A Software Defined Underwater Acoustic Communications Hardware Platform

Author: Keelan Burns
Student ID: 2085747        FAN: burn0207
Supervisor: Associate Professor Karl Sammut
Submission Date: 15 November 2013

Submitted to the School of Computer Science, Engineering, and Mathematics in the Faculty of Science and Engineering in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Electronics) at Flinders University - Adelaide Australia
“I certify that this work does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

Keelan Burns

15/11/2013
Abstract

The purpose of this Thesis is to investigate and design a low cost Software Defined Acoustic Communications Hardware Platform, to support underwater communications and to supply a low-cost solution to an otherwise expensive and difficult practice. The Thesis will base its work on previous literature in the underwater communications field, and will focus on using acoustic signals over other communication forms. By utilising a Software Defined approach to the problem, the low cost requirements can be met. For the platform to work however, a specifically designed transceiver will be required. The design will include the structure of the overall platform, incorporating the battery, processor and transducer and the creation of a PCB as a transceiver. The designed transceiver will be designed to perform all required hardware aspects that are not accomplished by the software defined radio, to achieve an open hardware platform. It will also include the complete selection of all components and eventual population of the board. The Thesis will investigate the use of software defined radio and its development which is to be applied to the hardware platform. The end result for the platform will depend upon final extensive underwater ocean tests, with result final result criteria based predominately around the final cost of the hardware platform. The Thesis succeeds in developing a transceiver and hardware platform, however work on software remains ongoing and improvements to hardware could be made. The Thesis finds that the developed hardware platform is of sufficiently low cost to meet the required design criteria.
Acknowledgements

I would like to thank the following for their assistance throughout this Thesis:

Associate Professor Karl Sammut, Dr Andrew Lammas, Jonathan Wheare and the Technical Staff at Flinders University.
Contents
Abstract .................................................................................................................. 3
Acknowledgements ............................................................................................... 4
Introduction ............................................................................................................ 8
Thesis Statement ................................................................................................... 8
Thesis Outline ........................................................................................................ 9
Background Information ......................................................................................... 10
  Underwater Communications ......................................................................... 10
  Software Defined Radios ............................................................................... 10
  GNU Radio ...................................................................................................... 11
  Hardware ......................................................................................................... 12
Proposed Design .................................................................................................... 13
Goals ...................................................................................................................... 14
  Cost .................................................................................................................. 14
  Data Rate ......................................................................................................... 14
  Range ................................................................................................................ 14
  Power and Lifetime .......................................................................................... 14
  Size .................................................................................................................... 15
Deliverables ........................................................................................................... 15
  Prototype Transceiver ...................................................................................... 15
  Prototype Hardware Platform ....................................................................... 15
  GNU Radio software .................................................................................... 15
  Completed Software ....................................................................................... 15
  Final Combined Platform ............................................................................. 16
  Current Status ................................................................................................. 17
Literature Review ................................................................................................ 18
Hardware Design ................................................................................................ 22
  Transducer ..................................................................................................... 23
    Soldering and Potting .................................................................................... 24
    Final properties ............................................................................................ 24
  Battery .............................................................................................................. 24
  Development Platform and Processor ........................................................... 25
  Transceiver ..................................................................................................... 26
  Power Regulation ............................................................................................ 26
Transmitter Design.................................................................................................26

Boost Converter ..................................................................................................27

H-Bridge ..................................................................................................................27

Receiver Design ..................................................................................................28

Filter ......................................................................................................................29

Wake-up ..................................................................................................................30

ADC ..........................................................................................................................30

Other Components ..............................................................................................31

BeagleBone Black Pins .......................................................................................31

3D Board Layout .................................................................................................33

Soldering ..................................................................................................................34

Final Hardware Platform ......................................................................................35

Overall Cost ...........................................................................................................35

Current Issues with Hardware .............................................................................35

Improvements to the Design ...............................................................................36

GNU Radio ............................................................................................................37

Phase-Shift Keying ................................................................................................37

Transmission .........................................................................................................38

Receiving ...............................................................................................................39

Current Issues .......................................................................................................40

Improvements to GRC .........................................................................................41

Future Work ..........................................................................................................46

Conclusion .............................................................................................................49

Bibliography ..........................................................................................................51

Appendices ............................................................................................................53

Bill of Materials ....................................................................................................53

GNU Radio Code ..................................................................................................59

Transmitter ............................................................................................................59

Receiver ................................................................................................................62

Hardware Schematics ............................................................................................65
Table of Images
Image 1 - Non-potted Steminc Model Transducer [Setiner & Martins Inc.] ........................................ 23
Image 2 - Top Layer PCB 3D ........................................................................................................ 33
Image 3 - Bottom Layer PCB 3D .................................................................................................. 34

Table of Figures
Figure 1 - Example GRC Flow Chart ......................................................................................... 12
Figure 2 - Top Level Hardware Platform Flow Chart ................................................................. 22
Figure 3 - Transmission GRC Flow Chart .................................................................................. 38
Figure 4 - Receiving GRC Flow Chart ...................................................................................... 39
Figure 5 - Transmitted Signal ..................................................................................................... 43
Figure 6 - Received Signal ........................................................................................................... 44
Figure 7 - Decrease in Amplitude from Increase in Range ....................................................... 45
Figure 8 - Increase in Amplitude from Increased Audio Level ............................................... 45
Introduction

Thesis Statement

The underwater environment remains difficult for communications and the rising importance of communicating in this area is leading to an increased amount of research being undertaken. Current communication platforms are limited by high cost and heavy power usage requirements, suggesting a need for further exploration and experimentation of options available.

Current rapid improvement in computing power and Software Defined Radios (SDRs) presents the opportunity to experiment further in an underwater environment, by taking advantage of the SDRs flexibility, re-configurability and lower cost. GNU Radio is a SDR toolkit widely used throughout the research community [Lang, 2011].

Previous studies have shown that SDRs can be used effectively in terrestrial environments, but far less research exists for using SDRs in underwater environments. Studies have proven the use of an underwater communications platform using GNU Radio [Karpeles, Torres], and some studies have focused on developing low cost underwater hardware platforms [Benson et al. 2010]. This Thesis will use these two papers listed as it base.

The purpose of this Thesis is to design a cheap, but effective, acoustic Software Defined Underwater Communications Platform. To accomplish this, the Thesis will focus on developing a low cost hardware platform that is compatible with a processor running GNU radio and a cheap transducer, using GNU radio to accomplish basic signal processing, including modulation and testing the effectiveness of this approach. This platform is designed to be an open hardware platform for further use.

The overall cost of the platform, the ease of implementation, the measured average bit error, the data rate and range will be used to determine the effectiveness of the SDR based platform.
Thesis Outline

The introduction presents the background information on the project and how the project is to be accomplished. A literature review will analyse material related to the project proposal to form the basis for the experimental project. The Thesis will attempt to first design the open hardware platform to be used for the underwater communication by using off the shelf components wherever necessary. Secondly, a proof of concept SDR using GNU Radio that can be simulated and checked using a PC and its appropriate sound card will be created. Finally, the Thesis will present the methods used to test the platform and the results of the system before concluding the overall platform outcome and effectiveness and future work to be done.

The Thesis is split into the following six parts:

1) An introduction detailing the Thesis Problem Statement, Background Information, Proposed Design, Goals and Deliverables of the project.

2) A detailed literature review of material related to the project.

3) The design of the hardware platform, detailing the processor, the Front End Hardware Design, transducer, battery and processor. An explanation of the choices made will accompany this.

4) The design of the SDR using GNU Radio and its Graphical Radio Companion (GRC). The design will be presented in the form of flow graphs created through the GRC. One modulation scheme will be designed.

5) The methods used to test the system will be detailed and the results will be presented. A discussion of the results will also occur. The testing will focus on the bit errors received by the receiver, bit rate and range.

6) An investigation into the future work required to complete the project, and how the project could be extended.

7) A conclusion of the work and the overall effectiveness of the platform.
Background Information

Underwater Communications

The underwater environment is far less conducive for communications than its terrestrial cousin. Communications are less conducive due to a number of factors including multipath propagation, time variation, surface scattering (from the rain or other effects), and greater attenuation of the sent signals because of the properties of the medium [Presig, 2006].

Multipath propagation is when signals reach the desired receiver by two or more paths resulting in interference and phase shifting. This effect is very significant in underwater communications because of the refraction and reflection properties of water. The low speed of sound in water causes great delay in receiving the signals, hence time variation.

Attenuation is the loss of intensity of a signal when travelling through a medium, and is greater in water than in air for the most part. Surface scattering is largely responsible for time variability, particularly in shallow communications channels.

These effects greatly reduce the effectiveness of communication signals in the underwater environment. Although these effects still affect acoustic signals, acoustic signals remain capable of better communication in the underwater environment over optical or electromagnetic signals. This is due to electromagnetic signals suffering far more greatly from attenuation than acoustic signals, and optical signals suffering far more from effects such as scattering [Stojanovic]. However, using acoustic signals results in far lower data rates compared to the other signal types, which is one of the reasons that underwater communication remains difficult. Overall, acoustic signals are the best choice for underwater communications.

