffer

As long as the FIFOs are not completely filled, the CB-SADC can accept a new trigger very quickly. But after a *burst* has been sent and the CB-SADC is waiting for the *clear buffers* command from the LEvB, it can evidently not fill the buffers with new packets, meaning the feature extraction and sample extraction have to be halted. This in turn will force the CB-SADC to signal *Busy* towards the trigger/sync system for an extended period, possibly blocking the DAQ.

This fluctuation is smoothed out with an additional AXI4-STREAM FIFO buffer, instantiated between the TX Arbiter module and the Reliability Protocol module (cf. Figure 7.14). Apart from the fluctuations due to the network, this also absorbs the stochastic fluctuations

of the experiment trigger. The burst buffers, contrary to the description in the previous section, can now almost immediately transfer new data to the UDP/IP core after they have been cleared. This happens without waiting for all 32 buffers to be loaded, but only if the de-randomizing buffer is filled up enough to contain at least 32 network packets, ensuring a complete burst transmission within a short time. With this, a substantial delay of at least $32 \cdot 352\,\text{byte}/125\,\text{Mbyte/s} \approx 90\,\mu\text{s}$ could be avoided, which is the time necessary to fill all 32 burst buffers with feature extraction packets. For sample extraction packets, the delay would even amount to as much as 264 µs. The dead time, throughput, and efficiency of the CB-SADC will be evaluated further in section 9.9.

This concludes the section of the network implementation in the CB-SADC firmware. The next section will describe the feature extraction.

## 7.5. Feature Extraction

In this section, the algorithms developed for the CB-SADC feature extraction will be described. Feature extraction, in the scope of this thesis, refers to the extraction of information from the full sampled waveforms, with the goal of retaining the relevant information without having to store the whole waveform. The most relevant information to be extracted is the waveform's integral, which is proportional to the energy deposited in a crystal of the Crystal Barrel Calorimeter, but more interesting features can be acquired, as will be shown. The basis for some algorithms was already laid out when the 16-channel PANDA-SADC was tested (chapter 4), and was improved in the course of this thesis. A major impact on the improvements was made by J. Schultes [111, 122], who has (re-)modeled several algorithms in `C++`, and provided optimized parameters based on offline-tests with recorded waveforms from the Crystal Barrel Calorimeter.

**Overview** Section 7.5.1 will show how the baseline is determined from the waveforms with a simple algorithm. Section 7.5.2 and 7.5.3 present two methods to determine a value proportional to the energy deposited in the Crystal Barrel Calorimeter crystals: either with a simple integration in a window that is fixed with regard to the trigger, or with a continuously running *peak finder* that is optimized to the characteristics of the pulses. The timing information of the pulses is determined with a constant fraction discriminator, explained in section 7.5.4. In section 7.5.5, the most recently added pile-up detection will be discussed. In section 7.5.6, the structure of the network packet will be summarized. In the last section, 7.5.7, the performance of the feature extraction will be evaluated.

All algorithms can be configured in some regards, which will be mentioned throughout the sections. A complete overview of all configuration options will then be provided in appendix C. Evaluations of the algorithms will be shown in the measurements throughout chapter 9.

### 7.5.1. Baseline Tracker

The extraction of the baseline is necessary for the energy determination presented in the following sections, since the quiescent level of the signal is non-zero (see also section 6.1.2). A trivial algorithm was implemented that extracts a value averaged over 800 ns in the window of 1.6 µs to 2.4 µs before the experiment's trigger (see Figure 7.19). This algorithm allows a baseline correction on a per-event basis, as opposed to the data extracted from the QDC, where averaged baselines are measured before each run (every $\approx 30$ min). This baseline value is sent to the LEvB as part of the feature extraction packet and needs to be subtracted from the energy-extraction features. The algorithm is sufficient for clean signals over a reasonably flat baseline. Pulses that fall out of this category, because they contain pile-up or have otherwise unexpected characteristics, are usually filtered by the pile-up detection (section 7.5.5). In this case the full waveform will be written to disk and a more careful baseline extraction is performed offline.

More complex algorithms, which track the baseline over a longer period of time and at the same time determine the noise, may be considered for future improvements [183, 184]. Currently studies for the implementation of an improved algorithm are conducted within a bachelor thesis [142].

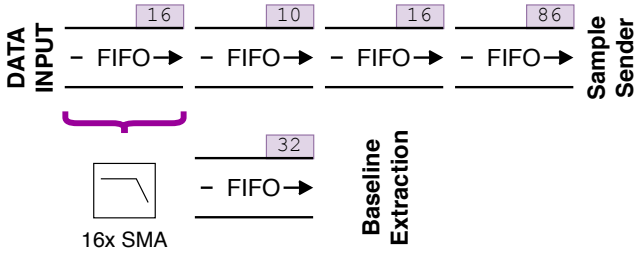



Figure 7.19.: The baseline is extracted from a simple moving average filter over 16 samples (800 ns). The value is delayed by a buffer by 32 clock cycles (1.6 µs) such that it represents a value before the pulse starts rising.

### 7.5.2. Integration

The *Integral* feature is the simplest and most reliable method for the energy determination. It has been designed to behave much like the LeCroy QDC: after a trigger signal is received, the area under the pulse is integrated. In the case of the QDC, this is achieved by charging a capacitor for a defined time, and it requires the analog signal to be delayed (by means of cabling) to arrive after the trigger signal (see section 2.8.8). This requirement does not exist for the CB-SADC, since it digitally buffers the data with sufficient depth.

A simple moving average filter is used to determine the integral, as shown in Fig. 7.13 on page 152. The ideal length for the integration was determined in [122] by calculating the SNR depending on start and end positions of the integration window. A recent result of this method is shown in Fig. 7.20 and yields a length of ≈ 369 samples (≈ 92 samples at 20 MS/s). Due to the limitations of the moving average filter (see section 7.3.5), a window of 128 samples (at 20 MS/s) was chosen, resulting in depth of 6.4 µs. The exact position of the integration can be shifted (even during operation), which is described in the following paragraph.

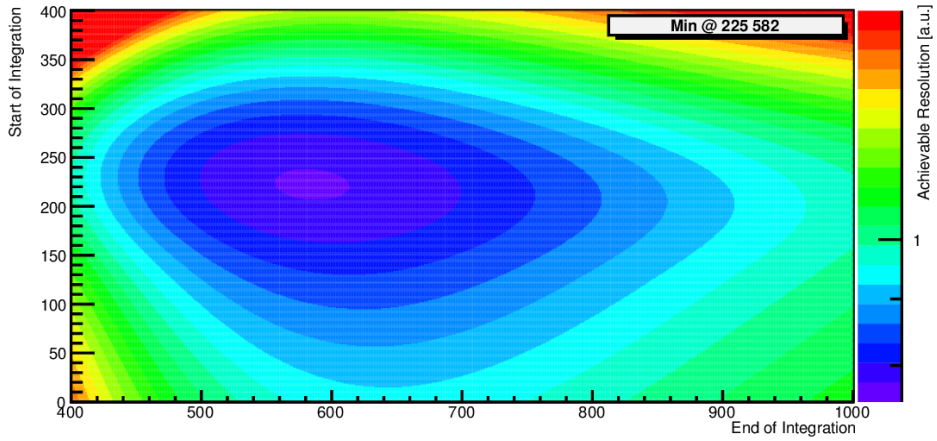



Figure 7.20.: Investigation of the ideal position and length of the integration window. The axes indicate the position in the sampling window at a sampling rate of 80 MS/s. *Courtesy of Jan Schultes.*

**Algorithm and Configuration** A 20 MHz timer is implemented in the firmware, which starts counting as soon as the trigger signal arrives. The timer stops as soon as it has reached a value of `config.sadc.integral_position`, at which point the value of the moving average filter (cf. Fig. 7.13) is stored as a 16-bit unsigned integer `event_data.int(i)`. The optimized position of the integration window starts at a value of `config.sadc.integral_position` = 125, meaning that the integration starts roughly around the time the trigger signal arrives.[5]

### 7.5.3. Peak Finder

The peak finder is a second method to obtain not only an energy-correlated value, but also a timing information. The algorithm works with the 16-fold moving average of the pulse, smoothing the signal removing higher frequency noise. It attempts to detect a maximum as described in the following.

**Algorithm and Configuration** The smoothed signal is evaluated continuously by a state machine, which demands the following conditions:

1. The pulse needs be rising for at least `config.channels(i).max.rising_time` = 20 samples (1 µs) (unchanged samples allowed, but not counted).

2. After the first falling sample (when the first condition was fulfilled) the timestamp and amplitude of the previous sample, which represents the maximum, will be saved.

After receiving a trigger signal, the *first* maximum found within a windows of 10 µs will be stored as a 16-bit unsigned integer `event_data.max(i)`, together with a 32-bit timestamp `event_data.timestamp_max(i).nanoseconds`.

This algorithm was promising in the beginning as P. Mahlberg has found an advantage of such a method over an integration in his bachelor's thesis [130]. But due to the continuous optimization of the integration, leading to a better energy resolution, it was preferred in the analysis over the peak finder [111]. It will however become clear in section 7.5.5 why the peak finder is still required. The timestamp is, naturally, not very precise, as the algorithm is sensitive in the flattest region of the pulse, giving raise to a higher influence of the noise.

### 7.5.4. Constant Fraction Discriminator

A Constant Fraction Discriminator (CFD) was implemented to obtain a timestamp with a better resolution than the previously mentioned *peak finder* timestamp. In contrast to the peak finder, the CFD is sensitive to the *steepest* region of the pulse, leading to a lower influence of the amplitude noise (cf. Fig. 1.22b on page 20). Furthermore, compared to a regular one-threshold discrimination, it does not suffer from *time walk* effects (where the time of the threshold crossing is dependent on the pulse amplitude, cf. Fig. 2.16a on page 44).

[5] Providing a more exact value is not trivial: the asynchronous trigger signal passes a large number of gates for synchronization and translation to different clock domains, and also ICs on the Backplane with non-zero propagation delay; likewise the digitized data from the ADCs is delayed not only by their pipeline architecture, but also the de-serializer and phase calibration.

**Algorithm and Configuration** Figure 7.21 shows how the constant fraction discriminator algorithm works. The original pulse is attenuated by a factor of two, and a copy of the pulse is delayed and subtracted (for both, a 16-fold moving average is used) and the resulting signal is now called the *CFD signal*. The delay has been determined from recorded waveforms as the time between the pulse reaching 50 % and 100 % of its amplitude on the rising slope, which corresponds to roughly 26 samples or 1.3 µs. If configured like this, it results in a signal which has a zero crossing at the time the original pulse reaches its maximum. The algorithm works as follows (listing E.3 shows the source code of the state machine):

1. The CFD signal has to stay above a level of `config.channels(i).cfd.threshold` for at least `config.channels(i).cfd.time_over_threshold` samples.
2. If after this the CFD signal reaches a negative value, a timestamp is created.
3. If the CFD signal stays below − `config.channels(i).cfd.threshold` for at least 2× `config.channels(i).cfd.time_over_threshold` samples, the timestamp is validated.

An additional timer, running up to `config.channels(i).cfd.time_defuse` samples, ensures that the state machine does not get locked up or registers atypical pulses. All three parameters have been continuously by J. Schultes [111, 122] by investigating the waveform data recorded during several beam times. A compromise had to be found between a low detection threshold and noise immunity. In this effort, the box-like criteria, which were used already for initial investigations with the 16-channel PANDA-SADC, have helped to tune the algorithm to the specific shape of the signals.

The 32-bit timestamp is saved in `event_data.timestamp_cfd(i).nanoseconds`. As for the peak finder timestamp, the detection window is up to 10 µs after the trigger.

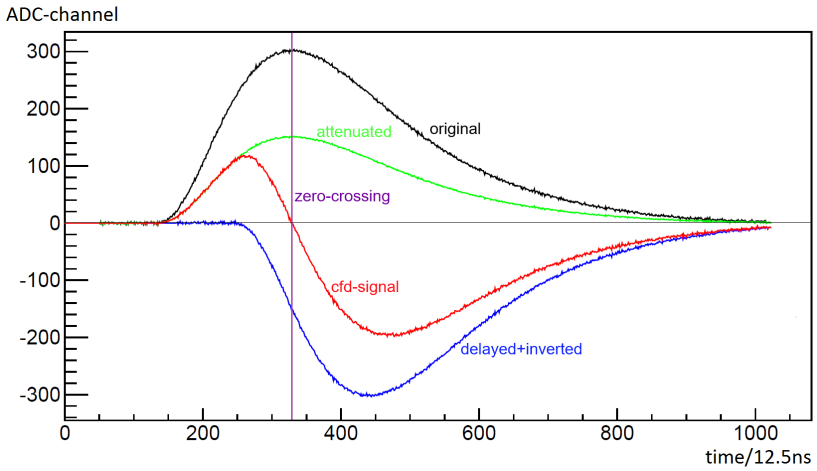



Figure 7.21.: Principle of a constant fraction discriminator. The original pulse (black) is attenuated by a factor of two (green), and a copy of the pulse is delayed and inverted (blue). Both are then subtracted from each other and yield the CFD signal (red), which exhibits a steep zero crossing (violet) at the time the original pulse reached its maximum. [122]

**Performance and Improvements** The resolution of the timing information is limited by the clock period that is used to determine trigger timestamp (`event_data.timestamp_trigger`) and the CFD timestamp (`event_data.timestamp_cfd(i)`), and further by the amplitude noise. In the current implementation, the trigger timestamp is determined with a step size of 12.5 ns. The CFD timestamp only has 50 ns steps, since it operates on the four-fold decimated data with 20 MS/s. The uncertainty, considering the uniformly distributed events, can be calculated with equation 1.7:

$$\sigma_{\mathrm{ts,q}} = \sqrt{\sigma^2_{\mathrm{t,trigger}} + \sigma^2_{\mathrm{t,cfd}}} \tag{7.13}$$

$$= \sqrt{\frac{(12.5\,\mathrm{ns})^2}{12} + \frac{(50\,\mathrm{ns})^2}{12}} \tag{7.14}$$

$$\approx \sqrt{(3.61\,\mathrm{ns})^2 + (14.4\,\mathrm{ns})^2} \tag{7.15}$$

$$\approx 14.9\,\mathrm{ns_{RMS}} \tag{7.16}$$

The amplitude noise influences the occurrence of the CFD's zero crossing, leading to an additional amplitude-dependent jitter (cf. Fig. 1.22b on page 20). The amplitude noise has been measured to be $\sigma_{\mathrm{v}} = 117.3\,\mu\mathrm{V_{RMS}}$ (see equation 9.5). The slew rate of the shaped pulses at 50 % amplitude on the rising edge, being equivalent to the slew rate at the zero crossing of the CFD (Fig. 7.21), was determined from the shaping simulation to be roughly

$$\frac{\Delta v}{\Delta t}(E) \approx 1.645\,\mathrm{mV\,ns^{-1}} \cdot \frac{E}{2.5\,\mathrm{GeV}}. \tag{7.17}$$

This leads to a timestamp jitter of

$$\sigma_{\mathrm{ts,j}}(E) = \sigma_{\mathrm{v}} \cdot \left(\frac{\Delta v}{\Delta t}(E)\right)^{-1} \tag{7.18}$$

$$= 117.3\,\mu\mathrm{V_{RMS}} \cdot \left(0.658\,\mathrm{mV\,ns^{-1}} \cdot \frac{E}{1\,\mathrm{GeV}}\right)^{-1} \tag{7.19}$$

$$\approx 178.3\,\mathrm{ns_{RMS}} \cdot \frac{\mathrm{MeV}}{E}. \tag{7.20}$$

This shows that theoretically, below $E = \frac{178.3\,\mathrm{ns\cdot MeV}}{14.9\,\mathrm{ns}} \approx 12.0\,\mathrm{MeV}$, the achievable resolution is limited by the amplitude noise, and above it is limited by the sampling rate of the trigger signal and the CFD algorithm. Further jitters (sampling clock, trigger signal, physical routing) would have to be taken into account for a more accurate prediction. Possible improvements will be further discussed in section 9.8, where the performance of the CB-SADC's timing capabilities will be investigated based on measurements from a beam time.

### 7.5.5. Pile-Up Detection

From the beginning of the CB-SADC project, the pile-up detection has been advertised as one of the most promising features (cf. chapter 3). Its goal is to automatically *tag* every non-standard waveform during run time, and to store those waveforms to disk for offline analysis and pile-up correction. The method and algorithm described in this section were developed by J. Schultes [111, 122] in a `C++` framework, such that it was prepared well for the translation to VHDL and the implementation in the firmware. The following paragraphs will first present the algorithm and then the implementation.

**Algorithm** The algorithm is based on the following assumption: *If a pulse is not overlapping with another pulse, then there should be a fixed ratio between the* ***integral*** *feature and the* ***peak finder*** *feature.* To develop and test the algorithm, a simulation framework was created in [122]. One million random events were generated, characterized according to Figure 7.22: a pulse with height **H1** is followed with a certain probability by a pile-up pulse at a distance of **Offset2** with a (true) height of **H2**. All parameters were drawn randomly from a uniform distribution, and in addition the positions of the pulses were convoluted with a jitter drawn randomly from a Gaussian distribution.

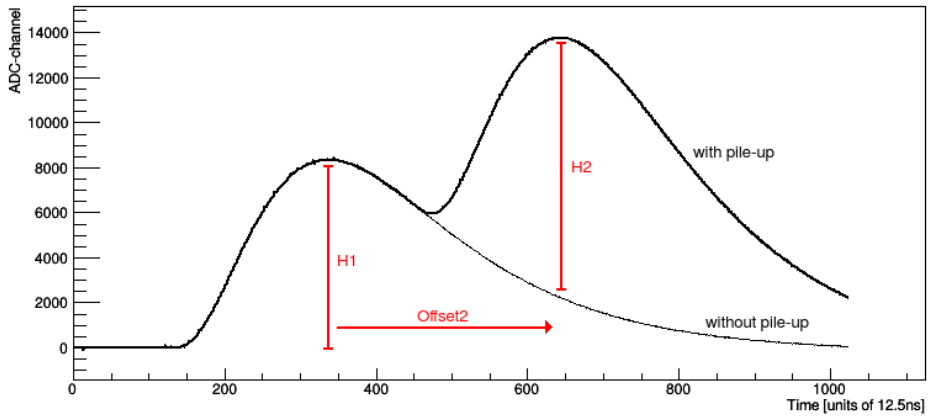



Figure 7.22.: Definition of a pile-up event. A second pulse with true amplitude **H2** is sitting on top of the falling slope of the first pulse with amplitude **H1**, with a distance (between the maxima) of **Offset2**. The event shown is an actual pulse recorded with the CB-SADC without decimation. [122]

The outcome of the simulation is shown in Figure 7.23, where the ratio of $\frac{\text{integral}}{\text{peakheight}}$ is histogrammed against the peak height of the first pulse. A prominent horizontal band shows the validity of the assumption.

It was now attempted to find a boundary to classify the pile-up region with the formula

$$\frac{\text{integral}}{\text{peakheight}} = A \pm \left( B + \frac{C}{\text{peak}} \right) \tag{7.21}$$

which is indicated with black lines. The inverse scaling with the peak height reflects the divergence for lower amplitude pulses, where the SNR has a strong effect on the calculation of the ratio. With a boundary chosen after some optimization, the outcome of the algorithm was compared to the simulation. It was in this case then possible to achieve 7.05 % false negatives and 1.60 % false positives, however, both numbers can be tuned further by modifying the parameters. It is beneficial to accept more false positives by *widening* the boundaries, which will lead to less false negatives at the price of some additional disk space.

**Implementation** For implementing such an algorithm in an FPGA, it is most helpful to refrain from using division, which would require an extensive use of resources. Equation 7.21 can easily be rewritten to contain only multiplications and additions/subtractions, such that the conditions for upper and lower limit (indicating that a pulse does NOT contain pileup)

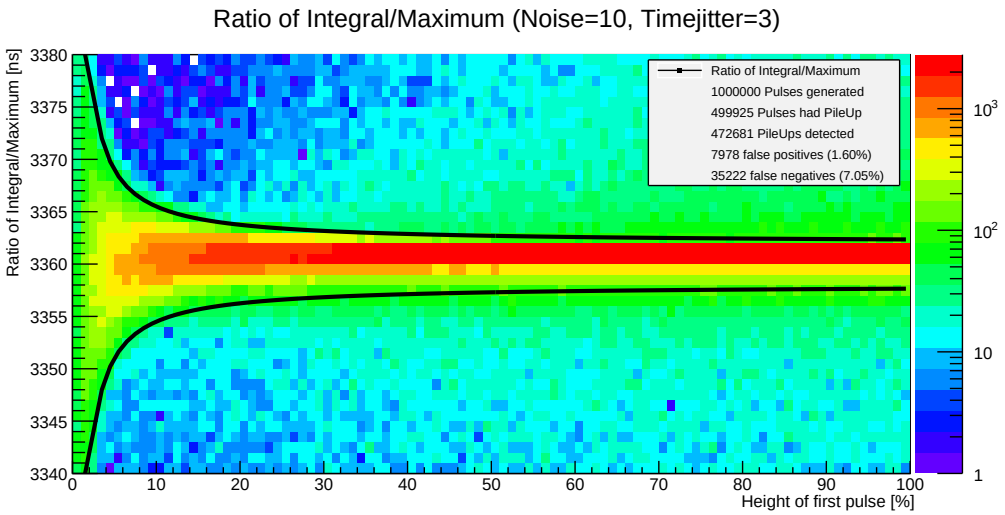



Figure 7.23.: Simulation of one million pile-up events with randomly drawn positions and amplitudes. The red region shows the expected accumulation of pulses with a similar ratio of $\frac{\text{integral}}{\text{peakheight}}$. The boundaries are deliberately not exactly centered around the most probable ratio. This allows to take pulses into account which are in fact not pile-up events, but only slightly shifted in time with respect to the expected position, leading to a slightly lower ratio. [122]

become

$$\text{integral} < (A + B) \cdot \text{peakheight} + C_u \qquad \text{and} \tag{7.22}$$

$$\text{integral} > (A - B) \cdot \text{peakheight} - C_l. \tag{7.23}$$

As indicated here already, the parameter describing the curvature of the boundaries differs from upper to lower boundary, since it was shown later that this leads to even better results [122]. In the firmware, all those parameters can be configured during runtime:

- **A** is called `config.sadc.pileup.ratio`
- **B** is called `config.sadc.pileup.threshold`
- $\mathbf{C_u}$ is called `config.sadc.pileup.upperslope`
- $\mathbf{C_l}$ is called `config.sadc.pileup.lowerslope`

The algorithm could be implemented with a *cost* of two DSP slices per channel (required for the multiplications). It was tested and evaluated within the first and second production beam time, where the boundaries have been set to very conservative levels, shifting the weight towards more false positives. This has led to more extracted and saved waveform samples, but in turn it will help to find more appropriate boundaries in the future. A detailed analysis will be provided in the dissertation of J. Schultes [111]. The pile-up detection can be seen at work in section 9.7.

### 7.5.6. Packet Content

This section summarizes all feature data being sent from the CB-SADC to the LEvB. The structure of the feature extraction packet is shown in Table 7.5. In the current implementation, the packet has a size of 352 byte. It starts with the 8-byte *Burst Header* (cyan), as described in section 7.4.5. The next 20 byte are the *Event Header* (magenta), containing:

- The **Event ID** is a counter that increases by one with every received trigger. It is stored as a 32-bit value `event_data.counter_event`. The counter is reset at the start of every data taking run.
- The **Buffer Number**, as transmitted with every event by the trigger/sync system, is stored as a 5-bit value `event_data.daq_bufnr`.
- The **Integral Position** reflects the configured position of the integration window relative to the trigger in units of samples. It is stored as a 16-bit value `integral_position`
- A **Timestamp** is created to mark the trigger's time relative to the power-up or last reset of the CB-SADC. The timestamp is stored as two 32-bit integers with a resolution of 12.5 ns, `event_data.timestamp_trigger.seconds` and `*.nanoseconds`.
- A 32-bit vector `event_data.send_sample` stores a bit mask for all 32 channels, indicating whether a sample extraction packet will be sent for that channel.

Additionally, 4 byte of information are stored at the end of the packet (magenta), which historically also contained a delimiter with content `0xCBCBCBCB`.

- A 32-bit vector `event_data.pileup` stores a bit mask for all 32 channels, indicating whether pile-up has been detected in the waveform.

The information in the burst header is important for the LEvB to validate the integrity of all packets, and in the later stage (*event saver*) to validate the synchronization of the CB-SADC's data with data from other detectors. The remaining data (white background) are the features sent per channel, as described in the previous sections.

| 0-7 | 8-15 | 16-23 | 24-31 |
|---|---|---|---|
| 0xCB | 0xCB | 0xCB | 0xCB |
| BUF_COUNT | TYPE | BUF_NR | BUF_TOTAL |
| counter_event | | | |
| daq_bufnr | - | integral_position | |
| timestamp_trigger.seconds | | | |
| timestamp_trigger.nanoseconds | | | |
| send_sample | | | |
| *for i in 0 to 31 loop* | | | |
| timestamp_cfd(i).nanoseconds | | timestamp_max(i).nanoseconds | |
| baseline(i) | | integral(i) | |
| max(i) | | | |
| *end loop* | | | |
| pileup | | | |

Table 7.5.: Content of the 352-byte feature extraction packet, with the *Burst Header* in cyan, and the *Event Header* in magenta.

### 7.5.7. Performance

The dead time introduced by sending the feature extraction packet (including Ethernet type II frame, preamble, inter packet gap, UDP/IP overhead) is

$$\frac{352\,\text{byte} + 64\,\text{byte}}{1\,\text{Gbit/s}} = 3.328\,\mu\text{s}. \tag{7.24}$$

Additionally, a fixed time of 10 µs is required after every trigger to wait for the detection of pulses. This limits the trigger rate to a theoretical maximum of

$$\frac{1}{10\,\mu\text{s} + 3.328\,\mu\text{s}} \approx 75.0\,\text{kHz}. \tag{7.25}$$

One has to keep in mind that this theoretical limit can only be reached if the trigger rate is either orders of magnitudes higher, or equally distributed with (or slightly below) this theoretical limit; furthermore, this is only possible in the **PUSH** mode of the reliability protocol, since the **ACK** mode introduces additional delay (cf. section 7.4.5).

**Zero Suppression** To save transmission bandwidth and disk space, it is planned to implement a zero suppression into the CB-SADC firmware. In cases where the energy deposited in the CsI(Tl) crystal of a specific channel leads to an integral value below a certain threshold, all features of this channel are omitted. The packet size will be reduced by 10 byte for every omitted channel. Since the majority of channels can be expected to be below this threshold, this will reduce the amount of data substantially. Currently an equivalent algorithm is implemented on the level of the LEvB, such that the full packet is transmitted from the CB-SADC to the LEvB, but the zero-suppressed packets are sent from LEvB to the central event saver.

This concludes the feature extraction section. In the next section, the sample extraction will be discussed.

## 7.6. Sample Extraction

Apart from the feature extraction packet, the CB-SADC can extract waveforms from all or selected channels upon triggering and send them to the LEvB. During regular data taking in production beam times this would create far too much data, so usually only the feature extraction data is stored. Access to the raw waveforms is still useful for many use cases:

- Development, test, and optimization of algorithms with original data instead of simulations
- Crosscheck of the feature extraction data with offline calculations
- Correction of pulses with detected pile-up
- Oscilloscope-like access to the channels

The following two sections will first describe how the sample extraction can be configured, and afterwards how the network packet is assembled before it is transferred to the TX Arbiter (cf. Figure 7.14 on page 153).

### 7.6.1. Configuration

This section explains the different configuration options for the sample extraction. All of them can be changed during runtime.

**Window Position** The waveform data is extracted from the buffered data stream (compare Figure 7.13 on page 152), delaying it by 6.4 µs. When a trigger arrives, a timer starts counting to a value of `config.sadc.sample_position` and then starts to transfer the data stream to a dedicated block RAM for all channels. Hence the window in which the waveform data is stored can start up to 6.4 µs before the trigger, and can be shifted dynamically to more than 3 ms after the trigger.

**Window Stretcher** In the current implementation, 512 samples can be stored, which, at the decimated sampling rate of 20 MS/s, is equivalent to a length of 25.6 µs. If it is required to observe a longer window, it is possible to *stretch* the window by skipping samples. This can be achieved by configuring `config.sadc.sample_stretcher`, which determines the number of samples that are skipped after one sample was saved.

**Enable Sending** In the regular mode, waveforms are not extracted during a run, and only the feature extraction packet is sent to the LEvB. To receive samples from a particular channel, it can be activated in the bit mask `config.channels(i).send_sample`. A second parameter, `config.sadc.sample_prescaler`, can be used to receive waveforms only from every N-th event (the number configured corresponds to the number of skipped events). It has proven to be a good compromise to operate with `config.sadc.sample_prescaler` $= 99$ during beam times, which will store the waveforms every 100 events and allows to investigate possible problems in the feature extraction with sufficient statistics. By configuring the parameter `config.channels(i).send_sample_on_pileup`, the waveforms will be transmitted automatically if a pile-up was detected by the feature extraction (which is the main purpose of the sample sender).

### 7.6.2. Performance and Packet Content

After all waveform data is saved to the block RAM, which takes $512 \cdot 50\,\text{ns} = 25.6\,\mu\text{s}$ at 20 MS/s, the network packets will be generated sequentially. For each activated channel (be it through the configuration of `send_sample`, `config.sadc.sample_prescaler`, or `send_sample_on_pileup`), one packet is assembled according to Table 7.6. Each activated channel increases the dead time of the CB-SADC by at least

$$\frac{1040\,\text{byte} + 64\,\text{byte}}{1\,\text{Gbit/s}} = 8.832\,\mu\text{s} \tag{7.26}$$

for the bare transmission time of the sample packet content (1040 byte) and the UDP/IP and Ethernet Type II protocol overhead (64 byte). If all channels are activated, this sums up to $32 \cdot 8.832\,\mu\text{s} \approx 282.6\,\mu\text{s}$. Furthermore, the time to send the feature extraction packet and to save the waveform to block RAM has to be taken into account. Sending the feature extraction packet takes $10\,\mu\text{s} + 3.328\,\mu\text{s}$ (cf. eq. (7.25)). During this time the waveform is already being stored, which in total takes at least 25.6 µs. The exact time depends on the decimation (here 20 MS/s is assumed), the configured window position and the window stretcher (see previous paragraphs)). This limits the theoretically possible trigger rate to at most

$$\frac{1}{32 \cdot 8.832\,\mu\text{s} + 25.6\,\mu\text{s}} \approx 3.24\,\text{kHz}. \tag{7.27}$$

As described already for the feature extraction packet in section 7.5.6, this is only attainable under otherwise ideal conditions and without the *burst* mechanism.

| **0-7** | **8-15** | **16-23** | **24-31** |
|---|---|---|---|
| 0xCB | 0xCB | 0xCB | 0xCB |
| BUF_COUNT | TYPE | BUF_NR | BUF_TOTAL |
| counter_event | | | |
| daq_bufnr | channel | part | - |
| $512 \cdot 2\,\text{byte}$ sample | | | |

Table 7.6.: Content of the 1040 byte sample extraction packet, consisting of the *Burst Header* (cyan), the event information (magenta), and the sample data.

**Improvements** The firmware can, in principle, be configured to support longer sampling windows. The MTU of 1480 byte, minus the sample extraction header of 16 byte, would allow to increase the sample window size from 1024 byte to 1464 byte or 36.6 µs. Additionally there is a possibility to split the sample window to more than one network packet, or to use *jumbo frames* to contain much larger windows in one packet. While this would save some transmission time due to a higher payload to overhead ratio, a much higher rate can be reached by decimating the data further (or skipping sample points) without extending the sampling window length, hence reducing the payload. This should be viable down to a decimated sampling rate of $\approx 5$ MS/s without losing information.

## 7.7. Trigger

The trigger/sync system of the CBELSA/TAPS experiment was introduced in section 2.9.4. Apart from this trigger used during the experiment, which is called the "External Trigger" in this section, there are others ways to *trigger* the CB-SADC. All trigger possibilities will be briefly introduced in the following sections.

### 7.7.1. Self Trigger

The CB-SADC can be configured to generate a trigger signal internally from the waveform data. For this purpose a leading edge discriminator is instantiated in the firmware for every channel. The input of the discriminator is an eight-fold simple moving average of the waveform data, and it is compared to the last determined baseline from the previous trigger. The threshold of the discrimination can be set using `config.channels(i).led.threshold`, which will eventually trigger on the noise in the data at lower settings. To enable the self trigger, it is necessary to configure a 32-bit mask `trigger_int_or_bitmask`, where the N-th bit will enable (1) or disable (0) the trigger for channel N. The self trigger can, for example, be activated for channels 24 to 31 by setting `trigger_int_or_bitmask=0xFF000000`. The internal trigger is useful for quick testing in the laboratory, or for using the CB-SADC in a standalone setup that does not supply a central trigger. Or, it is possible to use the constant fraction discriminator's signal as an internal trigger source; but since the CFD algorithm is optimized for generation of precise timestamps, it may not be suitable for the lowest thresholds. Moreover, it is not versatile for laboratory tests, since its parameters are tuned to the exact shape of the CBELSA/TAPS experiment's (shaped) waveforms.

### 7.7.2. External Trigger

The external trigger is being used in the CBELSA/TAPS experiment as described in detail in sections 2.9 and 2.9.4. Physically this signal is connected to the CB-SADCs through the Backplane (in the future: the CB-SADC Crate Controller), shown in section 6.4.2. Through the trigger/sync system it is ensured that the CB-SADC will stay synchronous to all other components involved in the readout of the experiment. To set the experiment trigger, the 8-bit mask `trigger_ext_bitmask` has to be configured to a value of `0x02`.

### 7.7.3. Network Trigger

It is possible to issue a single trigger impulse by sending a specific network packet to the CB-SADC, activating the `trigger_network` bit. This can be useful for debugging purposes. To configure it, the `trigger_ext_bitmask` has to be set to a value of `0x40`, after which the network trigger packets will be taken into account.

### 7.7.4. Clock Trigger

It can be useful for testing, debugging, or determination of the baseline, to use a so-called clock trigger. A clock trigger will generate trigger signals with a fixed frequency, uncorrelated to the waveform data. The 16-bit parameter `trigger_clock` can be configured accordingly, and its value represents the trigger rate in Hz, ranging from 0 Hz to 65 535 Hz.

## 7.8. Further Firmware Development

The previous sections described the firmware development for the CB-SADC (and the previous 16-channel PANDA-SADC prototype). In addition, it was necessary to develop a firmware for the Backplane, as well as for the ELB-VME-VFB, which was used in the CB-SADC rack as a trigger/sync master and slowcontrol master. Both will be described briefly in the next sections.

### 7.8.1. Backplane Firmware

The SPARTAN-6 on the Backplane (cf. section 6.4) does not have any demanding tasks. The *Event* (trigger) and *Sysreset* signals, which it receives from the experiment's central trigger/sync system (cf. section 2.9.4), are presently distributed to up to four connected CB-SADCs (asynchronously for the best time resolution). The *Busy* and *OK* signals from the CB-SADCs are `OR`'d and then returned to the central trigger.

Apart from that the main task of the Backplane firmware is to provide a diagnosis interface through Integrated Logic Analyzers (ILAs) and Virtual Input/Outputs (VIOs) (cf. appendix B.3). This allows to inspect all signals of the trigger/sync system (*Event*, *Sysreset*, *Busy*, *OK*, and the *Buffer Number*). The diagnostic interface can be seen in Figure 7.24. It is possible for the Backplane to simulate the *Buffer Number* if the *Sync Bus* flat-ribbon cable is not connected, it will then be generated by the SPARTAN-6 and increased with every trigger. Furthermore, since it was suspected at the time that the trigger/sync signals may interfere with the programming procedure through JTAG, the trigger/sync signals may be inhibited during programming.

**Identification of a Trigger Crosstalk Problem** The debugging possibilities of the Backplane have proven helpful for the identification of a fatal problem. It has been observed that the CB-SADCs behaved erratically while working in conjunction with the DAQ, in such a way that the data would quickly become *out of sync*. The effect presented itself by feature extraction packets that seemingly originated from the same event, but would not have a matching *Buffer Number*. It was suspected that this was due to either missed or superfluous trigger signals.

The debug interface has been used to investigate this situation, and suspicious events were recorded, as shown exemplary in Figure 7.25. What can be seen is that, almost immediately after the *Buffer Number* changes, at least one of the CB-SADCs is issuing the *Busy* signal. Note that his can only happen if the CB-SADC has seen a trigger signal, which evidently is not present on the Backplane and could not have been sent by it. The only possible explanation is that somewhere between the SPARTAN-6 on the Backplane, and the KINTEX-7 on the CB-SADC, enough charge was induced on the trace connecting the *Event* signal such that it was interpreted by the KINTEX-7, or buffering ICs in the path on the PCB, as a high level. The sources of this induction/crosstalk are obviously the *Buffer Number* traces. Indeed the effect was only observed when the *Buffer Number* was changing from `0x1F` to `0x00`, hence all lines were toggling from *high* to *low* at the same time.

The problem could be solved by configuring the output drivers of the SPARTAN-6 that connect the *Buffer Number* towards the CB-SADCs, as `SLEW = QUIETIO`. In this setting, the slew rate is as low as possible (still more than fast enough for this use case), limiting the higher frequency components originating from "sharp edges" of a fast signal transition.

Since the high current necessary for fast slew rates is responsible for the transmission of the electromagnetic fields causing the crosstalk, it was possible to avoid the problem with this workaround. Yet, the problem definitely had to be addressed on the level of trace routing on the PCB. This was one of the motivations to changing the routing of the trigger connections on the CB-SADC as discussed in section 5.7.4.

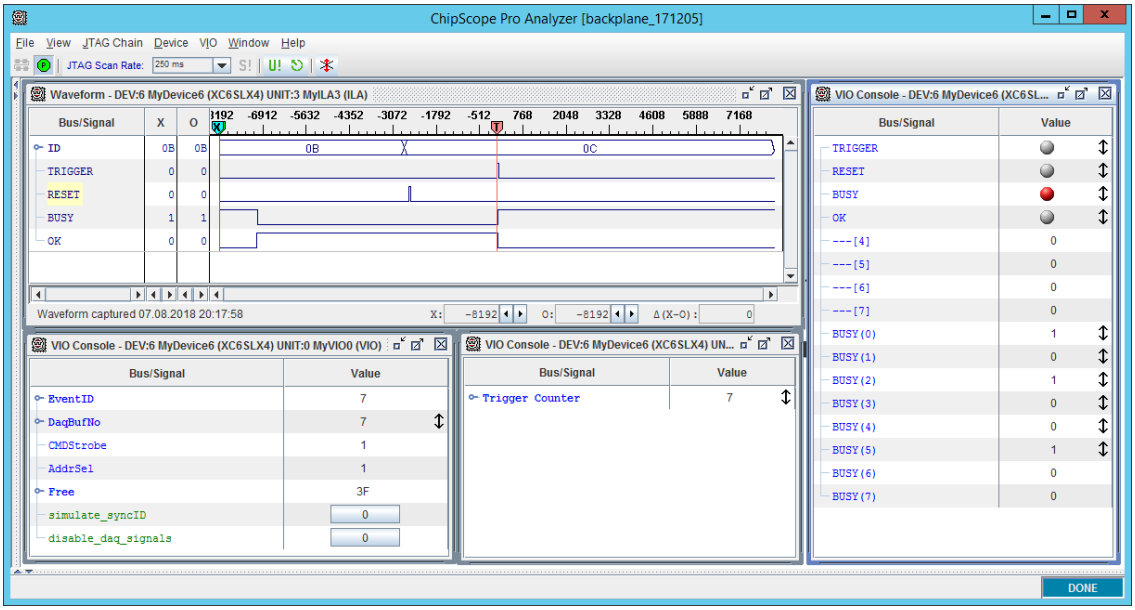



Figure 7.24.: Overview of the debug signals offered by the CB-SADC Backplane, viewed with ChipScope Pro Analyzer. The **top left** shows the recorded trace of a triggered event, one can see the delay of the DAQ between the release of the *Busy* signal and the occurrence of the next trigger. On the **right**, a *live view* of the trigger/sync signals and the respective *Busy* signals of the CB-SADCs' FPGAs, can be seen. The **bottom left** shows the signals of the *Sync Bus*, and allows to turn on the simulation of the *Buffer Number*, or to disconnect the trigger/sync signals from the CB-SADCs altogether.

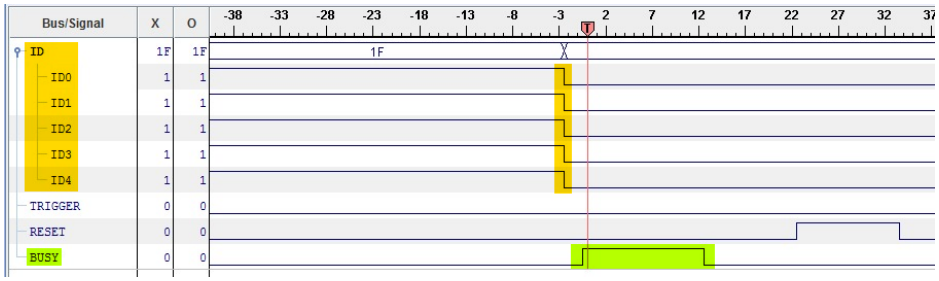



Figure 7.25.: Identification of crosstalk in a detailed view similar to the top left of Figure 7.24. Although no trigger signal is visible, the *Busy* signal (green) is getting active right after the *Buffer Number* (yellow) is changing, indicating a charge induction from the *Buffer Number* traces to the *Trigger* trace.

### 7.8.2. ELB-VME-VFB Slowcontrol/Trigger Firmware

The so-called VME-VFB, a versatile FPGA module for the VME bus from the company ELB [93], is used in the CB-SADC rack for two purposes:

1. $I^2C$ master for the CB-SADC slowcontrol for access to the sensor data and configuration, when the CB-SADC is not used in the experimental hall
2. trigger/sync master for evaluation of the compatibility with the CBELSA/TAPS experiment's DAQ.

The integration into the CB-SADC rack is described in section 9.1.3, also showing a picture of the assembly. The following paragraphs will provide a brief summary.

**$I^2C$ Master** The $I^2C$ implementation is based on the work done within [88], which uses an open source implementation for the $I^2C$ controller [185]. This controller is extended by a higher level of abstraction, such that generic $I^2C$ commands can be issued without too much offload and error-proneness on the user side. Since the ELB-VME-VFB does not offer a network connection directly, all functions are accessed through the VME interface. The necessary registers are mapped to the PCI address space of the CPU in the VME crate, such that it can finally be accessed by the slowcontrol libraries. A summary of the software development can be found in section 8.5.

**trigger/sync Master** A logic circuit mimicking the behavior of the trigger/sync system of the CBELSA/TAPS experiment (cf. section 2.9.4) was implemented as shown in Figure 7.26. The trigger can be sourced either from a NIM input (LEMO) on the ELB-VME-VFB, or from an internal clock generator with adjustable frequency. As soon as the CB-SADCs no longer issue the *Busy* signal, a 100 ns *Sysreset* pulse is sent, and the trigger gate is opened. When a valid trigger signal arrives from the selected source, a 100 ns *Event* signal is generated and the trigger gate is closed.

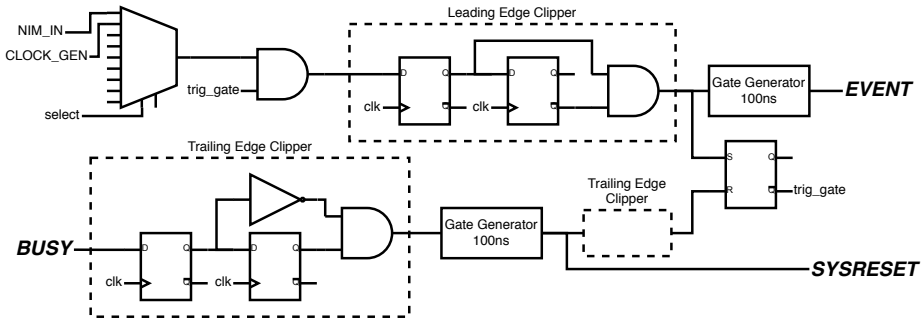



Figure 7.26.: Logic implementation of the trigger/sync Master module in the ELB-VME-VFB. The source selection is realized with a 2-bit muxer (top left). Edge clippers using flip-flops are instantiated to detect the leading edge of the trigger signal and the trailing edge of the *Busy* signal. Gate generators ensure a minimum length of 100 ns for the *Event* and *Sysreset*.

This section concludes the firmware development chapter of the CB-SADC and the associated projects. The following chapter will present the software development.

# 8. Software Development

After the description of the hardware development in chapters 5 and 6, and the firmware development in chapter 7, this chapter will present the associated software development.

