2020

Development of a Software-Defined Underwater Acoustic Communication System

Zijian Zhu

DEVELOPMENT OF A SOFTWARE-DEFINED UNDERWATER ACOUSTIC COMMUNICATION SYSTEM

By

Zijian Zhu

A REPORT

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In Electrical Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY

2020

This report has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Electrical Engineering.

Department of Electrical and Computer Engineering

Report Advisor:	*Dr. Zhaohui Wang*
Committee Member:	*Dr. Aurenice M. Oliveira*
Committee Member:	*Dr. Anthony Pinar*
Department Chair:	*Dr. Glen E. Archer*

Table of Contents

List of figures

Acknowledgements

I would like to thank Dr. Zhaohui Wang for being a wonderful advisor to me, she has been super supportive and patient during my master study. When I had doubts in myself, she was one who motivated me keep on going on my journey in the world of communication systems.

I want to show my appreciation to Dr. Aurenice M. Oliveira and Dr. Anthony Pinar for reviewing my report and providing valuable feedback.

I would also thank my friends and colleges in our research group, who have been always on my side when I had any obstacle in my way during my master study, they are best teammates and best friends to me.

Finally, I would like to show my appreciation to my parents and grandparents on the different side of the planet, even though they cannot be here with me physically at Michigan Tech, their love and support has been real, I could never do this without their support.

List of Abbreviations

GR – GNU Radio

GRC – GNU Radio Companion

USRP – Universal Software Radio Peripheral

UHD – USRP Hardware Driver

SDR - Software Defined Radio

ADC – Analog to Digital Converter

DAC – Digital to Analog Converter

RRC – Root Raised Cosine Filter

BER – Bit Error Rate

LMS – Least Mean Square

DD – Decision Directed

FIR – Finite Impulse Response

QPSK – Quadrature Phase Shift Keying

IF – Intermediate Frequency

Abstract

This report started with a brief history and recent development of underwater acoustic communication systems as well as software-defined radio technologies. Then, some challenges from underwater acoustic channels and available underwater acoustic communication modems are discussed. After finished introducing the basics of SDR and GNU Radio, a detailed description of implementing a software defined acoustic communication system in GNU Radio are presented, along with some key concepts of the system.

Then, some hardware specifications are presented, following by a detailed documentation on a software defined acoustic communication system experiment with a host computer, a USRP, an acoustic hydrophone and a hydrophone. At the end of Section 4, the results of the experiment are discussed.

Lastly, conclusions of this report are made. Some possible directions for future work are suggested in Section 6.

1 Introduction

The need for information is increasing for both individuals and society as a whole. New radio communication technologies have been added to new devices, such as Wi-Fi 6, Bluetooth 5.1, and 5G. In the meantime, software-defined radio plays an important role in developing new radio protocols and technologies, as it uses software to define hardware components like filters, detectors, and amplifiers, it would simplify the process of radio development. GNU Radio (GR) is an open-source Software Defined Radio (SDR) tool kit widely used by engineers in the field of signal processing and radio development. This report is focusing on using GR to develop a single carrier communication system with USRP that is suitable for underwater acoustic communication.

1.1 Underwater Acoustic Communications

Compared with radio communications, underwater acoustic communications have more challenges. Which have been brought up and investigated by researchers in the past decades.

The first underwater acoustic communication system appeared when the military was trying to implement a communication system on manned submersibles. They used a single-sided band modulation with a carrier frequency from 2-15kHz. Since it was all analog, this communication can only carry voice or Morse code, despite its poor signal fidelity. However, because of its simplicity, it is still used for underwater communication today.

There are two main problems in underwater communications. Firstly, due to the physical property of water, high frequencies wildly used in radio communications are not available for underwater communications as the energy of high frequency acoustic signals attenuates significantly in water bodies. This property results in low data rates in underwater acoustic communications.

The second problem that researchers have encountered is the multipath effect. As the acoustic signal propagates through the channel, acoustic waves often reflect off the bottom of the riverbed and the surface of water/ice[1]. The multipath effect is more severe for acoustic communications because the sound speed is a lot slower than the propagation speed of radio waves. Also, this slower propagation speed would introduce significant Doppler effect when nodes in an acoustic communication system are having small displacement. Therefore, researchers have been trying to model the underwater acoustic channel using different equalization techniques. However, due to the complexity of underwater environments, there is not a universal model for every unique situation[2].

Apart from the issues of an individual link between two nodes in underwater acoustic channels, some researchers at Michigan Tech also have done some great works regarding UWA networks [3, 4].

1.2 Underwater Acoustic Modems

There are several manufacturers are manufacturing acoustic modems for commercial and/or research proposes. They all have different specifications in range and data rates. However, these specifications are tied closely to a specific underwater acoustic channel. For instance, the multipath effect would be less significant in a vertical channel where the signal travels from a transmitter close to the surface to a receiver at a deeper depth. The channel equalization needed for this case would be minimum. Also, the range and data rate specification would better compare to the horizontal channels[5].

During the years of underwater communication experiments have been conducted at Michigan Tech, a specific underwater acoustic modem was used. It allows researchers to send and receive underwater acoustic signals through a serial port built into it. It is highly integrated with plenty of features; however, it is not as flexible as software-defined radio (SDR)[6]. In Table 1, a simple comparison between traditional underwater acoustic modems and USRP is presented. While the USRP has a higher degree of flexibility, it tends to consume more power than acoustic modems, but it would not be an issue when the communication node is stationary with no power limitation.

	USRP	Acoustic Modems
Flexibility	High	Low
Power consumption	40W	25W

Table 1. Comparison between USRP and acoustic modems.

Besides power consumption, each underwater acoustic modem is tuned for a specific channel. Some well-funded research team would purchase various modems for different channel conditions, while others have to pick the one that they would use the most. Therefore, using an SDR to adapt different channel conditions with only one piece of hardware would be more cost effective.

In other words, developing a communication system on SDR for underwater acoustic communication systems will help researchers to be more versatile on what the signal is being transmitted, how are they being transmitted (modulation) and how are they being decoded from waveforms (detection, demodulation, and equalization).

1.3 Motivation

As the reasons mentioned in sections above, researching in underwater acoustic communications are currently facing challenges from both underwater channels as well as inflexible underwater acoustic modems. For the channels, the low propagation speed of acoustic waves in water makes channel equalization an essential task when developing underwater acoustic communication systems. Also, because of the inflexibility of currently available underwater acoustic modem on the market, different modems have different performances under different channel conditions. This could be a challenge to researchers who would like to test a communication system in different channels.

While all these challenges above exist, this report will focus on building an SDR-based underwater acoustic communication system with USRP.

1.4 Organization of This Report

Firstly, Section 2 will go over some basics of SDR and GR, then moved on to the implementations of a QPSK audio modem on GR platform, following by some details on specific modules and components in GR. In the end, the results of GR simulation are presented.

In Section 3, some hardware specifications are presented for later experiment setup. In Section 4, some details of how to tune parameters in a GRC flowgraph for hardware experiment. Some experimental results are presented at the end of Section 4.

Finally, the conclusion, references are at the end of this report.

2 System Implementation in GNU Radio

2.1 A General Communication System

Figure 1 is the conceptual diagrams of a general communication system. The left part of Figure 1 shows the general transmitter converting digital information to analog bandpass signal. The right part of Figure 1 shows the general setup of a single carrier digital receiver. It takes the bandpass signal and convert it to the baseband signal. The receiver would first try to synchronize the clock with the received signal, then it would estimate the channel based on the received signal. After an estimation was obtained, the receiver would equalize the channel based on the channel estimation. Then, a phase lock loop would lock the phase. Finally, the receiver would demodulate the recovered baseband signal before the texts were converted from raw bytes.

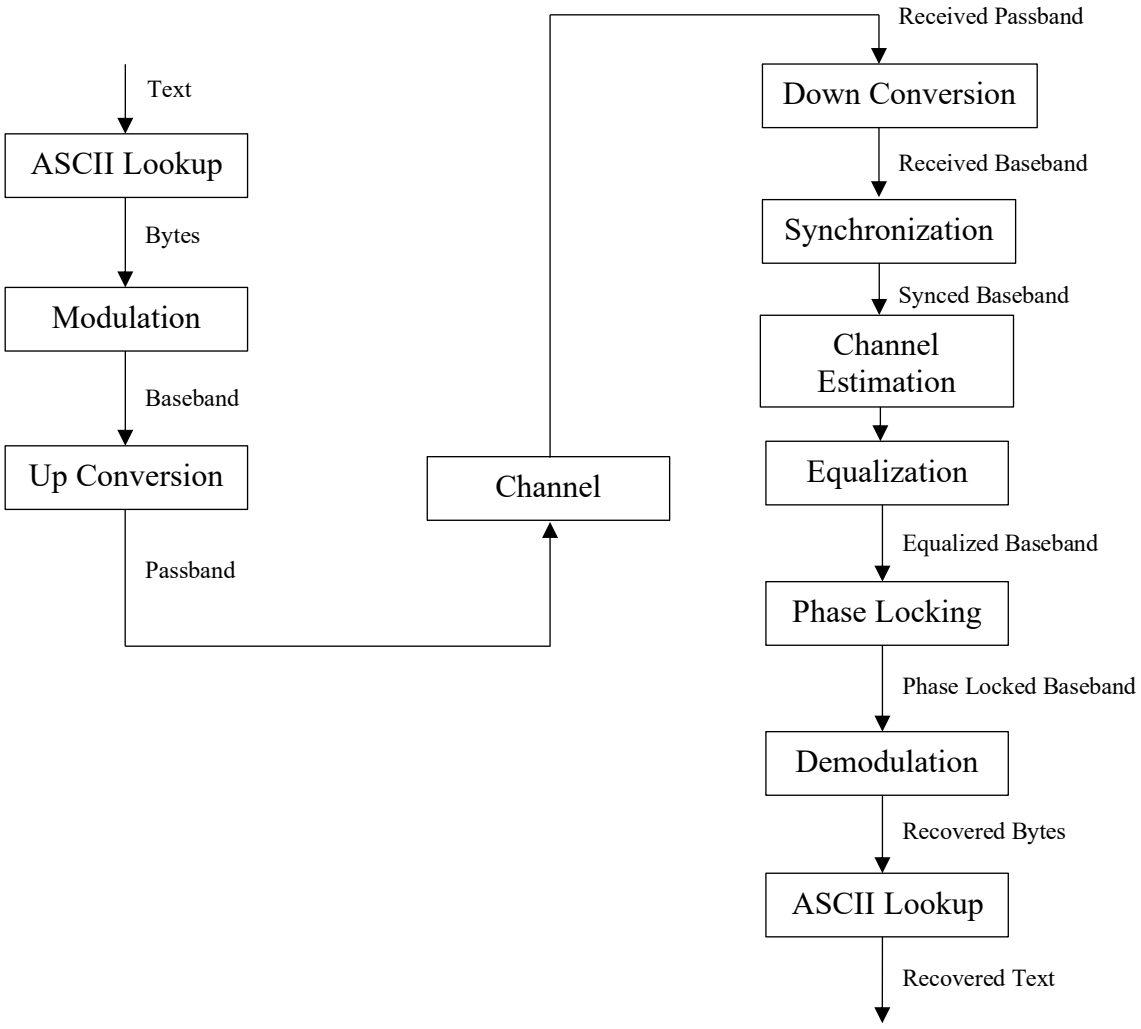

Figure 1. Conceptual diagram of the communication system implemented.

The modulation scheme used for this project is QPSK. This modulation scheme uses two sinusoid waves at the same frequency with 90 degrees of phase difference. These two sinusoid functions are called basis function. The mathematical representation is:

$$f_n(t) = \sqrt{\frac{2}{T_s}} \cos\left(2\pi f_c t + \Delta\emptyset\right)$$

Where n is the index of the basis function, T_s is the symbol duration, $f_s$is the carrier frequency, and $\Delta\emptyset$ is the phase shift. With four different combinations of 2 basis functions at a time, 2 binary bits can be represented in each case.

