CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

A Signal Dependent Compression Scheme for Radio Astronomy

Master's thesis in Embedded Electronic System Design

Oskar Andersson
Eric Bergdahl

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

A Signal Dependent Compression Scheme for Radio Astronomy

Oskar Andersson
Eric Bergdahl

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

A Signal Dependent Compression Scheme
for Radio Astronomy
Oskar Andersson
Eric Bergdahl

Supervisor: Gary Hovey, Department of Space, Earth and Environment
Co-supervisor: Mugundhan Vijayaraghavan, Department of Space, Earth and Environment
Examiner: Per Larsson-Edefors, Department of Computer Science and Engineering

Master's Thesis 2023
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2023

A Signal Dependent Compression Scheme
for Radio Astronomy
Oskar Andersson
Eric Bergdahl
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Radio astronomy requires the capture of cosmic radiation. The signals of interest can span large bandwidths and high dynamic ranges, and are often contaminated by powerful man-made radio frequency interference (RFI) from mobile communications, radar equipment and much more. Capturing such signals requires receiver systems that produce high bit level samples at tremendous data rates, much of which is wasted whenever the input signals are free of RFI and other transients. This thesis has explored methods of compressing the output streams of observatory data recorders, by way of varying the bit level of samples depending on the momentary characteristics of the input signal, in order to reduce the effective output data rate and in turn more effectively utilize data storage equipment and transmission links. Simulations performed on exemplary, pre-recorded input signals consisting of noise contaminated by aircraft RFI have shown a potential data rate reduction of up to 70% compared to receiver systems used by some contemporary radio astronomy receiver systems.

Keywords: radio astronomy, compression, FPGA, adaptive, streaming

Acknowledgements

We would like to give our sincerest thanks to our supervisors Gary Hovey and Mugundhan Vijayaraghavan who have given much needed guidance during this thesis, as well as all the other people of Onsala Space Observatory for welcoming us and granting us the opportunity to conduct our thesis work there. We would also like to thank our examiner Per Larsson-Edefors for all his insightful feedback and support.

Oskar Andersson, Eric Bergdahl, Gothenburg, August 2023

Contents

1

Introduction

This chapter begins by presenting some background information on the matter of radio astronomy, examples of existing systems and a motivation for the project. It is followed by a description of the project goals, and is concluded with an outline of the rest of the thesis report.

1.1 Background

The study of radio frequency (RF) signals, both man-made and naturally occurring, requires electromagnetic (EM) waves to be captured by antennas, and then sampled and quantized by a digitizer for further processing and storage. One design parameter of a digitizer is the quantization depth; a larger bit depth enables better signal-to-noise ratio (SNR) and dynamic range. In many cases, receiver systems are designed to handle the worst-case scenario signals in terms of their SNR and required dynamic range, but such signals may occur very rarely. In such cases, more bits are thus often used to represent measurement data than is necessary, which leads to a waste of computational resources, digital storage and communication links, as unnecessarily large amounts of data need to be processed, stored and moved.

In the case of radio astronomy, much of the cosmic radiation of interest is weak enough to be below the inherent noise of the receiver systems. There also exists more powerful signals that the receivers have to cope with; examples of interesting signals are fast radio bursts (FRB) [1] [2] and solar transients [3]. These require receivers with high bandwidth and dynamic range, resulting in data volumes measured in several billions of samples per second. The probability of these phenomena occurring is very low however, so recording at full quantization depth at all times is wasteful. At the same time, there is radio frequency interference (RFI), which includes essentially all non-cosmic RF signals. Much of this is man-made, and consists of emissions from cell towers, satellites, WLAN transmitters, radars on aircraft and ships, among many others. RFI from such sources has greatly increased since the early days of radio astronomy research in the middle of the 20th century, especially with the proliferation of cellphones and wireless internet, and some of these signals can be as powerful as 100 dB above the receiver noise level. The need to accommodate powerful transient signals, and to avoid RFI-related issues such as receiver saturation, has necessitated systems with larger quantization bit-depths, which has in turn led to greatly increased receiver output data rates.

This thesis explores the idea of compressing the output data produced by a radio

astronomy receiver system. The concept of data compression itself is not new; it has its roots in the mid-20th century with the development of information theory [4], but it has not been applied to any great extent in radio astronomy. While this thesis targets the radio astronomy domain, a compression scheme such as the one that will be presented could potentially apply to any scenario in which the aim is to capture radio signals.

1.2 Existing radio astronomy receiver systems

Interferometry, or more specifically aperture synthesis, is a technique used by observatories in data capture. Through the mixing of signals output by multiple separate antenna dishes, a synthetic aperture can be produced, the size of which is proportional to the distance between the constituent antennas. This is beneficial as the larger the antenna aperture is, the greater the effective angular resolution is, leading to higher resolution images.

Two examples of currently existing radio astronomy receivers are the square-kilometre array observatory (SKAO) and the European very long baseline interferometry (VLBI) network (EVN), the data flows of which are shown in Figure 1.1.

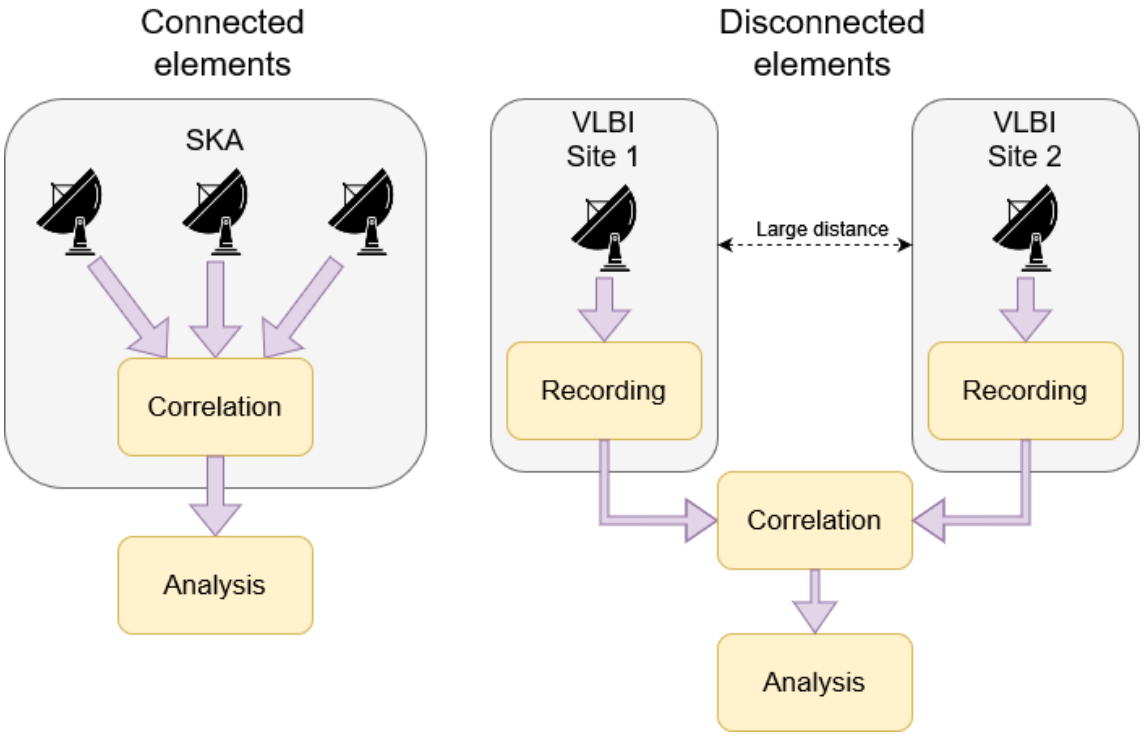


Figure 1.1: Data flow of SKA and VLBI systems

The SKAO consists of two branches, one in Australia and one in South Africa. The South African branch, SKA1-Mid, is designed to capture signals in the 350 MHz - 15.4 GHz range. Its synthetic aperture consists of 197 antennas, each of which produces 4 or 12 bit samples [5] at a data rate of up to 95.04 Gb/s. The large number of bits (12) allows the array to cope with the strong man-made RFI that

pollutes the frequency band it operates in, but the downside is the massive output data rate. However, as the most distant antennas in the array are located within 150 km of each other, all the data streams can be correlated on-site, thus reducing the final output data rate.

The EVN is a VLBI network which consists of antennas that are located across the Earth, in Europe, China, South Africa and Puerto Rico. The vast distances that separate the constituent antennas result in an extremely high image resolution, but the downside is that the many data streams cannot be immediately correlated such as in the SKAO case, but they need to first be recorded locally and then transmitted for further processing elsewhere. As recording is expensive, the data rates need to be limited. The OSO, which is a part of the EVN, utilizes the DBBC2 recorder, which is set up to output 512 Mb/s of 2 bit samples. This low bit level makes RFI problematic as it can easily saturate the receivers, forcing the affected samples to be discarded.

The SKA1-Mid is an example of a high bit-depth signal-capturing system, while that of the OSO is an example of a low bit-depth signal-capturing system. The drawbacks of each of the systems are evident; the former produces output data at a very large rate, while the latter is incapable of coping with powerful transient signals as these invariably saturate the 2-bit signals produced by its quantizers.

1.3 Project goal

The purpose of this thesis is to develop a digital signal processing scheme that combines the benefits of the two existing signal capturing systems described above while minimizing their drawbacks. Specifically, the aim is to develop a dynamic system that adapts the quantization depth to optimally digitize the received signal and achieve a good balance between output data rate and signal fidelity. Effectively this will compress the signal as much as possible, while maintaining a high enough dynamic range to allow for the reliable capture of powerful transients. This increases the efficiency by reducing the effective data rate for processing, storage and communication links. Specific goals include:

- Develop a digital signal processing scheme capable of dynamically adapting the quantization depth of a digitized signal based on the present characteristics of a captured signal at any given moment.
- Simulate the developed processing scheme using some captured RFI signals as input, measure the results and compare data usage and, if possible, average quantization error with the DBBC2 and SKA recorders.
- Implement the processing scheme using off-the-shelf hardware such as field-programmable gate array (FPGA) boards or graphical processing units (GPU) such that it can be integrated into the existing systems at OSO, and run tests on live data. Furthermore, it should be as agnostic as possible to the rest of the signal capturing system such that it would theoretically be relatively easy to integrate into other systems.

- Measure and compare data usage and average error size of the hardware implementation with the DBBC2 at OSO.

1.4 Ethical considerations

The subject matter of this thesis has a very limited, if any, effect on groups of people or society at large. However, since the primary aim of the thesis is to design a processing scheme to make it more economical to store large amounts of captured radio signals, it could be used in fields other than radio astronomy research. As the thesis is concerned with the capture and processing of radio signals, an ethical question that could conceivably be raised is the matter of eavesdropping on radio communications. There exists the potential of, inadvertently or not, capturing private or sensitive communiques. We feel however, that as free as the RF space is, there will be little risk that we could capture anything that could present any serious danger to individuals or society at large, as any truly confidential messages would presumably be transmitted either encrypted, or through different means entirely. Furthermore, there is no plan to perform any deeper data analysis beyond the minimum required to facilitate effective storage and processing.

1.5 Thesis outline

This thesis report is outlined as follows: Chapter 2 presents some technical background around the core concepts upon which this project is based and Chapter 3 describes our work methodology. Chapter 4 describes the design and implementation of the signal processing system that has been developed during the course of the project and Chapter 5 presents test results of testing this design, both in software simulations and in live radio receiver hardware. Finally, Chapter 6 discusses the results and Chapter 7 concludes the report.

2

Technical background

This chapter presents the fundamental concepts upon which this project relies. It begins by describing the nature of the cosmic signals that are of interest in radio astronomy, as well as the interference that pollutes the signal space. Then it presents a few basic principles of signal capture and processing, which is followed by a description of the main digitizer board upon which the algorithm is implemented. Finally, the chapter concludes with an overview of the entire end-to-end hardware chain within which the digitizer, and our algorithm, will be integrated.

2.1 Typical input signal

Commonly, cosmic radiation sources as seen from Earth are extremely weak, and are characterized by their spectral flux density which is measured in Janskys. One Jansky is equivalent to 10^{-26} W·m^{-2} · Hz^{-1} (Watts per square meter per Hertz). Such weak signals need to be amplified before they can be processed further, which is achieved using low-noise amplifiers (LNA) operating at temperatures of tens of Kelvin. Still, the interesting signals are often overshadowed by the inherent noise of the LNAs, which results in the digitized signal looking very much like white, Gaussian noise. Techniques exist to extract the interesting signals from this noise, such as correlation of multiple signal streams, but that is beyond the scope of this thesis. Figure 2.1 shows a chunk of a digitized captured signal. The system noise can be seen as the low-voltage regions. The large spikes are man-made RFI.

RFI refers to unwanted EM emissions captured by receiver antennas. In the case of radio astronomy this includes all emissions other than the cosmic signal of interest. As radio astronomy research is concerned with EM radiation across a wide frequency spectrum, from about 10 MHz to 30 GHz, RFI is a serious concern to which much consideration must be given. Examples of common sources of RFI are, in the case of man-made sources, satellite communications, WLAN systems, radar systems, as well as emissions from power lines or other electrical equipment. Natural, terrestrial events such as electrical storms also cause RFI within the above mentioned band [6].

2.2 Analog-to-digital conversion

In order to capture EM radiation signals for study, receiver antennas convert such radiation into analog AC voltage, which in turn is digitized by an analog-to-digital

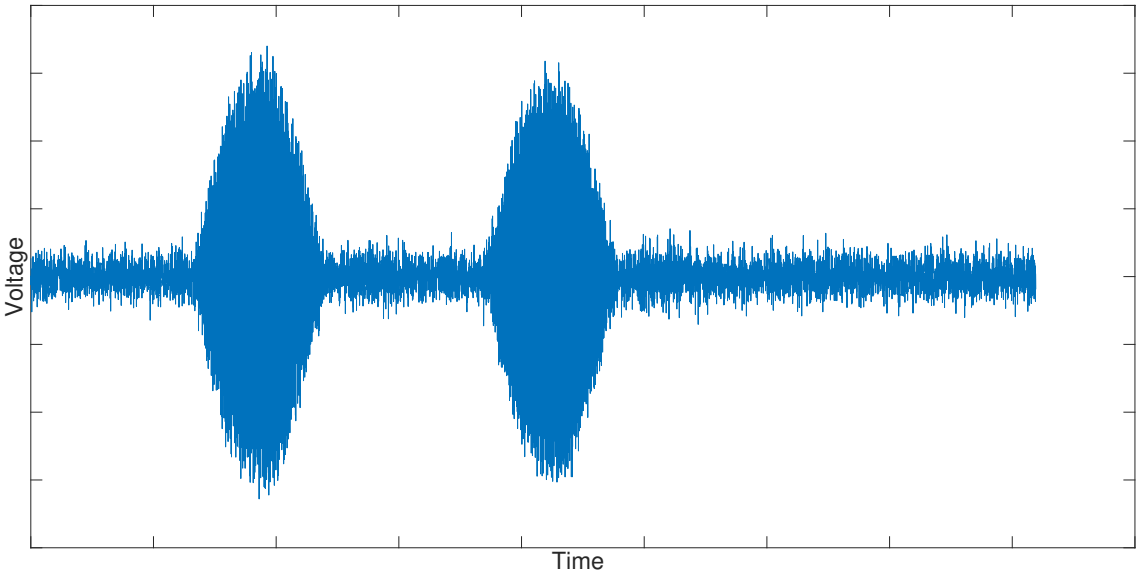


Figure 2.1: Typical input signal contaminated by pulses from an aircraft's distance measuring equipment (DME). (Courtesy of Onsala Space Observatory.)

converter (ADC). It is done in two stages; sampling and quantization. Sampling is the process of turning a continuous-time signal into a set of discrete-time points. Figure 2.2 shows sampling of an analog signal. In order to not lose any information in the sampling process, such that the continuous time signal could be reconstructed from the discrete-time data set, the sampling need to be performed according to certain criteria.