Section 8.1 will present a tool with a graphical user interface that can receive the network packets sent by the CB-SADC, and that can also display the waveforms. Section 8.2 briefly introduces a PYTHON3 framework, allowing to configure the CB-SADCs, record feature extraction and sample extraction packets, and analyze and display the waveforms. Section 8.3 will describe two tools with a command line interface that can be used to receive and store the feature extraction and sample extraction packets, and to configure the CB-SADC. The LEvB, required for integrating the CB-SADCs into the experiment's DAQ, is shown in section 8.4. Finally, in section 8.5 it will be discussed how the CB-SADC was integrated in the CBELSA/TAPS experiment's slowcontrol system.

## 8.1. Qt Debugging Tool

The QT DEBUGGING TOOL, written for LINUX, was the first software able to receive and display the CB-SADC waveforms, and to allow basic configuration of the CB-SADC's operational parameters. Before its existence, the sample extraction packets were validated only based on the raw network packet content, much like what is shown in Figure 4.2 on page 65; alternatively, the data was converted by hand to plot a single snapshot of the waveform. With the QT DEBUGGING TOOL, an oscilloscope-like operation allowed a true visualization of the (sample extraction) data, which has proven helpful for the development of the firmware and the laboratory test of the CB-SADC and Analog Input Card. The following sections will first present the functionality of the tool in operation, and then provide some information and examples of the code.

### 8.1.1. In Operation

The tool can connect to a CB-SADC by configuring the IP address and the UDP port accordingly. Two different tabs can be opened to view the waveform: the first one allows to examine the signal from a single channel and configure the trigger conditions (shown in Figure 8.1). The following parameters are accessible for configuration: `trigger_ext_bitmask`, `trigger_int_bitmask`, `send_sample`, `led.threshold`, `cfd.threshold`, `sample_position` (cf. appendix C). With the last parameter, the waveform can be shifted with regard to the trigger time to the *left* or *right*. Furthermore, it is possible to either automatically scale the amplitude axis, or to fix it manually, which is helpful for optically (qualitatively) investigating the noise level. If the second tab is selected (see Figure 8.2), the CB-SADC is automatically configured to send samples for every channel (while the trigger condition will stay unmodified). This view provides a quick overview over all channels with automatically scaling graphs, allowing to judge changes in signal content immediately.

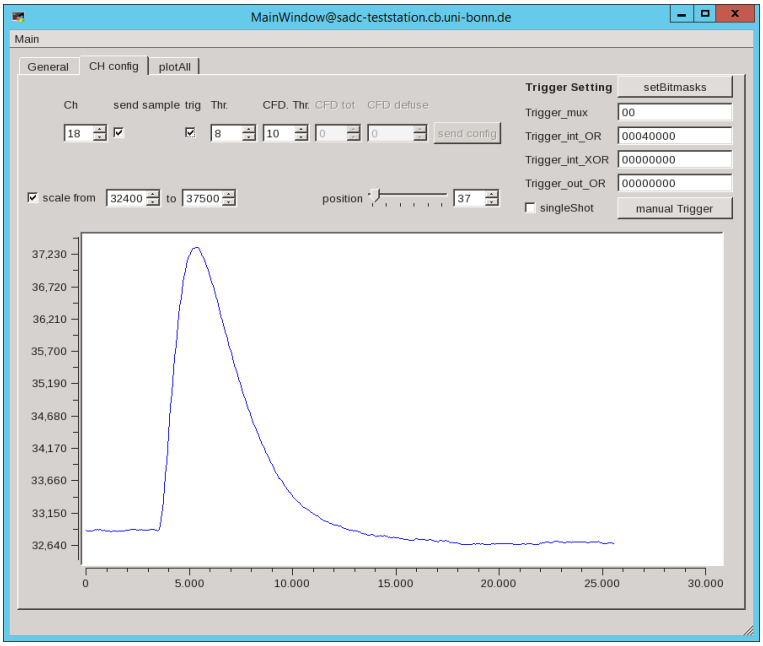



Figure 8.1.: Display of a waveforms on channel 18, originating from an arbitrary waveform generator. An undershoot is visible since the pulse shape and the pole-zero cancellation were not matched yet. In this tab, some of the CB-SADC's parameters can be configured. In this instance, the trigger bit mask of `0x00040000` shows that channel 18 is also used in the internal OR trigger, with a threshold (*Thr*) of 8. The x-axis comprises 512 samples, the y-axis shows the amplitude expressed in 16-bit raw values. The auto-scaling is deactivated.

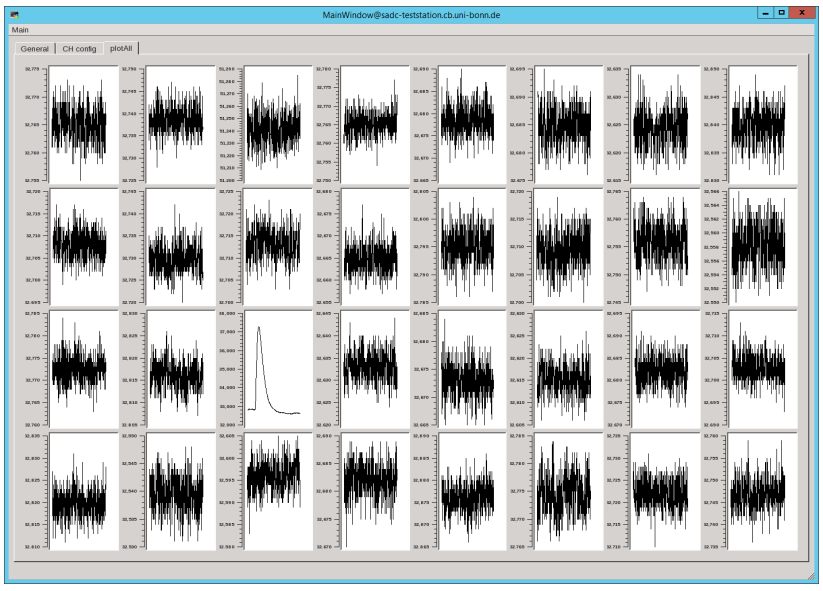



Figure 8.2.: Display of waveforms from 32 channels, with the same signal visible on channel 18, and noise (*baseline*) on other channels. The x-axis comprises 512 samples, the y-axis, which scales individually for all channels, shows the amplitude.

### 8.1.2. Code

The Qt Debugging Tool has been developed mid 2016 in the Qt5 framework with Eclipse and QtCreator for the graphical user interface. For the display of the waveform data, the QwtPlot class from the Qwt framework[1] was used.

The central function responsible for receiving the UDP datagrams is shown in Listing E.4 in the appendix. Depending whether the datagrams contain feature extraction or sample extraction packets, the data is being processed by different functions, `addFEPacket` or `addSEPacket`, which in turn extract the data using functions defined in Listings E.5 and E.6. The sample data, provided by `extractSample()`, is then used in the QwtPlot framework to visualize the data. Although functions to process the feature extraction data were indeed implemented for future purposes, the Qt Debugging Tool in fact does not use them, and processes only the sample extraction data to visualize the waveforms.

The Qt Debugging Tool has proven to be a useful tool to examine waveforms during operation, but is limited to Linux machines (whereas the laboratory environment was mostly Windows based), and the maintenance of the code and expansion of the features were quite tedious. Motivated by this, a platform-independent Python3 framework was developed for the tests of the final CB-SADC revision. It will be presented in the following section.

## 8.2. Python3 Tool Set

Within the master thesis of Y.-C- Wang [140], a set of Python3 scripts was developed for the test of the CB-SADCs. More specifically, the framework's goal is to test the signal processing of the Analog Input Card (shaping, baseline-shifting, pole-zero cancellation) in a standalone environment. This entails that it is platform independent and works with direct connection to a CB-SADC, without requiring any other software/hardware. The complete framework, including the shared libraries, is explained in more detail in [140] and was briefly introduced in section 6.2.1. Based on this framework, three standalone tools have been condensed in the scope of this thesis to aid everyday laboratory tests:

**Sensor Readout** to access all slowcontrol sensors on the CB-SADC, the power supply, and the Analog Input Card, through an Arduino-based USB-to-$I^2C$ interface,

**Waveform Recorder** to configure the CB-SADC through UDP packets and receive the sample extraction and feature extraction packets, and record them to disk,

**Waveform Display** to display the content of recorded sample extraction packets and calculate and visualize the baseline, the noise, the amplitude, and the peak position.

Listing E.7 shows a minimal example for the waveform recorder. A more elaborate GUI was developed in the scope of a A. Sonnenschein's bachelor's thesis for the PANDA project [186], with the goal to offer a live view like the Qt Debugging Tool.

The Python3 libraries and tools are hosted together with the CB-SADC's firmware in a Gitlab archive (`https://agthoma.hiskp.uni-bonn.de/gitlab/CB/vhdl/CB-SADC64`). For more elaborate tests that focus on the network performance and the combination of multiple CB-SADCs, a command line interface tool was developed, which will be introduced in the next section.

[1] `http://qwt.sourceforge.net/`

## 8.3. Receiver and Configuration Tools

To test the performance of the burst mechanism (see section 7.4.5) and to examine the synchronization between multiple CB-SADCs, separate tools were necessary. Those command line interface (CLI) tools, called `sadc64-receiver` and `sadc64-config`, have been written by C. Schmidt. The *libraries/classes* used in the tools are the same libraries which are now used in the LEvB (see next section), so the tools were also important to test the libraries before using them in the experiment in conjunction with the DAQ.

`sadc64-receiver` This tool is responsible for receiving and decoding the feature extraction and sample extraction packets from one or more CB-SADCs. In contrast to the Qt Debugging Tool, it is not meant to display the waveforms. Instead, it analyzes the content of the *Burst Header* (Table 7.4 on page 159) and the *Event Header* (Table 7.5). The headers of all received packets are inspected continually and are evaluated in the following regards:

- Were all 32 packets of one *burst* received?
- Were the event IDs (`counter_event`) received in sequence without IDs missing?
- Did the `daq_bufnr` of the trigger/sync system match for all packets of one event?
- Were all sample extraction packets, which are expected according to the `send_sample` bit mask in the feature extraction packet, indeed received?

If one of the conditions is not fulfilled, packets can either be requested again from the CB-SADC that keeps them stored in a buffer, or the readout will be stopped.

Figure 8.3 shows the `sadc64-receiver` in operation while it ran on a Linux system connected to multiple CB-SADCs in the CB-SADC rack. Figures 8.3a and 8.3b compare the performance with sample extraction packets disabled or enabled, with one CB-SADC connected, using the *burst* mechanism. The CB-SADC is configured to use a self trigger on all channels (`trigger_int_bitmask=0xFFFFFFFF`) with the threshold set to the lowest setting, such that it is below the noise level (`led.threshold=1`). In essence, this means that a new trigger is issued almost immediately when the CB-SADC is no longer busy from the previous event. Without the sample extraction packets, a trigger rate of more than 35 kHz could be achieved. This is neither limited by the data rate, which theoretically has a limit of 125 Mbyte/s, nor by the packet rate, comparing it to Figure 8.3b. Comparing it with equation 7.25 on page 170 shows that less than half the theoretical maximum is reached. This discrepancy is expected and caused by the *burst* mechanism, it will be investigated further in section 9.9. Preliminarily it can be stated that the readout rate—since it is still one magnitude above the typical readout rates currently achieved by the CBELSA/TAPS experiment—seems sufficient. The matter will be investigated in more detail in section 9.9.

If the sample extraction packets are activated, the event rate drops to around 2.77 kHz, which is only 14.5 % below the theoretical maximum (equation 7.27 on page 172). This shows that, regarding only the limitations of the CB-SADC firmware, the CB-SADC could keep up with the current readout rates in the experiment, even with full waveform readout.

Figure 8.3c can be examined to explain some of the other information provided by the `sadc64-receiver`, which are helpful in case of a malfunction. Six CB-SADCs are connected and have, at the time of the screenshot, transmitted more than 100 000 000 events. As can be

```
                              received    received    received   data rate  packet rate event rate
            SADC-IP             bytes      packets      events     [MB/s]       [kHz]       [kHz]      status
-------------------------------------------------------------------------------------------------------------
        192.168.0.11          2619656096    11272768     6989906     11.783      35.072      35.058
-------------------------------------------------------------------------------------------------------------
Total                          2498.30MB     11.27M                 11.783      35.072
         events: 6990897, Msg:

            SADC-IP             events    complete        ---       retry     timeout
-------------------------------------------------------------------------------------------------------------
        192.168.0.11            6990898      352305          0           0          76
```

(a) One FPGA is connected, configured in self-trigger with low threshold. No samples are sent (just feature extraction packets). An event rate of above 35 kHz is reached.

```
                              received    received    received   data rate  packet rate event rate
            SADC-IP             bytes      packets      events     [MB/s]       [kHz]       [kHz]      status
-------------------------------------------------------------------------------------------------------------
        192.168.0.11           686272752     673377       20406     88.971      91.540       2.774
-------------------------------------------------------------------------------------------------------------
Total                          654.48MB      0.67M                 88.971      91.540
         events:  20409, Msg:

            SADC-IP             events    complete        ---       retry     timeout
-------------------------------------------------------------------------------------------------------------
        192.168.0.11              20410       21047          0           0          76
```

(b) Samples from all channels are activated, the data rate increases from $\approx 11.8$ Mbyte/s to $\approx 89.0$ Mbyte/s while the event rate drops to below 3 kHz.

```
                              received    received    received   data rate  packet rate event rate
            SADC-IP             bytes      packets      events     [MB/s]       [kHz]       [kHz]      status
-------------------------------------------------------------------------------------------------------------
        192.168.0.12          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(8fffffff) 100035385
        192.168.0.13          263511040  3722240160  3722240160       0.000       0.000       0.000  RT(7fffffff) 99458950
        192.168.0.14          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(7fffffff) 115315274
        192.168.0.15          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(3fffffff) 115315274
        192.168.0.16          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(    1fff) 115315274
        192.168.0.17          263556448  3722240289  3722240289       0.000       0.000       0.000  RT( 3ffffff) 115315274
        192.168.0.18          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(  7fffff) 115315274
        192.168.0.19          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(  1fffff) 115315274
        192.168.0.20          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(  1fffff) 115315274
        192.168.0.21          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(  1bffff) 115315274
        192.168.0.22          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(   7ffff) 115315274
        192.168.0.23          263556448  3722240289  3722240289       0.000       0.000       0.000  RT(   37fff) 115315274
-------------------------------------------------------------------------------------------------------------
Total                          3016.12MB  44666.88M                  0.000       0.000
         events: -572727137, Msg:

            SADC-IP             events    complete        ---       retry     timeout
-------------------------------------------------------------------------------------------------------------
        192.168.0.12          3722240210   116320009          0           6           0
        192.168.0.13          3722240160   116320005          0         128       11667
        192.168.0.14          3722240210   116320009          0           6           0
        192.168.0.15          3722240210   116320009          0           6           0
        192.168.0.16          3722240210   116320009          0           6           0
        192.168.0.17          3722240210   116320009          0           4           0
        192.168.0.18          3722240210   116320009          0           5           0
        192.168.0.19          3722240210   116320009          0           5           0
        192.168.0.20          3722240210   116320009          0          10           0
        192.168.0.21          3722240210   116320009          0           7           0
        192.168.0.22          3722240210   116320009          0           5           0
        192.168.0.23          3722240210   116320009          0           3           0
```

(c) Multiple CB-SADCs connected to the receiver, which has stopped because of an error. The lower section reveals the problem: One of the CB-SADC's FPGA with IP `192.168.0.13` has stopped sending packets, which can be told from the high timeout number.

Figure 8.3.: Screenshot of the `sadc-receiver` in operation. The upper parts of each screenshot displays the network statistics and the event (trigger) rate, and additionally different status messages. The lower parts shows the counter for successfully assembled events, and also counters for `retry` and `timeout`. One connected SADC-IP represents half of a CB-SADC (each of the two FPGAs has an independent network interface). The `retry` count shows that every once in a while the UDP packets from the CB-SADCs did not reach the server (possibly due to network congestion), and the reliability protocol was used to request the packets again.

noticed, the readout is halted: all rates have dropped to zero, although at the time the trigger condition was still fulfilled and the readout should be running. It can be seen in the lower section that the `timeout` counter of IP `192.168.0.13` has increased. This means that the `sadc64-receiver` is expecting to receive packets, but the regarding FPGA of the CB-SADC has become silent. At the time this was most likely caused by bug in the firmware that did cause this condition. It can also be seen that the *burst* mechanism is at work by looking at the `status` notifications in the upper right section: the last row, for example, shows the entry `RT( 37fff)`. `RT` stands for ReTry, while the hex number in the brackets corresponds to the bit mask of the `CMD_RESEND_BUFFER` packet shown in Table C.3 on page 237, followed by the event counter. Hence, 17 packets from event number 115315274 where not received and had to be requested again from the CB-SADC. Interestingly, almost all IPs seemed to be affected in this event, which likely points out that there was a problem of some sort on the computer running the `sadc64-receiver`, which resulted in a delay in processing the network packets, and they were dropped from the buffers of the network interface.

Observations like those have helped greatly to identify problems in the firmware, but also to notice difficulties with the network hardware and the configuration of the Network Interface Card of the receiving server, which could then be fixed.

`sadc64-config` This tool is used to send configurations (shown in Table C.1) to the CB-SADC's FPGAs via UDP packets. The available parameters for execution of the tool are shown in Listing E.8 in the appendix. The libraries behind this tool are also used in conjunction with the LEvB and DAQ for the automated configuration of the CB-SADCs.

## 8.4. Local Event Builder

The concept of the LEvB was introduced as part of the DAQ in section 2.10, and is explained in detail in section 7.2 of the dissertation of C. Schmidt [100]. The LEvB for the CB-SADC, which was developed by C. Schmidt, performs similar consistency and synchronicity checks as the `sadc64-receiver` after receiving the UDP packets from the CB-SADCs; but instead of discarding or dumping the data (as the `sadc64-receiver` does), the feature extraction data is re-packed into the `ZEBRA` file format that is used throughout the whole CBELSA/TAPS experiment, and then transferred to the `Event Saver` (`saver3`). The `Event Saver` compares the *Buffer Number* of each event to the expected value to ensure synchronicity in the readout chain (cf. section 2.9.3). The sample extraction data is in the current implementation, due to its size, not processed by the *Event Saver*, but instead stored locally on the `cbsadcserver`, which is running the LEvB. It is planned to upgrade the network architecture of the DAQ such that in the future all data can be processed and stored together.

The configuration of the CB-SADC during the experiment does not happen manually with `sadc64-config`; instead, the LEvB loads the configuration options from an `xml` file at the start of every run, and send the according configuration packets to the CB-SADCs automatically. An example for such an `xml` configuration file is shown in Listing E.9.

The following two sections introduce compression algorithms for the sample extraction and feature extraction data, developed and integrated into the LEvB by J. Hartmann [187]. Both are planned to be integrated into the CB-SADC firmware in the future (to the extent of the FPGAs' capability and capacity), such that the additional load of the LEvBs can be reduced.

### 8.4.1. Waveform Compression

An algorithm named `wavezip` compresses the waveforms, using the following assumption: due to a limited slew rate, successive samples differ only by a limited number of bits, thus after the initial sample, only the difference to the next sample is transmitted. After encoding those values with Ibsen code [188], an even better compression is obtained. Since the compression is lossless, all information is retained in the process. A factor of 3 to 4 is feasible, and it was shown in a test measurement that with a full waveform readout of six CB-SADC prototypes, the event rate of roughly 1.3 kHz did not decrease if compression was performed during runtime (despite the not-too-modern CPUs of the current `cbsadcserver`). This test helps to decide on the necessary performance of the final `cbsadcserver` hardware.

### 8.4.2. Zero Suppression

In the last beam times, the CB-SADC's feature extraction packet was shown to be responsible for about 50 % of the data traffic towards the event saver server `saver3`. This is due to a current drawback in the firmware: the feature extraction packet is fixed in size, and information for each channel is provided, even if only an insignificant energy deposit was detected. Since most of the calorimeter crystals do not exhibit a signal distinguishable from electronic noise, it makes sense to discard the information based on a threshold applied to the integral or maximum feature. An algorithm automatically discarding negligible information is (in this context) referred to as *zero suppression*, and is used for the QDC readout. For the CB-SADCs, it has been successfully implemented on the LEvB level, meaning that all data is transferred from the CB-SADCs to their LEvBs, but greatly reduced from the LEvBs to the event saver.

## 8.5. Slowcontrol

The $I^2C$ based slowcontrol of the CBELSA/TAPS experiment, which was described in section 2.11.2, has been extended to include the CB-SADC into its sensor readout and control. Before the implementation was started, a *standalone* solution has been programmed to access all of the CB-SADC's $I^2C$ bus components in conjunction with the CB-SADC rack outside of the CBELSA/TAPS experiment's environment. Both will be presented briefly in the following sections.

### 8.5.1. Standalone Readout Tool

The CB-SADC rack, which will be presented in section 9.1, includes an FPGA-based $I^2C$ slowcontrol readout using the $I^2C$ extension card presented in section 6.5. A tool has been written in `C++` with the NCURSES GUI, allowing for a comfortable readout of the sensors on the CB-SADCs, as well as for a control over their power supplies and the Analog Input Cards. The sensor readout is depicted in Figure 8.4. The tool was used for a long time for tests in the laboratory, for example for the development of heatsinks as shown in section 5.7.6, but also during the commissioning beam time in the experimental hall, when the integration into the $I^2C$ daemon of the CBELSA/TAPS experiment was not yet completed.

| # | ID | FPGA1 | FPGA2 | LMK | MAX | BLV1 | BLV2 | PS | U6 | I6 | U12 | I12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | d57597 | 46.438 | 43.562 | 48.625 | 42.688 | 38.188 | 38.562 | 48.69 | 5.87 | 3.37 | 11.969 | 0.18 |
| 2 | d52ee1 | 54.812 | 50.938 | 58.062 | 51.438 | 41.062 | 39.250 | 50.88 | 5.88 | 3.41 | 11.985 | 0.17 |
| 3 | d51057 | 54.188 | 51.188 | 58.312 | 51.188 | 38.750 | 39.250 | 45.62 | 5.88 | 3.50 | 11.977 | 0.17 |
| 4 | d52064 | 45.250 | 43.312 | 49.062 | 41.812 | 37.438 | 38.062 | 41.44 | 5.87 | 3.57 | 11.981 | 0.17 |

Figure 8.4.: Sensor readout of the Standalone Readout Tool. Four CB-SADCs are being read out. After their unique ID, the temperature of four sensors on the CB-SADC are shown (two for the FPGA, one for the PLL LMK04806, one for the linear regulators MAX8556), followed by temperatures on the two Analog Input Cards (BLV) and the power supply (PS). The voltage and current on the primary side of the power supply (6 V and 12 V from the NIM supply) are shown. The notable differences in temperature are caused by tests of various heatsinks and the varying air distribution in the NIM crate.

### 8.5.2. Integration into the I²C Daemon

Before the first production beam time in 2018, six CB-SADCs were moved from the CB-SADC rack into the readout racks close to the Crystal Barrel Calorimeter (see Figure A.2). In this occasion they have been integrated into the central I²C slowcontrol readout of the CBELSA/TAPS experiment. Four steps were necessary for the full integration:

1. Implementation of CB-SADC specific commands into the library `i2cd_thread.cpp`
2. Implementation of the TELNET commands into the daemon `i2cd_server.cpp`
3. Automated readout and storage of sensor data in the SQL-database
4. Extension of the slowcontrol web page to display the sensor data

The first step was already mostly completed during the development of the Standalone Readout Tool, such that the functions only needed to be ported to different libraries. The implemented functions are shown in Listing E.10 (page 258), and the associated TELNET commands are shown in Table 8.1. The commands can be entered after connecting to the slowcontrol's TELNET server at `cbvxt1.cbhardware.hiskp` on port 4442. Integrating the CB-SADCs' sensor readout into the automated readout of all other sensors, which happens at regular intervals ($\approx 10\,\text{min}$), as well as the integration into the slowcontrol web page, is currently an open task.

| | |
|---|---|
| `SADC [N] ON` | `SADC [N] GETPZC [CH]` |
| `SADC [N] OFF` | `SADC [N] GETTEMP-FPGA1` |
| `SADC [N] GETID` | `SADC [N] GETTEMP-FPGA2` |
| `SADC [N] GET6V` | `SADC [N] GETTEMP-PLL` |
| `SADC [N] GET12V` | `SADC [N] GETTEMP-LDO` |
| `SADC [N] SETBASELINE [CH] [VALUE]` | `SADC [N] GETTEMP-BLV1` |
| `SADC [N] GETBASELINE [CH]` | `SADC [N] GETTEMP-BLV2` |
| `SADC [N] SETPZC [CH] [VALUE]` | `SADC [N] GETTEMP-PS` |

Table 8.1.: Telnet commands for the CB-SADC slowcontrol.

# 9. Measurements Performed with the CB-SADC

After *hardware*, *firmware*, and *software* have been treated in the last chapters, this chapter will show the CB-SADCs *in action*. Although at the end of this thesis the full CB-SADC readout of the Crystal Barrel Calorimeter was not fully installed yet, it was possible to prove its good performance in several scenarios. In the beginning, the CB-SADCs were thoroughly tested in the laboratory. Since spring 2017 six prototypes were installed in the experimental hall and were integrated into the data-taking in various beam times.

Before the measurements are presented, the CB-SADC rack, which was the environment for many tests in the laboratory and the experimental hall in different versions, will be discussed in section 9.1.

The following sections will be presented more or less in chronological order. In section 9.2, the results from the master thesis of J. Schultes [122] will be presented, who investigated the resolution and linearity of the first prototype. It will be shown that the resolution of the CB-SADC outperforms the QDC without requiring a dual-range setup. The first measurement taken together with the Crystal Barrel Calorimeter, comparing the analyses of minimum ionizing particles with CB-SADC and QDC, will be presented in section 9.3. The readout of the *Energy Sum* signal with the CB-SADC, which was recorded for analysis in S. Ciupka's master thesis [91], will be shown in section 9.4. In section 9.5, measurements of the noise under different conditions will be shown and compared to the theoretical expectation and requirements of the experiment, and further possible improvements will be discussed.

The most important measurement, and final benchmark for the CB-SADC prototypes, was the recording of *features* and waveforms during the first production beam times after the APD upgrade of the Crystal Barrel Calorimeter. The results and a comparison to the QDC readout will be shown in section 9.7. After the evaluation of the performance with regard to the energy information, the performance of the timing determination will be presented in section 9.8. Lastly, section 9.9 will discuss the deadtime and efficiency of the CB-SADC's readout chain including the Crystal Barrel DAQ and compare it to the expectations presented throughout this thesis.

## 9.1. The CB-SADC Rack

The CB-SADC rack was built as a complete standalone environment to operate one to twelve CB-SADC prototypes in a variety of scenarios. Prior to its existence, the prototype was operated only in the laboratory, directly connected to a Windows or Linux computer, which was sufficient for the early phases of firmware development. But, already with the tests performed in the scope of [122], which required to move the setup to another laboratory with different infrastructure, the necessity to have an independent test setup arose.

The central components of the CB-SADC rack are a NIM crate to host the CB-SADCs, and a Linux server. The server will be described in more detail in section 9.1.1. In the first revision, everything was hosted in a Pentair SlimLine 12HE rack that can be seen in Figure A.1, but as it was necessary to add VME infrastructure, it was soon moved to a Pentair HeavyDuty 16HE rack. The final setup can be seen in Figure 9.1. The server, the network infrastructure, and the VME components will be explained in the following sections.

(a) Front, showing six CB-SADCs in the NIM crate, the VME crate, a drawer, and the `cbsadcserver`.

(b) Back, additionally showing the two Ethernet switches and the power distribution.

Figure 9.1.: Picture of the second revision of the CB-SADC rack. In the front, from top to bottom: The NIM crate with six CB-SADC prototypes on the right, a network-to-USB converter on the left; the horizontal VME crate with VME CPU and Elb-Vme-Vfb board with slowcontrol and trigger interface; a drawer; and the `cbsadcserver`. The cabling consists of two yellow *trigger* cables, two magenta *slowcontrol* cables, the flat ribbon *sync* cable (which connect to the two Backplanes), and a blue network cable for integration of two USB-JTAG programmers into the network. In the back, two switches are visible, the upper one (Huawei S5720S-28X) dedicated for the communication with the CB-SADCs, the lower one (3COM 2920-SFP Plus) for regular network access. From the Huawei switch, two 10 Gbit/s cables are connected to the `cbsadcserver`, and twelve regular network cables to the CB-SADCs. Not visible are the two Backplane prototypes, which each connect to three of the CB-SADCs.

### 9.1.1. CB-SADC-Server

The `cbsadcserver` has been chosen specifically to evaluate the performance under experiment conditions, with the goal to extrapolate the necessary hardware specifications for the operation with the full CB-SADC setup at the CBELSA/TAPS experiment. A suitable and very affordable option was the **IBM System x3650 M2** with the following specification:

- 2x Intel XEON 4C X5570 2.93 GHz
- 16 Gbyte PC3-10600R ECC RAM
- 8x hot-swappable 2.5 in hard disk slots
- IBM SERVERAID BR10I RAID controller[1]
- 2x HotSwap power supply
- 4x Gigabit Ethernet port

The server has been set up with the LINUX distribution **CentOS 6.8** using the `cobbler` system for rapid and reproducible deployment, which is customary in the CBELSA/TAPS experiment. After the initial tests, the server underwent several upgrades.

**Storage Configuration** The SERVERAID BR10I RAID controller has been replaced by the MR10I version, which allowed to use a RAID5 configuration (N+1 redundancy) additional to RAID0 (striping), RAID1 (mirroring), and RAID10 (combination of RAID0 and RAID1). Two 73-Gbyte RAID-grade hard disks in a RAID-1 configuration are used as system disks. Two 1-Tbyte customer-grade solid state disks (CRUCIAL MX300) in a volatile RAID-0 configuration are used as a fast temporary storage for the recorded waveforms. After some time (usually within a day) the waveforms are moved to a RAID-5 storage consisting of four 2-Tbyte customer-grade hard disks (TOSHIBA MQ03ABB200). This storage space is already filled after roughly three weeks of beam time. For long-term storage, a separate file server has been installed in 2018 with the start of the production beam times.

The performance of the system has been evaluated carefully. The disks are connected to the MR10I through the SATA-II interface with a net data rate of 300 Mbyte/s per port, unfortunately limiting the solid state disks that are specified with a write rate of 510 Mbyte/s (in a more up-to-date setup, an SATA-III controller with twice the speed would be chosen). The MR10I itself is connected to the motherboard through the PCIE 2.0 x8 interface, allowing a performance of up to 4 Gbyte/s, hence a sufficient speed for the solid disk stripe set. The real writing speed to the solid state disk array has been measured to be around 400 Mbyte/s, which is only two-thirds of the expected theoretical speed. This may be a limitation of the MR10I and has not been investigated further, as it has been sufficient for all tests. It shows, in essence, that all components should be well matched in the final version of the `cbsadcserver`, such that no unnecessary bottlenecks are introduced.

[1] Redundant Array of Independent Disks, a special controller that allows to combine multiple disks for higher speed or redundancy.

**Network Configuration** The four Gigabit Ethernet ports were clearly not fast enough to support the incoming data streams from multiple CB-SADCs. Furthermore, they are required for connection to other networks (Control and Data network, cf. Fig. 2.20). Therefore, to allow for higher traffic throughput, the `cbsadcserver` was equipped with two Mellanox ConnectX-2 (`MNPA19-ATR`) network interface cards, supporting an SFP+ module with speeds up to 10 Gbit/s (10GBASE-SR/LR), connected to the motherboard through a PCIe 2.0 x8 interface. Thus, from the network point of view, the `cbsadcserver` is well-equipped to be connected to ten CB-SADCs with two 1 Gbit/s connections each. The total throughput of one 10 Gbit/s = 1.25 Gbyte/s port is, currently, already limited by the write speed of the solid state disk array with its empirical 400 Mbyte/s limit.

### 9.1.2. Network Infrastructure

The CB-SADCs have to be connected to the `cbsadcserver` through an Ethernet switch. Each CB-SADC has two 1-Gbit SFP ports, which can be equipped with either copper or fiber modules. Although it was first planned to use a topology completely based on fiber connections to eliminate every possible ground connection, this was later revised since a ground connection in the network domain has not shown negative effects, and the cost of fiber switches and SFP modules was substantially higher compared to the copper equivalent. Candidates for the connection of up to twelve CB-SADCs, requiring 24 Gigabit Ethernet ports, have been evaluated. The requirements for the candidates were: 24 Gigabit Ethernet (copper) connectors, at least two SFP+ ports for 10 Gbit/s uplinks, and support for jumbo frames (see page 158). After diligent research, four switches have been taken into consideration:

| Name | Uplinks | Packet Rate | Switching Capacity | Latency | Depth | Price |
|---|---|---|---|---|---|---|
| Dell N1524 | 4 | 128 MP/s | 128 Gbit/s | unknown | 25.7 cm | 525€ |
| Hp 2530 (J9776A) | 2 | 65 MP/s | 88 Gbit/s | <7.4 µs | 25.4 cm | 395€ |
| Hp 2540 (JL354A) | 4 | 95.2 MP/s | 128 Gbit/s | <3.8 µs | 20.0 cm | 865€ |
| Huawei S5720S-28X | 4 | 96 MP/s | 336 Gbit/s | 6.0 µs | 22.0 cm | 655€ |

Table 9.1.: List of candidates for Ethernet switches, with the parameters that are important for the evaluation. The prices reflect the gross price in August 2018.[2]

**Evaluation** Based on the numbers stated in Table 9.1 a decision for a switch was made. It will be explained in this paragraph how the requirements have been calculated and evaluated.

- The number of **Uplinks** (which are 10 Gbit/s-capable) determines the bandwidth with which the data from the CB-SADCs can be transferred to the LEvBs (`cbsadcserver`). In theory, when all CB-SADCs would send with the highest data rate, a bandwidth of 24 Gbit/s would be necessary (actually a bit less due to more efficient encoding). In practice, it is very likely that two 10 Gbit/s ports would be sufficient, but since all the regular 1 Gbit/s copper ports are occupied, a third and fourth port can also prove valuable for debugging or not-yet-foreseen use cases. Due to this, the Hp 2530 (J9776A) with only two ports is not a preferred choice.

[2] At the time the decision was made and the first switch was bought, the prices were 560€, 560€, 1440€, and 780€ respectively.

- The **Packet Rate** reflects how many packets per second the switch can forward, summed up between all ports. Considering that twelve CB-SADCs with two network interfaces (and 32 channels) each may theoretically operate with up to 75.0 kHz in unacknowledged *Feature Only* mode (see equation 7.25 on page 170) with one packet per event, this adds up to

$$12 \cdot 2 \cdot 75.0\,\text{kHz} \cdot 1\,\text{Packet} \approx 1.80\,\text{MP/s}; \tag{9.1}$$

  and in *Feature+Sample* mode (waveform samples sent for each channel and event), with a rate of up to 3.24 kHz (see equation 7.27 on page 172) with 33 packets per event, to

$$12 \cdot 2 \cdot 3.24\,\text{kHz} \cdot 33\,\text{Packets} \approx 2.57\,\text{MP/s}. \tag{9.2}$$

  Taking into account that the *burst* mechanism requires one additional packet to each of the CB-SADCs's network interfaces for every 32 packets received, those numbers increase by $\frac{1}{32}$ to 1.86 MP/s and 2.64 MP/s respectively. Although it was already shown that other bottlenecks reduce those event rates to 35.0 kHz and 2.77 kHz respectively under real operating conditions (see section 8.3), and will be reduced furthermore in conjunction with the experiment's DAQ, it would not be advisable to introduce yet another bottleneck. Still, even the switch with the worst specification (HP 2530) has a sufficient performance regarding the packet rate.

- The **Switching Capacity**, in theory, needs to be no more than $24 \cdot 1\,\text{Gbit/s}$, which is easily fulfilled by all switches.

- The **Latency** affects the maximum packet rate, if the *burst* mechanism is activated, as it adds up to the round trip time of the packet. One can only say that is should be reasonably low compared to the delay introduced by the *burst* handling of the LEvB, which, as section 9.9.1 will reveal, is some 100 µs. Indeed the latency is below 10 µs for three of the switches. It may be expected that the same is true for the DELL model, but since it is not specified it might sound a note of caution.

- The **Depth** may appear a strange criterion at first, but the rack space at the CBELSA/TAPS experiment does not offer a lot of space in the back, if the switch is mounted behind a NIM crate. To avoid the excessive use of spacer sleeves, a smaller depth is desirable. This favors HP 2540 (JL354A) and HUAWEI S5720S-28X over DELL N1524 and HP 2530 (J9776A).

- Last but not least, the **Price** should be taken into consideration. At the time that the decision was made, the two most promising candidates, HUAWEI S5720S-28X and HP 2540 (JL354A), differed in the price by almost a factor of two.

The evaluation was in favor of the HUAWEI S5720S-28X, which as been obtained and integrated in to the CB-SADC rack. A second module is now in operation at the CBELSA/TAPS experiment since the first production beam time, in conjunction with a 10 Gbit/s 10GBASE-LR module to send the data to the `cbsadcserver`, situated more than 100 m away. Both switches have performed well and did not show any unexpected limitations in speed, or any amount of dropped packets at all.

### 9.1.3. Sync Master and Slowcontrol

In the first revision of the CB-SADC rack, the *trigger/sync* and *slowcontrol* connections were not yet of importance for the tests; but at some point it was necessary to test and mimic the conditions of the CBELSA/TAPS experiment, which required the integration of a VME crate containing a VME CPU and an ELB-VME-VFB board (a versatile FPGA platform with extension slots for custom hardware). The board together with extensions can be seen in Figure 9.1, and in closeup and operation in Figure 9.2. The left side shows the *Sync Master Extension*, which is used in the central trigger distribution of the CBELSA/TAPS experiment, and the right side shows the *Slowcontrol Extension* that was developed in the scope of this thesis (see section 6.5). In addition to that, the free extension slot in the middle was equipped with an eight-fold NIM output that was helpful for debugging the trigger signals with an oscilloscope, since a JTAG debugging interface like on the CB-SADC Backplane was not available. An application of this debugging interface can be seen in Figure 9.24 (page 215).

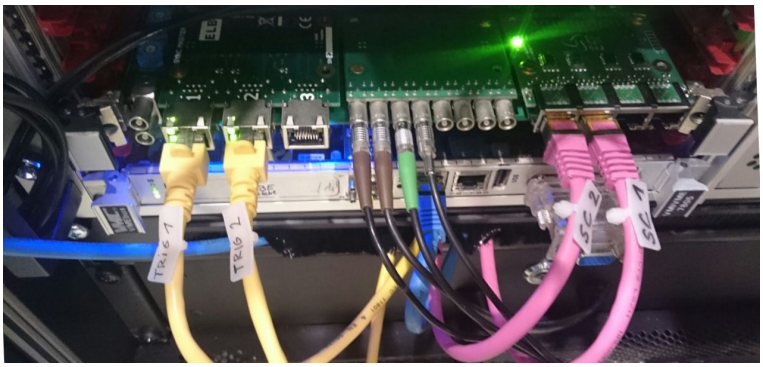



Figure 9.2.: Close-up view of the ELB-VME-VFB board, with the *Sync Master* on the left, an eight-fold NIM output in the middle, and a four-channel *Slowcontrol Extension* (see section 6.5) on the right. One of the trigger (yellow) and slowcontrol (magenta) cables connect to each of the two Backplanes.

## 9.2. Comparison of CB-SADC and QDC in the Laboratory

In his master thesis [122] J. Schultes has investigated the performance of the first CB-SADC prototype (`ADC-64K-CB v1.0`). He has compared the CB-SADC with the QDC readout in regard to linearity and resolution. Those investigations were not conducted with the Crystal Barrel readout chain in the experimental hall, but instead in a laboratory with a signal generator, such that the digitizers' performance could be tested with arbitrary amplitudes. Unfortunately, the investigations of the linearity were inconclusive for both the QDC and the CB-SADC due to the non-linear behavior of the signal generator itself, so the results will not be reproduced here. In the following, the laboratory setup will be presented in more detail, followed by the comparison of both system's resolutions.

### 9.2.1. Setup

The CB-SADC rack was integrated into an existing setup where also the Crystal Barrel Shaper and QDC (cf. sections 2.8.7 and 2.8.8) were available. This allowed to compare the performance of both readout systems simultaneously. A recording of an original waveform stemming from a cosmic particle measurement of the Crystal Barrel Calorimeter with APD readout has been used to generate a test signal with an arbitrary waveform generator. The single-ended signal was converted to a differential signal and then fed to the Crystal Barrel Shaper and from there to the QDC, resembling the existing readout of the Crystal Barrel Calorimeter. For the investigation of the CB-SADC, the signal was connected through a shaping amplifier (NIM module) with a shaping time of 1 µs, coarsely resembling the characteristics of the Crystal Barrel Shaper and the Analog Input Card. Since at the time of those tests the Analog Input Card was not yet developed, an adapter board was installed in the CB-SADC such that the differential test signal could be directly connected (this solution is shown in Figure A.5 on page 224). The signal was scaled in amplitude during the measurement to cover almost the full dynamic range of both digitizers. The trigger was generated either from the arbitrary waveform generator or (for an uncorrelated trigger) from a NIM clock generator. Those modules are shown in Figure 9.3.

Although an early version of the feature extraction was already implemented in the firmware, the investigation was focused on the extraction of information from the raw waveforms. Comparisons of the features calculated on the CB-SADC's FPGAs with the features calculated *offline* from the recorded waveforms were conducted in [122] as well, and have led to corrections and optimizations of the FPGA algorithms.

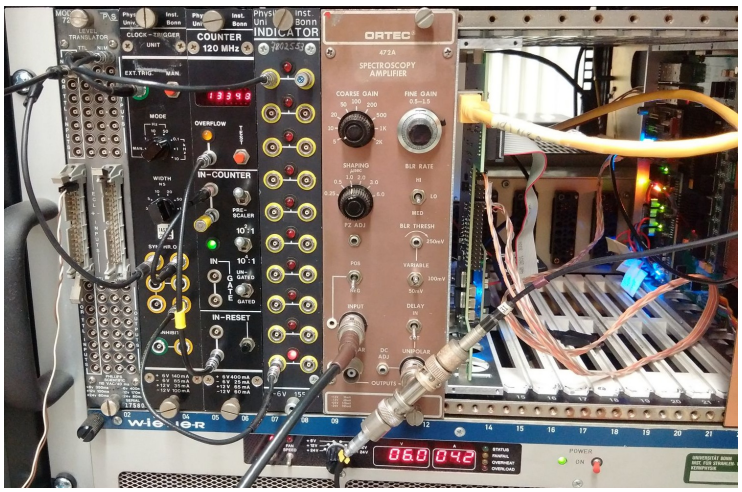



Figure 9.3.: NIM modules in the CB-SADC rack for a measurement of the energy resolution. From left to right: NIM-to-TTL converter (for the conversion of the trigger signal from the arbitrary waveform generator), Clock-Trigger Unit (to create a trigger pulse with a fixed width, or an independent trigger), Counter, LED Indicator (for debugging), ORTEC Spectroscopy Amplifier with 1 µs shaping (as surrogate for the Analog Input Card which was still in development). The open module is the SADC Controller used for slowcontrol (cf. section 6.4.1). The right-hand side shows the open NIM cassette of the CB-SADC.

### 9.2.2. Measurements and Analysis

Several thousand events were recorded for different amplitude settings of the arbitrary waveform generator, for QDC and CB-SADC. The integral value of the QDC, the integral value of the CB-SADC and a waveform sample containing the pulse, were recorded. The QDC's gate length was optimized from 6 µs to 4 µs, yielding a better resolution. From the CB-SADC's waveform sample, an integral with optimized position and length as described in Figure 7.20, was extracted and used for evaluation (as it yielded a better result than the FPGA algorithm at the time). For each amplitude setting, the resulting distribution of integral values was evaluated, as shown exemplary for the CB-SADC in Figure 9.4. The resolution $\frac{\sigma_A}{A}$ was extracted with the help of a Gaussian fit, $A$ being the amplitude and $\sigma_A$ the standard deviation of the Gaussian distribution.

Although the QDC's resolution has been improved during the tests, the CB-SADC could still outperform it over the whole range. This is shown in Figure 9.5.