To receive QPSK signals, corresponding basis functions will be multiplied to the passband signal. After a lowpass filter removed the high frequency in the IF signal, the original QPSK baseband signal can be recovered.

2.2 SDR and GNU Radio

From Section 2.1, a communication system is made by various radio components. With SDR, researchers and developers would be able to implement different communication systems using a generic radio hardware. Unlike using a communication modem, SDR allows researchers to specify each parameter on multiple levels of a communication system, from carrier frequency to packet format. There are plenty of them on the market as of today, but USRP, also known as Universal Software Radio Peripheral, made by Ettus Research, has a wide variety of daughterboards selections to accommodates different frequency bands. The versatility of USRP is impressive, therefore it is chosen for this project.

GNU Radio is an open source project for SDR development. It is written in C++ and Python. The signal processing is mostly done by C++ for better efficiency, and Python connects those C++ signal processing libraries together to form communication systems. GNU Radio also offers a graphical user interface in GNU Radio Companion (GRC). It is referred as flowgraph diagrams. However, using the graphical interface does not mean users would lose control over the design of the communication system they are working on. User can use existing signal processing blocks or Out-of-Tree modules they made by themselves to develop communication system in GRC. Also, the GRC still let users to call Python functions in all the blocks. It is a good starting point for communication system engineers who are new to SDR.

In a GNU Radio flowgraph, the basic elements are blocks. As an example, shown Figure 2, two signal sources are connected together by a multiply block, which as the name implies, multiply two signals into one single data stream. Which is an ideal mixer from the radio hardware aspect. The concept of data stream is crucial in GNU Radio flowgraphs. GNU Radio is similar to a factory taking in raw material and producing products. In this example, the data stream from these to signal sources is the raw material, after the multiply block, the data stream then goes to two different sinks, wav file sink and audio sink. This flowgraph, therefore, produces one .wav file and plays the two-tone audio through a speaker. Which is essentially a sound wave consist of two different frequencies.

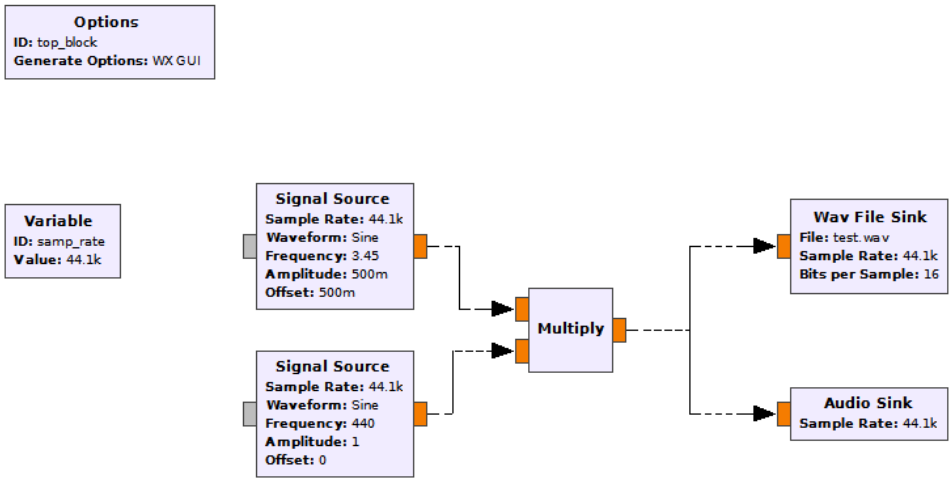

Figure 2. Two Tone GRC Sample. Image source: https://wiki.gnuradio.org/index.php/Guided_Tutorial_GNU_Radio_in_Python

2.3 Implementation of a QPSK Audio Modem

2.3.1 GRC Flowgraph

Following the Dr. Scher's example [10], a new flowgraph of QPSK audio modem was made. The first issue encounter was the packet decoder/encoder. Online documentation pointed out that these blocks are no long supported in GRC as they drop bits during transmissions, therefore they are no longer exist in GRC. From an example GR flowgraph provided with the software, it points out a new way to format a packet using Protocol Formatter and Formatter Object. As shown in Figure, the packet formatter takes the input data stream and generates a header for this packet based on a given formatter object in GNU Radio. This format object mainly specifies the header for the packet.

The header has three parts. The first part is the access code mentioned above, it is the beginning of each packet. This is important because it will be used for packet and symbol synchronization. The second part is a 16-bit long binary number representing the packet length, the third part is the same packet length repeated. After the packets are generated, the data stream then will be sending to a constellation modulator. This is where the binary data stream will be converted into constellation symbols according to a given constellation object, which is QPSK in this case.

Baseband data stream will be then up converted to band pass signal by multiplying a sine and cosine wave on the carrier frequency. They will be added together before being transmitter in the channel.

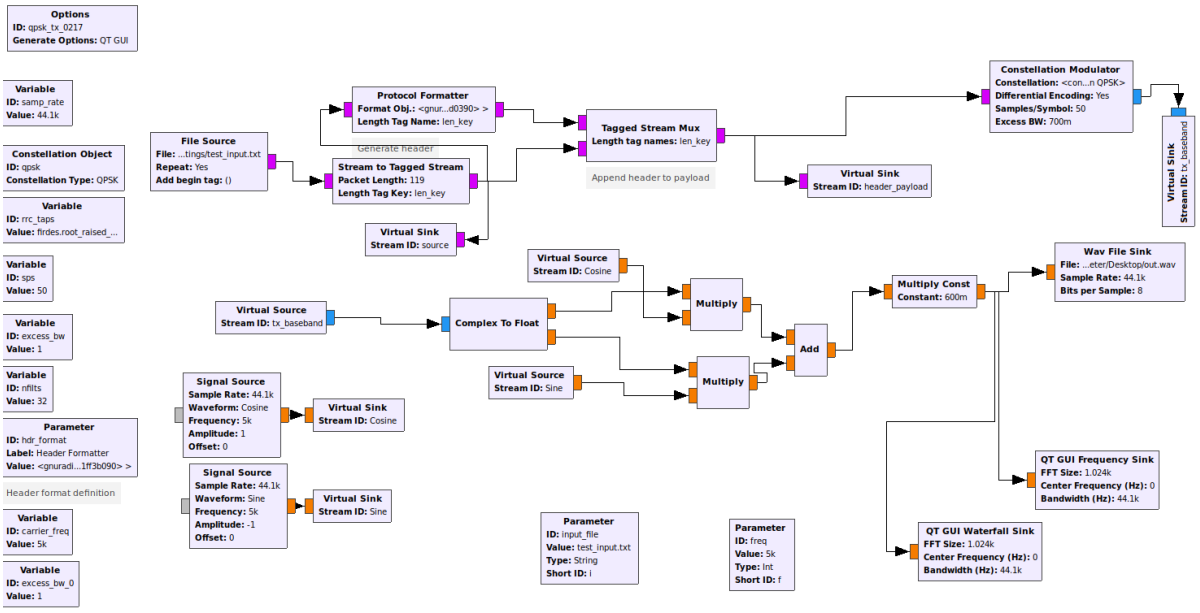

Figure 3. QPSK audio modem in GRC (Transmitter).

The audio sink received the transmitted waveform from the channel. The first operation is down conversion. A following Correlation Estimator block will perform a cross-correlation to the baseband signal. This will help the receiver to identify the beginning of the signal. In another word, this block performs packet synchronization. This correlation estimator will take a variable which is a waveform of a predefined vector modulated with a predefined modulation scheme. In this case, the waveform here is the access code modulated in QPSK. With this waveform, the correlation estimator runs a cross-correlation against the received baseband signal, then generates tags on the data stream to inform other blocks where is the starting point of current packet.

A polyphase clock synchronization is following behind for symbol clock synchronization, it takes derivative of the baseband signal, which would help the system to determine the correct sampling point for each symbol. After the synchronization, an equalizer is used to equalize the channel, to compensate uneven frequency response of the channel as well as remove the multipath effects. After that, a Costas loop is following for phase locking.

The demodulated binary bit stream coming out of constellation demodulator then passed through a differential decoder as differential encoding was used in the constellation modulator on the transmitter side. However, each 8-bit byte here only contains 2 bits while the rest of the bits are zeros, since each symbol in QPSK scheme representing 2 bits of data. Therefore, the 2-bit-per-byte data stream here is then being passed to an unpack k bits block, where the data stream can be unpacked to 1 bit per byte.

Bit slipping happens when the receiver has recovered the correct order of the bits but fails on arranging them in right slots for the corresponding slots in a byte. To prevent bit slipping, another correlation block is used here to perform symbol synchronization mentioned before. This block would look for the predefined 64-bit access code in the incoming bit stream, then put them

into the right spots for reconstructing the original bytes. The following repack bits block then takes these 1-bit-per-byte data stream, repack them into bytes on the ASCII table.

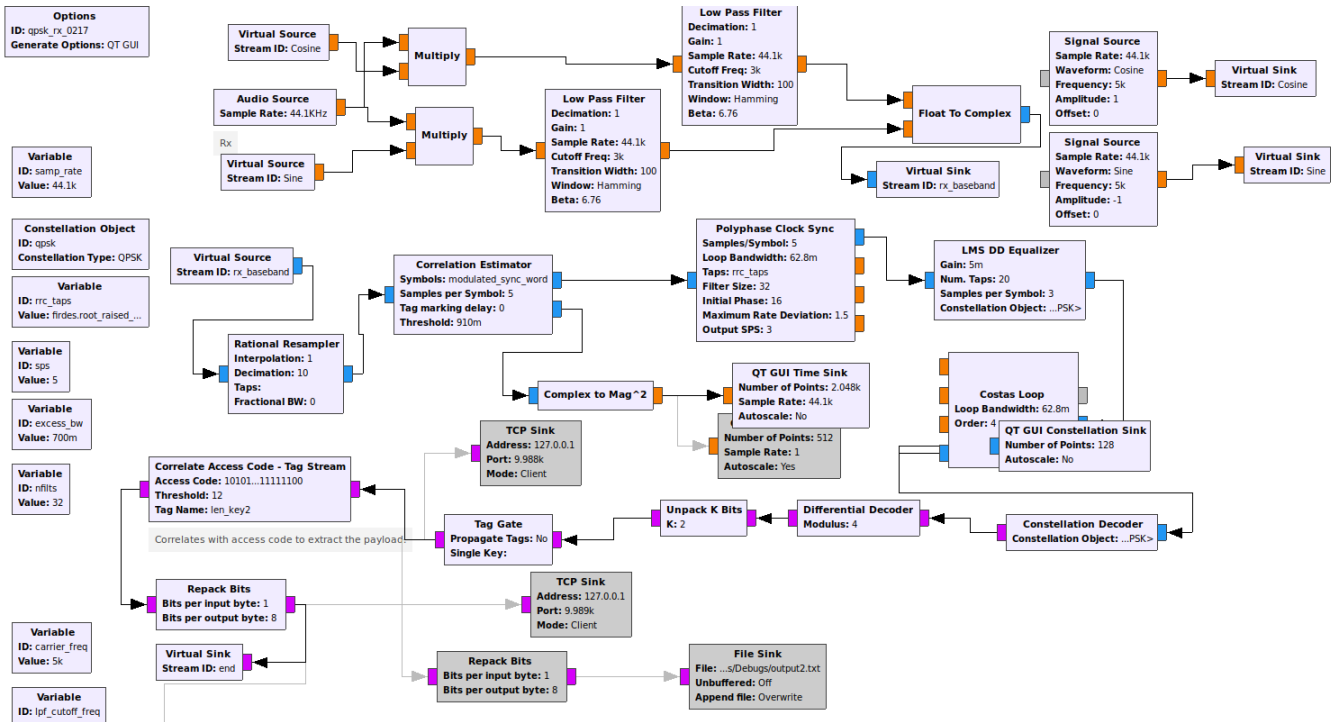

Figure 4. QPSK audio modem in GRC (Receiver).

This system is using QPSK modulation scheme, with a sampling frequency of 44.1kHz and 50 samples per symbol. Leads a bandwidth of 1.764kHz and a bit rate of 1.764kbit/s.

