FPGA based readout of a silicon PIN detector

by

Eino J. Oltedal

THESIS
for the degree of
MASTER OF SCIENCE

(Master in Electronics)

Faculty of Mathematics and Natural Sciences
University of Oslo

June 2015

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Contents

List of Figures

List of Tables

Glossary

^{241}Am Americium-241, Alpha radiation source commonly used for calibrating radiation sources. 3

^{252}Cf Californium-252, A widely used radiation source that undergoes spontaneous fission. 3, 6

ADC Analog to digital converter. ii, 2, 12, 13, 19, 22, 29

ALICE A Large Ion Collider Experiment. 12

ALTRO ALICE TPC Read Out. iv, 11–13, 23–25

ARM Processor architecture based on a reduced instruction set computing architecture. 15

CERN European Organization for Nuclear Research. 4, 12

CMOS Complementary metal–oxide–semiconductor. 19

DPP Digital Pulse Processing. 13

DSP Digital Signal Processing. ii, 2, 9, 11, 13, 15, 21, 22, 29–31, 45

DUT Device Under Test.. 31, 34

eV Electron volt 1 eV $= 1.602 \times 10^{-19}$ C. 4, 35

FIFO First In First Out. 16, 55

FPGA Field Programmable Array. ii, 3, 15, 19, 21, 22

FWHM Full Width at Half Maximum. Difference between the two values of the independent variable where the dependent variable is at half of its maximum.. v, 34–37, 39, 46, 50, 53, 55

GeV Giga electron volt (10^9). 4

GPIO General-purpose input/output. 19, 20

HPS Hard Processor System. A processor implemented in CMOS as opposed to soft processors which are implemented in FPGA fabric.. 28

I2C A multi-master multi-slave, single ended serial bus. ii, 19, 20

IC Integrated Circuit. 15

kBq Kilo Becquerel. One Becquerel is the activity of a radioactive material which undergoes one nuclear decay per second.. 44

keV Kilo electron volt (10^3). 4, 5

LHC Large Hadron Collider. 12

LSB Least Significant Bit. 18

LTspice Linear Technology Simulation Program with Integrated Circuit Emphasis. 14

LVDS Low-voltage differential signaling. ii, 19, 20

MeV Mega electron volt (10^6). 4, 6

MSB Most Significant Bit. 17

MSPS Mega Samples Per Second. 12, 13

NFS Network File System, a distributed file system protocol.. 29

OOP Object-Oriented Programming.. 33

PCB Printed Circuit Board. ii, 2, 19

PIN diode A diode with a wide undoped semiconductor region between the p- and n- type semiconductor. v, 2, 9, 45, 46

scp Secure copy, a remote file copy program.. 28

SD-card Secure Digital nonvolatile solid state memory card.. 28

SINTEF An independant Norwegian reasearch organization. 2, 9

TB Test Bench.. 34

TeV Tera electron volt ($10^1$2). 4

TID Total Ionizing Dose. 19

VHDL VHSIC Hardware Description Language. 2, 30, 31, 55

Abstract

In this master thesis digital readout systems for radiation detection and measurement has been studied and designed. The project was split in two where one student developed analog front end electronics, and the other the digital readout and processing system. This thesis focuses on the development of the digital signal processing part. Traditionally a detector readout system is large and not very user friendly. This project aims to produce a detector setup that is both portable and user friendly. The resulting system was calibrated using a test capacitor and a signal generator, characterized using two alpha sources (californium-252 and americium-241), as well as verified through simulation.

Chapter 1

Introduction

1.1 Motivation

Modern radiation detectors such as ionization chambers, scintillators with photomultiplier tubes and semiconductor detectors all have in common that they produce an electrical signal as output. Analysing the output of the detectors therefore requires some sort of signal conditioning front end electronics that amplifies the signal and digitises it in order to enable digital analysis of the

Figure 1.1 shows the simple block diagram of one such system. It has been divided into two parts, a digital and an analog. This thesis will focus on the development and testing of the digital part of the readout system, starting at the output of the ADC and ending in data on the Users laptop. The front end electronics were designed in parallel with this project and can be read about in detail in the thesis of Maris Tali [7].

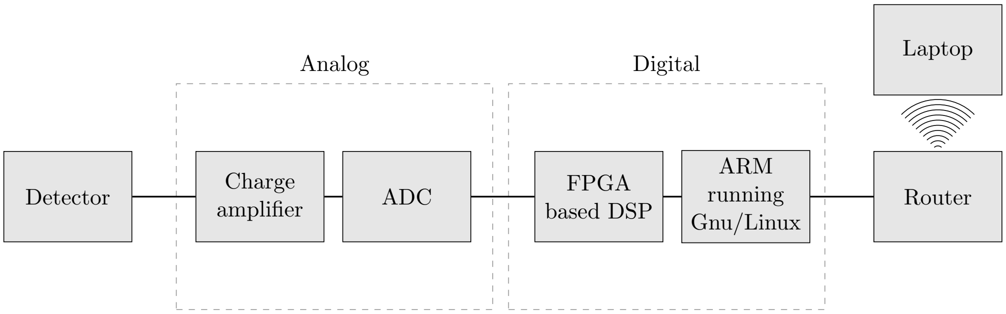


Figure 1.1: Block diagram showing an example setup of readout electronics used together with a radiation detector. The different modules are conveniently divided into an analog and a digital part.

An electronics readout system can often be big, cumbersome, expensive and not very easy to scale or to add features. The main goal of this project is to create a portable, cheap, scalable system that is cheap and easy to modify and that can be used by students in a lab setting learning about detector readout systems or brought of site to characterize radiation sources. Semiconductor detectors have superior energy resolution and smaller size as compared to other commonly used radiation detectors such as scintillators and ionization chambers [3]. This makes them perfect for use in a portable system. The detector used in this thesis is a single pad silicon PIN diode detector from SINTEF A common application of radiation detectors is particle spectroscopy. Semiconductor detectors produce a charge pulse when charged particles interact it, the pulse height can be used to identify what particle interacted with the detector, a goal of the project is thus to successfully characterize an alpha emitter.

1.2 Goal

The thesis has a couple of goals which were used as guidelines when making design choices during the development of the project. The main goal is to develop a compact digital readout system that is capable of characterizing an alpha emitter.

- Develop a way of storing data that is output by the ADC
- Research what DSP modules are commonly used in similar systems and make a decision on which to implement, and implement them.
- Find out how to send the data from the FPGA to a users laptop
- Decide on a protocol for communicating with hardware on the PCBs design by Maris Tali [7]
- Write testbenches for simulating/verifying the VHDL modules
- Characterize the system using alpha sources Americium-241 and Californium-252.

1.3 Structure of thesis

This thesis has been structured to look similar to other scientific master theses. Below follows a short description of each chapters contents.

Chapter 1 presents the motivation and background of the thesis, and lists the goals of the thesis.

Chapter 2 outline the theory behind radiation detection and measurement by giving examples of ionizing radiation, detectors, and signal processing involved in radiation detection.

Chapter 3 describes in detail the system that was developed to solve the goals of this thesis.

Chapter 4 explains how the system was tested in order to verify that it functioned as expected. In addition the calibration procedure is presented in this chapter.

Chapter 5 details the experimental setups used when the system was tested using radiation sources $^{241}\mathbf{Am}$ and $^{252}\mathbf{Cf}$.

Chapter 6 contains the results from the experiments detailed in chapter 5.

Chapter 7 discusses the results and concludes the thesis.

Appendix A The final resource usage of the FPGA.

Chapter 2

Theory

2.1 Radiation

Different types of radiation are often classified as being either non-ionizing or ionizing. Ionizing radiation is radiation that has enough energy to completely remove an electron from an atom, thus ionizing it. Ionizing radiation can be both electromagnetic such as γ-radiation, or energetic particles such as α- and β particles. There are also neutrons which are not directly ionising but produce secondary ionizing radiation when colliding with matter. [8]. Examples of non-ionizing radiation are radio waves, microwaves and visible light.

In this thesis we are only concerned with the detection of ionizing radiation. The types of of ionizing radiation already mentioned can be further categorized into two subgroups, charged particulate radiation and uncharged radiation. See table 2.1 for examples of the two different types. [3] In the following sections the sources of these types of ionizing radiation will be discussed as well as their interaction with matter.

Charged particulate	Uncharged
Fast electrons	Electromagnetic radiation
Heavy Charged particles	Neutrons

Table 2.1: The two major types of ionizing radiation, and examples. β-particles are fast electrons, α-particles fall under heavy charged particles and γ-radiation is high energy electromagnetic radiation.

2.1.1 Energy

The unit of measurement used for radiation energy is the electron volt, eV. The energies involved in ionizing radiation often range in the thousands or millions, or at large accelerator facilities like CERN billions or trillions, of electron volts. Therefore the multiples kilo electron volt keV, mega electron volt MeV, giga electron volt GeV and tera electron volt TeV are more commonly used. Electron volts are handy unit when dealing with particle radiation becuase you can easily obtain the energy gained by a particle from an electric field by multiplying the electronic charge with the potential difference. For example an alpha particle with electronic charge +2 accelerated by a potential difference of 2000 V will gain an energy of 4keV. The electron volt is obtained experimentally and expressed in relation to the SI unit Joule (J) as

$$1.602 \times 10^{-19}$$

2.2 Charged Particles

The two main types charged particle radiation are fast electrons an heavy charged particles as shown in table 2.1. Fast electrons includes positrons in addition to negatively charged electrons. Alpha particles, heavier fission fragments and proton beams from particle accelerators are examples of heavy charged particle radiation.

2.2.1 Beta particles

The most common source of fast electrons is a radio isotope that undergoes negative beta decay Schematically the process is written

$$ {}^{A}_{Z}\mathrm{X} \rightarrow {}_{Z+1}{}^{A}\mathrm{Y} + \beta^{-} + \bar{v} \tag{2.1}$$

X and Y are the initial and final nuclear species and $\bar{v}$ is an anti neutrino. There is also positive beta decay which involves the emission of a positive beta particle, a fast positron, and a neutrino. Neutrinos and anti neutrinos interact primarily through the weak force and are detected by weak interactions[9], and require massive detectors which are buried deep under ground to shield them from cosmic rays and other background radiation [10] [11]. Therefore, for our purposes, they are undetectable. The recoil nucleus Y's energy is below the ionization threshold and is therefore also not detected with conventional means. This leaves the beta particle as the signification contributor to ionizing radiation from beta decay [3]. The beta particle resulting from beta decay carries away some or all of the energy involved in the mass change as kinetic energy. This means that the energy of the particle ranges from zero up to the maximum energy represented by mass loss [8]. Beta emitters are produced through neutron bombardment of stable materials, and species with different half lives can be obtained [3] Particle accelerators can also be used to produce fast electrons.

2.2.2 Alpha particles

Alpha particles are high energy ^{4}He particles, and as opposed to beta particles are emitted at a more or less discrete energy level. Alpha decay can be written schematically as shown in equation 2.2 where X is the initial nuclide, Y is the final nuclide.

$$ {}^{A}_{Z}\mathrm{X} \rightarrow {}^{A-4}_{Z-2}\mathrm{Y} + {}^{4}_{2}\alpha \tag{2.2}$$

The probability of decay is governed by the *barrier penetration* mechanism, which is described in most texts on on nuclear physics but is outside the scope of this thesis. The barrier penetration probability becomes very low for energies below 4 MeV and the half-life of the isotope is very large. Beyond about 6.5MeV the half-life can be expected to be less than a few days, making the source of limited utility. On the other hand this strong correlation between alpha particle energy and half-life is the reason most Alpha particle energies are limited to between about 4 and 6 MeV [3].

There exists a large amount of alpha source isotopes, [3] lists 19 and americium-241 as probably the most common. Table 2.2 lists one alpha source, americium-241, because it was used when testing the completed system. Figure 2.1, taken from [1], shows the upper portion of the ^{241}Am alpha spectra recorded by a high-resolution surface barrier detector. Note there are five tops showing that the alpha particles from americium-241 are mostly emitted at those five discrete energy levels.

Source	Half-Life	Alpha Particle Kinetic Energy in MeV
^{241}Am	433 y	5.48574 ± 0.00012 5.44298 ± 0.00013

Table 2.2: Americium alpha Source data, from [3]

2.2.3 Spontaneous Fission

The fission process is the only spontaneous source for charged particle radiation with masses greater than that of alpha particles. For this reason sources that undergo spontaneous fission are widely used in th calibration and testing of detectors intended for general application to heavy ion measurements. Spontaneous fission is when transuranic elements break up with the production of lighter elements called fission products, and neutrons. An example is the spontaneous fission of californium-252 into xenon, ruthenium, and four neutrons shown in equation 2.3 [8].

$$^{252}_{98}\mathbf{252Cf} \rightarrow {}^{140}_{54}\mathrm{Xe} + {}^{108}_{44}\mathrm{Ru} + 4\mathrm{n} \tag{2.3}$$

Spontaneous fission is only a significant process for transuranic elements because of the large potential barrier that must be overcome in the distortion of the nucleus from its original near spherical shape [3]. Californium-252 is mentioned here because it is, according to [3], the most widely used source that undergoes spontaneous fission, but also because it was used when testing the completed system. If spontaneous fission was the only decay process of californium-252 it would have a half-life of 85 years. However as with most transuranic elements, ^{252}Cf undergoes alpha decay. Because of the alpha decay its actual half-life is 2.65 years, and a 1 μg sample of ^{252}Cf will emit 1.92×10^7 alpha particles and undergo 6.14×10^5 spontaneous fissions per second [3].

The two fission products that are produced in the decay shown in figure 2.3 are emitted in opposite directions because of conservation of momentum. In a typical spontaneous fission source the material that undergoes fission is

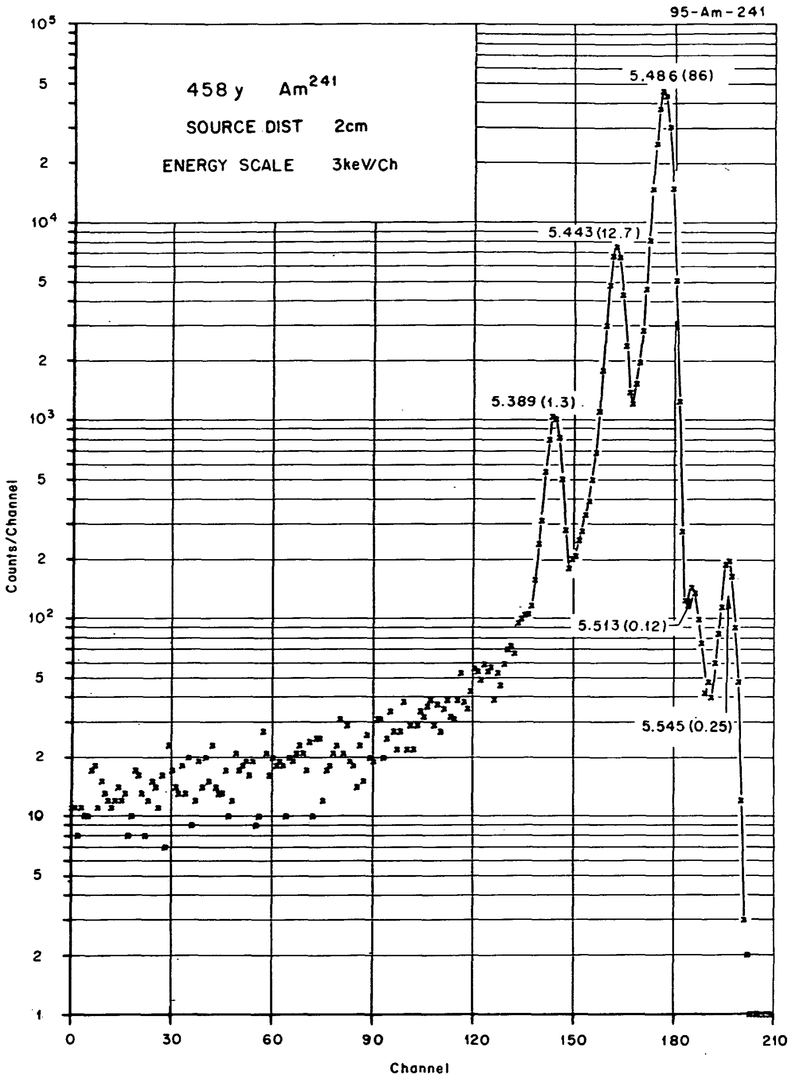


Figure 2.1: Upper portion of the ^{241}**Am** alpha spectrum as recorded by a high-resolution surface barrier detector. From [1].

usually deposited thinly on a metal backing [3]. This means that only one of the two fission products will escape the source and thus be detected, while the other product will be absorbed by the backing. Therefore the kinetic energy distribution of ^{252}Cf will have two peaks, one corresponding to heavy fragments and the other to light fragments, this is demonstrated in figure 2.2 taken from [2].

2.2.4 Accelerated Particles

Charged particles can also be accelerated up to energies where they become ionizion. There are several types of accelerating machines, they involve either positively or negatively charged particles being injected into the machine and accelerated by in electric and magnetic fields. The particles are either acceler-

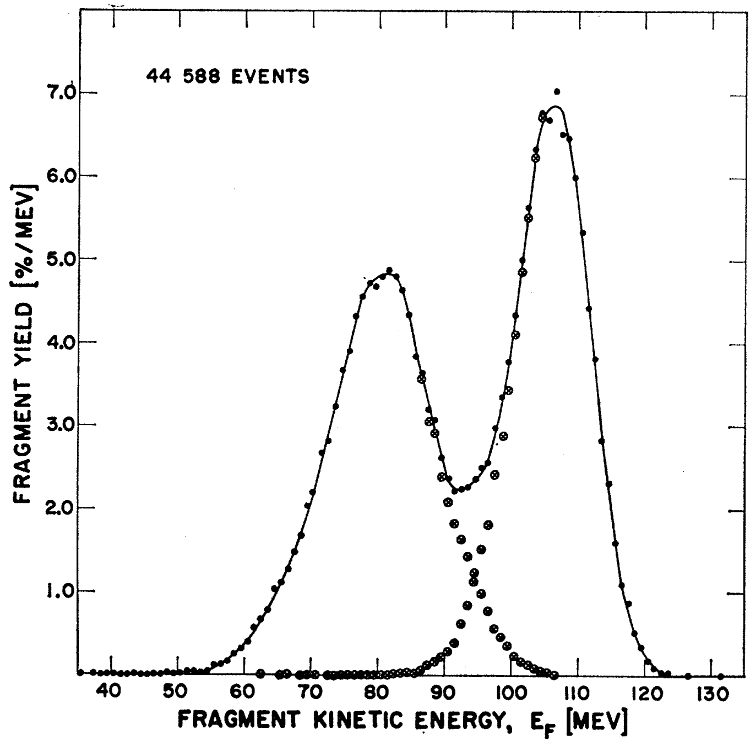


Figure 2.2: Distribution in kinetic energy of the ^{252}Cf spontaneous fission fragments normalized to a total yield of 200%. The ions fragments are shown seperatly with ⊗ and combined with •. The distributions of the heavy and light fragments are also shown, each normalized to 100%. Taken from [2].

ated in a straight line in linear accelerators, or in a spiral in cyclotrons. The high-energy particles are then directed onto a target where they transfer enough energy to cause nuclear reactions.