From the Nyquist–Shannon sampling theorem we know that if a signal is perfectly band limited in such a way that the signal contains no frequencies above B then the signal can be recreated when the sampling frequency f_s is

$$f_s > 2B$$

[7]. This theorem is simple, reality however isn't, as signals are rarely perfectly band limited and could have harmonics far above its expected frequency [8]. If this is not taken into consideration when sampling recreations of the signal could lose vital information. This leads to difficulties when sampling high frequency signals such as those studied in radio astronomy, up to 30 GHz. In the best case scenarios such a signal would require a 60 GHz sampling frequency, which requires very expensive, specialized equipment. In order to avoid the need for such equipment the signal is usually down-mixed first, as described in Section 2.3.

Quantization is the process of converting values in an uncountably infinite, continuous set (e.g. real values) into values in a finite, discrete set of quantization levels. Figure 2.3 shows an example of how a real-valued signal produced by sampling a sinusoid could be quantized into a signed 3-bit value range.

The quantization operation introduces errors into the data, as an analog value is binned into a certain digital value according to the scheme defined by the system. The error between the true analog value and the digital value represents the quantization error.

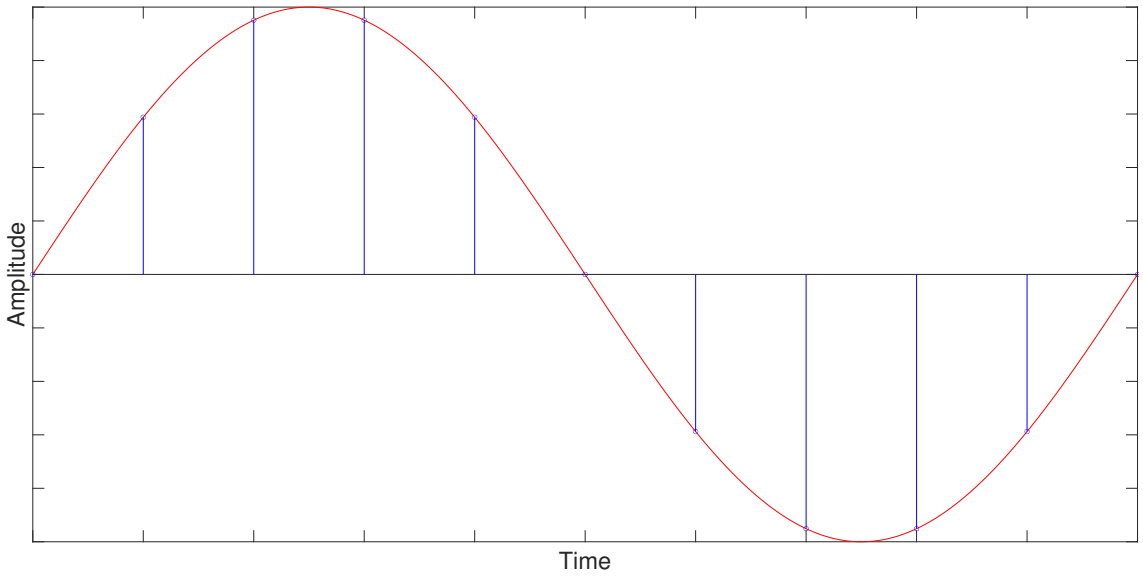


Figure 2.2: Analog signal marked in red and sample points shown in blue.

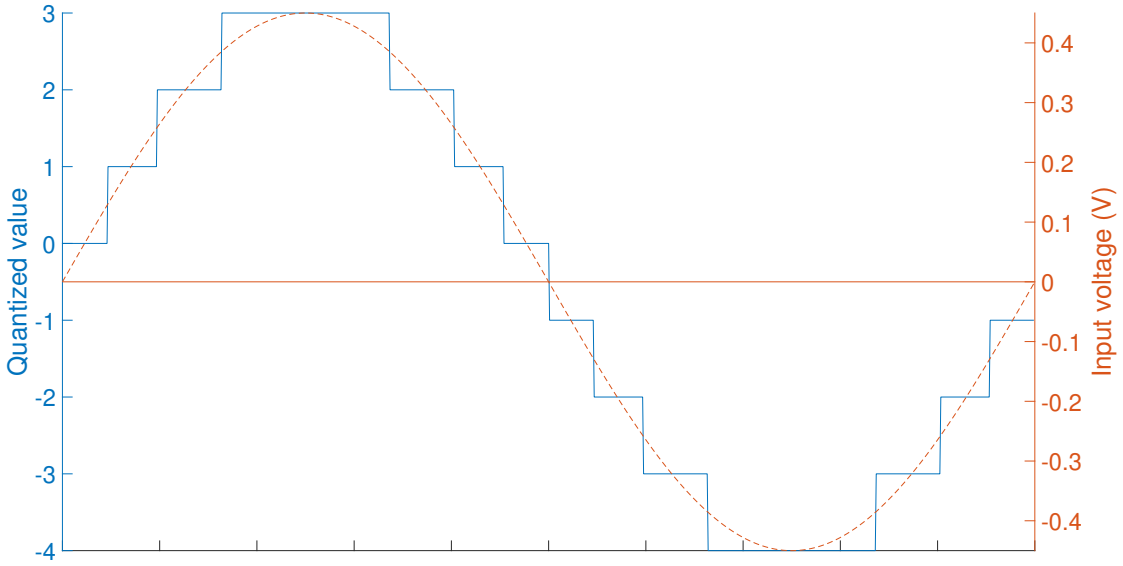


Figure 2.3: A real-valued sinusoid (dashed curve) quantized to 3 bits, signed (blue stair-wave)

For radio astronomy signals, quantization noise is an unavoidable property of quantization. The noise goes towards zero when the number of quantization bits goes towards infinity, making zero quantization noise an unfeasible goal for practical designs. This is because quantization error in a sample can at most be equal to $\frac{\Delta}{2}$, where Δ is the voltage difference between two quantization levels. This assumes equidistant spacing. Assuming the full-scale range, which is the range of acceptable input voltages to an ADC, remains the same, more quantization levels results in a smaller voltage difference between them. This changes the error power and can be seen in Equation 2.1 where it shows that quantization error power is proportional to the square of the difference between two levels. Thus doubling the number of bits halves the delta but reduces the noise power by a factor of four. The probability function of a quantization error E_Q required to use this equation can be seen in Equation 2.2.

$$P_Q = \int_{-\infty}^{\infty} E_Q^2 \cdot p(E_Q) dE_Q = \int_{\frac{-\Delta}{2}}^{\frac{\Delta}{2}} \frac{E_Q^2}{\Delta} dE_Q = \frac{\Delta^2}{12} \tag{2.1}$$

$$p(E_Q) = \begin{cases} \frac{1}{\Delta} \ for \ E_Q \ \in \left\{\frac{-\Delta}{2}, ..., \frac{\Delta}{2}\right\} \\ 0 \ otherwise \end{cases} \tag{2.2}$$

A final important detail regarding the ADC full-scale range is that if the input voltage amplitude to the digitizer exceeds this range, the ADC will be saturated and its output will flat-line at one of its extreme values. This distorts the information content of the signals and the affected samples are often discarded.

2.3 Analog signal pre-processing

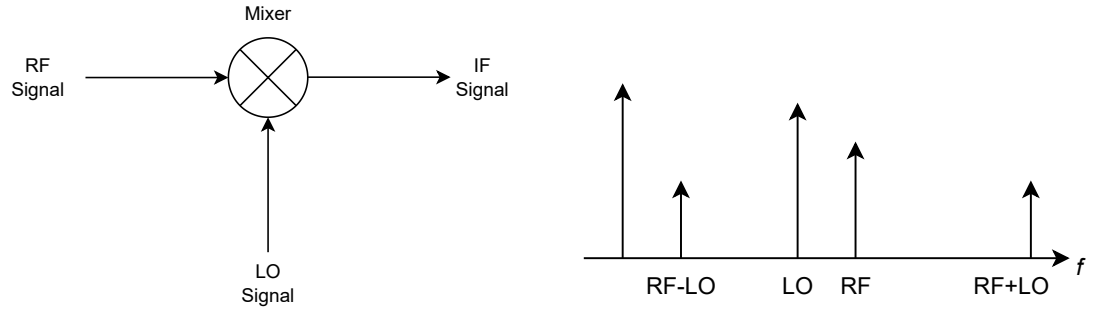
A mixer is a frequency conversion device, used in both transmitters and receivers, which produces an output consisting of the sum and difference frequencies of the two input signals. It is useful for translating a signal up or down in frequency to better suit the transmission medium or the capabilites of the DSP systems. This device would be used in conjuntion with the compression scheme to select bands of interest to be compressed and recorded. Figure 2.4a shows the logical schematic of the mixer in addition to input and output signals. Selecting either sum or difference can be accomplished using filters and is called up-conversion and down-conversion respectively. The resulting output can be seen in Figure 2.4b. This allows for sending and receiving signals with a band centered at a desired frequency [9].

Described below is the down-conversion of an RF-signal using a local oscillator (LO). The modulating LO signal is described as

$$v_{LO}(t) = \cos(2\pi f_{LO} t) \tag{2.3}$$

and the RF input signal as

$$v_{RF}(t) = \cos(2\pi f_{RF} t) \tag{2.4}$$



(a) Frequency mixer. **(b)** Frequency mixer output.

Figure 2.4: Frequency mixer diagram and output.

Using the product-to-sum identity

$$\cos(\theta)\cos(\phi) = \frac{\cos(\theta-\phi)+\cos(\theta+\phi)}{2} \tag{2.5}$$

we get the following equation

$$\begin{aligned} v_{IF}(t) &= K v_{LO}(t) v_{RF}(t) = K\ \cos(2\pi f_{RF} t)\ \cos(2\pi f_{LO} t) \\ &= \frac{K}{2}\left[\cos 2\pi(f_{RF}-f_{LO})t + \cos 2\pi(f_{RF}+f_{LO})t\right] \end{aligned} \tag{2.6}$$

where K is the voltage loss of the mixer. In Equation 2.6 the output has both sum and difference components but as previously stated, the down-converted signal can be acquired by using low pass filtering and this yields

$$f_{IF} = f_{RF} - f_{LO} \tag{2.7}$$

The process is very similar for up-conversion. In up-conversion we instead have an IF signal we want to send. By simply changing the input signal in Figure 2.4a to IF instead of RF in addition to changing filters to high-pass or band-pass we achieve up-conversion.

2.4 Field Programmable Gate Arrays

Field Programmable Gate Arrays (FPGA) are integrated circuits that consist of a large array of small hardware units; programmable logic blocks and reconfigurable interconnects. These can be configured such that they connect together and realize arbitrary combinational circuits, thus becoming capable of performing specific computational tasks. In terms of clock frequency they are slow, generally operating no faster than a few hundred MHz, while general purpose CPU:s operate at GHz frequencies. However, the advantage of an FPGA is its very high potential for parallelized computing as multiple instances of a circuit can be implemented on a single FPGA. Furthermore, unlike traditional computers based on the von Neumann architecture, FPGAs do not necessarily need to write back to memory after each

operation; much of the processing can be performed in a streaming fashion, with intermediary results stored in simple registers located between larger combinational circuits. This computation model is known as spatial computing, as opposed to the temporal computing model which general CPU based computers adhere to.

FPGAs are especially useful for the types of operations and the types of data that this thesis concerns itself with, the details of which will be made more clear in Chapter 4. This is for several reasons:

- Constantly streaming input data -> high data throughput supported
- Samples are collected into blocks -> parallel operations on samples within blocks, or across several blocks
- Little to no feedback required within data processing chain -> pipe-lining possible

The spatial computation capabilities that an FPGA provides suit these characteristics very well.

2.5 RF Processing Hardware

Two FPGA-based RF processing units have been used during development:

- **SP Devices ADQ7WB digitizer**

The final design will run on the ADQ7WB digitizer [10]. It is a combination of an ADC and an FPGA; the ADC has a full-scale range of +0.45 V to -0.45 V and is capable of sampling one analog input channel at a rate of 10 Gsample/s, or two analog input channels at a rate of 5 Gsample/s each. It produces 16-bit quantized data samples which are fed to the FPGA, which provides a user-definable logic space for high-throughput, highly parallelized DSP operations. As the board connects to a PC through a PCIe interface, processed data is passed along to the host PC for storage and analysis.

Earlier work in radio astronomy signal processing on the ADQ7WB board for the OSO has been performed by Bocheng & Xiao [11]. Their work on an FRB detection scheme, while only superficially related to our work, lends credence to the concept of conducting high-performance radio astronomy digital signal processing by utilizing a combined ADC and FPGA such as the ADQ7WB.

- **Xilinx RFSoC4x2 board**

This is a complete system-on-chip (SoC) for RF processing applications [12]. It provides many of the same capabilities as the ADQ7WB but is in some aspects easier to use, and has proved to be more readily available to run tests on than the ADQ7WB during the development of this project. Therefore it has been used for hardware testing and verification of our design.

Figure 2.5: ADQ7WB data acquisition board.

3

Methodology

The approach for designing and implementing the system was continuous and included step-wise background research, development and evaluation. Research was done on different types of existing data recording systems, as well as real-time signal processing systems of telescopes such as the ngVLA [13] and SKA [14]. The aim was to understand the existing techniques for system elements such as signal capture, quantization and further processing, which could provide inspiration for developing our system and performance baselines against which to compare it.

The development of the signal compression system began with software development in MATLAB and Python. This was because these environments allow for much faster and easier development and data visualization than any hardware description language (HDL) development. Using these tools we experimented with many different ideas which we developed, verified, reiterated and evaluated. We analyzed those performance metrics which we feasibly could on a purely software-based model of the system; primarily compression rate and error rate.

After a component had been thoroughly tested and verified we proceeded to implement it in VHDL. Debugging was performed first using behavioral simulation, with output data being written to files for later plotting in MATLAB. Input data for both the MATLAB/Python models and the VHDL simulations consisted of pre-captured sample collections. When enough of the full system had been implemented and verified in simulations, we began testing it on an RFSoC4x2 board connected to a signal generator. While the final goal was to implement the system on an ADQ7WB board that is connected to the actual OSO radio receiver systems, the RFSoC4x2 board proved easier to work with, and was used as an additional testing and debugging aid.