Software Defined Radios

An SDR is a radio communication system, where the majority of the signal processing such as filtering and modulating, is accomplished in the software of the processor. It is noted that a normal radio will accomplish all of this in hardware.

An SDR will generally consist of a processor which takes care of the software signal processing, an analogue-to-digital converter and a front end which includes the transceiver and all circuitry between the processor and transducer.
This results in a large number of advantages over the traditional hardware approach, including flexibility, reliability, re-configurability, density of hardware and cost among others, as no signal mixing is required in hardware.

SDRs allow for high flexibility for the created platforms as a simple software update allows for changing of the functionality of the platform without changing the hardware [Valerio, 2008]. For this reason, an SDR has huge advantages in upgradability and re-configurability, as the software can easily be adjusted or replaced.

As the software replaces a large amount of the hardware in the platform, the amount of hardware in the system is decreased and likewise, so is the cost. Lastly, the reliability of the system is increased, as generally, well written software improves reliability over hardware as it cannot break down.

Current disadvantages of the system include the reduction in security of the system due to its software nature – although normal software security features can be implemented.

**GNU Radio**

GNU Radio is an open sourced SDR development toolkit. It provides signal processing blocks which can be used to implement the SDR rather than forcing the developer to develop the code from the beginning. GNU Radio signal processing blocks are developed in C++ due to their critical nature and speed requirements, while the blocks are wrapped together in Python [Lang, 2011]. GNU Radio is highly used within the research community, one of the reasons for its selection for use within this Thesis.

The Graphical Radio Companion (GRC) is a graphical user interface used to design SDR through the creation of flow-graphs using a drag and drop method of the GNU Radio signal processing blocks. It allows for real time visualisation of the SDR and the ability to change parameters in real time to immediately view the effects. An example of a basic GRC flow graph can be seen in Figure 1:
GNU Radio is compatible with most Digital Signal Processors (DSP) that are available off-shelf, and can also be used in a simulation like environment. As GNU Radio is compatible with most DSP’s, it is possible to develop the SDR without the hardware currently available, allowing for the SDR to be designed off-site. GNU radio is also capable of running on all operating systems, although it prefers a Linux environment for full development and fewer errors. PC sound cards are capable of being used for testing audio communications using GNU Radio.

Hardware

Transducers are devices which convert a signal between types of energy. In this particular case for acoustic underwater communications, the transducer will generally convert electrical energy into acoustic waveforms, allowing for communication to take place.

Transducers are capable of both receiving and transmitting signals, hence they are suitable for an underwater platform. All transducers have different properties which must be taken into account in any hardware design.

Other hardware required includes the processor capable of running the GNU Radio package and designed software, therefore requiring the processor to have relatively large clock
speed and be relatively small of size, due to the hardware platforms mobile applications. Traditional SDR hardware platforms also consist of the hardware between the processor and transducer, which generally involves analog-to-digital (ADC) and digital to analog (DAC) converters for appropriate signal conversion. A driver circuit for the transducer, such as a pre-amplifier, can generally be included in the platform as well.

**Proposed Design**

The proposed platform will consist of a low cost main processor (likely a development board), connected to a similarly low cost transducer with a transceiver in-between. It is expected that a driver circuit will be implemented for driving the voltage to the transducers ideal state. A filter will also be implemented on the receiving side, so that only the wanted frequencies will be received that will amplify the signals, allowing for improved range.

The software will be created as flow graphs in GRC and will be downloaded into the processor. The software will consist of both transmission and receiving software, so that each platform is capable of both. Testing of the software will utilise many different modulation schemes.
Goals
The following lists the goals for the project. There are five final desired goals of the project listed in order of importance to the project:

Cost
The final price of the platform is desired to be as low as possible. Current systems are expensive and part of the aim of the project is to present an inexpensive alternative, hence the goal of low cost. This allows for the possibility of using numerous platforms in a single system, the ability for easy and cheap experimentation and the decrease in risk of using the platform, as a lost platform would not be debilitating cost wise.

Data Rate and Error Rate
The data rate is required to be greater than 200bps. This rate is considered to be near the minimum data rate to allow for sufficient message communication for the platforms, to provide things such as basic commands for Autonomous Underwater Vehicles. A data rate less than this will greatly reduce the usability of the platform. Generally, this data rate will be dependent on the speed at which the SDR code is capable of, and the method in which GNU Radio is used, including the modulation scheme implemented. The error rate is required to be as low as possible to allow for efficient communication, and will also depend predominately on the software, in particular, the use of error correction codes.

Range
A minimum range of 200m is the aim. A range less than this will affect the situation in which the platform will be used as more platforms will then be required to perform the same function. The range will likely be dependent on the amplification of the signals being transmitted and the amplification of the received signals.

Power and Lifetime
The power consumption of the platform should be relatively low, to allow for long battery lifetime and functionality underwater. This will allow the platforms to be deployed for the longest amount of time feasible and depends on the transceiver design, components and the battery used.
The platform should be of a small size factor to allow it to be placed comfortably within smaller underwater vehicles or other proper casings. A large form factor would limit the platforms usability in certain situations.

**Deliverables**

A deliverable is a tangible product that can be produced from the project. The following are the stated deliverables of the project and their explanations.

**Prototype Transceiver**

The prototype transceiver will include the designed Printed Circuit Board (PCB) for the hardware, which will form the majority of this project. It will be responsible for all hardware processors between the SDR and transducer. The prototype board will be populated with the selected components.

**Prototype Hardware Platform**

The prototype hardware platform will consist of the designed and populated transceiver in conjunction with the decided processor, battery and transducer. It will be a prototype of the hardware system that will be used in the actual underwater environment. The overall expense of one platform shall be calculated based on this deliverable.

**GNU Radio software**

The GNU Radio software will include the deliverable proof of concept and the completed GNU Radio code that will be uploaded into the prototype hardware platform. The code is the actual deliverable, not the GRC flow charts, as this is what is finally uploaded.

**Completed Software**

The completed software will include the GNU Radio code and *all* other codes required for the correct functioning of the final hardware platform. This includes the code required to successfully run the processor and interact with the designed transceiver. The completed software will also be dependent on the pin outs used in the hardware design.
Final Combined Platform

The final combined platform will consist of the Prototype Hardware Platform, which has been uploaded with the completed software. It will form the basis for the majority of the tests that will be used to detail the effectiveness of the platform.
Current Status
Currently on the hardware side, the transceiver has been successfully designed and the PCB has been manufactured. The PCB is partially populated, as a number of components are still being waited on. Two boards have been acquired. The Transducer has arrived however, the potting compound has yet to preventing the final soldering of the transducer. The battery and BeagleBone Black have arrived.

On the software side, basic modulation schemes for transmitting and receiving software have been completed. Transmission between the transmitting and receiving software using two PC soundcards has been tested and shown to work, albeit with occasional errors. Work on completing the GNU Radio code is still required and uploading this to the BeagleBone Black is still to be tested. Error reduction codes are to be introduced along with the appropriate up conversion of frequencies to the desired frequency level.
Underwater communications remain difficult, particularly when compared to terrestrial communications due to multiple issues [Presig, 2006] including: multipath effects and surface scattering among others. [Nam and An, 2007] state that acoustic communication methods remain the default method for underwater communications as these effects greatly diminish the use of other methods generally used in terrestrial communication, such as electromagnetic communications. [Stojanovic] agrees with this, stating that the only radio waves that propagate any worthwhile distance in conductive sea water are extra low frequency signals which hence require large antennas and high transmission power.

Stojanovic also states that optical signals, while suffering less from attenuation, are greatly affected by scattering. Traditional approaches to underwater acoustic communications generally involve heavy power usage and expensive equipment. [Wills et al. 2006] instead suggests using simple but numerous devices close together. To combat this, researchers are looking into alternative approaches by attempting to utilise software to a greater extent, hence avoiding additional hardware costs and increasing the flexibility and configurability of the systems. The large improvement in recent years of SDR [Efstathiou, et al. 1999 ] allows the entirety of the signal processing to be accomplished in software such as open-source GNU Radio with only the requirement of front ends remaining in particular analog-to-digital converters (ADC’s) and pre-amplification. They also state that SDR’s are limited by the performance of ADC’s. However, due to technological improvements since 1999, ADC performances have improved hence decreasing the extent of this limitation. In general, a SDR should be divided into two sub-systems; hardware-defined (containing the front-end) and software-defined.

Field Programmable Gate Arrays (FPGA) have also started to be researched as an alternative method to traditional underwater acoustic communications. [Nowsheen, 2011] have investigated the use of FPGA’s and believe that they can also achieve reduction in overall hardware, and a high level of re-configurability compared to traditional digital signal processors (DSP). FPGA’s themselves are also mostly programmed in software. In general however, FPGA’s are more expensive than DSP processors and would greatly increase the scope of this project if an FPGA was to be used as the processor but does appear to be a viable alternative.
While much research has been done on using SDR’s in terrestrial environments, their use in underwater environments has been less documented. [Karpeles, Torres] paper presents an investigation into creating a low-cost communications platform based around GNU radio, utilising only GNU radio (on PC’s) and transducers. However, Karpeles and Torres found that after a small period of time, their experiment suffered from relatively large error rates, which was detailed as being caused mostly by their software. While it was clearly found that communication is possible, the errors restricted the achievement of accurate results, as they could not send data as fast as they could sample. It was also noted that the experimentation was limited by bandwidth.