2.3 Neutral Particles

Uncharged radiation is the other major type of ionizing radiation listed in table 2.1. The two main types of uncharged radiation are electromagnetic radiation and neutrons generated in nuclear processes.

2.3.1 γ-radiation

γ-radiation is a form of high energy electromagnetic radiation, and can be regarded as massless particles, *photons*, with energy equal to the difference in the excited and the lower-lying nuclear levels. The difference in energy level exists after nuclear decay, where the decay products often has sine residual energy. Gamma-ray emission usually happens on the order of pico seconds after the primary disintegration. If it however lasts longer than 1 μs the exited state of the nucleus is called an isomer [8]. Gamma-rays are also commonly emitted after the absorption of thermal neutrons by a nucleus.

2.4 Interaction with Matter

Radiation detection depends the radiations interaction with matter of the detector.;

2.5 Detectors

The detector used in this thesis is a silicon PIN diode detector from SINTEF. It has a thickness of 300 μm and needs a reverse bias of 90 V for full depletion. Then the depletion region will extend virtually across the entire thickness of the silicon wafer resulting in a fully depleted detector [3]. The large electric field created by the high biasing voltage makes charged carriers move very rapidly through the detector, thus charges can be readily lost as a result of trapping and recombination [3].

2.6 Pile-Up

Pile-Up effect are a result one of the distinguishing characteristics of radiation applications compared to other fields of electrical measurements, that in most cases the events in the detector are randomly distributed. The random distribution can cause events to occur very close to each other or event on top of one another. This process is called pulse pule-up and can at high counting rates cause problems by reducing the resolution of the system [3]. There are two main type of pile-ups, *tail pile-up* and *peak pile-up* Peak pile-up is when two pulses are sufficiently close to be treated as a single pulse. One pulse will then be recorded instead of two, and the recorded pulse hight might sometimes be higher than any of the two original pulses. Tail pile-up involves an event occurring on the long duration tail or on the undershoot from a preceding event, see figure 2.3

2.7 Readout and DSP

A basic readout system consists a charge sensitive pre amplifier that amplifies the charge input and outputs a voltage that is proportional to the charge generated by the detector. The Voltage is digitized by an ADC and the digital signal is processed by a digital signal processor. The focus of this thesis is the digital signal processing and the rest of this section will be spent discussing it.

The DSP module in modern detectors can be very simple and only include an event trigger and a FIFO, or it can be more complex and include processes such as baseline correction, zero cancellation, pulse height analysis, glitch filtering, pile up rejection. Which of these processes are implemented depends on amount of resources and development time available and more importantly the

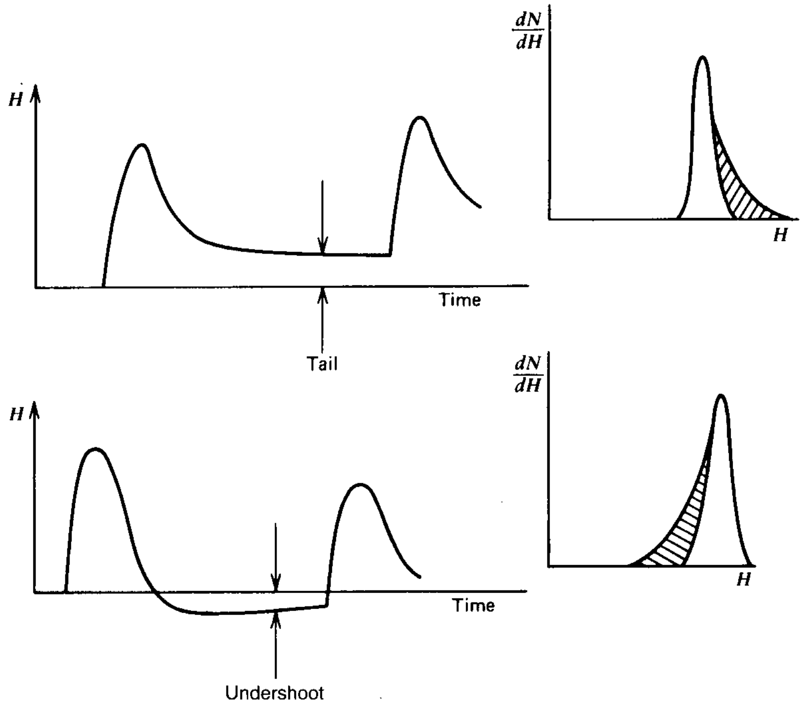
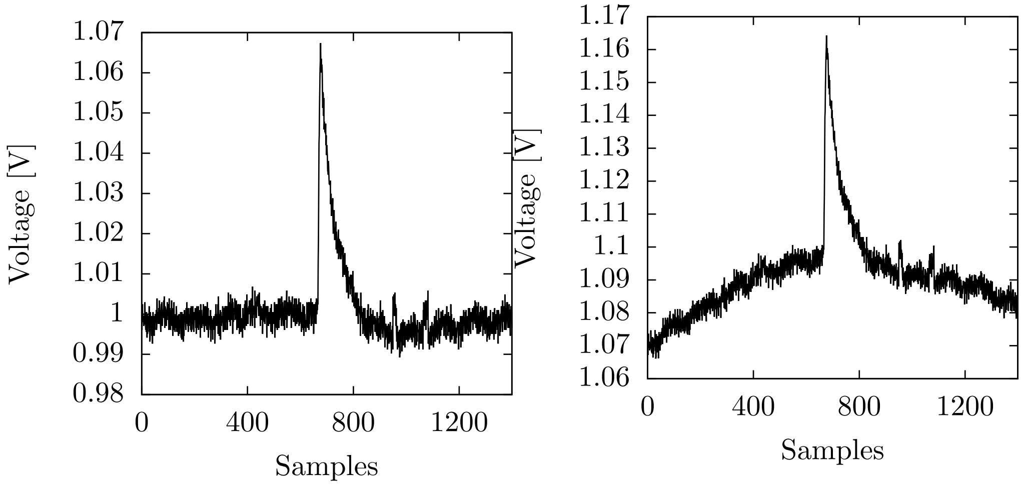


Figure 2.3: Tail and undershoot pile-ups showing the effect on the pulse height histogram. Figure from [3]

application of the detector, whether it will be used as a simple event counter, or if it needs to output pulse heights or entire event pulses. Extracting pulse heights requires more processing than just counting events, and storing events requires memory for storage.

A baseline correction module is necessary if we are to find the correct height of a pulse. If a rising baseline was ignored the height of the following pulses would be recorded as having higher a higher pulse height than they actually have and thus deduced as having a higher energy than they actually have, see figure 2.4.

Another important module is the zero cancellation module. Its purpose is to remove samples that are outside of an event, or rather only store samples that are taken within an event and thus suppress zero samples. Figure 3.6 shows an example of the zero suppression used in the ALTRO chip, where only samples within an event and a set of pre and post samples are stored in memory. Removing samples greatly reduces the amount of data produced, for example an ADC at sampling frequency of 100MHz at a resolution of 10-bit produces 1 Gbit of data every second. However if there is only one event within that second, with an event length of 23 μs, the amount of stored samples can be reduced to $23e - 6 \times 100e6 = 2300$ samples, and the data generated has been reduced to about 23 kbit. This is a huge difference in memory usage compared to the naïve approach of saving all the samples. It allows the use of cheaper hardware by using less memory, and allows sampling at even higher frequencies without running out of memory space.



(a) No baseline shift. Pulse height of 1.06 V

(b) Baseline shifted by about 1 V. Pulse height of 1.16 V

Figure 2.4: Showing the effect of a shift in the baseline.

The length of an event depends on the value of R and C in the feedback of the pre amplifier. If we use R = 4.7 MΩ and C = 2.2 pF the time for the pulse to fall to 10% is 23.8 μs.

$$\begin{aligned} e^{-t/RC} &= 0.1 \\ -t/RC &= \ln(0.1) \\ t &= -\ln(0.1) * RC \\ t &\approx 23.8\ \mu\text{s} \end{aligned} \tag{2.4}$$

2.7.1 State of the Art DSP for Radiation Spectroscopy

ALTRO

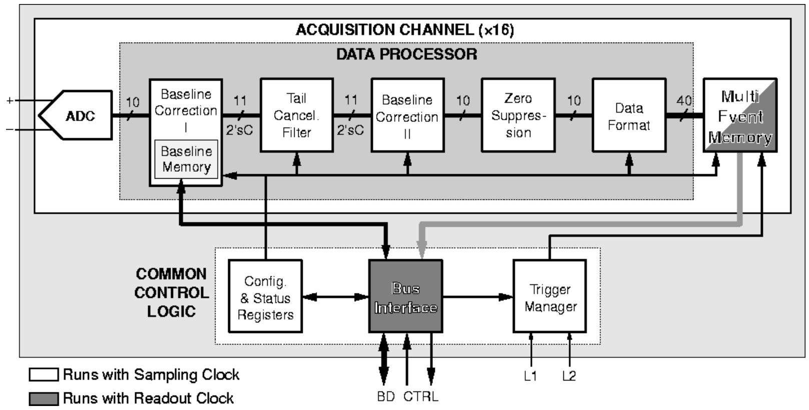


Figure 2.5: ALTRO chip block diagram. (From [4])

One example of a modern digital readout system is the ALTRO chip. It is a 16-Channel A/D converter and digital processor for gas detectors originally conceived and optimized for the time projection chamber of the ALICE experiment at the CERN LHC [4]. Each channel of the ALTRO chip consists of an ADC a data processing module and a RAM module. There have been produced two versions of the ALTRO chip, each with a 10-bit word ADC, one with maximum sampling rates of 25 MSPS and the other 40 MSPS. The Data processing module is divided into five smaller modules and the data is piped through each of them sequentially. The five modules are, a baseline correction module, a tail cancellation filter, a second baseline correction module, a zero suppression module and a data format module. Baseline correction and zero suppression was discussed in the previous section.

The first baseline correction module uses cumulative average also known as running average to self calibrate the input signal. It is calculated using equation 2.5[12], where avg_n is the value of the current baseline, avg_{n-1} is previous baseline value, and din_n is current sample value.

$$avg_n = \frac{avg_{n-1} + din_n}{2} \tag{2.5}$$

The average calculation is halted on the arrival of a first level trigger. This average is subtracted from all the samples during acquisition, and corrects slow baseline perturbations such as temperature drifts. It also incorporates a pattern 1 k × 10 bits memory whose values can be subtracted from the input signal on every acquisition and thus removing systematic perturbations, for example effects linked to the triggering of the detector.

The tail cancellation filter shortens the signals tail, thus minimizing the effects of pileups. It is described in detail in [13].

The second baseline filter implements a moving average window with a window size of 8 samples. As long as the next sample is within an acceptance window defined by two thresholds, one above and one below the current average, it is shifted into the moving average window. The average is then calculated over the entire window using equation 2.6

$$y[n] = \frac{1}{N+1} \sum_{k=0}^{N} x[n-k] \quad N = 7 \tag{2.6}$$

Each sample is corrected using the value of the last calculated average by the moving average filter. Thus removing signal perturbations created by non systematic effects.

Zero suppression works by only storing data that is over a threshold, thus compressing the data stream. It also removes glitches by only storing data if a consecutive number of samples are over the threshold level.

The last module, the Data format module, tags each event with two words, a time stamp and the event size this allows reconstruction of the acquisition afterwards. Time stamps corresponds to the time stamp of the last sample in the set, thus showing the time between events. The size represents the number of 10-bit words in the set including the number of samples in addition to the time stamp and itself. Lastly the module packs the data into 40-bit words. At the end of an event a special trailer word is appended. The trailer word contains information on how to unpack the data, the time stamp, number of words and address data to identify the chip address and channel.

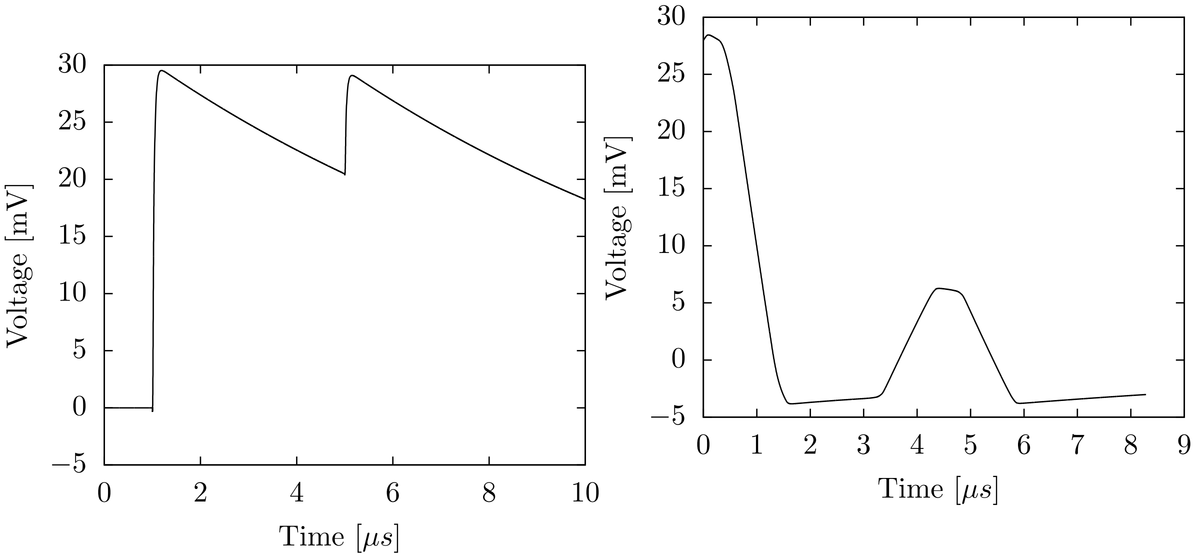
CAEN Digital Pulse Height Analyser series x724

The CEAN digital pulse height analyser series x724 is another example of a modern digital approach to radiation spectroscopy. DPP algorithms are implemented on FPGA firmware. It uses an ADC with word length of 14-bit and a maximum sampling rate of 100 MSPS [14]. The x724 does all the pulse analysis in the digital domain, the only step between the detector and the ADC is a charge sensitive preamplifier. Since it does not have an analog shaping amplifier it uses a digital trapezoidal filter to shorten the long tailed output of the preamplifier. A trapezoidal filter works by using two moving average windows on of length L, one of them shifted by G points. The ith value produced by the first window is subtracted from the ith value of the second window to produce the output, the process is shown in equation 2.7, obtained from[15].

$$\begin{aligned} V_{win1}[i] &= \frac{1}{L}\sum_{j=0}^{L-1} V_i n[i+j] \\ V_{win2}[i] &= \frac{1}{L}\sum_{j=0}^{L-1} V_i n[L+G+i+j] \\ V_{out} &= V_{win2} - V_{win1} \end{aligned} \tag{2.7}$$

When the operation is applied to all input data points the output pulses become trapezoidal and the peaking time is equal to Δt_L and the width of the top is equal to Δt_G. The height of the trapezoid above the baseline is proportional to the amplitude of the input pulse, which means it is proportional to the energy deposited by a particle in the detector. A simulated example of the trapezoid filter is shown in figure 2.6, the filter was simulated using equation 2.7 implemented in python. It is important to note that the trapezoidal filter plays the role of the shaping amplifier in an analog system, shortening the pulse length from the pre amplifier.

In order to correctly evaluate the height of the trapezoid, the x724 uses a moving average window to correct the baseline similar to the second baseline corrector in the ALTRO. When two events are so close or closer that the rising edge of one trapezoid begins on the falling edge of the previous trapezoid, the x724 will determine that a pileup has occurred and rejects the piled up event. If there is a pile up on the rising edge, this method of rejection is not effective, as we do not know the height of the first pulse. In this case the x724 can reject signals that exceeds a settable rise time value.



(a) Output from a charge sensitive amplifier simulated in LTspice. Figure 3.3 shows an example charge sensitive pre amplifier.

(b) Output after it has been processed by the trapezoidal figure. Note that the pulses from (a) has been separated and the tops of the trapezoids are proportional to energy that created the pulse from the pre-amplifier.

Figure 2.6: Showing the effect of a shift in the baseline.

Chapter 3

System

3.1 Overview

As stated in the introduction, this project was divided into two parts The Digital system is divided into two main parts, the DSP on the FPGA and the Linux system running on an ARM processor. The FPGA used in this system is a Cyclone V with an integrated ARM HPS on the same IC. Two different development boards have been used during the development of this project, the DE1-SoC from terasIC with the Cyclone V SoC 5CSEMA5F31C6 IC and the SoCKit from Arrow with the Cyclone V SX SoC 5CSXFC6D6F31C6N. Instead of having a system with only an ARM processor, the FPGA allows us to implement truly parallel modules and to read data on the GPIO pins at a higher rate. Also having an ARM processor on the same IC as the FPGA allows for simpler development of data storage solutions, user interface, and ethernet communication since hardware modules does not have to be developed for these problems. The top level of the system is based on a golden reference design by terAsic written in verilog. It was however rewritten to VHDL fit in with the rest of the DSP and memory modules which were also written in VHDL. The following sections will go into depth explaining all the modules whose code was written for this thesis.

3.2 Memory management

The Arrow SoCKit has 5662720 block memory bits that can be used to store event data. In order to make proper use of these memory blocks a memory management module is needed. Before deciding on a module a list of needed functionality was written down, see list below.

- Separate read and write clocks.
- Possibility to set the read address pointer.
- Setting for either continuous write overwriting the first value when the module is full, or blocking writing when the module is full.
- Flags for indicating full and empty state.

- Data needs to be read chronologically on a first in first out basis.

There are a couple of options that fits a few of these options not all of them. A simple shift register for example will function as a first in first out memory module, it would be pretty simple to implement full and empty flags, but setting read address pointer and having separate read and write clocks is not possible. The Altera FIFO from the IP catalog, it functions as a FIFO, it can use separate read and write clocks, it is possible to disable underflow and overflow protection enabling reading from an empty FIFO and writing to full FIFO. It is however not possible to set the read address pointer, this is a function that is required in the zero cancellation module described later in this chapter. It was therefore decided that a custom FIFO was needed, then all the functionality needed can be implemented and it will also be simple to add functionality than if the Altera FIFO had been used.