**Exceeding the Dynamic Range** As part of the analysis it was evaluated how the CB-SADC would perform in cases where the amplitude would exceed the dynamic range of the LTM9009-14 ADC ICs. This operation mode can be useful to increase the resolution in the lower amplitude region, assuming that the SNR is effectively increased by the amplification ($A$ increases while $\sigma_A$ stays unchanged). Naturally, this is only possible if a method can be found to reconstruct the true amplitude (or integral) for large pulses that are *cut off* (in the region where the LTM9009-14 is operating in saturation). An example for a cut-off pulse can be seen in Figure D.2. It is evident that this can work only for the CB-SADC but not for the QDC: while the QDC is limited in the integral value it can determine, the CB-SADC can still extract information from the pulse shape. Although the information of the height of the pulse is not present in the data directly, it can still be reconstructed from the width of the pulse (e.g. just below the clipping). This feature is called *Time over Threshold*. Measurements of this method are included also in Figure 9.5 for two different thresholds. Surprisingly the resolution almost matches with the resolution of the calculated integral when the pulse height is just corresponding with the threshold (but does not get better with higher amplitudes).

From a technical point of view, such an operation mode seems possible. The datasheet of the ADC ICs (LTM9009-14) states that when "*[the analog input] voltages are taken [...] above VDD, they will be clamped by internal diodes.*" [149], thus it should not be a problem to expose it briefly to an overvoltage (similarly for the op-amps). After all, it can be expected that pulses of the highest amplitude will occur with a low probability, and the overvoltage should not be present for more that a few microseconds. A drawback of this method is that detecting piled-up pulses will not work with clipped pulses, since it relies on the ratio of calculated pulse's integral to its maximum (both of which cannot be directly determined). It also further complicates the data analysis, since both features (integral and time-over-threshold) have to be calibrated individually and matched in a transition region.

**Conclusion** This first quantitative test of the CB-SADC has yielded a positive result. It could beat the resolution of the QDC in a setup which was designed as competitive as possible. One might argue that this does not come as a surprise, considering that more than 20 years of technological advances have been made. Nonetheless the QDC is a high precision digitizer, and had been optimized in conjunction with the custom Crystal Barrel Shaper, when the

CB-SADC was still in its infancy. At the time of this investigation, J. Schultes found that the data from the feature extraction of the CB-SADC firmware did not yet reach the potential shown in the *offline* feature extraction (but still outperformed the QDC) [122]. His master thesis and ongoing doctoral thesis allowed to continually improve the feature extraction, thus the performance of the CB-SADC.

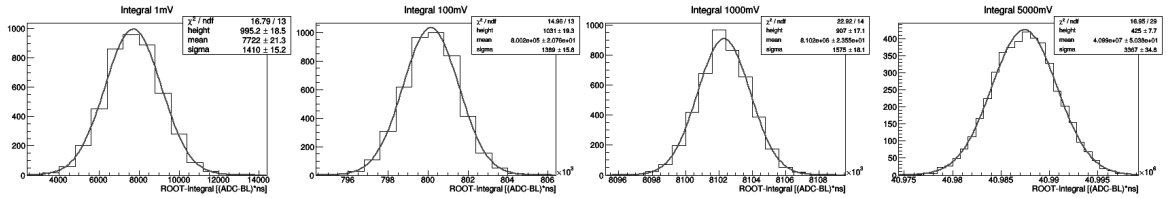



Figure 9.4.: Example histograms of the integral values calculated from recorded CB-SADC waveform samples. The setting from 1 mV to 5000 mV represents the amplitude of the signal generator for each set of measurements. The resolution is calculated from the *mean* and *sigma* of the Gaussian fit. For the first value, $\frac{\sigma_A}{A} = \frac{1410}{7722} \approx 0.18$, which can be identified as the first data point in Figure 9.5. [122]

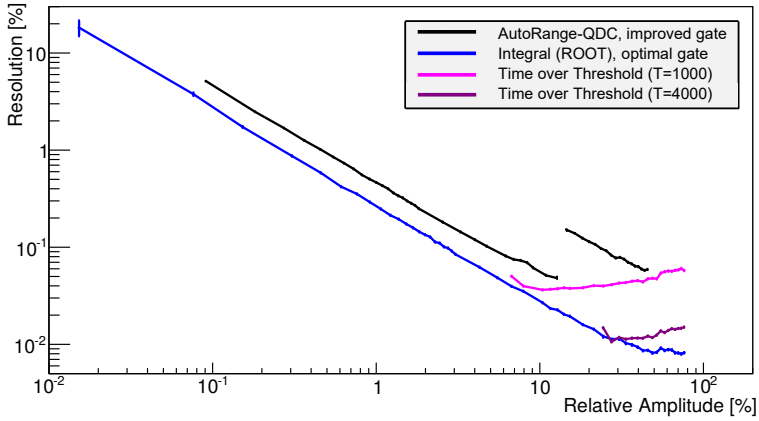



Figure 9.5.: Comparison of QDC and CB-SADC resolution. The x-axis shows the relative amplitude in terms of each digitizer's full scale rage. Due to a limitation of a single-ended-to-differential converter, only 80 % could be reached for the CB-SADC and 60 % for the QDC (in any case the lower amplitudes are more crucial). The black line, showing the QDC data, exhibits a *jump* when the dual-range threshold is crossed after $\frac{1}{8}$ of the amplitude. The blue line, showing the CB-SADC data (integral calculated from the waveform sample), is almost straight. The deviations towards higher amplitudes could stem from the arbitrary waveform generator. The magenta and purple line show the resolution for time-over-threshold measurements with two different thresholds. [122]

## 9.3. Minimum Ionizing Particles @ CBELSA/TAPS

In October 2016 the first test beam time with APD readout of the Crystal Barrel Calorimeter took place. At the time only the forward half of the calorimeter was equipped with the new APD readout, and only a part of the new readout electronics had been installed. The first CB-SADC prototype (`ADC-64K-CB v1.0`) has been installed in the experimental hall, and data was taken for a full day in a dedicated setup for the measurement of Minimum Ionizing Particles (MIPs). The setup will be presented in this section, and the results from the analysis of J. Schultes [122] will be summarized.

### 9.3.1. Setup

Although the first version of the CB-SADC rack was already available at the time, the CB-SADC was installed in one of the NIM crates housing the BuffTi modules (close to the Crystal Barrel Calorimeter), and a network connection was established to the CB-SADC rack hosting the `cbsadcserver`. Before the test beam time, a first prototype of the Analog Input Card had been produced. This allowed the direct connection from the BuffTi module to the CB-SADC prototype with two MRJ21 cables, which carried the signals of one slice with 23 CsI(Tl) crystals. Unfortunately, the Backplane was not finalized at the time, so the CB-SADC could not be fully integrated into the experiment's trigger/sync system. A connection to the trigger (the *Event* signal) was then established through the first revision of the Extension Board (Fig. 6.27a page 129). A new trigger configuration dedicated for this test has been configured. It was required that a hit in the Inner Detector and in at least one of the 23 crystals was present. This is meant to ensure two conditions: first, the trigger in the Inner Detector (consisting of scintillating fibers) most likely indicates the passage of a charged particle. Secondly, a path of a particle from the Inner Detector to a Crystal is defined. While this excludes particle tracks not passing through the Inner Detector, it is unfortunately not a sufficient condition to guarantee that only one crystal was traversed in its full length. Moreover, it is still possible that uncorrelated events are measured (e.g. a photon hitting a calorimeter crystal while coincidentally a charged particle passes through the Inner Detector), which leads to a substantial background.

### 9.3.2. Analysis

The analysis focused on the reconstruction of energy signatures from MIPs. According to the Bethe-Bloch formula, the energy loss of a particle traveling through a medium is minimal at a momentum of 3 to 4 times its rest mass ($\beta\gamma = \frac{p}{M_0 c} \approx 3$ for CsI(Tl)). This leads to a steady deposition of energy in the scintillator crystals. Charged pions, which are produced during the beam time, may leave such a signature (but are usually not of interest, since the CBELSA/TAPS experiment focuses on final states where neutral mesons decay into photons). Figure 9.6 shows the energy spectrum for a crystal in the most forward region, which has the best statistics (due to the Lorentz boost). The increase is well visible at an non-calibrated ADC value of $\approx 75 \times 10^3$.

The position of the MIP peak can now be determined more exactly by a Gaussian fit on an exponential background. Now, the non-calibrated values of the CB-SADC's integral feature can be coarsely calibrated, knowing that the expected energy deposit of a MIP in the

Crystal Barrel Calorimeter's CsI(Tl) crystals with a length of 30 cm should yield an energy of 164.6 MeV; this energy value stems directly from a fit of calibrated QDC data to a MIP peak [122]. Subsequently, the width of the peak can be calculated from the calibration and the fit values. This has been done for all 23 crystals, which is shown for nine different crystals in Figure 9.7. The corresponding Gaussian fits are shown in Figure 9.8.

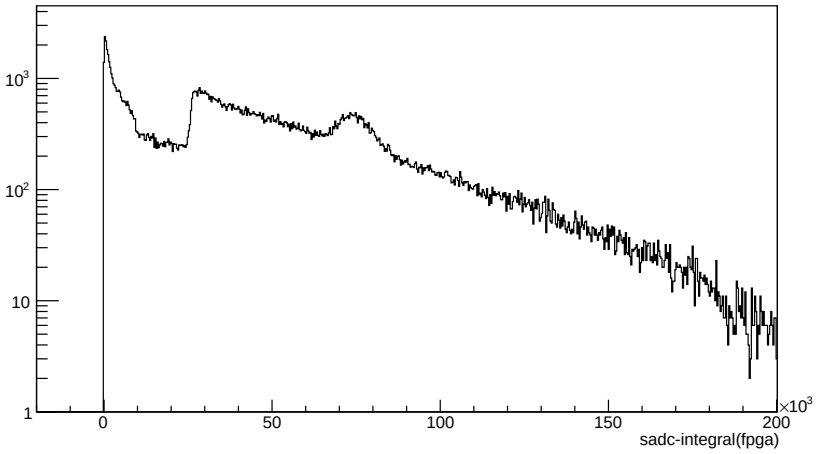



Figure 9.6.: CB-SADC energy spectrum, using the integral feature calculated on the FPGA (`event_data.int(i)`). The data is taken from one crystal with $\Theta = 5$ (crystal type 9). The peak around $75 \times 10^3$ (non-calibrated units) corresponds to energy deposits by MIPs with roughly 164.6 MeV. [122]

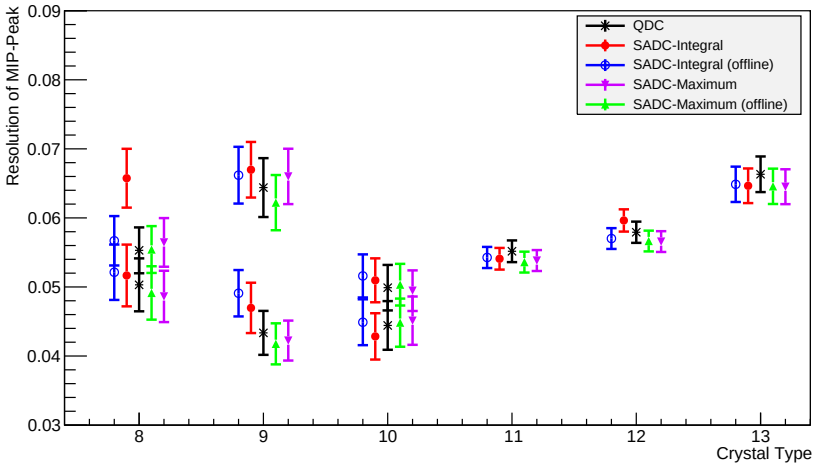



Figure 9.7.: Comparison of the resolution of the MIP peak, for QDC and CB-SADC (with different methods of extracting the energy information). The resolution is calculated as $\frac{\sigma_E}{E}$, with values from Gaussian fits that are shown in Figure 9.8. Due to higher statistics in the forward region (highest value of the crystal type) the errors get smaller, but for type 13 it grows again due to shower losses. The reason for the outlier was not investigated at the time. [122]

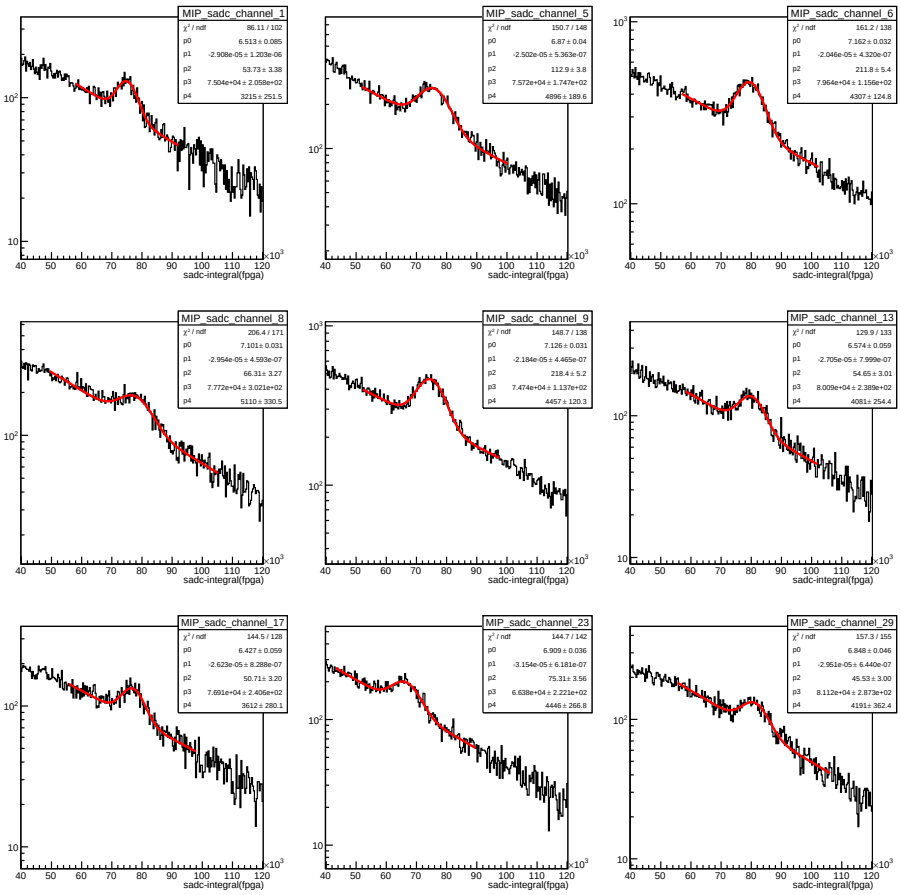



Figure 9.8.: Exemplary fits of a Gaussian function with an exponential background subtraction for nine different channels (crystal types) of the CB-SADC. The data is from the *integral* feature extracted by the FPGAs onboard the CB-SADC.

**Conclusion** The analysis shows that the width of the MIP peak of both systems are in good agreement with one another. It has to be noted that indeed this is only a comparison of the widths and not the energy resolution: the energy resolution calculated from MIPs does not state the performance of the calorimeter. The presented measurement is merely a validation of the CB-SADC's correct functionality with respect to the whole experiment. With this in mind, it can be concluded that the CB-SADC delivered data consistent with the QDC, and that the features calculated on the CB-SADC's FPGA, namely integral and maximum of the pulse, seem to yield a comparable result to features extracted from the recorded waveforms (*offline*). Interestingly, the CB-SADC was accidentally operated with a dynamic range of 5.0 GeV instead of 2.5 GeV, due to a misunderstanding of the gain specifications of the APD's line driver. This still leaves a slight advantage for later measurements, where the dynamic range was corrected.

## 9.4. Readout of the Energy Sum

During the October 2016 test beam time, the energy sum signal (cf. section 2.8.5) has been evaluated with the CB-SADC. Since the energy sum signal is produced in the BuffTi module, it was decided to transmit it through the free channel of the MRJ21 connector[3] to the CB-SADC. The energy sum signal is the sum of the timing-filtered signals that are sent to the TDC, with a band pass center frequency of 3.2 MHz, compared to 160 kHz as used on the Analog Input Card for the energy signals. This makes the signal steeper and shorter. Since the energy sum signal passes through the Analog Input Card, this unsuitable 160 kHz band-pass filter has been removed for the affected channels, such that the signal is only buffered before it enters the CB-SADC. During the test beam time, the functionality was evaluated with the help of the light pulser system. While the light pulser was changed in intensity, the waveforms of the 23 crystals and the according energy sum signal were observed with the QT DEBUGGING TOOL. An example is shown in Figure 9.9. Qualitatively the signal has behaved as expected: the amplitude of the energy sum has changed according to the light pulser intensity, and the signal was clearly visible already for very small intensities. Quantitative investigations were done in the scope of the master thesis of S. Ciupka [91].

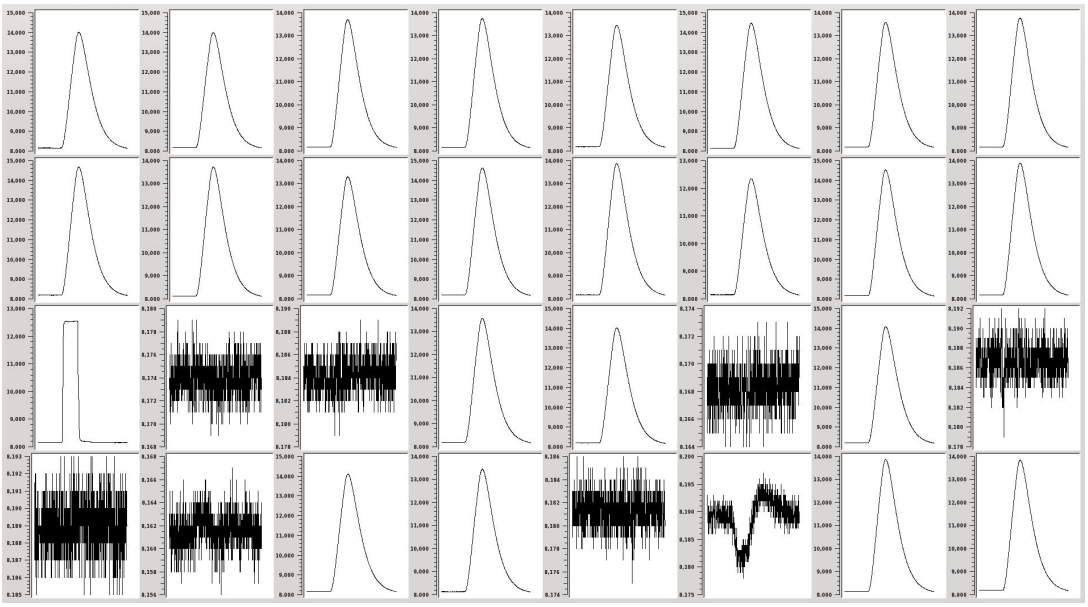


Figure 9.9.: Investigation of the energy sum signal with the QT DEBUGGING TOOL, using a light pulser to produce scintillation-like light signatures for the APDs. The 23 identical-looking pulses are the signals from the connected slice. The energy sum signal for this slice is visible in the first column of the third row. The almost rectangular shape is expected from the light pulser. The seven channels showing regular noise are unconnected channels from an adjacent slice of the Crystal Barrel Calorimeter, and the third-last channel is its also unconnected energy sum signal, showing strong crosstalk behavior as the wires are physically unconnected in the BuffTi, leading to an antenna-like behavior.

[3] One of 24 differential pairs is free, exactly as needed for the transmission of one energy sum signal per slice.

## 9.5. Noise

In this section, the measurements which allowed the determination of the electronic noise level from the recorded features and waveforms will be presented: The calculated noise levels will be compared with the theoretical expectations from section 1.4, as well as measurements/simulations from the dissertation of C. Honisch [61], the simulation from M. Steinacher [97], and the pedestal width measured by the QDC. Afterwards it will be shown how the CB-SADC was used to identify a noise source in the experiment with the help of a Fourier analysis of the waveform data.

### 9.5.1. Analysis

The signals from the calorimeter settle at a certain level in the absence of pulses, which is commonly referred to as *baseline* or *pedestal*. It can be observed that this baseline fluctuates, meaning it is *noisy*. Possible sources of this noise can be found throughout the whole signal chain (starting with the dark current of the APDs), and have been discussed with a focus on digitizers in section 1.4. In section 6.1.1 it was shown how the high-frequency noise is being suppressed by the CB-SADC's Analog Input Card in the analog domain. After digitization, the noise can be further reduced as explained in section 7.3.5. Two different methods to quantify this noise will be presented in the following paragraphs.

**RMS Noise of Recorded Waveforms** The baseline can now be investigated to see if its noise matches the expectation. A common quantifier is the RMS noise, which is equivalent to the standard deviation of a baseline distribution with uniformly distributed noise. The according Gaussian function is defined by the equation

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}. \tag{9.3}$$

The parameter $\sigma$ is the standard deviation of the Gaussian distribution, and is usually referred to as the *width*. If the mean of the signal ($\mu$) is zero, the standard deviation is indeed equivalent to the electrical RMS noise.

The quantization step of the ADC, according to equation 5.1 on page 75, is 122 µV. Since the data is decimated from 80 MS/s to 20 MS/s, the data is internally processed as a 16 bit integer type instead of 14 bit, and the step size is decreased to an equivalent of 30.5 µV.

A set of baselines has been recorded for the test of the baseline shifting circuit of the Analog Input Card. During the recording, the Crystal Barrel Calorimeter was fully operational including the high voltage supply of the APDs. The electron beam from ELSA was not extracted to the Crystal Barrel area. For each event and each channel, a histogram of the baseline was created and a Novosibirsk fit has been applied to the data. The Novosibirsk distribution is a modification of the Gaussian distribution, taking into account that the distribution may in fact be asymmetric. Strictly speaking only the standard deviation of a Gaussian fit is equivalent to the RMS noise. The Novosibirsk fit can be justified by the fact that cosmic radiation may leave a signal in the moment the baseline is recorded, leading to the observed asymmetry. With good conscience it was assumed that in total absence of energy deposits in the calorimeter a Gaussian fit would have yielded a comparable standard deviation.

The standard deviations of the Novosibirsk fits have been extracted event- and channel-wise, and are visualized in a 2D-histogram in Figure 9.10. A summation of the data from all channels yields a histogram as shown in Figure 9.11. This is a distribution of the RMS noise levels of all channels, and its peak position value corresponds to the average standard deviation. According to the fit, this yields a number of 3.847 (raw 16 bit ADC units), which can be translated to an RMS noise of

$$\sigma_{\mathrm{v}} = 3.847_{\mathrm{RMS}} \cdot 30.5\,\mu\mathrm{V} \approx 117.3\,\mu\mathrm{V}_{\mathrm{RMS}} \tag{9.4}$$

At a full scale range of 2.5 GeV at a digitizer input voltage of 2 V, this corresponds to

$$\sigma_{\mathrm{E}} = 117.3\,\mu\mathrm{V}_{\mathrm{RMS}} \cdot \frac{2.5\,\mathrm{GeV}}{2\,\mathrm{V}} \approx 146.6\,\mathrm{keV}_{\mathrm{RMS}}. \tag{9.5}$$

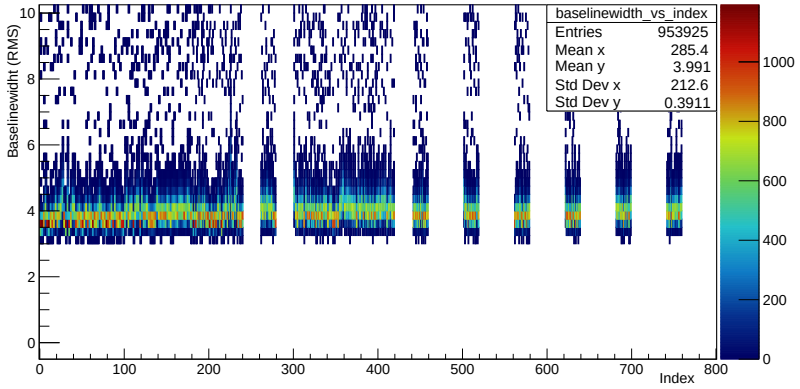



Figure 9.10.: Distribution of the baseline widths. Baseline waveforms have been recorded, and the RMS values have been calculated channel- and event-wise. The RMS values are plotted in this histogram against the channel, colored based on the frequentness. The dataset includes the four most forward rings, with a smaller crystal index indicating more forward direction. At indices below 180, every other index is empty due to the specific counting method and the calorimeter's geometry. The outliers (seemingly higher noise) are most likely caused by energy deposits from cosmic radiation. *Data analysis performed by J. Schultes.*

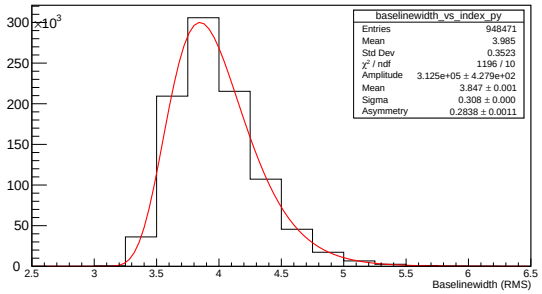
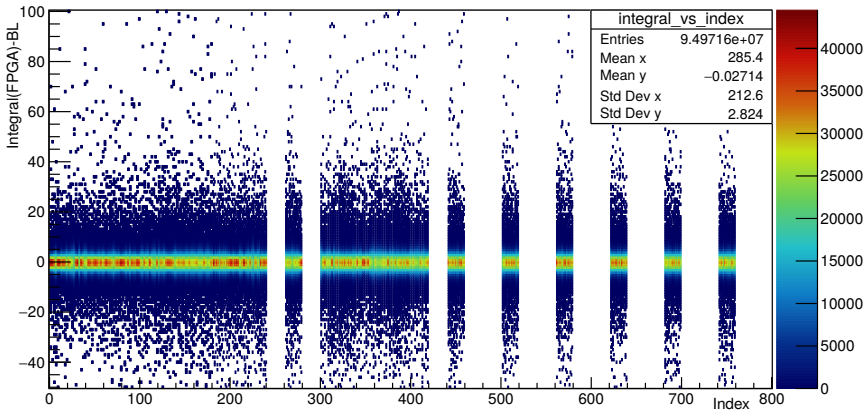


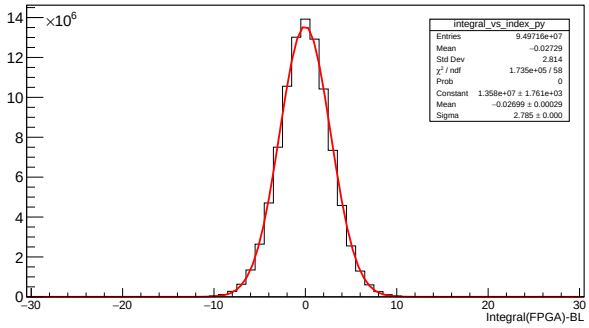

Figure 9.11.: Projection of all widths shown in Figure 9.10. A Novosibirsk fit was applied to account for the tail towards higher noise values. The fit shows a peak position of 3.847 representing the mean RMS noise. *Data analysis performed by J. Schultes.*

**Width of Integral Measurements** It is interesting to investigate a different quantifier, one that is better comparable to the *noise* of the QDC. While the true RMS noise can be calculated for the CB-SADC, this is not possible for the QDC. Its quantifier is called the *pedestal width* and is determined before each run (every $\approx 30\,\text{min}$). In this time, the QDC measures the integral for $\approx 5000$ events with an uncorrelated trigger (*clock trigger*). The center value of the distribution is then taken as the *pedestal* for this run, and is subtracted from all measured integrals. The width of the distribution of the integral values is called the *pedestal width* and is the quantifier for how *noisy* the regarding channel is. In a similar way, the CB-SADC could calculate the integral to find a mean *pedestal* (baseline), but due to its sampling character it is able to measure the baseline for every event. This way a per-event-based baseline subtraction is possible. Figure 9.12 shows the 2D-histogram of the CB-SADC's recorded integral value during a run with clock trigger. The width of the resulting distribution is 2.785 (uncalibrated CB-SADC integral value). Using a mean calibration factor for the integral value [189], this can be translated to the RMS *noise* of the integral distribution:

$$\sigma_\text{I} = 2.785_\text{RMS} \cdot 0.053\,\text{MeV} \approx 147.6\,\text{keV}_\text{RMS} \tag{9.6}$$




(a) 2D-histogram of the recorded integral values. The x-axis shows the crystal index (exhibiting empty space where no CB-SADC readout is present). The values are centered around 0 due to the per-event baseline subtraction.




(b) Projection of the data of all channels. The Gaussian fit yields $\sigma \approx 2.785$.

Figure 9.12.: Investigation of the integral distribution to determine a *noise* quantifier similar to the *pedestal width* of the QDC. *Data analysis performed by Jan Schultes.*

### 9.5.2. Discussion

First, the RMS noise measurement can be compared to the expectations from simulations. Comparing $\sigma_{\mathrm{v}} = 117.3\,\mu\mathrm{V_{RMS}}$ to equation 5.3 on page 75 seems odd at first, since $117.3\,\mu\mathrm{V_{RMS}} < 146\,\mu\mathrm{V_{RMS}} = \sigma_{\mathrm{trnoise}}$. This can be explained by the four-fold decimation (see section 7.3.5), which effectively cuts off parts of the quantization noise (see equation 1.17 on page 17). It is now interesting to compare the energy equivalent of $\sigma_{\mathrm{E}} = 146.6\,\mathrm{keV_{RMS}}$ to the noise simulations of the APD readout chain, which were done by M. Steinacher in 2010 [97]. From his simulations it follows that with a 4th-order band-pass filter, a noise level of $126\,\mathrm{keV_{RMS}}$ can be achieved. With the Crystal Barrel Shaper, after slight modifications, the simulations yielded a noise of $147\,\mathrm{keV_{RMS}}$; but those numbers do not include the transition noise introduced by the digitizer, instead only the electrical noise of the readout chain.

Secondly, the measurement of the width of the integral distribution can be compared to the QDC's pedestal width. In his dissertation, [61] C. Honisch has summarized that the previous (PIN-diode) readout of the Crystal Barrel Calorimeter (with Crystal Barrel Shaper and QDC) yielded a pedestal width of $350\,\mathrm{keV_{RMS}}$, and that a test setup with the new APD readout improved this to $150\,\mathrm{keV_{RMS}}$. With the CB-SADC, a comparable measurement method has produced a pedestal width of $\sigma_{\mathrm{I}} = 147.6\,\mathrm{keV_{RMS}}$.

It has to be taken into account that the shaping topology on the Analog Input Card is much less intricate than the simulated scenarios and the topology of the Crystal Barrel Shaper. Moreover, the Analog Input Card offers the baseline-shifting circuit and a digitally-adjustable pole-zero cancellation. Taking this into account, both measurements put the CB-SADC in a very good light, and it can be concluded that it fulfills the expectations.

### 9.5.3. 312 kHz Noise Investigation

In June 2017 it was found that the Crystal Barrel Calorimeter signals exhibited a higher noise level than anticipated. This has been investigated directly at the Crystal Barrel Calorimeter with an oscilloscope, which has shown to be a tedious procedure, since the cable connections from the Crystal Barrel Calorimeter have to be disconnected, and an adapter needs to be installed to connect the oscilloscope. At most four channels can be studied at a time.

The CB-SADCs have proven to be helpful for this situation. In a first step, the Qt Debugging Tool has been connected to one CB-SADC unit, which allowed to inspect all 32 channels of that unit at once. The resulting waveforms are shown in Figure 9.13. Even to the untrained eye, it would seem clear that this is not the regularly expected *white* noise structure, instead there seems to be one dominant frequency contribution, expressing itself with what seems to be a periodic oscillation on top of the expected noise.

J. Schultes has prepared an analysis script which allowed to immediately assess the scenario after taking a small set of waveform data from the CB-SADCs. Figure 9.14 shows the resulting plots from this script. In Figure 9.14a the position and the width of the baselines can be seen. During the measurement, only a section of the Crystal Barrel Calorimeter was connected to the low voltage supply, which is clearly prominent due to the higher baseline width. Figure 9.14b shows a Fourier analysis of the waveforms of all recorded channels. Here a prominent horizontal band is clearly visible, not only in the channels connected to the low voltage supply, but apparently also somehow induced weakly in the unconnected channels. Since the waveform window has a length of 512 samples, and the (decimated) sampling

frequency was 20 MS/s, the *binning* of the Fourier transformation is $\frac{20\,\mathrm{MHz}}{512} \approx 39\,\mathrm{kHz}$. This allowed to estimate the interfering frequency to a range of 293 kHz to 332 kHz.

**Conclusion** An accurate measurement with an oscilloscope gave a more precise value of 312 kHz for the noise oscillation. Indeed it was found that the CAEN low voltage supply uses a switching frequency of 312 kHz with a specified ripple of $10\,\mathrm{mV_{PP}}$. This was confirmed with a measurement yielding $13\,\mathrm{mV_{PP}}$. The problem has been resolved by adding a 3rd-order low-pass filter to the low voltage power supply. It can be concluded that the CB-SADCs were valuable in the analysis of the problem, as they could offer an oscilloscope-like access to all connected channels at once.

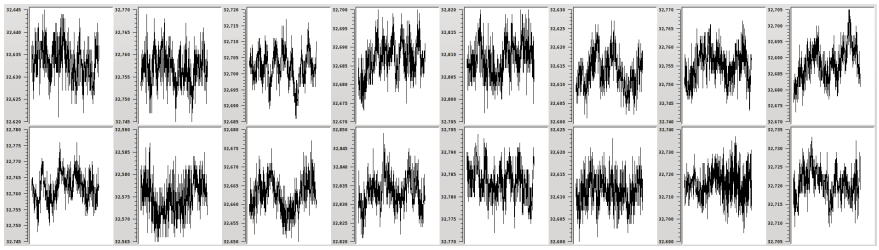
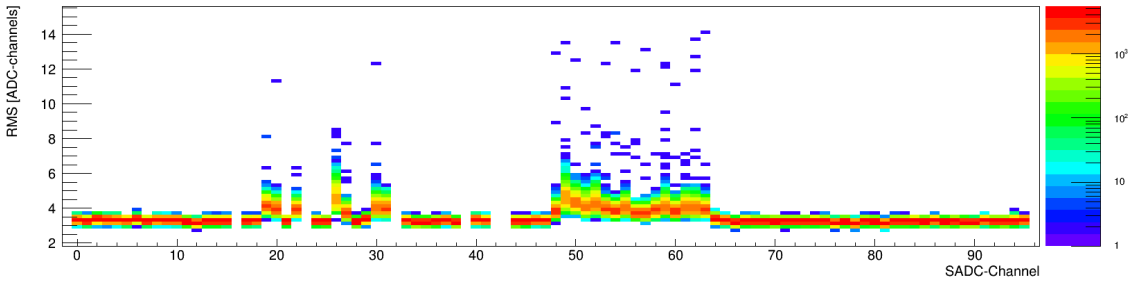
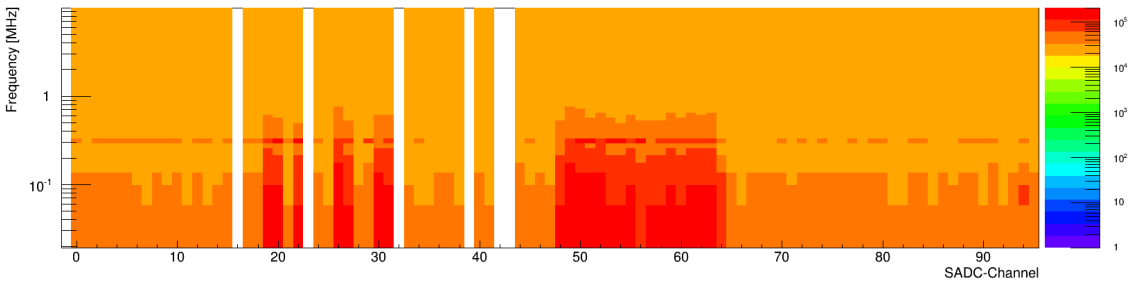


Figure 9.13.: Partial waveform view with the QT DEBUGGING TOOL connected to a CB-SADC. A periodic noise signature is visible in most of the channels.




(a) Baseline widths with data from the $\sigma$ parameter of event-wise Gaussian fits.




(b) Fourier analysis of the waveforms, showing a horizontal band around 310 kHz.

Figure 9.14.: Analysis of the 312 kHz noise oscillation for a selected number of channels, showing one of the tested scenarios where only one slice of the Crystal Barrel Calorimeter was connected to the CAEN low voltage supply. The 23 channels of this slice can be identified by their higher noise. Due to the channel numbering, the section is not contiguous in the plot. *Data analysis performed by Jan Schultes.*

## 9.6. Pile-up Detection and Recovery

The pile-up detection of the CB-SADC firmware was in operation during all beam times in 2018 and 2019, and its performance has been investigated in the scope of the dissertation of J. Schultes [111]. In this section, the results from one of the beam times will be shown. First, the pile-up detection will be discussed and from the angular dependency the expected amount of data for a full readout will be estimated. Secondly, the first steps for a pile-up recovery will be shown.

**Pile-Up Detection** The algorithm for pile-up detection, which is based on determining the ratio between the calculated integral and maximum, has been explained in section 7.5.5. During the beam time, the pile-up flag has been automatically set by the CB-SADC's FPGA on a per-event and per-channel basis, and a waveform sample was automatically transferred to disk if the flag was set. Typical examples of such events can be found in the appendix in Figure D.4.

During the beam times, the distribution of pile-up flagged events could be observed immediately in the *Online Monitor* (a full overview of the *Online Monitor* display of the CB-SADC data can be found in Figure D.5 on page 242). Figure 9.15 shows a 2D-histogram of the events marked by the pile-up detection. The content of the histogram matches the expected behavior. Figure 9.16 shows a 2D-histogram of all events marked by the pile-up detection, the data is sorted by $\frac{\text{integral}}{\text{peak heigth}}$ ratio versus peak height. The observed shape matches the expectation from Figure 7.23 on page 168.

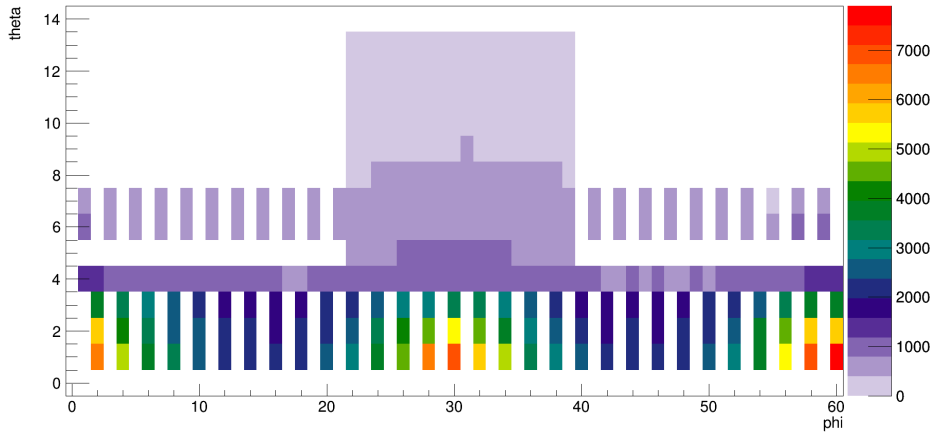



Figure 9.15.: Investigation of the pile-up detection during the October/November 2018 production beam time. The plot is a 2D-histogram spanning over half the Crystal Barrel Calorimeter's solid angle, only the forward half with $\Theta < 14$ is shown. Each entry shows how many events have been flagged as pile-up in each crystal. The matrix reveals the regions of the calorimeter that were connected to the CB-SADC readout. The probability for pile-up is highest in the forward region (small theta) due to the Lorentz boost. With transversally-polarized target, an enhancement of events is visible horizontally ($\Phi = 30; 60$), stemming from $e^+e^-$-production and Compton scattering.

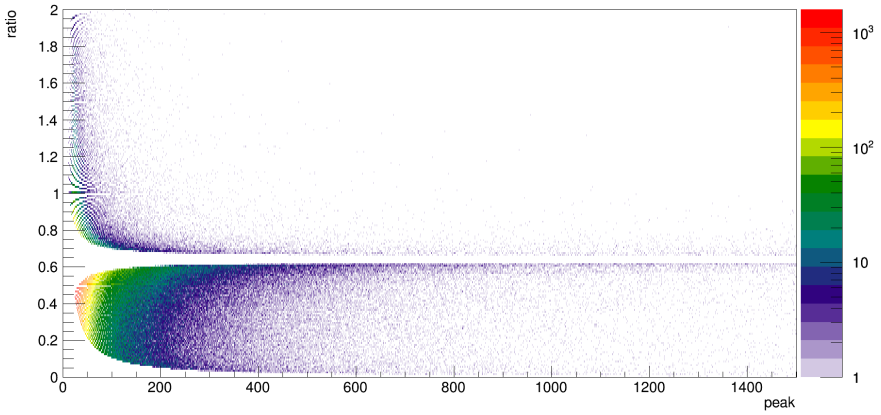


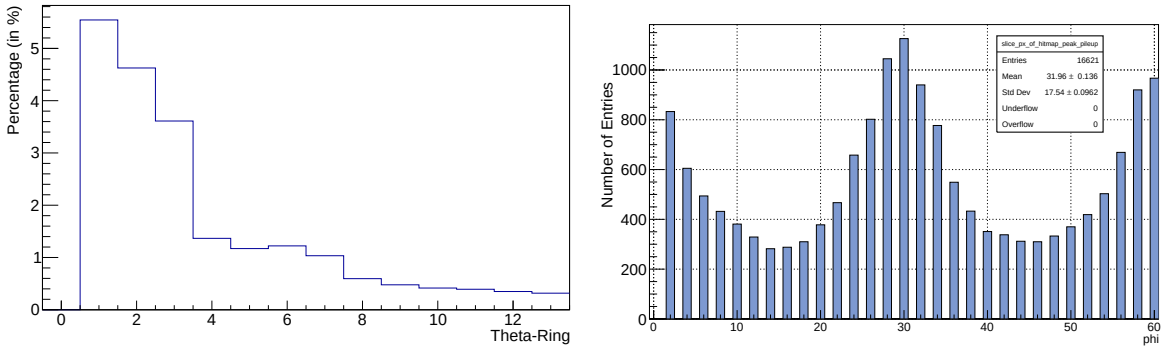

Figure 9.16.: Tagging of pile-up based on the algorithm shown in Fig. 7.23. The x-axis shows the pulse amplitude (in uncalibrated units), the y-axis shows the ratio of $\frac{\text{integral}}{\text{peak heigth}}$. Events without pile-up have an expected ratio of around 0.65 and are cut out in this plots, only events not passing the criterion are shown.

**Data Rate** This calculation aims to assess the data rate caused by the pile-up detection compared to a full waveform readout. For one run, consisting of 2 000 000 events, a full waveform readout with a compression factor of $\approx 3.5$ leads to a file size of

$$2000000 \cdot 6 \cdot 64 \cdot \frac{1}{3.5} \cdot 1040 \text{ byte} \approx 228 \text{ Gbyte}, \tag{9.7}$$

whereas the recorded pileup waveforms, together with a full waveform readout for every 100th event (*waveform prescaler* for debugging purposes), has resulted in a size of only $\approx 6$ Gbyte per run. Filtering the waveforms with this algorithm on the CB-SADC has led to a drastic reduction of the produced data, and at the same time also to an increased readout rate (see section 9.9.1). To conclude from this limited setup (readout of only one quarter of the Crystal Barrel Calorimeter) to the full setup, it is helpful to look at the rate of events with detected pile-up pulses. This is shown in Figure 9.17. It is well visible and expected from the Lorentz boost that the highest pile-up rates are in the forward direction. Since the first four rings (which are most forward) were included in the CB-SADC readout, it can be expected that the total data rate caused by the recorded waveforms will (very roughly) be doubled, once the remaining 3/4 of the Crystal Barrel Calorimeter will be read out by the CB-SADC. This leads to a very coarse estimate of 4000 Gbyte per week of uninterrupted beam time.

**Pile-Up Recovery** J. Schultes has developed algorithms to recover the information from pile-up pulses. First approaches to find an analytical description of the pulse shape were not satisfactory. For this reason, a mean pulse shape has been determined from recorded waveforms, which is scaled in amplitude for the purpose of fitting it to individual pulses. Figure 9.18 shows an example of a successful recovery of the shapes of two overlapping pulses. From this information, the integral, maximum and timestamp of the pulse can be recovered and used in the analysis. With the corrected integral, and therefore energy deposit information, it is expected that a higher number of events will pass the kinematic cut procedure in the analysis of the data. This is part of an ongoing development, and a first analysis taking into account the information from recovered pile-up will be presented in [111].




(a) Pile-up rate depending on the azimuthal angle. The first three rings have twice the solid angle coverage compared to the other rings, leading to a larger step between rings 3 and 4.

(b) Pile-up rate depending on the polar angle, for the most forward ring. Due to the transversally polarized target, an enhancement of events is visible in the horizontal axis (cf. Figure 9.15).