Karpeles and Torres also found that utilising only GNU radio a soundcard and a transducer, that not only could this be an effective method for underwater communications, but also that it was possible to “implement relatively any modulation scheme that we want”. Many different modulation schemes currently exist, and many have been explored. Karpeles and Torres explored in their paper both Multiple Frequency Shift Keying (MFSK) and Orthogonal Frequency Division Multiplexing (OFDM) and found OFDM to have a higher bit rate. They however, did not analyse any other modulation standard, particularly non frequency based schemes. [Akyildiz et al. 2004] suggests there exists no current modulation standard, as many schemes are more beneficial in certain situations and vice versa, such as the trade-off for power efficiency and bandwidth between coherent and non-coherent modulation schemes. One of the unique benefits of a software-defined approach is to be able to reconfigure the software quickly, allowing for different modulation schemes to be used in different situations, without changing the hardware. GNU Radio contains a large number of modulation schemes already implemented as blocks available to the user. Utilising this investigation, the University of California Electrical Engineering department developed a research platform, called UANT, an expensive endeavour [Torres et al. 2009]. A lower cost alternative will greatly improve research and application opportunities.

[Valerio 2008] suggests that an ideal SDR would consist solely of an antenna (in the case of terrestrial communications), ADC/DAC and the software-defined subsystem. He noted however, that in reality, the ADC’s and DAC’s are not fast enough for all uses, and that antenna’s and other transceivers (such as transducers) are designed for specific frequencies. Due to this, the hardware-defined subsystem, the front-end of the platform, becomes an
important section of the overall platform design and the need for a transceiver design becomes apparent for current research to improve.

Although a software-defined approach is shown to be effective in greatly reducing the cost, the performance and many power requirements of the system will remain dependent on the RF front end of the platform, most particularly the transducer used. To this end, research has been conducted on using cheap transducers in platforms while affecting the performance as little as possible. [Sánchez, et al. 2012] have used a cheap fish-finding transducer (the Humminbird XP 9 20) rather than more costly and usual alternatives, and were able to meet their requirements. The price of this transducer is approximately AUD $50. Utilising this and their designed modem, Sanchez were able to achieve a bit rate of approximately 1kbps. It should be noted that as they designed the full modem, called ITACA, other costs such as hardware were associated. However, the overall price remains much cheaper than current commercial high grade alternatives, such as LinkQuest which can cost thousands, for the trade-off of lower performance in data rate transfer and range.

Another approach in low cost hardware, taken by [Benson et al. 2010] was to design a front end based entirely around making the transducer as cheap as possible, resulting in very cheap hardware, but with low data rates (of around 200 BPS) and acceptable range (>350m). The transducer used was a $14 Stemmic model ring PZT. The properties of this transducer was used as the basis for all other designs, leading to a final low cost design. The signal processing of the system must also take into account the hardware’s properties, but can be easily changed to meet the required needs due to its reconfigurable nature. If the intent is for very cheap underwater communication platforms, such as nodes in a network, then the expensiveness of the platform is one of the most inhibiting factors for a large array of systems. Long baseline acoustic positioning systems are a great example of where cost needs to be decreased [Vickery 1998], due to utilising a network of nodes.

While the transducer used in the hardware design can be considered the most inhibiting factor in cost, data rate and range, the remainder of the front end must also be considered including the ADC/DAC and amplifiers. Benson proposed an amplifier system which consists of a class AB amplifier and class D amplifier in parallel, which would also help increase the
range of the system. As Efstathiou and Valerio have suggested, the ADC is a limiting factor, so this further increases the importance of the transceiver design.

Most research of acoustic underwater communications detail cost and flexibility as some of the greatest factors in determining what is able to be accomplished underwater. To this reason, the development of a software-defined radio hardware platform offering the characteristics of re-configurability and allowing for the replacement of large amounts of hardware from traditional solutions, and a cost centric design of the transceiver is important to the improvement of underwater communications.
Hardware Design

The developed hardware platform is required to perform all the intermediary steps between the processor and the transducer. Hence, the transceiver includes pre-amplification, analog-to-digital conversion and digital-to-analog conversion, as well as power management. It was also decided the original filtering of the received signal to be done in hardware. This section of the Thesis will detail the design, and the reasoning behind particular choices in the design and hardware selections made.

To do so, a printed circuit board was created. The design was based on the transducer, the development platform with processor and the battery selected for the project. Following this, the transceiver design was split into transmission and receiving sections before being combined at the top level. Figure 2 is the outline of the hardware platform, including the designed transceiver, and the parts contained within.

Figure 2 shows the design of the overall hardware platform, and how sections will interact with each other. These sections, and their functionality are explained below. The designed transceiver includes all sections contained within the red dotted square.
Transducer

The selection of the transducer was the required first step in the design of the requisite hardware. The transducers power and resonant frequency properties serve as the basis for much of the design. In selecting the transducer, Benson’s work proved the feasibility of using the Steminc model “SMC26D22H13111” Piezo Ceramic Cylinder as a cheap solution. Hence, this model was selected for this project. The manufacturer, Steminc, states the resonant frequency of the transducer to be 43kHz ± 1.5kHz, in its native form [Steiner & Martins Inc.]. One of the major considerations for selecting this transducer was its very low cost, at approximately $14 per cylinder, in attempts to meet the design criteria of a low cost platform. Due to a reasonably low resonant frequency, a software approach is possible, as very large sampling rates would not be required.

Image 1 - Non-potted Steminc Model Transducer [Setiner & Martins Inc.]

Commercial Transducers were considered for the project. Of major consideration was the Humminbird XP 9 20. The higher price of the Humminbird (approximately $50) [Johnson Outdoors Marine Electronics Inc., 2013] was against the design criteria of a low cost solution, although for a transducer this cost remains low. Of greater import in the selection process, was the lowest usable frequency band of the transducer listed at 85kHz [Sanchez, 2012]. This relatively high frequency would remove the transducer far from the acoustic
range. However, it would also cause large problems by limiting the choices of processor, due to requiring a very large sampling rate. Other commercial transducers, such as the Navman 51864 were rejected on similar grounds.

**Soldering and Potting**

When soldering wires to the transducer, the recommended Steminc procedure is followed [Steiner & Martins Inc.]. This includes utilising a solder with 1-2% silver content, and using a soldering temperature between 250 and 270 degrees Celsius.

The potting material for the transducer was selected to be the same as what Benson used throughout their paper, with the intent to achieve the same final transducer properties. The potting compound “CONATHANE EN-12“ from Cytek Industries, is a two component urethane compound. By potting the transducer with this compound, it will help to ensure the performance as well as help protect the transducer in the underwater environment. The potting takes place in-house, in a controlled laboratory environment.

**Final properties**

After potting and soldering, the transducers properties change, as found by Benson. The potted transducer has a resonant frequency at approximately 35kHz with a resistance of 1.5kohm. Part of the design goals of the project calls for a low power solution, to enable a long battery life for the underwater platforms, however the power output must be large enough to achieve sufficient range. For these reasons, a compromise of approximately 1W power usage when transmitting data was decided upon.

**Battery**

A two cell lithium ion battery is used for original testing purposes of the project. The battery has an approximate lifetime of 4.9 Ampere-hours; a substantial amount. As one of the hardware design goals is for a long lifetime product, a battery supplying 4.9Ah should allow for a very substantial battery life cycle for the platform. As a lithium ion based battery, the nominal voltage output is 7.4V. However, it does generally vary and the transceiver circuitry is designed to function for a range between 6.6V and 8.4V. This forces the PCB design to accommodate a possible range of input voltages.
This particular battery was decided upon due to its relatively large Ampere-hours, which would offer a very large lifecycle. However, investigations into batteries that are not lithium ion based could be beneficial to the final hardware platform, due to their relatively high cost.

The PCB also performs the appropriate protection of the battery and circuitry on the rest of the board. This battery protection circuit performs over current protection, reverse polarity and under voltage (or over discharge) protection. It was determined that over voltage protection was not required, as the circuitry using direct battery input was designed to be capable of handling slightly higher voltages. The schematic can be seen in the Appendix.

A protection circuit was required for the hardware due to the nature of using lithium ion batteries. Lithium ion batteries have a natural variation in the voltage they output. Under voltage occurs when the output drops below the minimum voltage the circuit is designed for. It damages the total lifetime of the battery as deep discharge will damage the total capacity of the battery. Protecting the battery lifetime will help increase the overall lifetime of the platform. Also, protection against under voltage will stop the circuitry being powered if the battery drops below the desired voltage level, protecting it from unwanted use. Reverse polarity protection is also implemented to stop damage occurring to the circuitry if the battery is positioned incorrectly. A fuse is used to protect from over current situations, which could damage the circuitry and users, and is a standard safety measure.