As preparation for the final step of implementing the system on the ADQ7WB, we had to assemble an analog down-mixer. This was because the device itself does not contain any advanced pre-sampling filters, and the received signal had to be band-limited before being input to the device.

4

Design and implementation

This chapter describes the main parts in developing the full system design and is split into three parts: software, adjacent hardware, and hardware implementation. The first chapter which describes the software implementation is focused on how the algorithm was developed. In the second, additional required hardware needed for the design to work properly is described. Lastly, the final chapter describes how the algorithm is translated to hardware and split into multiple pipeline stages to better work with the data stream of the FPGA.

4.1 Initial software design and simulation

The first part of design and development has been performed purely in software, specifically in MATLAB. This has been in order to ease testing and reiteration, to produce theoretical performance metrics by which to evaluate the design, and to simplify later hardware development by ensuring design correctness before transferring to an HDL implementation.

The following subsections describe the different parts of the system and argue for the design choices that have been made. In this text, a few terms are used; "quantization depth" or just "depth" refers to the amount of bits a sample is represented by, "quantization level" or just "level" refers to the nominal values that a certain depth can represent, and "break point" or "threshold" refers to a specific value against which comparisons are made in order for the algorithm to make decisions.

4.1.1 Model overview

The ultimate goal of the system is to compress digital samples of RF signals into as few bits as possible by utilizing variable bitrate. It achieves this by collecting a number of digital samples into a data block, which is in turn analyzed in order to determine the appropriate bit depth that the samples within should be requantized into.

In our model we have elected to scale the quantized data over a the full-scale range of a signed 16 bit integer, as is done in the ADQ7WB. Accordingly, in the model the input signal is first quantized into 2^{16} linearly spaced quantization levels, in a two's complement fashion. This means that digital value 2^{15} represents voltage value -0.45 V and digital value $2^{15} - 1$ represents $+0.45$ V. The digital signal that results from this is the signal upon which our algorithm operates. The collection of samples

into blocks is also a reflection of the computational structure that is utilized in the FPGA implementation. It should be noted however, that the software model has not been parallelized, and thus shows the worst case when measuring performance metrics such as latency or throughput.

Figure 4.1 shows an overview of the software model. The actual compression system is represented by the blocks following the initial 16-bit quantization. These component blocks, as well as other details are explained in the following sections. Figure 4.2 shows the result of an RFI-afflicted signal passing trough the entire processing chain. The quantization depth for each block is noted, as well as the threshold values which dictate them.

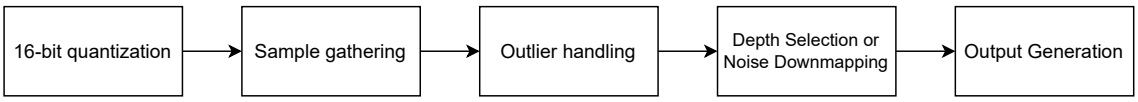


Figure 4.1: Software model overview

4.1.2 Block size for sample gathering

As a radio receiver system operates continuously, the incoming signal is practically infinitely long. For simplicity of implementation, the signal is divided into fixed size blocks. The performance and the adaptivity of the system is dependent on the size of these blocks, which generally is selected to be a power of two. Within a block the quantization depth is fixed, and a change can only occur when a new block starts to be processed. In a configuration with a small block size the adaptivity is high as quantization depth can change frequently. To describe the block a 16-bit header is used which adds an overhead. An extreme case would be if the block size were 1 byte , as it would require 24 bits to communicate 8 bits of information. A configuration with small blocks therefore comes at the cost of more headers and thus a less efficient higher output data rate.

Optimizing the block size is important as too large a block size can also be inefficient. For example, a block may contain a high-amplitude spike which requires the block to be quantized into 8 bits, but the spike may only span across a small fraction of all the samples within the block, leading to most of the block being quantized at an unnecessarily high depth. Signals with frequent short but powerful spikes are especially bad for large block systems and could substantially increase the output data rate.

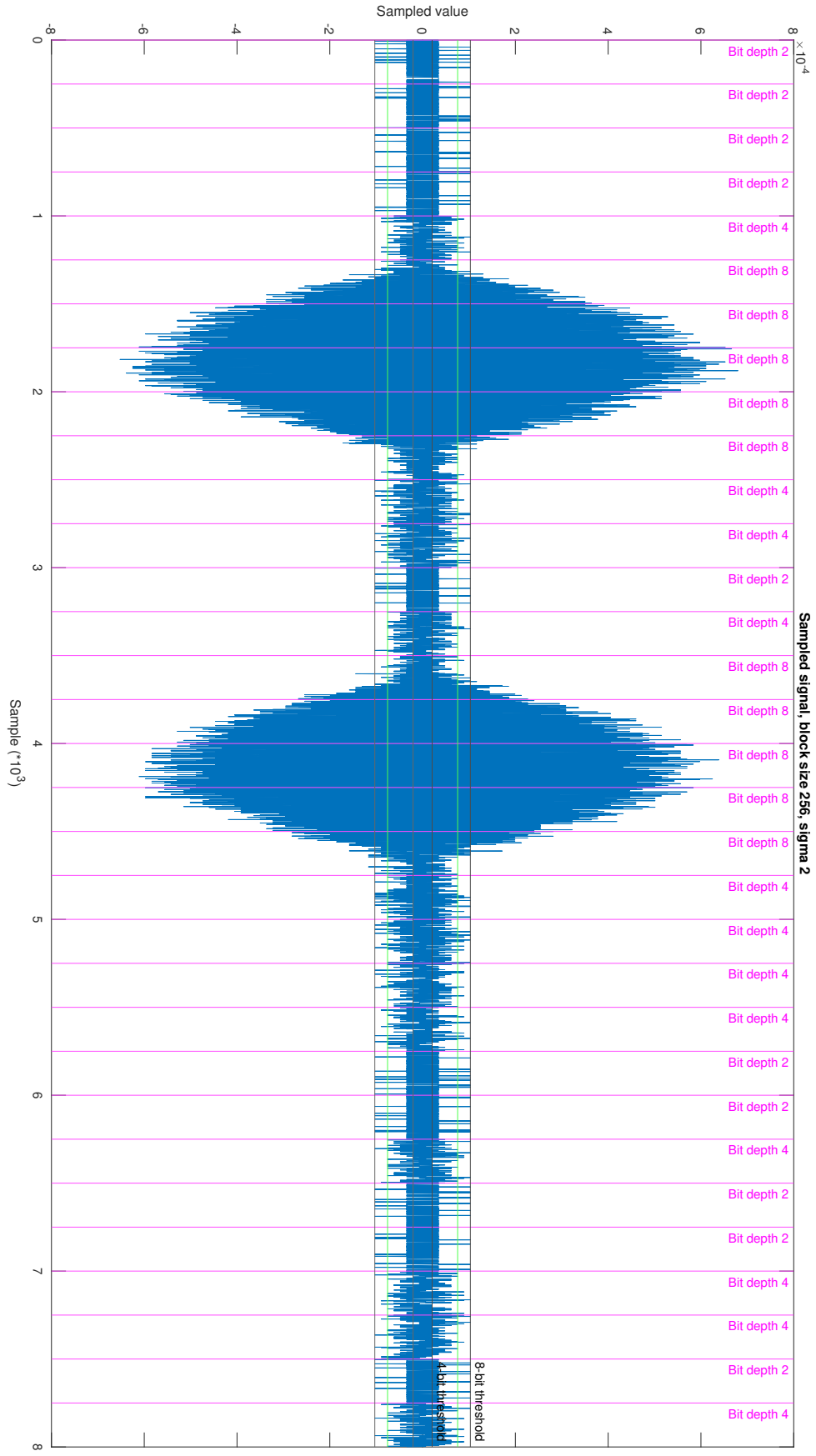


Figure 4.2: Signal split into 256-sample blocks

4.1.3 Quantization depth

The AC voltage samples within a block are represented by digital values -2^{15} to $2^{15} - 1$. In order to determine what bit depth a block of samples will be quantized into, the largest value of a sample in that block is found, and then the smallest bit depth into which this value fits is selected. The only depths considered are 2, 4, 8 and 16 bits, as these are in common use in radio astronomy and DSP applications and divided evenly into standard word boundaries.

An example is shown in Figure 4.3, where the block of samples is represented by a sine wave. Such a signal is unlikely to appear on the telescopes, but it will serve as illustration. In the right figure the entire block fits within the value boundaries provided by 2 bits, so it will be quantized as such. In the left figure there are some samples within the block that exceed these boundaries, which results in the block being instead quantized into 4 bits. The exact same principle applies for the other depth levels.

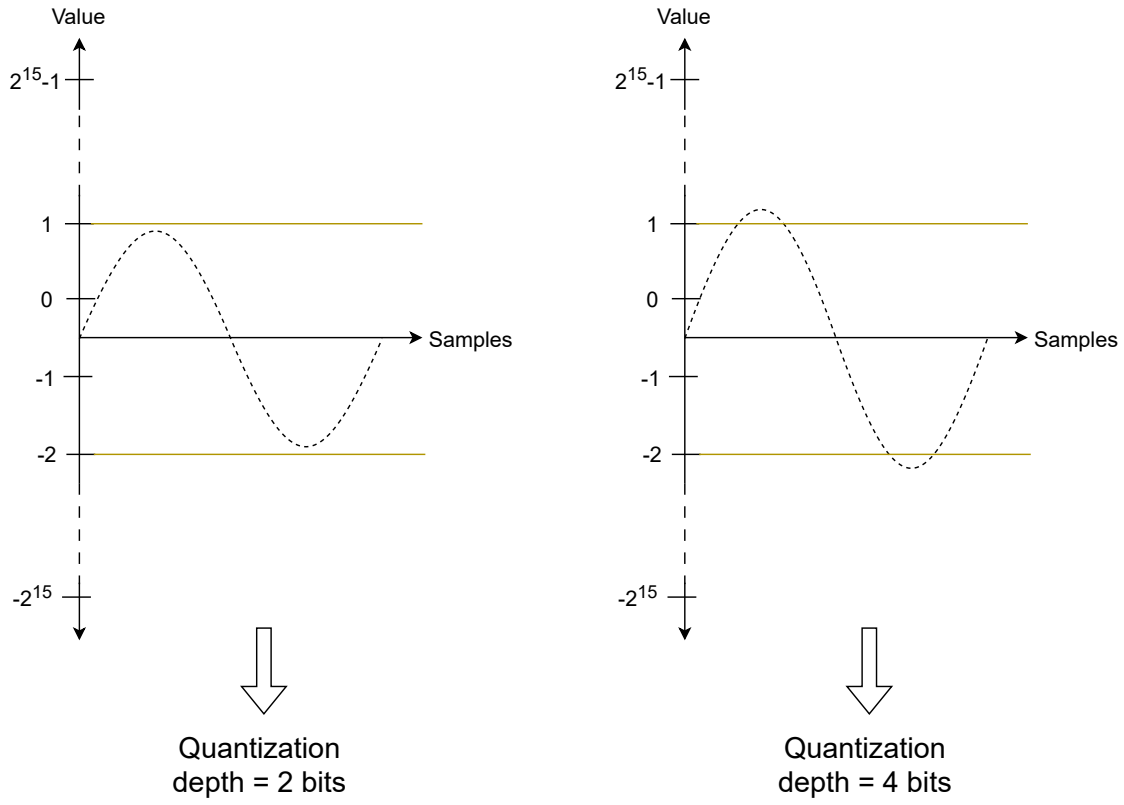


Figure 4.3: Depth selection performed on a block

This is the simplest way of determining required bit depth and it gives the same quantization error as a full 16 bit quantization system. However, outlier values in the input could result in a quantization depth greater than what is necessary; a singular value within a block that exceeds the bounds of a certain quantization depth will cause the entire block to become twice as large, as the next greater depth is selected. But a handful of down-rounded or discarded samples mean little in a system with an input data rate measured in billions of samples per second, so measures can be

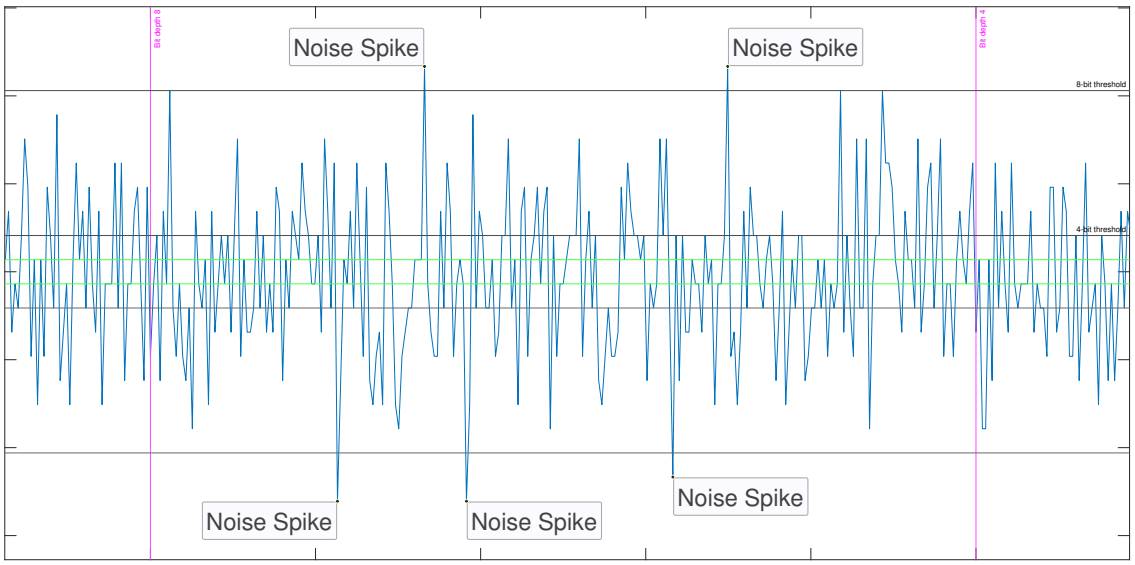


Figure 4.4: Nosie spikes.

taken to ignore such outliers in order to optimize for output data rate. A few such measures are explored in this thesis.

4.1.4 Outlier handling

Noise is present in all signals because no matter how clean the captured signal is, the receiver system itself will introduce noise by way of its inherent physical temperature. This noise is generally considered to be additive white Gaussian noise (AWGN), meaning that its value is normally distributed with zero mean, that individual values are independent of each other across frequency and time, and that any underlying "real" signals do not affect it. Therefore, a block may contain a small amount of samples whose amplitudes, because of system noise rather than any actual signal of interest, exceed the bounds of a certain quantization depth that might otherwise be enough to accommodate a majority of the samples in the block. Figure 4.4 shows how these spikes may look.

Such samples could be discarded or rounded down in order to make the block suit a smaller quantization depth, which come at the cost of introducing some amount of error. For this thesis a few different methods have been explored, two of which are based on the standard deviation (SD) and median absolute deviation (MAD) of the samples within a block. Both methods are used to balance including or ignoring the outliers of spiky inputs in order to optimally select bit depth. A third method is the N samples above break point (NSAB), which is substantially simpler to implement in hardware, entails simply counting the amount of outlier values and truncating them if they together amount to no more than a set, by the user, percentage of the block size.