Figure 9.17.: Events flagged as pile-up by the FPGA on the CB-SADC. The rates are normalized to the number of channels read out by the CB-SADC in the regarding ring. *Data analysis performed by Jan Schultes.*

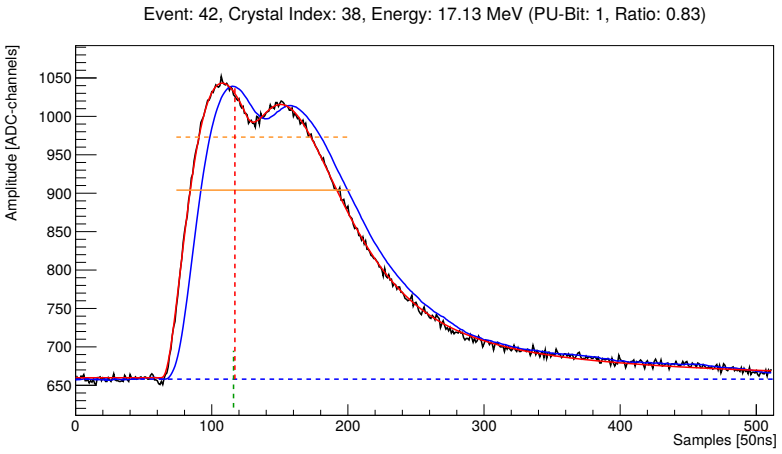



Figure 9.18.: Recorded waveform from an event marked as pile-up. A smaller pulse is sitting on the falling edge of a larger pulse that is correlated to the trigger. The blue dashed line shows the baseline as extracted by the FPGA of the CB-SADC. The blue solid line shows a 16-fold moving average to visualize how the FPGA determines the maximum value and timestamp. The latter is indicated by the red dashed line. The green dashed line shows the timestamp of the constant fraction discriminator. The orange dashed line shows the width of the integration window, its height represents the integral value. The red line is the result of a fit of two pulses, using a scaled version of the mean expected pulse shape. The solid orange line is the corrected integral value, taking into account only the fit for the first pulse. Due to the subtracted second pulse the integral is corrected by roughly 20 %. *Data analysis performed by J. Schultes.*

## 9.7. Pion Decay Analysis

The first production beam time of the Crystal Barrel Calorimeter with APD readout took place in April/May 2018, a second in October/November 2018. Roughly one quarter of the calorimeter was read out by six CB-SADC prototypes in a preliminary setup. After a calibration, it was possible to investigate the electromagnetic decay of a $\pi^0$ meson stemming from the meson photoproduction with the reaction $\gamma p \to p\pi^0 \to p\gamma\gamma$.

The setup, the analysis, and the results will be discussed in the following sections.

### 9.7.1. Setup

For the beam times, six CB-SADCs were installed in a rack close to the Crystal Barrel Calorimeter, as it was proven to be tedious to keep them in the CB-SADC rack. The installation is shown in Figures A.2, A.3 and A.4 (page 222 sq.). In the back of the NIM crate, three CB-SADCs are each connected to a Backplane prototype revision 1.1, and each of the two groups is called a *block*. Both blocks are independently controlled by an instance of the LEvB on the `cbsadcserver`, and each Backplane acts as a *Sync Client* for the DAQ and trigger/sync system.

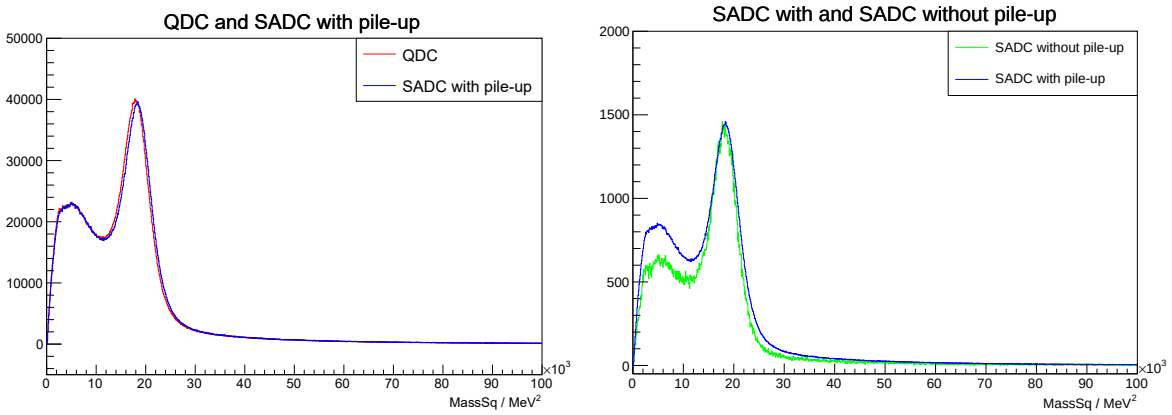
The CB-SADCs were connected with a full coverage in $\phi$ only to the most forward four rings of the Crystal Barrel Calorimeter, plus an additional coherent section covering the forward half in 54° of $\phi$. For the first time, the connection was made with the commercially-available MRJ21 cables from COMMSCOPE (instead of spare cables from the Crystal Barrel Calorimeter), which were already installed as planned for the final setup. The `cbsadcserver` was installed close to the calorimeter in April/May, but has been moved to the server room adjacent to the CBELSA/TAPS control room in October/November (which was the first instance of a 10-Gbit network connection from the experimental hall to the control room).

### 9.7.2. Analysis

This data will be analyzed thoroughly by J. Schultes in the scope of his dissertation [111]. A preliminary analysis was conducted by B. Salisbury in terms of the energy calibration, which is performed by analyzing the $\gamma p \to p\pi^0 \to p\gamma\gamma$ reaction, calibrating the data iteratively to the well-known $\pi^0$ mass[4] of 134.98 MeV. Unfortunately the partial coverage of the calorimeter with the CB-SADCs leads to some drawbacks in the calibration and the analysis: if a cluster's center is in the third or fourth ring, it is likely that a part of the shower extends to the fifth ring, for which there is no CB-SADC information available. This could be handled by employing the so-called energy correction function [60], which takes into account shower losses in edge regions (for example the whole in the forward region below the first ring). In this case it was attempted to calibrate the CB-SADC data for the first four rings by taking into account the data from the QDC for the fifth ring (and the remaining calorimeter). The resulting invariant mass spectra are shown in Figure 9.19.

Figure 9.19a shows the comparison of QDC and CB-SADC when the pile-up flag is ignored, and when only the most forward four rings are taken into account for both digitizers. Ideally an overlap is expected. Figure 9.19b shows a comparison of the CB-SADC data with and without events flagged as pile-up.

[4] This method is described in detail in [190].




(a) QDC and CB-SADC with events that were marked as pile-up.

(b) CB-SADC with and without events that were marked as pile-up. Data scaled for comparison.

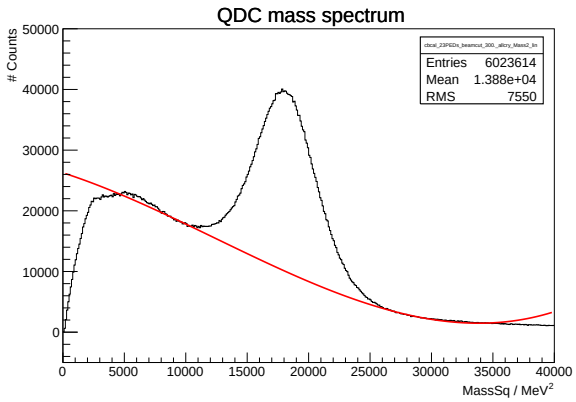
Figure 9.19.: Invariant mass spectra of the $\gamma\gamma$ system from the April 2018 beam time. A peak is clearly visible at the squared mass of the $\pi^0$, $m_\pi^2 \approx 18\,220\,\mathrm{MeV}^2$. The data was selected only if the reconstructed beam energy was above 300 MeV. *Data analysis performed by Ben Salisbury.*

The $\pi^0$ peak is slightly shifted to the right for the CB-SADC, which is due to the non-ideal matching of CB-SADC to QDC data between the fourth and fifth ring. Other than that, the CB-SADC data seems to match the QDC data well. When pile-up events are excluded, the number of events is reduced notably by a factor of more than $\approx 33$. This is unsurprising, keeping in mind the pile-up probability shown in Figure 9.18. A relative suppression of background data is evident compared to the analysis that includes pile-up. Comparing the height of the background peak, it is reduced by more than 20 %, but this number is an indication at best: the pile-up exclusion introduces a bias in the event selection due to the angular dependency of the rate. Only after a true pile-up recovery, the background can be compared quantitatively.

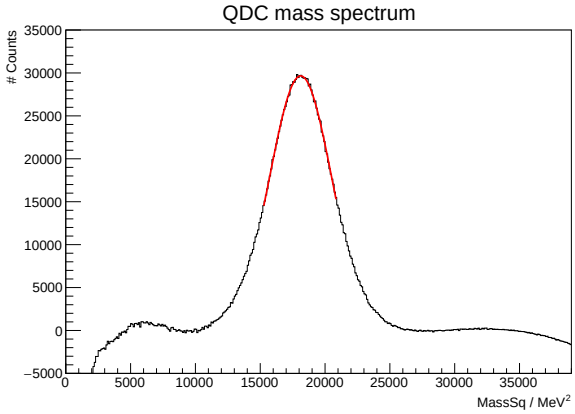
Figure 9.20 shows the analysis of the $\pi^0$ peak after background subtraction for all three scenarios. The extracted parameters are shown in Table 9.2. A difficulty in the analysis was finding a suitable analytic description of the background. In this analysis, a third-order Chebyshev polynomial has been used, after linear and exponential fit produced worse results. Deviations of the data from the background fit are notable in all three instances. The results will be discussed briefly in the next section.

| Setup | $\pi^0/10^3$ | $m_{\pi^0}$/MeV | $\sigma$/MeV | Events/$10^6$ |
|---|---|---|---|---|
| QDC | 883(2) | 134.60(2) | 8.897(2) | 6.02 |
| CB-SADC | 880(2) | 135.97(2) | 8.841(2) | 6.04 |
| CB-SADC w/o pile-up | 33.0(5) | 135.4(1) | 8.63(1) | 0.18 |

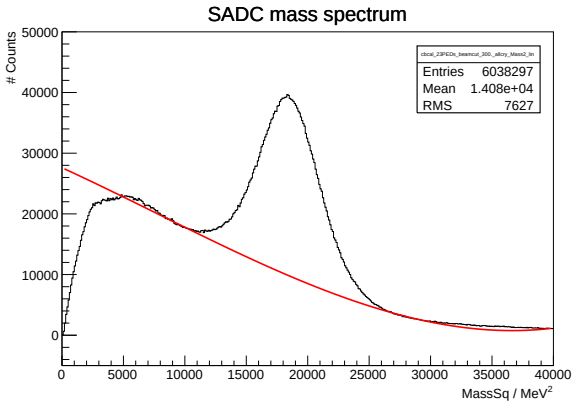
Table 9.2.: Results of the preliminary $\pi^0$ calibration, comparing the CB-SADC (with and without events marked as pile-up by the firmware) to the QDC. Due to the high pile-up probability in the first four rings, the dataset for the last case is drastically reduced. The lower statistics have lead to larger uncertainties.



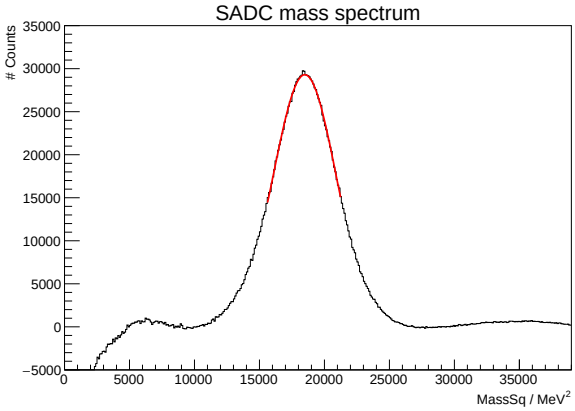

(a) QDC.



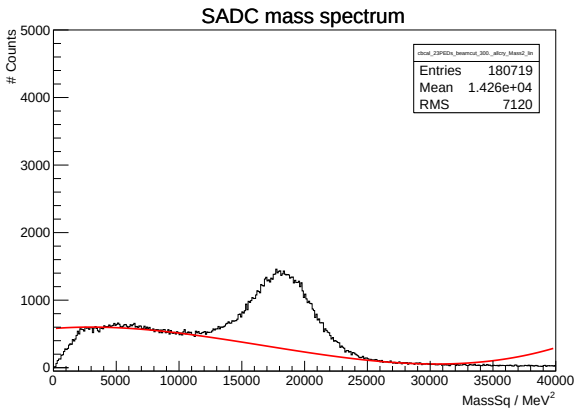

(b) Fit yielding $\sigma = 8.854(2)\,\mathrm{MeV}$.



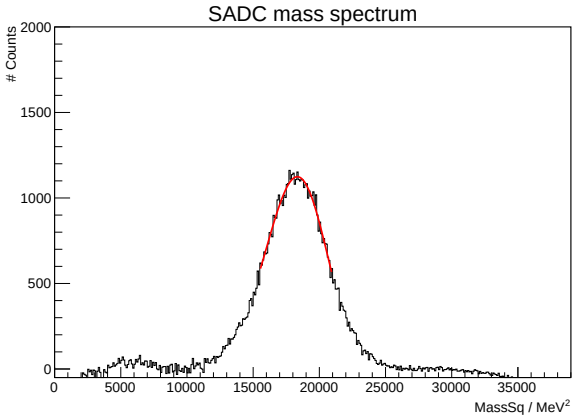

(c) CB-SADC including events flagged as pile-up.




(d) Fit yielding $\sigma = 8.841(2)\,\mathrm{MeV}$.




(e) CB-SADC excluding events flagged as pile-up.




(f) Fit yielding $\sigma = 8.63(1)\,\mathrm{MeV}$.

Figure 9.20.: Analysis of the $\pi^0$ decay width from the April 2018 beam time. Data selection with reconstructed beam energy above 300 MeV. Background fit with third-order Chebyshev polynomial, $\pi^0$ fit with Novosibirsk function. The graphs show the invariant mass spectrum of the $\gamma\gamma$ system, together with a fit after background subtraction. *Data analysis performed by Ben Salisbury.*

### 9.7.3. Discussion

The analysis shows once more that the data taken by the CB-SADCs is consistent with the QDC's data, and at least on par looking at the width of the reconstructed $\pi^0$. The data analysis was challenging because of a number of workarounds that were applied in the calibration of the data, as only constrained number of channels could be read out by the CB-SADC. Further fine-tuning and the analysis with a full CB-SADC readout promise to shed an even better light on the CB-SADC.

The pile-up detection on the CB-SADC's FPGA has been investigated as part of this analysis and was found to work as expected. The study of the pile-up excluded data is particularly interesting, as the spectra show a noticeable suppression of background, and a decreased width of the $\pi^0$. In this preliminary $\pi^0$ reconstruction analysis, the events that contained at least one pulse with detected pile-up were discarded, reducing the dataset substantially. This clean dataset is a good indicator of the possible improvement of the data quality, once all pile-up events are corrected: in the future, especially in the analysis that will be conducted in the scope of the dissertation of J. Schultes [111], the data of the recent beam times will be processed, and the corrected energy information for the events marked as pile-up will be restored to the dataset. One can look forward to this upcoming milestone, which is expected to improve the data quality and statistics considerably.

This concludes the analysis of the extracted energy information, and in the next section the investigation of the timing capabilities of the CB-SADC will be presented.

## 9.8. Timing Capabilities

The determination of timestamps for energy deposits in the Crystal Barrel Calorimeter is the task of the Crystal Barrel TDC, which was introduced in section 2.8.6. The TDC is optimized to recognize an energy deposit very fast due to the primary functionality as *Clusterfinder* and thus the *trigger* source. To fulfill these requirements, the timing filter that is located in the BuffTi module comprises a band-pass filter with a center frequency of roughly 3.2 MHz, resulting in a comparably fast rise time. This also results in a lower SNR and limits the discriminator thresholds.

To account for the higher rate caused by the Lorentz boost, the thresholds vary with azimuthal angle and are higher in the forward direction, starting with $\approx 11.3\,\mathrm{MeV}$ for the first ring, going down to $\approx 3.9\,\mathrm{MeV}$ for the fifth ring and above[5]. This threshold is at the same time the lowest energy for which it is possible for the TDC to determine a timestamp.

The CB-SADC itself also has the ability to extract such an information from the data, as was discussed in section 7.5.4. Due to the different shaping and therefore less steep signals, and also the much slower sampling rate compared to the TDC, it is expected that the CB-SADC's timing information has a worse resolution than the TDC's. It has been investigated if the CB-SADC can still contribute in the timing domain, and possibly complement the TDC's data for the low energy regime. The capabilities of the CB-SADC have been investigated with the data of the April 2018 beam time, with the setup as described in section 9.7.1.

### 9.8.1. Analysis

In Figure 9.21, the time difference between the timestamp of the tagging hodoscope and the CB-SADC's Constant Fraction Discriminator (CFD) algorithm is plotted as a 2D-histogram. The data is kinematically constrained, such that only photons stemming from a $\pi^0$ production are selected, which greatly reduced the background data. A second selection criterion is that no pile-up should have been detected by the CB-SADC. A prominent band with a mean value of roughly $-2200\,\mathrm{ns}$ is visible. This time corresponds to the time difference between the deflected electron from the Bremsstrahlung process hitting the tagging hodoscope's scintillator bar/fiber, and the shaped pulse reaching its maximum, which results in a zero crossing of the CFD algorithm. More negative times thus correspond to later-detected pulses. The data is shown with linear (Figure 9.21a) and logarithmic (Figure 9.21b) energy scale.

**Low Energy Shift** In Figure 9.21b an unexpected behavior is noticeable in the region below 10 MeV: it seems that the timestamp moves slightly to earlier times, but then back to the expected band towards energy bins of 1 MeV to 2 MeV. To investigate this behavior, the data was sliced into energy bins of 0.5 MeV and Gaussian fits have been applied to the histograms. This is shown in Figure 9.22. Indeed it is visible that the center of the distribution moves by more than 40 ns in the observed range, and it can also be observed that the width does not continuously increase towards lower energies. Rather it increases up to the bin of 2.5 MeV to 3.0 MeV, but then decreases again. The two effects might be connected, since the movement of the timestamp in one energy bin also results in a wider distribution. Without this movement the distributions would likely appear more narrow.

[5] In the very backward direction, the thresholds are again higher to account for the electromagnetic background from the region before the beam enters the Crystal Barrel Calorimeter.

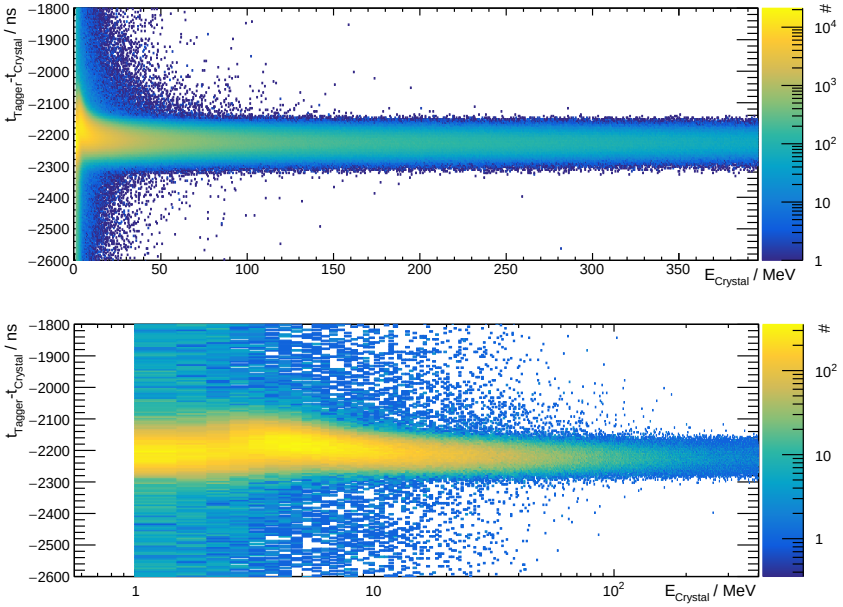
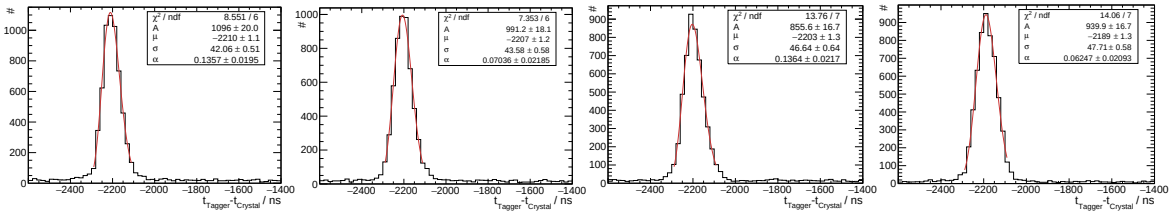



Figure 9.21.: Distribution of the CB-SADC's timestamp generated by the CFD algorithm. The x-axis is the associated energy of the event (in this case from the QDC) and is shown with linear (top) and logarithmic (bottom) scale. The y-axis shows the difference of the tagging hodoscope timestamp to the CB-SADC's CFD timestamp. Only data from the most-forward ring of the Crystal Barrel Calorimeter is included. *Data analysis performed by Nils Stausberg.*

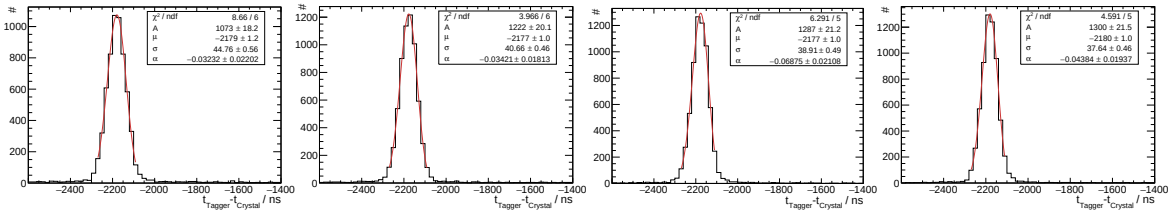


(a) 1.0 MeV to 1.5 MeV with $\sigma \approx 42.1$ ns and $\mu \approx -2210$ ns.

(b) 1.5 MeV to 2.0 MeV with $\sigma \approx 43.6$ ns and $\mu \approx -2207$ ns.

(c) 2.0 MeV to 2.5 MeV with $\sigma \approx 46.6$ ns and $\mu \approx -2203$ ns.

(d) 2.5 MeV to 3.0 MeV with $\sigma \approx 47.7$ ns and $\mu \approx -2189$ ns.



(e) 3.0 MeV to 3.5 MeV with $\sigma \approx 44.8$ ns and $\mu \approx -2179$ ns.

(f) 3.5 MeV to 4.0 MeV with $\sigma \approx 40.7$ ns and $\mu \approx -2177$ ns.

(g) 4.0 MeV to 4.5 MeV with $\sigma \approx 38.9$ ns and $\mu \approx -2177$ ns.

(h) 4.5 MeV to 5.0 MeV with $\sigma \approx 37.6$ ns and $\mu \approx -2180$ ns.

Figure 9.22.: Gaussian fits to the data shown in Figure 9.21. The data is sliced into energy bins of 0.5 MeV width, only the fits for the region $E < 5$ MeV are shown. *Data analysis performed by Nils Stausberg.*

The behavior has been investigated to some extent. No reason was found so far to blame this on kinematic effects or a biasing in the event selection. At the moment it is believed that this effect is caused by the algorithm of the CFD algorithm implemented in the CB-SADC's FPGA. The criteria for the timestamp validation as described in section 7.5.4 may indeed be responsible, considering that in this low energy regime the baseline noise of $\approx 150\,\mathrm{keV_{RMS}}$ will play a role. A final verdict can only be made after the next beam time, when it is planned to vary the parameters to investigate their influence on the observed timestamp distribution. In the mean time, it will be attempted to reproduce the behavior with simulations of the algorithm, using the recorded waveforms from the beam times.

**Comparison to TDC** The widths of the energy-binned distributions of the timestamps, calculated from fits as shown in Figure 9.22, are a good measure for the accuracy of the timestamp determination. Similarly, the timestamp distribution of the TDC data can be analyzed, and then compared to the CB-SADC data. This has been done by N. Stausberg and is shown in Figure 9.23. The analysis is shown for the four most forward rings, where the discriminator thresholds were set to (11.3, 8.8, 7.1 and 5.0) MeV. Starting with the fifth ring, the threshold for most of the remaining calorimeter crystals is set to 3.9 MeV. The corresponding plot is shown in Figure D.7.

The analysis reveals that the CB-SADC's timestamp has a better resolution than the TDC' timestamp at energies of roughly

$$E \lesssim E_{\mathrm{thr}} + 2\,\mathrm{MeV}, \tag{9.8}$$

e.g. for the most-forward ring, the TDC threshold is at 11.3 MeV, and the resolution of the CB-SADC's timestamp outperforms the TDC at around 13 MeV and below. At energies above 50 MeV, the resolution of the CB-SADC settles at around $\sigma_{\text{CB-SADC}} \approx 25\,\mathrm{ns}$. In this region it is no match to the TDC, which provides timestamps below $\sigma_{\mathrm{TDC}} \approx 5\,\mathrm{ns}$.

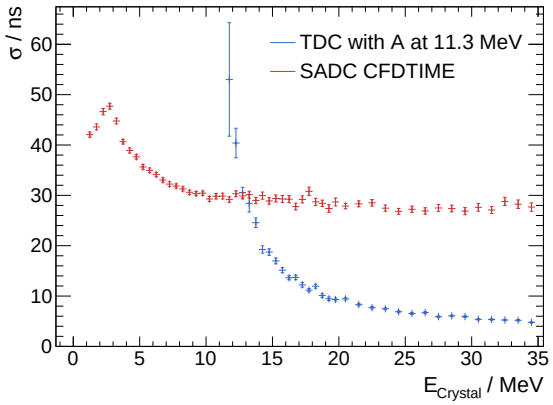
The plots show again the peculiar behavior as described in the previous paragraph: towards lower energies, the distribution appears broader, but then gets narrower again.

### 9.8.2. Discussion

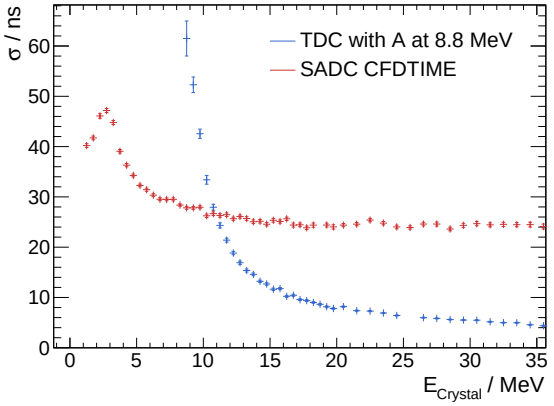
The analyses have shown that the timestamp determination of the CB-SADC works well. The timestamps have been investigated down to an energy of 1 MeV, where the resolution was still well below 50 ns, improving to 25 ns for higher energies. It was shown that the CB-SADC's timestamps outperform the TDC roughly 2 MeV above its threshold. With this, one of the demands for the SADC upgrade, the *Sub-Threshold Timestamps* (cf. section 3.1), is indeed fulfilled. This could be helpful for determining, for example, whether a low-energetic energy deposit at the edge of a cluster does still belong to this cluster: a time difference to the cluster's central energy deposit might hint that the deposit in fact stems from an uncorrelated event. In the following paragraph, possible improvements will be discussed.

**Improvements** The shift of the timestamps for energies $< 10\,\mathrm{MeV}$ has to be investigated further in the next beam times and in simulations of the algorithm. Hopefully with a tuning of the parameters or improvements to the algorithm, a more continuous behavior can be achieved in the future. Furthermore, it is possible to improve the overall resolution of the

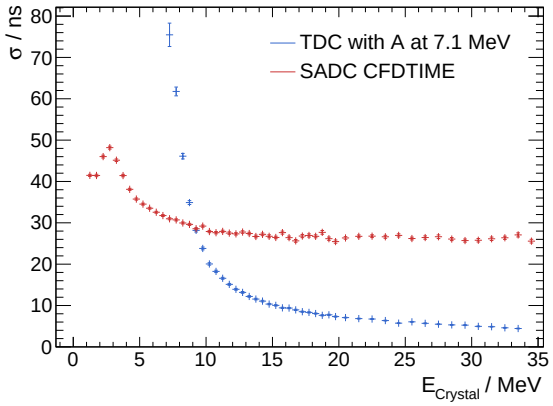
timestamps: the reference signal for the measurement of the timestamps is generated by the trigger electronics with a granularity of 5 ns, which the TDC can resolve. The CB-SADC on the other hand currently samples this signal with 80 MS/s resulting in a granularity of 12.5 ns. The signal could be sampled with a clock with at least 200 MHz to improve the resolution. Preparatory work for this is done and an algorithm exploiting an input deserializer primitive of the KINTEX-7 for highest sampling rates was successfully simulated, and might be implemented in a beam time in late 2019. In addition, the CFD algorithm could be optimized using a linear regression around the zero crossing, from which a more exact timestamp could be created due to the reduced influence of the signal's noise. This is currently being investigated and will possibly also be attempted in one of the next beam times.



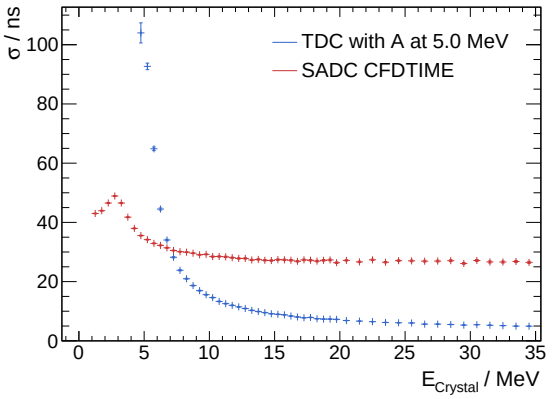

(a) First ring. Crossing around 13 MeV.




(b) Second ring. Crossing around 11 MeV.




(c) Third ring. Crossing around 9 MeV.




(d) Fourth ring. Crossing around 7 MeV.

Figure 9.23.: Comparison of the standard deviation of the timestamps of CB-SADC and TDC. The data is binned in energy intervals of 0.5 MeV, and has been analyzed for different azimuthal angles (corresponding to the rings of the calorimeter) to account for the different discriminator thresholds of the TDC. The crossing point of both curves shows below which energy the CB-SADC can deliver a better resolution than the TDC. In addition, the plots are shown in the appendix in Figure D.6 with logarithmic energy scale. *Data analysis performed by Nils Stausberg.*

## 9.9. Deadtime and Efficiency

In this section, the readout performance of the CB-SADC will be discussed. The theoretical and real readout rates, which were mentioned throughout the thesis, will be summarized in the following. It will be investigated whether the performance is sufficient for the application in the CBELSA/TAPS experiment, and how it can be further optimized.

### 9.9.1. Readout Rates

The readout rates have been investigated under different circumstances as summarized in Table 9.3. Three readout modes were tested. First with only the feature extraction active, secondly with feature extraction and sample extraction for every 100th event, and lastly feature extraction and sample extraction for all events. The **theoretical limits** have been calculated in equation 7.25 on page 170 and equation 7.27 on page 172. In section 8.3 it was then shown how one CB-SADC performs under laboratory conditions and with the implementation of the *burst* mechanism. Two further scenarios were investigated together with the Crystal Barrel DAQ, once with six prototypes being read out, and once with only three. An interpretation of the results will be presented in the following paragraphs.

| Configuration | theoretical limit | measured /w *Burst* (1 CB-SADC) | DAQ (3 CB-SADCs) | DAQ (6 CB-SADCs) |
|---|---|---|---|---|
| FE only | 75.0 kHz | 35.1 kHz | 10.0 kHz | 10.0 kHz |
| FE and SE (PS100) | 61.4 kHz | N/A | 10.0 kHz | 6.54 kHz |
| FE and SE (full) | 3.24 kHz | 2.77 kHz | 1.75 kHz | 1.31 kHz |

Table 9.3.: Readout rates of the CB-SADC under different conditions. *PS100* stands for *prescaler=100*, meaning that for every 100 events, the waveforms are sent for every channel. The DAQ rate is averaged over a run of 5 min and was taken with the trigger `vme_clock_viel_nospill.xml`, which supplies 10 kHz, uniformly distributed. Faster readout rates could not be tested with the DAQ. Waveform compression was activated, zero suppression was not implemented yet.

**Rate with *Burst*** It can now be investigated why the real readout rates differ from the theoretical ones. First of all, the theoretical rates are calculated without taking into account the burst mechanism. The burst mechanism ensures the reception of all packets by the LEvB. After each burst of 32 packets, the LEvB will send a *clear buffer* signal to the CB-SADC, allowing it to send the next 32 packets (cf. section 7.4.5). This adds an additional delay caused by the round trip time of the network packet and the time needed by the LEvB to check the integrity and completeness of the packets.

It is now assumed from the theoretical limit of 75 kHz that one event requires a processing time of $T_{\text{event}} = 13.328\,\mu\text{s}$, and that after one event the trigger signal for the new event is immediately present, which is approximately ensured with a self-trigger at the noise of the signal. The delay caused by the burst mechanism can be calculated based on the measured

readout rate of $R_{\text{real}} = 35.1\,\text{kHz}$.

$$\frac{\text{Delay from } \textit{Burst}}{\textit{Bursts} \text{ per second}} = \frac{1 - T_{\text{event}} \cdot R_{\text{real}}}{R_{\text{real}}/32} \tag{9.9}$$

$$= \frac{1 - 13.328\,\mu\text{s} \cdot 35.1\,\text{kHz}}{35.1\,\text{kHz}/32} \tag{9.10}$$

$$\approx 485\,\mu\text{s} \tag{9.11}$$

$$\tag{9.12}$$

With the setup as shown in Figure 9.2 on page 190, it was possible to investigate the trigger signals during this test with an oscilloscope. Figure 9.24 shows the *trigger/sync* signals. A structure of 32 consecutive events can be identified, followed by a *Busy* phase longer than 500 µs that is introduced by the delayed acknowledgment of the 32-packet burst by the LEvB. This seems to be roughly consistent with the previously calculated number of 485 µs. It has to be noted that much shorter delays than this have been observed, and it seems to be a certain condition that causes the LEvB to exhibit it to that extent. Currently this cannot be investigated further due to the installation of the prototypes in the experimental hall.


Figure 9.24.: Oscilloscope screenshot of the trigger/sync signals during a high-rate performance test with a single CB-SADC, limited only by the Ethernet's and `cbsadcserver`'s performance. On the top, the trig (*Event*) and rst (*Sysreset*) signals of the trigger/sync system can be seen; the bottom shows the individual *Ok* and *Busy* signals of each of the two FPGAs. In the middle, the 5-bit *Buffer Number* is shown (the white areas show the quick transitions with each trigger).

Figure 9.25 shows a counter example, a scenario which was recorded during the beam time by accessing the debug interface of the Backplane. It shows a busy phase of $\frac{15490}{50\,\mathrm{MHz}} \approx 310\,\mu\mathrm{s}$. This is the time needed until the three connected CB-SADCs (each with two network interfaces) have received the *clear buffer* packet of the . This proves that the previously observed $\approx 500\,\mu\mathrm{s}$ are not a lower limit to the achievable performance of the burst protocol.

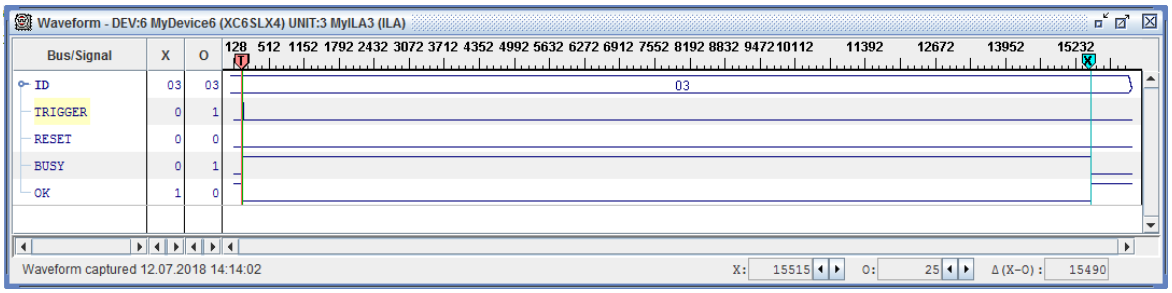



Figure 9.25.: Screenshot of the debug interface of the Backplane, showing the signals of the trigger/sync system for a whole *block* of three CB-SADCs. A busy period with a length of 15490 samples (at 50 MHz) can be observed.

**Rate with DAQ** The readout rate has also been investigated in connection with the DAQ, with the setup as described in section 9.7.1. Since the six CB-SADCs are divided into two *blocks*, the rates could be determined with three or six units. With only the feature extraction activated, the readout rate was limited by the DAQ to 10 kHz, and it may be assumed that the setup can perform faster. With the sample prescaler activated, which will send the full waveforms for every 100th event for debug purposes, the rate is still limited by the DAQ for one block. For two blocks, the rate drops to 6.54 kHz, and with sample extraction enabled for all events even to 1.31 kHz or 1.75 kHz, for six or three CB-SADCs respectively.

At the time, it could not be pinned down what exactly is limiting the readout rate. Possible reasons, which will be discussed in the following, are:

- Storage limitations of the `cbsadcserver`
- Network limitations between CB-SADCs and switch
- Network limitations between switch and `cbsadcserver`
- CPU/Memory limitations of the `cbsadcserver`
- Limitations of the DAQ
- Other

The maximum possible throughput to the solid state disks (SSDs) was measured in section 9.1.1 to be around 400 Mbyte/s. During full waveform recording with `wavezip` compression, a write rate of roughly 200 Mbyte/s was observed (see Figure 9.26). Only if the compression is turned off, the limit could be reached in this configuration. This seems to exclude limitations of the storage bandwidth as the cause of the observed discrepancy.

```
Total DISK READ: 3.77 K/s | Total DISK WRITE: 139.54 M/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
 1288 be/4 daq-tr      0.00 B/s   77.93 M/s  0.00 %  1.53 % cs_cbsadc2 --cmdport 4095 -f /home/daq-tr/Config/cbsadc/cbsadc2
 1260 be/4 daq-tr      0.00 B/s   77.27 M/s  0.00 %  0.76 % cs_cbsadc1 -f /home/daq-tr/Config/cbsadc/cbsadc1_nowave.xml -d
 1255 be/4 root        3.77 K/s   46.05 M/s  0.00 %  0.43 % [flush-8:16]
```

Figure 9.26.: Analysis of the disk load during readout of six CB-SADCs with feature extraction and sample extraction enabled for all channels and events. The LEvB processes can be identified as `cs_cbsadc1/2`. The *flush* process shows parts of the disk write load that has been cached.

The network limitations between the CB-SADCs and the switch can in principle be excluded, since they have been considered in the calculation of the theoretical limitations (cf. Table 9.3). Figure 9.27 shows the network load during the readout of both blocks with sample extraction activated for all events and channels. Indeed one can see that the average throughput per FPGA is below 300 Mbit/s, using only about one third of the net capacity of the 1 Gbit/s Ethernet connection. This is dominantly limited by the *burst* mechanism). In sum, the throughput is around 3.5 Gbit/s, which amounts to 35 % of the 10 Gbit/s Ethernet connection between the switch and the `cbsadcserver`. Therefore, it would seem that the network is not limiting, either.

Figure 9.28 shows a screenshot of the CPU/memory load displayed with `htop`. Although the system is undoubtedly under quite some load on all 16 logic cores, it was tried to exclude this as a limitation by deactivating the `wavezip` waveform compression (cf. section 8.4). Afterwards the load had dropped significantly, yet the rate limitation has stayed unchanged. The memory load, on the other side, is mostly uncritical, as none of the procedures used in the LEvB are heavy in memory use. This leaves most of the memory to be used as a buffer by the kernel, effectively de-randomizing the load to the disks.

Limitations of the DAQ should in fact not play a role when only looking at the sample extraction packets, as they are not processed by the DAQ system. After the LEvB receives the data from the CB-SADCs, only the information from the feature extraction packet is forwarded to the DAQ system, and the sample extraction information is stored locally on the `cbsadcserver`. From this point of view, the DAQ should not be responsible for a limitation.

### Conclusion

The cause for the observed limitations could not be determined with the presented investigations, and further investigations were hardly possible at the time because of the setup in the experimental hall. Since the extent of the limitation was not critical for data taking, it is planned to take care of the issue after the installation of the final CB-SADC setup, such that the prototypes can be investigated in the laboratory in a more systematic manner. At the moment it seems that the network connection(s) and the CPU/memory/storage still have some headroom. The reason for the observed limitations may be hidden in the LEvB's architecture or the data processing on the *Event Saver*, or even a reason that was not in the focus of the presented investigations.

It has been shown that the CB-SADC readout can nonetheless reach much faster rates compared to the QDC readout. The presented back-end hardware (`cbsadcserver`, network switches) will be easily scalable to the full setup with 24 CB-SADCs.

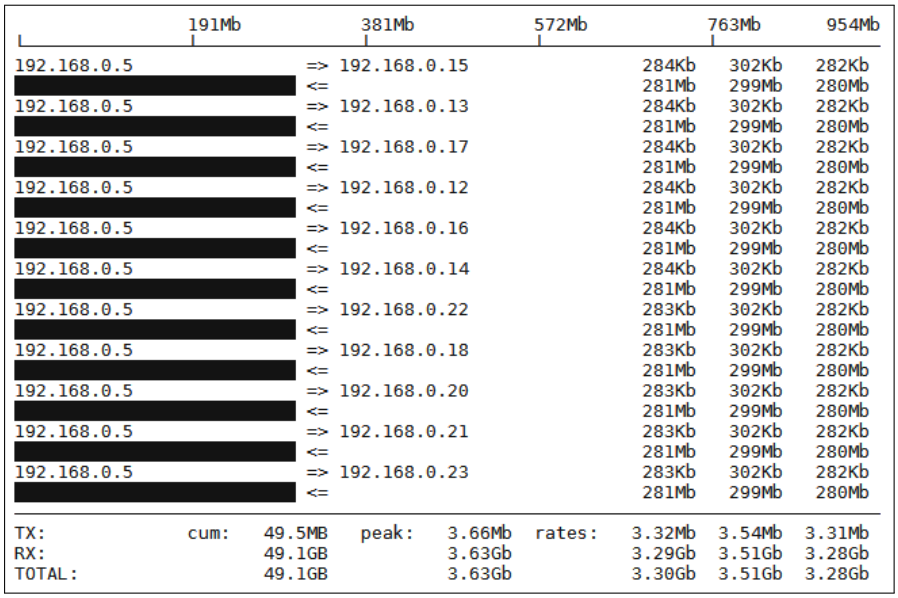



Figure 9.27.: Analysis of the network throughput with `iftop` during the readout of six CB-SADCs with feature extraction and sample extraction enabled for all channels and events. Each of the CB-SADCs can be identified by its two IPs (ranging from `192.168.0.12` to `*.23`, one is not shown). The peak receive (RX) data rate is 3.63 Gbit/s, the last three columns show rates averaged over (2, 10 and 40) s.

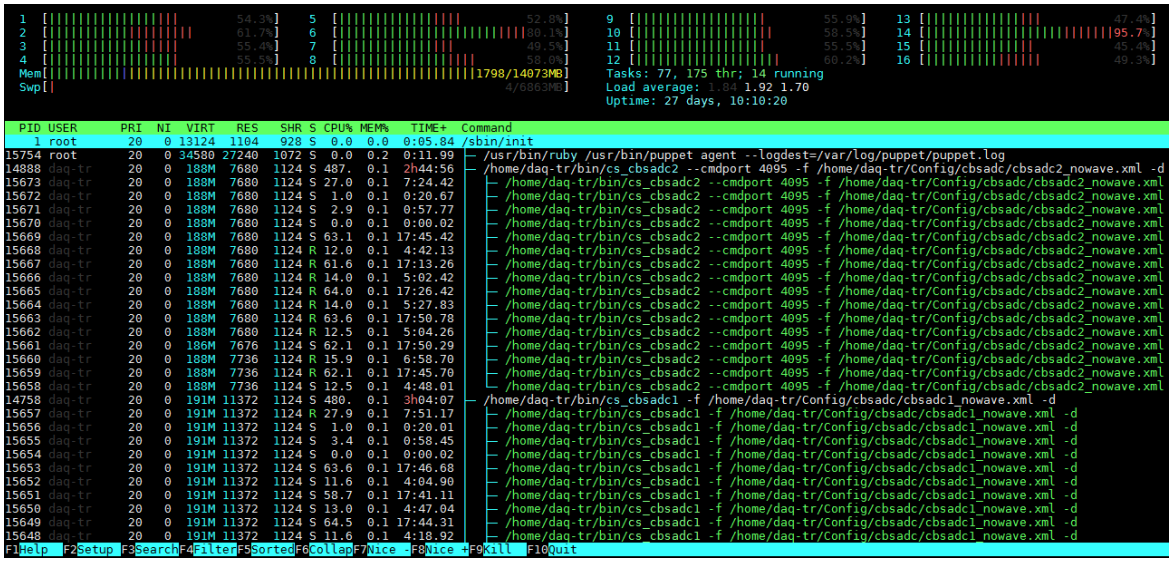



Figure 9.28.: Analysis of the CPU and memory load with `htop` during readout of six CB-SADCs with feature extraction and sample extraction enabled for all channels and events. The LEvB processes can be identified as `cs_cbsadc1/2`. It is nicely shown that the processes are extensively threaded for waveform compression and data handling.