2.3.2 GRC Blocks

Before going over the first working QPSK GRC flowgraph, here are some explanations on the functionalities of used blocks:

- **Protocol formatter**: takes in payload and generate a header for it.
- **Stream to Tagged Stream**: add tags to a none tagged data stream, this can be seen as delimiters for packets.
- **Tag Stream Mux**: takes two tagged streams and append the second one to the end of first stream.
- **Constellation Modulator**: modulate digital baseband signal to constellation symbols.
- **QT GUI Waterfall Sink**: shows a waterfall graph of the input stream.
- **QT GUI Frequency Sink**: shows the spectrum of the input stream.
- **WAV File Sink**: save input stream to a .wav audio file.

2.4 Key Components in the Implemented Communication System

In this section, some key components will be introduced as they are crucial parts in the following sections.

2.4.1 **Polyphase Clock Sync**

Polyphase Clock Sync is a block mostly performs symbol synchronization. It uses a filter bank to find derivatives of different phases of the signal. Looking at the Figure 5, a blue curve is the original input baseband signal. The other curves are the derivatives of the baseband signal with various phase offsets. In this example, there are five different phase offsets and only one is the correct offset as it has a derivative of 0 at the highest point of the blue curve, which is the optimal sampling point for this symbol[7].

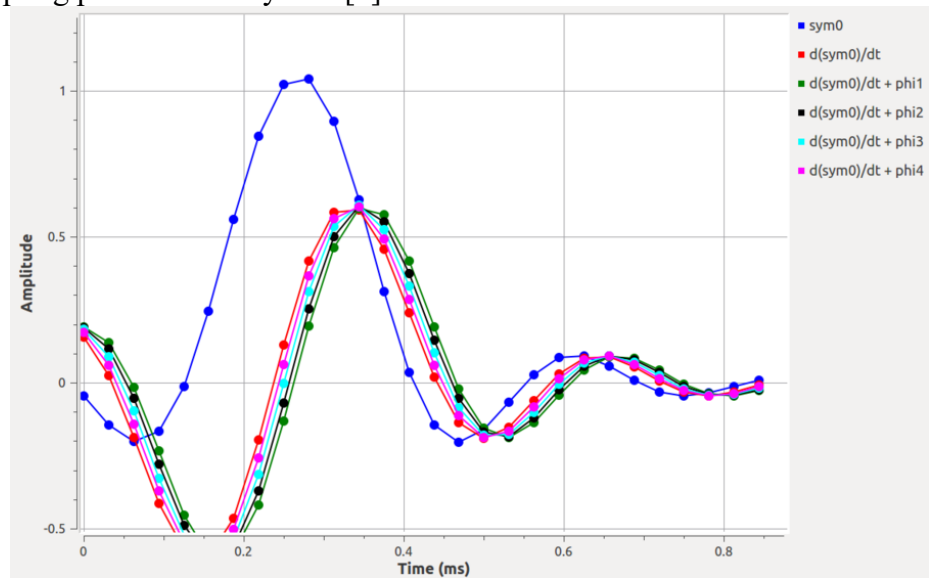

Figure 5. Polyphase Clock Synchronization. Image source: https://wiki.gnuradio.org/index.php/Guided_Tutorial_PSK_Demodulation

2.4.2 **LMS DD Equalizer**

This block is used to equalize the channel response by utilizing a Decision-Directed Equalization method. It uses an internal FIR filter to cancel out the channel effects.

It has three major parameters: The first one is gain. It represents how fast the equalizer updates its internal FIR filter towards the convergence. The second parameter is the length of the filter, which represents the estimation of the channel length. The third parameter is the constellation object. From Figure 6, an estimation is made by a slicer, then the difference between the estimation and received signal, i.e. error is multiplied by the gain to be used as a new coefficient in the internal FIR filter[8].

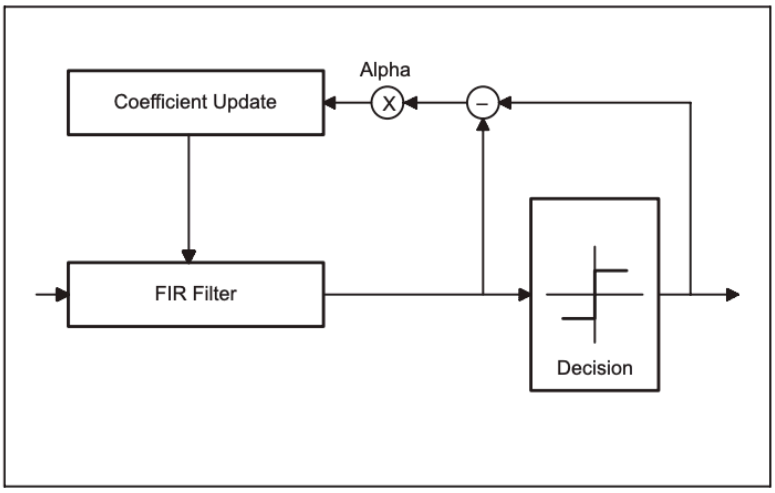

Figure 6. Block diagram of Decision-Directed Equalizer.

2.4.3 **Costas Loop**

Costas Loop is a phase lock loop to lock the phase of the bandpass signal. The concept of this phase lock loop is assuming the local oscillator is generating a sinusoid wave at the carrier frequency of the bandpass signal, and it has a constant phase error between this sinusoid signal and the bandpass signal. From Figure 7, $v_1(t)$ is going to be significantly larger than $v_2(t)$, as the value of phase error θ_e is close to zero.

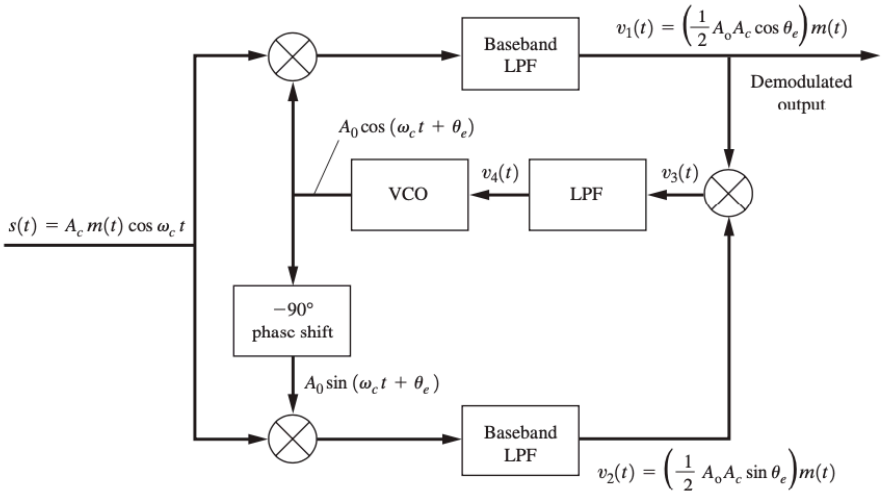

Figure 7. Illustration of Costas Loop[9].

2.4.4 **Up and Down Conversion**

The last key component of the communication system shown in Figure 8 is not necessarily a block in GRC. Instead, it is a signal processing approach to mitigate the problem where an ADC cannot sample the bandpass signal fast enough to avoid aliasing, i.e. an ADC cannot sample higher than a rate doubles the maximum frequency in the signal. This is where up and down conversion shines.

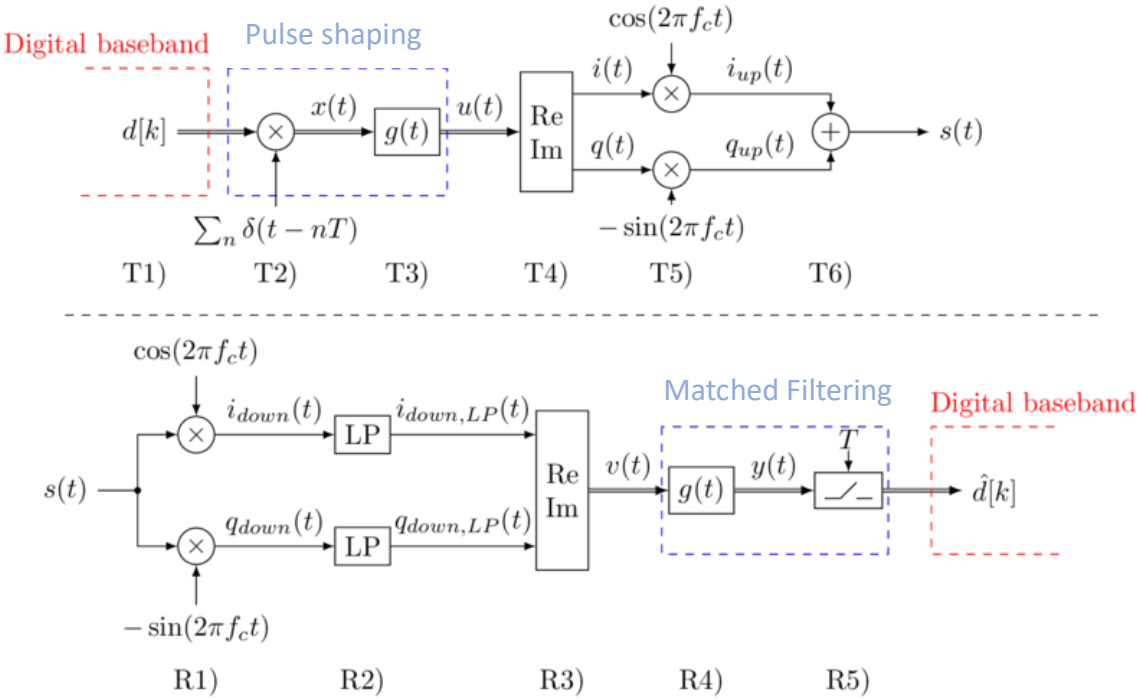

Figure 8. Illustration of up & down conversion. Image source: https://dspillustrations.com/pages/posts/misc/baseband-up-and-downconversion-and-iq-modulation.html

From Figure 8, up and down conversion is the first thing happened to the bandpass signal (T1). The idea of it could be not so intuitive at the first, but it can be explained by math easily. Take the carrier wave is a vector, it is multiplied by the baseband signal in the transmitter. Then the bandpass signal was passed through a channel. Now, the received bandpass signal is consist of the spectrum of the carrier wave and the spectrum of the baseband signal (R1). As mentioned before, the carrier frequency is a vector. The dot product of a vector and itself is going to be the magnitude of this vector. This means the result of a baseband signal multiplied by the carrier wave, a copy of the original baseband signal could be recovered. This multiplying process will produce intermediate frequency, where it contains the baseband signal with some artifacts on high frequencies. Since the baseband signal has a center frequency of 0Hz, a low pass filter is applied to here (R2). Finally, a recovered baseband signal will be acquired.

The binary bit stream in digital baseband will be converted to constellation symbols. After that, the constellation symbols will be sent to g(t) for pulse shaping (RRC – Root Raised Cosine Filter), this will smooth out the sharp edges on digital baseband signal.