4.1.4.1 SD method

The SD method entails calculating the mean and the standard deviation of the signal within the block. Adding the mean and a number of standard deviations together yields a value that is compared to the maximum of the range of each quantization depth. By using different numbers of standard deviations (σ) the function can be tuned to ignore few or many spikes. However, this method has a drawback when faced with large outliers since they are more heavily weighted. This could cause of overestimation of the number of bits required.

4.1.4.2 MAD method

Median absolute deviation is a more resilient method than standard deviation for handling outliers, as extremes are weighed less heavily. The method takes the median of the samples and then calculates the absolute difference between each sample and the median. The median of these differences is then the resulting MAD. If there is a sequence of numbers [0, 5, 8 ,9, 42] then the median of these numbers is 8 and the difference sequence is [8, 3, 0, 1, 34]. The median of the new sequence is then 3, much smaller when compared to a standard deviation of around 16.7.

4.1.4.3 N samples above break point (NSAB) method

In addition to the SD and MAD methods a simplified algorithm to determine quantization depth was developed to quickly be transferable to VHDL. In this algorithm, a break point value is set by the user before software or HDL compilation. This value defines an upper limit for how many samples in a block can exceed the boundaries of a certain quantization depth without the depth being increased. As an example, if the break point value is set to 15, then a block containing 10 samples with values > 1 or < -2 will still be quantized into a depth of 2 bits. The exceeding values are then truncated into 1 or -2. The break point value needs to be considered carefully, as the larger it is the longer a powerful transient could be and still be destroyed by the truncation operation.

4.1.5 Noise quantization

Most celestial signals are weaker than the system noise level of the receivers. If there is no RFI stronger than the system noise level present, and additional techniques such as correlation of multiple data streams is used during signal analysis, it is possible to quantize the noise signal (and any interesting signals within it) with at a very low depth, e.g. 2 bits or fewer [15]. A special noise threshold was therefore implemented to further reduce data rates, and it is used when the input signal has very weak or no RFI.

If the signal is found to be below a certain user-definable threshold it will be re-scaled and quantized down to the lowest bit depth, which is 2 bits in this implementation. This is firstly done in a similar way to what is described in Section 4.1.3 with the additional step of coarse quantization, in which all sample values in the block are altered such that they maintain the same dynamic range but can instead be

quantized as 2 bit values. Figure 4.5 shows how coarse quantization of a 4 bit sample to a 2 bit depth is performed. The numbers in the boxes indicate samples values. Essentially, the 4 bit values are scaled down to fit a 2 bit range. Figure 4.6 shows the practical result of this. Two different blocks are separated by vertical red line. Enough of the samples of the left block were originally within the thresholds represented by the green lines, so the block has been coarsely quantized to two bits and then up-scaled in order to cover the same value range as before coarse quantization. The right block contained too many outlier values, so it has not been coarsely quantized.

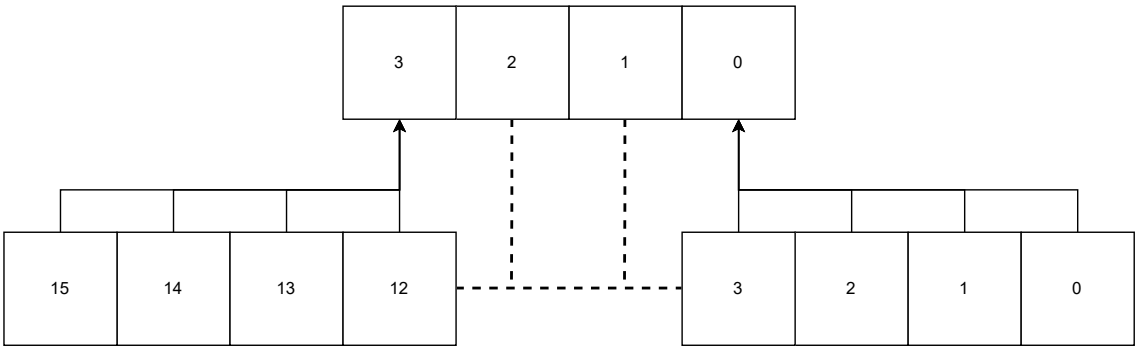


Figure 4.5: Mapping 4 bit samples to 2 bits

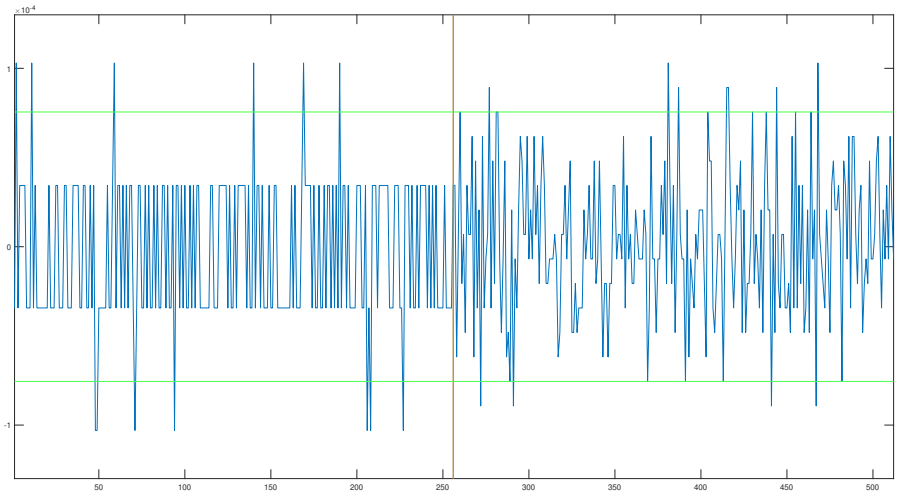
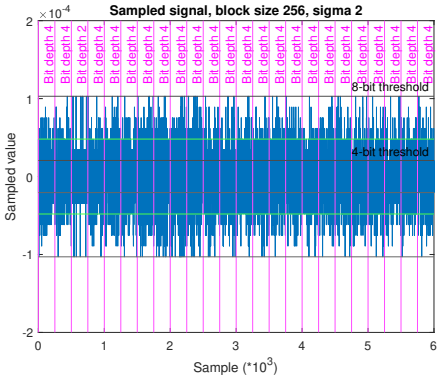
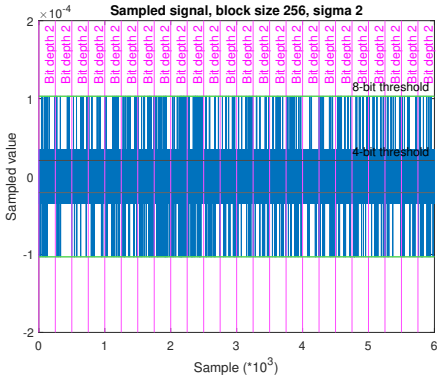


Figure 4.6: Different quantization depths between blocks due to coarse quantization threshold

A comparison between different quantization thresholds can be seen in Figure 4.7. In order to reconstruct the signal for later study, the scaling factor is stored in the block header.



(a) Quantization when threshold set to 2^2.



(b) Quantization when threshold set to 2^3.

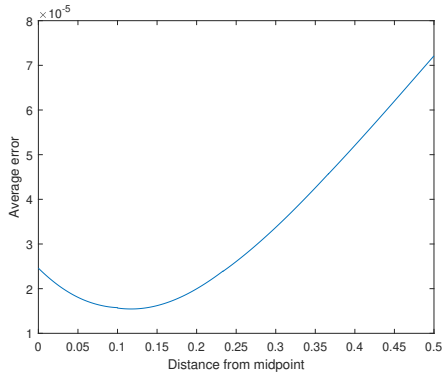
Figure 4.7: Comparison between two different thresholds for coarse quantization. Threshold marked in green.

4.1.6 Minimizing error for coarse quantization

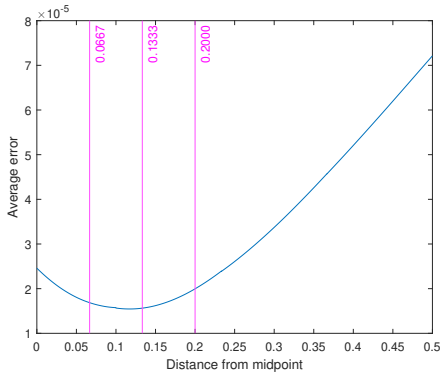
All quantization invariably introduces errors into the sample data; coarse quantization especially so. Two sources, as well as methods to mitigate their effects, have been considered.

One source of errors is the up-scaling process, which may handle blocks whose samples are close to the threshold between two quantization depths poorly. Assume that a block contains samples that are barely above the maximum 4 bit quantization value of 2^3, such that the block would ordinarily be quantized into 8 bits. Assume then also that the coarse quantization threshold is set above the maximum value within the block, which leads to the block being coarsely quantized to 2 bits. The 2 bit quantization allows for four different values to be represented, which after up-scaling would correspond to values $[-2^7, \frac{-2^7}{3}, \frac{2^7}{3}, 2^7]$, assuming perfectly linearly spaced levels. Because the samples originally had values of around 2^3, they would then end up being represented by $\pm\frac{2^7}{3}$, which is a large discrepancy. This introduces large errors into the data. In order to minimize this error, additional analysis is made to find whether settling for a lower quantization depth, in this case 4 bits, before coarse quantization would be preferable.

Another side effect of coarse quantization is an increase in average error, but this can be mitigated somewhat by clever placement of the break points that decide into which 2 bit value a larger value is to be mapped to. Lowering the bit depth while keeping the dynamic range will in most cases increase the quantization error delta, Δ, as resolution is decreased. Higher bit depth quantization have assumed linearly spaced break points. However, empirical analysis of noise indicates that this is not optimal for minimizing average Δ. This is fairly intuitive; the magnitude of AWGN is normally distributed, meaning that most values cluster around the mean. Therefore, increasing the resolution around the mean by moving the break points for the center-most quantization levels closer together will decrease the average Δ after coarse quantization.



(a) Average Δ for different breakpoints for 4 bit quantization with 2 bit coarse quantization of an AWGN signal



(b) Markings indicating possible break point locations

Figure 4.8: Optimal quantization level placement.

Figure 4.8a shows the relation between average Δ and different break point placement when an AWGN signal quantized to 4 bits is coarsely quantized to 2 bits. The x-axis represents the normalized absolute distance between the nominal sample value 0 and where the two intermediary break points are placed. An x-value of 0.5 represents that the break points are placed on the maximum and minimum quantization levels. If the breakpoints were linearly spaced they would be placed at a distance of around $x = 0.167$ from the midpoint which, as can be seen, does not result in the lowest possible average Δ.

One thing to note about setting these breakpoints for minimal average Δ is that they can not be set arbitrarily. In the ADQ7WB there are 2^{16} fixed quantization levels linearly spaced over a range of 0.9 volts, from -0.45 to +0.45, which equals a minimum voltage difference between levels of 1.4 μV. With this in mind the easiest way to set quantization levels is to analyze the noise and look at the corresponding break-point-to-error-rate graph, as seen in Figure 4.8b, and select a quantization level that minimizes the error.

4.2 Hardware implementation

This section presents the physical hardware setup that has been assembled in order to provide a signal feed to the FPGA board, as well as detail around the HDL implementation of the software model described in the previous section.

4.2.1 Band-selection assembly

To select a band of interest to be recoded by the ADC, a mixer, together with a couple of low-pass filters are connected in-between the antenna and ADC as seen in Figure 4.9. Figure 4.10 shows the down-mixing assembly. The black box in the top left corner is the local oscillator and is controlled through serial by a network con-

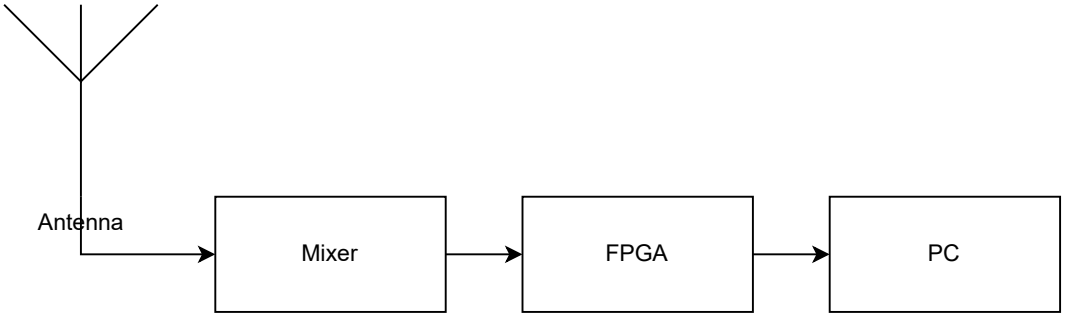


Figure 4.9: High level system overview.

nected Raspberry Pi which can be seen in the top right corner. The local oscillator output is connected to a power divider whose output feeds the two mixers which down converts two antenna RF signals, indicated by the two red squares in Figure 4.10. Each down-converter mixer output is bandlimited to 250 MHz by a lowpass filter, which is marked in yellow. Power is supplied using a 12 V/6 A power supply which powers both the local oscillator and Raspberry Pi. The setup is confirmed to function correctly as shown by Figure 4.11, which shows the signal output after down-mixing 10.4 GHz satellite-TV signals into a 0-250 MHz band.

4.2.2 HDL implementation

For implementation on hardware, the algorithm has been split into multiple pipelined steps as seen in Figure 4.12 and consists of accumulation and depth selection followed by two stages of decimation. Dividing the algorithm into multiple stages was done in order for the design to better handle a stream of data as well as simplify implementation. Additionally, as of writing, the hardware implementation does not include the error minimization methods described in Section 4.1.6.

4.2.2.1 Accumulator

In the first pipeline stage, samples are gathered into a blocks for processing further down the pipeline. Each sample is in this stage 16 bits. Concurrently with accumulation, samples are checked against quantization levels and the result used to increment counters corresponding to each level, except the highest. As an example, if a sampled value is -9 then this value would be outside what is quantizable using 2 and 4 bits and thus the algorithm will increment their corresponding counters by 1. For 8 bits however, the sample is quantizable, therefore this counter will not be incremented.

A threshold counter has also been added in addition to the quantization level counters. This threshold value corresponds to the minimum values which we want to quantize at higher depths. Therefore if a block is below this threshold it will be mapped to the lowest level. The accumulated block and the counters are then sent to the depth selector.