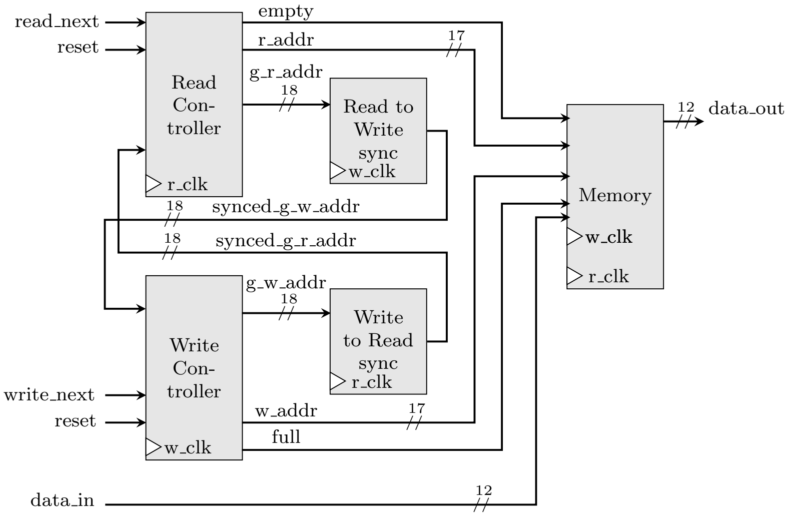


Figure 3.1: Top level view of the asynchronous FIFO .

The FIFO is split into four different modules as shown in figure 3.1, the read controller, the write controller, memory, and a synchronization module. The design is based on FIFO designs from [5]. During development of the FIFO three different versions were written. The first version had a fixed size that was too small to be practical, it was written to test the read and write controllers. Then a parameterized version was written in which the size of the memory buffer could easily be changed during compile time by change a generic Word size. The third version implemented gray address counters instead of binary address counters to avoid errors caused by multiple bits switching when incrementing the read and write addresses and synchronising those addresses across clock domains. The read address has to be sent to the write controller in order to determine whether the status of the full flag, and vice versa for the read controller and the empty flag. The generation of the full and empty flags will be explained after a short introduction to Gray code.

Gray code is used because only a single bit changes when the code is incremented. This minimizes the number of transitions when a signal switches between words, which means the capture of erroneous intermediate transition values when synchronizing a binary counter across clock domains is avoided, because only one bit changes at a time in Gray code [5]. Table 3.1, from [5] shows a comparison between a 4-bit incrementing binary code and a 4-bit incrementing Gray code. Take note of how only one bit changes between each increment in the Gray code, and even when the code wraps around only one bit changes, while for binary every bit transitions from 1 to 0. There is unfortunately no easy algorithm to derive the next Gray code directly [5], therefore an algorithm to convert from binary to Gray code is used instead. The algorithm is based on the fact that the ith bit of the Gray code word is '1' if the ith bit and the $(i + 1)$th bit of the corresponding binary code are different [5], logic equation:

Binary code $b_3b_2b_1b_0$	Gray code $g_3g_2g_1g_0$
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Table 3.1: Incrementing 4-bit Gray code compared to incrementing 4-bit binary code.

$$g_i = b_i \oplus b_{i+1} \tag{3.1}$$

The FIFO is circular, which means that after writing eight values to a 3-bit FIFO the write pointer will be pointing to the first address. Figure 3.2 shows an 3-bit FIFO in three different stages, empty, half full, and full. Note that in the full and empty states the write and read pointer are equal. In order to differentiate between the full and empty state we need an extra bit. Thus for a FIFO buffer of n-bit size, the read and write pointers must be of size (n+1). For binary code we can the tell whether the FIFO is empty or full comparing the MSB of each pointer. If the MSB of both pointers are equal and bits MSB-1

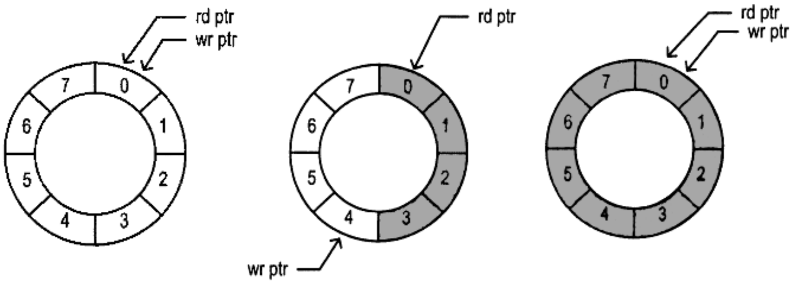


Figure 3.2: FIFO in three different stages, empty, half full and full. Illustration from [5].

down to LSB are equal then the FIFO is empty, while if the MSB's are not equal the FIFO is full, an example with a 3-bit FIFO is shown in table 3.2 Checking whether the FIFO is full or empty when using Gray code for read and write addresses functions similarly. This works because the MSB of a Gray counter is the same as the MSB of a binary counter, see table 3.1. The difference is that in addition to comparing the MSB's, the xor of MSB and MSB-1 of each pointer has to be compared and found to be equal both for the empty and full flag and instead of checking whether MSB-1 down to 0 of each pointer is equal, MSB-2 down to 0 is checked. An example with 3-bit Gray addresses is shown in table 3.3. In row three the two least significant bits are equal and the most significant bits are equal, the empty flag is however not set because the xor of the two most significant bits of each pointer is not equal, thus using this xor method we avoid setting a false empty or full flag. More about Gray code as address counters can be read about in [5].

Write pointer	Read pointer	Operation	Status
0 000	0 000	initial state	empty
0 100	0 000	4 writes	
0 100	0 001	1 read	
1 001	0 001	5 writes	full
1 001	1 001	8 reads	empty

Table 3.2: Example of full and empty flags generated from binary code read and write pointers. The MSB is used to determine whether the FIFO is full or empty.

The two sync modules shown in figure 3.1 are synchronizers using a single flip flop. If no synchronizer had been used the asynchronous signal might cause timing violations and the system register might enter a meta stable state locking up the system [5].

The read address is incremented by the read controller every clock cycle that read_next is high as long as the FIFO is not empty and the write address is incremented by the write controller every clock cycle that write_next is high. The memory module will write data_in to the address pointed to by the read

Write pointer	Read pointer	Write xor	Read xor	Operation	Status
0 000	0 000	0	0	initial state	empty
0 110	0 000	1	0	4 writes	
0 110	0 010	1	0	3 read	
1 110	0 010	0	0	7 writes	full
1 110	1 110	0	0	8 reads	empty

Table 3.3: Example of full and empty flags generated from Gray code read and write pointers. The MSB and MSB $\otimes$ MSB $-$ 1 is used to determine whether the FIFO is full or empty.

address, and output the data that is pointed to by the write address.

The read controller was later modified to allow changing the read pointer, this is explained under the zero cancellation section in this chapter.

3.3 Pre-Amplifier, ADC, and Other Front-End Electronics

The development of the front-end electronics are outside of the scope of this thesis project, they were developed in parallel another master project by Maris Tali. However since all the inputs this project is produced by the front-end electronics they will be described here briefly. For a more in depth description see [7].

The front-end electronics consists of two PCBs, one analog and one digital. The analog PCB has five inputs, channel 1, channel 2, channel 3, I2C bus, and bias voltage for the detector. Channel 1 has a pre-amplifier and a shaping stage for conditioning the input signal from the detector. Channel 2 has only the pre-amplifier. Channels 3 is identical to channel 1. A multiplexer [16] controlled by I2C is used to control which Channel outputs are sent to the ADC on the digital card. The ADC has two inputs and therefore two of the channel outputs can be sent at once [17].

There are also an I2C controlled current measuring device, measuring the current used by channel A. The increase in current used is an indication of the TID damage in the CMOS transistors. The increase in current is primarily caused by a reduction in the threshold voltage [18].

3.4 Interconnect between FPGA and ADC PCB

3.4.1 LVDS vs Single-ended singaling

For the interconnection between the FPGA and the ADC two solutions were looked at.

- Single-ended signaling connected to standard GPIO pins on the FPGA
- Differential signaling using LVDS.

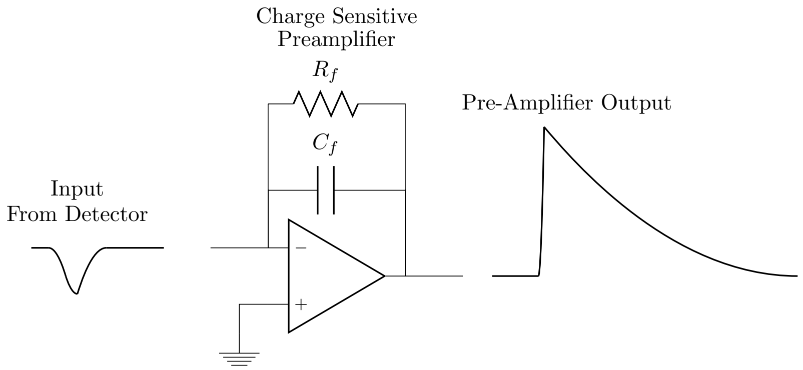


Figure 3.3: .

Two ADC PCBs were produce one with a resolution of 10-bit, 250MHz sampling rate, and LVDS signaling [19]. The other with a resolution of 12-bits, a sampling rate of 125 MHz and single-ended signaling. LVDS, Low Voltage Differential Signaling, allows for very high-speed data transfer in the Gb/s range [20] by use of low voltage swings. The low voltage swing also minimizes power dissipation. Differential signaling makes less susceptible to external noise, and the equal and opposite currents create cancelling electromagnetic fields which dramatically reduces electromagnetic emissions which could disturb other electronic equipment [21]. A disadvantage with LVDS is that it doubles the number of data lines over using single-ended signalling.

Single-ended signaling has a major advantage over LVDS, it uses half the number of data lines. It does however not support as high rate of data transfer because it is more susceptible to external noise. In the end single-ended was chosen over LVDS because we could use the extra GPIO pins for an additional channel and we did not need the higher sampling rate than what a single ended ADC could give us, e.g. 125 MHz, it also had a higher resolution of 12-bit instead of 10-bit [17].

3.4.2 I2C

The I^{2}C-bus or the Inter IC bus was developed by Philips Semiconductors (now NXP Semiconductors) in the early 1980's. The I^{2}C-bus has several desirable features[22]:

- It uses only two bus lines, a serial data line (SDA) and a serial clock line (SCL), which reduces the hardware complexity.
- Each device connected to the bus is software addressable by a unique address and simple master/slave relationships, masters can operate as master-transmitters or master-receivers.
- There can be multiple masters on one bus, and includes collision detection and arbitration to prevent data corruption if two or more master simultaneously initiate data transfer.

- 8-bit oriented bidirectional data transfer can be made at up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, up to 1 Mbit/s in Fast-mode Plus, or up to 3.4 Mbit/s in the High-speed mode.

- On-chip filtering rejects spikes on the bus data line to preserve data integrity

The LTV2242-10 ADC sampling rate is controlled by and external clock and has a minimum and maximum encode rate of 1 Msps and 250 Msps respectively[19], similarly the 12-bit ADC LTC2283 has a minimum and maximum sampling rate of 1 Msps and 125 Msps [17]. Using a programmable clock generator enables the end user to choose the desired sampling frequency. There are also other units on the analog and digital PCBs designed from Maris Talis's thesis [7] which need to communicate with the FPGA, such as the multiplexer and current measuring devices and more could be added in the future. There are several communication interfaces used by programmable devices other than I^{2}C, SPI being another widely used option. I^{2}C was chosen over the other options for the following reasons. I^{2}C uses only two output pins on the SoCKit, while SPI uses three pins and a selection pin for each slave if the slaves are independent or one selection pin shared by cooperative slaves. Although the current project has enough output pins to accommodate SPI, I^{2}C enables more slave devices to be added easily without using up extra pins which can in turn be used to adding more ADC channels. SPI is faster than I^{2}C and does not have a 8-bit word limit, however since the interface will be used to configure a clock generator and not for high throughput data transfer, the I^{2}C controller on the Cyclone V which supports Fast-mode at up to 400kbit/s of 8-bit words is enough. The ARM processor on the Cyclone V chip already has three I^{2}C controllers implemented, which avoids the disadvantage of I^{2}C being harder to implement than SPI. This also means we don't have to use logic blocks on FPGA to implement the communication interface, which frees them up for use in DSP logic and other uses instead.

3.5 FPGA digital signal processing

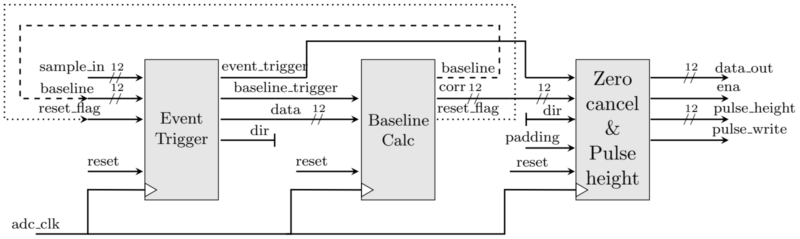


Figure 3.4: Top level view of the digital signal processing modules.

The Digital processing on the FPGA is modular by design, this allows for turning on and off modules or adding more modules in the future. As the modules are split into separate files, it also makes the code easier to read and debug and enables simulation of a single module at a time. A block diagram of the DSP system is shown in figure 3.4. Data enters the FPGA in parallel from the ADC and is sent to the Event trigger module where it is buffered through a shift register of length five before it is sent further into the baseline calc module together with an event trigger. The baseline calc module calculates the average of the last sixteen samples using a moving average window. The average value is used to baseline correct the signal so that zero samples, samples in between events, are corrected to 2048 the 0 voltage level of the ADC. The average value is also sent back to the event trigger which uses it to generate trigger flags if input samples are larger or lower than a trigger level set by the user. It generates two triggers; a baseline trigger that stops the baseline calculation when there is an event, and an event trigger that starts sampling the event. The reason for having two triggers is to stop the baseline correction as soon as possible, because setting a high baseline trigger attenuates the pulse heights, this is shown in the section on calibration. The event trigger can be higher so we avoid storing samples of noise. Baseline corrected samples, event trigger, and padding lengths are sent into the Zero cancellation & Pulse height module which controls the storing of samples and pulse heights into two separate FIFOs. The padding controls how many samples to store before and after an event. The following sections goes into depth explaining each module.

3.5.1 Event Trigger

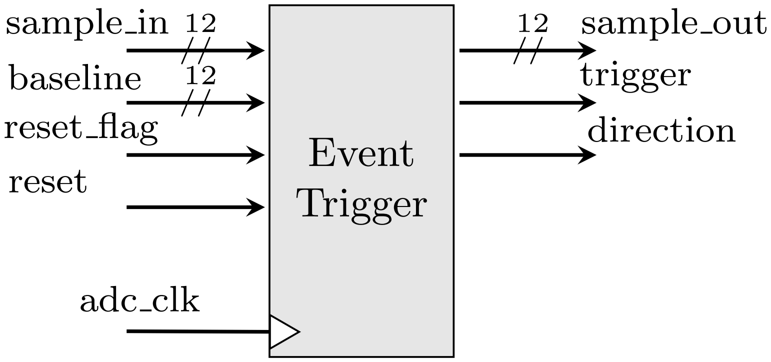


Figure 3.5: Top level view of the Event trigger module.

Proper event triggering is important for a variety of reasons and there are several ways of generating event triggers. In this project it is used by the zero cancellation and pulse height module by only enabling storing of sampled data during an event, and by the baseline correction module by only calculating a new baseline outside of events. During development of the system two ways of generating the event trigger was tested. The first was the naive method of enabling the trigger flag whenever a sample was over or under a preset trigger level. Say for example the trigger level is set to 30 and the baseline is at 2048, then if the next input sample is above 2078 or below 2018 then the event flag is set high, baseline calculation is stopped and event storing starts. This causes problems when there are glitches or noise that produce a few samples above the trigger level. The noise will cause the storing of a false event and stop the baseline calculation. To prevent this the system implements a shift register with a depth of five samples. The linear shift register consists of five daisy chained flip flops. The data is clocked through the flip flops and when the output value of all the flip flops are below the lower event threshold or above the high event threshold an event flag is set. This is similar to how the ALTRO chip does glitch rejection which is shown in figure 3.6 (figure from [4]). However the ALTRO chip does the glitch removal at the zero suppression stage, while in this project the zero rejection is not activated before a signal that is above the trigger level is determined not to be a glitch which is when five or more consecutive samples is above the trigger level. This project could have put the event trigger on the inside of the zero suppression module as in the ALTRO, the reason that this was decided against is that the baseline calculation module also needs an enable signal that enables it outside of event. Keeping all the event flag generation in one module simplifies the design. ALTRO avoids this by receiving trigger signals L1 and L2 from an external circuit that enables and disables the individual modules, see trigger manager block in figure 2.5. It was later discovered that setting a high trigger attenuates the pulse heights, an explanation of why this happens is given in the Baseline Calc section. To alleviate this problem a new version of the event trigger was written which has two trigger levels. One for stopping the baseline calculation and the other for starting the sampling in the Zero suppression module. Thus the problem of the baseline correcting away the event is avoided by setting a lower baseline trigger and we can set a higher event trigger so we avoid sampling noise. Both trigger levels can be set during run time through software running on the ARM processor which communicate with the FPGA logic using the Avalon bus.

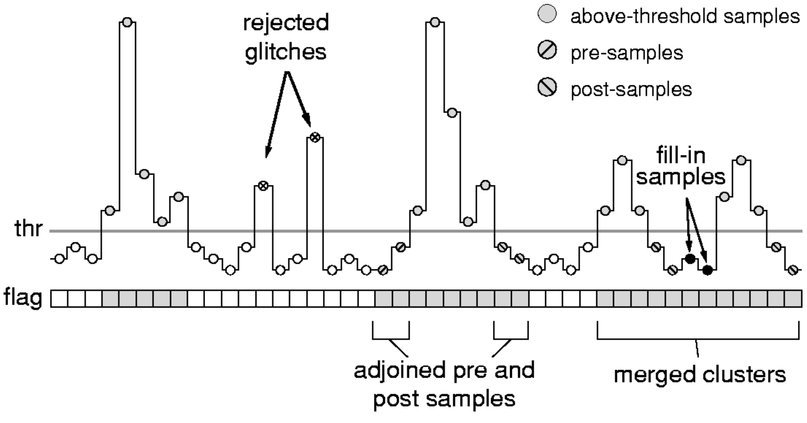


Figure 3.6: Glitch rejection and zero suppression scheme in the ALTRO chip. (From [4])

3.5.2 Baseline Calc

Baseline correction is essential for getting the correct height of a pulse. Changes in measuring environment can cause a drift of the baseline as shown in figure

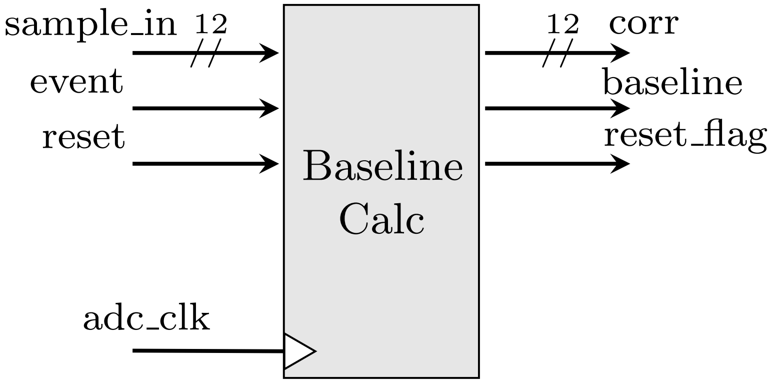


Figure 3.7: Top level view of the baseline calculation module.