The approximate cost of the battery protection circuit is AUD $20. While this is expensive, it greatly increases the lifetime of each individual hardware platform. This circuit can be seen in the Appendix. Note, that the resistor passive components have not been calculated, as this depends on the batteries characteristics. At the time of writing, the required characteristics for the battery used in the project were not available.

Development Platform and Processor
A development board, allowing digital programming of the processor and other digital input and output pins, was required for ease of development and utilising GNU Radio, as well as for utilising enable pins and Pulse Width Modulation (PWM) output.
The board selected was the BeagleBone Black. The BeagleBone Black has a relatively high clock speed processor, the AM335x 1GHz ARM Cortex-A8; for a low cost and small sized development board. A high clock speed is required to allow for the running of GNU Radio, without which the hardware platform would be meaningless. A capable clock speed was one of the major considerations in choosing the Beaglebone Black as the development board.

However there are a number of other reasons at to why the BeagleBone Black is advantageous. This includes the option to utilise PWM through specific output pins; its ease of use and development for; its large development community; its Linux based operating system (for GNU Radio); its small size to allow it to fit into many small underwater vehicles, such as AUV’s; and finally its relatively low cost.

Other processors considered for the development board include the BeagleBone and Rasberry Pi. The BeagleBone was decided against because of its higher cost compared to the BeagleBone Black, where cost of the overall platform is one of the main considerations of the project. The Rasberry Pi on the other hand, has a lower cost than the BeagleBone Black; however, its clock speed is slightly lower unless overclocked.

**Transceiver**
The Transceiver design consists of the PCB. This includes the battery protection circuit, the transmitter and receiver sections. It is the midsection between the processor and the transducer.

**Power Regulation**
The PCB is supplied by the typically 7.4V battery supply. However, the BeagleBone Black, which is powered through the board rather than separately, needs a regulated 5V power supply, as does a number of further on board circuitry. To accomplish this, a 5V regulator is used. This regulates the voltage on board to a stable 5V with negligible variation. The Schematic for this circuit can be seen in the Appendix.

**Transmitter Design**
The transmission stage of the designed hardware was required to perform a number of important functions. Of major import, was the procedure of accepting the signal processed signal (including modulation) from the processor, converting this from digital to an analog
signal, and carrying it to the transducer, which would perform the electrical to acoustic conversion of the signal. Also required is the appropriate power manipulation to achieve the required voltage levels of the transducer. As the transmitter need only be active when data is being transmitted, power is only supplied to the section when required.

**Boost Converter**

Based on the final properties of the transducer, when transmitting at this ideal 1W power, it is found that the ideal supply to the transducer for transmission would be at 48V with approximately 30mA of current. The boost converter however, is always powered on, due to no mechanism or enable signal being used. This results in a relatively large amount of power always being drawn.

This leads to the design being required to boost the input data to the transducer to a 48V voltage level when transmission of data is occurring. Hence, the transmitter requires a boost converter which is capable of boosting the battery’s variable voltage taking into account a minimum of 6.6V, for a typical voltage boost from 7.4V. The schematic for this design can be seen in the Appendix. The boost converter was also required to enable a boost to 48V from a relatively high voltage input of approximately 8.4V.

Due to the very high voltage levels of the boost converter and subsequent signals, the boost converter was required to be relatively isolated from the other electrical circuitry, to prevent it from interfering with other signals. It was also required to be as close as possible to the battery, to prevent too large a voltage drop. The approximate cost of the boost converter, which can be seen in the Bill of Materials (Appendix), is AUD $10. The boost converter was designed using Texas Instruments WEBBENCH Designer, which can be located on their website.

**H-Bridge**

The H-Bridge serves the purpose of the digital to analog conversion in the transceiver design. A normal Digital-to-Analog Converter (DAC) was considered. However, the requirements of the data signal to be boosted to 48V, prevented using a general cheap off-the shelf DAC, particularly due to current requirements of most 48V DAC’s. Simply using a DAC would not permit the data signals to be driven to the appropriate voltage. A full H-bridge was instead decided as a solution. The data signals are PWM modulated signals,
making this possible. This is done using the BeagleBone Black, which has a number of PWM outputs which can be utilised and programmed. Two PWM outputs are used for each transmission, to successfully control the H-Bridge and represent the signal processed data.

By using the switching mechanism of the H-Bridge, a sinusoid data signal at 48V can be applied to the transducer. The H-Bridge schematic can be seen in the Appendix. A number of problems were encountered when designing the H-Bridge. Particularly, most H-Bridge’s at around 48V are designed for far higher current requirements than applicable to the underwater platform. The solution to this was to use a mosfet driver in combination with four smart power switches in a H-Bridge form. This achieved the goal to not only cause the data signal to switch between positive and negative, but to allow the signal to be driven to the required voltage level.

The H-Bridge is powered from the boost converter at all times, as there is no enable signal path or mechanism in the design that will turn it off. However, signals are only outputted from the H-Bridge to the transducer if there are incoming PWM data signals, hence preventing the transducer from transmitting signals when nothing is actually being transmitted.

The approximate cost of the H-Bridge components used in this transceiver design is AUD $6. For a H-Bridge at this voltage, this is relatively inexpensive. A general DAC, of the resolution and sampling rate required, may be less expensive but it would not achieve the other required aspects of the transmitter and is not guaranteed to be so.

**Receiver Design**

The Receiver section of the hardware platform involves receiving the electrical signal from the transducer, filtering and amplifying this signal so that operations can be performed on it, before passing the signal through an ADC and sending the signal into the processor to allow it to be demodulated in software (GNU Radio). Unlike the transmitter, the receiver must be continuously listening for a received signal. To do so, and to save power, a wake-up circuit is also implemented, which is constantly listening.
Filter

The filter is required to attenuate frequencies outside of the required bandwidth or amplify signals in the bandwidth. As most signals will be operated at around 35kHz, the resonant frequency of the transducer, a 12Khz bandwidth for receiving was decided. This 12kHz bandwidth is a realistic passband for acoustic communications. The filter implemented is a 5-stage Tow-Thomas biquad filter. This 5 stage filter consists of one lowpass stage, and 4 bandpass stages. This Tow-Thomas biquad design was utilised because of its ease of design and implementation, along with its ability to amplify the passband, rather than simply attenuating signals not passed. This 5-stage design results in an approximately 50dB gain, for signals that are passed. The schematic for this filter can be seen in the Appendix. 

It was decided that the filter would be implemented in the hardware, due to the large voltage gains that could be achieved by doing so. Considering that the maximum range the underwater platform can communicate between, this becomes an important consideration. At large ranges, the received data signals can be of very low voltage levels which need to be amplified to higher levels, to allow for the digital processor to use them. This is because, generally, in digital systems, voltage levels very close to 0V are considered as being 0V.

A 5 Stage filter that has a 50dB gain is used in the design, as the larger the amplification of the signal the greater the maximum range that can be achieved by the communication platform. This 5 stage filter is a trade-off between cost, range and power. While a higher stage filter would increase the maximum range, it would also increase power consumption and substantially increase the cost, as each additional stage results in 3 more op amps. By itself, the filter costs approximately AUD $28, and is the most expensive section of the platform. As cost is a major consideration in the design of the hardware platform, 5 stages is the maximum that is really feasible without ignoring the project goals as further stages would likely cause the cost of the filter to be far too high.

A 5 stage Tow-Thomas biquad filter results in 15 operational amplifiers. This large number of op amps results in a large amount of power usage. For this reason, it is inadvisable to have the filter powered on and listening at all times, hence, to save battery, a wake-up circuit which turns the filter on was implemented.
The Filter depends on a positive and negative 5V voltage. To achieve this, a +5V DC/DC converter is used, which thereby supplies the necessary voltage for the filter to run. The DC/DC converter is usually off, hence the filter is usually off, unless an enable signal is received from the processor. This enable signal will only be received if the wake-up circuit observes a signal.

### Wake-up

The Wake-up circuit is an always on circuit, which constantly listens for a received signal. If a signal is received, and it is an actual data signal, the wake-up circuit will turn the filter on. It is designed to utilise relatively low power compared to the filter, to retain platform lifetime. The circuit is powered through regulated 5V voltage. The schematic can be seen in the Appendix.

The Wake-Up circuit uses a reference voltage and comparator to see if an actual transmitted signal has been received based on the voltage level, instead of just background noise being observed. If the circuit is designed to turn on any signal, the filter will be turned on far more often than required, taking a lot of power. If the circuit decides there is a received signal, it will send a high signal to a digital input/output pin of the processor. The processor is programmed to send an enable signal if this is high, which is used to activate the DC/DC Converter, which converts the signal to a +5V signal and a – 5V signal.

The wake-up circuit is valued at approximately AUD $9. The additional cost of the wake-up circuit is justified by the improvements it produces in the lifetime of each platform, which is substantial enough to overcome the increased cost, as the platform does not need to be replaced as often as it otherwise would.