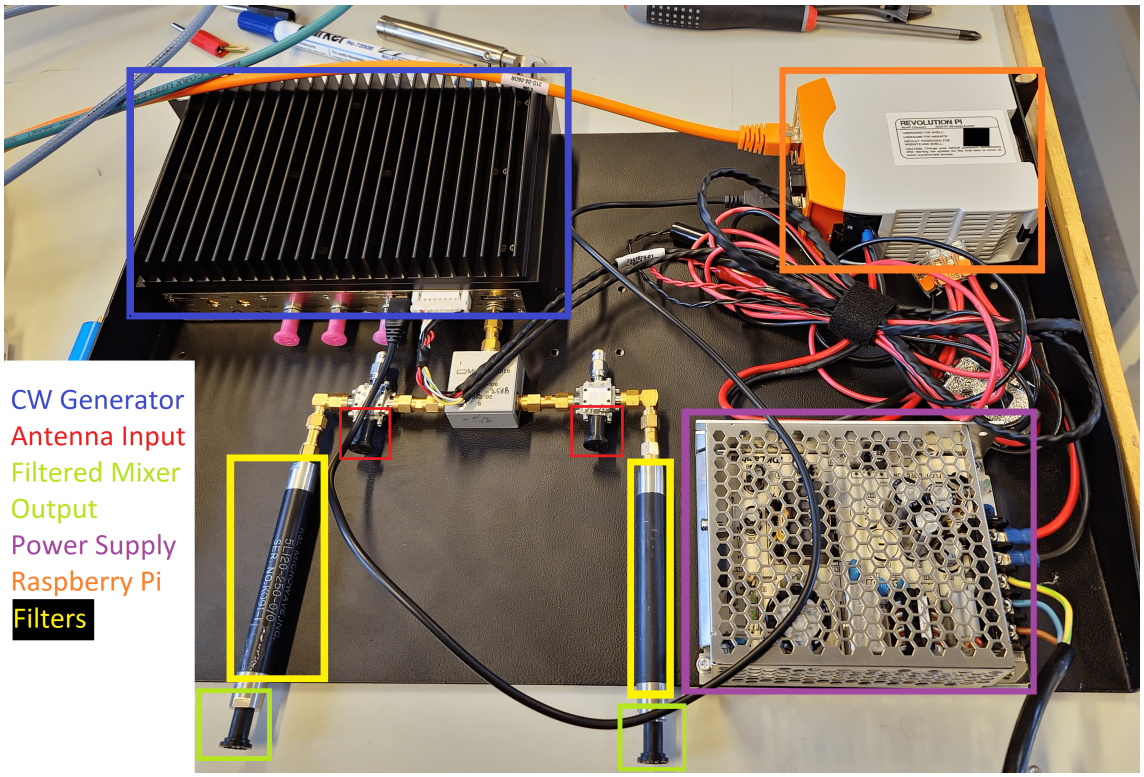


Figure 4.10: Mixer hardware

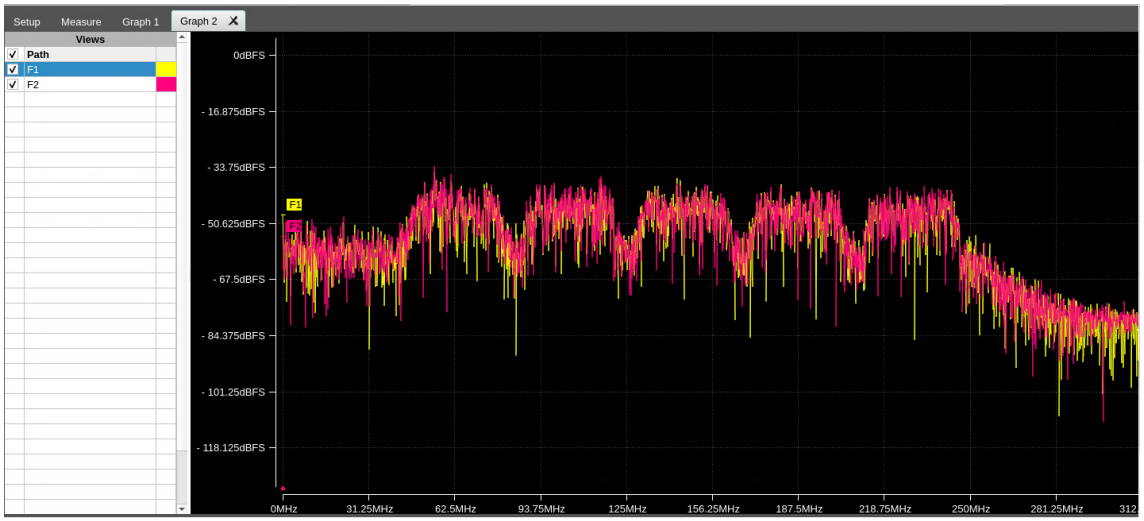


Figure 4.11: Result of mixer hardware downmixing satellite-tv broadcasts from 10.4 Ghz.

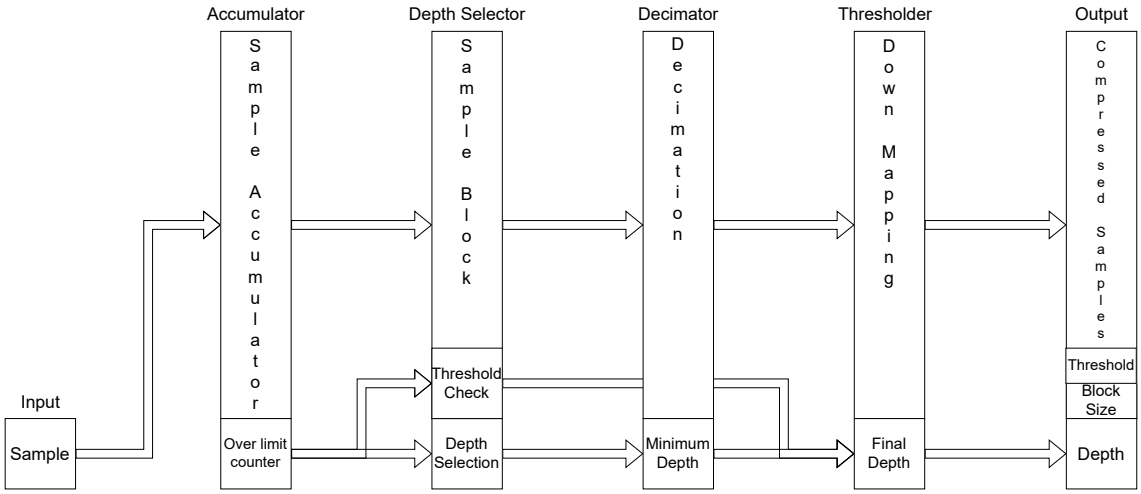


Figure 4.12: System pipeline flowchart.

4.2.2.2 Depth selector

As the name implies, the depth selector selects the quantization level for the entire block. For the first decimation stage this is level is selected using the counters from the accumulator and a value indication the maximum allowed samples above the quantization level capability. This value in essence allows the user to select how strict the decimator should be. If the number of samples above the quantization capability is higher than the maximum allowed, the quantization is done at a higher level. This is checked against all counters and the lowest depth fulfilling this requirement is selected and output to the decimator. This selection is also done for the threshold and is then pipelined through the decimator to the thresholder.

4.2.2.3 Decimator

The decimator is the first block in which the data samples are compressed. This process is very simple and uses the depth output by the depth selector. Values outside the quantization capabilities of this depth are set to the minimum and maximum values of said depth while values within simply keep their value but with excess 0's removed. This is possible since the system uses unsigned values internally. The compressed samples are then output to the thresholder.

4.2.2.4 Thresholder

This final step of the re-quantization is optional and depends on if the sampled block is below the threshold described in Section 4.2.2.2. When a block is above the threshold it is simply output directly with corresponding depth, no remapping applied. If a block is below however it is coarsely quantized in a similar way to what is shown in Figure 4.5. The levels for coarse quantization are selected to be linearly spaced within the threshold range, equal to 2 times the threshold value. In comparison to the previous coarse quantization used in software simulation, this scales the values to be at maximum equal to the threshold where as the previous

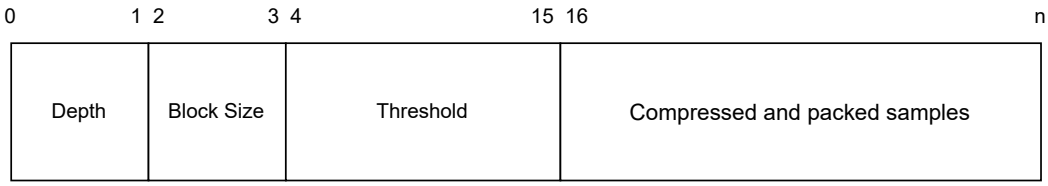


Figure 4.13: Data structure of the output where n is equal to $Depth \cdot Blocksize + Header$.

implementation could scale them to the required depth. This coarse quantization will introduce errors, in the same way as designs described in earlier chapters. The final output data is then presented as shown in Figure 4.13.

5

Results

In this chapter, graphs and results are presented presented from software and hardware simulations, as well as, results from the algorithm running on actual hardware.

5.1 Simulation results

In this section, graphs and tables for the different designs presented in Chapter 4 are shown and compared. These results are the basis of choices made in hardware implementation. The input signal used to create these results can be seen in Figure 5.1 and exist of 8192 samples.

The baseline for storage performance comparisons is the theoretical storage requirements of existing recorders with bit-depths of 2 and 12-bits. The required storage is equal to the $depth \cdot samples$. Since the length of the input signal used in these results is 8192 samples the theoretical storage requirement equals $2 \cdot 8192 = 16384$ bits = 2048 Bytes for 2-bit quantization. For 12-bit quantization this equals 98304 bits = 12288 Bytes.

5.1.1 No outlier handling method

The performance of the no outlier handling method is presented in Figures 5.2a and 5.2b where no coarse quantization threshold has been used, and Figures 5.3a and 5.3b where a coarse quantization threshold has been used. When not utilizing coarse quantization, a data rate reduction of up to 50% compared to the 12-bit quantization case is achieved for the smallest block size, and 33% for the largest block size, without the introduction of error. With coarse quantization the data reduction for the smallest block size is increased to 60%, at the cost of introducing some error.

5.1.2 NSAB method

The performance of the NSAB method is shown in Figures 5.2c, 5.2d where no coarse quantization has been performed, and Figures 5.3c, 5.3d where coarse quantization has been performed. This method yields a worst-case data rate reduction of 49% when a large block size and no coarse quantization is used, and peak data rate reduction of 70% when using a small block size in conjunction with coarse quantization.

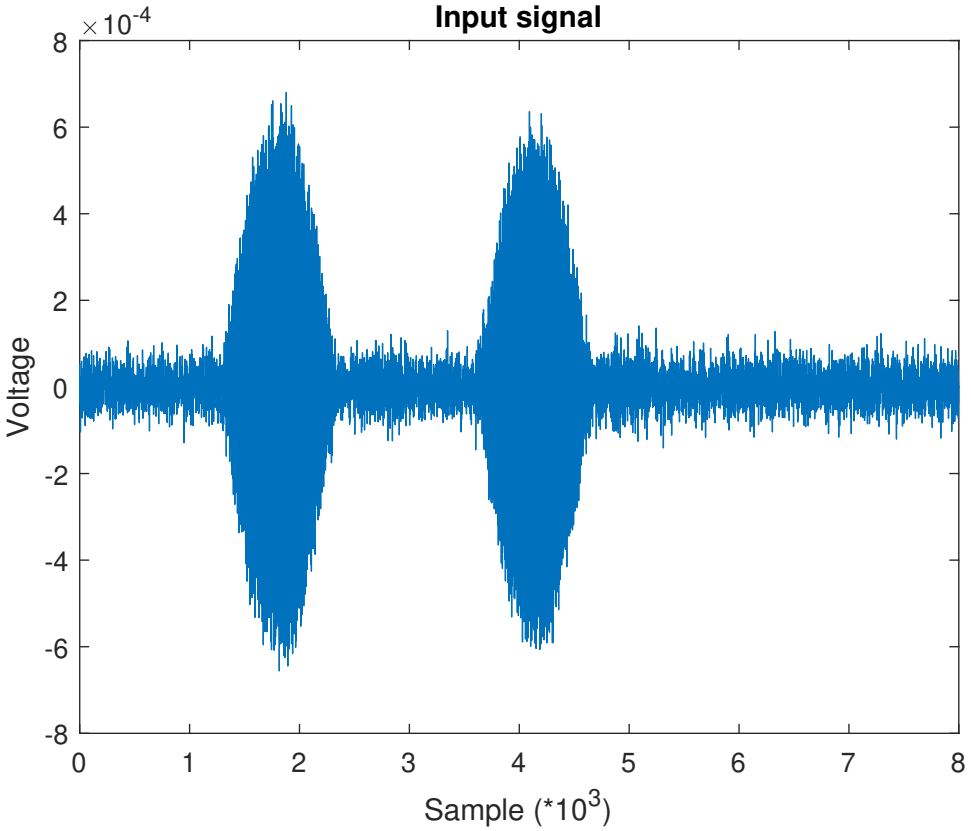
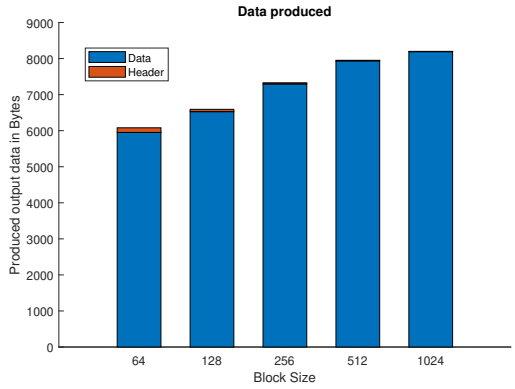
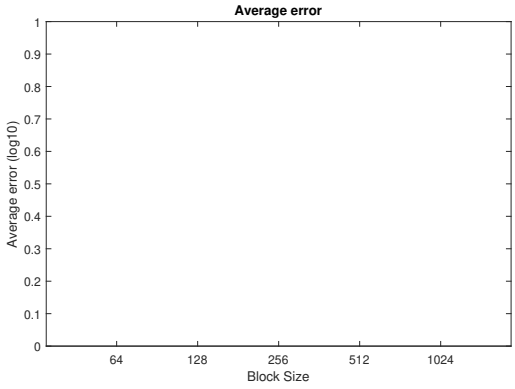


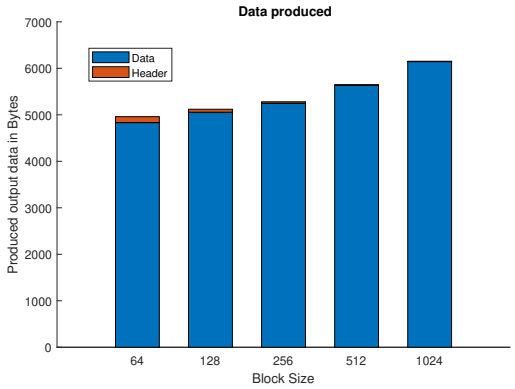
Figure 5.1: The pre-captured input signal used to produce the results presented in this chapter



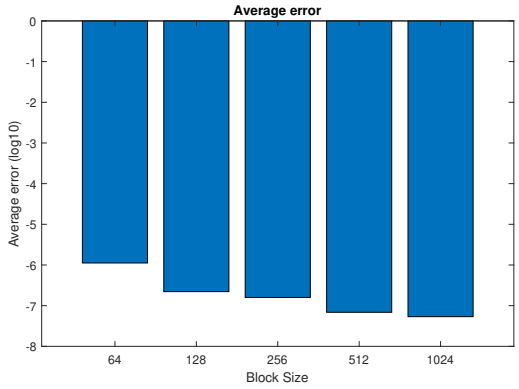
(a) Data rate no outlier handling method, no coarse quantization threshold.