2.4. If the drift is not corrected for the height of the pulse will be found to be either higher or lower than it actually is. The module has two main functions, calculating the baseline and baseline correct the signals. Two different methods of finding the baseline was tested in this project, running average also known as cumulative average and moving average. Running average is what the ALTRO chip uses in its first baseline correction module when running in *self-calibrated subtraction mode* [12]. The algorithm used for generating a running average is shown in equation 2.5. The baseline starts out at 0 and when a new sample enters, with a value of say 2048, the new baseline is calculated as $\frac{0+2048}{2} = 1024$, if the next sample after that is also 2048 the third baseline value will be calculated as $\frac{1024+2048}{2} = 1536$. The baseline converges quickly because each successive sample has a big influence on the baseline value, there is however a downside to the running average method. When each sample has such a large effect on the calculated baseline, the event trigger needs to stop the baseline calculation as soon as possible when there is an event. If it does not, the entire signal pulse will be removed by the baseline correction. This can easily be shown by example. Since we are using offset binary with a resolution of 12-bits, 0 V is at $2^{11} = 2048$ everything above 2048 is a positive voltage and everything below 2048 is a negative voltage. If we assume the baseline is at 2048 and we get an event with a height of 100 over the baseline that has 5 samples on the rising edge equally spaced and the trigger level is set to 35 the following baselines will be calculated.

$$\begin{aligned} \frac{2048+2068}{2} &= 2058 \\ \frac{2058+2088}{2} &= 2073 \\ \frac{2073+2108}{2} &= 2090 \end{aligned} \tag{3.2}$$

The next sample, 2128, is larger than 2090+35 and will therefore be considered an event. What has happened though is that the baseline has been found to be 2090 instead of 2048 and the pulse height is found to be 58 instead of 100. This problem gets even worse as the sampling rate increases, because the sample increments will be more gradual and no one sample will go over the threshold. The glitch protection implemented in the event trigger was the straw that broke the camels back since it needs 5 values over the threshold before baseline calculation is halted. The ALTRO gets around this by using a level 1 trigger supplied to it externally, our system does unfortunately have this functionality.

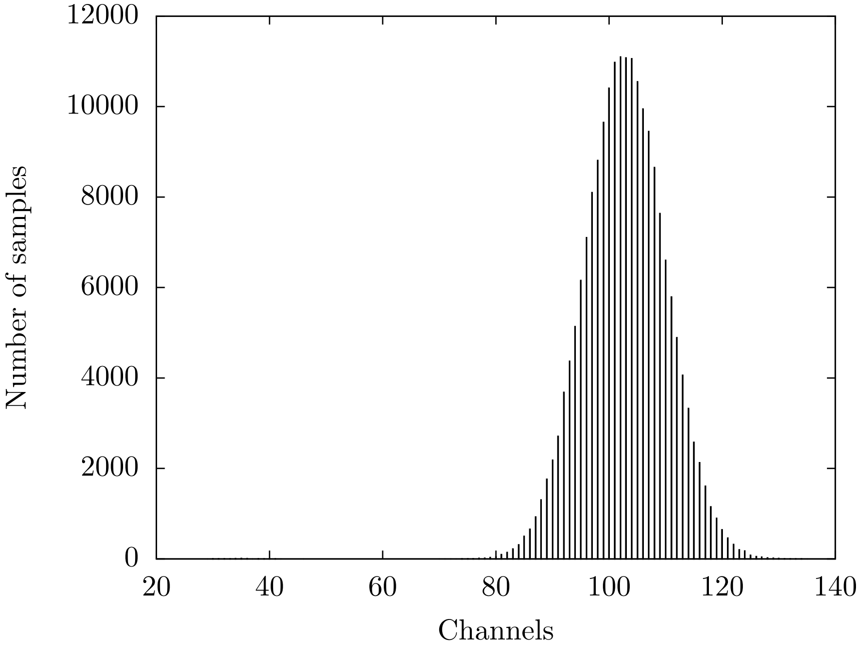


Figure 3.8: Histogram of pulse heights produced by 30 mV exponential rise pulses when trigger level is set to 40. Mean at 102, approx 11400 samples at mean.

A quick fix was implemented by setting the threshold level lower however this has the unwanted consequence of rising the baseline because not all the noise is taken into the baseline calculation which results in pulse heights being measured higher than supposed to. Pulse height histograms will also be cluttered by noise. Both these problems can be seen in figures 3.8 and 3.9 where pulse heights are shifted up 10 channels when the trigger level is set to 10 instead of 40. A better solution was to implement a moving average window instead of a runnig average. The moving average implentaition written for this thesis uses a window length of 16, which means that the average is calculated using the last 16 samples. This results in each induvidual sample having a smaller influence over the average, and thus events do not cause as big a shift in the baseline as when using a running average.

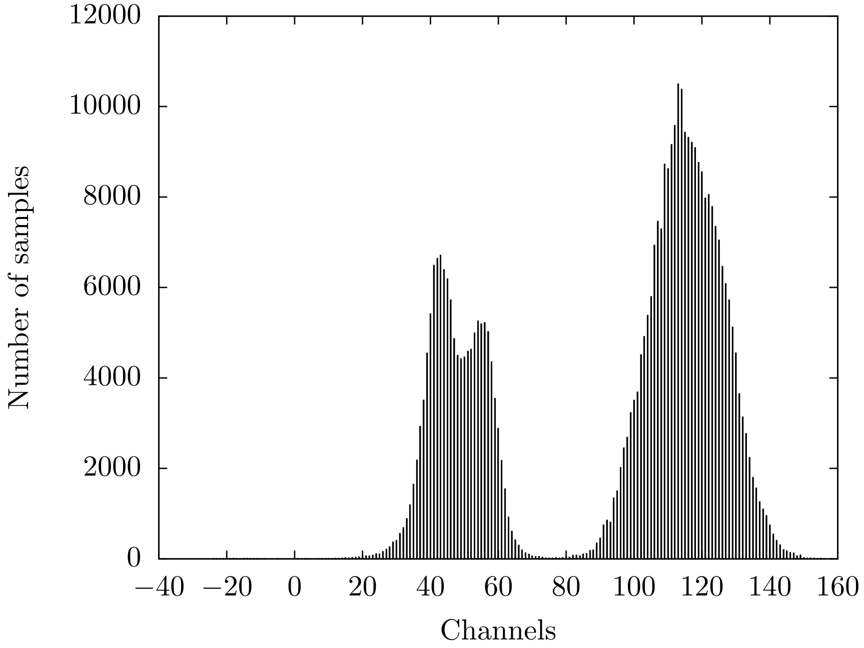


Figure 3.9: Histogram of pulse heights produced by 30 mV exponential rise pulses when trigger level is set to 10. Mean at 113, approx 10500 samples at mean.

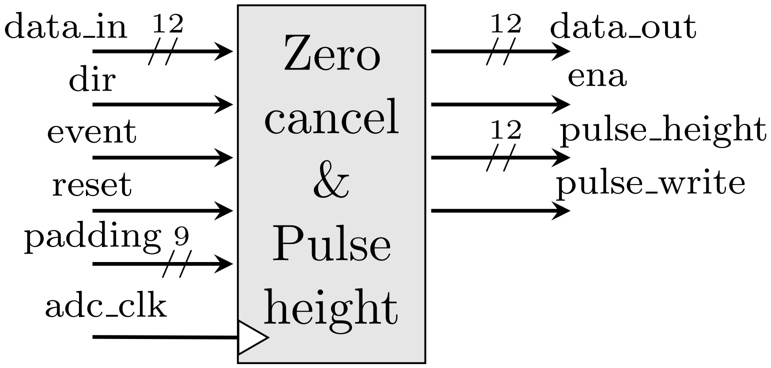


Figure 3.10: Top level view of the Zero suppression & and Pulse height module. The pre padding and one for post padding are drawn as one line to save space in the figure.

3.5.3 Zero suppression and Pulse height storing

The Zero suppression & Pulse height box shown in figure 3.4 contains the zeros suppression function and the Pulse height extraction module. Figure 3.10 shows the top view of the zeros suppression module. It receives 12-bit data words from the baseline correction module on every falling clock edge. The event flag input comes from the event trigger module and is set high synchronous with adc_clk when an event is detected. The reset flag is a global reset. Out of the zero suppression module comes four signal lines, data_out, ena, pulse_height, and pulse_write. The data_out signal is a 12-bit word whose output is controlled by the internal state machine, the data is read out of an internal buffer FIFO on falling clock edge. When new data on the data_out line is available the ena flag is set high. Pulse_height and pulse_write functions similarly in that there is an internal FIFO storing pulse heights which is read from and the value output on the pulse_height signal, the pulse_write flag is set high when pulse_height is updated.

The zero suppression has two main modules, a synchronous buffer FIFO with a default depth of 512 and a state machine that controls writing to and reading from the buffer. The state machine has three states, Idle, Event, and Padding.

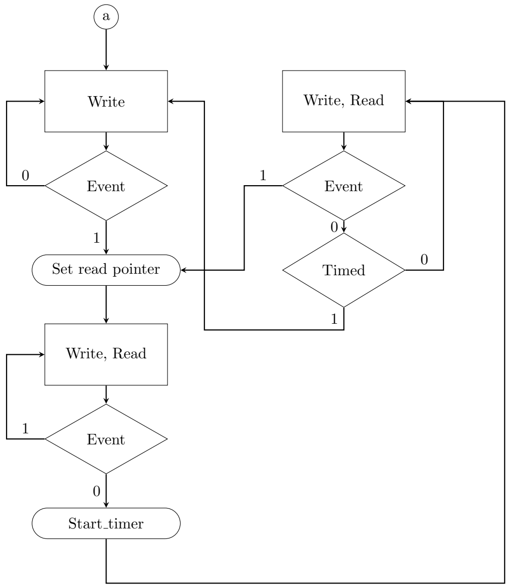


Figure 3.11: ASM diagram showing how reading and writing to the buffer FIFO is controlled.

In the Idle state the machine waits for the event flag to go high, all the while writing to the buffer. This means that when there is an event the beginning of the event is also saved in the buffer and reading can start from the beginning of an event even if the event flag was not set until a few samples into actual the event. This works because the buffer is a circular FIFO, a modified version of the FIFO described earlier in this chapter. The FIFO differs from the one described above in that it never becomes full or empty, it is just continuously written to and its circular function means that the last 512 samples are stored in it. When the event flag is set to 1 the read pointer of the buffer is set a few values behind the current write pointer, by default it is moved five steps back but it can be set manually by the user from the user interface. The maximum number of pre samples possible is 511, its a limit because of the size of the buffer. If we set the read pointer to 511 behind the write pointer then there will only be room for one more write before the FIFO is full, but since the FIFO will start to be emptied in the next state it will never actually become full. The next state is set to the Event state when there is an event.

The event state enables reading from the buffer FIFO while still writing to it. Since reading and writing happens at the same rate, the write pointer will never overtake the read pointer, and the pre event data is safe. The reading happens by setting the flag i_r_next high which tells the buffer FIFO to output a new value. The flag is also connected to a FIFO outside of the buffer module, which tells it that a new sample is available. The Event state also controls the Pulse height finder. The Pulse height finder depends on the direction flag. If it is set to 1 it means the event is positive and it will store the largest value in the event. If the flag is set to 0 the event is negative and it will find the smallest value in the event. When the event ends the currently stored pulse height will be written to an external FIFO. The Event state simultaneously starts the timer of the timed process and sets next state to Padding.

State Padding keeps writing and reading to the buffer FIFO until the timed process has finished or it stops the timed process if a new event is detected. If the former happens the next state will be set to Idle, if the latter happens the next state will be set to Event. If an event is detected early on the tail of the signal its pulse height will be higher than it should be and the energy will be determined to be higher than it should, this would be an example of a pile up.

3.6 Communication between FPGA and ARM

Communication between the FPGA and ARM is done using Avalon memory mapped bridge. Two bridges are used, the HPS-FPGA bridge for reading data from the FIFOs and the Light-weight HPS-to-FPGA Bridge for controlling the settings of the DSP system such as the padding length and trigger levels. Both bridges allows for masters in the HPS to access logic in FPGA fabric. On the FPGA side are three register modules one for sending data to the DSP modules, such as padding length and trigger levels. The other two are for reading data from the event FIFO and pulse height FIFO of each of the two channels.

The HPS ARM is running a stripped down version of Linux maps the addresses of the register modules into memory, so that the standard Linux functions write and read could be used to transfer data between the ARM and FPGA. Three different ways of transferring the data from the ARM to the end users laptop were tested. The first was simply writing data to the On board SD-card, then downloading the data using scp. This worked well with the SoCKit as it had 1 Giga Byte of system ram to buffer data in before writing to the SD-card, which meant that the SD-cards slow write speed was not a bottle neck. This way of doing it was however not found to be very user friendly as it requires intimate knowledge of the command line in order to download the data. A more user friendly system was developed by Maris Tali. It uses TCP/IP socket server/client system to transfer data from the ARM running the server to the users laptop running the client. A GUI was also written which allows for live histogram plotting, setting of padding and trigger values, storing of data etc. A more in depth explanation can be found in [7].

The third way of transferring data was mounting a disk on the users laptop using NFS. The light weight Linux system that was used with this project did however not have NFS installed, and it proved hard to cross compile a version for ARM. A bug was discovered in the Makefile of NFS and reported and was patched by the NFS developers [23]. In the end it was simpler to build a Linux image from scratch using Yocto, an open source collection of tools for building custom Linux-based systems. The option of using NFS to transfer data was ultimately abandoned in favor of the other two methods.

3.7 Top Level

The following figure 3.12 is included to give the reader an idea of how all the modules are connected in the top level. It shows a simplified version of how the DSP modules, and FIFOs, ADC and ARM processor are connected, and illustrates that they run off of two separate clocks.

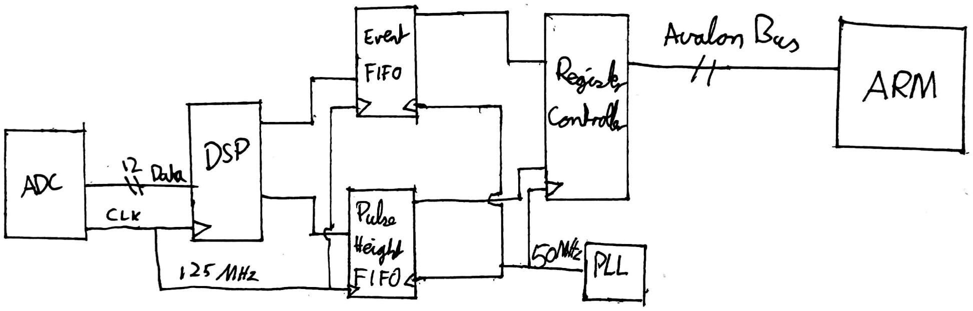


Figure 3.12: Top level view Of the Digital system.

Chapter 4

Testing and Calibration

During the development process the system was put through several steps of testing, verification and calibration before experiments with a radiation source was run. The steps were as follows

1. Research other similar projects.
2. Write down needed functionality and split it into modules.
3. Emulate each module in python for rapid prototyping.
4. Translate the modules into VHDL
5. Write a test bench that compares the output of the VHDL module to the output of the emulated module.
6. Synthesize the module and test with input pulses from signal generator.
7. When all modules are in place, calibrate the system.
8. Measure noise in system.
9. Run experiment with radiation source.

After researching what functions were needed to produce energy distribution histogram by reading about similar systems such as [14] and [4], the functionality was split into modules whose functionality could be drafted up on paper. Making the design modular made it possible to develop and test parts of the system separately. All DSP modules were emulated one by one in python as this allowed to rapidly get a working prototype whose output can be used to verify the functionality of the VHDL modules. When a working prototype had been written in python it was translated into VHDL which was verified using an automated test bench that compared the output of the VHDL program to the output of the python emulation. After the VHDL module was verified to function the same way as the emulated module it was synthesized and tested using signals from a signal generator. After all modules had been finished the system was calibrated, the noise was measured and charactered and finally experiments with radiation sources preformed.

4.1 Emulation

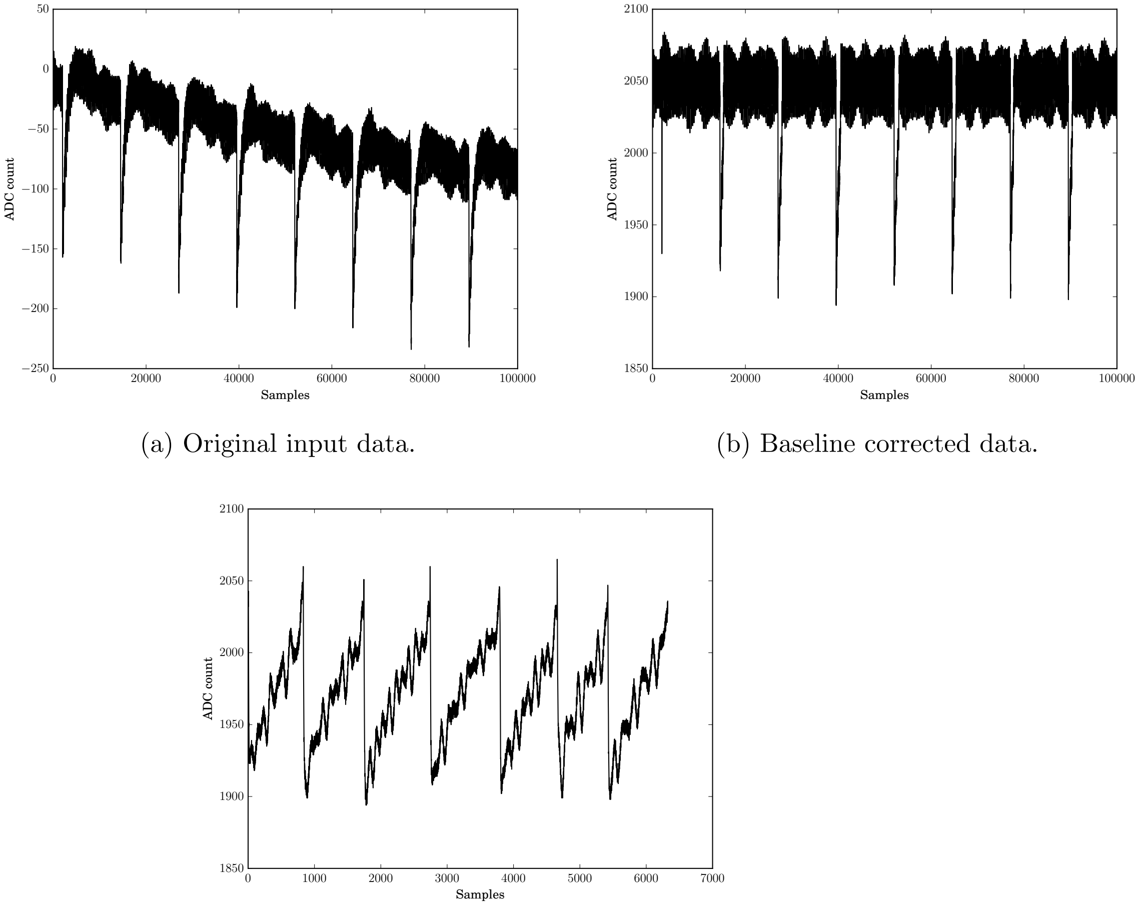
Emulating The separate modules allows for quickly testing concepts and seeing what works and what does not. Each module was developed and tested separately, first the baseline calculation and correction, then the zero cancellation and finally the pulse height counter. The emulated modules were written in python were each module has its own class to allow for individual instantiation. The data used as input for the emulation was generated by HMF2525; an arbitrary function generator [24], sampled with the readout system described in this thesis but with the DSP disconnected so that the signal was stored unaltered. The sub figures in figures 4.2 shows plots of the output data when different modules are connected. Sub figure 4.1a shows the data input which has been modified to have a falling baseline. Further in sub figure 4.1b shows the output of the emulation when the baseline correction module is connected. It lifts the baseline up so that it is centered around the ADC zero value which is 2048 for LTC2283 [17], this is important so that the proper pulse height can be found. One of the events has been almost removed, this happens because of the reset state of the baseline corrector. After initialization the system uses 5000 samples to let the baseline correction stabilize. Five thousand samples when sampling at 125 MHz takes $\frac{5000}{125\times10^6} = 40\ \mu$s, which is practically instantaneous for the user. However when using a moving average window it is not necessary to to sample more than the length of the window, which is 16 samples. Even when using running average five thousand samples is excessive and should be reduced in future iterations. Sub figure 4.1c shows the output when both the baseline correction and the zero cancellation is connected. Values that are outside the events have been removed thus reducing the number of samples from 100000 to 6500, a reduction in data by over 93%. The events are cut short, but if the user wants to see more sample before and after the events the padding can be increase from software.