# 10. Summary and Outlook

The CBELSA/TAPS experiment at the electron accelerator ELSA in Bonn investigates the spectrum and the properties of baryon resonances to gain a better understanding of the strong interaction in the non-perturbative regime. A photon beam with energies up to 3.5 GeV, created through Bremsstrahlung from the electron beam, impinges on a target where it can excite the nucleons. The photon beam and the target can both be polarized, which is essential to be able to disentangle the broad and overlapping excitation spectrum, using a partial wave analysis. The main electromagnetic calorimeter, the Crystal Barrel, is very well suited to detect photons from neutral meson decays.

To achieve a high trigger efficiency for uncharged final states, the Crystal Barrel has recently been fully integrated into the experiment's first-level trigger, with a successful commissioning phase in 2017, followed by several production beam times. In the scope of this upgrade, the PIN-photodiode and photomultiplier readouts of the 1320 CsI(Tl) crystals were replaced with a modern avalanche-photodiode readout. A new TDC (Time-to-Digital Converter) now measures the arrival time of the signals, and the embedded Clusterfinder counts the clustered energy deposits. From this information, the new trigger signal is generated. The measurement of the energy that is deposited in the Crystal Barrel's CsI(Tl) crystals is still performed with the integrating Fastbus ADCs (Analog-to-Digital Converters) that were already used before the trigger upgrade. Although their resolution suits the needs of the experiment, their readout speed is now a limiting factor of the CBELSA/TAPS experiment. Due to the dead time introduced by the ADCs' conversion time and the data transfer, the theoretically achievable rate is below 2 kHz. In addition, the integrating Fastbus ADCs cannot detect overlapping pulses in their integration window. Such *pile-up* signals are therefore assigned a wrong energy information and add an error to the recorded event.

Within this thesis, the successor of the integrating Fastbus ADCs has been developed and tested. The so-called CB-SADC (Crystal Barrel Sampling-Analog-to-Digital Converter) was adapted from a prototype design for the PANDA experiment. It features 64 channels with a sampling rate of 80 MS/s and a resolution of 14 bit. The data processing is performed with Xilinx Kintex-7 FPGAs and transferred to a computer via 1 Gbit/s Ethernet links.

In the first step, the design of the PANDA-SADC was modified to suit the needs of the Crystal Barrel Calorimeter. The form factor was changed to fit into a NIM cassette, and a power supply was designed, produced and tested. The CB-SADC was equipped with several sensors and was integrated into the monitoring system of the experiment. In collaboration with bachelor and master students, an analog front end for signal processing, and a module for distribution of trigger, sensor and programming signals were developed, produced and tested. In the second step, the firmware for the FPGAs of the CB-SADC was developed and several methods for the extraction of data from the waveforms were implemented. Apart from the determination of the deposited energy from an integral or the peak height, timestamps are created with a constant fraction discriminator algorithm. In addition, algorithms for the event-wise baseline subtraction and the detection of pile-up signals were developed.

Two prototype revisions of the CB-SADC were tested extensively in the laboratory. Afterwards, six prototypes were installed in the experimental hall and connected to the forward region of the Crystal Barrel Calorimeter, where the highest hit rates occur. The CB-SADCs were successfully integrated into the experiment's data acquisition system (DAQ). While the integrating Fastbus ADCs require Fastbus and VME architecture for their readout, and long cables to delay the analog signal, the CB-SADC's readout is fully based on 1 Gbit/s Ethernet, and the CB-SADC can delay the signal with storage elements inside the FPGAs.

In the experimental hall, the CB-SADCs have proven their capabilities in the commissioning phase of the upgraded CBELSA/TAPS experiment and the following production beam times. Readout rates of 10 kHz have been obtained in connection with the DAQ, with a theoretical limit of 75 kHz for non-acknowledged data transfer. This puts the CB-SADC far ahead of the integrating Fastbus ADC, and shows that it can eliminate the Crystal Barrel Calorimeter as the slowest component of the experiment.

In the analysis of the data that was recorded during the beam times, the CB-SADC has also fulfilled the expectations. The calibration of the data, using the well-known mass of the $\pi^0$ meson, was successful and has shown the expected consistency with the data of the integrating Fastbus ADC. When signals with detected pile-up were excluded, the distribution became more than 2 % narrower, which indicates the performance that can be expected after the correct information is recovered from the pile-up pulses. The analysis of the timestamps supplied by the CB-SADC has confirmed that it can complement the information from the Crystal Barrel's TDCs. Although the TDC's resolution is below $10\,\mathrm{ns_{RMS}}$ for energies down to $\approx 20\,\mathrm{MeV}$, it is limited by the threshold of its discriminators. This is especially true in the forward region where the threshold needs to be increased due to high hit rates. Here the CB-SADC can play to its strength: with its constant fraction discriminator at a much lower threshold, it can supply timestamps down to 1 MeV with a resolution of $< 50\,\mathrm{ns_{RMS}}$, where the TDC delivers worse timing resolution or no timestamps at all.

**Outlook** Exciting results can be expected from the analysis of the data taken with a partial CB-SADC readout in the most forward section of the Crystal Barrel [111]. For the first time, a correction of pile-up events will be possible and an event-wise baseline subtraction can be performed. Therefore, based on the presented studies, it can be expected that the CB-SADC will provide a better data quality than the integrating Fastbus ADC, with a higher number of reconstructed meson decays and an overall lower background. The already impressive readout speed is expected to increase further after optimization of the back-end software (local event builder).

The 24 modules necessary for the complete setup have arrived and have undergone a testing procedure together with the dedicated power supplies and the analog front ends. In a current master thesis [141], the crate controller for the CB-SADCs is being developed. It is the last important element of the project, and it can be expected that the full CB-SADC readout will be installed by the end of 2019. After the readout of the Crystal Barrel Calorimeter is then no longer the limiting factor, the MiniTAPS detector will be one of the next candidates for an upgrade of the readout electronics. It is conceivable to also use the CB-SADC to replace the existing readout. Since the signals of its $BaF_2$ scintillator crystals are much shorter, a modification of the analog front end will be necessary. The feasibility of this project is planned to be investigated in the near future.

# A. Pictures

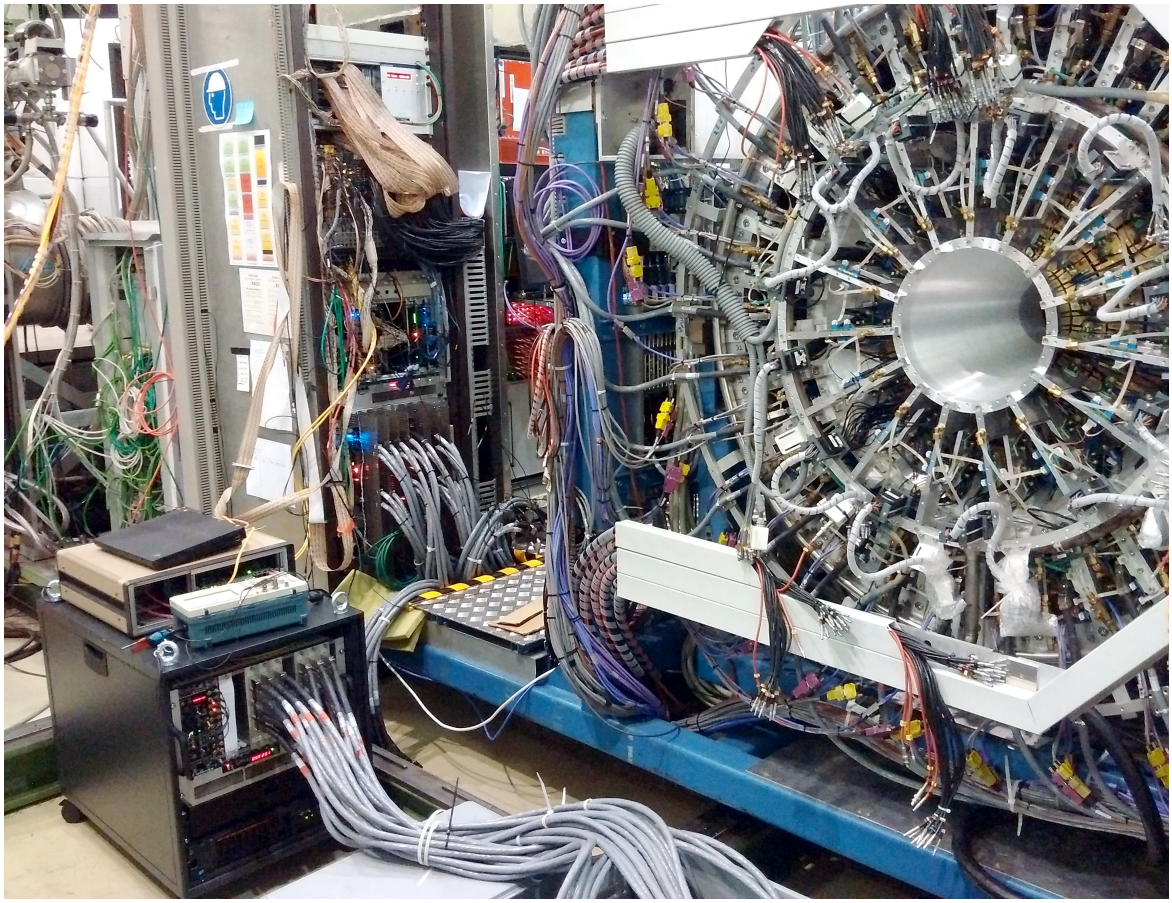


Figure A.1.: The CB-SADC rack, for the first time at the CBELSA/TAPS experiment. Six CB-SADCs are installed in the NIM crate and connected with MRJ21 cables to roughly 50 % of the forward half of the Crystal Barrel Calorimeter that can be seen on the right side of the picture. In this revision, there was no space for the horizontal VME crate in the rack, which is why it is installed in a temporary enclosure on top. An oscilloscope is installed temporarily to inspect the trigger signals. The gray MRJ21 cables are spare cables for the cabling from the Crystal Barrel Calorimeter to the BuffTi, since at the time the cables dedicated for the CB-SADCs were not yet purchased.

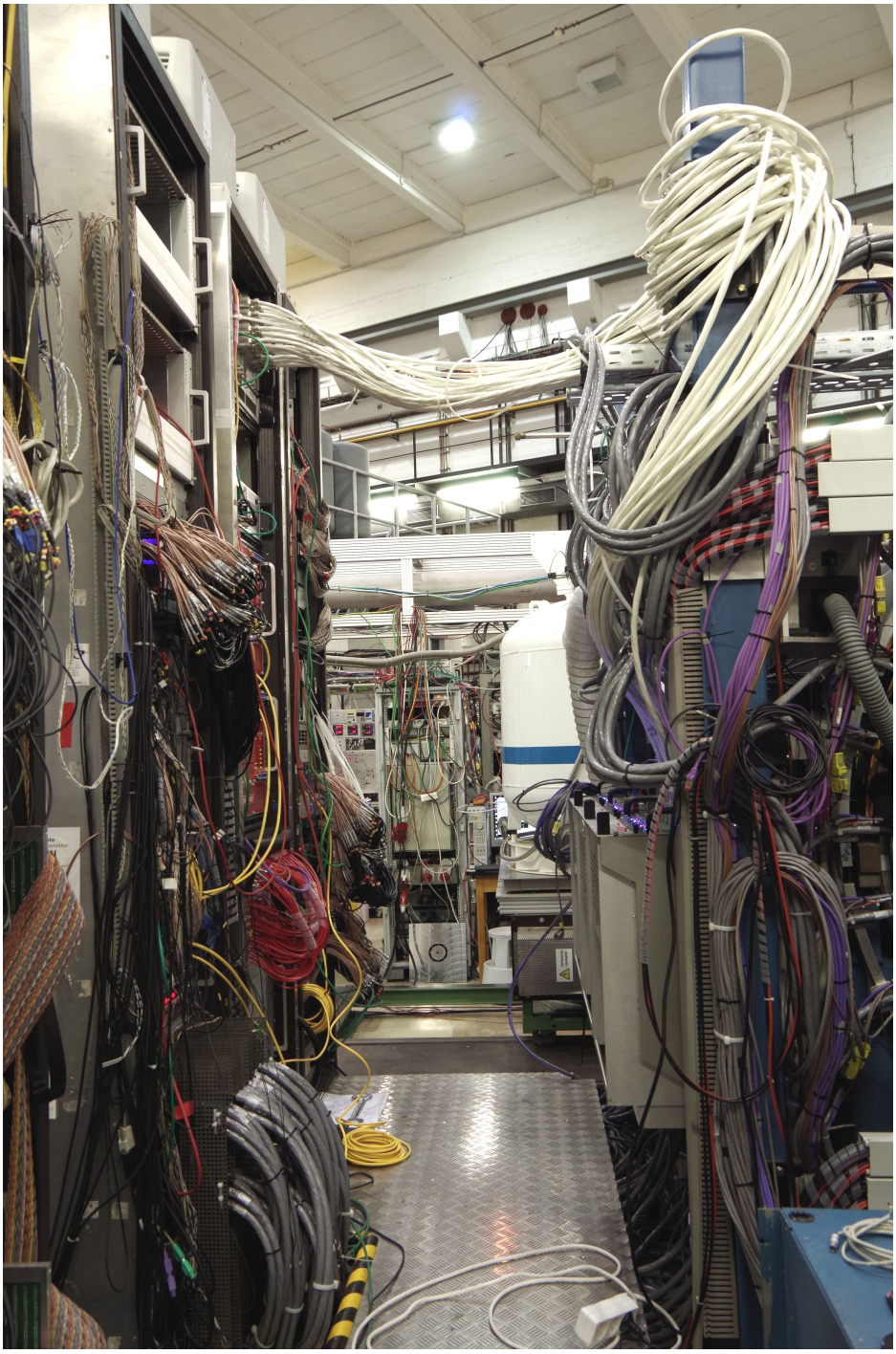


Figure A.2.: Six CB-SADCs installed in the NIM crate in the racks of the CBELSA/TAPS experiment (top left), and connected with MRJ21 cables to the most-forward four rings of the Crystal Barrel Calorimeter. The BuffTi modules can be seen in the tilted NIM crates at the right. The cabling is temporary, since at the time the cables of shorter length were not yet purchased.

Figure A.3.: Close-up view of the six CB-SADCs in the NIM crate. A total of 24 MRJ21 cables, carrying 16 channels each, is installed. The NIM module on the very left is the USB-to-network bridge that was built in the scope of this thesis to remotely connect to the USB-JTAG programmers that are installed together with the Backplanes in the back of the NIM crate.

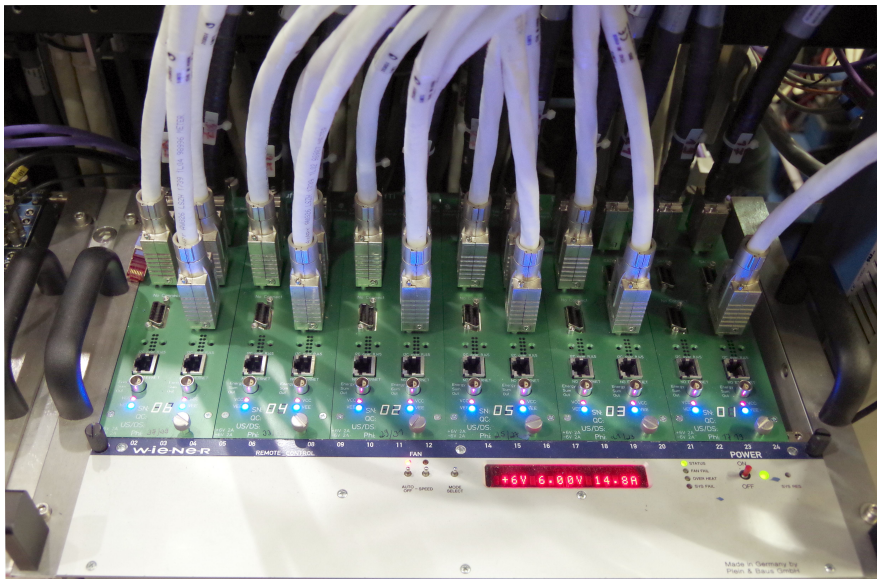
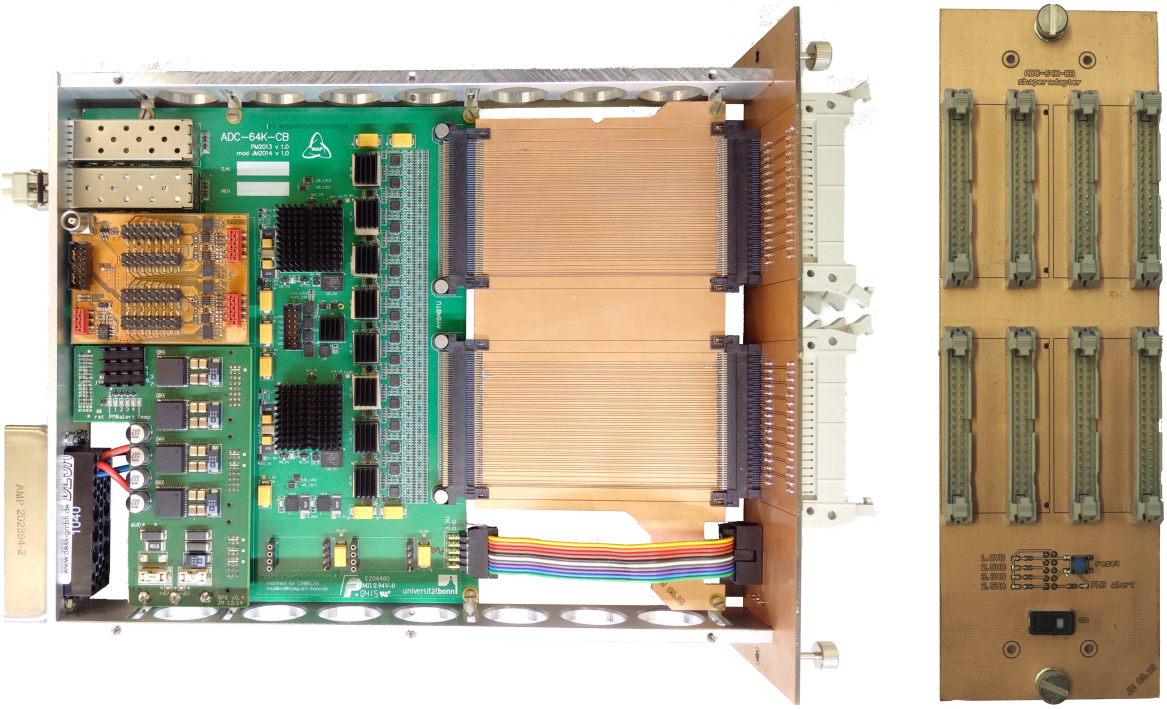



Figure A.4.: Close-up view of the BuffTi modules. One module has two MRJ21 plugs (with 24 differential pairs) to connect to the Crystal Barrel Calorimeter's slices (dark gray cables at the back), and three MRJ21 plugs (with 16 differential pairs) to connect to the CB-SADCs (white cables). The back of the module offers further MRJ21 plugs to connect to the QDC and the TDC.




(a) `ADC-64K-CB v1.0` with signal extension and pin header front face. Regular small heatspreaders are mounted on higher power ICs since the custom heat spreader was not yet developed.

(b) Pin header front face board.

Figure A.5.: `ADC-64K-CB v1.0` with signal extension board and pin header front face, which allows to connect signals directly through pin headers/pole plugs without using the Analog Input Card. The front facing PCB functions as the faceplate of the NIM cassette. Four LEDs show the status of the Power Supply rails, a fifth LED monitors the PMBus alert signal. A switch is used to toggle the PMBus control signal to turn the CB-SADC on/off, and a pushbutton can issue a reset of the power supply.

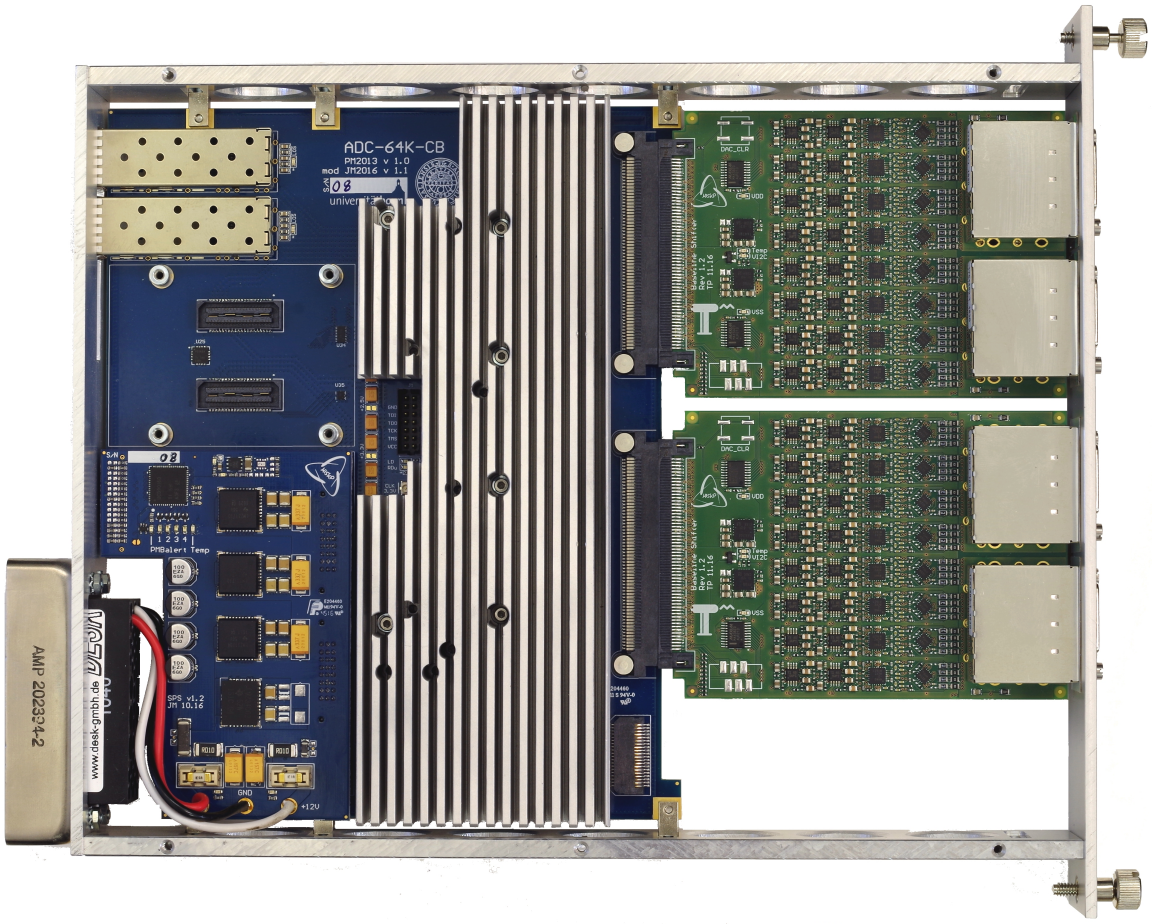



Figure A.6.: Assembled `ADC-64K-CB v1.1` consisting of the NIM cassette frame, the CB-SADC board with custom heatspreader, two Analog Input Cards, and the Power Supply. The Extension Card options to connect to the backplane or the SADC Controller are not mounted.

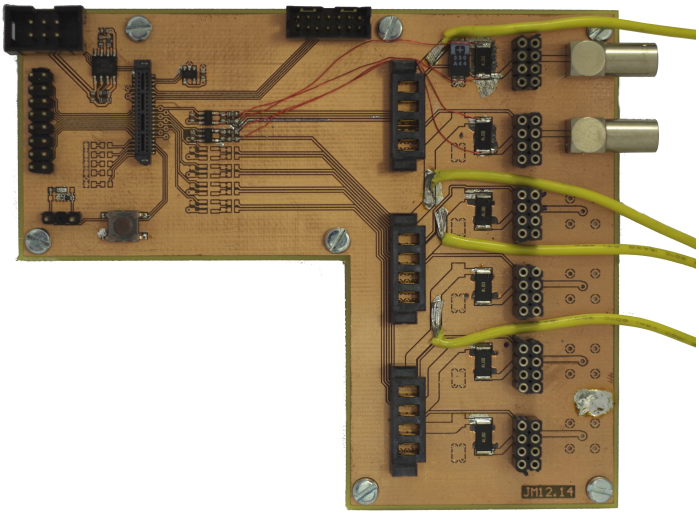



Figure A.7.: Test adapter board for the CB-SADC Power Supply. The four-channel prototype can be plugged into the power/signal connectors. Access to the sequencer's programming is possible via $I^2C$ (top left) or JTAG (top middle). The current is measured with instrument amplifiers and 4 mΩ shunt resistors. The LEMO connectors allow low-noise measurement of the residual ripple. The board was manufactured with the LPKF PROTOMAT S63.

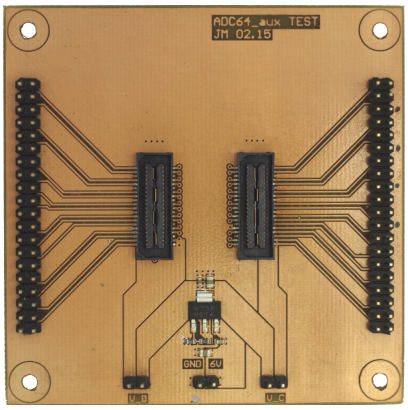



Figure A.8.: Test adapter board for the CB-SADC Extension Card. The two plugs in the middle represent the Extension Card slot present on the CB-SADC. Through measurements on the pin headers it can be ensured that the unit under test does not produce voltage levels damaging to the CB-SADC. The board was manufactured with the LPKF PROTOMAT S63.

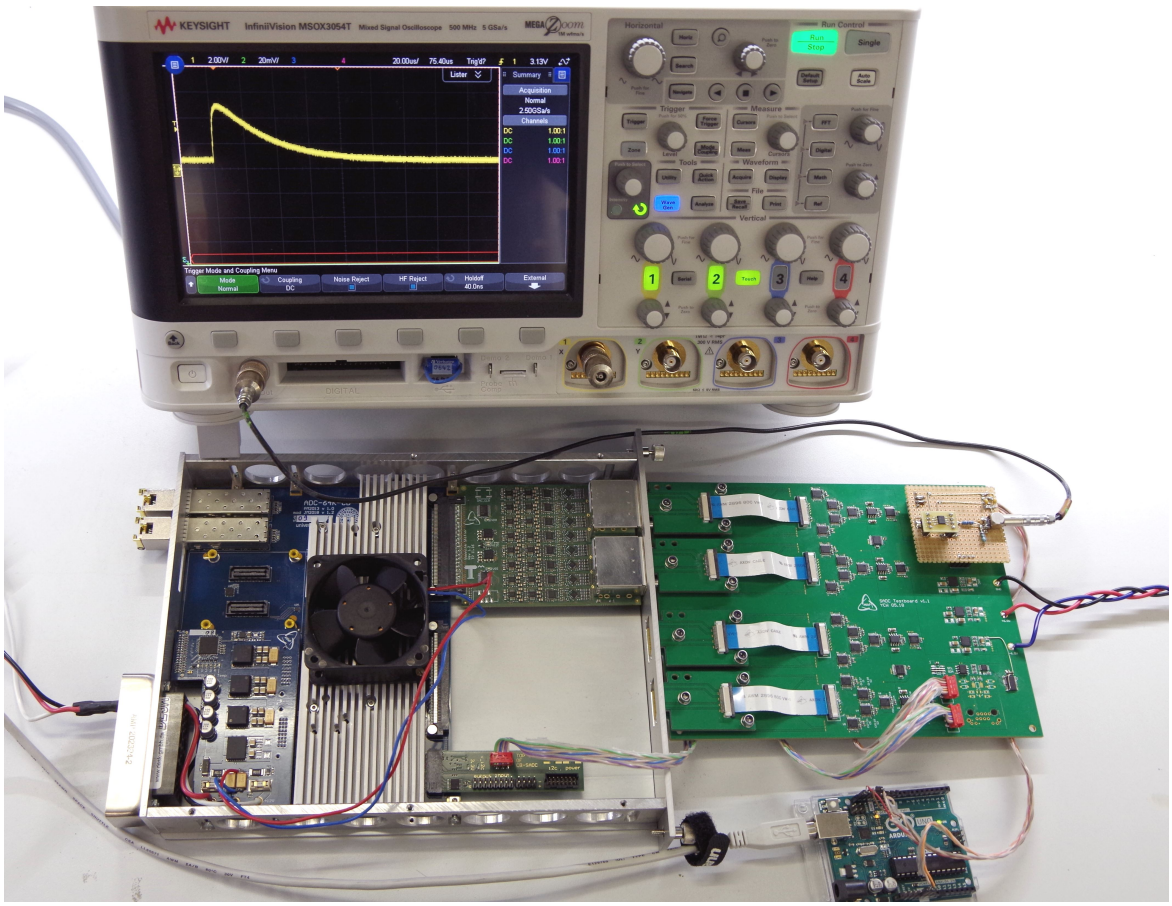



Figure A.9.: The Testboard attached to the `ADC-64K-CB v1.2`, together with the Arduino USB-to-I$^2$C bridge and an arbitrary waveform generator supplying the test signal.

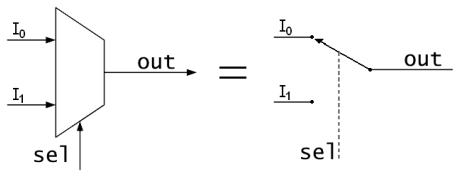
# B. FPGA Firmware Development

This appendix provides an overview of the language VHDL, the design flow leading to the firmware, and the possibilities of debugging and simulation to verify its correct functionality.

## B.1. VHDL Coding

First the developer has to write the code in the language VHDL or Verilog. For this thesis, VHDL was chosen as it seems more widely adapted in Europe, and many other projects within the Crystal Barrel Calorimeter are written in VHDL. Compared to Verilog it is strongly typed, which means that data types have to be clearly specified and operations and conversions between different types are not implicit. This is beneficial as many errors or ambiguities become apparent at very early stages of checking/synthesizing the code.[1]

VHDL can be written in a specific style, or even a mixture of styles (the CB-SADC firmware is not strictly written in one style). The two most extreme styles are classified as *Structural* and *Behavioral* description. They shall be compared in the next paragraphs by looking at the implementation of a *muxer* as shown in Figure B.1.




Source: `https://en.wikipedia.org/wiki/Multiplexer#/media/File:Multiplexer2.png`

Figure B.1.: Schematic of a 2-to-1 Multiplexer. It can be equated to a controlled switch: depending on the state of `sel`, either the signal $\texttt{I}_0$ or $\texttt{I}_1$ are connected to `out`.

In the example codes the signals are called $\texttt{I}_0$=a, $\texttt{I}_1$=b, out=y.

**Structural Description** is describing the code only by connections between low-level logic primitives. The code below shows how to describe the *muxer* by means of two `AND` gates, an `INV` gate (inverter), and an `OR` gate.

- Lines 5-8 describe the input (`a,b,s`) and output (`y`) signals of the module.
- Lines 12-25 describe the primitives (components) that are used within the module.
- Line 27 lists the signals (and signal type) that will be used to connect the primitives.
- Lines 30-37 instantiate the primitives and connect them to the signals.

[1] *"Verilog, like C, is quite content at letting you shoot yourself in the foot." [191]*

```
1  entity muxer is
2      port(a, b, s : in  std_logic;
3           y       : out std_logic);
4  end entity muxer;
5
6  architecture muxer_arch of muxer is
7
8      component AND2
9          port(a, b : in  std_logic;
10              y    : out std_logic);
11     end component;
12
13     component NOT1
14         port(a : in  std_logic;
15              b : out std_logic);
16     end component;
17
18     component OR2
19         port(a, b : in  std_logic;
20              y    : out std_logic);
21     end component;
22
23     signal a_and_nots, not_s, b_and_s : std_logic;
24
25 begin
26     Invert_S : NOT1
27         port map(s, not_s);
28     An_And_Operation : AND2
29         port map(a, not_s, a_and_nots);
30     Another_And_Operation : AND2
31         port map(s, b, b_and_s);
32     Final_Or_Operation : OR2
33         port map(a_and_nots, b_and_s, y);
34 end architecture muxer_arch;
```

**Behavioral Description** The behavioral equivalent of this code is much shorter and easier to read, especially for *programmers*. The muxer is realized by a simple `if` condition starting in line 14. Both styles will produce the same Netlist/Bitstream.

```
1  entity muxer is
2      port(a, b, s : in  std_logic;
3           y       : out std_logic);
4  end entity muxer;
5
6  architecture muxer_arch of muxer is
7  begin
8      process(a, b, s)
9      begin
10         if (s = '0') then
11             y <= a;
12         else
13             y <= b;
14         end if;
15     end process;
16 end architecture muxer_arch;
```

## B.2. Synthesis and Implementation

The code is *synthesized* by XILINX VIVADO [192], which is a software provided by the FPGA vendor; but some third-party tools can also achieve the same or even better results.

As a first step, the syntax of the code is validated. Afterwards it is processed and converted into one big netlist, describing how components will be connected on a Basic Element (BEl) level. BEls are the components native to the FPGA, such as logic gates, flip-flops, LUTs (lookup tables), distributed RAM, input/output buffers, and so on.

The Synthesis process can be configured by the *run strategies*, which usually lead to a trade-off between **runtime** (needed to synthesize), **performance**, and **area** optimization.

**Implementation** Synthesis is followed by the *implementation*, at this point the actual hardware (FPGA) is taken into account. This process can only be done by XILINX VIVADO, since detailed knowledge of the FPGA is necessary. Implementation can be split into four sub-processes [193]:

1. **Opt Design:** Performs a netlist connectivity check and optimizes the logical design. As an example, one step of Opt Design is the *Constant Propagation*, in which logic elements may be reduced/eliminated (e.g. if a constant 0 is propagated to an `AND` gate).
2. **Place Design:** This takes into account that the FPGA's combinatorial/sequential logic is organized in Configurable Logic Blocks (CLBs), and that the content of a CLB differs between FPGA families. In the KINTEX-7 FPGA, each CLB consists of two so-called slices (see Fig. F.6 on page 263).
3. **Route Design:** The placed logic blocks are connected with the *Interconnect*, which is a programmable network of signal pathways. During the routing process, all paths are evaluated for timing violations.
4. **Write Bitstream:** In this last step, a file is generated which is used to configure the FPGA. The bitstream can also be converted and stored in a ROM, such that the volatile FPGA can *boot* from it.

There are several (optional) optimization steps in between, which can be configured through *run strategies* like the Synthesis process. They are usually invoked only if the regular implementation cannot eradicate timing or area violations.

**Constraints** The implementation of XILINX VIVADO is timing-driven, which means that all paths are optimized until all timing conditions are met. XILINX VIVADO has to be made aware of the clock frequencies of all nets, and it will check if the routing obeys setup-and-hold conditions of the data paths relative to the clock. Those constraints are often too hard and may impede timing closure. Consequently, for several conditions, constraints can be formulated that may relieve the timing closure. This is very important and is used extensively for two cases:

- **Reset signals** can be allowed to take multiple clock cycles to propagate.
- **Clock domain crossings** are almost impossible to meet timing, they have to be manually constrained. This was a major challenge for the CB-SADC firmware.

## B.3. Debugging

It is possible to investigate internal signals of the FPGA during runtime, i.e. when the device works under realistic operating conditions. The requirement on the hardware side is the connection to a JTAG programmer/debugger, which is usually connected to a computer via USB.

On the firmware side, it is necessary to select the interesting signals a priori, i.e. before synthesizing the code. This is done by instantiating dedicated debugging primitives and connecting them to the signals. Two options are available:

- A **VIO** is used to monitor and drive internal FPGA signals in real time.
- An **ILA** can record a trace of the signal for a certain time.

While a VIO can be regarded as *cheap* when it comes to the FPGA's resources, an ILA is quite *expensive* as it requires block RAM to store the data. Consider for example recording 32 different 16-bit signals for a duration of 4096 clock cycles, which is actually being used in the firmware to analyze the waveform data of all 32 channels: this will require 57 `RAMB36E1` primitives (each can store up 32 kbit). The KINTEX-7 used in this thesis has 325 `RAMB36E1` primitives, from which more than 50 % are used for the firmware implementation without any debugging. Hence one has to consider carefully when it comes to debugging larger signal buses in greater depth.

Figure B.2 shows an example of the debugging view of the XILINX VIVADO ANALYZER using ILAs. The screenshot shows the digitized waveforms, which are displayed as *analog signals*.

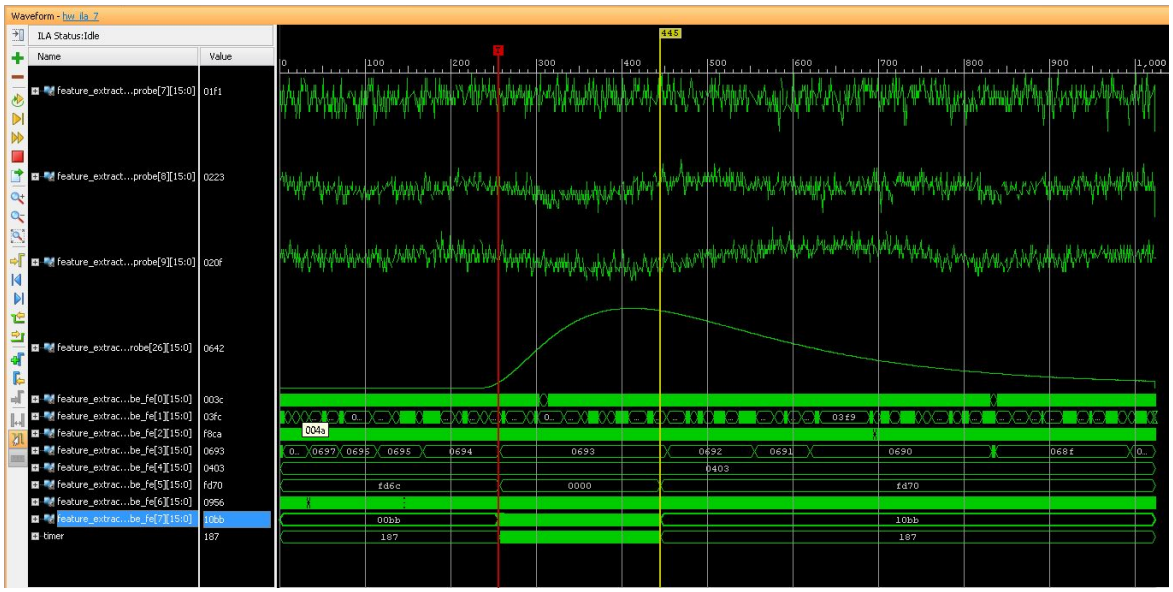



Figure B.2.: Example of debugging via JTAG. The signals that are shown had to be selected before Synthesis. The first four signals are raw data from the LTM9009-14 ADCs, while the first three only show the noisy baseline, the fourth shows a pulse.

## B.4. Simulation

With a certain complexity, it becomes unfeasible to evaluate the correct behavior of a firmware during *live* operation as described in the previous section. Consider going after a bug, marking and investigating more and more signals for JTAG debugging, and creating a new Bitstream over and over: this is extremely tedious, scaling with the complexity of the design (creating the Bitstream for the firmware described in this thesis takes roughly 30 min).

It is a more efficient workflow to simulate all code prior to creating a Bitstream. To achieve this, a so-called test bench has to be devised, usually not for the full design but for submodules, such that they can be evaluated independently. While for the debugging interface only a limited number of signals is accessible at a time and for a restricted depth, this is not a problem in simulations. In principle all signals can be investigated for an arbitrary time, they do not have to be selected a priori. Figure B.3 shows the signal view of the simulation, which at first glance is very similar to the ILA screenshot of the previous section.

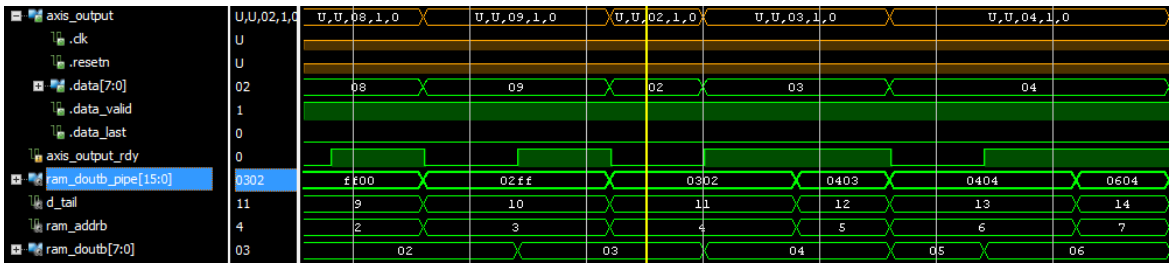



Figure B.3.: Example of simulating a module with Xilinx Vivado Analyzer. An Axi4-Stream transmitting data from a RAM can be observed. The first two signals are uninitialized (`U`).

Three different kinds of simulations have to be distinguished.

- **Behavioral Simulation** is more or less the evaluation of the code, similar to running a scripted programming code. This is the fastest method of simulation.
- **Post-Synthesis Simulation** simulates based on the Netlist generated during synthesis, taking into account the conversion into BEls.
- **Post-Implementation Simulation** works with the placed design, taking into account the specific FPGA.

With dedicated tools it is even possible to perform a timing-analysis in the **Post-Implementation Simulation**. This takes into account the routing of the design, thus the actual propagation delays from one element to the next, through the interconnect network. Only with this is it possible to investigate timing problems that might stem from unconstrained or wrongly constrained paths.

# C. Configuration

In this chapter, all configuration options mentioned throughout sections 7.3, 7.5, 7.6, and 7.7 will be summarized.

As shown in Figure 7.14 on page 153, and explained in section 7.4.4, the configuration is received via UDP packets and forwarded to different configuration parsers depending on the UDP port. The parsers' algorithms require the expected count of bytes to be matched, and there is no content sensitive validation of the packets. Although it is not possible to *destroy* the CB-SADC through malicious configuration, it is easily achievable to configure it to such an extent that it becomes unresponsive and requires a power cycle. Therefore, the configuration should always be carried out with care.

## C.1. Configuration of Feature/Sample Extraction

The packets addressed to port 4096 contain configurations for feature extraction and sample extraction. All commands have a length of 4 byte and a structure of `0xCC PP XX XX`, where `0xCC` stands for the channel, `0xPP` for the parameter, and `0xXX XX` for the 16-bit configuration word. If the configuration does not require a channel, for example for the configuration of the global integral position, the regarding byte will not be evaluated. Table C.1 shows all configuration parameters.

## C.2. Configuration of Ltm9009-14 ADCs

The packets addressed to port 4097 contain configurations for the (re-)initialization of the Ltm9009-14 ADCs and consist of just 1 byte. The parameters are listed in Table C.2 and are explained in detail in section 7.2.3.

## C.3. Configuration and Control of the Reliability Protocol

All received network packets with UDP port 4098 are forwarded to the parser for the Reliability Protocol, which was introduced in section 7.4.5. The packets have a content of 5 byte and are shown in Table C.3.