### ADC

The receiver requires the use of an analog-to-digital converter. This ADC is required to convert the filtered signal into the equivalent digital signal, so that the processor is capable of using the information. The ADC used in this hardware platform, the ADCS7478 from Texas Instruments, has a 12bit resolution with a sampling rate of 1 Million Samples per Second (MSPS). As the sampling rate is far greater than required, where the minimum required is the nyquist rate which in this situation is 92Khz per second, the ADC is capable of perfect reconstruction of the signal. As the maximum frequency which will pass through the filter is
41kHz, 92Khz is the minimum double. As well, some of the sampling rate can be used to force the resolution to be increased if required. The ADC requires a clock input from the BeagleBone Black and also has an enable signal, allowing it to be activated only when receiving data or otherwise programmed to.

The ADC costs approximately AUD $4. The cost of this section of the platform could be decreased by utilising a different ADC with a lower sampling rate. However, this would offer fewer options, in particular through improving the resolution of the ADC. The schematic for the ADC is relatively simple and can be seen in the Appendix.

Other Components
Other components of the board include the placement of two male connector headers, each of 2x23 pins. The two 2x23 male headers are designed to be exactly compatible with the connectors on the BeagleBone Black, of equal distance apart on the board, to allow for easy connection of the designed transceiver and the BeagleBone Black.

Two other connectors are placed on the board. This includes the headers for power input from the battery, a 4 pin header, and a 2 pin header for the connections to the transducer, placed on opposite sides of the board.

Also included on the Board are a number of mounting holes, for mounting of the PCB on the Beaglebone Black in the same locations.

BeagleBone Black Pins
A number of pins on the BeagleBone Black are utilised. P9, one of the headers, has a number of these important pins. The Appendix has the schematic detailing each of the pins of the BeagleBone Black. P8 is not used, except for connecting between board purposes.

Of these pins, a number are very important to the functioning of the hardware platform. Of particular import are pins 1 and 2 (GND), which must be grounded and pins 5 and 6 (Vdd_5V), which are capable of powering the BeagleBone from the PCB with a regulated 5V signal by supplying 5V to the pins. This is how the development board is powered in the overall hardware platform.
A large number of other digital input and output pins were used. In most cases these were used as enable signals, where if data is inputted into a pin, the processor sends out a high or low signal on another pin, enabling a piece of hardware. The other key pins used, are pins 21 and 22 (PWM enabled pins), which are used for outputting the signal processed data. Another digital input pin is used to receive the filtered data signal from the receiver.

Of these pins, the majority, excluding the ground and voltage pins, are programmable through the BeagleBone Black. This is particularly important for enabling PWM signals and other enable signals.
3D Board Layout

The following images show a 3D image of the designed PCB. Image 2 presents the top layer, and image 3 the bottom layer.

Image 2 - Top Layer PCB 3D
The board’s dimensions are 54.86mm x 86.87mm. The PCB is designed to be the same size as the BeagleBone Black, to allow for ease of connection between the two, and to save space, as it is likely the hardware platform will be contained in small volumes.

Other key dimensions include the locations of the 2x23 header connectors, and the mounting hole locations. The 2x23 connectors have a gap of 45.72mm between them along the y-axis. When connecting PCB to the BeagleBone Black, the connectors are slightly unaligned along the y axis causing difficulty when connecting; further PCB orders should fix this minor issue.

**Soldering**

Due to the nature of the footprints and a number of the components being of small size and some of very small size, a finer tip is used when soldering. However, for some of the components used, hand soldering is not possible; hence reflux soldering is used for accuracy.
Final Hardware Platform

The final hardware platform will consist of the populated designed transceiver board, connected directly on top of the BeagleBone Black through the two 2x23 header connectors and the mounting holes. The battery will be located off the platform and connected to the board through a 4 pin right angled header connector. The transducer will be connected to the board through a 2 pin right-angled connector, also located off the board.

Overall Cost

The approximate overall cost of a single hardware platform, including the transducer and BeagleBone Black but not the battery and minus the cost of shipping is AUD $167.74. The cost of each component (including shipping costs) and the brand of make can be seen in the Bill of Materials, attached in the Appendix. For an underwater communications platform, albeit one only capable of low data rates, the cost is very low compared to other commercial alternatives, such as the LinkQuest System.

The cost is more expensive than originally expected, due to the required addition of a battery protection circuit and the number of stages used in the Tow-Thomas biquad filter. Any further decrease in cost is a wanted feature involving further research into the platform. It is also suspected that the cost of each platform would decrease to a relatively large extent if components were ordered in larger volumes.

Current Issues with Hardware

The Diode D3, which is the component LT1389 from Linear Technologies, has the wrong connected pins in the schematic and PCB. The footprint is correct, however, the nets are not connected to the correct pins. Testing can still be done without this functioning, as it remains part of the battery protection circuit and does not perform an essential function to the actual operation, although no under voltage protection would work. This issue can be overcome for testing by bridging the correct nets; however, this is only a temporary fix and should be corrected when ordering more boards.

A likely issue with the hardware will be when receiving and transmitting at the same time. This is due to the top level design not having any mechanism to prevent transmission and receiving on the same net. This will be investigated when further testing of the board occurs.
Improvements to the Design

As mentioned, currently the top level design dictates that the transceiver receives and transmits through the same net, which can cause issues if transmitting and receiving at the same time. It is recommended that this could be improved through the use of an enable signal for the wake-up circuit, which currently does not have one, to force the wake-up circuit to be active only when not transmitting data. It could also be improved through the use of a mosfet switch, or hardware circuit, that determines which path the signal passes through, dependent on if receiving or transmitting, by utilising differences such as the large voltage level.

Large improvements could be made to the design in terms of power saving, through boosting the voltage only when transmitting and by having the boost converter switched off for the majority of the time. The most likely way of implementing this is through the use of a boost converter that is capable of being switched on and off through the use of an enable signal. This would require more use of the processor, which is not detrimental, but greatly saves power usage and the lifetime of the hardware platform.

A trade-off between cost and amplification of the filter can be achieved through decreasing the number of stages of the filter, if it is found that fewer stages would allow the system to have the required range. The cost could also be decreased by using different operational amplifier packages. It is also likely the cost of the battery protection circuit could be decreased, by utilising a different battery of equal Ampere-hour’s but that allows for the design of a more simple circuit. Other designs could be investigated that work with the current model and that could be achieved with lower capital.
GNU Radio

This section details the workings of GNU radio, and the design of a simple Phase-Shift Keying modulated audio signal, which can be outputted from the processor and sent through the hardware platform. It will be used for the test case of the hardware platform, as well as the quality tests, including range and bit rate.

The PSK modulator is created entirely in the Graphical Radio Companion using already developed blocks, or libraries, packaged with the program and developed by the open-sourced community. This is to achieve the greatest ease of development and eventually, allow for the greatest ease of re-configurability of the platform by allowing run time manipulation of the signal processing. Secondly, the ease of use of the GRC allows for less experienced programmers to change the functionality and presents a simple method of developing test cases for the designed hardware platform.

The GNU Radio code is split into the transmitter and receiving sections for ease of testing. They will later be combined, with the process that is to happen being decided by the programing of the BeagleBone Black. This section splits the discussion into these two sub-sections. Most importantly, this section serves as a proof-of-concept for using GNU Radio, limited to PC sound cards sampling rates, and is not investigated to the same detail as the hardware design.

Phase-Shift Keying

Phase-Shift Keying (PSK) is the modulation scheme utilised for the test GNU Radio case. PSK is a digital modulation scheme that changes the phase of a signal to convey information. Each phase of the signal is assigned a unique pattern of binary digits and these binary digits are a symbol which represents a unique phase. The modulation stage of PSK involves the mapping of these symbols to a phase and then transmitting the signal, while the demodulation stage maps the phase back to the correct symbol, hence recovering the data sent. Differential Phase-Shift Keying (DPSK) is the type of PSK used in this system. This allows for the demodulator to work out the correct symbol through the changes in phase rather than using a reference signal. DPSK is used as it is simpler to implement than PSK, as no reference signal is required. This ease of use does transfer over to GNU Radio.
Transmission

The transmission side of the GNU Radio signal processing focuses on the modulation of the data, by mapping the data to the correct phase. It also includes a number of tools for viewing and debugging. It outputs the modulated signal through a file.

Figure 3 - Transmission GRC Flow Chart

Figure 3 presents the GRC flow chart and shows the transmission side of the GRC developed through the use of blocks. Each block is a developed open source library, with the characteristics being decided in the blocks. The file source block is responsible for outputting the test data through the rest of the graph. It accepts data from a text file at the specified file location, and outputs the data as a byte form. In future applications, it is likely that the type of data and where the data is gathered from, will depend on the designed transceiver and processor used. The sampling rate of the current test system is set to 44.1K, although real application when uploaded to the hardware platform will likely require a sampling rate of at least 92k due to the designed bandwidth of the system. 44.1K is used, as it is a default sampling rate of a PC soundcard.
The bytes to symbol block, scope sink and histogram sink, are merely used to see the characteristics of the outputted data. The bytes to symbol block is responsible for mapping bytes to either + or − 1 real values, depending on the symbol.

The PSK modulator is the most important block in the GRC. It applies PSK modulation to the data. Due to the options selected in the block, it also applies differential encoding, and utilises a Gray code scheme. As the number of constellations selected is 4, it hence has a 4 symbol alphabet (i.e it assigns a symbol for every two bits, with a max of 4 to select from).