A rising baseline was also added to make sure the system could handle that as well, the result of adding a rising baseline is shown in figure 4.2a. The sub figures 4.2a, 4.2b, and 4.2c, show that the system works just as well with rising baselines as with falling baselines.

In addition to baseline correction and zero cancellation, the pulse height finding module was also emulated. The pulse heights, zero cancellation, and baseline correction data were all written to file so that it could later be used for verification with VHDL.

4.2 Verification with VHDL

After completing emulation of one module it was translated into VHDL code. In order to verify that VHDL code actually worked as expected, the code needed to be simulated. The goal of the verification process is to make sure that the system can accomplish the task its been designed for successfully. During verification a testbench is written to test the DUT. The purpose of a testbench is to determine the correctness of the DUT, which according to [25] is accomplished with the



(a) Original input data.

(b) Baseline corrected data.

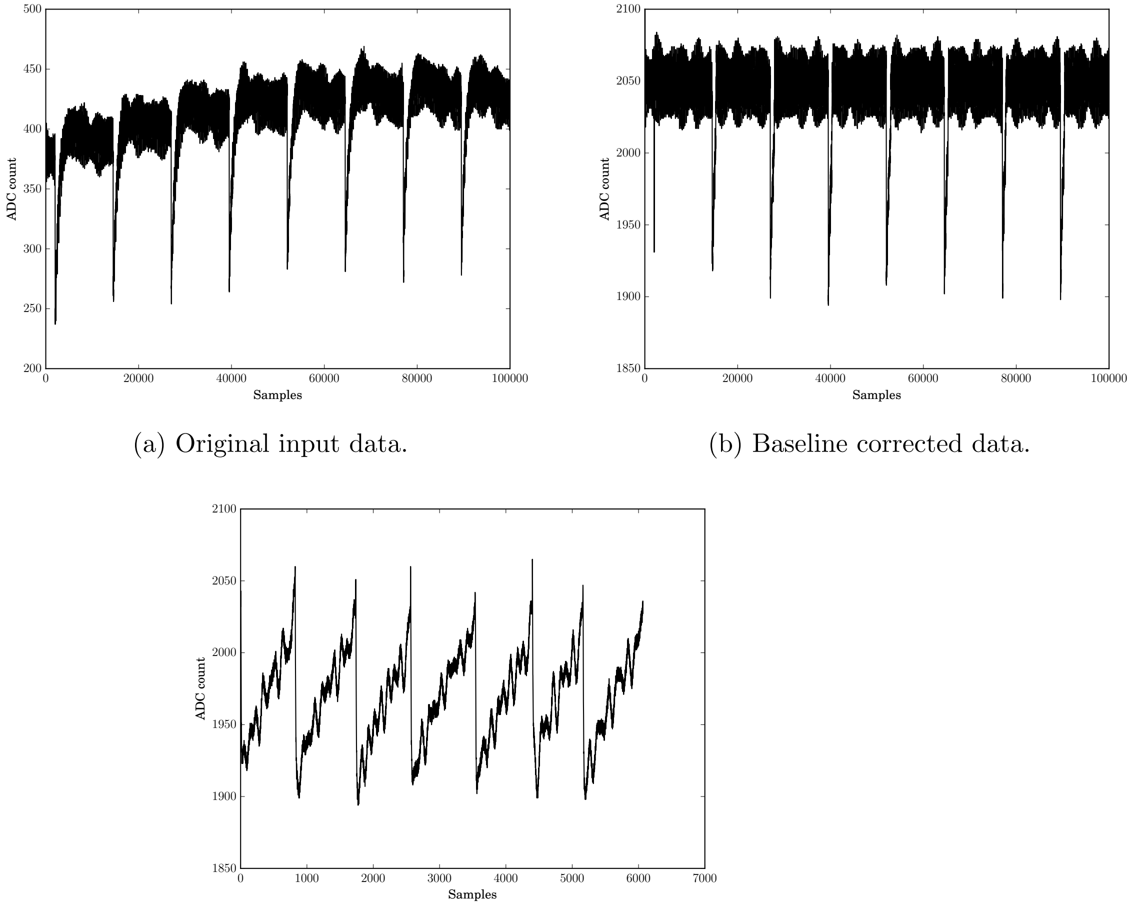
(c) Zero cancelled and baseline corrected data.

Figure 4.1: Emulation of baseline correction and zero cancellation when the baseline is falling.

following steps.

- Generate stimulus
- Apply stimulus to the DUT
- Capture the response
- Check for correctness
- Measure progress against the overall verification goals

These steps can be accomplished both with manual or automatic testbenches, however automatic testbenches allows for quicker testing and thus enables testing with more data. It would have been impossible to test the modules with the data containing one hundred thousand samples that was used during the emulation. When designing the testbench a choice had to be made about what



(a) Original input data.

(b) Baseline corrected data.

(c) Zero cancelled and baseline corrected data.

Figure 4.2: Emulation of baseline correction and zero cancellation when the baseline is rising.

language to use, System Verilog or VHDL. While System Verilog has some advantages over VHDL, most prominently supporting Object-Oriented Programming (OOP) which allow for creating complex data types and tying them together with routines that work with them [25]. Systen Verilog also supports some advanced features that VHDL does not, [25] lists the following.

- Constrained-random stimulus generation
- Functional coverage
- Higher-level structures, especially Object-Oriented Programming, and transaction level modeling
- Multi-threading and interprocess communication (IPC)
- Support for HDL types such as Verilog's 4-state values

- Tight integration with event-simulator for control of the design

However since the modules are written VHDL, writing the testbenches in VHDL would simplify the verification process, and since we only want to verify that the deign works the same way as the emulation the advanced features of System Verilog are not needed. In addition the Bitvis Utility Library, a open source VHDL testbench infrastructure, simplifies the writing of testbenches in VHDL. The libraries key benefits according to [26] are as follows.

- Simpler and faster testbench development
- Significantly less code per test case
- Far more readable and modifiable test cases
- A more uniform test sequencer coding style
- A good simulation progress and result report
- An excellent log for debugging DUT and TB

Thus VHDL was chosen over System Verilog. The simulations were run using the HDL simulation environment ModelSim by mentor, and ModelSim scripts were written automate the entire process of compiling the modules, the testbench, and the Bitvis libraries, and running the simulation. This made testing the features and debugging a lot quicker. The testbench instantiates the baseline calculation, event trigger, zero cancellation, and pulse height modules, and checks their output against the output generated during the emulation. If there was a mismatch between the data from the emulator and the VHDL simulation a warning was printed with the index indicating where in the process the error occurred. Once the source of and error in a module were discovered the module was rewritten so that the VHDL modules functioned identically to the emulated modules.

The async FIFO was also tested using bitvis. Two separate tests were performed, one with a large FIFO writing a hundred thousand samples to it and reading them out again and checking that the data had not been corrupted. Two separate clocks were used to verify that it functioned correctly with one clock for writing and another for reading. The second test was run using a smaller FIFO testing the full and empty status flags, checking that they were generated correctly and that reading from an empty FIFO and writing to a full FIFO was not allowed.

4.3 Calibration

Calibration was preformed in order to find an equation for converting from ADC values into energy measured in MeV. Another goal of the calibration is to find the FWHM of the system when the no detector was connected which gives us an idea of the electrical noise in the system. Figure 4.3 gives a visual explanation of what FWHM is.

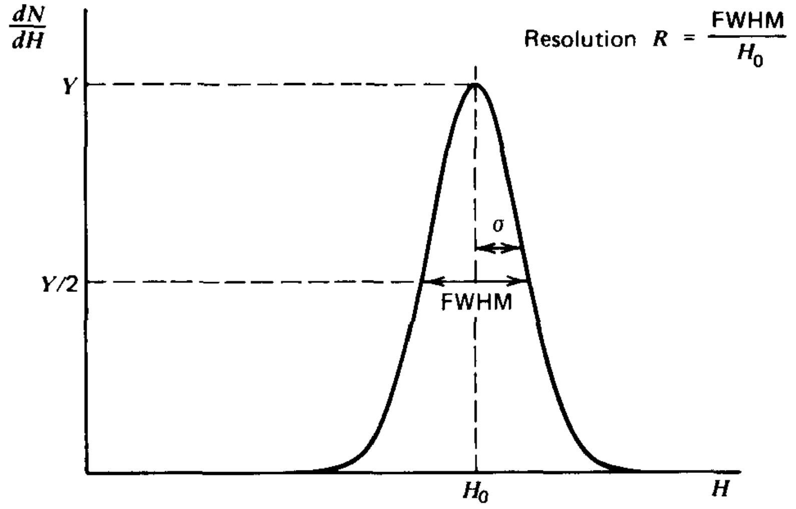


Figure 4.3: FWHM is given by 2.35 σ for peaks whose shape is Gaussian with standard deviation σ. Figure from [3]

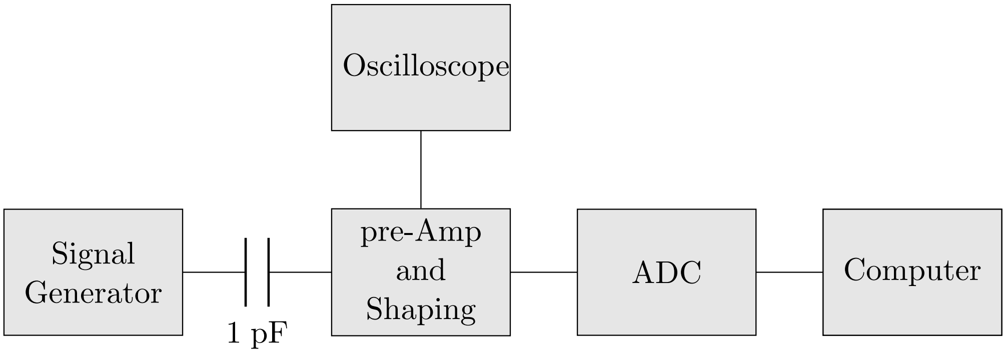
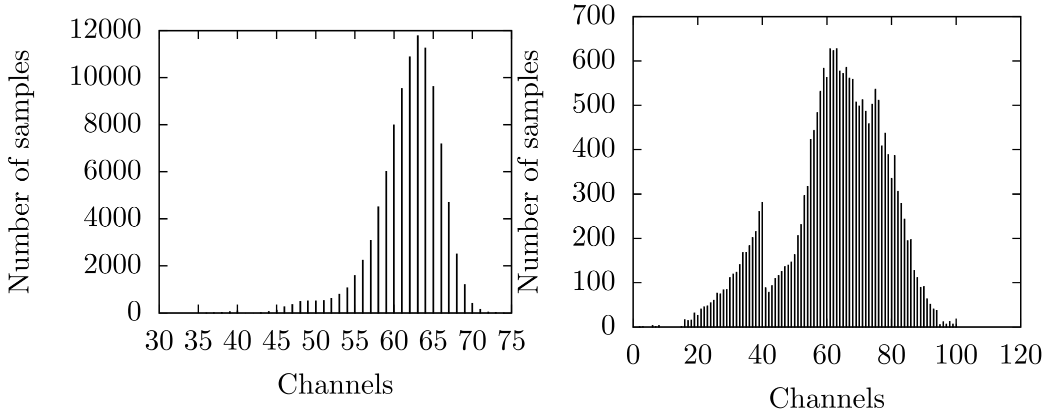


Figure 4.4: Calibration setup for Channel 1. Channel 2 was calibrated similarly but with a 2.2 pF capacitor. The oscilloscope had to be disconnected because of the noise it introduced to the system, see figure 4.5.

Figure 4.4 shows a block diagram of the setup when calibrating the system. The 2.2 pF capacitor placed between the signal generator and the pre-Amp is used to simulate charge generated when a charged particle hits the detector. The height of the voltage peaks from the signal generator depends on the size of the test capacitor. Generating an electron hole pair in silicon requires 3.62 eV [3]. 1 eV equals 1.602×10^{-19} C, thus a 1 MeV particle releases all its energy in a Si detector it will produce $1\,\text{MeV}(Si) = \frac{1\text{ MeV}\times 1.602\text{e}{-19}\text{ C}}{3.62\text{ eV}} = 0.0444\,\text{pC}$. To generate 0.0444 pC from a test capacitor of 2.2 pF we need voltage pulses with peak $V = \frac{Q}{C_t} = \frac{0.0444\text{ pC}}{2.2\text{ pF}} = 20.2$ mV similarly for a 2.2 pF test capacitor we need pulse heights of 20.2 mV. Pulse heights were counted at the equivalent of 1, 2, 5, 10, 20, 30, and 40 MeV. The oscilloscope had to be disconnected when because it introduced a lot of noise which can be seen by the pulse height histograms in figure 4.5

4.3.1 Channel 1 calibration



(a) Pulse heights with 44 mV input pulses through a 1 pF, equivalent to 1 MeV, with oscilloscope disconnected and turned off.

(b) Pulse heights with 44 mV input pulses through a 1 pF, equivalent to 1 MeV, with oscilloscope connected

Figure 4.5: Showing the effect of a shift in the baseline.

Figure 4.6 shows a histogram of pulse heights from calibration with the equivalent of 20 MeV pulses. A histogram for each energy tested under calibration was plotted and the full width at half maximum was found, the results can be seen in table 4.1. Some of the data samples had a few values less than 60, which were caused by noise that was mistaken for pulse signals by the event trigger, removing these improved the standard deviation. These small pulses are okay to remove as they are so far below Gaussian mean of the actual signal, they are outliers that should have been filtered out any ways. Fifty nine were removed from 2 MeV, fifty nine from 5 MeV, twenty nine from 10 MeV and 15 from 20 MeV. Nothing was removed from 30 and 40 MeV.

mV	MeV	ADC count	$FWHM$
44	1	62	9.80
88	2	120	6.72
222	5	285	4.94
444	10	558	4.79
888	20	1103	6.04
1333	30	1651	7.17
1777	40	2039	4.56

Table 4.1: Calibration results of channel 1 with baseline trigger set at 15 and event trigger set at 40.

In order to see what effect if any changing the triggers had, a second cal-

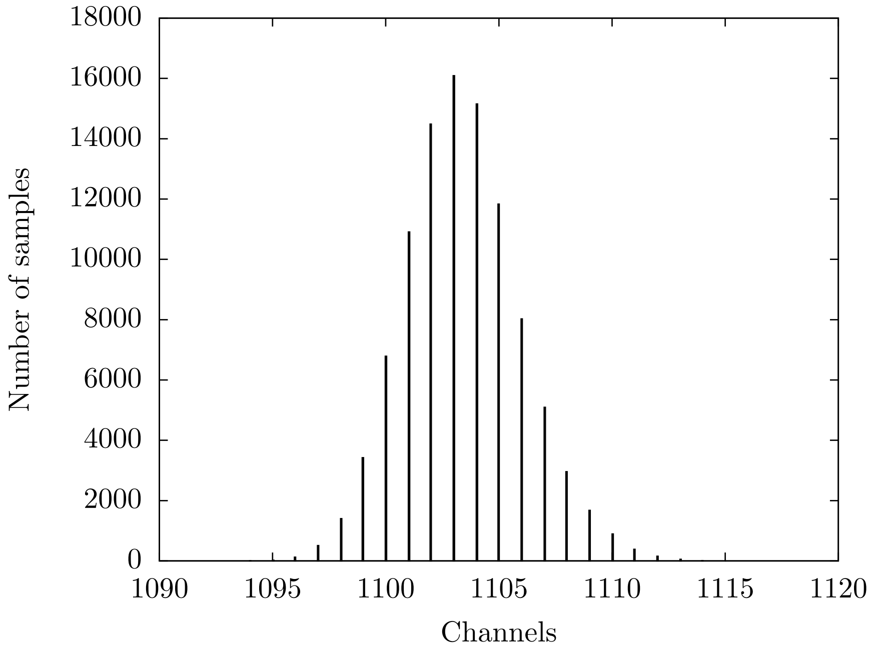


Figure 4.6: Histogram of pulse heights produced by the equivalent of 20 MeV. under calibration of channel 1. Mean at 1108.51

ibration was run with the baseline trigger and event trigger set higher to 40 and 80 respectively. Setting the triggers so high means we cant resolve energies much lower than 5 MeV, thus the range we tested for was 5, 10, 15, 20, 30 and 40 MeV. In the same way way as the first calibration a few values were removed from the samples that were far out side the signal Gaussian, values below 60. One was removed from 10 MeV, 38 from 15 MeV, and 20 from 19 MeV.

mV	MeV	ADC count	$FWHM$
222	5	271	6.25
444	10	545	6.09
666	15	815	5.55
888	20	1085	5.50
1333	30	1622	6.42
1777	40	2029	6.16

Table 4.2: Calibration results of channel 1 with baseline trigger set at 40 and event trigger set at 80.