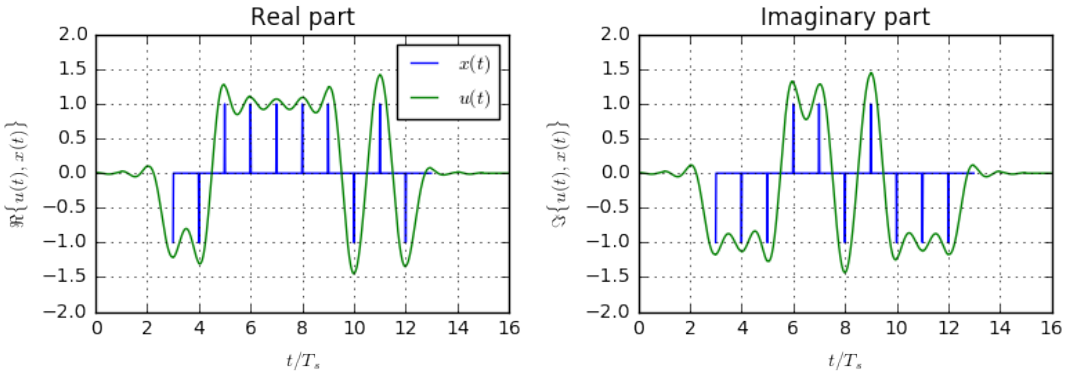

Figure 9. Pulse shaping of a QPSK signal. Image source: https://dspillustrations.com/pages/posts/misc/baseband-up-and-downconversion-and-iq-modulation.html

2.5 GRC Simulation Results

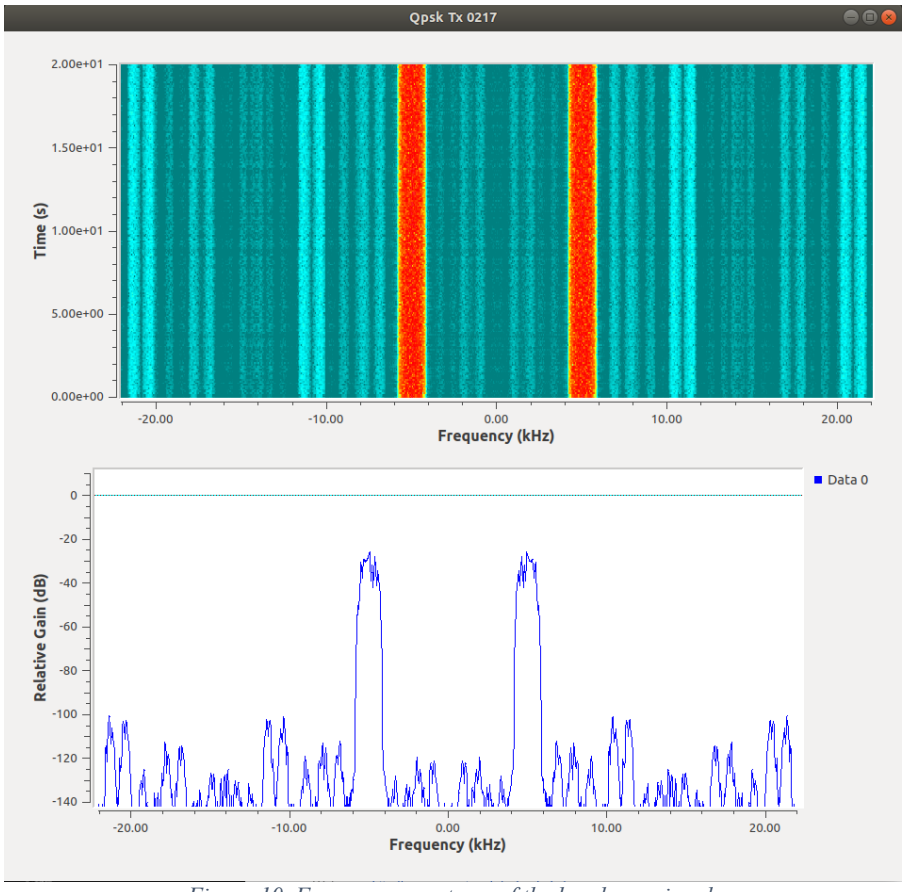

Figure 10. Frequency spectrum of the bandpass signal.

Figure 10 shows the visual representations of bandpass signal on the transmitter. The bottom graph is showing the spectrum of the bandpass signal, where there are two peaks at the carrier frequency of 5kHz. On the top, a waterfall graph showing the spectrum with the respect of time. However, it could be confusing that information stream is not visible on the waterfall graph,

since it looks like two straight lines with smooth edges. But it is due to the low resolution on the waterfall graph which cause the missing representation of data stream.

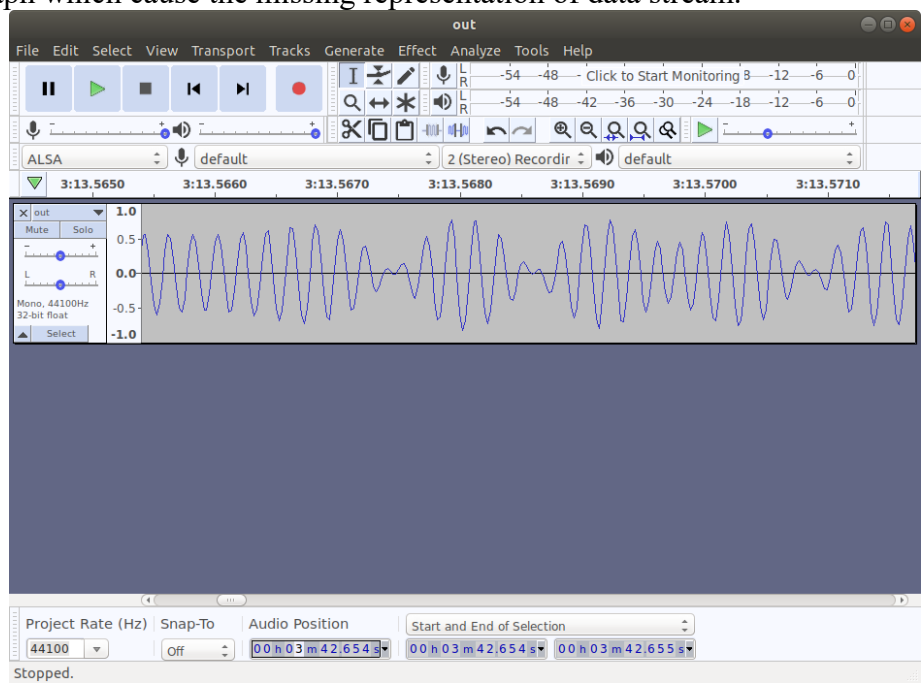

Figure 11. Generated waveform of the bandpass signal.

Figure 11 shows a time domain QPSK signal in Audacity software. As the QPSK waveform generated by the transmitter are being saved to a .wav file. Viewing the waveform in Audacity would reveal problems such as audio clipping, where the waveform amplitude exceeded the hardware limitation.

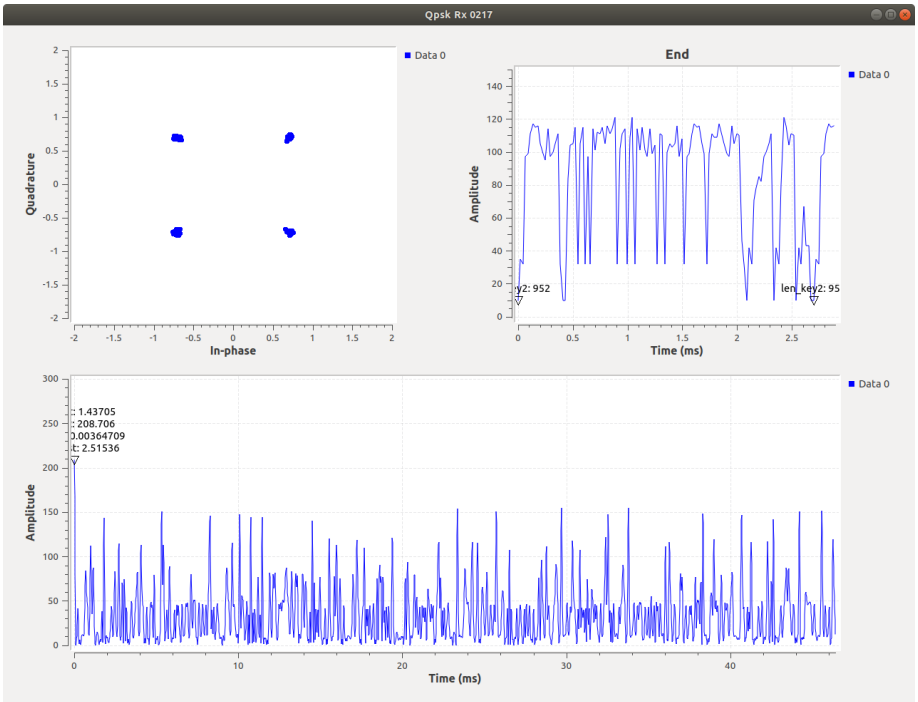

Figure 12. Receiver figures.

In Figure 12, a clear constellation map is shown on the top left corner, a time domain output byte stream is shown on the top right corner, notice every value is above zero since the ASCII value of English letters are 0b0xxxxxxx. At the bottom, a correlation graph is shown, the peak value on the left indicates the correlation estimator has found the packet header using a matched filter.

After finished with running an acoustic communication system solely on a host computer with built-in microphone and speaker, the next step will be using SDR hardware to implement an underwater acoustic communication system. In the next section, related hardware specifications will be presented. Also, the optimal way to setup UHD and GR is discussed at the end of the next section.

3 Hardware

3.1 USRP X310

With all the signal processing working for laptop audio transmission, it is time to move on to the underwater acoustic part. In this part, a software-defined radio hardware, USRP, is being used for up and down conversion.

The hardware configuration is shown in Figure 13. The host computer is connected with the USRP through an Ethernet cable. The USRP connects to the acoustic transducer through an amplifier on the Tx. Lastly, the USRP connects with the hydrophone on the Rx.

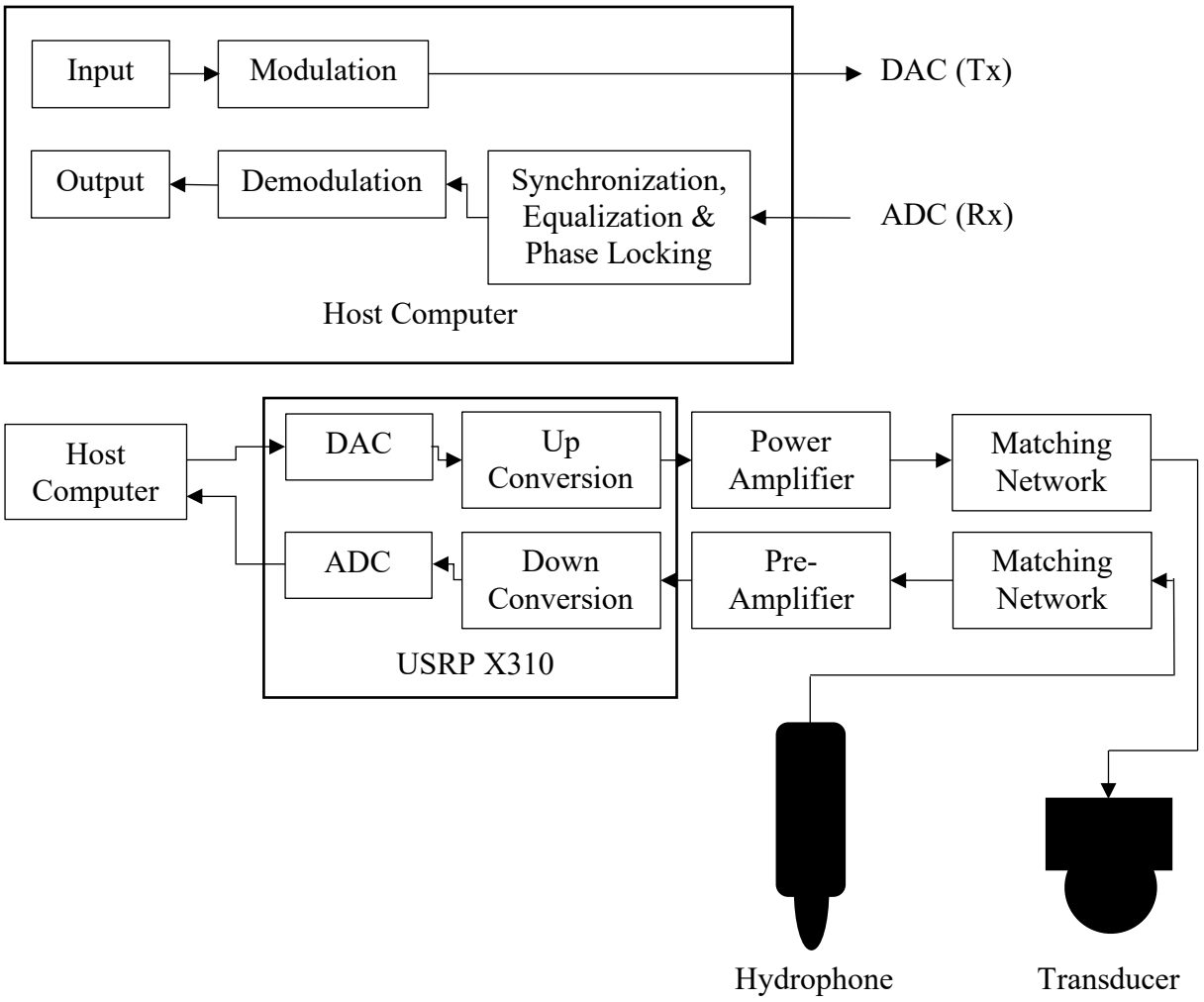

Figure 13. Hardware connection.