## C.4. Configuration of Operating Parameters

A small set of other parameters can be configured through UDP port 4096. The parameters consist of 4 byte. The first set of two parameters can be used to change the IP address of the CB-SADC itself, the second set to change the LEvB address (where all packets are sent to).

| Parameter | Name | per channel | Size | Section |
|---|---|---|---|---|
| 0x00 | *reserved* | | | |
| 0x01 | send_sample | yes | 1 bit | 7.6 |
| 0x02 | led.threshold | yes | 16 bit | 7.7.1 |
| 0x03 | cfd.time_over_threshold | yes | 10 bit | 7.5.4 |
| 0x04 | cfd.time_defuse | yes | 10 bit | 7.5.4 |
| 0x05 | cfd.threshold | yes | 16 bit | 7.5.4 |
| 0x06 | max.rising_time | yes | 16 bit | 7.5.3 |
| 0x10 | pileup.ratio | no | 10 bit | 7.5.5 |
| 0x11 | pileup.threshold | no | 10 bit | 7.5.5 |
| 0x12 | pileup.upperslope | no | 12 bit | 7.5.5 |
| 0x13 | pileup.lowerslope | no | 12 bit | 7.5.5 |
| 0x14 | pileup.sendsample | yes | 10 bit | 7.5.5 |
| 0xA0 | trigger_ext_bitmask | no | 8 bit | 7.7 |
| 0xA1 | trigger_int_bitmask | no | 16 bit | 7.7 |
| 0xA2 | trigger_int_bitmask | no | 16 bit | 7.7 |
| 0xA3 | *reserved* | | | |
| 0xA4 | *reserved* | | | |
| 0xA5 | trigger_network | no | 1 bit | 7.7 |
| 0xC0 | sample_position | no | 16 bit | 7.6 |
| 0xC1 | *reserved* | | | |
| 0xC2 | *reserved* | | | |
| 0xC3 | integral_position | no | 16 bit | 7.5.2 |
| 0xC4 | sample_prescaler | no | 16 bit | 7.6 |
| 0xC5 | sample_stretcher | no | 16 bit | 7.6 |

Table C.1.: List of parameters for the configuration of feature extraction and sample extraction, to be received on UDP port 4096.

| Parameter | Name |
|---|---|
| 0x00 | adc_bitslip |
| 0x01 | adc_calibrate |
| 0x02 | adc_wake |
| 0x03 | adc_sleep |
| 0x04 | adc_pattern_off |
| 0x05 | adc_pattern_on |
| 0x06 | adc_init |
| 0x07 | adc_reset |
| 0x08 | adc_reinit |

Table C.2.: List of parameters for the configuration of the LTM9009-14 ADC, to be received on UDP port 4097.

| Parameter | Name |
|---|---|
| 0xA0 00 00 00 00 | CMD_RST |
| 0xA1 00 00 00 00 | CMD_PUSH |
| 0xA2 00 00 00 00 | *reserved* |
| 0xA3 00 00 00 00 | CMD_ACK |
| 0xA4 00 00 00 00 | *reserved* |
| 0xA5 00 00 00 00 | *reserved* |
| 0xA6 ?? ?? ?? ?? | CMD_RESEND_BUFFER |
| 0xD0 FF FF FF FF | CMD_CLEAR_BUFFER |

Table C.3.: List of parameters for the configuration of the Reliability Protocol/Burst Buffers, to be received on UDP port 4098. The 0x?? ?? ?? ?? mask determines which packet(s) of the burst is/are requested again.

It is also possible to manually issue reset signal for all seven reset signal described in the state machine in Figure 7.3 on page 138 through the 8 bit RESET_VECTOR, whose content is shown in Table C.5.

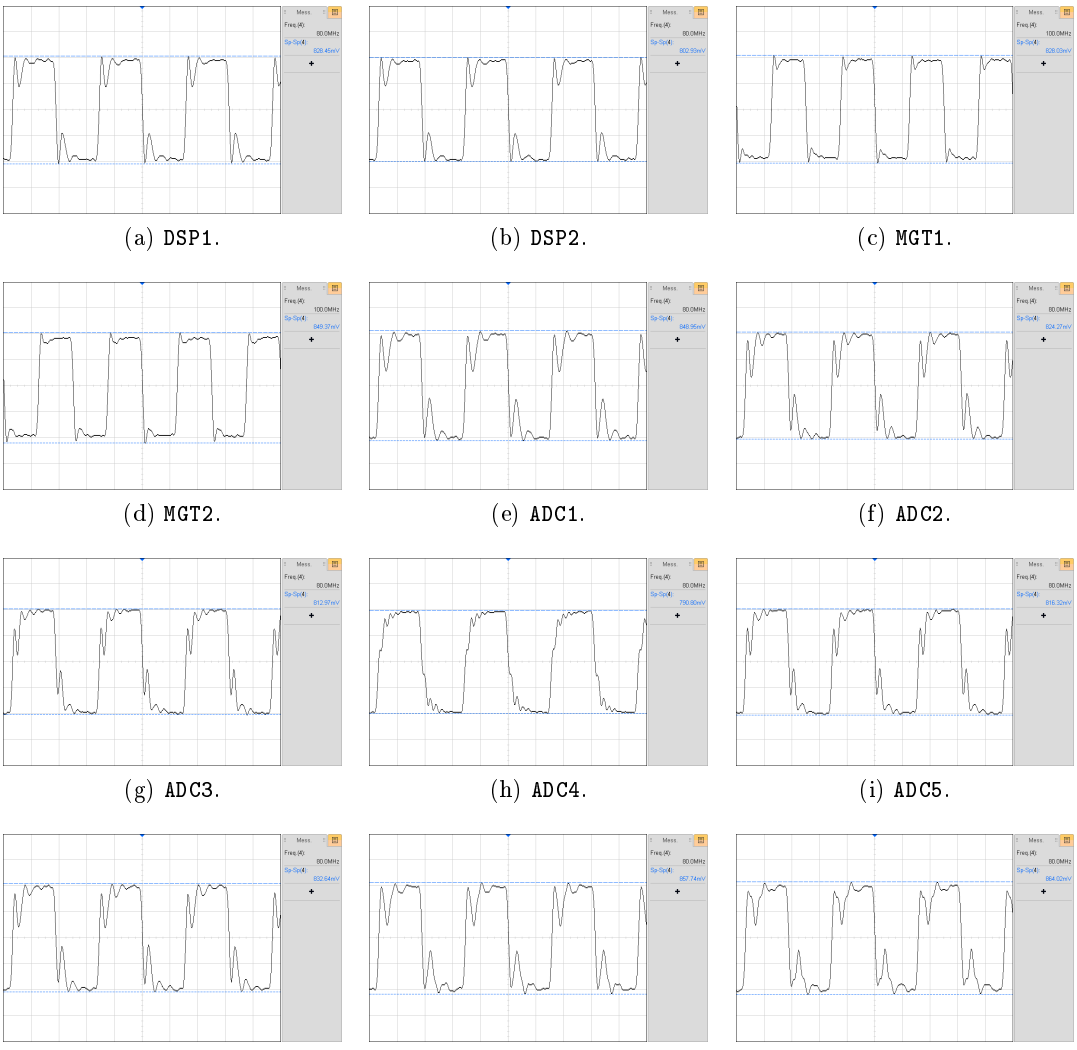
| Parameter | Name |
|---|---|
| 0x00 B0 ?? ?? | CB-SADC IP (upper octets) |
| 0x00 B1 ?? ?? | CB-SADC IP (lower octets) |
| 0x00 B2 ?? ?? | LEvB IP (upper octets) |
| 0x00 B3 ?? ?? | LEvB IP (lower octets) |
| 0x02 C2 ?? ?? | RESET_VECTOR |

Table C.4.: List of further parameters that can be configured through UDP port 4096.

| Bit | Reset Signal |
|---|---|
| 7 | Reset LMK04806 (and reprogram) |
| 6 | Reset FPGA clock managers |
| 5 | Reset the Ethernet cores |
| 4 | Reset the UDP core |
| 3 | Reset the ADC de-serializer |
| 2 | Reset and reprogram the LTM9009-14 ADCs |
| 1 | Reset the feature extraction |
| 0 | *reserved* |

Table C.5.: Reset signal that can be issued through the RESET_VECTOR configuration.

# D. Plots



(a) DSP1. (b) DSP2. (c) MGT1.

(d) MGT2. (e) ADC1. (f) ADC2.

(g) ADC3. (h) ADC4. (i) ADC5.

(j) ADC6. (k) ADC7. (l) ADC8.

Figure D.1.: Waveforms of the clock signals, measured close to the PLL (LMK04806). DSP are the clocks for the DSP algorithms, MGT are the clocks for the network interface, ADC are the ADC digitization clocks (cf. Figure 5.8). All signals show artifacts, possibly from non-ideal matching of termination and trace impedance.

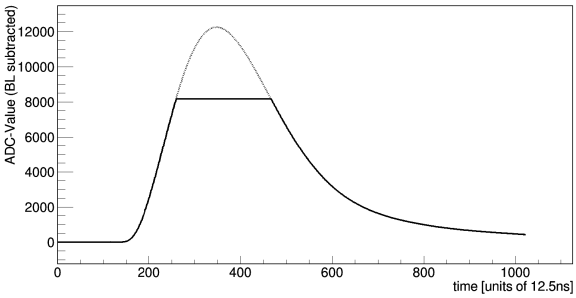
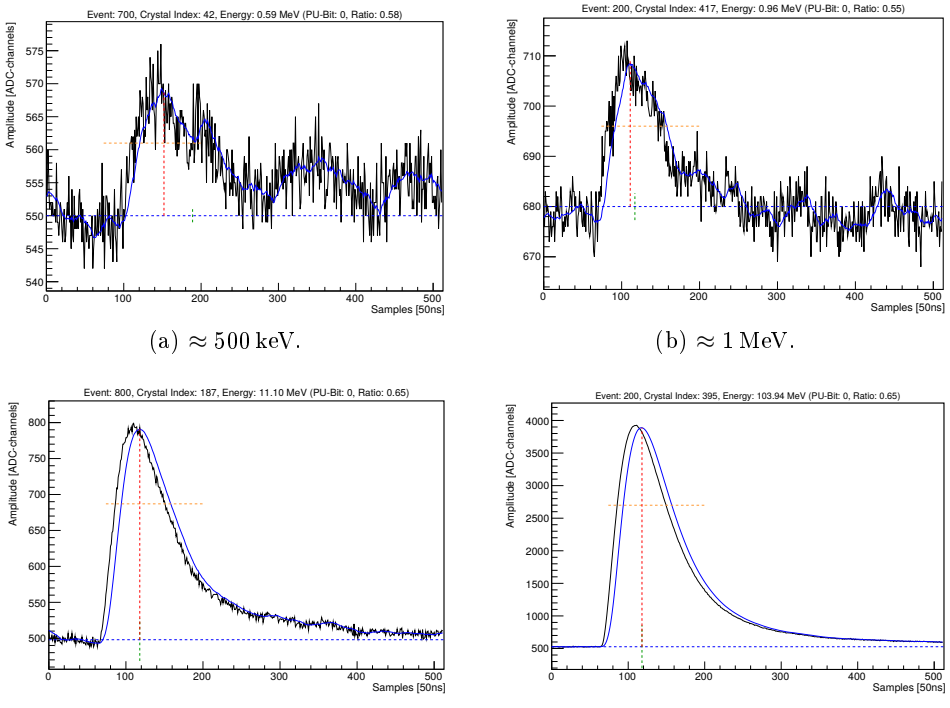



Figure D.2.: Pulse recorded with the CB-SADC. The signal originated from an arbitrary waveform generator and was recorded with the Crystal Barrel Calorimeter. It was processed by a NIM shaper (band-pass filter) with a 1 µs shaping time. Since the pulse amplitude exceeds the LTM9009-14 ADC's full scale range, it is clipped off at the top (solid line) at a value of $2^{13} - 1 = 8191$. The dashed pulse is a (scaled up) waveform without clipping, indicating that the signal follows the expected curve despite the clipping. [122]



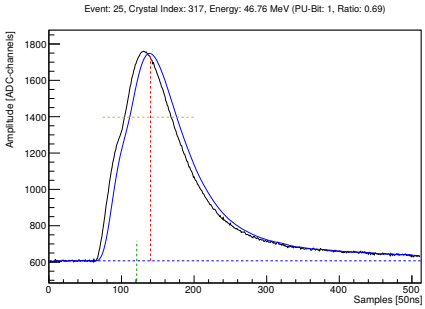

(a) ≈ 500 keV.

(b) ≈ 1 MeV.

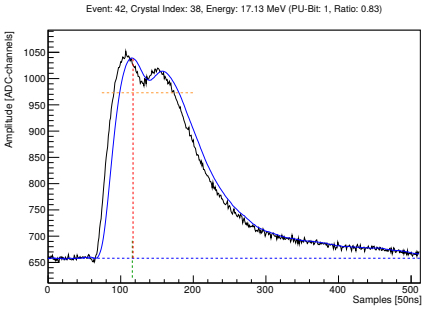
(c) ≈ 10 MeV.

(d) ≈ 100 MeV.

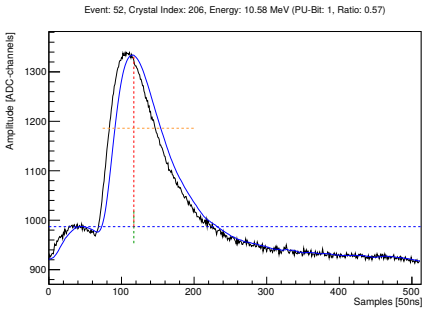
Figure D.3.: Typical waveforms recorded at the Crystal Barrel Calorimeter, for different deposited energies. The blue dashed line shows the baseline as extracted by the FPGA of the CB-SADC. The blue solid line shows a 16-fold moving average to visualize how the FPGA determines the maximum value and timestamp. The latter is indicated by the red dashed line. The green dashed line shows the timestamp of the constant fraction discriminator. The orange dashed line shows the width of the integration window, its height represents the integral value. *Data analysis performed by J. Schultes.*



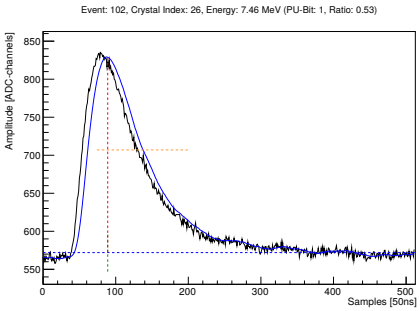

(a) Two pulses with similar amplitude, very close to one another.




(b) A smaller pulse sitting on the falling edge of a larger pulse.




(c) A larger pulse sitting on the falling edge of a smaller pulse.




(d) A regular pulse (no pile-up), but shifted to the left with respect to the expected position.

Figure D.4.: Typical waveforms recorded at the Crystal Barrel Calorimeter, with pile-up detected by the FPGA algorithm (**PU-Bit: 1**). The **ratio** can be compared to Figure 9.16. The algorithm is sensitive to the ratio of the integral in a fixed window to the maximum of the pulse. This also marks pulses that are not at the expected position with regard to the trigger time, and thus possibly uncorrelated to the triggered event. *Data analysis performed by J. Schultes.*

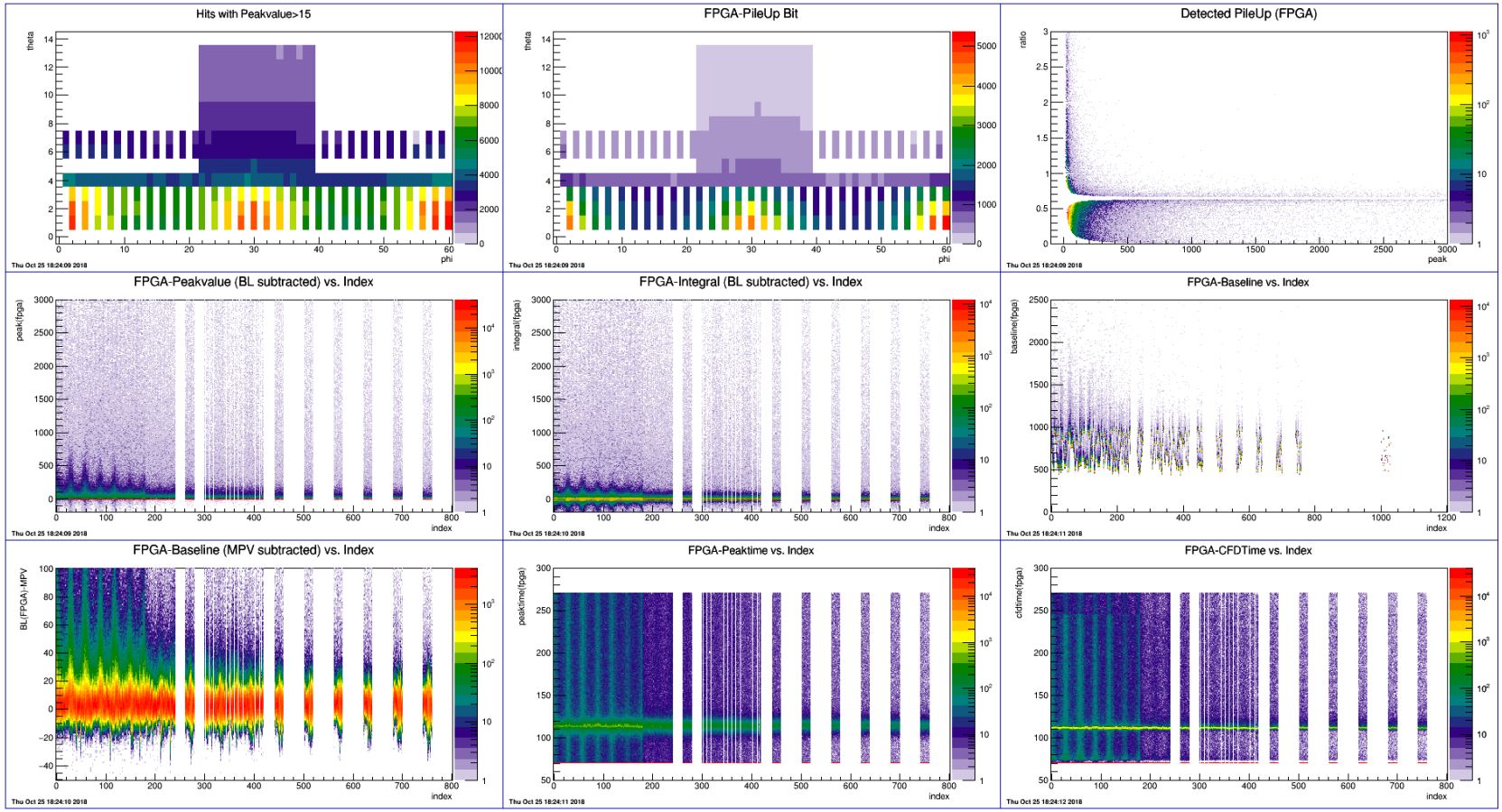
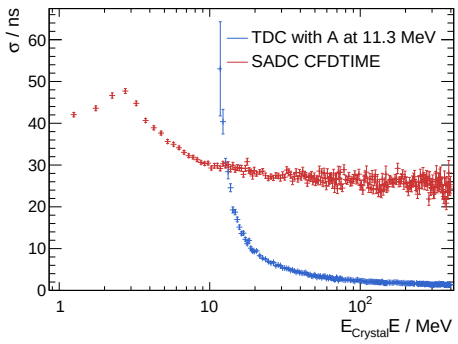
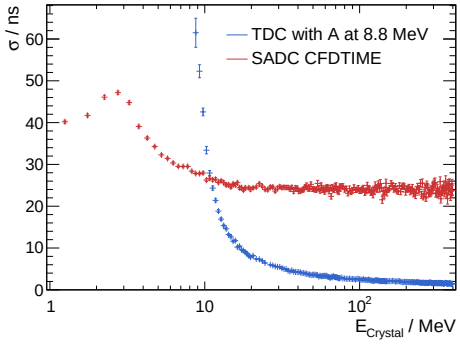


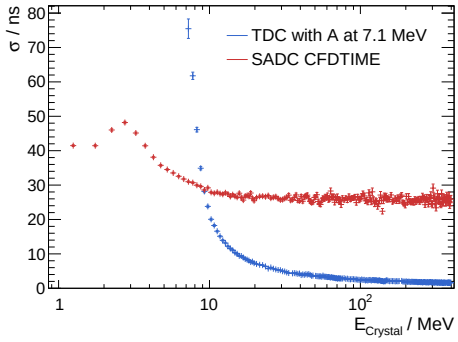

Figure D.5.: Screenshot of the Online Monitor during the October 2018 production beam time. The theta-phi hitmaps (top left) show the connection of the CB-SADCs to the most forward four rings of the barrel, plus the additionally connected areas. The top-right histogram shows the pile-up detection algorithm at work (cf. Fig. 7.23 page 168). The bottom-right histograms show the timestamp distributions, a clear accumulation is visible at the expected pulse position.



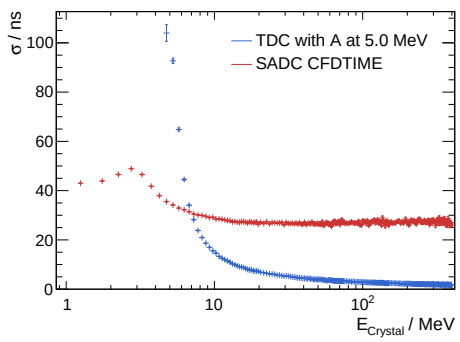

(a) First ring. Crossing around 13 MeV.




(b) Second ring. Crossing around 11 MeV.

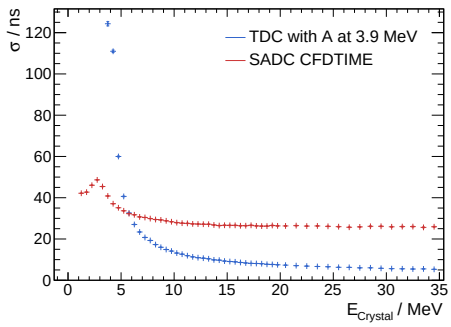



(c) Third ring. Crossing around 9 MeV.

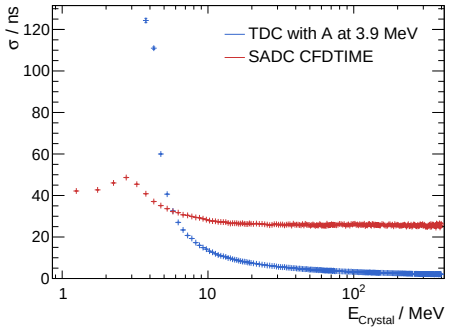



(d) Fourth ring. Crossing around 7 MeV.

Figure D.6.: Comparison of the standard deviation of the timestamps of CB-SADC and TDC. The data is binned in energy intervals of 0.5 MeV, and has been analyzed for different azimuthal angles (corresponding to the rings of the calorimeter) to account for the different discriminator thresholds of the TDC. The crossing point of both curves shows below which energy the CB-SADC can deliver a better time resolution than the TDC. *Data analysis performed by Nils Stausberg.*




(a) Linear scale. Crossing around 6 MeV.




(b) Logarithmic scale.

Figure D.7.: Comparison of the standard deviation of the timestamps of CB-SADC and TDC. The data is binned in energy intervals of 0.5 MeV and has been analyzed for all rings from the fifth on. Apart from a small region in the backwards direction, the TDC discriminator threshold was set to 3.9 MeV. *Data analysis performed by Nils Stausberg.*

# E. Code

Listing E.1: Commented content of the configuration registers of the Texas Instruments Lmk04806.

```
x"00020000",  --R0 (CLKout_1_PD = 1, RESET=1, CLKout0_1_DIV=10)
x"001403C0",  --R0 (CLKout0_1_DIV=30)  80MHz
x"001403C1",  --R1 (CLKout2_3_DIV=30)  80MHz
x"00140302",  --R2 (CLKout4_5_DIV=24) 100MHz
x"001403C3",  --R3 (CLKout6_7_DIV=30)  80MHz
x"001403C4",  --R4 (CLKout8_9_DIV=30)  80MHz
x"001403C5",  --R5 (CLKout10_11_DIV=30)80MHz
x"11110006",  --R6 (CLKout3_TYPE=1, CLKout2_TYPE=1, CLKout1_TYPE=1,
    CLKout0_TYPE=1, CLKout2_3_ADLY=0, CLKout0_1_ADLY=0)
x"11110007",  --R7 (CLKout7_TYPE=1, CLKout6_TYPE=1, CLKout5_TYPE=1,
    CLKout4_TYPE=1, CLKout6_7_ADLY=0, CLKout5_4_ADLY=0)
x"11110008",  --R8 (CLKout11_TYPE=1, CLKout10_TYPE=1, CLKout9_TYPE=1,
    CLKout8_TYPE=1, CLKout10_11_ADLY=0, CLKout8_9_ADLY=0)
x"55555549",  --R9 (fixed pattern)
x"1000410A",  --R10 (OSCout1_LVPECL_AMP=0(-1600mV), OSCout0_TYPE=0(LVDS),
    EN_OSCout1[23]=0,
--EN_OSCout0[22]=0 (disabled), OSCout1_MUX[21]=0 (bypass MUX),
--OSC_out1_MUX[20]=0(bypass MUX), OSC_out0_MUX[20]=0(bypass MUX), PD_OSCin
    [19]=0(powered), OSCout_DIV[18:16]=0,
--VCO_MUX[12]=0(select VCO), EN_FEEDBACK_MUX[11]=0 (feedback mux powered down
    ),
--VCO_DIV[10:8]=1(divide by 1), FEEDBACK_MUX[7:5]=0
x"3400800B",  --R11 (MODE[31:27]=6, EN_SYNC=1, NO_SYNC_CLKout10_11=0,
    NO_SYNC_CLKout8_9=0,
--NO_SYNC_CLKout6_7=0, NO_SYNC_CLKout4_5=0, NO_SYNC_CLKout2_3=0,
    NO_SYNC_CLKout0_1=0, SYNC_MUX[19:18]=0,
--SYNC_QUAL=0, SYNC_POL_INV=0, SYNC_EN_AUTO=1, SYNC_TYPE[14:12]=0,
    EN_PLL2_XTAL=0)
x"130C00AC",  --R12 (LD_MUX[31:27]=2 (PLL2 DLD (digital lock detect)),
--LD_TYPE[26:24]=3 (output push-pull), SYNC_PLL2_DLD[23]=0 (sync not forced),
--SYNC_PLL1_DLD[22]=0 (sync not forced), EN_TRACK[8]=0, HOLDOVER_MODE[7:6]=2)
x"0400800D",  --R13 (holdover_mux[31:27]=0 (logic low), holdover_type
    [26:24]=4(output inv (pp)),
--status_clkin1_mux[22:20]=0(logic low), status_clkin0_Type[18:16]=0 (input),
--disable_dld1_det[15]=1(disables), status+clkin0_mux[14:12]=0(logic low),
--clkin_select_mode[11:9]=0 (clkin0 man),
--clk_in_select_inv[8]=0 (not inv), en_clkin1[6]=0 (dis), en_clkin0[5]=0 (dis
    ))
x"003FC00E",  --R14 (LOS_TIMEOUT[31:30]=0 (1200ns), EN_LOS=0 (disabled)
--Status_CLKin0_Type[26:24]=0 (input), CLKinX0_BUF_TYPE=1 (cmos),
    CLKinX1_BUF_TYPE=1
--DAC_HIGH_TRIP=3F (VCC), DAC_LOW_TRIP=0 (VCC/64), EN_VTUNE_RAIL_DET=0)
x"0000004F",  --R15 (MAN_DAC [31:22]=0
--EN_MAN_DAC[20]=0 (enables manual DAC), HOLDOVER DLD+CNT[19:6]=1
--FORCE_HOLDOVER[5]=0)
```

```
34 x"01550410",  --R16 (XTAL_LVL[31:30]=0
35 x"00000018",  --R24 (PLL2_C4_LF[31:28]=0(10pF), PLL2_C3_LF[27:23]=0(10pF),
36 --PLL2_R4_LF[22:20]=0(200 Ohm), PLL2_R3_LF[18:16]=0(200 Ohm),
37 --PLL1_N_DLY[14:12]=0, PLL1_R_DLY[10:8]=0, PLL1_WIND_SIZE[7:6]=0
38 x"00400059",  --R25 (DAC_CLK_DIV[31:22]=1, PLL1_DLD_CNT[19:6]=1)
39 x"AFA8001A",  --R26 (PLL2_WIND_SIZE[31:30]=2
40 --EN_PLL2_REF_2X[29]=1 (allows wider loop bw filter),
41 --PLL2_CP_POL[28]=0 (negative to use internal VCO),
42 --PLL2_CP_GAIN[27:26]=3,
43 --PLL2_DLD_CNT[19:6]=2000,
44 --PLL2_CP_TRI=0
45 x"0000005B",  --R27 (PLL1_CP_POL[28]=0, PLL1_CP_GAIN[27:26]=0,
       CLKin1_PreR_DIV[23:22]=0,
46 --CLKin0_PreR_DIV[21:20]=0,
47 --PLL1_R[19:6]=1, PLL1_CP_TRI=0(because PLL1_CP_GAIN is not equal to XXXX)
48 x"0040005C",  --R28 (PLL2_R[31:20]=4, PLL1_N[19:6]=2(not used))
49 x"0100031D",  --R29 (OSC_IN_FREQ[26:24]=1 (63  127 MHZ), PLLP_FAST_PDF[23]=0,
        PLL2_N_CALIB[22:5]=3)
50 x"0100031E",  --R30 (PLL2_P[26:24]=1 (2x), PLL2_N[22:5]=3)
51 x"001F001F"   --R31 (READBACK_LE[21] (guessing low)=0, READBACK_ADDRESS
       [20:16]=1F (from LMK03806),
52 --uWire_LOCK[5]=0 (from LMK03806)
```

Listing E.2: Part of the test bench for the simulation of the phase calibration and bitslip.

```
library ieee;
use work.cb_pkg.all;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;
entity test_adc is
end test_adc;

architecture behavior of test_adc is

    component ADC_INTERFACE_wrapper
        port(
            REF_CLOCK           : in  std_logic;
            RESET               : in  std_logic;
            reset_idelayctrl    : in  std_logic;
            data_in_from_pins_p : in  std_logic_vector(71 downto 0);
            data_in_from_pins_n : in  std_logic_vector(71 downto 0);
            DCO_P               : in  std_logic_vector(7 downto 0);
            DCO_N               : in  std_logic_vector(7 downto 0);
            ADC_DAT_OUT         : out std_logic_vector(511 downto 0);
            ADC_FRA_OUT         : out std_logic_vector(63 downto 0);
            ADC_CLK_OUT         : out std_logic_vector(7 downto 0);
            adc_calibrate       : in  std_logic;
            adc_bitslip         : in  std_logic
        );
    end component;

    type timearray is array (7 downto 0) of time;
    type realarray is array (7 downto 0) of real;
    signal dco_jitter            : time      := 0 ps;
    signal dco_delay             : timearray;
    signal data_delay            : timearray;
    signal clk_200               : std_logic := '0';
    signal dco_int               : std_logic_vector(7 downto 0);
    signal dco_p                 : std_logic_vector(7 downto 0);
    signal dco_n                 : std_logic_vector(7 downto 0);
    signal adc_dat_out           : std_logic_vector(511 downto 0);
    signal adc_fra_out           : std_logic_vector(63 downto 0);
    signal adc_clk_out           : std_logic_vector(7 downto 0);
    signal adc_calibrate         : std_logic := '0';
    signal adc_bitslip           : std_logic := '0';
    signal reset                 : std_logic := '0';
    signal data_in_from_pins_int : std_logic_vector(71 downto 0);
    signal data_in_from_pins_p   : std_logic_vector(71 downto 0);
    signal data_in_from_pins_n   : std_logic_vector(71 downto 0);
    signal r_int                 : realarray;
    signal s_int                 : realarray;
    constant clk_200_period      : time      := 5000 ps;
    constant clk_320_period      : time      := 3125 ps;
    constant jitter_LTM9009      : time      := 750 ps; --Datasheet page 6
    constant SEEDINIT            : positive  := 1;

begin
    ADC_INTERFACE_inst : ADC_INTERFACE_wrapper
        port map(
```

```
56              REF_CLOCK           => clk_200,
57              RESET               => reset,
58              reset_idelayctrl    => '0',
59              data_in_from_pins_p => data_in_from_pins_p,
60              data_in_from_pins_n => data_in_from_pins_n,
61              DCO_P               => dco_p,
62              DCO_N               => dco_n,
63              ADC_DAT_OUT         => adc_dat_out,
64              ADC_FRA_OUT         => adc_fra_out,
65              ADC_CLK_OUT         => adc_clk_out,
66              adc_calibrate       => adc_calibrate,
67              adc_bitslip         => adc_bitslip
68          );
69
70      clk_200_process : process
71      begin
72          clk_200 <= '0';
73          wait for clk_200_period / 2;
74          clk_200 <= '1';
75          wait for clk_200_period / 2;
76      end process;
77
78      dco_jitter_process : process
79          variable seed1 : positive := SEEDINIT * 1;
80          variable seed2 : positive := SEEDINIT * 2;
81          variable r     : real;
82      begin
83          uniform(seed1, seed2, r);
84          dco_jitter <= (r - 0.5) * jitter_LTM9009;
85          wait for clk_320_period;
86      end process;
87
88      random_process : process
89          variable seed1 : positive := SEEDINIT * 3;
90          variable seed2 : positive := SEEDINIT * 4;
91          variable r     : realarray;
92          variable s     : realarray;
93      begin
94      for i in 0 to 7 loop
95        uniform(seed1, seed2, r(i));
96        uniform(seed1, seed2, s(i));
97          end loop;
98          r_int <= r;
99          s_int <= s;
100         wait;
101     end process;
102
103     dco_inv_gen : for i in 0 to 7 generate
104         clk_data_process : process
105         begin
106             data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"
                    000000000";
107             dco_int(i) <= '0';
108             wait for clk_320_period / 2;
109             data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"
                    000000000";
```

```
110             dco_int(i) <= '1';
111             wait for clk_320_period / 2;
112             data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"
                    011111111";
113             dco_int(i) <= '0';
114             wait for clk_320_period / 2;
115       data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"000000000";
116             dco_int(i) <= '1';
117             wait for clk_320_period / 2;
118       data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"111111111";
119             dco_int(i) <= '0';
120             wait for clk_320_period / 2;
121       data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"100000000";
122             dco_int(i) <= '1';
123             wait for clk_320_period / 2;
124       data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"111111111";
125             dco_int(i) <= '0';
126             wait for clk_320_period / 2;
127             data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) <= B"
                    100000000";
128             dco_int(i) <= '1';
129             wait for clk_320_period / 2;
130         end process;
131
132         dco_delay(i)  <= r_int(i) * clk_320_period;
133         data_delay(i) <= s_int(i) * clk_320_period;
134         dco_p(i)    <= transport dco_int(i) after (dco_delay(i) + dco_jitter
                + jitter_LTM9009);
135         dco_n(i)    <= not dco_p(i);
136         data_in_from_pins_p(((i+1)*9)-1 downto i*9) <= transport
                data_in_from_pins_int(((i + 1) * 9) - 1 downto i * 9) after
                data_delay(i);
137         data_in_from_pins_n(((i+1)*9)-1 downto i*9) <= not
                data_in_from_pins_p(((i + 1) * 9) - 1 downto i * 9);
138     end generate;
139
140     stim_proc : process
141     begin
142         reset         <= '1';
143         wait for 100 ns;
144         reset         <= '0';
145         adc_calibrate <= '0';
146         adc_bitslip   <= '0';
147         wait for 1000 ns;
148         adc_calibrate <= '1';
149         wait for 50 ns;
150         adc_calibrate <= '0';
151         wait for 6000 ns;
152         adc_bitslip   <= '1';
153         wait for 1000 ns;
154         adc_bitslip   <= '0';
155         wait;
156     end process;
157 end;
```

Listing E.3: Constant fraction discriminator state machine.

```
case fe_cfd_state is
  when cfd_wait =>
    timer_low    := 0;
    timer_reset  := 0;
    output       <= '0';
    output_valid <= '0';
    -- Check that the CFD is above THRESHOLD for at least TIME_OVER_THRESHOLD
       /clock
    if (((input - baseline_int) / FRACTION - (input_buf - baseline_int)) *
       PULSE_SIGN) > THRESHOLD then
      timer_high := timer_high + 1;
    else
      if timer_high > TIME_OVER_THRESHOLD then
        fe_cfd_state <= cfd_was_above_treshold;
        timer_high := 0;
      else
        fe_cfd_state <= cfd_wait;
      end if;
    end if;

  when cfd_was_above_treshold =>
    -- Check that the CFD drops below Zero within TIME_DEFUSE/clock
    if (((input - baseline_int) / FRACTION - (input_buf - baseline_int)) *
       PULSE_SIGN) < 0 then
      fe_cfd_state <= cfd_armed;
      output       <= '1';
      output_valid <= '0';
    else
      output       <= '0';
      output_valid <= '0';
      if timer_reset = TIME_DEFUSE then
        timer_reset  := 0;
        fe_cfd_state <= cfd_wait;
      else
        timer_reset  := timer_reset + 1;
        fe_cfd_state <= cfd_armed;
      end if;
    end if;

  when cfd_armed =>
    -- Check that the CFD is below negative Threshold for
    -- at least 2*TIME_DEFUSE/clock
    if (((input - baseline_int) / FRACTION - (input_buf - baseline_int)) *
       PULSE_SIGN) < -THRESHOLD then
      if timer_low = 2 * TIME_OVER_THRESHOLD then
        output       <= '0';
        output_valid <= '1';
        fe_cfd_state <= cfd_wait;
      else
        output       <= '0';
        output_valid <= '0';
        timer_low    := timer_low + 1;
        fe_cfd_state <= cfd_armed;
      end if;
    -- Allow a transition time o TIME_RESET/clock between THRESHOLD and ZERO
```

```
52      else
53        output       <= '0';
54        output_valid <= '0';
55        if timer_reset > TIME_DEFUSE then
56          fe_cfd_state <= cfd_wait;
57        else
58          timer_reset  := timer_reset + 1;
59          fe_cfd_state <= cfd_armed;
60        end if;
61      end if;
62  end case;
```

Listing E.4: Function to process pending UDP datagrams. For better readability the code has been extended by a few lines, which originally appear in different places in the programm.

```
#define PD_SIZE_OF_SYSTEM_HDR   12
#define PD_SIZE_OF_FE_HDR       16
#define PD_SIZE_OF_SP_HDR       4
#define PD_SIZE_OF_HEADER       PD_SIZE_OF_SYSTEM_HDR+PD_SIZE_OF_FE_HDR
#define PD_SIZE_OF_FE           NR_OF_CHANNELS*2*5
#define PD_SIZE_OF_TRAILER      4
#define PD_SIZE_TOTAL           PD_SIZE_OF_HEADER + PD_SIZE_OF_FE +
    PD_SIZE_OF_TRAILER

void sadc64_test::processPendingDatagrams()
{
  // file handling
  QFile m_file("default.hex");
  QDataStream m_outStream(&m_file);
  // console
  QTextStream qout(stdout);
  // UDP data types
  QByteArray datagram;
  QHostAddress sender;
  quint16 senderPort;
  // init File Handling
  m_file.open(QIODevice::WriteOnly | QIODevice::Truncate);

  do
  {
    datagram.resize(udp_rcv_socket.pendingDatagramSize());
    udp_rcv_socket.readDatagram(datagram.data(), datagram.size(), &sender, &
        senderPort);
    if (saveToFile)
    {
      m_outStream << datagram;
    }
    if (DEBUG)
      qout << "UDP␣received␣with␣size:␣" << datagram.size() << endl;

    if (datagram.size() == PD_SIZE_TOTAL)
    {
      eventBuilder->addFEPacket(datagram);
    }
    if (datagram.size() == LENGTH_OF_SAMPLE_PACKET)
    {
      eventBuilder->addSEPacket(datagram);
    }
  } while (udp_rcv_socket.hasPendingDatagrams());
}
```

Listing E.5: Class `SADC_FE_wrapper`, responsible for extracting the information contained in the feature extraction network packet.

```
class SADC_FE_wrapper
{
public:
  SADC_FE_wrapper(const QByteArray & dataFE);
  ~SADC_FE_wrapper();
  unsigned int getCompilationDatecode() const;
  unsigned int getBurstCounter() const;
  unsigned int getPacketPositionInBurst() const;
  unsigned int getTotalPositionsInBurst() const;
  unsigned int getEventID() const;
  unsigned int getDaqBufNum() const;
  long double getTimestampTrigger() const;
  unsigned int getExpectedSamples() const;
  double getTimestampCFD(int channel) const;
  double getTimestampMax(int channel) const;
  double getBaseline(int channel) const;
  double getIntegral(int channel) const;
  double getMax(int channel) const;

private:
  const QByteArray& m_data;
};
```

Listing E.6: Class `SADC_SE_wrapper`, responsible for extracting the information and sample data contained in the feature extraction network packet.

```
class SADC_SE_wrapper
{
public:
  SADC_SE_wrapper(const QByteArray & dataSP);
  ~SADC_SE_wrapper();
  unsigned int getBurstCounter() const;
  unsigned int getPacketPositionInBurst() const;
  unsigned int getTotalPositionsInBurst() const;
  unsigned int getEventID() const;
  unsigned int getDaqBufNum() const;
  unsigned int getChannel() const;
  unsigned int getPart() const;
  const QByteArray extractSample() const;

private:
  const QByteArray& m_data;
};
```

Listing E.7: Minimal example to record waveforms with a Python script. The libraries are shown in [140].

```
import sys
sys.path.append('..')
from lib.network.Network import sender as senderLib
from lib.network.Network import receiver as receiverLib
from lib.network.sadcConfig import sadcConfig as sadcConfigLib
import time
import logging
import threading
import os

# define path where waveform files will be stored
path = "D:/Workspaces/SADC-Test/"

recordBaseline  = True
recordingTimeBL = 1
thresholdNoise  = 2
waveformShift   = 60
fpgaPosition    = 'upper'

# define local IP and receiving UDP port
# upper/lower refers to the position of the FPGA/signals,
# note that the SFPs are inverted (upper connected to lower FPGA)

ipLocal      = "192.168.0.5" # IP is hardcoded to the firmware, don't change!
ipSadcLower  = "192.168.0.12"
ipSadcUpper  = "192.168.0.13"
portLower    = 50012
portUpper    = 50013

if fpgaPosition == 'upper':
    ip = ipSadcUpper
    port = portUpper
elif fpgaPosition == 'lower':
    ip = ipSadcLower
    port = portLower
else:
    print("Error! Please select upper or lower for the FPGA")
    raise SystemExit

# init logging
logging.basicConfig(level=logging.DEBUG)
logger = logging.getLogger("main")
# init network
receiver        = receiverLib(ipLocal, port)
sender          = senderLib(ip, debug=True)
config          = sadcConfigLib(sender, debug=True)

##################################
#       starting script          #
##################################

serialnumber = input('Enter serial number: ')

path_recording = path + "SN" + str(serialnumber) + "/"
```

```
55
56 # check whether data was taken already for this SN
57 if not os.path.exists(path_recording):
58     os.makedirs(path_recording)
59     os.makedirs(path_recording + "raw/")
60 else:
61     print("Warning! Folder exists! Script will be halted, so that data is not
           overwritten. Please manually delete the folder.")
62     raise SystemExit
63
64 ##################################
65 #    start recording baselines   #
66 ##################################
67
68 if recordBaseline:
69     input('Please connect network cable to correct SFP slot')
70     # always issue this reset before recording, e.g. to reset event counter
71     config.modeReset()
72     # shift the waveform with regard to the trigger, set stretcher
73     config.setWaveformPosition(waveformShift)
74     config.setWaveformStretcher(0)
75     # activate sample sender for all waveforms;
76     # alternatively activate only the channels you need.
77     for i in range(32):
78         config.waveformEnable(i)
79     # disable trigger on all channels
80     config.setInternalTriggerMask(0)
81     # enable sadc to send (after all configurations were done)
82     config.modePush()
83     # set filename and open file
84     filename = path_recording + "raw/" + serialnumber + ".bin"
85     file = open(filename, "ab")
86     logger.info("Open file " + filename)
87     # start UDP receiver
88     receiver_thread = threading.Thread(target=receiver.start, args=(file,
           True))
89     receiver_thread.daemon = True
90     logger.info("Starting receiver thread")
91     receiver_thread.start()
92     # set SADC to trigger on channel 0 at noise level,
93     config.setTriggerThreshold(0, thresholdNoise)
94     # activate internal trigger for channel 0
95     # after this, SADC will start sending
96     config.setInternalTriggerSingleChannel(0)
97     # record for "recordingTimeBL" seconds
98     time.sleep(recordingTimeBL)
99     # stop the receiver, thread will terminate after next packet is received
100    receiver.stop()
101    receiver_thread.join()
102    # disable trigger
103    config.setInternalTriggerMask(0)
104    file.close()
```