Comparing the results in table 4.1 and 4.2 it becomes apparent that raising the trigger level significanlty decreases the pulse heights, which can be seen as a decrease in the ADC counts at all energy levels. This decrease in pulse height is the direct result of the way in which the baseline is calculated. Since setting the baseline trigger higher means the baseline calculation will be halted later, more samples of the actual event will be stored in the moving average window.

Event samples have a higher value than samples outside of an event and they will therefore cause the calculated baseline to have a higher value than the actual basline for positive event pulses and a lower baseline for negative event pulses. This causes the pulse heigths to be lower because the pulse heights are calculated relatively to the baseline.

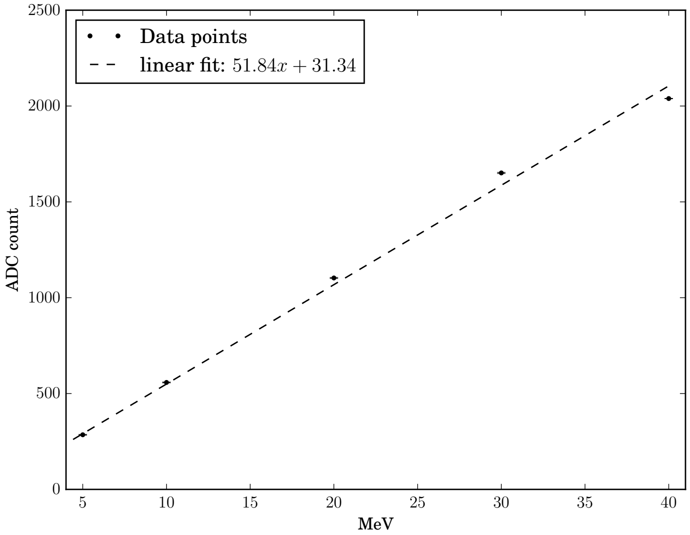


Figure 4.7: Linear regression of data from calibration of channel 1 with baseline trigger 15 and event trigger 40.

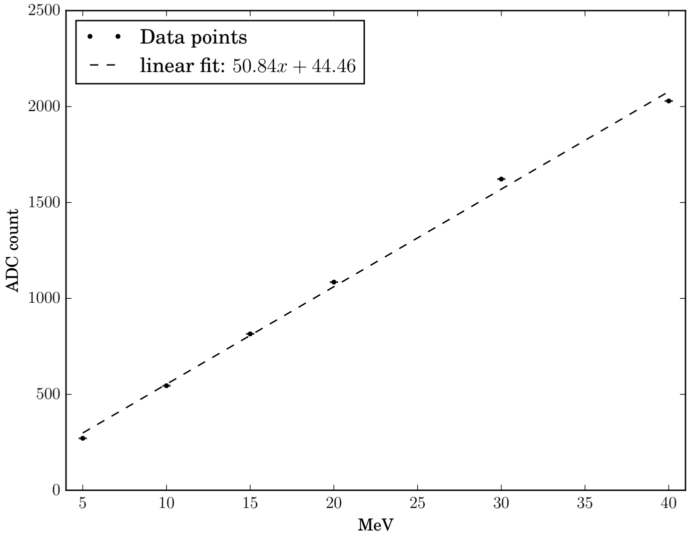


Figure 4.8: Linear regression of data from calibration of channel 1 with baseline trigger 40 and event trigger 80.

The last data point in the linear regression figures 4.7 and 4.8 is lower than what should be expected looking at the other data points in the set. This is most likely caused by the system going into saturation a around 40 MeV. Removing the last data point gives a better linear fit, and the linear fit $54.01x + 3.51$ from figure 4.10 was used for converting the experimental data from ADC values into energy measured in MeV. Solving for x gives the energy in MeV:

$$Energy = \frac{\mathrm{ADC}_{\mathrm{value}} - 3.51}{54.01}\ \mathrm{MeV} \tag{4.1}$$

The calibration results of channel 1 shown in table 4.2, gives an average FWHM of 5.99 which can be converted to MeV using equation 4.1 $Energy = \frac{5.99-3.51}{54.01}$ MeV = 0.046 MeV.

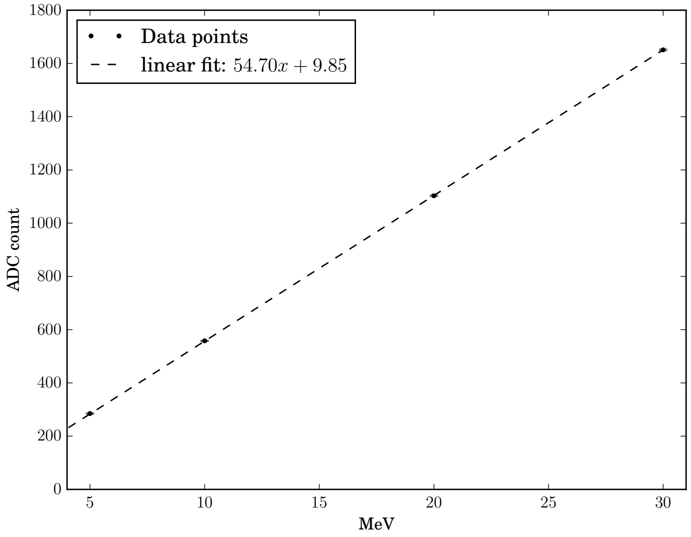


Figure 4.9: Linear regression of data from calibration of channel 1 with baseline trigger 15 and event trigger 40. The last data point has been removed because the ADC went into saturation 40 MeV.

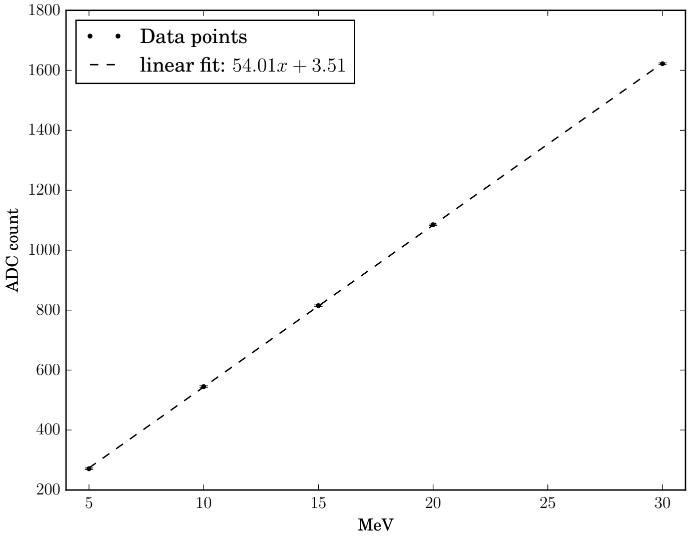


Figure 4.10: Linear regression of data from calibration of channel 1 with baseline trigger 40 and event trigger 80. The last data point has been removed because the ADC went into saturation at 40 MeV.

4.3.2 Channel 2 calibration

Channel 2, the channel without analog shaping was also calibrated and the calibration data has been summarized in table 4.3. The test capacitor used when calibration channel 2 was surface mount capacitor soldered to the input of the channel. This capacitor had to be removed before we could use the channel for experiments with real radiation sources. Unfortunately after the removal of the test capacitor, high frequency high amplitude noise was observed on the channel. Because of the noise, shown in figure 4.11 channel 2 was rendered useless and thus further analysis of the calibration data was not deemed necessary.

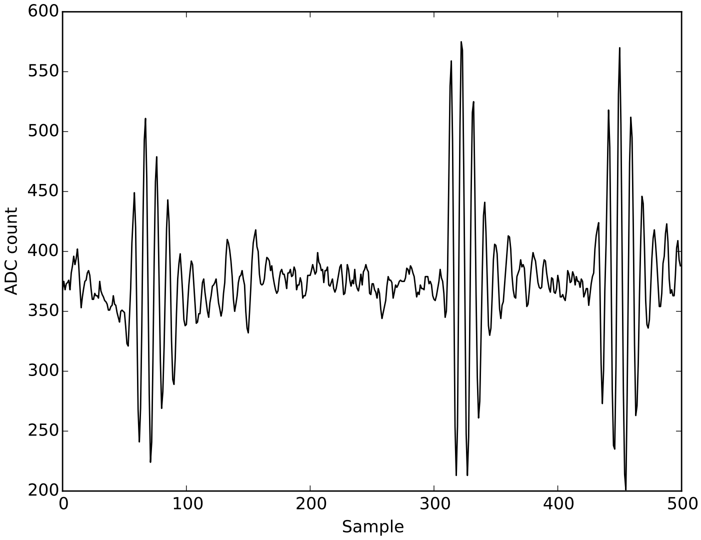


Figure 4.11: High frequency high amplitude noise observed on channel 2.

mV	MeV	ADC count
20	0.450	38
30	0.680	63
44	1	99
88	2	155
220	5	325
440	10	602
660	15	897
880	20	1169
1320	30	1169
1760	40	2047

Table 4.3: Calibration results of channel 2

Chapter 5

Experiments

A couple of experiments were run to verify that the system actually works with a real radiation source. The experiments were preformed with two separate sources, one containing americium-241 and the other californium-252. The americium-241 is an unsealed alpha emitting radiation source from Isotrak [6]. It is a stainless steel disc with diameter of 25 mm and a thickness of 0.5 mm. The radionuclides are deposited on the center of the disc with diameter of 7 mm. It decays virtually 100% by the emission of alpha particles, with 85.2% of the alpha particles having an energy of 5.486 MeV [6]. The chance of americium-241 decaying by spontaneous is only 3.6×10^{-10} % [27].

On the californium-252 source the radionuclides are deposited in the cavity of an aluminium container. Californium-252 undergoes spontaneous fiction in addition to alpha decay. The probability of alpha emission is however considerably higher than for spontaneous fission, branching fraction of spontaneous fission is 3.092 % and 96.908 % for alpha decay according to [28]. [3] gives similar numbers saying that a sample of 1 μ californium-252 will emit 1.92×10^7 alpha particles and undergo 6.14×10^5 spontaneous fissions per second which gives a probability of spontaneous fission of 3.198% and probability of 96.802 % of alpha decay. Conversion from ADC values to energy was accomplished using the equation 4.1 from the calibration. In order for the calibration to be as correct as possible, only data capture with a baseline trigger of 40 is presented here because changing the baseline trigger changes the pulse height. The Properties of both sources are summarized in table 5.1.

Figure 5.2 shows the experimental setup when using the americium-241 source. The washer is placed between the source and the aluminum plate is to protect the radionuclides that have been deposited onto the stainless steel shown as active area in 5.1. The aluminium plate has a thickness of 1 mm and the distance from the plate to the detector is 5 mm. Figure 5.3 shows the experimental setup using the californium-252 source. It uses the same aluminium plate and detector, but the source is slightly different in that the radionuclides are deposited in a cavity.

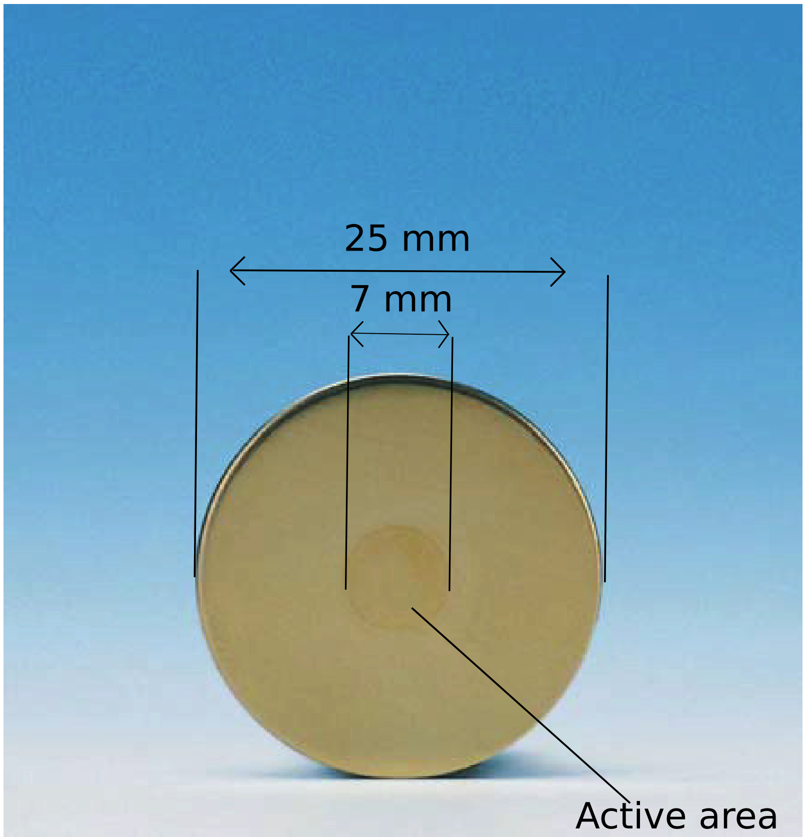


Figure 5.1: Americium-241 source from Isotrak [6].

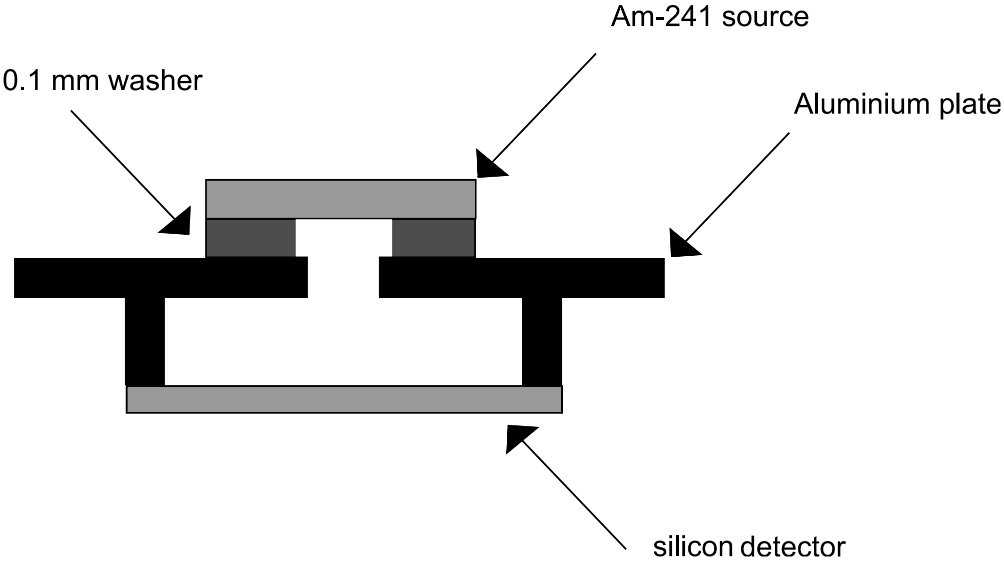


Figure 5.2: Experimental setup for the americium-241 source. Approximately 5 mm gap from aluminium plate to the detector.

Source	^{241}Am	^{252}Cf
Activity	5 kBq	3.7 kBq
Half-life [years]	433	2.65
Alpha particle	5.48574 ± 0.00012 MeV (85.2 %)	6.11824 MeV (84.2 %)
kinetic energy (% Branching)	5.44298 ± 0.00013 MeV (12.8 %)	6.07577 MeV (15.7 %)
Spontaneous fission	n/a	105.71 MeV
kinetic energy (% Brancing)	n/a	80.01 MeV
Spontaneous fission % Branch	3.092%	3.6×10^{-10}

Table 5.1: Properties of radiation sources americium-241 and californium-252 [3] [6] [2]. It is unknown when the ^{252}Cf was calibrated, and because of the short half life the activity might be significantly lower.

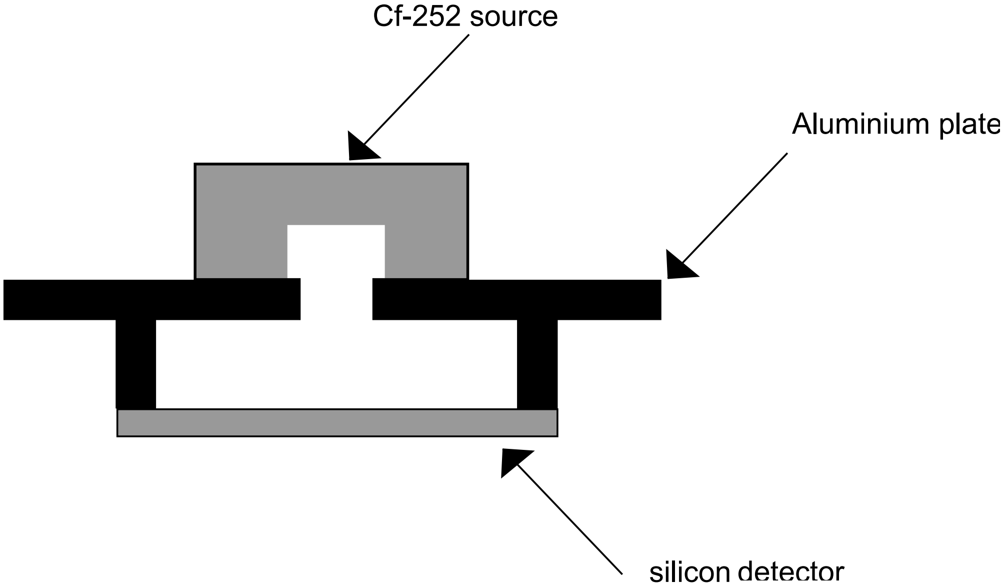


Figure 5.3: Experimental setup for the californium-252 source. Approximately 5 mm gap from aluminium plate to the detector. The depth of the cavity is a couple of millimeters.

Chapter 6

Results

This chapter presents the results from the testing with the two radiation sources; americium-241 and californium-252. The results of each of the two sources are presented in separate sections, each containing event data sampled both with and without DSP enabled and pulse height data. The data is presented in plots, as there are too many data points to tabulate it. Only data collected from channel 1 has been included as there was so much noise on channel 2 that no valuable data could be produced.

6.1 Detector Noise

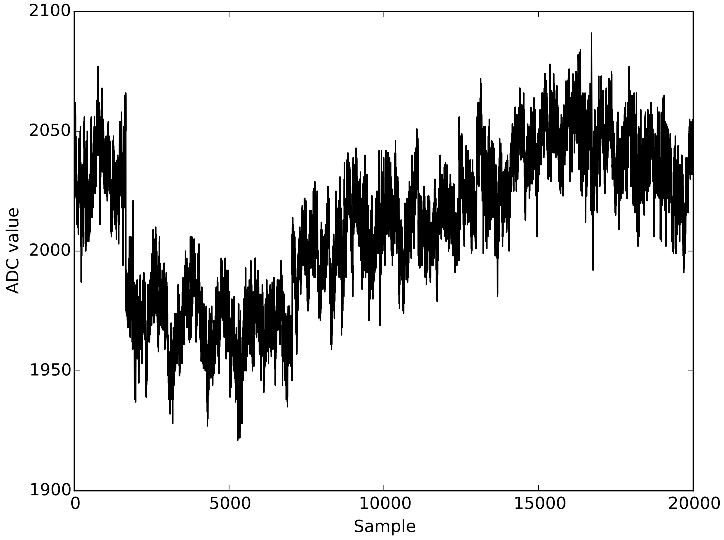


Figure 6.1: Sampling of the noise when the silicon PIN diode detector was attached.