USRP stands for Universal Software Radio Peripheral, it is made by Ettus Research under National Instruments. The model of USRP used in this project is USRP X310, it has Kintex 7-410T FPGA built-in, with 200M samples per second of streaming capability, and a clock rate up to 200Mhz. For the RF front end, it has two LFTX and LFRX daughterboards. They are capable of transmitting and receiving signals with carrier frequencies from DC to 30Mhz.

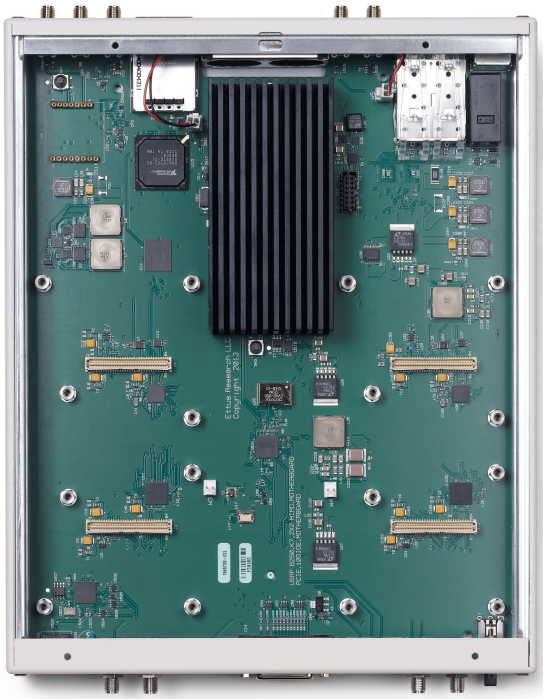

Figure 14. Internal of USRP X310. Image source: https://www.ettus.com/all-products/x310-kit/

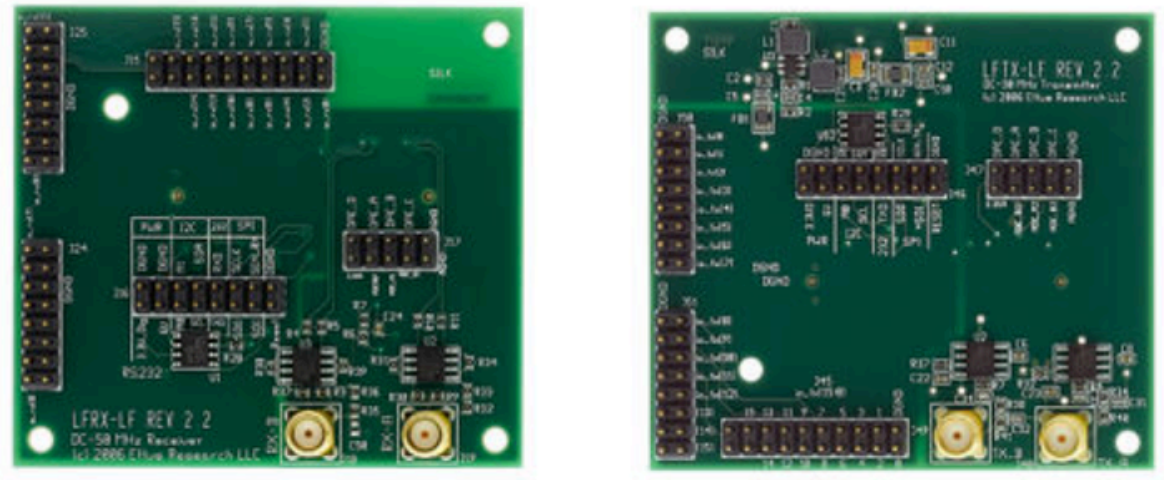

Figure 15. LFRX and LFTX daughterboards. Image source: https://kb.ettus.com/LFTX/LFRX

3.2 Acoustic Transducer and Hydrophone

To be able to transmit and receive acoustic signal in underwater environment, a pair of acoustic transducer and hydrophone is required.

3.2.1 Acoustic Transducer

Benthowave Instrument Inc (BII) is a manufacture produces underwater acoustic hardware. The acoustic transducer used for this experiment is BII-7526. To use this transducer with the USRP, a matching network (BII-6002 T1) and a power amplifier (BII-5002) is needed. The physical dimensions of this transducer are shown in Figure16. The beam pattern of the acoustic transducer is shown in Figure 17. The voltage response curve of this transducer is shown in Figure 18. Where the largest responses are located within 80-90kHz. This frequency response characteristic lead to the 85kHz carrier frequency for the implemented communication system.

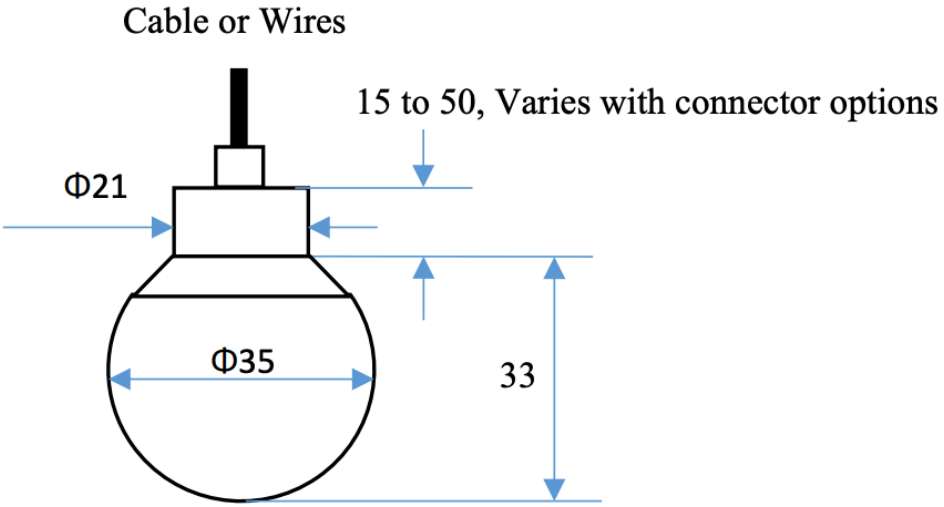

Figure 16. Physical Dimensions of BII-7526.

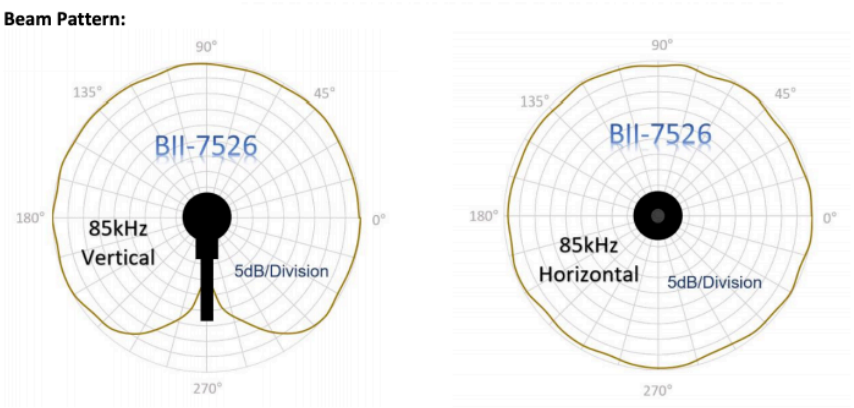

Figure 17. Beam pattern of the transducer.

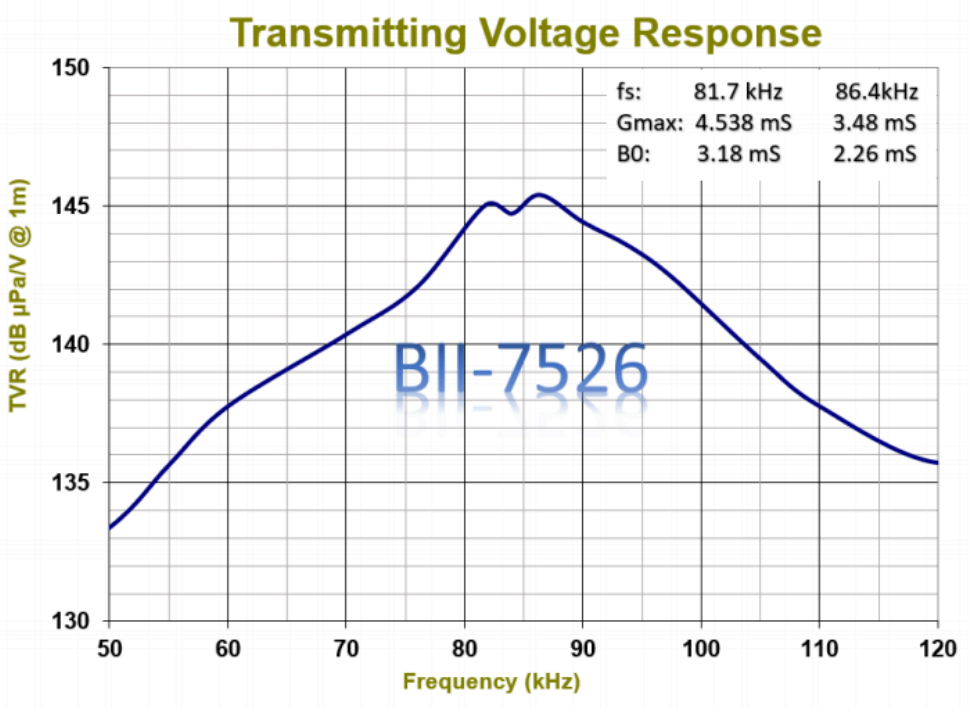

Figure 18. Transmitting voltage response of the BII-7526

BII-5002 is the linear wideband amplifier came with the BII-7526. It can be powered by a DC power supply with voltages ranging from 8 to 38 volts. The bandwidth of its outputting signal varies with the power supply voltage. It has 100kHz bandwidth with 36 volts DC, 200kHz at 24 volts DC, and 300kHz at 12 volts DC. The power supply voltage used in this experiment was 9 volts DC.

This amplifier did not come with a housing, it is built on a circular printed circuit board with a diameter of 101.6mm. Including the height of the heat sink, the height of the whole unit is 76mm. It weighs about 260 grams.

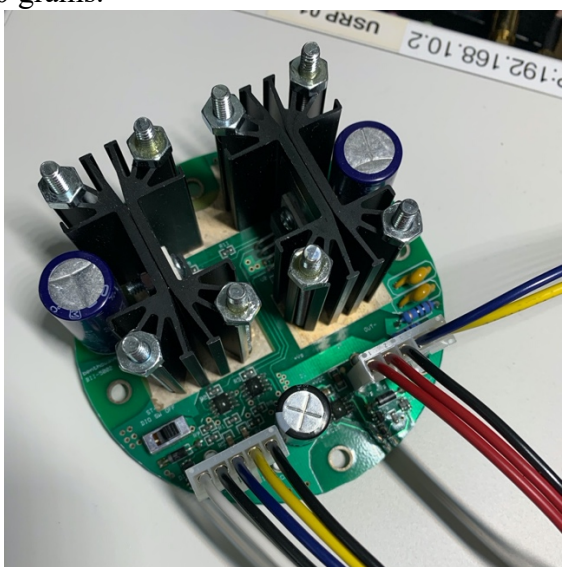

Figure 19. The power amplifier for the acoustic transducer.