Listing E.8: Configuration options of the `sadc64-config` command line tool written by C. Schmitt.

```
[daq-tr@sadc-teststation ~]$ /home/muellers/workspace/GIT-DAQ-TR/build/master/utilities/sadc64/test/sadc64-config --help
Allowed options:
  --help                        output help message
  --sadc-ip arg (=10.0.1.35)    ip of sadc to send config data
  --dest arg                    sets ip to where UDP packets will be sent
  --clear-counter               clear counter of sadc
  --waveform-all-on             activates ALL channel for which waveform should be
                                sent
  --waveform-all-off            deactivates ALL channel for which waveform should
                                be sent
  --waveform-on arg             activates channel for which waveform should be
                                sent (ch number)
  --waveform-off arg            deactivates channel for which waveform should be
                                sent (ch number)
  --waveform-prescaler arg      set scaler after how many counts a waveform is
                                sent
  --exttrigger arg              activates external trigger (e.g. 0=off/4=on)
  --inttrigger arg              activates channel for which waveform should be
                                sent (ch number)
  --ortrigger arg               activates channel for or trigger (e.g. 0
                                x00001111)
  --triggerdelay arg            shift triggerposition-dependent routines in 12.5
                                ns
                                steps
  --waveform-mask arg           activates channel for which waveform should be
                                sent (bitmask)
  --led-all arg                 LED threshold for all channels
  --led-ch00 arg                LED threshold for channel 0
  --cfd-all arg                 CFD threshold for all channels
  --cfd-ch00 arg                CFD threshold for channel 0
  --cfd-tot-all arg             CFD time over threshold for all channels
  --cfd-def-all arg             CFD defuse time for all channels
  --max-all arg                 MAX rise time for all channels
  --reset                       set reset mode (reset FE/buffers)
  --sync                        set sync mode (send ack every 16 packet)
  --async                       set async mode (push)
  --default                     set default values
```

Listing E.9: XML file as used by the LEvB for the configuration of one CB-SADC. The configuration options correspond to Table C.1.

```
<TESTNEWSA>
  <MODULES>
    <VME>
      <CVMEFPGA_0
        name=      "VME␣Sync␣Client"
        baseaddress=  "0x00fc0000"
        binfile=     "/home/hoffmeis/fpgabinfiles/sync_client5_vfa.bit"
        syncid=      "13"
      />
      <CSIMPLETRIGGER
        name=      "VME␣Sync␣Client"
        baseaddress=  "0x00fc0000"
        binfile=     "/home/hoffmeis/eclipse/DAQ-TR/utilities/vme_fpga_utility
            /simple_trigger_sync_mezz.bit"
        syncid=      "1"
      />
    </VME>
  </MODULES>
  <READOUT>
    <SADC_0 index="12" ip="192.168.0.12"
      server-ip="192.168.0.5"    server-port="50012"
      burstsize="32"
      triggermask="0x00000002"  triggerdelay="50"
      waveformmask="0xffffffff" waveformprescaler="99"
      waveformstretcher="0"
      intpos="125"
      ledthreshold="40"    maxtime="20"
      cfdthreshold="4"     cfdtot="12" cfddef="32"
      pileupratio="164"    pileupthreshold="5"
      pileupupperslope="1023" pileuplowerslope="1023"
    />
    <SADC_1 index="13" ip="192.168.0.13"
      server-ip="192.168.0.5"    server-port="50013"
      burstsize="32"
      triggermask="0x00000002"  triggerdelay="50"
      waveformmask="0xffffffff" waveformprescaler="99"
      waveformstretcher="0"
      intpos="125"
      ledthreshold="40"    maxtime="20"
      cfdthreshold="4"     cfdtot="12" cfddef="32"
      pileupratio="164"    pileupthreshold="5"
      pileupupperslope="1023" pileuplowerslope="1023"
    />
  </READOUT>
  <USERDEFINE>
    <ZEBRA bankname="X1AF" />
    <!-- mode= 3:all, 2: waveforms, 1:feature, 0: nothing -->
    <DUMPFILE namepattern="/mnt/SSD_RAID0/cbsadc_Waveforms_%d_1.zbin" mode="2
        "/>
  </USERDEFINE>
</TESTNEWSA>
```

Listing E.10: Function declarations for the readout of the CB-SADC I$^2$C sensors, as implemented in the Crystal Barrel slowcontrol.

```
bool SadcBlockRead();
// automated readout of all sensors
bool SadcBlockInitNonVolatile();
// initialization of all I2C components with non-volatile settings
// --> required only once
bool SadcBlockInitVolatile();
// initialization of all I2C components with non-volatile settings
bool SadcTurnOn(unsigned int block, unsigned int sadc_nr);
// turn on the PMBus Control signal on the backplane
bool SadcTurnOff(unsigned int block, unsigned int sadc_nr);
// turn off the PMBus Control signal on the backplane
bool SadcGetId(unsigned int block, unsigned int sadc_nr, unsigned int &id);
// read the unique 48 bit ID embedded on the extension card
bool SadcGet6V(unsigned int block, unsigned int sadc_nr, unsigned int &
    voltage, unsigned int &current);
// read 6V voltage and current on the primary side of the power supply
bool SadcGet12V(unsigned int block, unsigned int sadc_nr, unsigned int &
    voltage, unsigned int &current);
// read 12V voltage and current on the primary side of the power supply
bool SadcGetTempFPGA1(unsigned int block, unsigned int sadc_nr, unsigned int
    &temperature);
// read temperature of the PCB close to lower FPGA
bool SadcGetTempFPGA2(unsigned int block, unsigned int sadc_nr, unsigned int
    &temperature);
// read temperature of the PCB close to upper FPGA
bool SadcGetTempPLL(unsigned int block, unsigned int sadc_nr, unsigned int &
    temperature);
// read temperature of the PCB close to LMK04806
bool SadcGetTempLDO(unsigned int block, unsigned int sadc_nr, unsigned int &
    temperature);
// read temperature of the PCB close to MAX8556 LDOs
bool SadcGetTempPS(unsigned int block, unsigned int sadc_nr, unsigned int &
    temperature);
// read temperature of the power supply PCB
bool SadcGetTempBLV1(unsigned int block, unsigned int sadc_nr, unsigned int &
    temperature);
// read temperature of the lower analog input card PCB
bool SadcGetTempBLV2(unsigned int block, unsigned int sadc_nr, unsigned int &
    temperature);
// read temperature of the upper analog input card PCB
bool SadcSetBaseline(unsigned int block, unsigned int sadc_nr, unsigned int
    channel, unsigned int baseline);
// set the baseline value on the analog input card
bool SadcGetBaseline(unsigned int block, unsigned int sadc_nr, unsigned int
    channel, unsigned int &baseline);
// read the baseline value on the analog input card
bool SadcSetPoleZero(unsigned int block, unsigned int sadc_nr, unsigned int
    channel, unsigned int polezero);
// set the potentiometer for pole zero compensation
bool SadcGetPoleZero(unsigned int block, unsigned int sadc_nr, unsigned int
    channel, unsigned int &r_aw, unsigned int &r_wb);
// read the potentiometer for pole zero compensation
```

# F. Other Figures

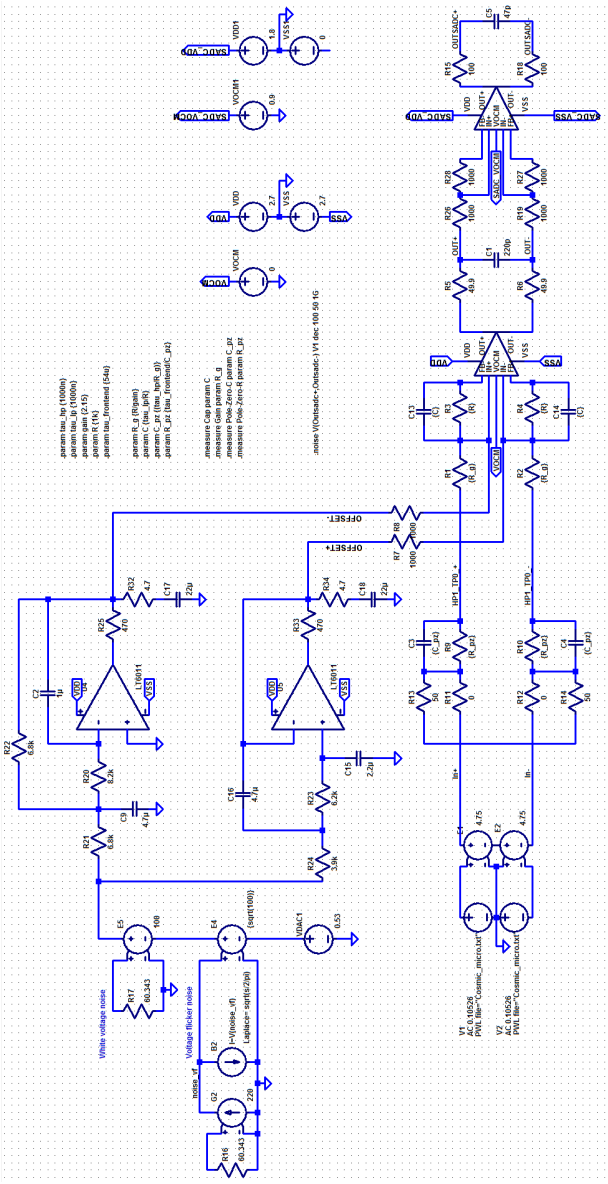


Figure F.1.: Schematics of the noise density simulation. The simulation was created by Timo Poller and modified within this thesis.

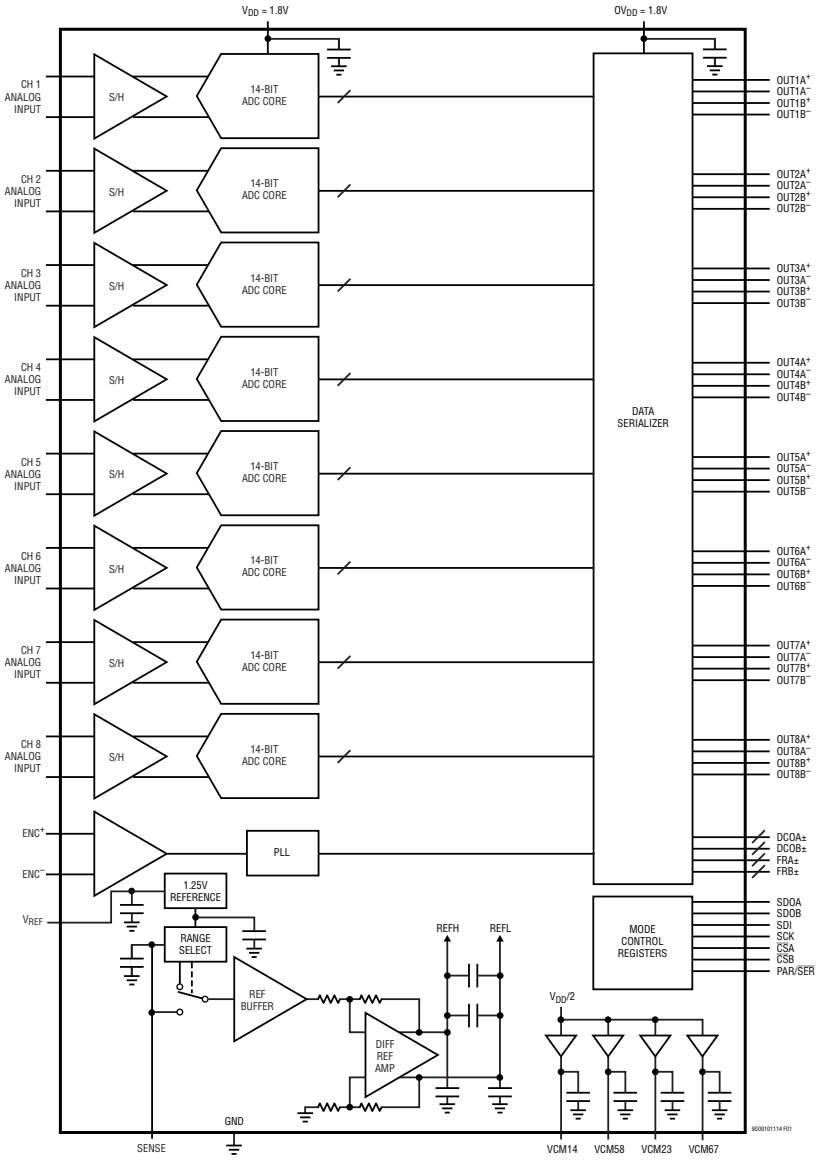



Figure F.2.: Functional block diagram of the Linear TechnologiesLtm9009-14. [149]

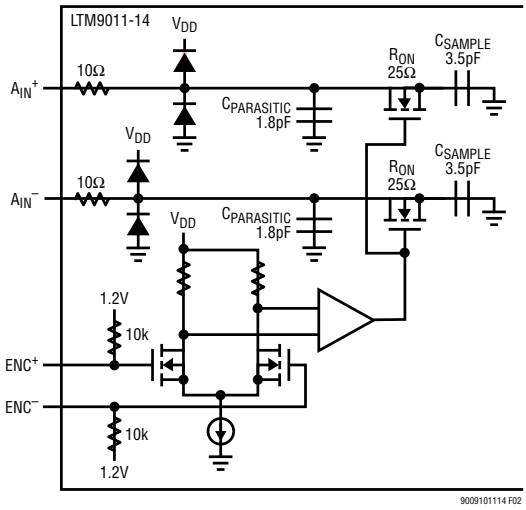



Figure F.3.: Sample and hold circuit of the Linear Technologies Ltm9009-14. [149]

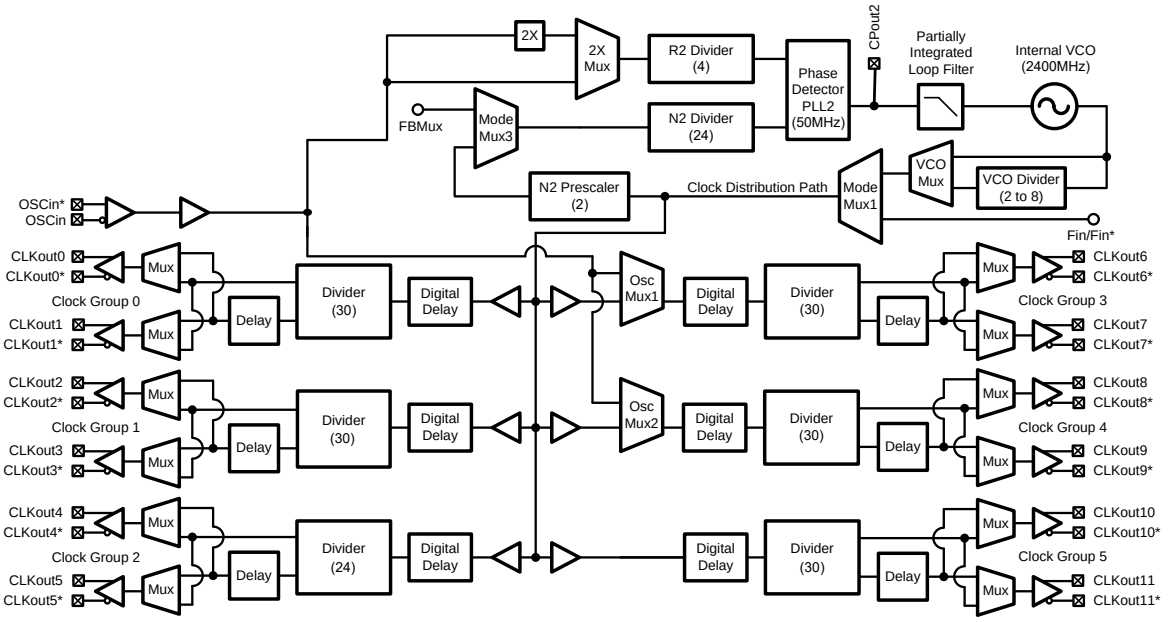



Figure F.4.: Detailed functional block diagram of the Lmk04806 in single PLL mode. The reference clock is fed from the `OSCin` inputs. The dividers and prescalers on top assure the correct VCO frequency. The clock output paths share common divider blocks in pairs. [150] (modified)

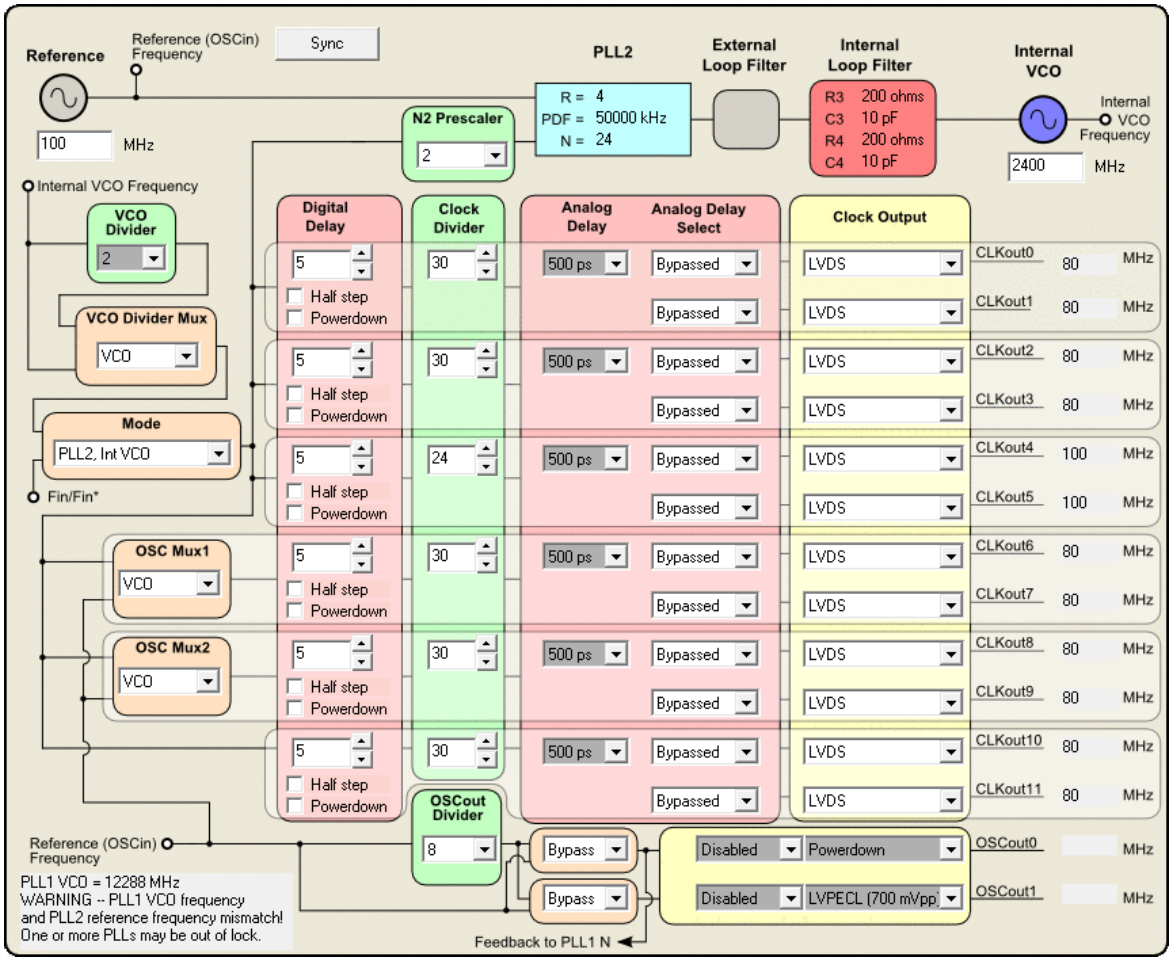



Figure F.5.: Configuration of the LMK04806 in single PLL mode, visualized by the TEXAS INSTRUMENTS CodeLoader software.

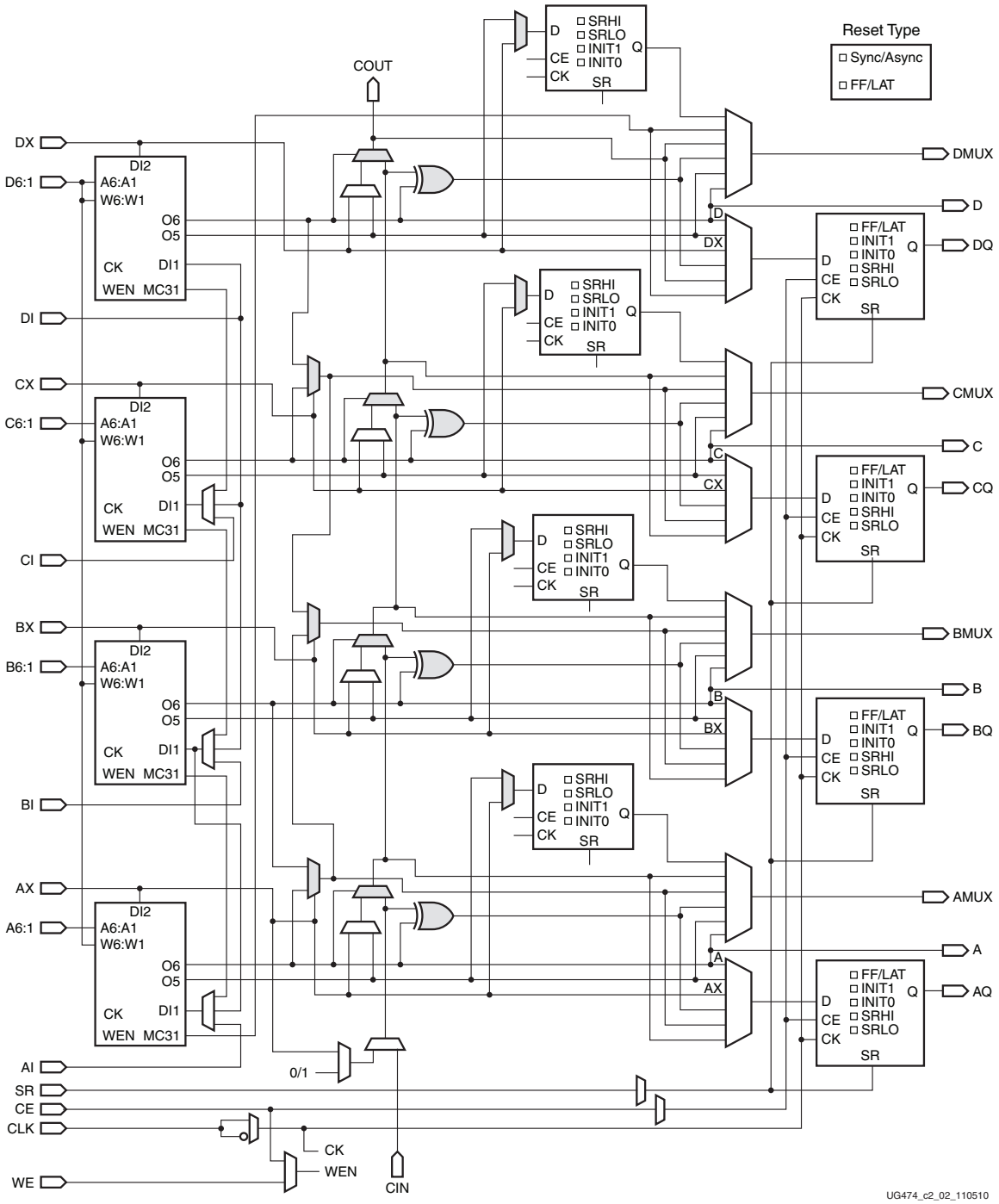



Figure F.6.: Diagram of a SLICEM primitive inside a CLB of the Xilinx Kintex-7 FPGA. The four boxes on the left are lookup-tables, the other eight boxes are flip-flops. Elements in between are logic gates and muxers. [194]

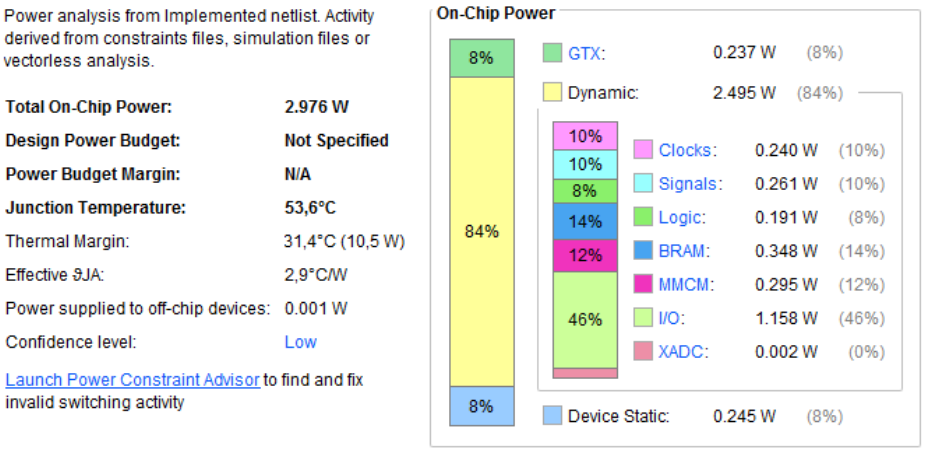



Figure F.7.: Summary of the power estimation of Xilinx Vivado, based on the firmware design (revision `1802031942`) and estimated toggle rates as per Fig. F.8a.

**Toggle Rate Settings**

| | Static Probability | | Toggle Rate | |
|---|---|---|---|---|
| Primary Outputs: | 0.5 | [0.0 - 1.0] | 12.5 | [0 - 100] |
| Logic | | | | |
| Registers: | 0.5 | [0.0 - 1.0] | 12.5 | [0 - 100] |
| Shift Registers: | 0.5 | [0.0 - 1.0] | 12.5 | [0 - 100] |
| Distributed RAMs: | 0.5 | [0.0 - 1.0] | 100 | [0 - 100] |
| LUTs: | 0.5 | [0.0 - 1.0] | 12.5 | [0 - 100] |
| DSPs: | 0.5 | [0.0 - 1.0] | 100 | [0 - 100] |
| Block RAMs: | 0.5 | [0.0 - 1.0] | 5 | [0 - 100] |
| GTs | | | | |
| RX Data: | 0.5 | [0.0 - 1.0] | 2.5 | [0 - 100] |
| TX Data: | 0.5 | [0.0 - 1.0] | 12.5 | [0 - 100] |

(a) Estimates for the toggle rate.

| Supply Source | Voltage (V) | Total (A) | Dynamic (A) | Static (A) |
|---|---|---|---|---|
| Vccint | 1.000 | 1.240 | 1.099 | 0.142 |
| Vccaux | 1.800 | 0.596 | 0.573 | 0.022 |
| Vcco33 | 3.300 | 0.000 | 0.000 | 0.000 |
| Vcco25 | 2.500 | 0.105 | 0.104 | 0.001 |
| Vcco18 | 1.800 | 0.051 | 0.050 | 0.001 |
| Vcco15 | 1.500 | 0.000 | 0.000 | 0.000 |
| Vcco135 | 1.350 | 0.000 | 0.000 | 0.000 |
| Vcco12 | 1.200 | 0.000 | 0.000 | 0.000 |
| Vccaux_io | 1.800 | 0.000 | 0.000 | 0.000 |
| Vccbram | 1.000 | 0.038 | 0.025 | 0.013 |
| MGTAVcc | 1.000 | 0.116 | 0.109 | 0.007 |
| MGTAVtt | 1.200 | 0.101 | 0.098 | 0.003 |
| MGTVccaux | 1.800 | 0.000 | 0.000 | 0.000 |
| Vccadc | 1.800 | 0.021 | 0.001 | 0.020 |

(b) Current consumption, divided in dynamic and static current.

Figure F.8.: Details of the FPGA power estimation. Although it is easily possible to find a configuration that matches the actual measured current, it is hardly possible to estimate the true toggle rates of the respective components in the real setup.

# Bibliography


[1] W. Kester, ed. *The Data Conversion Handbook*. Newnes, 2005. ISBN: 0-916550-27-3. URL: http://www.analog.com/en/education/education-library/data-conversion-handbook.html.

[2] W. Kester. 'Which ADC Architecture Is Right for Your Application?' In: *Analog Dialogue* 39 (June 2005). URL: http://www.analog.com/media/en/analog-dialogue/volume-39/number-2/articles/the-right-adc-architecture.pdf.

[3] *Thermorezeption (german article)*. URL: https://de.wikipedia.org/wiki/Thermorezeption (visited on 05/30/2017).

[4] E. Smith. *Understanding the Delta-Sigma ADC*. July 2016. URL: https://www.allaboutcircuits.com/technical-articles/understanding-the-delta-sigma-analog-to-digital-converter/.

[5] A. v. R. E. Janssen. *Basics of Sigma-Delta Modulation*. Springer, Dordrecht, 2011. DOI: 10.1007/978-94-007-1387-1_2.

[6] H. Spieler. *Semiconductor Detector Systems*. Oxford University Press, 2008. ISBN: 978-0-19-852784-8.

[7] P. Dutta and P. M. Horn. 'Low-frequency fluctuations in solids: $\frac{1}{f}$ noise'. In: *Rev. Mod. Phys.* 53 (3 July 1981), pp. 497–516. DOI: 10.1103/RevModPhys.53.497.

[8] *Datasheet ADA4940*. Analog Devices. URL: http://www.analog.com/media/en/technical-documentation/data-sheets/ADA4940-1_4940-2.pdf (visited on 05/01/2016).

[9] W. R. Bennet. 'Spectra of Quantized Signals'. In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 446–471. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01340.x.

[10] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice-Hall, Inc., 1999. ISBN: 0-13-754920-2.

[11] R. H. Walden. 'Analog-to-digital converter survey and analysis'. In: *IEEE Journal on Selected Areas in Communications* 17.4 (Apr. 1999), pp. 539–550. ISSN: 0733-8716. DOI: 10.1109/49.761034.

[12] W. Kester. 'ADC Input Noise: The Good, The Bad, and The Ugly. Is No Noise Good noise?' In: *Analog Dialogue* 40 (Feb. 2006). URL: http://www.analog.com/media/en/analog-dialogue/volume-40/number-1/articles/adc-input-noise.pdf.

[13] B. Brannon and A. Barlow.
*AN-501: Aperture Uncertainty and ADC System Performance*. Analog Devices.
URL: http://www.analog.com/media/en/technical-documentation/application-notes/AN-501.pdf.

[14] M. Löhning and G. Fettweis.
'The effects of aperture jitter and clock jitter in wideband ADCs'.
In: *Computer Standards and Interfaces* 29.1 (Jan. 2007). Ed. by D. Petri and S. Rapuano, pp. 11–18. DOI: 10.1016/j.csi.2005.12.005.

[15] H. Nyquist. 'Certain Topics in Telegraph Transmission Theory'. In: *Transactions of the American Institute of Electrical Engineers* 47.2 (1928), pp. 617–644.
ISSN: 0096-3860. DOI: 10.1109/T-AIEE.1928.5055024.

[16] C. E. Shannon. 'A mathematical theory of communication'.
In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423.
ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[17] P. A. Tipler and R. A. Llewellyn. *Modern Physics*. 5th ed.
W. H. Freeman and Company, 2008. ISBN: 0-7167-7550-6.

[18] M. T. et al. (Particle Data Group). 'The Review of Particle Physics'.
In: *Phys. Rev. D* 98.030001 (2018). URL: http://pdg.lbl.gov/2018/.

[19] A. Gerhardus and P. Stinner.
'Determining the Chemical Compositions of Planetary Nebula by Spectroscopy'.
In: *Spektrum* (Feb. 2011), pp. 16–20.
URL: http://spektroskopie.vdsastro.de/files/pdfs/Spektrum41.pdf.

[20] J. Hartmann. 'Measurement of Double Polarization Observables in the Reactions $\gamma p \to p\pi^0$ and $\gamma p \to p\eta$ with the Crystal Barrel/TAPS Experiment at ELSA'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2017.
URL: http://hss.ulb.uni-bonn.de/2017/4833/4833.pdf.

[21] U. Loering, B. C. Metsch, and H. R. Petry.
'The light baryon spectrum in a relativistic quark model with instanton-induced quark forces I. The non-strange baryon spectrum and ground-states'.
In: *Eur.Phys.J. A* 10 (Mar. 2001), pp. 395–446. DOI: 10.1007/s100500170105.

[22] M. Ronniger and B. C. Metsch.
'Effects of a spin-flavour-dependent interaction on the baryon mass spectrum'.
In: *Eur.Phys.J. A* 47.162 (Dec. 2011). DOI: 10.1140/epja/i2011-11162-8.

[23] K. N. et al. (Particle Data Group). 'The Review of Particle Physics'.
In: *J. Phys. G* 37.075021 (2010). URL: http://pdg.lbl.gov/2010/.

[24] R. G. Edwards et al. 'Excited state baryon spectroscopy from lattice QCD'.
In: *Phys. Rev. D* 84.074508 (2011). DOI: 10.1103/PhysRevD.84.074508.

[25] W. Hillert. *Technical overview of ELSA*.
URL: https://www-elsa.physik.uni-bonn.de/Beschleuniger/elsatech.html (visited on 06/07/2002).

[26] M. Schedler. 'Intensitäts- und Energieerhöhung an ELSA'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2015.
URL: http://hss.ulb.uni-bonn.de/2015/4207/4207.pdf.

[27] F. Klarner. 'Konzeption, Aufbau und Inbetriebnahme eines neuen Vorbeschleunigersystems an ELSA'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2011.
URL: http://hss.ulb.uni-bonn.de/2011/2721/2721.pdf.

[28] M. Schedler et al. 'A New High Current and Single Bunch Injector at ELSA'.
In: *Proceedings of the 27th International Linear Accelerator Conference* (Geneva, Switzerland). Ed. by C. Carli et al. JACoW, Dec. 2014, pp. 847–849.
ISBN: 978-3-95450-142-7.
URL: http://jacow.web.psi.ch/conf/linac14/prepress/THPP005.PDF.

[29] F. Klarner. 'Ein neues Injektorsystem zur Erzeugung von Einzelpulsen für den Elektronenbeschleuniger ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.
URL: https://www-elsa.physik.uni-bonn.de/Publikationen/texte/klarner_dipl.pdf.

[30] W. Hillert. 'Erzeugung eines Nutzstrahls spinpolarisierter Elektronen an der Beschleunigeranlage ELSA'.
Habilitation. Rheinische Friedrich-Wilhelms-Universität Bonn, 2002. URL: https://www-elsa.physik.uni-bonn.de/Publikationen/texte/hillert_habil.pdf.

[31] D. Heiliger. 'Erzeugung intensiver spinpolarisierter Elektronenstrahlen an der Beschleunigeranlage ELSA'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2014.
URL: http://hss.ulb.uni-bonn.de/2014/3725/3725.pdf.

[32] D. Heiliger, W. Hillert, and B. Neff.
'A New Load Lock System for the Source of Polarized Electrons at ELSA'.
In: *Proceedings of the 4th International Particle Accelerator Conference* (Shanghai, China). Ed. by Z. Dai et al. JACoW, June 2013, pp. 294–296.
ISBN: 978-3-95450-122-9. URL:
http://accelconf.web.cern.ch/AccelConf/IPAC2013/papers/mopfi006.pdf.

[33] K. H. Althoff et al. 'The 2.5 GeV Electron Synchrotron of the University of Bonn'.
In: *Nuclear Instruments and Methods* 61 (1968), pp. 1–30.
DOI: 10.1016/0029-554X(68)90443-6.

[34] H. D. Nuhn. 'Schnelle Extraktion aus dem 2.5 GeV Synchrotron und Strahltransfer zum neuen Stretcherring ELSA'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 1988.

[35] J. Hartmann. *Personal communication.* 2019.

[36] D. Husmann et al. 'ELSA - The Continuous Beam Accelerator at Bonn'.
In: *First European Particle Accelerator Conference* (Rome, Italy). Ed. by L. Invidia. JACoW, June 1988, pp. 356–358.
URL: http://accelconf.web.cern.ch/AccelConf/e88/PDF/EPAC1988_0356.PDF.

[37] W. Hillert. 'The Bonn Electron Stretcher Accelerator ELSA: Past and future'. In: *Eur.Phys.J. A* 28 (May 2006), pp. 139–148. DOI: 10.1140/epja/i2006-09-015-4.

[38] F. Frommberger. *Publications of the accelerator group.* URL: https://www-elsa.physik.uni-bonn.de/Publikationen/index_en.html (visited on 02/09/2018).

[39] W. Hillert. *The ELectron Stretcher Accelerator ELSA.* URL: https://www-elsa.physik.uni-bonn.de (visited on 03/01/2013).

[40] D. Walter. *Technical drawing.* 2013.

[41] M. Lang. *Personal communication.* 2019.

[42] D. Elsner. 'Untersuchung kleiner Partialwellenbeiträge in der Nähe dominierender Resonanzzustände des Protons mit linear polarisierten Photonen'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2007. URL: http://hss.ulb.uni-bonn.de/2007/1050/1050.pdf.

[43] H. Olsen and L. C. Maximon. 'Photon and Electron Polarization in High-Energy Bremsstrahlung and Pair Production with Screening'. In: *Phys. Rev.* 114 (1959), p. 887. DOI: 10.1103/PhysRev.114.887.

[44] S. Kammer. 'Strahlpolarimetrie am CBELSA/TAPS Experiment'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2010. URL: http://hss.ulb.uni-bonn.de/2010/2056/2056.pdf.

[45] G. Urff and P. Hoffmeister. *Personal communication.* 2017.

[46] K. Fornet-Ponse. 'Die Photonenmarkierungsanlage für das Crystal-Barrel/TAPS-Experiment an ELSA'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2010. URL: http://hss.ulb.uni-bonn.de/2010/1998/1998.pdf.

[47] C. Hammann. 'Aufbau eines Flüssigwasserstofftargets zur Durchführung von Kalibrationsmessungen am Crystal-Barrel Experiment an ELSA'. Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[48] C. Bradtke et al. 'A new frozen-spin target for $4\pi$ detection'. In: *Nuclear Instruments and Methods* A 436 (1999), pp. 430–442. DOI: 10.1016/S0168-9002(99)00383-6.

[49] J. Ahrens et al. 'Helicity Dependence of $\gamma p \to N\pi$ below 450 MeV and Contribution to the Gerasimov-Drell-Hearn Sum Rule'. In: *Phys. Rev. Lett.* 84 (June 2000), pp. 5950–5954. DOI: 10.1103/PhysRevLett.84.5950.

[50] H. Dutz et al. 'First Measurement of the Gerasimov-Drell-Hearn Sum Rule for $^1$H from 0.7 to 1.8 GeV at ELSA'. In: *Phys. Rev. Lett.* 91 (Nov. 2003), p. 192001. DOI: 10.1103/PhysRevLett.91.192001.

[51] S. Runkel. *Personal communication.* 2018.

[52] S. Goertz. *Internal Note: The Art of Defining the Target Spin Degrees of Freedom - Requirements, principles and major components.* Apr. 2018.

[53] S. Runkel et al. 'CFD-Simulations of a 4$\pi$-continuous-mode dilution refrigerator for the CB-ELSA experiment'. In: *Proceedings, 16th International Workshop on Polarized Sources, Targets, and Polarimetry (PSTP 2015)* (Bochum, Germany). Vol. PSTP2015. Sept. 2015.
URL: http://pos.sissa.it/archive/conferences/243/018/PSTP2015_018.pdf.

[54] A. Foesel. 'Entwicklung und Bau des Innendetektors für das Crystal-Barrel-Experiment an ELSA/Bonn'.
Doctoral Thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg, 2000.
URL: http://www.pi4.nat.uni-erlangen.de/novel-detectors/publications/phd-diplomathesis/foesel_dr.pdf.

[55] G. Suft et al.
'A scintillating fibre detector for the Crystal Barrel experiment at ELSA'.
In: *Nuclear Instruments and Methods* A 538 (Feb. 2005), pp. 416–424.
DOI: 10.1016/j.nima.2004.09.029.

[56] M. Grüner.
'Modifikation und Test des Innendetektors für das Crystal Barrel Experiment'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.

[57] M. Grüner and D. Schaab. *Personal communication.* Jan. 2017.

[58] M. Grüner. *Internal note: Technical drawing of the Inner Detector.* Feb. 2019.

[59] E. Aker, C. Amsler, and I. Augustin. 'The Crystal Barrel spectrometer at LEAR'.
In: *Nuclear Instruments and Methods* A 321 (1992), pp. 69–108.
DOI: 10.1016/0168-9002(92)90379-I.

[60] J. Müller. 'Bestimmung einer Energiekorrekturfunktion für das Kalorimeter des Crystal-Barrel-Experiments an ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2007.

[61] C. Honisch. 'Design, Aufbau und Test einer neuen Ausleseelektronik für das Crystal-Barrel-Kalorimeter'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2014.
URL: http://hss.ulb.uni-bonn.de/2015/4111/4111.pdf.

[62] E. Gutz. 'Entwicklung eines zusätzlichen Trigger-Szintiallationsdetektors für den Vorwärtskonus des Crystal Barrel Detektors an ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2003.

[63] C. Wendel. 'Entwicklung eines Szintillations-Detektors zur Identifizierung geladener Teilchen im Crystal-Barrel Vorwärtsdetektor'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2004.

[64] C. Wendel. 'Design und Aufbau eines Szintillationsdetektors zur Identizierung geladener Teilchenim Crystal-Barrel-Vorwärtsdetektor'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2008.
URL: http://hss.ulb.uni-bonn.de/2008/1470/1470.pdf.

[65] M. Grüner. *Technical drawing.* 2016.

[66] D. Kaiser. 'Aufbau und Test des Gas-Cerenkov-Detektors für den Crystal-Barrel-Aufbau an ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2007.

[67] R. Novotny. 'Performance of the $BaF_2$-calorimeter TAPS'.
In: *Nuclear Physics* B(61B) (1998), pp. 137–142.

[68] P. Drexler. 'Entwicklung und Aufbau der neuen TAPS-Elektronik'.
Doctoral Thesis. Universität Gießen, 2004.

[69] S. Diehl. 'Neue Auslesekonzepte für elektromagnetische Kalorimeter'.
Bachelor Thesis. Justus-Liebig-Universität Gießen, 2010.

[70] S. Diehl et al. 'Readout concepts for the suppression of the slow component of BaF2 for the upgrade of the TAPS spectrometer at ELSA'.
In: *16th International Conference on Calorimetry in High Energy Physics* (Justus-Liebig-Universität, Apr. 6, 2014–Apr. 11, 2014). Vol. 587.
Gießen, Germany, 2015, p. 012044. DOI: 10.1088/1742-6596/587/1/012044.

[71] W. R. McGehee. 'The Gamma Intensity Monitor at the Crystal-Barrel-Experiment'.
Bachelor Thesis. Massachusetts Institute Of Technology, 2008.

[72] J. Dielmann. 'Entwicklung, Aufbau und Test eines Detektors zur Bestimmung des Photonenflusses an der Bonner Photonenmarkierungsanlage'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2008.

[73] U. Thoma. *D.3 Timing and tracking for the Crystal Barrel detector.*
URL: http://sfb-tr16.physik.uni-bonn.de/vfs/en/projects/d3/ (visited on 03/18/2013).

[74] M. Urban, C. Honisch, and M. Steinacher.
'The New APD Based Readout for the Crystal Barrel Calorimeter'.
In: *16th International Conference on Calorimetry in High Energy Physics* (Justus-Liebig-Universität, Apr. 6, 2014–Apr. 11, 2014). Vol. 587.
Gießen, Germany, 2015, p. 012043. DOI: 10.1088/1742-6596/587/1/012043.

[75] A. Winnebeck. 'Design Studies for a Tracking Upgrade of the Crystal Barrel Experiment at ELSA and Installation of a Tracking Test Bench'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[76] T. Seifen. 'Verbesserung der Rekonstruktion und Entwicklung eines First-Level-Triggerschemas für das Crystal-Barrel-Kalorimeter'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[77] C. Honisch. 'Untersuchungen zu einer neuen Avalanche-Photodioden-Auslese für das Crystal-Barrel-Kalorimeter'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[78] F. Zenke. 'A new avalanche photodiode readout for the Crystal Barrel experiment'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[79] M. Wehrfritz. 'Entwicklung und Test eines Silizium-Photomultiplier-Triggers für das Crystal-Barrel-Kalorimeter'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2008.

[80] M. Gottschall. 'Verbesserung der Triggereigenschaften des Crystal-Barrel-Detektors an ELSA mit einer Silizium-Photomultiplier-Auslese der Kristalle'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2007.

[81] J. Bloemer. 'Verbesserung der Online-Zeitauflösung für die neue APD-Auslese'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2012.

[82] Hamamatsu. *Technical Information SD-28, Characteristics and use of Si APD (Avalanche Photodiode)*. KAPD9001E03. Hamamatsu, May 2004. URL: http://neutron.physics.ucsb.edu/docs/Avalanche_photodiodes_info.pdf.

[83] J. Kataokaa et al. 'Recent progress of avalanche photodiodes in high-resolution X-rays and γ-rays detection'.
In: *Nuclear Instruments and Methods* 541 (2005), pp. 398–404.
DOI: 10.1016/j.nima.2005.01.081.