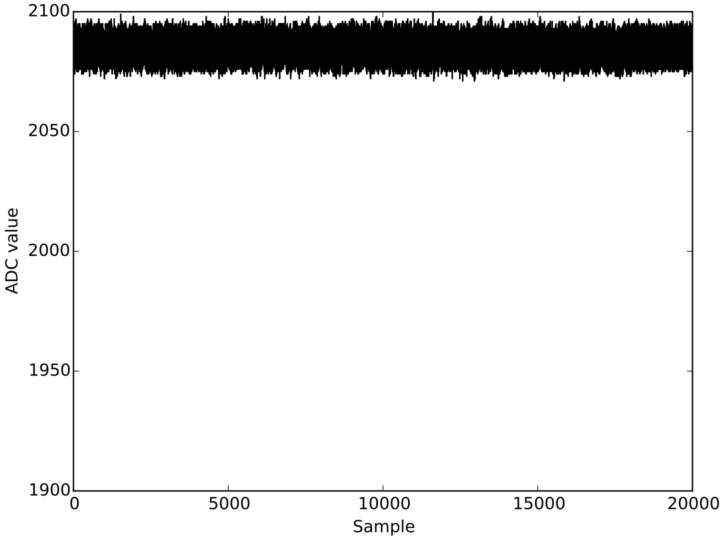


Figure 6.2: Sampling of the noise when the silicon PIN diode detector was detached. The y-axis were kept the same as in figure 6.1 in order to better illustrate the increase in noise when attaching the detector.

6.2 Americium-241

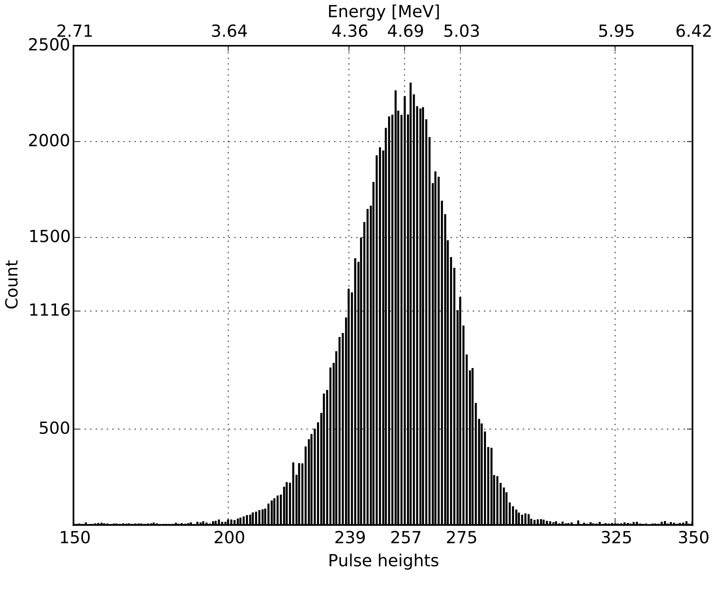


Figure 6.3: Pulse height histogram produced by the Americium-241. The histogram has the shape of a Gaussian distribution with a top at 257 and a FWHM of 36. The FWHM markers a drawn with dashed lines on the plot. The baseline trigger is set to 40. The Histogram is made up of 89718 data points.

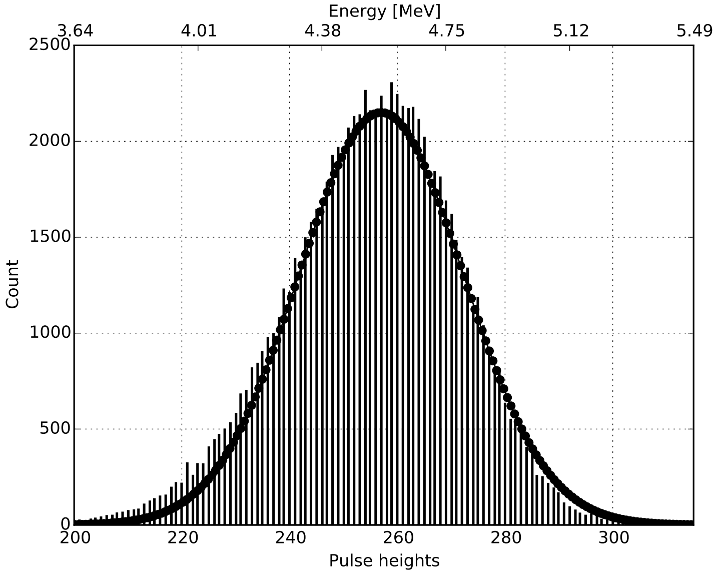


Figure 6.4: Same data as used in figure 6.3 but fitted to a normal distribution.

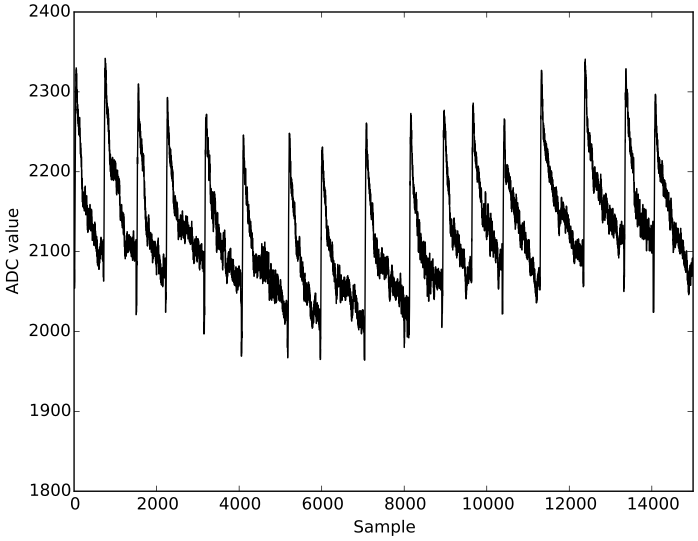


Figure 6.5: Samples taken of the pulses produced by the Americium-241 source when the baseline correction is disconnected. Baseline trigger set to 40.

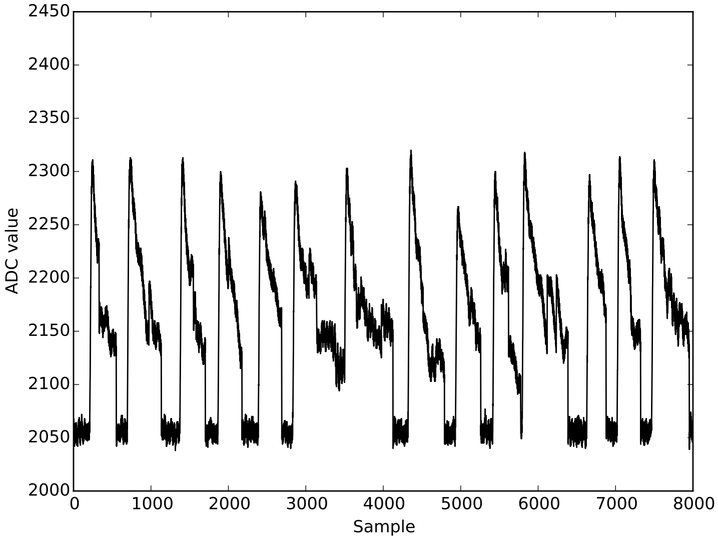


Figure 6.6: Samples taken of the pulses produced by the Americium-241 source when the baseline correction is connected. The padding is set higher than in figure 6.5 which is why there are more zero samples in between the event pulses. Note the pile up between samples 3000 and 4000. Baseline trigger set to 40.

6.3 Californium-252

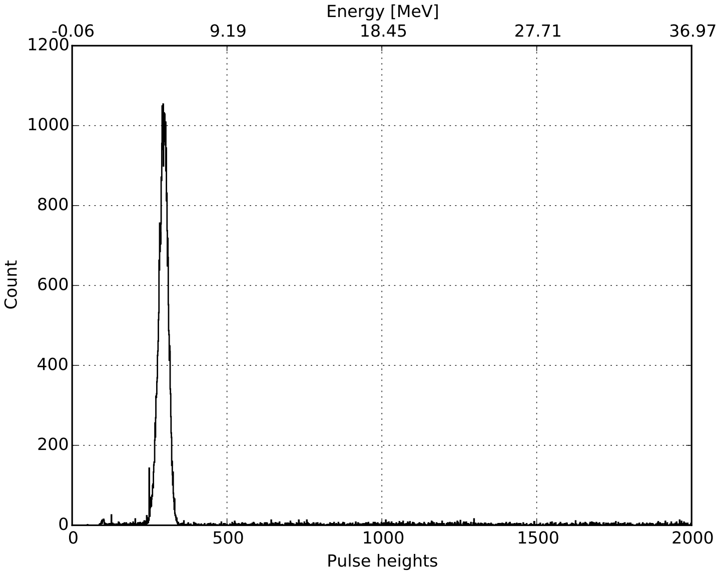


Figure 6.7: Pulse height histogram produced by ${}^{252}Cf$ using pulse heights 41787. The larges peak is produced by alpha particles, while the peaks above 6 MeV are produced by fission products.

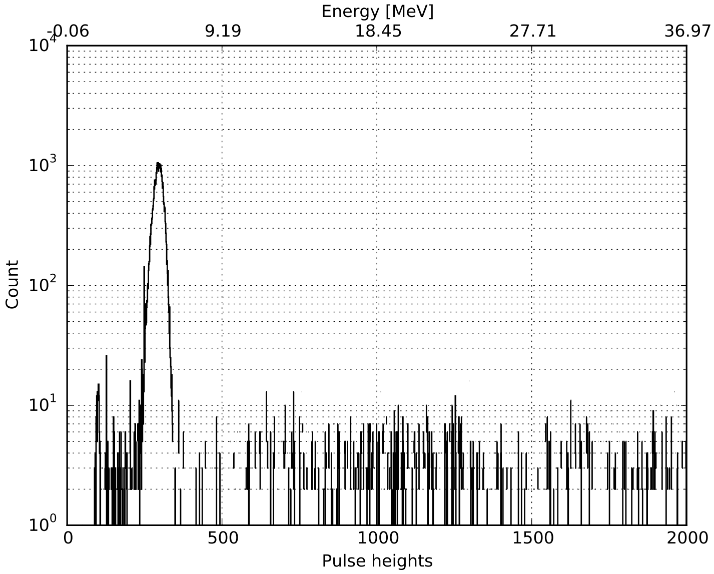


Figure 6.8: Pulse height histogram with the same data as in figure 6.7 but with a logarithmic Y-axis to better show the peak produced by the alpha particles.

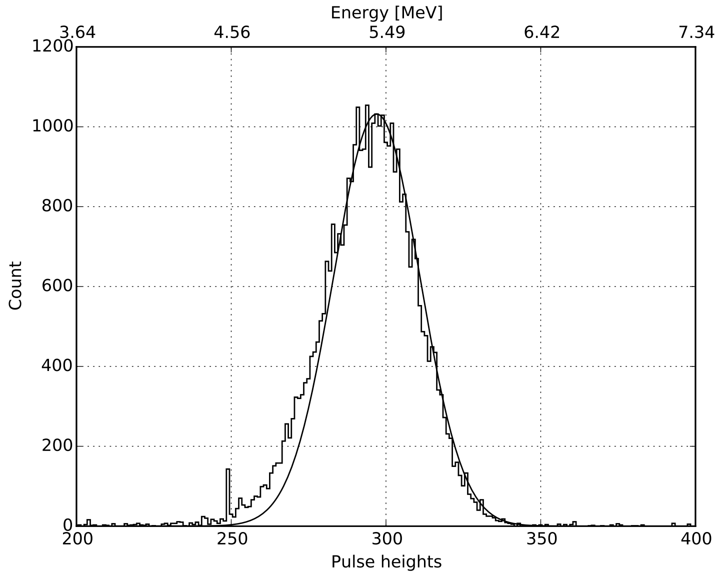


Figure 6.9: Same histogram as in 6.7 but zoomed in on histogram produced by alpha particles. A Gaussian fit is overlain the histogram.

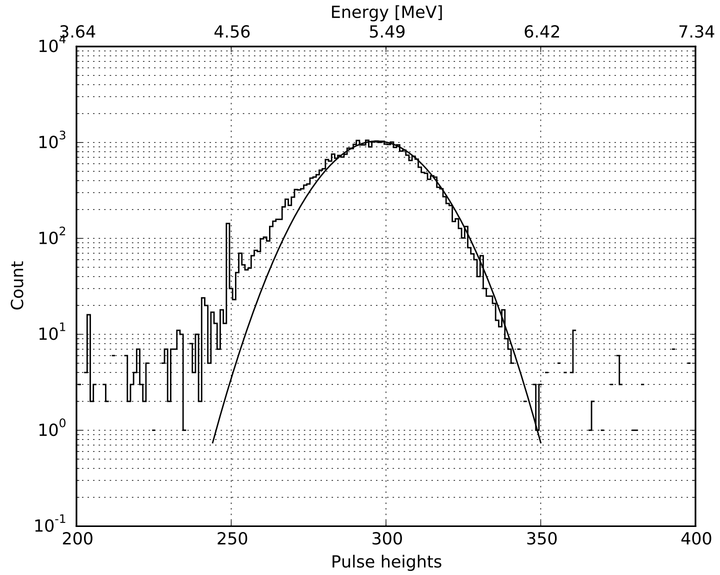


Figure 6.10: Zoomed in on the alpha particle histogram from figure 6.8. Gaussian fit is also included.

Source	Mean energy [MeV]	FWHM	FWHM in MeV	σ in ADC values	σ in MeV
^{241}Am	4.693390	36.00	0.602	15.27	0.218
^{252}Cf	5.433937	32.80	0.542	13.93	0.193

Table 6.1: Summary of histogram data.

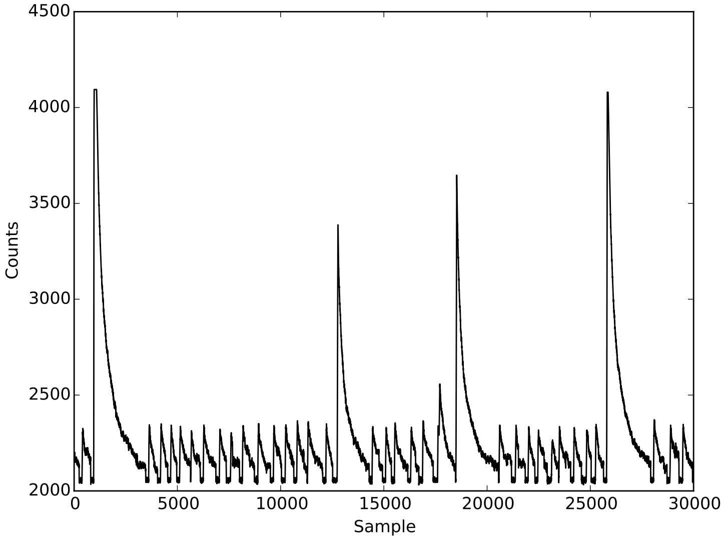


Figure 6.11: Pulses produced by the ^{252}Cf source. The small pulses are made by alpha particles and the larger pulses by fission products. This specific sampling had an unusually high concentration of fission products.

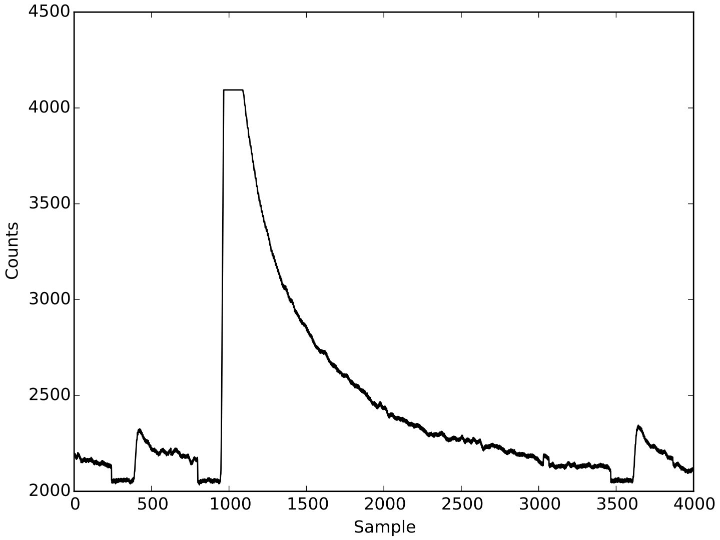


Figure 6.12: Zoomed in view of the left most fission product pulse in figure 6.11. It shows that pulses produce ADC values over 4096 it gets cut off.

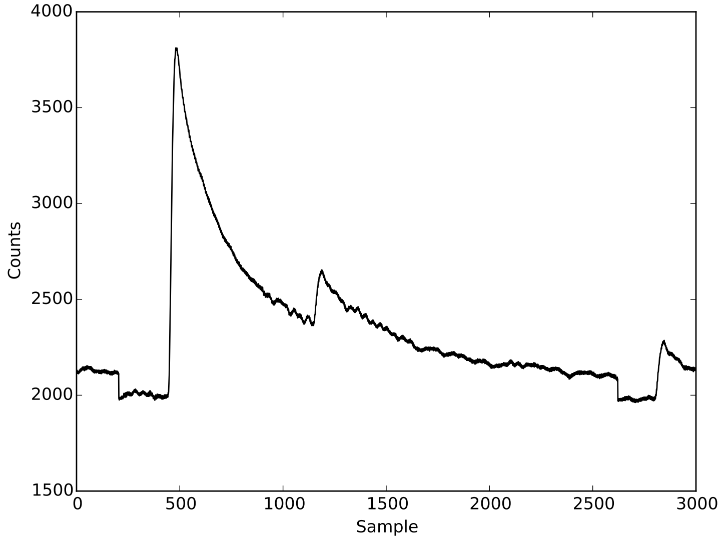


Figure 6.13: An alpha particle piled up on a fission product.