3.2.2 **Hydrophone**

The hydrophone used for this experiment is also made by BII, the model number is BII-7008. It has a preamplifier built in. From Figure 21, the frequency response is quite uniform from 1Hz to 100Hz, which is suitable for signals transmitted from BII-7526. Figure 22 shows the noise spectrum of the hydrophone-preamp combination, as well as a reference noise spectrum. The noise spectrum of the hydrophone-preamp combination is quite similar to the reference noise spectrum.

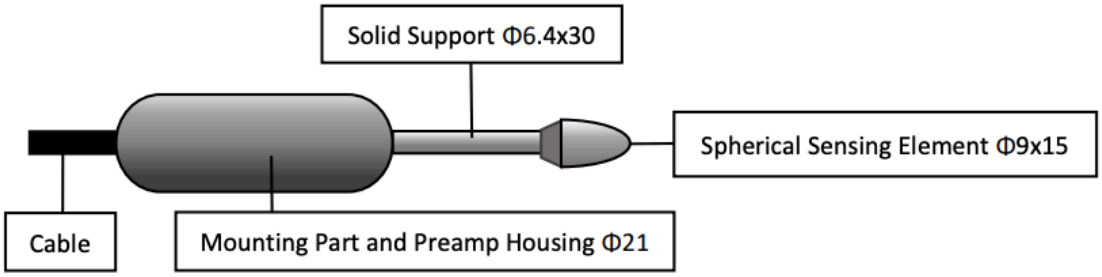

Figure 20. Physical dimensions of BII-7008.

Free-field Voltage Sensitivity (FFVS):

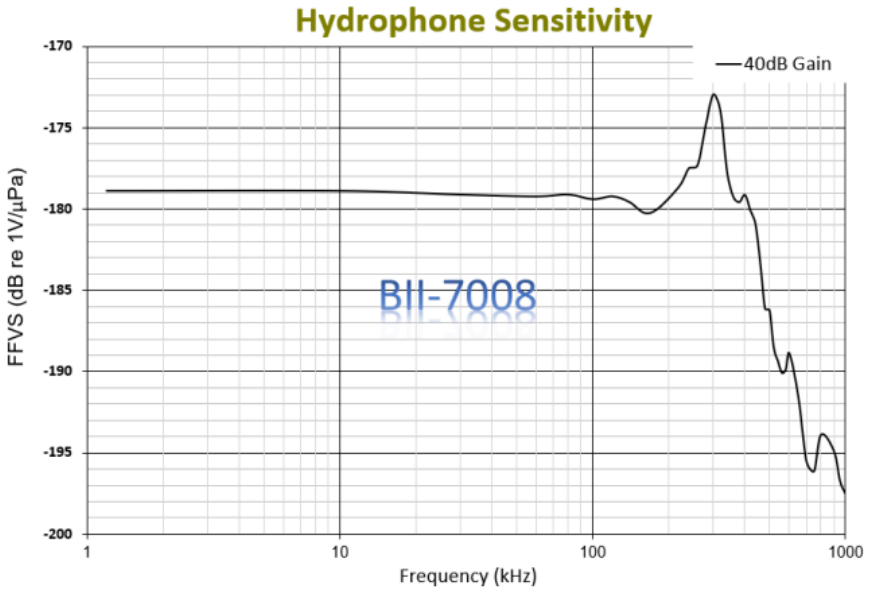

Figure 21. Hydrophone sensitivity.

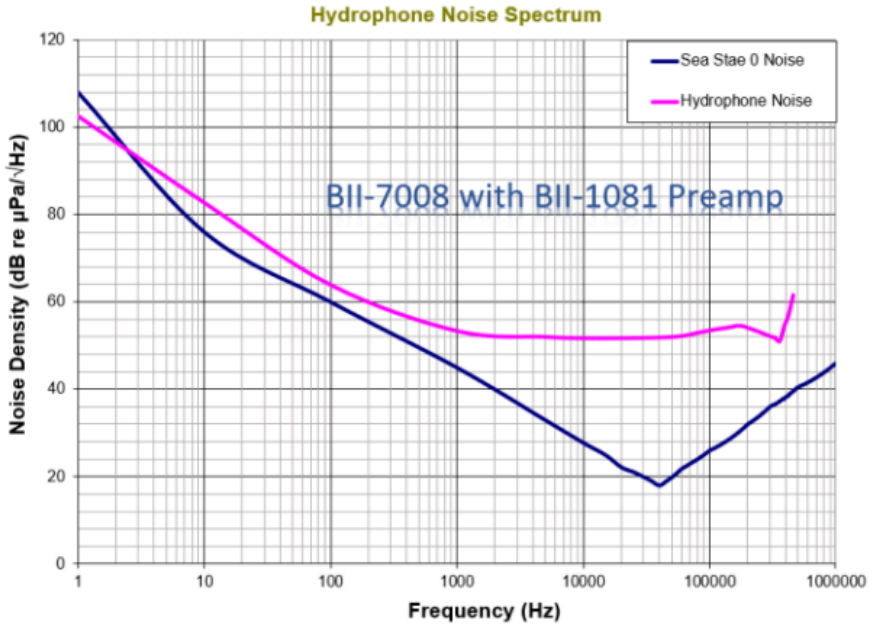

Figure 22. Frequency response of BII-7008 with BII-1081 preamp.

3.3 Setting Up UHD and GNU Radio

To begin with, the driver (UHD, USRP Hardware Driver) has to be setup before connecting the USRP to a host computer. After countless hours of reinstalling different software, turns out that the GR installed from the package management software, APT, is included with a specific version of UHD. However, in order to use the USRP with the host computer, the version of FPGA on the USRP has to match with the version of UHD on the host computer. This would not be a problem usually, but since the UHD installed in the host computer and the one came with GR are not the same, the FPGA image would not work with the UHD came with GR.

The solution to this issue is to build a specific version of UHD from source, then build a compatible version of GR. In this case, there will only be one version of UHD on the host computer. After finishing setting up the software on the host computer, a simple GRC example proofed all the hardware configurations was completed.

After finished with introducing hardware specifications, next section will be detailing the hardware setup for the experiment and experimental results. At the end of the next section, the existing issues are discussed.

4 Experiment
4.1 Experiment Setup

The experiment setup was pretty straightforward. One transmitting node is consisted with a host computer, a USRP, and a hydrophone with its amplifier, a receiver node is the same setup, but instead of hydrophone, the receiver has an underwater acoustic transducer. The block diagram in Figure 23 shows how data will flow through the system.

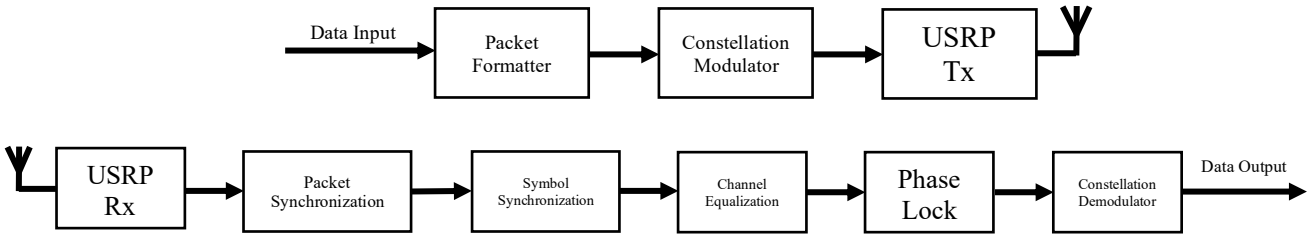

Figure 23. Connection diagram.

Before connecting the hydrophone and acoustic transducer to the USRP, a loop back test was performed. The setup is really close to the audio modem flowgraph mentioned earlier in previous sections, however, there are some minor changes.

First of all, because the USRP will perform the up and down conversion using the daughter boards, that part of the flowgraph is no longer needed, as they are replaced by USRP blocks.

Secondly, from the experimental results, the lowest sample rate accepted by USRP is 400kHz. According to the Ettus Research website, the USRP X310 supports up to 25MHz sampling rate using a Gigabyte Ethernet connection. However, due to the limitation of the host computer used in this report and the need to reduce the baseband bandwidth to work with low carrier frequency, the sampling rate was selected as 400kHz. The equation below showing the relationship between sampling rate, samples per symbol and excess bandwidth.

$$bandwidth = \frac{sampling\ rate}{samples/symbol} * (1 + excess\ bandwidth)$$

Lastly, because the sampling rate of the system increased from 44.1kHz for soundcard to 400kHz for USRP, the bandwidth of baseband signal was increased alone with it. From the datasheet of the acoustic transducer used in this experiment, the optimal operating frequency range for the acoustic transducer is from 80 to 90 kHz, therefore, the carrier frequency was picked as 85kHz. Because the bandwidth of the baseband signal has to be way less than the carrier frequency, the samples per symbol has to increase to 100. This lead a baseband bandwidth of 5.4kHz, which is far less than the carrier frequency, and the data rate is 8kbit/s.

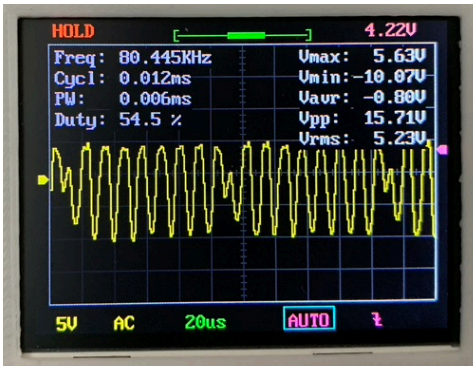

Figure 25. Amplifier output.

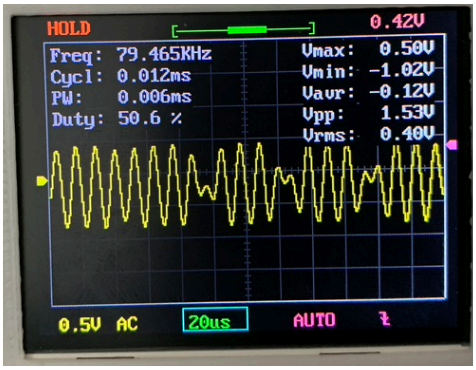

Figure 24. Hydrophone output.

After making sure the loopback test is passed, an oscilloscope was used to examine the signals from different parts of the system to see if they meet the expectation. Figure 24 shows a BPSK bandpass waveform from the amplifier, and Figure 25 shows the received BPSK bandpass signal from the channel. A brief visual checking on these two waveforms indicates the hardware is working as expected. However, when comparing the waveforms on the oscilloscope and GR, some clipping was visible on GRC. This means that the amplitude of the outgoing signal from the hydrophone is too high for LFRX to handle. Therefore, a 30dB inline attenuator was added between the hydrophone and the USRP. Then, a low pass filter was added to the flowgraph to filter out the high frequency noise generated by the hydrophone.

From Figure 26, the host computer is connected with the USRP through an Ethernet cable. The amplifier is connected with the USRP, powered by a power supply on the left. Figure 27 shows an underwater view of the transducer and the hydrophone in a plastic container.