The data outputted from the modulator is the complex modulated signal, which contains the data. The signal is then put through sinks to analyse the signal, and conversions to allow it to be outputted as audio through the PC soundcard. The audio level of transmission in this case, can be increased by increasing the volume level of the PC.

The resulting code of the GRC transmitter can be seen in the Appendix.

**Receiving**

The receiving side of the GNU Radio signal processing focuses on the demodulation of the incoming data, by mapping the phase to the correct symbol, to collect the correct bit values.

**Figure 4 - Receiving GRC Flow Chart**
Figure 4 shows the receiving side of GNU Radio, developed in the GRC. It assumes a real input signal from the audio card, which would be sent from the transmitter. However, any ambient noise would also be inputted into the system. After the data is received, it is converted to the appropriate complex form to undergo PSK demodulation. This demodulates the signal, with all options being set to the same as the PSK modulator, in terms of encoding and the number of constellation points, Gray code and samples/symbol. The data is then outputted as the byte, which assuming everything has progressed correctly, will output data as a byte that is the same as what was transmitted previously.

The flow chart includes a number of instrumentation blocks, for analysing the signals passing through.

The resulting code can be seen in the Appendix.

Current Issues
A number of issues currently affect the developed GNU Radio code. Of the most import is the difficulty had in outputting the modulated data through a digital output. This could likely be accomplished through appropriate changes to the code, as it works successfully when using an audio output block. The same issue is faced when attempting to store the demodulated bytes into a file on the receiver.

Another important difficulty is the modulating of the signal, when only one data stream is inputted. While the modulation works in general if the data to be modulated is repeated, no data appears to be modulated or transmitted when the data is only sent once. This could be due to the process stalling if there is enough data coming through for the code to run. Another reason for this issue could be the software requiring more data to successfully modulate. A likely method of solving the matter could be the addition of message parsing and framing of messages to the code.

Also to be implemented is the up conversion of the frequency of the data signal when transmitting to the resonant frequency (35kHz). This is required when applying the SDR to the final hardware platform.

A problematic issue occurs in the receiver flow chart. When the data is converted to a complex with the addition of a null signal, this is no longer an exact representation of the
transmitted signal, even if the real signal is the same. This could likely be overcome using the actual DPSK blocks instead.

**Improvements to GRC**

A large number of improvements could be made to the GNU Radio software. These include message parsing, framing data and error correction code. Message Parsing could allow for the solving of one of the issues mentioned previously.

Framing data could prevent the difficulty in modulating a signal, by adding redundant bits to tell the modulator and other code when to run. It would also help in the transmission and receiving of data, by allowing the receiver to know when it is about to receive actual data.

Error correction code is a fundamental improvement that could be made to the GNU Radio code. It would allow for a large reduction in the number of errors which occur in transit allowing for a more reliable communication platform. These error correction codes, which are capable of detecting and correcting errors such as bit flips and swaps, would allow the effective bit rate of transmission to also increase, as there would be less need for actions such as retransmission of the data as the data would more often be correct. There are a number of error correction codes which have been developed as blocks within GNU Radio’s library.

A number of other small improvements can be made to the current GRC flow charts when testing. These include changing the sample rate of the program for optimal bit rate and by changing the number of distinct phases used for modulation and demodulation of PSK.
Testing

The following section focuses on detailing the testing methodology, the testing currently completed and testing still required.

Methodology

The testing method is split into two sections relating to GNU Radio and the Hardware Platform, before software is uploaded into the processor. At the current time of this Thesis, the required testing has only partially been completed, so this section will also detail how the testing should proceed.

For GNU Radio, testing will first focus on achieving communication between two PC’s utilising their sound cards. After improvements have been made to the system, including the improvements mentioned so back and forth communication has been achieved, different modulation schemes should be utilised.

The Hardware Platform will only be fully tested when all programming has been completed, although voltage tests when power is being supplied can be utilised. After being completely programmed and the GNU Radio uploaded, testing will take place first in underwater environments.

The result of the final hardware platform will be evaluated according to the initial desired final goals. These include cost, bit rate, range and power usage. The cost of the system is already known. If the platform meets the initial goals outlined in the introduction of this Thesis, then the platform can be considered a success.

GNU Radio testing will be accomplished originally in terrestrial environments using two PC soundcards, and microphones for receiving, directed in the appropriate direction. The distance between the two sound cards (or sound card and microphone), will be experimented with to witness the effect this has, and the audio level of the sound cards will also be manipulated, to simulate changing the voltage level and to test the effects amplification of the signal will have. For underwater testing, the soundcard should be placed in a waterproof vehicle allowing unhindered transmission of acoustic signals, to test how the underwater environment effects transmission.
Hardware testing will be first based in controlled underwater environments, using two completed platforms with transducers. Following successful testing in controlled environments, testing will take place in a riverbed or ocean, in both deep and shallow water, to determine the platforms overall characteristics.

**Current results**

Currently, only a small amount of testing has been accomplished, limited to the testing of basic transmission between PC’s running the GNU Radio code. Figure 5 shows the real modulated signals transmitted, and Figure 5 shows the received, non-demodulated signal.

*Figure 5 - Transmitted Signal*
The x axis for each Figure shows the time of the signal, and the Y axis shows the amplitude of the signal transmitted or received.

This test was accomplished by using PC sound codes attached to microphones and running GNU Radio. The microphones would be placed a relative distance apart, within the same room. At a predefined audio level, one PC would run the transmission code and monitor the output. The other PC would run the receiving code, and monitor the input.

The Figures show that successful transmission of signals can be achieved. However, they also show that the voltage levels of the transmitted acoustic signal fall between transmission and receiving. This has a drastic effect on the range of the platform, as at a relatively small distance (the length of a small room) and with average audio levels, the voltage levels are slightly less than a tenth of the original transmission, implying, at these settings, low effective range. It can be seen that the signal on the transmission side and the receiving side are not an exact representation. This occurs due to time differences between the transmission side and receiving side, the addition of noise to the system from the channel and time delays from processing.
Figure 7 shows how the voltage level decreases at a lesser rate when the microphones are closer together and Figure 8 shows the lesser decrease when the audio on the sound cards are increased.
As can be seen from Figure 7, decreasing the distance decreases the voltage drop, showing the effect the range between hardware platforms will have. At this low level, the effects noise has can also be more easily seen. Figure 8 shows that by amplifying the voltage level sent and received, the range of the system can increase greatly.

Further testing is still required.

**Future Work**

There is a large amount of future work to both complete the project and to extend it.

Required work to complete the project includes: fixing any outstanding issues detailed in their respective sections; applying any recommendations mentioned previously; uploading GNU Radio to the BeagleBone Black and making it compatible with the designed hardware; other programming required for the BeagleBone Black and the completion of testing for the overall software defined acoustic communications hardware platform. Future work to extend the project includes: the investigation of different modulation schemes and application of them; extending the hardware platform to be full-duplex; investing methods to reduce the cost of the platform; the addition of extra functionality and potentially the networking of multiple platforms.

Firstly, issues with GNU Radio must be fixed and the transducer is to be potted with the potting compound. Following this, the finished GNU radio should then be uploaded to the BeagleBone Black, part of the hardware platform. This should not be an issue, as the BeagleBone Back runs Linux natively, which is the preferred OS for using GNU Radio, although it is likely that a number of particular libraries will have to be installed. However, part of the GNU Radio code, particularly the code which results in inputs and outputs, must be made usable with the hardware platform, which will likely require testing.

Once Gnu Radio has been completed and uploaded into the system, the BeagleBone Black is required to have additional programming. The majority of this programming will include coding for the appropriate enable signals used throughout the hardware, such as sending a high enable signal out if the wake-up circuit sends a high signal into the BeagleBone. Of most import, particular programming is required for the PWM outputs to function correctly.
As the PWM outputs are the actual data to be transmitted, this is vital to the overall system. Any additional features to be programmed will be added later.

The completion of all the programming allows for the first fully functional prototype of the overall hardware platform to be tested. Hence, the testing still required, as detailed previously, can be attempted. This allows for any further issues that arise to be analysed and fixed, as well as for the characteristics of the platform to be detailed.

Through the completion of the platform, there does exist room for further exploration of improvements. Improvements for both GNU Radio and the designed transceiver have been suggested in their sections. However, the exploration of different modulation schemes used in GNU Radio, and the appropriate testing of these schemes implemented in the finalised hardware platform, could benefit the overall project. While PSK offers a number of advantages with ease of use one of the greatest, other modulation schemes could provide multiple advantages to bit rate and error rate. Modulation schemes that would be worth investigating include Frequency Shift Keying (FSK) and Orthogonal Frequency-Division Multiplexing (OFDM). Both of these are capable of being implemented in the GRC of GNU Radio, using already developed blocks, making them good alternatives by offering the required re-configurability of the system. FSK is a common multiplexing scheme which would remain relatively easy to implement but would be limited to frequencies within the designed bandwidth (29kHz to 41kHz), which is enough for a decent sized alphabet of symbols. However, it is generally agreed that FSK for audio signals are not used for high-speed communications, due to being less efficient in power and bandwidth, suggesting a small increase in bit rate. OFDM offers a number of benefits compared to FSK and PSK however, it is a far more complex modulation scheme which could cause issues when attempting to re-configure the system. The biggest benefit it offers is its ability to cope with extreme conditions such as multipath, which is one issue largely in faced underwater communication systems. This suggests that OFDM could offer a greater increase in reliability of transmission.