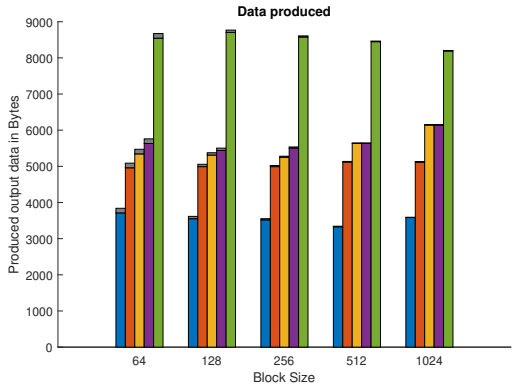
(b) Average error no outlier handling method, no coarse quantization threshold.



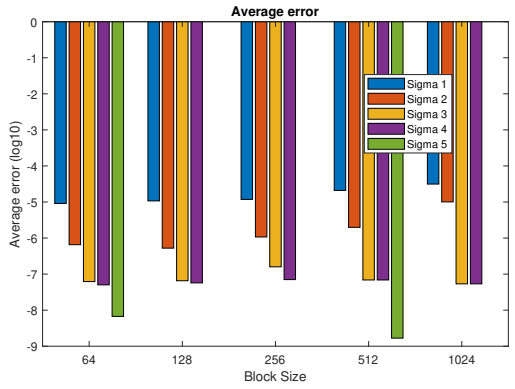
(c) Data rate NSAB method, no coarse quantization threshold.



(d) Average error NSAB method, no coarse quantization threshold.

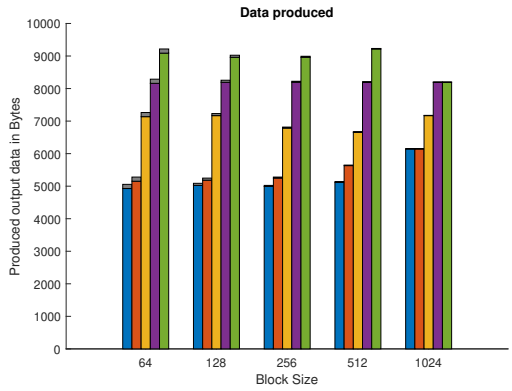


(e) Data rate MAD method, no coarse quantization threshold.

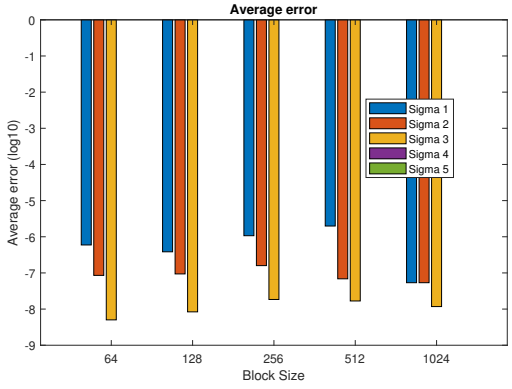


(f) Average error MAD method, no coarse quantization threshold.

Figure 5.2: [Part 1 of 2]



(g) Data rate SD method, no coarse quantization threshold.



(h) Average error SD method, no coarse quantization threshold.

Figure 5.2: [Part 2 of 2] Data rate and error comparison between no outlier handling, NSAB, MAD and SD methods. These runs did not use any threshold values. No error bar equals an average error size of 0.

When using coarse quantization the block size has little effect on the average error, but when not using coarse quantization a larger block size yields smaller error.

As this method was chosen for hardware implementation, additional analysis in form of a Fourier analysis of the processed signal using the NSAB method at different block sizes is shown in Figure 5.4. Comparing this with the Fourier transform of the 12-bit quantization in Figure 5.5 shows a similar result with a worst case difference of approximately 4 pW, and that the impact of the block size is minimal.

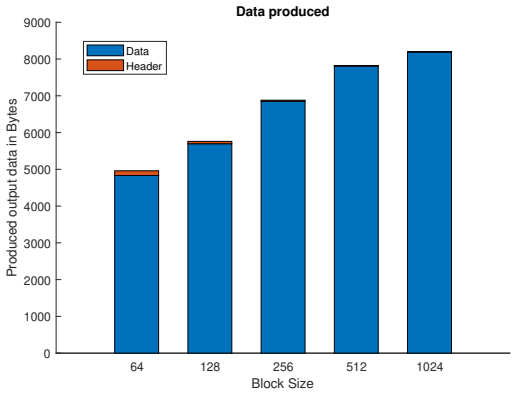
5.1.3 Median absolute deviation method

Results of processing the signal using the median absolute deviation method are presented last in Figures 5.2e, 5.2f where no coarse quantization has been performed, and Figures 5.3e, 5.3f where coarse quantization has been performed. The figures show how different block sizes and number of sigmas affect data rates and error sizes.

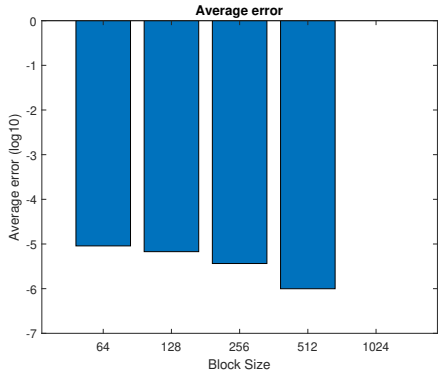
Processing with this method results in a data rate reduction of 25% in the worst case, with no coarse quantization and the highest sigma value. Relaxing the sigma and introducing coarse quantization improves the performance, with the downside of introducing more error. The greatest data reduction of 75% is achieved using the largest block size with the fewest sigma.

5.1.4 Standard deviation method

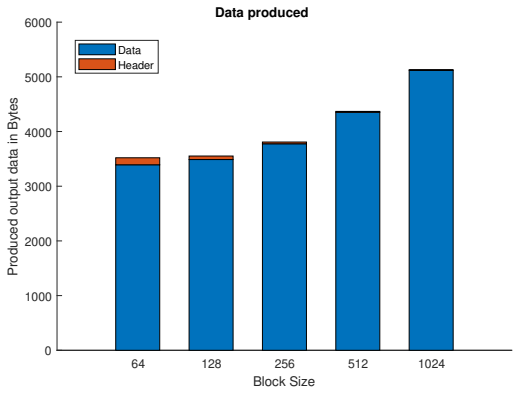
Results of processing the signal using the standard deviation method are presented Figures 5.2g and 5.2h where no coarse quantization has been performed, and Figures 5.3g and 5.3h where coarse quantization has been performed. Again, yields for several different block sizes and sigma values are shown.



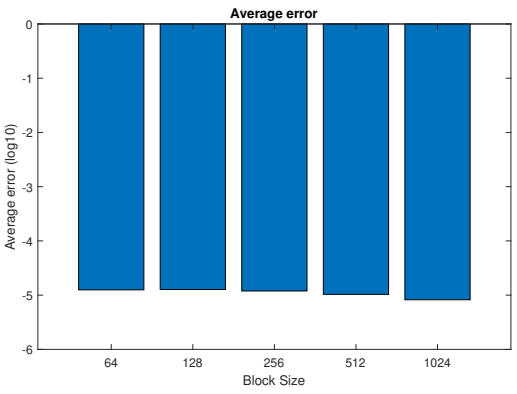
(a) Data rate no outlier handling method, with coarse quantization threshold.



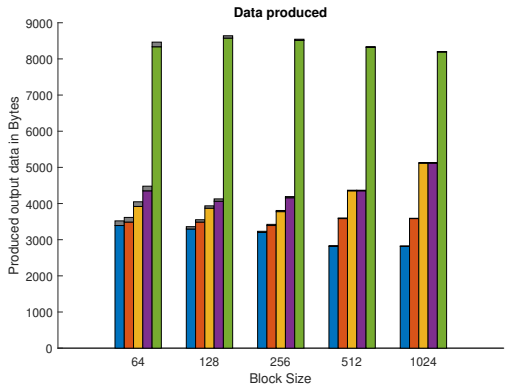
(b) Average error no outlier handling method, with coarse quantization threshold.



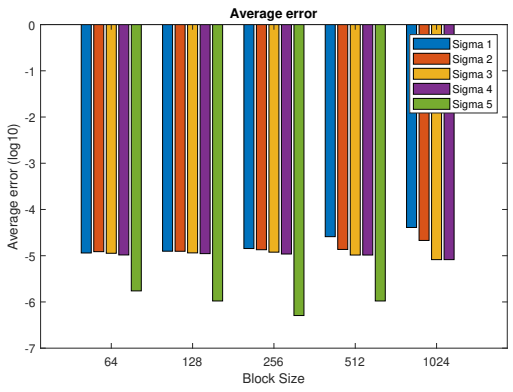
(c) Data rate NSAB method, with coarse quantization threshold.



(d) Average error NSAB method, with coarse quantization threshold.

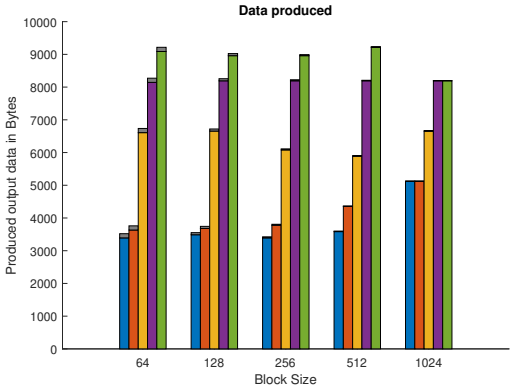


(e) Data rate MAD method, with coarse quantization threshold.

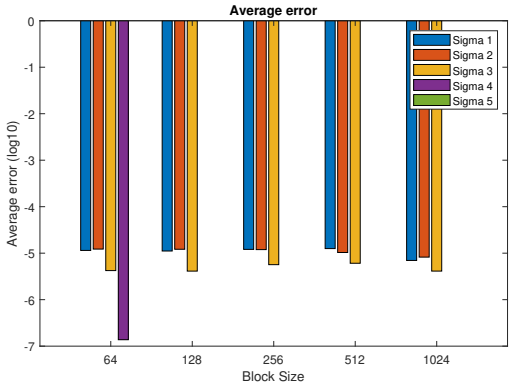


(f) Average error MAD method, with coarse quantization threshold.

Figure 5.3: [Part 1 of 2]



(g) Data rate SD method, with coarse quantization threshold.



(h) Average error SD method, with coarse quantization threshold.

Figure 5.3: [Part 2 of 2] Data rate and error comparison between no outlier handling, NSAB, MAD and SD methods. These runs used a down-mapping threshold of 2^3. No error bar equals an average error size of 0.

The method yields a worst-case data reduction of 25% with no coarse quantization and the highest sigma value. This improves as the sigma value is relaxed and coarse quantization is utilized, up to a peak data reduction of 71%, at the cost of introducing more error.

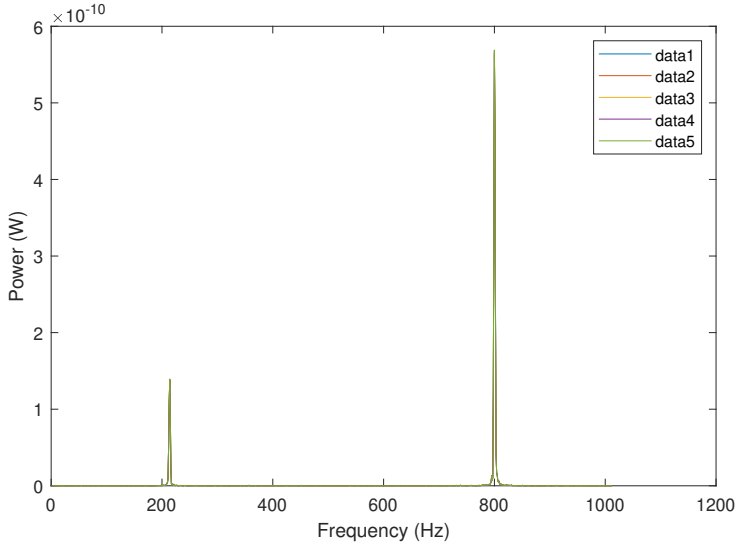
5.1.5 Hardware simulations

Hardware simulations of the algorithm showed similarities to the software simulations with a difference in the power spectrum as shown in Figure 5.6, where the spikes show a more than 40% increase in power. To store the processed signal approximately 4.25KB was required for the input signal.

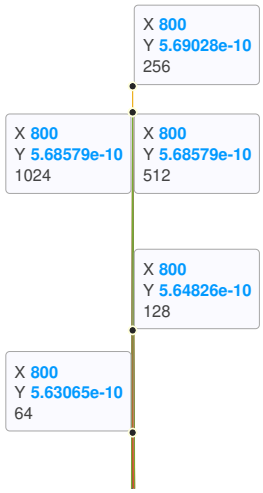
5.1.6 System comparisons

For the last part of the results, comparisons with current systems are made. Firstly, in Figure 5.7, the frequency spectrums of a signal quantized using the currently available systems as well as the dynamic system is presented. The dynamic system show similarities to the 12-bit system. The 2-bit system on the other hand show deviations with the signal clipping becoming evident.

In the second part, Figure 5.8, comparisons between the amount of data produced by different systems are made. The figure shows how much data is produced for each type of input signal. For all the signals, the dynamic system produced less data than the 12-bit system. In the case of only noise, the dynamic system had a data increase of less than 0.4% when compared to the 2-bit system. This increase is due to the addition of the headers.



(a) Fourier analysis of processed real RFI signal.



(b) Frequency component at 800 Hz for different block sizes.

Figure 5.4: Fourier analysis of signal processed using the NSAB method., where (a) displays the frequency spectrum with a clear component at 800 Hz, and (b) displays the power levels of this frequency component when processing the signal with block sizes from 64 up to 1024.