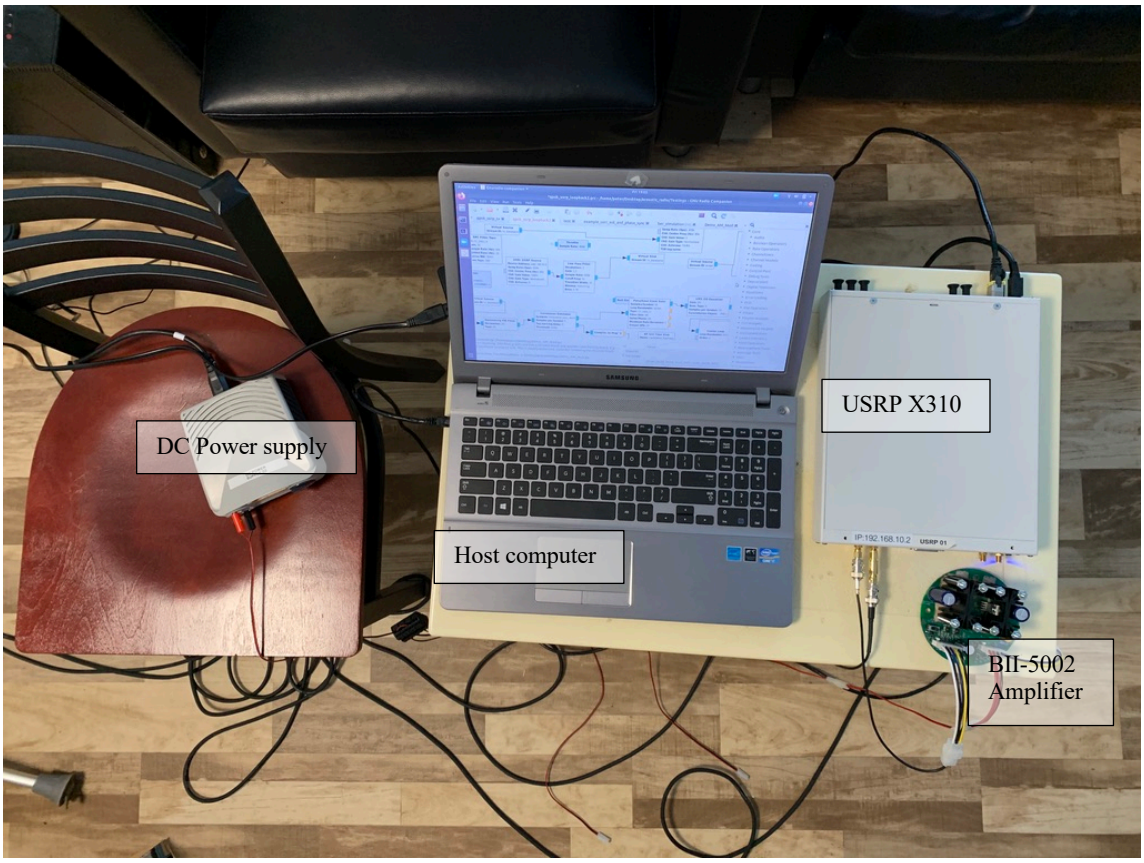

Figure 26. A picture of the experiment setup.

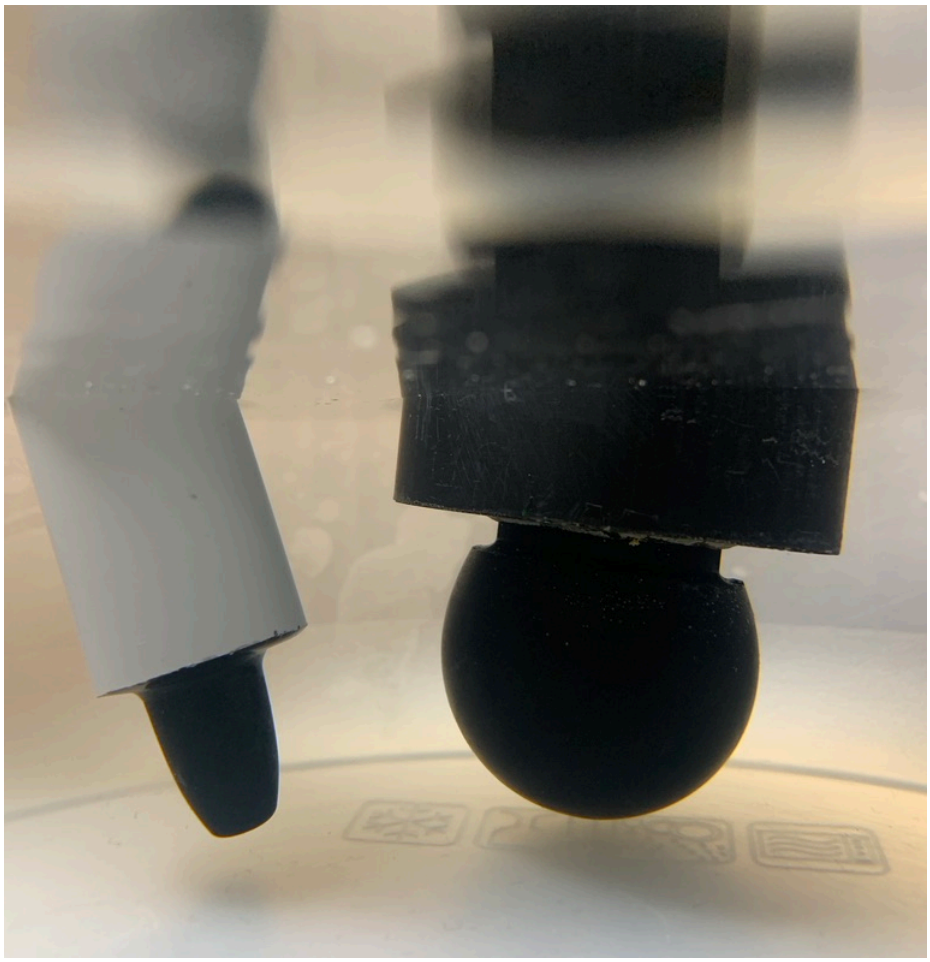

Figure 27.The acoustic transducer and hydrophone in a plastic container.

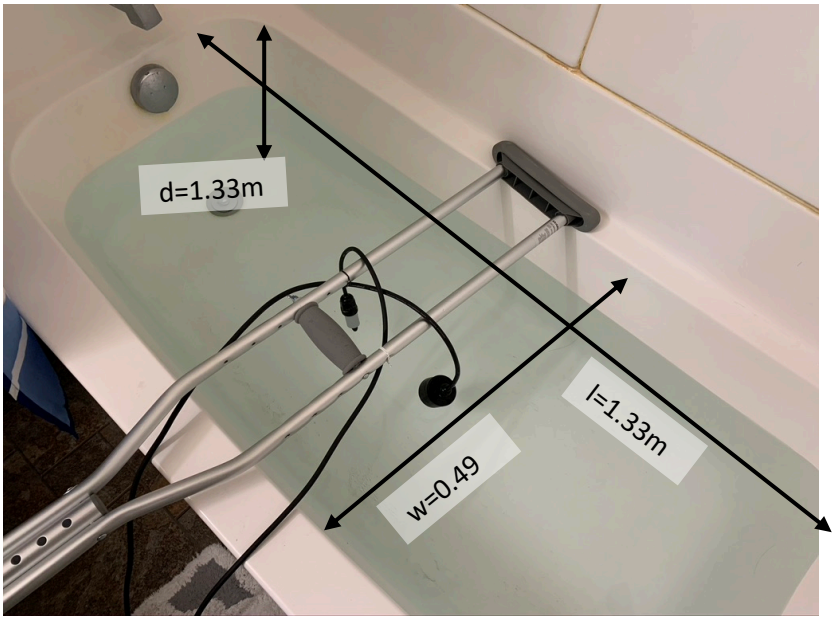

Figure 28. Physical dimensions of the bathtub.

The first test in a plastic container was not very successful. The constellation map is not clearly separated into 4 symbols and the decoded messages are mostly wrong. The reason is the channel. In the loopback test with direct Tx-Rx connection, the channel is simple. Under this situation, the channel equalization required is minimal.

However, when moving from direct-connect-loopback test to plastic-container test, the channel characteristic changed. To equalize the signal in the new channel, some parameters have to be adjusted accordingly. First, the tabs number in the Polyphase Clock Sync was increased to accommodates potential severer clock drift between the transmitter and receiver. Second, the channel length was increased in the LMS DD Equalizer block to accommodates the more complicated channel in the plastic container.

Due to the COVID-19 epidemic and the lockdown situation, a field experiment in the portage canal was not an option. Therefore, another experiment was conducted in the author's bathtub after the successful test in the plastic container. From Figure 28, the bathtub has a 1.33 by 0.49 meters footprint, with 0.33 meters of depth. The transducer and the hydrophone were placed around 14cm from the bottom of the bathtub. Serval tests with different distances between the transducer and the hydrophone were conducted. The results are in the following section.

4.2 Experiment Results

After all the changes are made to adapt the new channel, a clear constellation map was acquired on the receiver panel, and the packets are passing through the Correlated Access Code, which means packets are decoding correctly. In the Figure 29, a screenshot of a binary inspection of an output file form the receiver is shown in the terminal. The numbers in the red box are line numbers of the output file. The yellow box includes raw binary bits from the file, corresponding to the text in the green box, where is the decoded text. Figure 29 shows the received bits are correct since they are being converted to the correct text in the green box.

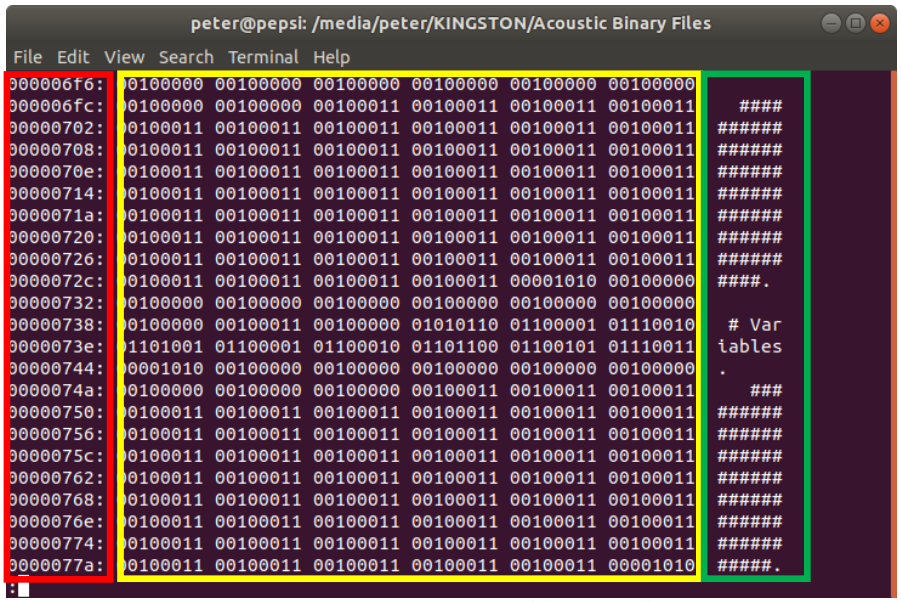

Figure 29. Binary inspection of the output file.

In order to obtain the system performance data, Bit Error Rate (BER) was tested on the system. A built-in BER block was used for BER test. However, the result is far worse (20~30%) than what expected, judging by all the correctly decoded messages. Because it is counter intuitive when the receiver misinterprets almost one third of the information, it still managed to output correctly decoded text.

After further investigation of the output file, the issue was located. The Correlated Access Code block will drop the packet if a predefined number of bits is wrong within the access code. In other words, if this predefined threshold is 12, and there are more than 12 wrong bits are incorrect in a packet, this block will drop the packet even the rest of the payload is perfectly intact. Also, the payload size was set to equal to the size of the input file. Therefore, missing packets are not going to be visible in the output file as the input file has been transmitted repeatedly.