A very useful extension to the project could be investigating ways to make the current designed PCB a full-duplex system (simultaneous transmission and receiving), rather than the current half-duplex system that it is. The current reason for the system being half-duplex
is due to the top level design, in which the same path is used for transmission of the signal from the H-bridge to the transducer and/or receiving from the transducer to the wake-up and filter, controlled by enable signals. This means that only transmission or receiving can happen at one time. Generally, this will not be an issue as the system is in default listening mode and will only transmit in short bursts when the system has something to send. However, issues could arise when any full-duplex situation is required, or any situation where the beginning of the receiving data suggests the platform begins transmission.

Further investigations into reducing the cost of the platform could also be attempted. While this may require some sections of the transceiver to be redesigned, it could prove more cost effective and further improve the validity of the platform.

By networking the platforms together, the total capability of the platform drastically increases. Through the addition of networking capabilities, a number of potential uses can be achieved, including sensor networks and baseline positioning techniques. It will allow for the developed platform to communicate easily with all of the platforms in its network and the sharing of information.

Additionally, GPS functionality could be added to the platform. This would increase the amount of information available to the platform, and with the addition of networks, would greatly improve its capability and overall instrumentation abilities.
Conclusion

The Thesis succeeds in creating a first prototype for a software defined underwater acoustic communications hardware platform. The hardware platform, consisting of the in house designed transceiver, the BeagleBone Black as the processor, the Steminc Transducer, and a 2 cell lithium ion battery, form the building blocks of this hardware platform. The platform is of a small size factor, capable of fitting within small Autonomous Underwater Vehicles and other casings, which is a requirement of the design. The designed transceiver, capable of all the required functions between the processor and the transducer, is the main deliverable produced from the project.

The cost of the hardware platform meets the design criteria outlined in the introduction. The total cost of a single hardware platform is calculated from the Bill of Materials to be AUD $167.74. This cost is significantly smaller than similar commercial and research platforms. At the cost found, the platform can be used in large quantities by the majority of prospective users and allows for the platform to be disposable and employed in large networks and arrays without being cost prohibiting. While it is likely that the cost of the platform could be decreased, the transducer used is one of the cheapest available. Along with the processor used, the design takes advantage of this, so that any further reduction in cost would likely require increased investigation and redesign of the platform.

The designed transceiver is also significant in minimising power usage, albeit a number of improvements can be made particularly to the boost converter. The transceiver is capable of minimising the power drawn from the circuit through the wake-up circuit. Most circuitry is designed to be low power usage. With the large Ampere-hours afforded by the battery, and the low power usage, the platform should theoretically last for lengthy periods, meeting the power usage design requirement, although exact characteristics cannot be found without further testing.

The Thesis also succeeds in proving the possibility of use of GNU Radio in an underwater environment, with the designed platform as a method of implementing software defined radio. The proof of concept does find, by merely using PC sound cards, that transmission and receiving is capable in a form that is simple to implement and re-configure, and that it is capable of being run sufficiently by the processor used in the hardware platform. It presents
the estimated extent of the distance between two platforms and what this will have on the
effectives of the software defined radio through voltage manipulation. This is implemented
in the designed hardware transceiver, showing that the effective range of the platform can
be greatly increased.

The other characteristics of the hardware platform that have been nominated as detailing
the effectiveness of the project; that of range, and in particular data rate and error rate,
require further work and testing to be found. The overall effectiveness will depend on the
values found for these two characteristics.

Of the suggested deliverables of the project, the prototype hardware platform has been
produced, which serves as the main deliverable product, and the proof of concept GNU
Radio code. The other deliverables, which include that of the finished software, GNU Radio
and combined platform, remain incomplete.

Further, this project is capable of being extended in a variety of ways, and further research
is recommended.
Bibliography


Karpeles, M, Torres, D, *Underwater Acoustic Communications*, University of California


Stojanovic, M, ‘Underwater Acoustic Communication’, Department of Electrical and Computer Engineering, Northeastern University, Boston.

Nowsheen, N, 2011, ‘Design of a High Frequency FPGA Acoustic Modem for Underwater Communication’, School of Engineering and Information Technology, University of New South Wales