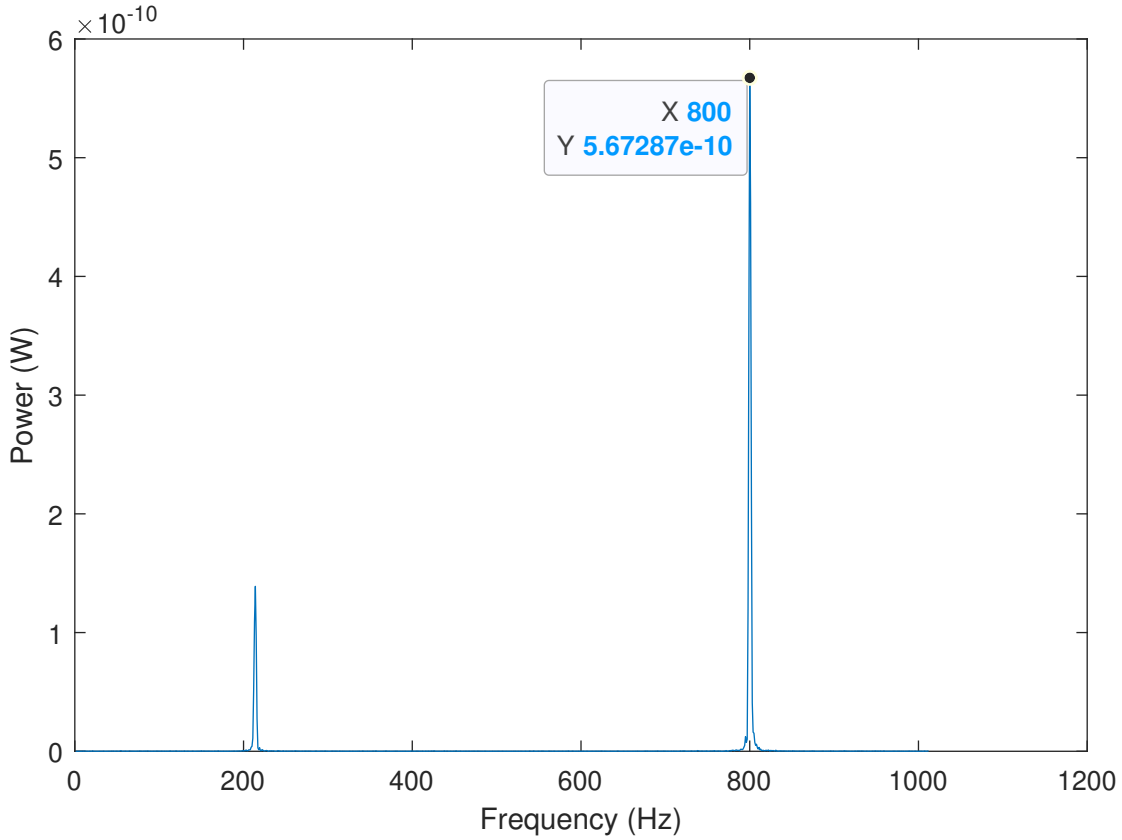


Figure 5.5: Fourier analysis of processed real RFI signal using a bit depth of 12-bits.

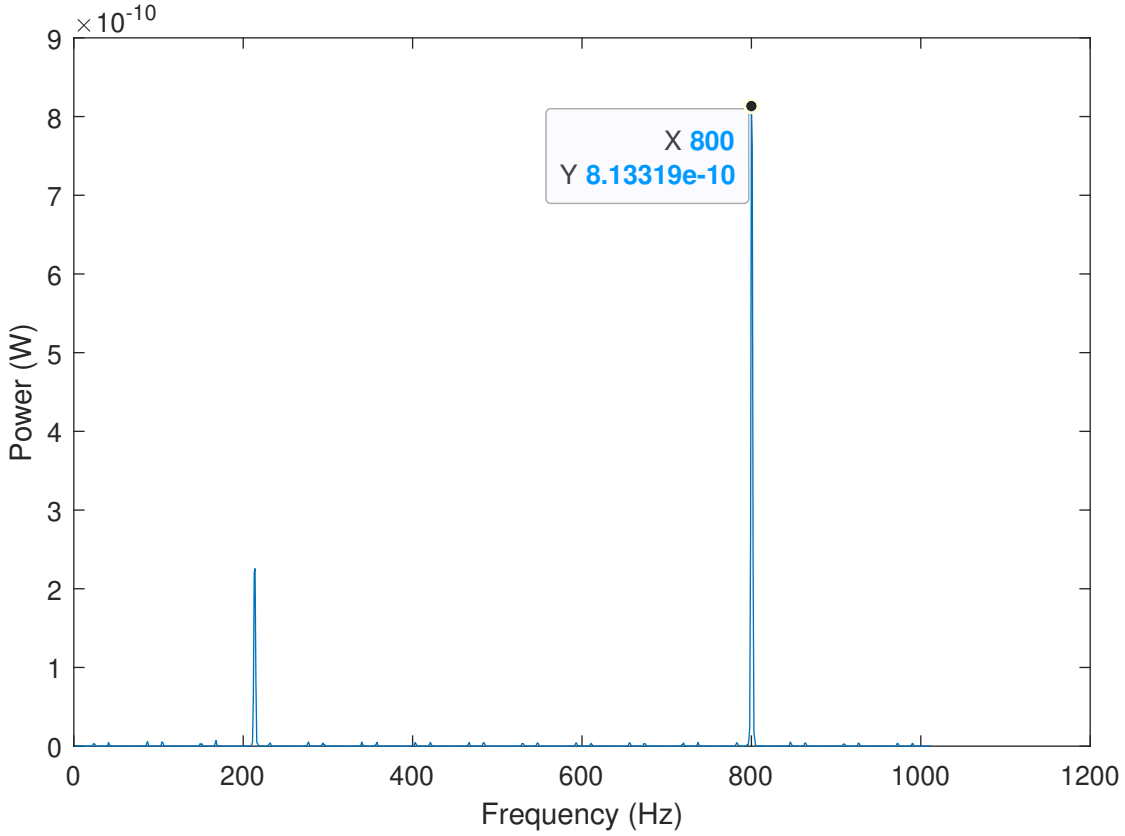


Figure 5.6: Fourier analysis of real RFI signal processed using hardware simulation.

5.2 Hardware implementation results

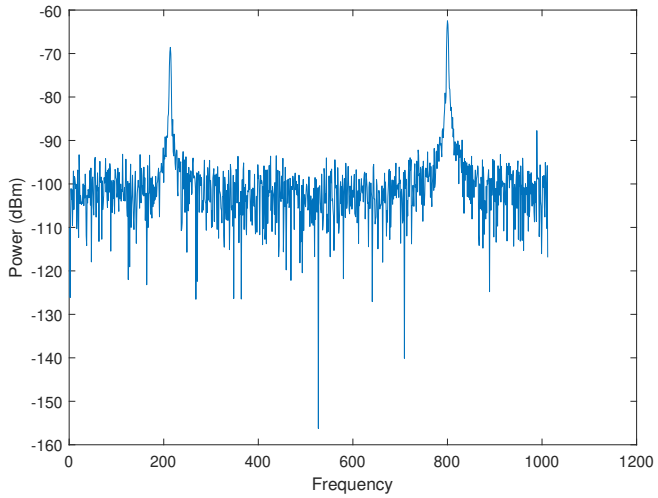
Presented in this section are the results from implementation and testing of the algorithm on hardware. Testing was done on the RFSOC 4x2 board and the design used the min-max method with a block size of 512.

5.2.1 Quantized sine wave

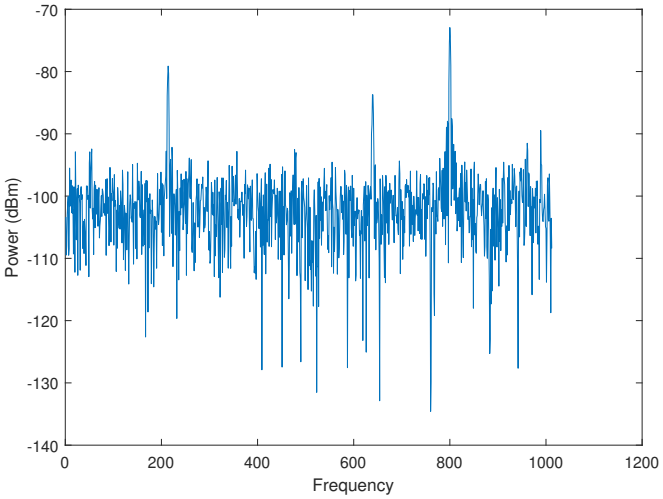
To test the algorithm on hardware a sine wave at different power levels was used and the resulting quantized signal is shown in Figure 5.9. For the first 3 plots the downmapping was turned off. In the plots, the raw data from the algorithm is shown, as such there are some periodical spiky outliers and these represent the headers. The spikes would not be present after processing. When the captured data was read the size was constant and this is why the amount of samples double when reducing quantization depth.

5.2.2 FPGA resource usage

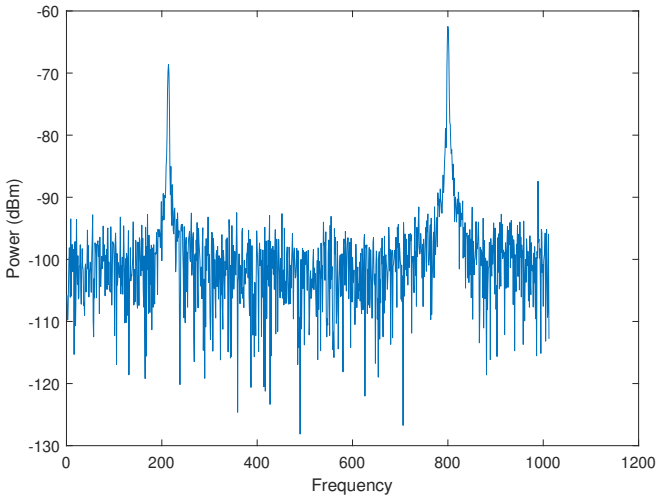
The final results regard the resource usage of this algorithm implemented on hardware. In Table 5.1 the resource usage of the design is shown. This only includes the algorithm and no other components such as wrappers. As a convenience the total



(a) Frequency spectrum when quantized to 12 bits.

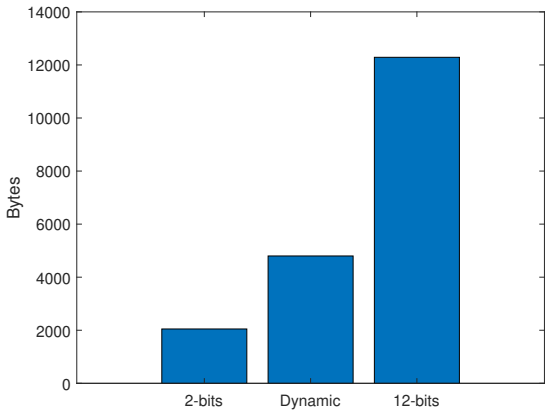


(b) Frequency spectrum when quantized to 2 bits

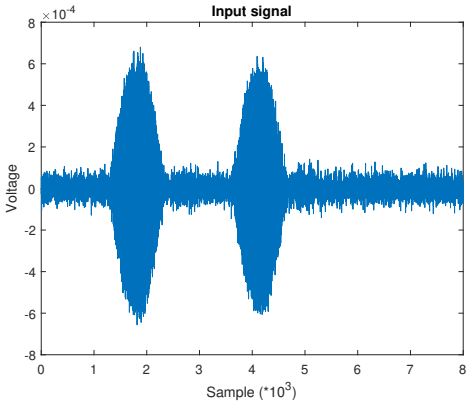


(c) Frequency spectrum when quantized in varying bit depth, NSAB method.

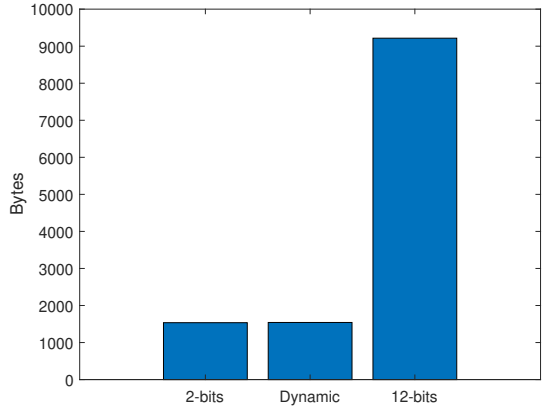
Figure 5.7: Comparison of frequency spectrums when using different systems.



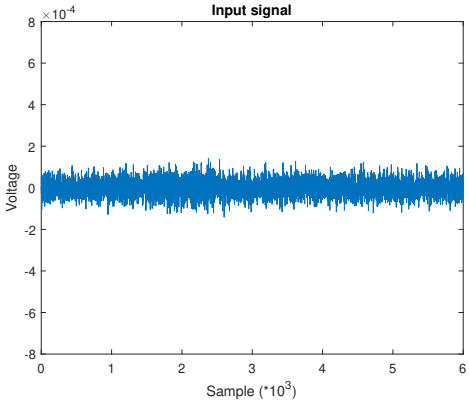
(a) Data produced when quantizing typical RFI.



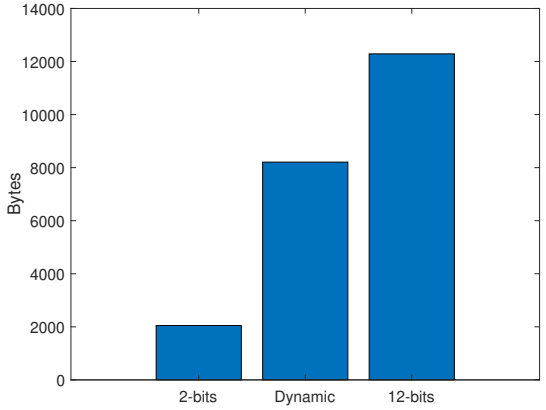
(b) Input signal used to get results in Figure 5.8a



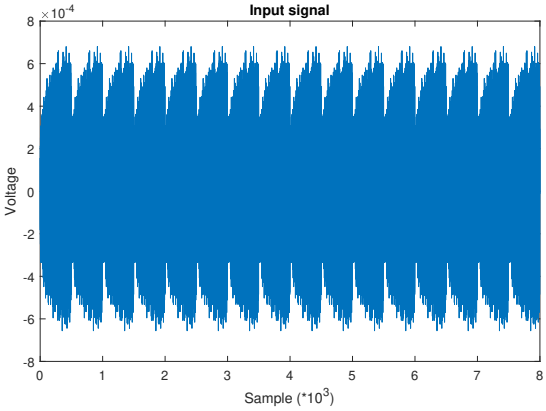
(c) Data produced when quantizing only noise.



(d) Input signal used to get results in Figure 5.8c

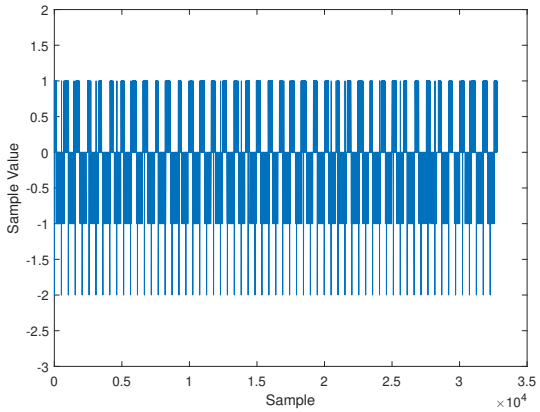


(e) Data produced when quantizing an in between extreme. Needs more than 2 bits but less than 12.