To obtain the correct BER from the system, the packet encoding/decoding part is completely removed, the BER block is only comparing the raw bits from the input file. Finally, the BER can be calculated as:

$$Bit\ Error\ Rate = 10^{-2.76} = 0.17\%$$

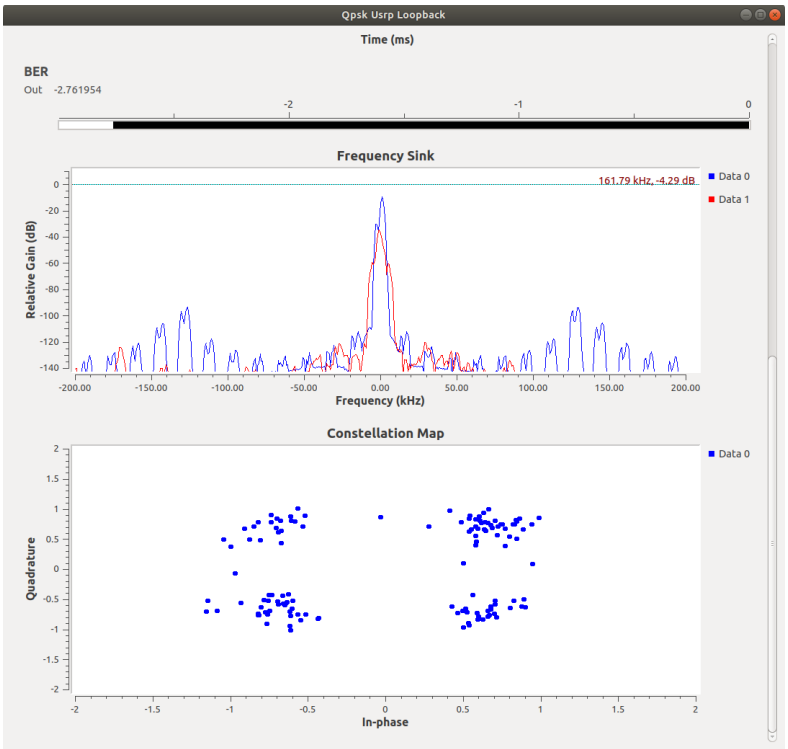

Figure 30. Testing result from a plastic container (2.5cm).

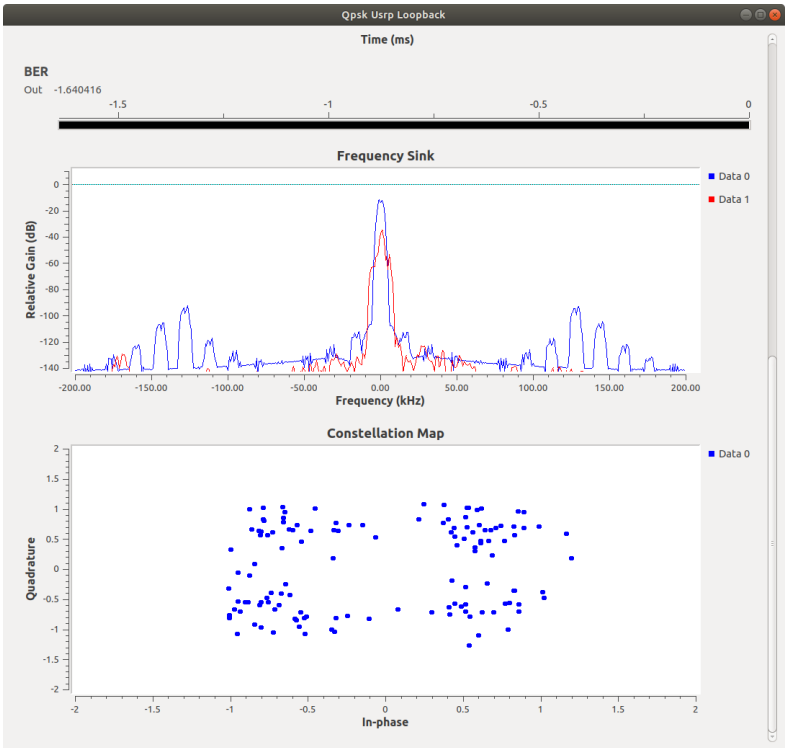

Figure 31. Testing result from a bathtub (8cm).

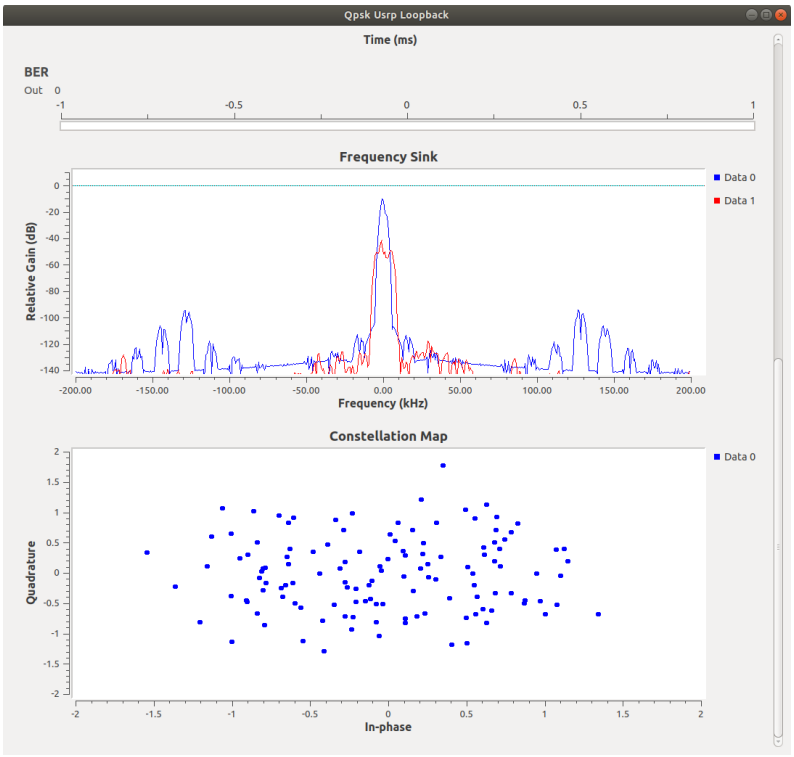

Figure 32. Testing result from a bathtub (26cm).

From Figures 30-32, the system performance (BER) worsen along with longer transmission distance. However, the signal strength is not the issue, as the Frequency Sinks showed the input baseband (blue curve) and the output baseband (red curve) are not necessarily have a huge difference in terms of signal power. Which is an indication of multipath effect is affecting the system performance. Since the transmission distance in the plastic container (2.5cm) is very short compared with the size of the container (24.5cm in diameter), the reflections of the signal would arrive at the receiver with longer delays. Whereas in the case of the bathtub test at 26cm, taking the size of the bathtub into consideration, the arrival delays between the signal reflections and the signal went line-of-sight, could be very minimal. Therefore, the receiver failed to decode the bandpass signals.

The system performance is expected to be better if the tests were performed in a lake/canal environment, as the size of the waterbody would significantly larger than a bathtub, which means the arrival delays of different copies of the signal would be also significantly larger. So, it would easier for the receiver to distinguish the line-of-sight transmission from other reflections, therefore, better system performance at 26cm or longer transmission distances.

In conclusion, based on the tests conducted in a plastic container and a bathtub, it is proven that multipath effect affects the system performance heavily. In this implementation, the communication system is not performing any channel estimation, as the channel equalizer used for this system is a blind equalizer. Therefore, it is believed that a channel estimator would help the communication system to estimate the channel condition. Hence, the system would have a better channel equalization result. Which would lead to a better system performance with a such harsh channel condition. After all, the goal of this project is to prove that it is possible and

beneficial to use software methods to develop more flexible and more robust communication systems.

5 Conclusion

This report detailed the process of a development of software defined underwater acoustic communication system using GR. It first introduced the history and current state of underwater acoustic communications, as well as the challenges in underwater communication systems. Then it introduced software-defined radio, compared SDR with conventional analog radio. It also laid out the perspective operational areas of SDR in the future. At the end of introduction, the motivation was mentioned to answer the question of why building software-defined underwater acoustic communication system is necessary.

In Section 2, a road map of SDR simulation using GR is presented. It started with explaining the relationship between SDR and GR, and some details about GNU Radio project. After that, two examples of GRC flowgraphs are presented, as well as the technical details in them. Later in Section 2, the first working communication system simulation was introduced. It is based on Dr. Aaron Scher's example, BPSK audio modem [10]. Some modification was made due to the different GR version and different modulation scheme used. The result of this successful GR simulation is that a text file can be modulated into acoustic wave, and to be transmitted using a speaker on a laptop. Then, the acoustic signal received by a microphone on a laptop can be decoded.

All the hardware used in this project was introduced in Section 3. This includes the USRP, an acoustic transducer, an amplifier for the acoustic transducer, a hydrophone. Some technical specifications were stated in this section. In the end of this section, the setup process for USRP working with GR was detailed.

Details of the experiment setup was recorded in Section 4. It stated the changes were made from GR simulation to adapt the new hardware added to the system. As the experiment results indicate the system is working correctly with an acceptable BER in short distances, the USRP based software defined underwater acoustic communication system on GNU Radio platform is proven to be a workable replacement for traditional acoustic-modem-based systems. While USRP based systems will likely consume more power than acoustic modems, the flexibility SDR will be more beneficial to researchers who would like to test their UWA communication system under different channel conditions.

This project started out a simple idea of using software-defined radio for underwater acoustic communication systems, some understanding of communication theory was obtained alone the way. The author had the chance to review materials in communication theory and digital communication class by going over the examples of GRC and issues encounter by others in online forums. This is not only a journey of develop a software defined underwater communication system, but a great learning experience in the area of communication theory and communication systems.

6 Future Work

Based on the results from the experiments, there are some work could be done in the future, in order to improve the system performance, as well as the robustness of the communication system. First, a channel estimator could be developed for this communication system in GR. Currently, the basic GR has a built-in channel estimator module for OFDM modulation scheme, but it does not have one for signal carrier communication systems. Developing a channel estimator in GR would help the communication system to obtain a better result from channel equalization, especially in a harsh condition such like a bathtub.

Secondly, even though the implementation in this report has a correlation estimator using a match filter to detect the arrival of packets. But it is merely calculating the correlation and adding tags to the data stream. It does not care about whether if there is a packet. A preamble detection module would not pass on the data stream if it does not match a preamble. Which could help the system to save power when there is no communication signal received.

Thirdly, a common algorithm in communication systems that does not exist in this GR implementation is error correction coding. Even though this will introduce more overhead in the packets, but it would increase the robustness of the communication system.

Another thing could be done in the future is to explore the OFDM modulation scheme as it could improve the data rate in short distance transmission, as well as better bandwidth efficiency.

Finally, due to the coronavirus epidemic and stay home policy of the state of Michigan, all the tests were conducted in author's apartment. More experiments should be done in lake/canal environments in the future to obtain data corresponding to the real underwater acoustic communication channel.

7 References

[1] H. B. Ross Williams, "Coherent Recombination of Acoustic Multipath Signals Propagated in the Deep Ocean," *The Journal of the Acoustical Society of America,* vol. 50, 1971.

[2] M. Chitre, S. Shahabudeen, L. Freitag, and M. Stojanovic, "Recent advances in underwater acoustic communications & networking," in *OCEANS 2008*, 15-18 Sept. 2008, pp. 1-10, doi: 10.1109/OCEANS.2008.5152045.

[3] C. Wang, "Intelligent and Secure Underwater Acoustic Communication Networks," Open Access Dissertation 2018.

[4] W. Sun, "Online Learning of the Spatial-Temporal Channel Variation in Underwater Acoustic Communication Networks," Open Access Dissertation 2019.

[5] F. K. Jondral, "Software-Defined Radio—Basics and Evolution to Cognitive Radio," *EURASIP Journal on Wireless Communications and Networking,* vol. 2005, no. 3, p. 652784, 2005/08/01 2005, doi: 10.1155/WCN.2005.275.

[6] T. Anxo, "Software Defined Radio: A Brief Introduction," *Multidisciplinary Digital Publishing Institute Proceedings,* vol. 2, no. 18, p. 1196, 2018, doi: 10.3390/proceedings2181196.

[7] F. H. M. Rice, "Multirate digital filters for symbol timing synchronization in software defined radios," *IEEE Journal on Select Areas in Communications,* vol. 19, pp. 2346--2357, 2001.

[8] D. Smalley, "Equalization concepts: a tutorial," *Atlanta Regional Technology Center, Texas Instruments,* pp. 1-29, 1994.

[9] L. W. Couch, "Digital and Analog Communication Systems," 2013.

[10] A. Scher. "GNU Radio Companion - BPSK Audio Modem." https://aaronscher.com/GNU_Radio_Companion_Collection/Audio_modem.html (accessed April, 20, 2020).