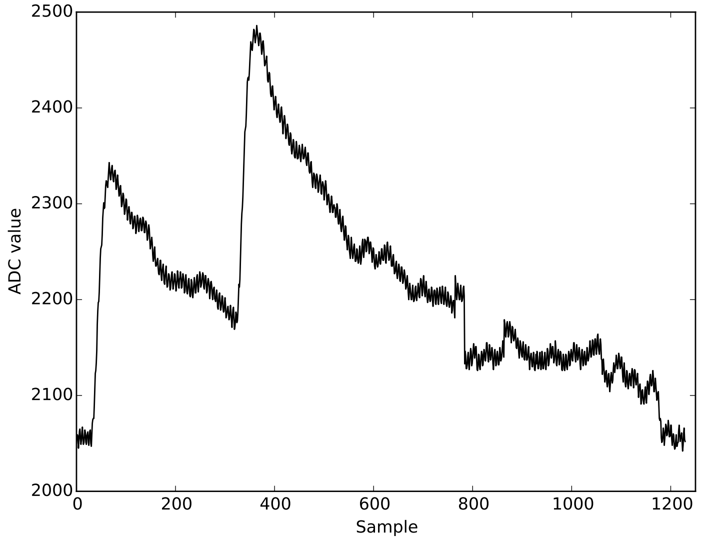


Figure 6.14: An alpha particle piled up on another alpha particle.

Chapter 7

Discussion and conclusion

7.1 Discussion of the experimental results

The experimental results are presented in two different types of plots. Histograms produced by event heights which gives us an idea about the energy of particles that hit the detector. The other plots the sampled events giving us an idea of how the DSP modules work. Information abut the histograms is summarized in the table 6.1.

7.1.1 Americium-241

At first glance the histograms of alpha particle pulse heights in figures 6.3 and 6.4 looks very different than figure 2.1. There is only one top in our results while figure 2.1 shows 5 distinct tops. The reason there is only one distinct top in our experimental data is that we have a relatively large FWHM and standard deviation. The difference between the middle peak and the lowest energy peak in figure 2.1 is only 0.097 MeV while the FWHM in our system is 0.602 MeV as listed in table 6.1. One of the causes for the large FWHM is electronic noise in the system. However the FWHM seen during calibration is only 0.046 MeV, and cannot account for all of the deviation. Another source of error is pile-ups. As explained in chapter to an shown in figure 2.3, pile-ups have a major effect on the width of the Gaussian distribution. Since the system does not implement any pile-up rejection on the digital signal processing, pile-ups such as the one seen in figure 6.6 will play a big part in the widening of the distribution. There is also an increase in the noise when connecting the detector which affects the histogram width, shown in figures 6.1 and 6.2. The most significant contributers to the detector noise according to [3] are:

1. Fluctuations in the bulk generated leakage current.
2. Fluctuations in the surface leakage current.
3. Noise associated with series resistance or poor electrical contacts to the detector.

These points should all be looked into in order to solve the noise problem.

Though the smaller peaks are not visible, the Gaussian fit shown in figure 6.4 shows that there is a slight bulge on the left side of the histogram. It is likely caused by the lower energy alpha particles that are more likely to happen than the higher energy particles, 12.7 % and 1.3 % branching percentage vs 0.12 % and 0.25 % for the higher energy particles seen in figure 2.1.

The mean alpha particle energy of 4.693 MeV is 0.793 MeV lower than the main peak in 2.1 and what is listed in table 5.1. The energy loss can be explained by alpha particle energy loss in air which according to [29] is approximately 1 MeV for 6 mm of air. The distance between the source and the detector needs to be more accurately measured to determine the correct energy loss by air. However However it would be preferable to run the experiment again with the detector in a vacuum in order to properly determine whether the system can be used for alpha particle spectroscopy.

Figure 6.5 shows event sampling where the baseline correction has been turned off but the zero cancellation is still enabled. The noise pictured in figure 6.1 can be seen causing the baseline to fluctuate. Figure 6.6 shows event sampling when the baseline correction has been turned on again demonstrating that it is capable of removing the fluctuations in the baseline without significantly affecting the pulse heights. What however becomes apparent is that the baseline correction cuts off the tail of the events. This is because the baseline correction is started again as soon as a sample goes below the baseline trigger level, thus removing the tail. In future revisions of the system the baseline correction should not be restarted before the end of an event tail.

7.1.2 Californium-252

Similarly to the americium-241 data the californium-252 data is presented as pulse height histograms in figure 6.7 and with a logarithmic y-axis in figure 6.8. The logarithmic y-axis helps highlight the pulse with energies above 6 MeV. These high energies are most likely produced by fission products. However since there is less than 10 pulse height samples for almost all of the energies above 6 MeV, there is not sufficient data to draw any conclusions about the fission products, other than that they are in fact there. Even if the system was left to sample for days in order to get enough statistical data on the fission products, it would not be possible to see a similar plot as the one in figure 2.2 because the two tops are produced by fission products with energies of 80.1 MeV and 105.71 MeV and our system saturates at approximately 40 MeV. The saturation results in clipping of events with energies above 40 MeV, this can be seen in figure 6.12, which shows that the top of a fission product event has been clipped.

Figures 6.9 and 6.10 show a closer view of the man Gaussian from figures 6.7 and 6.8 respectively with Gaussian fits overlain. The Guassian fit shows that the histogram is wider on the left side, this might be caused by the lower energies alpha particles. However since the difference between the two alpha particle energies released by californium-252 is only about 30 keV, as seen in table 5.1, it is unlikely to be the cause. The reason is more likely to be pile-ups on the undershoot tail of events, see figure 2.3

Pile-ups were observed on the event pulses shown in figures 6.13 and 6.14. In the former, the pile-up will not have an effect on the pulse height histogram as the top produced by the fission product is much higher than the piled up alpha. The height of the alpha particle will not be recorded as it happened within the event of the fission product which does not end until after sample 2500. In the latter figure the pile up has a much more detrimental effect. Instead of being recorded as two pulses each with a energy of approximately 4.9 MeV, they will be recorded as one pulse with energy of 8.6 MeV. Thus some of the pulses from figure 6.8 with energies above 6 MeV are likely to be caused by pile-up effects. This really illustrates the need to implement either a pile-up cancellation module to remove signals with pile up or better a pile-up correction module that can separate the two pulses.

7.2 Conclusion

A working asynchronous FIFO was developed as a way of storing data output from the ADC. The FIFO design was verified using an automatic VHDL testbench and the Bitvis Utility Library. Event triggering baseline, calculation and correction, zero suppression, and pulse height modules were developed and verified automatic VHDL testbenches and the Bitvis Utility Library. However an important module that was not implemented is a pile-up cancellation or correction module. Such a module would significantly improve the energy resolution of the system by reducing the FWHM. Sending data from FPGA fabric to the user was made simple by having an on chip ARM processor, capable of communicating with and receiving data from the FPGA using an Avalon bridge. The data could then be downloaded of the ARM through Ethernet. For communicating with front end hardware the I2C protocol was chosen for its simplicity in use and expandability. Finally the system was characterized using ^{252}cf and ^{241}am. The observed alpha particle energies, 4.69 ± 0.301 for americium and 5.43 ± 0.271, are somewhat lower than expected but that can be explained by energy loss to the air. Thus the main goals listed in the introduction were completed.

7.3 Future work

Future iterations should include either a pile-up corrector or cancellation module in order to reduce the FWHM. A pile-up cancellation module would be simplest and should probably be developed first, then an eventual pile-up corrector could build on the pile-up cancellation module. The problem with the baseline correcting module should be looked into. One suggestion would be to only re enable the baseline calculation a number of samples after the event has ended. The specific number of samples to wait should by finding the average event lengths. Even more important for the baseline correction is to find a way to avoid the pulse height attenuation that occurs when setting high baseline triggers. One way of doing this could be to store earlier baseline data and restore that stored baseline when there is an event. Another way could be to find a better way of triggering. A larger moving average window would not stop the attenuation completely, but it would attenuate the pulse heights less, because each sample has a smaller influence on the total average than if a small moving average window is used.

The number of clock cycles used in the reset state of the baseline calculation should be reduced from 5000 to the size of the moving average window, which is 16. There is no point in using more than 16 clock cycles, as that will fill the moving average window with new samples.

The possibility of increasing the clock rate of the read out system in order to enable continuous readout with no dead time should be looked into. Another change that could facilitate a continuous readout system would be to implement interrupt signals for the FIFO status flags. This would decrease the time between data becoming available and it being read out, thus decreasing the probability of the FIFO becoming full.

The Avalon bus that is used for communication between the FPGA and the ARM processor has not been simulated. This needs to be done if 100 % verification of the system is desired.

Bibliography

[1] R.N. Chanda and R.A. Deal. Catalogue of semiconductor alpha-particle spectra. Technical report, Idaho Nuclear Corp.

[2] Stanley L. Whetstone. Coincident time-of-flight measurements of the velocities of cf^{252} fission fragments. *Phys. Rev.*, 131:1232–1243, Aug 1963.

[3] Glenn F. Knoll. *Radiation Detection and Measurement.* WILEY, fourth edition, 2010.

[4] R Esteve-Bosch, A Jiménez de Parga, B Mota, and L Musa. The ALTRO Chip: A 16-channel A/D Converter and Digital Processor for Gas Detectors. *IEEE Trans. Nucl. Sci.*, 50(6):2460–69, 2003.

[5] Pong P. Chu. *RTL Hardware Design Using VHDL.* WILEY-INTERSCIENCE, first edition, 2006.

[6] Alpha spectrometry sources. `http://www.gammadata.se/assets/Uploads/04-alpha-spectrometry-sources2.pdf`. Accessed: 2015-05-27.

[7] Maris Tali. Front-end signal conditioning for a silicon pin detector. Master thesis, University of Oslo, June 2015.

[8] Peter Jackson Geir Anton Johansen. *Radioisotope Gauges for Industrial Process Measurements.* WILEY, 2004.

[9] Fumihiko Suekane. Neutrinos and weak interactions in the standard model. In *Neutrino Oscillations*, volume 898 of *Lecture Notes in Physics*, pages 7–29. Springer Japan, 2015.

[10] J. Ahrens, E. Andrés, X. Bai, G. Barouch, S. W. Barwick, R. C. Bay, T. Becka, K.-H. Becker, D. Bertrand, F. Binon, A. Biron, J. Booth, O. Botner, A. Bouchta, O. Bouhali, M. M. Boyce, S. Carius, A. Chen, D. Chirkin, J. Conrad, J. Cooley, C. G. S. Costa, D. F. Cowen, E. Dalberg, C. De Clercq, T. DeYoung, P. Desiati, J.-P. Dewulf, P. Doksus, J. Edsjö, P. Ekström, T. Feser, J.-M. Frère, T. K. Gaisser, M. Gaug, A. Goldschmidt, A. Hallgren, F. Halzen, K. Hanson, R. Hardtke, T. Hauschildt, M. Hellwig, H. Heukenkamp, G. C. Hill, P. O. Hulth, S. Hundertmark, J. Jacobsen, A. Karle, J. Kim, B. Koci, L. Köpke, M. Kowalski, J. I. Lamoureux, H. Leich, M. Leuthold, P. Lindahl, I. Liubarsky, P. Loaiza,

D. M. Lowder, J. Madsen, P. Marciniewski, H. S. Matis, C. P. McParland, T. C. Miller, Y. Minaeva, P. Miočinović, P. C. Mock, R. Morse, T. Neunhöffer, P. Niessen, D. R. Nygren, H. Ögelman, Ph. Olbrechts, C. Pérez de los Heros, A. C. Pohl, R. Porrata, P. B. Price, G. T. Przybylski, K. Rawlins, C. Reed, W. Rhode, M. Ribordy, S. Richter, J. Rodríguez Martino, P. Romenesko, D. Ross, H.-G. Sander, T. Schmidt, D. Schneider, R. Schwarz, A. Silvestri, M. Solarz, G. M. Spiczak, C. Spiering, N. Starinsky, D. Steele, P. Steffen, R. G. Stokstad, O. Streicher, P. Sudhoff, K.-H. Sulanke, I. Taboada, L. Thollander, T. Thon, S. Tilav, M. Vander Donckt, C. Walck, C. Weinheimer, C. H. Wiebusch, C. Wiedeman, R. Wischnewski, H. Wissing, K. Woschnagg, W. Wu, G. Yodh, and S. Young. Observation of high energy atmospheric neutrinos with the antarctic muon and neutrino detector array. *Phys. Rev. D*, 66:012005, Jul 2002.

[11] Y. Ashie, J. Hosaka, K. Ishihara, Y. Itow, J. Kameda, Y. Koshio, A. Minamino, C. Mitsuda, M. Miura, S. Moriyama, M. Nakahata, T. Namba, R. Nambu, Y. Obayashi, M. Shiozawa, Y. Suzuki, Y. Takeuchi, K. Taki, S. Yamada, M. Ishitsuka, T. Kajita, K. Kaneyuki, S. Nakayama, A. Okada, K. Okumura, C. Saji, Y. Takenaga, S. T. Clark, S. Desai, E. Kearns, S. Likhoded, J. L. Stone, L. R. Sulak, W. Wang, M. Goldhaber, D. Casper, J. P. Cravens, W. Gajewski, W. R. Kropp, D. W. Liu, S. Mine, M. B. Smy, H. W. Sobel, C. W. Sterner, M. R. Vagins, K. S. Ganezer, J. Hill, W. E. Keig, J. S. Jang, J. Y. Kim, I. T. Lim, K. Scholberg, C. W. Walter, R. W. Ellsworth, S. Tasaka, G. Guillian, A. Kibayashi, J. G. Learned, S. Matsuno, D. Takemori, M. D. Messier, Y. Hayato, A. K. Ichikawa, T. Ishida, T. Ishii, T. Iwashita, T. Kobayashi, T. Maruyama, K. Nakamura, K. Nitta, Y. Oyama, M. Sakuda, Y. Totsuka, A. T. Suzuki, M. Hasegawa, K. Hayashi, I. Kato, H. Maesaka, T. Morita, T. Nakaya, K. Nishikawa, T. Sasaki, S. Ueda, S. Yamamoto, T. J. Haines, S. Dazeley, S. Hatakeyama, R. Svoboda, E. Blaufuss, J. A. Goodman, G. W. Sullivan, D. Turcan, A. Habig, Y. Fukuda, C. K. Jung, T. Kato, K. Kobayashi, M. Malek, C. Mauger, C. McGrew, A. Sarrat, E. Sharkey, C. Yanagisawa, T. Toshito, K. Miyano, N. Tamura, J. Ishii, Y. Kuno, M. Yoshida, S. B. Kim, J. Yoo, H. Okazawa, T. Ishizuka, Y. Choi, H. K. Seo, Y. Gando, T. Hasegawa, K. Inoue, J. Shirai, A. Suzuki, M. Koshiba, Y. Nakajima, K. Nishijima, T. Harada, H. Ishino, Y. Watanabe, D. Kielczewska, J. Zalipska, H. G. Berns, R. Gran, K. K. Shiraishi, A. Stachyra, K. Washburn, and R. J. Wilkes. Measurement of atmospheric neutrino oscillation parameters by super-kamiokande i. *Phys. Rev. D*, 71:112005, Jun 2005.

[12] CERN - EP/ED. *ALICE TPC Readout Chip User Manual*, 0.2 edition, June 2002.

[13] B Mota, L Musa, R Estève, and A Jiménez de Parga. Digital implementation of a tail cancellation filter for the time projection chamber of the ALICE experiment. 2000.

[14] CAEN. *CAEN Digital Pulse Height Analyser - a digital approach to Radiation Spectroscopy*, September 2011.

[15] Soo Hyun Byun. Radioisotopes and radiation methodology chapter 6 pulse processing. Lecture Notes, 2014.

[16] Linear Technology. *LTC1393 - Differential 4-Channel Analog Multiplexer with SMBus Interface.*

[17] Linear Technology. *LTC2283 - Dual 12-Bit, 125Msps Low Power 3V ADC*, 2006.

[18] H.J. Barnaby. Total-ionizing-dose effects in modern cmos technologies. *Nuclear Science, IEEE Transactions on*, 53(6):3103–3121, Dec 2006.

[19] Linear Technology. *LTC2242-10 10-Bit, 250Msps ADC*, 2006. Rev D.

[20] R.G. Bozomitu, V. Cehan, and C. Barabasa. A vlsi implementation of a 3gb/s lvds transceiver in cmos technology. In *Design and Technology of Electronics Packages, (SIITME) 2009 15th International Symposium for*, pages 69–74, Sept 2009.

[21] Ieee standard for low-voltage differential signals (lvds) for scalable coherent interface (sci). *IEEE Std 1596.3-1996*, pages i–, 1996.

[22] NXP Semiconductors. *I^{2}C-bus specification and user manual*, sixth edition, April 2014.

[23] Kinglong Mee. nfs-utils patch to fix problem with make file reported by eino juhani oltedal. http://git.linux-nfs.org/?p=steved/nfs-utils.git;a=commitdiff;h=3e28fe73af023bf42e0d654146bfb9314b4d8438. Accessed: 2015-05-07.

[24] HAMEG Instruments. *25MHz Arbitrary Function Generator HMF2525*, 2012.

[25] Chris Spear and Greg Tumbush. *System Verilog for Verification.* Springer, third edition, 2012.

[26] Bitvis Utility Library description. `http://bitvis.no/products/bitvis-utility-library/`. Accessed: 2015-05-26.

[27] V. P. Chechev and N. K. Kuzmenko. Lne – lnhb/cea – table de radionucléides. `www.nucleide.org/DDEP_WG/Nuclides/Am-241_tables.pdf`. Accessed: 2015-05-27.

[28] M.R. Schmorak. Www table of radioactive isotopes. http://ie.lbl.gov/toi/nuclide.asp?iZA=980252. Accessed: 2015-05-27.

[29] K.N. Yu, C.W.Y. Yip, D. Nikezic, J.P.Y. Ho, and V.S.Y. Koo. Comparison among alpha-particle energy losses in air obtained from data of srim, {ICRU} and experiments. *Applied Radiation and Isotopes*, 59(5–6):363 – 366, 2003.

Appendices

Appendix A

Final FPGA resource usage

```
Flow Status    Successful - Fri May 15 11:50:12 2015
Quartus II 64-Bit Version    14.0.0 Build 200 06/17/2014 SJ Web Edition
Revision Name    SoCKit\_Linux\_FB
Top-level Entity Name    SoCKit\_Linux\_FB
Family    Cyclone V
Device    5CSXFC6D6F31C6
Timing Models    Final
Logic utilization (in ALMs)    6,313 / 41,910 ( 15 % )
Total registers    9262
Total pins    239 / 499 ( 48 % )
Total virtual pins    0
Total block memory bits    3,696,734 / 5,662,720 ( 65 % )
Total DSP Blocks    0 / 112 ( 0 % )
Total HSSI RX PCSs    0 / 9 ( 0 % )
Total HSSI PMA RX Deserializers    0 / 9 ( 0 % )
Total HSSI TX PCSs    0 / 9 ( 0 % )
Total HSSI PMA TX Serializers    0 / 9 ( 0 % )
Total PLLs    3 / 15 ( 20 % )
Total DLLs    1 / 4 ( 25 % )
```