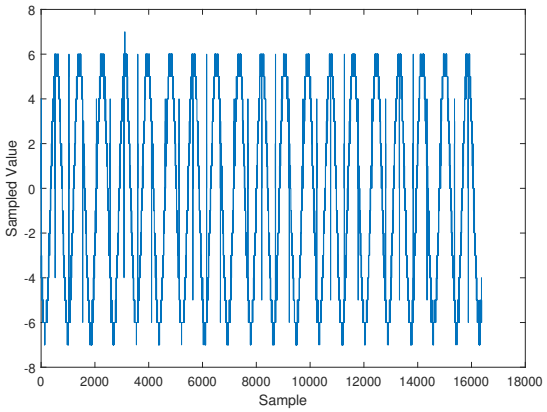


(f) Input signal used to get results in Figure 5.8e

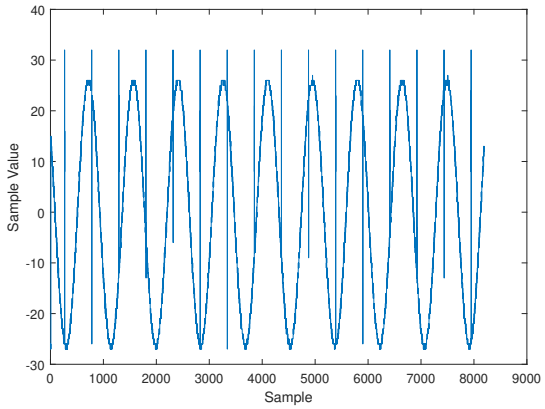
Figure 5.8: Comparison of data produced when using different systems.



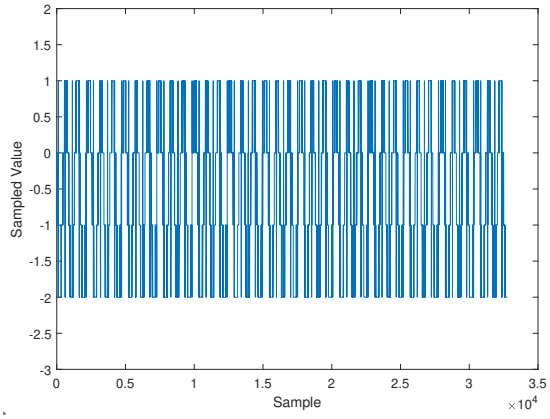
(a) -56dBm sine wave quantized to 2 bits.



(b) -36dBm sine wave quantized to 4 bits



(c) -23dBm sine wave quantized to 8 bits.



(d) -23dBm sine wave quantized to 2 bits using coarse quantization.

Figure 5.9: Comparison of quantization of sine waves at with different power.

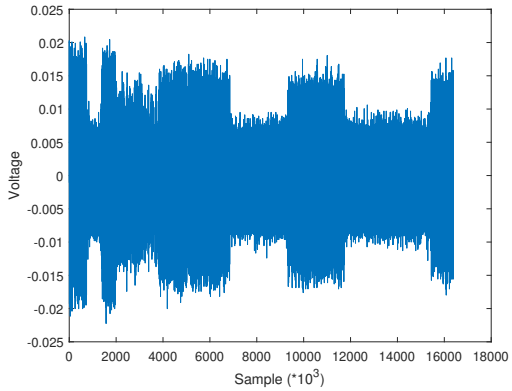
	CLB LUTs	CLB Registers	CARRY8	F7 Muxes	F8 Muxes	CLB
Design	56944	41598	4126	1296	544	9026
Available	425280	850560	53160	212640	106320	53160

Table 5.1: FPGA resource usage.

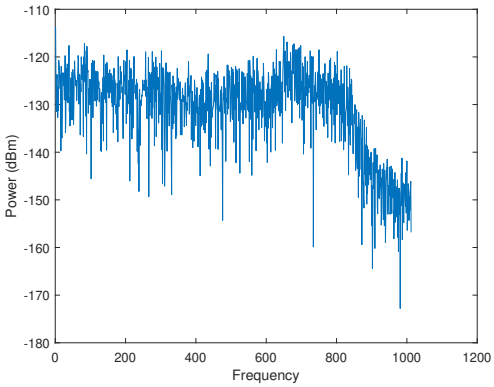
available resources on the FPGA has been included in the bottom row. Resource usage is dependent on the block size used, in this case 512 samples and as seen in the table, for this FPGA, larger block sizes are possible.

5.3 Onsala Twin Telescope snapshot

To further investigate the performance of the NSAB algorithm an additional signal was tested. The input signal and spectrum can be seen in Figure 5.10 and consists of a ≈ 10ms snapshot containing RFI captured by the OTTS with ≈16.8 million samples. Figure 5.11 shows the signal after having been processed using a MATLAB implementation of the NSAB algorithm. The resulting size of the data set has been reduced from ≈32.8 MB to ≈13.8MB, or by 58%.

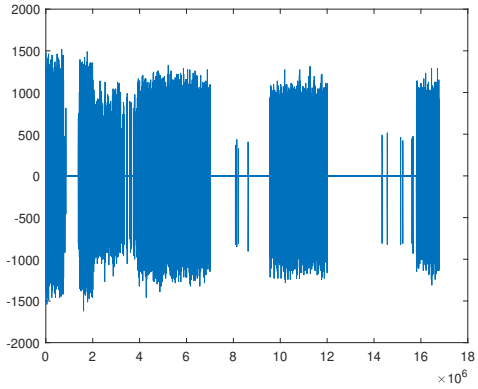


(a) OTTS input.

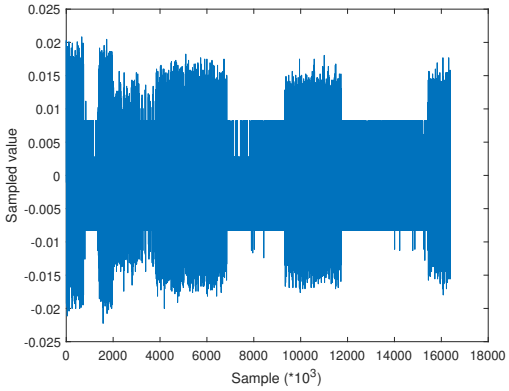


(b) OTTS input spectrum.

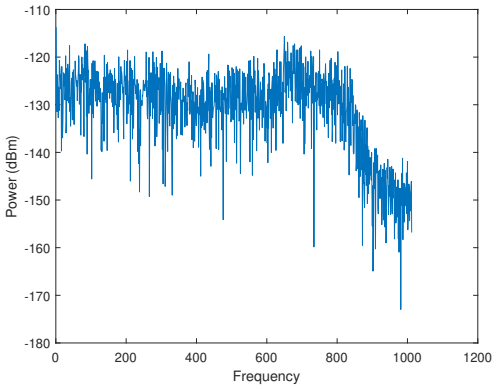
Figure 5.10: 10ms snapshot from the OTTS.



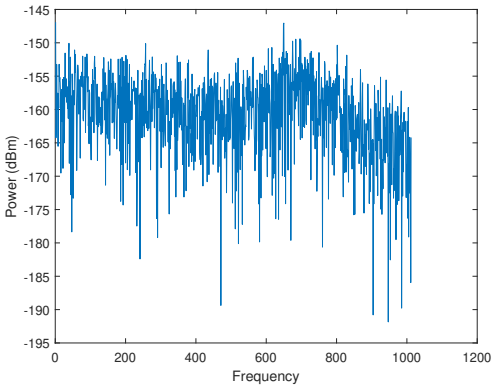
(a) Signal after compression.



(b) Signal after the compressed signal has been scaled.



(c) Spectrum after compression, using the compression algorithm.



(d) Spectrum, using 2-bit quantization. Included as a comparison.

Figure 5.11: The signal and spectrum after compression.

6

Discussion

In this chapter the results are discussed, and some suggestions for possible future developments are given.

6.1 Results

In this section, each major system variable, and its impact on the output metrics, is discussed separately.

6.1.1 Block size

As can be seen in Figures 5.2 and 5.3, the general trend is that a larger block size results in a greater amount of output data and a larger average error, when the minmax or NSAB methods for selecting depth are used. When the SD or MAD methods are used however, the output size and the errors vary less, with a slight tendency towards producing less data and error as the block size increases. Overall, these methods produce larger amounts of data but less error than the minmax or NSAB methods. This can be understood as the SD and MAD methods taking the overall characteristics of the block into greater consideration rather than just reacting on the largest, or the few largest samples. Furthermore, as can be expected, the share of the output data occupied by headers increases as the block size decreases, but even at the smallest block size of 64 samples, this share is only a few percent of the total size, making it almost negligible in all but the most extreme cases.

It is important to note, that these results are very much dependent on the exact shape of the input signal; as a consequence of the per-block processing that the algorithm employs, transients that occur frequently, or just happen to line up poorly with the block boundaries will lead to sub-optimal compression.

6.1.2 Depth selection methods

As seen in the Results chapter, the different methods to select what quantization level should be used have their own benefits and drawbacks. If lossless compression is most important then the min-max method is still capable of some compression. Minimizing data produced handled well by the MAD method and if storage is most important and errors are acceptable then this is a good choice. The over limit method provides good compression while also being easy to implement on hardware.

As for the SD algorithm, it is the worst of the methods and should not be used since it is never better than any of the others.

6.1.3 Length and shape of input signals

An important aspect of the algorithm is the amount of data it produces when compared to existing implementations. This however is not necessarily easy to calculate as this algorithm is dependent on the power of signal and how much of the range of the ADC is utilized. In the best case scenario, a signal without RFI and with a power fitting within down-mapping threshold would only produce 2 bytes of extra data for each block when compared to a 2-bit quantizer. If the block is 512 samples long, this would equal a 1.56% increase in data, $\frac{512 \cdot 2+16}{512 \cdot 2}$. Longer blocks result in the header making up a smaller portion of the total data. Also, since the input signal used to produce the results in Chapter 5 mainly consist of RFI, it may not reflect a real world scenario where RFI is more sparse.

6.1.4 Deviations in hardware simulations

In Section 5.1.5, there was a deviation in the power in the signal spectrum, an increase in almost 40%. At this moment, the source of this increase in power is unknown but is speculated to have happened due to errors in the process of re-importing and re-scaling the produced data in MATLAB.

6.2 Future work

In this section, potential improvements and comparisons will be discussed.

6.2.1 Implementation on correct hardware

At the current stage, the compression scheme has been implemented on the hardware, though not on the ADC7 but on the RFSOC 4x2. As such a new wrapper must be implemented for the correct card and our algorithm integrated into the new design. To ease integration our compression scheme has been implemented in VHDL and made into an IP. It has been pipelined to better adhere to the timing requirements.

6.2.2 Data stream and correlator

This thesis has only explored the compression of signals, not de-compression. Therefore it could be interesting to explore how a correlator would de-compress the data stream and at what data rates.

6.2.3 More accurate comparisons with existing systems

While comparisons between the data rate of the algorithm presented in this thesis and existing recorders have been shown, no such comparisons have been made for

average error or between Fourier transforms for quantization using 2-bits. This is due to the need to make assumptions on how these recorders place quantization levels. Levels placed too low will cause the ADC to saturate and placing them to high will essentially turn the ADC into a 1-bit quantizer. As such there is a need to research the way these levels are placed to show if there are benefits to this new algorithm.

6.2.4 Additional compression stages

Further compression of the output data may be possible; while the scheme presented in this thesis operates on a per-sample and per-block basis, more work could potentially be done on the output data. As an example, experiments have indicated that running the LZ-algorithm, which is a dictionary-based algorithm, on a set of output data produced by our scheme, may yield compression gains of between 5 and 10%. One concern would then be how such an algorithm could be applied to an infinite output data stream; one possibility would be to clear the dictionary and rebuild it periodically, to not deplete the memory of the host system.

7

Conclusion

This thesis has presented an adaptive signal compression scheme that has been developed for the purpose of enabling proper capture of unpredictable, transient, cosmic radiation signals while maintaining output data rates at a minimum. As shown by the simulation results, the scheme is successful in performing this task.

However, while the initial goal of implementing the compression scheme on FPGA hardware has not entirely been fulfilled, the groundwork has been done. The compression scheme has been implemented in VHDL as an IP and what remains is, in essence, to improve and test the wrapper and then connect it to the recording systems at the Onsala space observatory.

Nevertheless, the compression scheme has been shown to have the potential to significantly improve the ability of current VLBI systems to capture and record the full dynamic range of signals as well as providing a solution to overcome the ever-changing RFI environment.

Bibliography

[1] D. R. Lorimer, "A decade of fast radio bursts," *Nature Astronomy*, vol. 2, no. 11, pp. 860–864, 11 2018. [Online]. Available: https://doi.org/10.1038/s41550-018-0607-9

[2] F. Kirsten, M. P. Snelders, M. Jenkins, K. Nimmo, J. van den Eijnden, J. W. T. Hessels, M. P. Gawroński, and J. Yang, "Detection of two bright radio bursts from magnetar sgr 1935 + 2154," *Nature Astronomy*, vol. 5, no. 4, pp. 414–422, 4 2021. [Online]. Available: https://doi.org/10.1038/s41550-020-01246-3

[3] J. Zhang, M. Temmer, N. Gopalswamy, O. Malandraki, N. V. Nitta, S. Patsourakos, F. Shen, B. Vršnak, Y. Wang, D. Webb, M. I. Desai, K. Dissauer, N. Dresing, M. Dumbović, X. Feng, S. G. Heinemann, M. Laurenza, N. Lugaz, and B. Zhuang, "Earth-affecting solar transients: a review of progresses in solar cycle 24," *Progress in Earth and Planetary Science*, vol. 8, no. 1, p. 56, 10 2021. [Online]. Available: https://doi.org/10.1186/s40645-021-00426-7

[4] S. Wolfram, *Chapter 10, Section 5, Notes.* Wolfram Media, 2019.

[5] G. Swart. and et. al, "Ska1-mid telescope design," Nov. 2019, https://indico.skatelescope.org/event/551/attachments/5836/8583/7._SKA1-Mid_Telescope_Design_-_Gerhard_Swart.pdf.

[6] *ELECTROMAGNETIC METHODS OF LIGHTNING DETECTION.* World Meteorological Organization, Geneva, Switzerland, 2014 edition, updated 2017.

[7] C. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[8] T. Wescott, "Sampling: What nyquist didn't say, and what to do about it," Aug. 2018, https://www.wescottdesign.com/articles/Sampling/sampling.pdf.

[9] D. M. Pozar, *Microwave and RF wireless systems.* Wiley, 2001. [Online]. Available: https://search.ebscohost.com/login.aspx?direct=true&db=cat09075a&AN=clpc.oai.edge.chalmers.folio.ebsco.com.fs00001000.de28407a.efc5.457f.a6b2.beb8ed776b5d&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds

[10] *RF digitizer with 6.5 GHz bandwidth*, SP Devices, 12 2022. [Online]. Available: https://www.spdevices.com/products/hardware/12-bit-digitizers/adq7wb

[11] B. Jia and X. Chen, "A fast radio burst (frb) capture system for the Onsala space observatory." ″, 2021. [Online]. Available:

https://search.ebscohost.com/login.aspx?direct=true&db=ir01625a&AN=cst.20.500.12380.304384&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds

[12] *RF system-on-chip*, Xilinx. [Online]. Available: https://www.xilinx.com/support/university/xup-boards/RFSoC4x2.html

[13] "Next generation very large array," 2023, https://ngvla.nrao.edu/.

[14] "Square kilometre array," 2023, https://www.skao.int/.

[15] A. R. Thompson, J. M. Moran, and J. Swenson, George W., *Interferometry and Synthesis in Radio Astronomy, 3rd Edition*, 2017, pp. 316–326.