[84] M. Urban. 'Design eines neuen Lichtpulsersystems sowie Aufbau und Inbetriebnahme der neuen APD Auslese für das Crystal-Barrel-Kalorimeter'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

[85] W. Erni. *Manual Preamplifier SP883d*. Uni Basel.

[86] M. Steinacher. *CB APD Single/Dual-Preamplifier*. Uni Basel, Nov. 2011.

[87] *Datasheet OPA2889*. Texas Instruments, Aug. 2008.
URL: http://www.ti.com/lit/ds/symlink/opa2889.pdf.

[88] J. Müllers. 'Design and Test of a Slowcontrol for the APD Upgrade of the Crystal Barrel Calorimeter'.
Master Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2013.

[89] *Datasheet THS4130*. Texas Instruments, Aug. 2015.
URL: http://www.ti.com/lit/ds/symlink/ths4130.pdf.

[90] *Datasheet OPA2613*. Texas Instruments, Aug. 2008.
URL: http://www.ti.com/lit/ds/symlink/opa2613.pdf.

[91] S. Ciupka. 'Test and calibration of a module for the new Energy-sum trigger for the Crystal Barrel Calorimeter at ELSA'.
Master Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

[92] B. Mitlasoczki.
'Design and Test of a New Energy Sum Trigger for the CBELSA/TAPS Experiment'.
Master Thesis (in preparation). Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[93] *ELB Elektroniklaboratorien Bonn*. ELB Elektroniklaboratorien Bonn.
URL: http://www.elbonn.de/.

[94] P. Klassen.
'Entwicklung eines neuen Cluster Finders für das Crystal-Barrel-Kalorimeter'.
Doctoral Thesis (in preparation). Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[95] F. N. Afzal. 'Measurement of the beam and helicity asymmetries in the reactions $\gamma p \to p\pi^0$ and $\gamma p \to p\eta$'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[96] K. Esslinger. *CB CsI-Shaper*. Physik-Institut der Universität Zürich, 1989.

[97] M. Steinacher. *Resultate PSpice Simulationen*. Uni Basel, Oct. 2010.

[98] *Datasheet 1885F ADC*. LeCroy. URL: http://teledynelecroy.com/lrs/dsheets/1885f.htm (visited on 01/01/1996).

[99] A. Ehmanns. 'Entwicklung, Aufbau und Test eines neuen Auslesesystems für den Crystal-Barrel-Detektor zur Messung photoinduzierter Reaktionen an ELSA'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2000. URL: http://hss.ulb.uni-bonn.de/2000/0144/0144.pdf.

[100] C. Schmidt. 'Entwicklung eines neuen Datenakquisitionssystems für das CB-ELSA-Experiment'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2004. URL: http://hss.ulb.uni-bonn.de/2004/0511/0511.pdf.

[101] *Datasheet ADC-521MC*. DATEL. URL: https://4donline.ihs.com/images/VipMasterIC/IC/DATI/DATID001/DATID001-2-29.pdf.

[102] P. Hoffmeister. *Personal communication*. Oct. 2017.

[103] *Product Information SIS 4100 NGF*. Struck Innovative Systems. URL: http://www.struck.de/sis4100.htm (visited on 04/07/2004).

[104] K. Braune. *The Fast Cluster Encoder*. 1986.

[105] H. Flemming. 'Entwurf und Aufbau eines Zellularlogik-Triggers für das Crystal-Barrel-Experiment an der Elektronenbeschleunigeranlage ELSA'. Doctoral Thesis. Ruhr-Universität Bochum, 2001. URL: http://www-brs.ub.ruhr-uni-bochum.de/netahtml/HSS/Diss/FlemmingHolger/diss.pdf.

[106] A. Winnebeck. 'Entwicklung und Implementierung eines universellen, FPGA basierten Triggermoduls für das Crystal-Barrel-Experiment an ELSA'. Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.

[107] H. van Pee. *Personal communication*. Sept. 2013.

[108] P. Hoffmeister. 'Das Datenerfassungssystem für das Crystal-Barrel/TAPS-Experiment an ELSA'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[109] A. Ehmanns. *Documentation of CEBRA banks*. Aug. 2002.

[110] D. Piontek. 'Entwicklung eines Online-Monitors für das Crystal-Barrel-Experiment an ELSA'. Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2007.

[111] J. Schultes. 'Thesis in preparation'. Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[112] B. Kämmle. 'Aufbau des Lichtpulsers für den CRYSTAL BARREL Detektor und Untersuchungen der Eigenschaften und Funktionsweise von Xenon-Blitzlampen'. Diploma Thesis. Universität Hamburg, 1989.

[113] J. Friedrich. 'Lichtpulsersystem zur Überwachung und Messung der Kennlinie der Kalorimeterelektronik des CRYSTAL BARREL Detektors'.
Diploma Thesis. Universität Hamburg, 1989.

[114] O. Bartholomy.
'Test und Modifikation des Lichtpulsersystems für den CB-ELSA-Detektor'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2000.

[115] S. Böse. 'Modifikation und Test des Lichtpulsersystems für den Crystal Barrel Aufbau an ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.

[116] M. Urban. 'Compensation of the Temperature Dependence of Avalanche Photodiodes for a new Readout of the Crystal Barrel Calorimeter'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2011.

[117] A. Thiel. 'Aufbau einer computergesteuerten Experimentüberwachung für den Crystal-Barrel-Aufbau an ELSA'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.

[118] S. Ciupka. 'Aufbau und Programmierung einer computergesteuerten Temperaturüberwachung des Crystal-Barrel-Experiments an ELSA'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2015.

[119] H. Stübner. 'Einbau der neuen Slowcontrol Parameter des CB-Kalorimeters in ein Datenbanksystem'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2016.

[120] *Application Report: Understanding the $I^2C$ Bus.* Texas Instruments, June 2015.
URL: http://www.ti.com/lit/an/slva704/slva704.pdf.

[121] *AN10658: Sending I2C-bus signals via long communications cables.*
NXP Semiconductor.
URL: http://www.nxp.com/documents/application_note/AN10658.pdf (visited on 07/19/2013).

[122] J. Schultes. 'Test and Improvement of Feature-Extraction Methods For the new SADC-Readout of the Crystal Barrel Calorimeter'.
Master Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2016.

[123] W. Skulski and M. Momayezi.
'Particle identification in CsI(Tl) using digital pulse shape analysis'.
In: *Nuclear Instruments and Methods* A 458.3 (2001), pp. 759–771. ISSN: 0168-9002. DOI: 10.1016/S0168-9002(00)00938-4.

[124] J. Silva et al. 'Pulse shape analysis using CsI(Tl) crystals'.
In: *3rd International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications* (June 23, 2013–June 27, 2013). Marseille, France: IEEE, 2013. DOI: 10.1109/ANIMMA.2013.6727967.

[125] P. Kreutz et al.
'Photodiode readout and pulse shape analysis of CsI(Tl) scintillator signals'.
In: *Nuclear Instruments and Methods* A 260.1 (1987), pp. 120–123. ISSN: 0168-9002. DOI: 10.1016/0168-9002(87)90392-5.

[126] S. Schaepe. 'Entwurf, Test und Aufbau einer Flash-ADC-Auslese fur das Crystal-Barrel-Kalorimeter'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2009.

[127] M. Gräf. 'Teilchenidentifikation mit der neuen SADC-Auslese des Crystal Barrel Kalorimeters'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[128] C. Funke. 'Untersuchungen zur Energieauflösung von CsI-Kristallen mit Hochgeschwindigkeits ADCs'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2002.

[129] T. Szczepanek. 'Photoproduktion γp -> π0π0 Ergebnisse und Entwicklung einer schnellen FADC-Auslese für Doppelpolarisationsexperimente'.
Doctoral Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2006.
URL: http://hss.ulb.uni-bonn.de/2006/0849/0849.pdf.

[130] P. Mahlberg. 'Untersuchungen zur neuen APD-Auslese mit Flash-ADC des Crystal-Barrel-Calorimeters'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2011.

[131] C. Schmidt. *Personal communication*. Oct. 2017.

[132] P. Marciniewski.
*Author of the "16-CHANNEL SERIAL ADC" schematic and PCB design*. 2011.

[133] M. Albrecht and the PANDA collaboration. 'The Forward Endcap of the Electromagnetic Calorimeter for the PANDA Detector at FAIR'.
In: *16th International Conference on Calorimetry in High Energy Physics* (Justus-Liebig-Universität, Apr. 6, 2014–Apr. 11, 2014). Vol. 587.
Gießen, Germany, 2015, p. 012050. DOI: 10.1088/1742-6596/587/1/012050.

[134] P. Marciniewski. 'ADC for EMC'.
Talk presented at the PANDA DAQ Workshop, GSI Darmstadt. 2015.
URL: https://indico.gsi.de/event/3430/session/3/contribution/5.

[135] P. Collaboration. *PANDA Technical Design Report*. 2008.
URL: https://panda.gsi.de/oldwww/archive/public/panda_tdr_EMC.pdf.

[136] A. Mann, I. Konorov, and S. Paul. 'A Versatile Sampling ADC System for On-Detector Applications and the AdvancedTCA Crate Standard'.
In: *15th IEEE-NPSS Real-Time Conference* (2007).
DOI: 10.1109/RTC.2007.4382728.

[137] P. Marciniewski. *Author of the "ADC-64K" schematic and PCB design*. 2014.

[138] G. Urff. 'Design of a Slowcontrol Module for the Crystal Barrel Sampling Analog to Digital Converters'.
Master Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2015.

[139] T. Poller. 'Simulation, Aufbau und Test einer Schaltung zur Baseline-Verschiebung der Crystal Barrel Kalorimetersignal für den Sampling-ADC'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2016.

[140] Y.-C. Wang.
'Development of a Quality Assurance Procedure for the Crystal Barrel SADC'.
Master Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

[141] J. Knaust.
'Development and Test of a Crate Controller for the Crystal Barrel SADC'. Master Thesis (in preparation). Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[142] L. Zywietz Rolon.
'Optimierung der Baseline-Bestimmung für die Crystal Barrel Sampling ADCs'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[143] *1G/2.5G Ethernet PCS/PMA or SGMII v16.0*. Xilinx, Apr. 2017.
URL: https://www.xilinx.com/support/documentation/ip_documentation/gig_ethernet_pcs_pma/v16_0/pg047-gig-eth-pcs-pma.pdf.

[144] *Virtex-5 FPGA Embedded Tri-Mode Ethernet MAC User Guide*. Xilinx, Feb. 2011.
URL: https://www.xilinx.com/support/documentation/user_guides/ug194.pdf.

[145] *7 Series FPGAs GTX/GTH Transceivers*. Xilinx, Aug. 2018.
URL: https://www.xilinx.com/support/documentation/user_guides/ug476_7Series_Transceivers.pdf.

[146] P. Marciniewski. 'ADC for EMC-Endcap'.
Talk presented at the PANDA Collaboration Meeting, GSI Darmstadt. 2012.

[147] P. Marciniewski. 'A compact size, 64-channel, 80 MSPS, 14-bit dynamic range ADC module for the PANDA Electromagnetic Calorimeter'.
Talk presented at the TWEPP conference, Santa Cruz (CA). 2017. URL: https://indico.cern.ch/event/608587/contributions/2614165/contribution.pdf.

[148] R. P. Sallen and E. L. Key. 'A practical method of designing RC active filters'.
In: *IRE Transactions on Circuit Theory* 2.1 (Mar. 1955), pp. 74–85. ISSN: 0096-2007. DOI: 10.1109/TCT.1955.6500159.

[149] *Datasheet LTM9009-14*. Linear Technologies.

[150] *Datasheet LMK0480x*. Texas Instrument.
URL: http://www.ti.com/lit/ds/symlink/lmk04803.pdf (visited on 12/2014).

[151] D. Buell et al.
'Guest Editors' Introduction: High-Performance Reconfigurable Computing'. In: vol. 40. 3. IEEE Computer Society, 2007. DOI: 10.1109/MC.2007.91.

[152] *7 Series DSP48E1 Slice*. Xilinx, Mar. 2018. URL: https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf.

[153] *7 Series FPGAs Data Sheet: Overview*. Xilinx.
URL: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf (visited on 08/01/2017).

[154] V. M. Placinta and L. N. Cojocariu. 'Radiation Hardness Studies and Evaluation of SRAM-Based FPGAs for High Energy Physics Experiments'.
In: *Topical Workshop on Electronics for Particle Physics*
(Sept. 11, 2017–Sept. 14, 2017). Vol. 313. Santa Cruz, CA: PoS, Mar. 2018, p. 085.
URL: https://pos.sissa.it/313/085/.

[155] *Soft Error Mitigation Controller v4.1*. Xilinx.
URL: https://www.xilinx.com/support/documentation/ip_documentation/sem/v4_1/pg036_sem.pdf (visited on 04/05/2017).

[156] M. Preston et al.
'Measurements and Simulations of Single-Event Upsets in a 28-nm FPGA'.
In: *Topical Workshop on Electronics for Particle Physics*
(Sept. 11, 2017–Sept. 14, 2017). Vol. 313. Santa Cruz, CA: PoS, Mar. 2018, p. 096.
URL: https://pos.sissa.it/313/096/.

[157] *Datasheet LMZ31710*. Texas Instruments.
URL: http://www.ti.com/lit/ds/symlink/lmz31710.pdf (visited on 04/2018).

[158] *Datasheet SY89546u*. Micrel.
URL: http://ww1.microchip.com/downloads/en/DeviceDoc/sy89546u.pdf.

[159] *Kintex-7 FPGAs Data Sheet: DC and AC Switching Characteristics*. Aug. 2017.
URL: https://www.xilinx.com/support/documentation/data_sheets/ds182_Kintex_7_Data_Sheet.pdf.

[160] J. Müllers et al.
'Adaption of an FPGA-based Sampling-ADC for the Crystal Barrel Calorimeter'.
In: *Topical Workshop on Electronics for Particle Physics*
(Sept. 11, 2017–Sept. 14, 2017). Vol. 313. Santa Cruz, CA: PoS, Mar. 2018, p. 052.
URL: https://pos.sissa.it/313/052/.

[161] T. Poller.
*Internal Note: Improvement of the Heat Management of the Crystal Barrel SADC*.
2017.

[162] E. Fix. 'Set-up and Commissioning of the new CBELSA/TAPS Read-out Electronics at the Student Experiment'. Master Thesis (in preparation). Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[163] *Datasheet MAX8556*. Maxim Integrated.
URL: https://datasheets.maximintegrated.com/en/ds/MAX8556-MAX8557.pdf.

[164] T. Poller.
*Internal Note: Simulation of Shaping Parameters for the Analog Input Card*. 2018.

[165] M. P. Biel. 'Entwicklung einer Analysemethode zur Bestimmung und Optimierung der Baseline-Einstellung und Pole-Zero-Kompensation für die Analogstufe der Sampling-ADCs des Crystal Barrel Experimentes'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2018.

[166] M. Usach.
*AN-1121: Replacing Mechanical Potentiometers with Digital Potentiometers*.
Analog Devices. URL: http://www.analog.com/media/en/technical-documentation/application-notes/AN-1121.pdf.

[167] *Datasheet AD5142a*. Analog Devices.
URL: http://www.analog.com/media/en/technical-documentation/data-sheets/AD5122A_5142A.pdf (visited on 06/2017).

[168] *AN 583: Designing Power Isolation Filters with Ferrite Beads for Altera FPGAs.* Altera. URL: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/an/an583.pdf.

[169] *Datasheet UCD90160.* Texas Instruments. URL: http://www.ti.com/lit/ds/slvsac8c/slvsac8c.pdf (visited on 08/2017).

[170] *Fusion Digital Power Designer GUI 7.0 for the UCD90xxx Sequencer.* Texas Instruments. URL: http://www.ti.com/lit/ug/slvub51/slvub51.pdf (visited on 05/2017).

[171] *UCD90SEQ64EVM-650: 64-Pin Sequencer Development Board.* Texas Instruments. URL: http://www.ti.com/lit/ug/slvu423/slvu423.pdf (visited on 12/2012).

[172] *UCD90xxx Sequencer and System Health Controller PMBus Command Reference.* Texas Instruments. URL: http://www.ti.com/lit/ug/slvu352f/slvu352f.pdf (visited on 05/2017).

[173] J. Tucker. *Using a buck converter in an inverting buck-boost topology.* Texas Instruments. URL: http://www.ti.com/lit/an/slyt286/slyt286.pdf (visited on 05/2017).

[174] *Datasheet LMZ34002.* Texas Instruments. URL: http://www.ti.com/lit/ds/snvs989/snvs989.pdf (visited on 07/2013).

[175] *Source-Synchronous Serialization and Deserialization (up to 1050 Mb/s).* Xilinx, Nov. 2013. URL: https://www.xilinx.com/support/documentation/application_notes/xapp1064.pdf.

[176] N. Sawyer. *LVDS Source Synchronous 7:1 Serialization and Deserialization Using Clock Multiplication.* Xilinx, Mar. 2015. URL: https://www.xilinx.com/support/documentation/application_notes/xapp585-lvds-source-synch-serdes-clock-multiplication.pdf.

[177] *FIR Compiler v7.2.* Xilinx, Nov. 2015. URL: https://www.xilinx.com/support/documentation/ip_documentation/fir_compiler/v7_2/pg149-fir-compiler.pdf.

[178] P. Isza. *T-Filter - Free Online FIR Filter Design.* URL: http://t-filter.engineerjs.com/.

[179] *AXI Reference Guide.* Xilinx, Mar. 2011. URL: https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf.

[180] I. T. Union. *ITU-T X.200 - Information Technology - Open Systems Interconnection - Basic Reference Model: The Basic Model.* July 1994. URL: http://handle.itu.int/11.1002/1000/2820.

[181] A. Fiergolski and P. Fall. *1G eth UDP / IP Stack.* OpenCores. URL: https://opencores.org/project/udp_ip_stack (visited on 03/05/2018).

[182] *AXI4-StreamInterconnect v1.1.* Xilinx, Oct. 2017. URL: https://www.xilinx.com/support/documentation/ip_documentation/axis_interconnect/v1_1/pg035_axis_interconnect.pdf.

[183] V. I. Stoica.
'Digital Pulse-Shape Analysis and Controls for Advanced Detector Systems'.
PhD Thesis. Rijksuniversiteit Groningen, 2012.

[184] J. Jungmann. 'A watchdog and radionuclide identification detection system for gamma-ray emitters in aquatic environments'. Master Thesis. Groningen, 2008.

[185] R. Herveille. *IP core for the implementation of an $I^2C$ bus on an FPGA*. OpenCores. URL: `http://opencores.org/project,i2c` (visited on 08/04/2013).

[186] A. Sonnenschein.
'Test eines Detektormoduls der Vorwärtsendkappe des elektromagnetischen Kalorimeters des PANDA-Experimentes mit finaler Sampling-ADC Auslese'.
Bachelor Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2019.

[187] J. Hartmann. *GIT Archive of the `wavezip` compression algorithm (restricted)*. URL: `https://agthoma.hiskp.uni-bonn.de/gitlab/J.Hartmann/wavezip`.

[188] P. Fenwick. *Introduction to Computer Data Representation*. Bentham eBooks, 2014. ISBN: 978-1-60805-883-9.

[189] B. Salisbury. *Personal communication*. 2019.

[190] J. Junkersfeld.
'Kalibration des Crystal-Barrel-ELSA Detektors mit Hilfe der Reaktion $\gamma p \to p\pi^0$'.
Diploma Thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, 2000.

[191] old_timer. *VHDL or Verilog?* URL: `https://electronics.stackexchange.com/questions/16767/vhdl-or-verilog`.

[192] *Vivado Design Suite User Guide - Synthesis*. Xilinx, Apr. 2018. URL: `https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug901-vivado-synthesis.pdf`.

[193] *Vivado Design Suite User Guide - Implementation*. Xilinx, Apr. 2018. URL: `https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug904-vivado-implementation.pdf`.

[194] *7 Series FPGAs Configurable Logic Block*. Xilinx, Sept. 2016. URL: `https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf`.

# List of Figures


1.1. N-Bit-ADC . . . 2
1.2. ADC Lookup Table . . . 3
1.3. ADC Architectures . . . 4
1.4. Comparator and Hysteresis . . . 5
1.5. 3-Bit Flash ADC . . . 6
1.6. Subranging ADC . . . 7
1.7. Missing Code Error in SRAs . . . 8
1.8. Error-corrected Subranging ADC . . . 9
1.9. Pipeline Architecture . . . 9
1.10. Delay of an ADC with Pipeline Architecture . . . 9
1.11. Scheme of the Successive-Approximation ADC . . . 10
1.12. Successive-Approximation Algorithm . . . 11
1.13. Principle of a Σ-Δ ADC . . . 12
1.16. ADC Noise Sources . . . 14
1.17. 1/f Noise . . . 15
1.18. Quantization Noise . . . 16
1.19. SFDR Showing Quantization Noise . . . 17
1.20. Transition Noise and Differential Non-Linearity . . . 18
1.21. ADC Input Referred Noise Histogram . . . 19
1.22. SHA Circuit and Aperture Jitter . . . 20
1.23. SNR Degradation due to Aperture Time Jitter . . . 21
1.24. Aliasing Effects . . . 22

2.1. Running Coupling Constant . . . 24
2.2. Spectrum of the Cat's-Eye Nebula NGC 6543 . . . 25
2.3. Photon Nucleon Cross-Section . . . 26
2.4. Predictions and Measurements of the Nucleon Excitation Spectrum . . . 27
2.5. Predictions of Lattice QCD . . . 28
2.6. Overview of the ELSA Setup . . . 30
2.7. Overview of the Crystal Barrel/TAPS Experiment . . . 31
2.8. The Goniometer . . . 32
2.9. Frozen Spin Target . . . 34
2.10. Crystal Barrel Calorimeter . . . 36
2.11. Overview of the Readout Chain . . . 38
2.12. APD Types . . . 39
2.13. Spectral Noise Density of the APD Preamplifier . . . 40
2.14. Front-end Electronics . . . 41
2.15. Slice Board . . . 41

2.16. CB-TDC Algorithms . . . 44
2.17. Trigger Level . . . 47
2.18. Trigger Efficiency . . . 47
2.19. Trigger System . . . 49
2.20. Crystal Barrel DAQ Network . . . 51
2.21. Example $I^2C$ Bus . . . 53
2.22. Example $I^2C$ Transmission . . . 53

3.1. PANDA EMC . . . 60
3.2. PANDA SADC History . . . 61

4.1. 16-Channel PANDA-SADC . . . 63
4.2. First Recorded Pulse . . . 65
4.3. 16-Channel PANDA-SADC with SP605 Evaluation Board . . . 66
4.4. Ethernet Type 2 Frame . . . 66
4.5. Feature Packet . . . 67
4.6. Sample Packet . . . 68

5.1. Development of the 64-Channel SADC Prototypes . . . 69
5.2. Data Flow . . . 71
5.3. 64-Channel SADC Topology . . . 71
5.4. SADC Input Buffers . . . 72
5.5. LTM9009 Functional Block Diagram . . . 74
5.6. LTM9009 Resolution . . . 76
5.7. LTM9009 DNL/INL . . . 76
5.8. Clocking Scheme of the CB-SADC . . . 77
5.9. LMK0480x PLL2 Single PLL . . . 78
5.10. Clock Probe Points on the CB-SADC. . . . 79
5.11. Comparison of the Clock Signal and Jitter . . . 80
5.12. Jitter Measurement of the FPGAs Output Clocks . . . 81
5.13. Comparison of CPU, GPU, FPGA, ASIC . . . 81
5.14. CLB . . . 82
5.15. FPGA Inner Life . . . 83
5.16. DSP48E1 . . . 83
5.17. Power Delivery Network . . . 87
5.18. Polygon Technique . . . 88
5.19. PDN Analyzer . . . 89
5.20. PDN Analyzer Result . . . 90
5.21. Early Ideas for the SADC Adaption . . . 91
5.22. Dip-switch on CB-SADC. . . . 93
5.23. Trigger Signal Routing . . . 94
5.24. Slowcontrol Connections on the CB-SADC . . . 96
5.25. Infrared Pseudocolor Image . . . 97
5.26. Heatsink Revisions . . . 98
5.27. Temperature Improvement with Conductive Pad . . . 98
5.28. Comparison of PANDA-SADC and CB-SADC . . . 99

6.1. Analog Input Card . . . 104
6.2. Shaping Circuit on the Analog Input Card . . . 104
6.3. Bode Plots of the Shaping . . . 105
6.4. Effect of the Band-Pass Shaping . . . 106
6.5. Differential Transmission . . . 107
6.6. Baseline Adjustment . . . 108
6.7. Basic Pole-Zero Cancellation . . . 109
6.8. AD5142A Block Diagrams . . . 111
6.9. Pole-Zero Cancellation Adjustment . . . 112
6.10. Pole-Zero Cancellation Circuit . . . 112
6.11. Effect of the Pole-Zero Cancellation . . . 113
6.12. Investigation of Pole-Zero Cancellation . . . 114
6.13. CB-SADC Testboard . . . 116
6.14. SADC Baseline Analyzed with the Testboard . . . 117
6.15. Pole-Zero Cancellation Analysis with the Testboard . . . 117
6.16. Buck Topology . . . 119
6.17. Prototype of the Two-Channel Power Supply . . . 120
6.18. Measurement of the Noise of the Power Supply Prototype . . . 121
6.19. Prototype of the Four-Channel Power Supply . . . 122
6.20. Functional Block Diagram of the UCD90160 . . . 123
6.21. Test of the Four-Channel Prototype under Load . . . 124
6.22. Recommended Switching Frequencies for the LMZ31710 . . . 125
6.23. Temperature of the Power Supply depending on the Switching Frequency . . . 125
6.24. Final Revision of the Four-Channel Power Supply . . . 126
6.25. Crate Controller Scheme . . . 127
6.26. Concept and Prototype of the SADC Controller . . . 128
6.27. Extension Card . . . 129
6.28. JTAG Implementation on the Backplane . . . 130
6.29. Prototype 1.0 and 1.1 of the Backplane . . . 131
6.30. Concept of the Future CB-SADC Crate Controller . . . 133
6.31. I2C over CAT5 schematic . . . 134
6.32. Slowcontrol Extension for ELB-VME-VFB . . . 134

7.1. FPGA Design Flow . . . 135
7.2. Firmware Structure . . . 136
7.3. Firmware Reset State Machine . . . 138
7.4. ADC Configuration State Machine . . . 140
7.5. Two-Lane Serialization of the LTM9009-14 . . . 142
7.6. Stable Sampling Region . . . 143
7.7. Phase Calibration State Machine . . . 145
7.9. Randomization Circuit . . . 146
7.8. Phase Calibration and Bitslip Simulation . . . 147
7.10. FIR Filter . . . 149
7.11. Transfer Function of a Half-band Filter . . . 150
7.12. FIR Filter Low-Pass . . . 151
7.13. Buffers and SMA Filter . . . 152

7.14. Network Structure . . . 153
7.15. AXI4-Stream Protocol . . . 154
7.16. PCS/PMA Scheme for Fiber/Copper Connection . . . 155
7.17. CPLL Frequency . . . 156
7.18. Custom Reliability Protocol . . . 160
7.19. Baseline Extraction . . . 163
7.20. Optimal Integration Window . . . 163
7.21. Constant Fraction Discriminator . . . 165
7.22. Pile-Up Definition . . . 167
7.23. Pile-Up Simulation . . . 168
7.24. Integrated Logic Analyzer of the CB-SADC Backplane . . . 175
7.25. Identification of Trigger Crosstalk . . . 175
7.26. trigger/sync Master Logic Implementation . . . 176

8.1. QT Debugging Tool Single-Channel View . . . 178
8.2. QT Debugging Tool All Channel View . . . 178
8.3. CLI Tool `sadc-receiver` . . . 181
8.4. CB-SADC Slowcontrol Standalone Readout Tool . . . 184

9.1. CB-SADC Rack . . . 186
9.2. VME Crate in the CB-SADC Rack . . . 190
9.3. NIM modules in the CB-SADC Rack . . . 191
9.4. Typical SADC Integral Histograms . . . 193
9.5. Comparison of QDC and CB-SADC Resolution . . . 193
9.6. Minimum Ionizing Particles . . . 195
9.7. Comparison of QDC and CB-SADC Resolution with MIPs . . . 195
9.8. Minimum Ionizing Particle Fits . . . 196
9.9. Investigation of the Energy Sum Signal . . . 197
9.10. Distribution of the Baseline Widths . . . 199
9.11. Histrogram of the Baseline Widths . . . 199
9.12. Analysis of the Noise . . . 200
9.13. Waveforms with 312 kHz Noise Signature . . . 202
9.14. Analysis of the 312 kHz Noise . . . 202
9.15. Pile-Up Detection Matrix . . . 203
9.16. Pile-Up Filter . . . 204
9.17. Pile-Up Rate . . . 205
9.18. Pile-Up Recovery . . . 205
9.19. Invariant Mass Spectra . . . 207
9.20. Analysis of the Pion Decay Width . . . 208
9.21. Energy Dependent Time Resultion of the CB-SADC Timestamp . . . 211
9.22. CB-SADC Timestamp Fit . . . 211
9.23. CB-SADC vs. TDC Timestamp . . . 213
9.24. Burst Performance . . . 215
9.25. Busy Period on Backplane Debug Interface . . . 216
9.26. Disk Load Analysis . . . 217
9.27. Network Performance Analysis . . . 218

9.28. CPU/Memory Load Analysis . . . . . 218

A.1. CB-SADC Rack in the Experimental Hall . . . . . 221
A.2. CB-SADCs in the Experimental Hall . . . . . 222
A.3. CB-SADCs in the Experimental Hall . . . . . 223
A.4. CB-SADCs in the Experimental Hall . . . . . 223
A.5. `ADC-64K-CB v1.0` with Adapter for Connection of Signals with Pin Header . . 224
A.6. Assembled `ADC-64K-CB v1.1` . . . . . 225
A.7. Test Adapter for the CB-SADC Power Supply . . . . . 226
A.8. Test Adapter for the CB-SADC Extension Card . . . . . 226
A.9. Testboard . . . . . 227

B.1. JTAG Implementation on the Backplane . . . . . 229
B.2. Example for JTAG Debugging . . . . . 232
B.3. Example for VHDL Simulation . . . . . 233

D.1. Waveforms of Clock Signal . . . . . 239
D.2. Clipped Pulse . . . . . 240
D.3. Typical Waveforms recorded at the Crystal Barrel Calorimeter . . . . . 240
D.4. Typical Pile-Up Waveforms recorded at the Crystal Barrel Calorimeter . . . . 241
D.5. SADC Online Monitor . . . . . 242
D.6. CB-SADC vs. TDC Time Stamp (Log) . . . . . 243
D.7. CB-SADC vs. TDC Time Stamp (Ring 5-24) . . . . . 243

F.1. Schematics of the Noise Density Simulation . . . . . 259
F.2. LTM9009-14 Block Diagram . . . . . 260
F.3. LTM9009-14 Sample And Hold Circuit . . . . . 261
F.4. LMK0480x Functional Block Diagram . . . . . 261
F.5. LMK0480x Configuration . . . . . 262
F.6. SLICEM Primitive . . . . . 263
F.7. FPGA Power Estimation Summary . . . . . 264
F.8. FPGA Power Estimation Details . . . . . 264

# Glossary

**beam time** A beam time refers to the operation of the CBELSA/TAPS experiment with extracted beam from ELSA hitting the photoproduction target, and the photons hitting the reaction target. The data is recorded with the experiment's data acquisition system (DAQ). 33, 62, 103, 114, 127, 129, 130, 140, 151, 165, 171, 183–185, 187, 194, 203, 206, 209, 212, 213, 216, 219

**daemon** A deamon refers to a process which runs in the background on a LINUX computer (analog to a *service* on a WINDOWS machine). 48, 50, 183, 184

**GPIO** General Purpose Input Output, refers to a connection to an FPGA or micro-controller which can be used as input or output (usually a tri-state buffer is used). 123, 129

**IP Core** Intellectual Property Core, a code snippet (which can be open- or closed-source) for FPGA design. Sometimes just referred to as *Core*. 64, 156, 158, 159

**JTAG** Joint Test Action Group, refers to an interface for in-design programming and debugging according to the IEEE standard 1149.1-1990. 64, 79, 84, 85, 92, 127–133, 140, 174, 186, 190, 223, 226, 232, 233, 283

**MMCM** The Mixed-Mode Clock Manager is a XILINX KINTEX-7 primitive. It serves as frequency synthesizer for a wide range of frequencies, a jitter filters for either external or internal clocks, and can deskew clocks. 138

**peaking time** Peaking time is a measure to characterize a filter. It is the time after which a signal reaches its maximum value after passing the filter. Usually for a filter the peaking time for a step response as well as the response to the actual signal waveform is characterized. 105

**pedestal** Pedestal (also called baseline) refers to the voltage present at the digitizer in the case that there are no signals present from events in the detector. Intuitively this potential is the ground level, but to allow headroom for the signal to undershoot, it is commonly shifted. 44, 45, 55

**run** In terms of the CBELSA/TAPS experiment, a run is a set of continuous measurements taken with the DAQ during a beam time. Apart from data taking with a *production trigger*, some other kinds of runs are mentioned throughout the thesis. A **Clock Run** refers to a random trigger (uncorrelated to physical processes), usually by a clock signal with several kHz. A **Cosmic Run** refers to recording of energy deposits by cosmic background radiation, which can be useful for a calibration, but are also a means of

investigating the readout when no electron beam is supplied by the accelerator. A **Lightpulser Run** is used for calibration of the low and high range of the QDC and also a means of readout investigation. 55, 113, 117, 182, 204

**TTL** Transistor-Transistor-Level, a definition for the transmission of digital signals with positive polarity. Usual TTL levels are 5 V, 3.3 V, 2.5 V, 1.8 V, the latter three are commonly referred to as LVTTL (low-voltage). TTL is often used synonymously for (LV-)CMOS, which is the equivalent definition for field effect transistors. 129, 191

**via** A via is a metalized hole that is drilled in a printed circuit board to connect signals on different copper layers. 79, 88

**VME** The VMEbus (Versa Module Europa bus) is a bus system with 32 bit wide address and data bus. A *VME Crate* consists of a VME CPU and a number of devices which it controls. 43, 45, 48–51, 57, 59, 91, 127, 132, 176, 186, 190, 221

# Acronyms

**Σ-Δ ADC** Sigma-Delta ADC. 11, 12, 279

**ADC** Analog-to-Digital Converter. 1–21, 45, 50, 51, 58, 63, 70–72, 74–77, 79, 80, 86, 88, 92, 95, 97, 100, 101, 107, 118, 121–123, 136–139, 141, 144, 146, 148, 158, 192, 194, 198, 199, 232, 235–237, 239, 240, 279, 287, 289

**APD** Avalanche-Photodiode. 2, 35, 37–41, 44, 47, 52, 53, 56, 58, 60, 108, 113, 185, 191, 194, 196–198, 201, 206

**ASIC** Application-Specific Integrated Circuit. 2, 137

**ATCA** Advanced Telecommunications Computing Architecture. 60, 91

**BEI** Basic Element. 231, 233

**CAD** Computer-aided Design. 91, 97

**CFD** Constant Fraction Discriminator. 164, 210–213

**CLB** Configurable Logic Block. 82, 83, 151, 231, 263

**CMOS** Complementary Metal Oxide Semiconductor. 3, 10, 11, 110

**DAC** Digital-to-Analog Converter. 7, 8, 10, 12, 104, 107, 108, 116

**DAQ** Data Acquisition. 50, 62, 127, 132, 160, 174–177, 180, 182, 185, 189, 206, 214, 216, 217, 286

**DC** Direct Current. 75, 86, 92

**DDR** Double Data Rate. 141, 142

**DNL** Differential Non-Linearity. 18, 19, 76

**DSP** Digital Signal Processing. 63, 77, 82, 83, 103, 138, 151, 152, 168, 239

**ECL** Emitter-coupled logic. 48, 49

**ELSA** Electron Strechter Accelerator. 29–31, 50, 57

**EMC** Electromagnetic Calorimeter. 60

**FACE** Fast Cluster Encoder. 46, 47

**FADC** Flash Analog-to-Digital Converter. 3, 5–7, 10, 18, 45, 57

**FIFO** First In, First Out. 152, 159, 160

**FIR** Finite Impulse Response. 149–152

**FPGA** Field Programmable Gate Array. 2, 33, 34, 43, 48, 53, 55–58, 61–66, 69–71, 74, 77–90, 92–97, 99–101, 103, 106, 122, 128–130, 133–139, 141–144, 147–149, 151–153, 155–157, 167, 175, 176, 181–183, 190–192, 195, 196, 203, 205, 209, 212, 215, 217, 219, 220, 231–233, 237, 240, 241, 263, 264, 280, 285

**FSR** Full Scale Range. 2, 3, 5

**FWHM** Full Width Half Maximum. 44, 75

**GUI** Graphical User Interface. 50, 132, 179

**I²C** Inter-IC Communication. 52, 53

**IC** Integrated Circuit. 74, 78, 87–89, 97, 101, 119, 120, 126, 128, 136, 137, 139, 140, 142, 164, 174, 192, 224

**ILA** Integrated Logic Analyzer. 64, 174, 190, 232, 233

**INL** Integral Non-Linearity. 76

**IOB** Input Ouput Block. 82, 83

**IP** Internet Protocol. 156–158, 172, 177, 181, 182, 235

**LED** Light Emitting Diode. 52, 88, 95, 97, 100, 113, 122, 123, 133, 134, 191, 224

**LEvB** Local Event Builder. 38, 45, 48–51, 153, 156–160, 162, 169–171, 177, 180, 182, 183, 188, 189, 206, 214, 215, 217, 218, 235, 237, 257

**LINAC** Linear Accelerator. 29

**LSB** Least Significant Bit. 2, 7, 59, 146

**LVDS** Low Voltage Differential Signaling. 139, 141

**LVPECL** Low Voltage Positive Emitter-coupled Logic. 48, 49, 129

**MAC** Media Access Control. 153

**MIP** Minimum Ionizing Particle. 194–196

**MSB** Most Significant Bit. 2, 4, 6, 7, 10, 146

**MTU** Maximum Transmission Unit. 158, 159, 172

**NIM** Nuclear Instrumentation Module. 42, 56, 59, 86, 87, 91, 92, 97–99, 110, 118, 122, 124–126, 128, 130, 132, 133, 176, 184, 186, 189–191, 194, 206, 219, 221–225, 240

**op-amp** Operational Amplifier. 1, 40, 42, 70, 72, 73, 86, 88, 97, 104, 107, 192

**PANDA** Proton Antiproton Annihilation in Darmstadt. 59–61, 64, 70, 179

**PCB** Printed Circuit Board. 72, 74, 77, 80, 86–88, 95, 97, 98, 100, 103, 115, 118, 119, 128, 130, 143, 175, 224

**PIN** Positive-Intrinsic-Negative. 37, 58

**PLL** Phase-Locked Loop. 20, 70, 78, 79, 85, 88, 92, 95–97, 100, 127, 129, 137, 138, 142, 155, 184, 239, 261, 262

**PMBus** Power Management Bus. 123, 130

**PMT** Photo-Multiplier Tube. 33–36, 46, 57, 59

**PWM** Pulse-Width Modulation. 119–121

**QCD** Quantum Chromodynamics. 24, 28

**QDC** Charge-to-Digital Converter. 38, 42, 44–47, 52, 55–59, 84, 162, 163, 183, 185, 190–193, 195, 196, 198, 200, 201, 206–209, 211, 217, 223, 286

**RAM** Random Access Memory. 84, 85, 89, 152, 171, 172, 187, 232

**RMS** Root Mean Square. 15, 16, 19, 117, 198–201

**ROM** Read-only Memory. 84, 85, 130, 131, 231

**SADC** Sampling Analog-to-Digital Converter. 11, 28, 37, 40, 42, 45, 48, 50, 53, 55–73, 77–79, 81, 82, 84–101, 103–105, 107, 110, 114–116, 118–142, 144, 145, 148, 150, 151, 153–163, 165–167, 169–198, 200–226, 229, 231, 235, 237, 240, 242, 243, 257, 258, 280–283

**SAR** Successive-Approximation Register. 10, 11

**SEU** Single Event Upset. 85

**SFP** Small Form-factor Pluggable. 64, 71, 77, 92, 153, 155, 188

**SHA** Sample and Hold Amplifier. 6–8, 10, 13, 19, 20, 74

**SiPM** Silicon Photo Multiplier. 37

**SNR** Signal-to-Noise ratio. 11, 14, 16–18, 20, 21, 37, 39, 58, 72, 75, 76, 103–105, 148, 150, 163, 167, 192, 210

**SPI** Serial Peripheral Interface. 95, 139

**SRA** Sub-Ranging ADC. 6–8, 45

**T/H** Track-and-Hold. 8, 9, 19

**TAPS** Two Arm Photon Spectrometer. 36, 47, 101

**TCP** Transmission Control Protocol. 128, 157

**TDC** Time-to-Digital Converter. 33, 34, 42, 43, 50, 51, 56, 58, 197, 210, 212, 213, 223, 243

**TDR** Technical Design Report. 60, 70

**UDP** User Data Protocol. 116, 136, 138, 156–159, 172, 177, 179, 181, 182, 235–237, 252

**VCO** Voltage-Controlled Oscillator. 77, 78, 155, 156

**VHDL** Very High Speed Integrated Circuit Hardware Description Language. 135, 139, 141, 166, 229

**VIO** Virtual Input/Output. 140, 174, 190, 232

# Danksagung

Was lange währt, wird endlich gut. In der Hoffnung niemanden vergessen zu haben, möchte ich mich bei einigen Personen für das Gelingen meiner Arbeit bedanken.

Zunächst möchte ich mich bei Frau Prof. Dr. Ulrike Thoma für das Angebot und die Betreuung der Doktorarbeit bedanken, vor allem dafür, dass ich gerade bei der Entwicklung der Hardware viel Freiheit genießen durfte und Gelegenheit hatte, interessante und neue Wege zu gehen.
Herrn Prof. Dr. Bernhard Ketzer, Herrn PD Dr. Bastian Kubis, sowie Herrn Prof. Dr. Andreas Weber danke ich für die Teilnahme an der Promotionskommission.
Bei Dr. Christoph Schmidt bedanke ich mich für die intensive Unterstützung vor allem am Anfang des Projekts, was mir den Einstieg in das Thema sehr erleichtert hat, sowie für die kontinuierliche Mitarbeit am CB-SADC Projekt. Gerade in der Softwareentwicklung wurde das Projekt so maßgeblich beschleunigt, und der CB-SADC konnte zeitnah mit der Datenakquisition des Crystal Barrel Experiments verbunden werden.
Dr. Pawel Marciniewski hat den PANDA-SADC entworfen, auf dessen Design der CB-SADC basiert. Nicht nur wäre ohne ihn das Projekt gar nicht erst entstanden, er hat mich vor allem am Anfang des Projektes und bei der Inbetriebnahme des ersten Prototypen unterstützt. Hierfür bedanke ich mich ganz herzlich!
Während meiner Promotion durfte ich einige Studenten anleiten, die mit ihren Arbeiten maßgeblich zum CB-SADC Projekt beigetragen haben. Der Wissenstransfer geschah dabei nicht immer in der klassisch zu erwartenden Richtung. Ich bedanke mich bei Mel Biel, Jens Knaust, Timo Poller, Jan Schultes, Georg Urff, und Ya-Chun Wang.
Ich danke auch Ben Salisbury und Nils Stausberg für das Bereitstellen der ersten Analyseergebnisse aus den CB-SADC Daten, ohne die der Nachweis der guten Performance des CB-SADC nicht möglich gewesen wäre. Dr. Jan Hartmann danke ich für seine zahlreichen Beiträge zur Backend Software des CB-SADC sowie viele hilfreiche Diskussionen. Dr. Philipp Hoffmeister, Dr. Christian Honisch, Peter Klassen und Philipp Mahlberg danke ich für vielfältige Hilfe bei Programmier- und Elektronikfragen. Dr. Marcus Grüner, Dr. Michael Lang, Dr. Martin Urban und Dr. Christoph Wendel danke ich für die unzähligen selbstlosen Gefälligkeiten und Hilfestellungen.
Meinen langjährigen Bürokollegen Georg, Dima, Janina, Matthias, Merlin und Sonja danke ich für die schöne Zeit im Büro, und der ganzen Crystal Barrel Kollaboration für die tolle Zusammenarbeit seit 2011.
Den Mitarbeitern der Feinmechanikwerkstatt und der Elektronikwerkstatt, sowie dem Hausmeister Herrn Leiendecker danke ich für die Beratung und Hilfe bei vielen Projekten.

Für das Korrekturlesen der Doktorarbeit bedanke ich mich ganz besonders bei Ya-Chun Wang. Zuletzt danke ich meiner Familie für die Unterstützung und den Rückhalt, den ich immer finden konnte, wenn ich ihn brauchte.