# Appendices

## Bill of Materials

<table>
<thead>
<tr>
<th>Description</th>
<th>Designator(s)</th>
<th>Quantity</th>
<th>Value</th>
<th>Manufacturer Name</th>
<th>Manufacturer Part Number</th>
<th>Element 14 Order Number</th>
<th>Cost</th>
<th>Cost for one part</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boost Converter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.9 nF Capacitor 0805</td>
<td>Ccomp2</td>
<td>2</td>
<td>3.9nF</td>
<td>TDK</td>
<td>CGA4J2C0G2A392J125A A</td>
<td></td>
<td></td>
<td>$0.840</td>
</tr>
<tr>
<td>100 nF Capacitor 0805</td>
<td>Ccomp_1, Cinx_1, Cbyp_1</td>
<td>6</td>
<td>100nF</td>
<td>TDK</td>
<td>C2012X7R2A104M125A A</td>
<td>1844411</td>
<td>$0.360</td>
<td>$0.060</td>
</tr>
<tr>
<td>1nF Capacitor 0805</td>
<td>Ccs_1</td>
<td>2</td>
<td>1nF</td>
<td>TDK</td>
<td>C2012C0G2A102J060AA</td>
<td>2284292</td>
<td>$0.160</td>
<td>$0.080</td>
</tr>
<tr>
<td>100 uF Electrolytic Capacitor 6.3 x 5.8</td>
<td></td>
<td></td>
<td></td>
<td>Nichicon</td>
<td>UUD1C101MCL1GS</td>
<td></td>
<td>$0.600</td>
<td>$0.300</td>
</tr>
<tr>
<td>1 uF Capacitor 0805</td>
<td>Cout1</td>
<td>2</td>
<td>1uF</td>
<td>TDK</td>
<td>C2012X5R1E105K125AA</td>
<td>1962124</td>
<td>$0.160</td>
<td>$0.080</td>
</tr>
<tr>
<td>22 uF Electrolytic Capacitor 10x20mm</td>
<td>Cout_1</td>
<td>2</td>
<td>22uF</td>
<td>Panasonic</td>
<td>EEUED2D220</td>
<td>9696261</td>
<td>$2.580</td>
<td>$1.290</td>
</tr>
<tr>
<td>8.2nF Capacitor</td>
<td>Css_1</td>
<td>2</td>
<td>8.2nF</td>
<td>AVX</td>
<td>08051C822KAT2A</td>
<td>1582725</td>
<td>$0.458</td>
<td>$0.229</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0805</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shottky Diode 1.78mm x 4.43mm</td>
<td>D1</td>
<td>2</td>
<td>850mV</td>
<td>Taiwan Semiconductors</td>
<td>SK310A-TP</td>
<td>1299279</td>
<td>$2.020</td>
<td>$1.010</td>
</tr>
<tr>
<td>82 μH Inductor 10.0 x 10.2mm</td>
<td>L1</td>
<td>2</td>
<td>82uH</td>
<td>Coilcraft</td>
<td>MSS1038-823MLC</td>
<td>2288216</td>
<td>$2.480</td>
<td>$1.240</td>
</tr>
<tr>
<td>80V mosfet</td>
<td>M1</td>
<td>2</td>
<td>80V</td>
<td>Infineon Technologies</td>
<td>BSC340N08NS3 G</td>
<td>2212853</td>
<td>$2.180</td>
<td>$1.090</td>
</tr>
<tr>
<td>4.53k Resistor 0805</td>
<td>Rcomp_1</td>
<td>2</td>
<td>4.53k</td>
<td>Vishay-Dale</td>
<td>CRCW08054K53FKEA</td>
<td>1653012</td>
<td>$0.206</td>
<td>$0.103</td>
</tr>
<tr>
<td>649 Resistor 0805</td>
<td>Rfb1</td>
<td>2</td>
<td>649</td>
<td>Vishay-Dale</td>
<td>CRCW0805649RFKEA</td>
<td>1185108</td>
<td>$0.040</td>
<td>$0.020</td>
</tr>
<tr>
<td>24k Resistor 0805</td>
<td>Rfb2</td>
<td>2</td>
<td>24k</td>
<td>Vishay-Dale</td>
<td>CRCW080524K0FKEA</td>
<td>1652964</td>
<td>$0.186</td>
<td>$0.093</td>
</tr>
<tr>
<td>100 Resistor 0805</td>
<td>Rs1</td>
<td>2</td>
<td>100</td>
<td>Vishay-Dale</td>
<td>CRCW0805100RFKEB</td>
<td>1767763</td>
<td>$0.280</td>
<td>$0.140</td>
</tr>
<tr>
<td>8.66k Resistor 0805</td>
<td>Rs2</td>
<td>2</td>
<td>8.66k</td>
<td>Vishay-Dale</td>
<td>CRCW08058K66FKEA</td>
<td>1610017</td>
<td>$0.160</td>
<td>$0.080</td>
</tr>
<tr>
<td>75m Resistor 0805</td>
<td>Rsns_1</td>
<td>2</td>
<td>75m</td>
<td>Panasonic</td>
<td>ERJ6RQFR75V</td>
<td>1892962</td>
<td>$0.516</td>
<td>$0.258</td>
</tr>
<tr>
<td>64.9k Resistor 0805</td>
<td>Rt_1</td>
<td>2</td>
<td>64.9k</td>
<td>Vishay-Dale</td>
<td>CRCW080564K9FKEA</td>
<td>1278123</td>
<td>$0.064</td>
<td>$0.032</td>
</tr>
<tr>
<td>3.01k Resistor 0805</td>
<td>Ruv1</td>
<td>2</td>
<td>3.01k</td>
<td>MULTICOMP</td>
<td>MC01W080513K01</td>
<td>2141817</td>
<td>$0.012</td>
<td>$0.006</td>
</tr>
<tr>
<td>Description</td>
<td>Quantity</td>
<td>Value 1</td>
<td>Value 2</td>
<td>Part Number</td>
<td>Cost 1</td>
<td>Cost 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>----------</td>
<td>---------</td>
<td>---------</td>
<td>------------------------------</td>
<td>----------</td>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10k Resistor 0805</td>
<td>2</td>
<td></td>
<td></td>
<td>CRCW080510K0FKEB</td>
<td>$0.280</td>
<td>$0.140</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60V Low Side Controller for Boost and SEPIC</td>
<td>2</td>
<td></td>
<td></td>
<td>LM5022MM/NOPB</td>
<td>$5.460</td>
<td>$2.730</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total Cost:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$19.042</td>
<td>$9.403</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Transducer, potting and processor</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Piezo Ceramic Cylinder 26x22x13mm 43 KHz</td>
<td>2</td>
<td></td>
<td></td>
<td>SMC26D22H13111</td>
<td>Not available at Element 14</td>
<td>$76.890 + $48.69 Shipping</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BeagleBone Black</td>
<td>2</td>
<td>CIRCUITCO</td>
<td></td>
<td>SMC26D22H13111</td>
<td>$103.80</td>
<td>$51.900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONATHANE EN-12 Two-part Urethane Potting Compound</td>
<td></td>
<td></td>
<td></td>
<td>Cytek</td>
<td>Currently only found on manufactures site</td>
<td>Price requires contact</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total Cost:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$180.690</td>
<td>$114.690</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H-Bridge Piezo Driver</td>
<td></td>
<td></td>
<td></td>
<td>Infineon BTS4140N</td>
<td>$12.32</td>
<td>$1.540</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hex</td>
<td>2</td>
<td>Texas</td>
<td></td>
<td>SN74LVC07AD</td>
<td>$0.44</td>
<td>$0.220</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The shipping cost for the piezo ceramic cylinder is not included in the total cost.
<table>
<thead>
<tr>
<th>Description</th>
<th>Component Details</th>
<th>QTY</th>
<th>Value</th>
<th>Part Number</th>
<th>Price</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer/Drive r With Open-Drain Outputs (SOIC package)</td>
<td>R1,R2,R3,R4</td>
<td>8</td>
<td>20</td>
<td>CRCW0805270RJNEA</td>
<td>$0.17</td>
<td>$12.928</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1514845</td>
<td></td>
<td>$6.46</td>
</tr>
<tr>
<td>Biquad Filter</td>
<td></td>
<td></td>
<td></td>
<td>Total Cost:</td>
<td>$56.640</td>
<td>$28.32</td>
</tr>
<tr>
<td>Quad Op Amp</td>
<td>OA1,OA2,OA3,OA4</td>
<td>8</td>
<td></td>
<td>LT6012CS#PBF</td>
<td>$51.60</td>
<td>$6.450</td>
</tr>
<tr>
<td>887 Resistor 0805</td>
<td>R1</td>
<td>2</td>
<td>887</td>
<td>CRCW0603887RFKEA</td>
<td>$0.24</td>
<td>$0.030</td>
</tr>
<tr>
<td>8.25k Resistor 0805</td>
<td>R3,R9,R15,R21,R27</td>
<td>8</td>
<td>8.25k</td>
<td>MC01W08051%8K25FR</td>
<td>$1.38</td>
<td>$0.069</td>
</tr>
<tr>
<td>10k Resistor 0805</td>
<td>R5,R6,R11,R12,R17,R18,R23,R24,R29,R30</td>
<td>20</td>
<td>10k</td>
<td>CRT0805-FZ-1002ELF</td>
<td>$1.32</td>
<td>$0.066</td>
</tr>
<tr>
<td>2.61k Resistor 0805</td>
<td>R7,R13,R19,R25</td>
<td>8</td>
<td>2.61k</td>
<td>ERJ-6ENF2611V</td>
<td>$1.36</td>
<td>$0.170</td>
</tr>
<tr>
<td>10nF Capacitor 0805</td>
<td>C1,C2,C3,C4,C5,C6,C7,C8,C9,C10</td>
<td>20</td>
<td>10nF</td>
<td>0805YC103KAT2A.</td>
<td>$0.68</td>
<td>$0.034</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4538717</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Total Cost:</td>
<td>$56.640</td>
<td>$28.32</td>
</tr>
<tr>
<td>Description</td>
<td>Model/Part Number</td>
<td>Quantity</td>
<td>Manufacturer</td>
<td>Part Number</td>
<td>Price</td>
<td>Cost</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------</td>
<td>----------</td>
<td>-----------------------</td>
<td>----------------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>Wake Up Circuit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rms2DC Converter</td>
<td>RD1</td>
<td>2</td>
<td>Linear Technology</td>
<td>LTC1967CMS8#PBF</td>
<td>1330888</td>
<td>$16.30</td>
</tr>
<tr>
<td>Comparator</td>
<td>COM1</td>
<td>2</td>
<td>Texas Instruments</td>
<td>TL331JDBVRG4</td>
<td>1287653RL</td>
<td>$0.76</td>
</tr>
<tr>
<td>68.1k Resistor 0805</td>
<td></td>
<td>2</td>
<td>Multicomp</td>
<td>MC01W0805168K1</td>
<td>2141892</td>
<td>$0.01</td>
</tr>
<tr>
<td>10k Resistor 0805</td>
<td></td>
<td>2</td>
<td>Vishay-Draloric</td>
<td>CRCW080510K0FKEA</td>
<td>1469856</td>
<td>$0.13</td>
</tr>
<tr>
<td>15k Resistor 0805</td>
<td></td>
<td>2</td>
<td>Multicomp</td>
<td>MC080558F1502T5E</td>
<td>1631376</td>
<td>$0.11</td>
</tr>
<tr>
<td>45.3k Resistor 0805</td>
<td></td>
<td>2</td>
<td>Vishay-Draloric</td>
<td>CRCW080545K3FKEA</td>
<td>1653001</td>
<td>$0.07</td>
</tr>
<tr>
<td>220pF Capacitor 0805</td>
<td></td>
<td>2</td>
<td></td>
<td>CGA4C2COG2A221J060AA</td>
<td>2210935</td>
<td>$0.18</td>
</tr>
<tr>
<td>0.1uF Capacitor 0805</td>
<td></td>
<td>4</td>
<td></td>
<td>MC0805B104M160CT</td>
<td>1708714</td>
<td>$0.04</td>
</tr>
<tr>
<td>Total Cost:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$17.608</td>
</tr>
</tbody>
</table>

Analog to Digital Converter

<table>
<thead>
<tr>
<th>Description</th>
<th>Model/Part Number</th>
<th>Quantity</th>
<th>Manufacturer</th>
<th>Part Number</th>
<th>Price</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>12bit ADC 1MSPS</td>
<td>A1</td>
<td>2</td>
<td>Texas Instruments</td>
<td>ADCS7476AIMF/NOPB</td>
<td>1693926</td>
<td>$8.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Cost:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$8.020</td>
</tr>
</tbody>
</table>

Power
<table>
<thead>
<tr>
<th>Supplies</th>
<th>Part</th>
<th>Supplier</th>
<th>Part Number</th>
<th>Quantity</th>
<th>Cost 1</th>
<th>Cost 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V Voltage Regulator</td>
<td>XC</td>
<td>Torex</td>
<td>XC6216D502PR</td>
<td>1</td>
<td>$4.78</td>
<td>$2.390</td>
</tr>
<tr>
<td>DC/DC Converter with ctrl</td>
<td>RS0</td>
<td>Recom Power</td>
<td>RSO-0505D</td>
<td>1</td>
<td>$52.76</td>
<td>$26.380</td>
</tr>
<tr>
<td>NPN Transistor</td>
<td>Q1</td>
<td>MULTICOMP</td>
<td>BCW72</td>
<td>1</td>
<td>$0.023</td>
<td>$0.01</td>
</tr>
<tr>
<td>PNP Transistor</td>
<td>Q2</td>
<td>ROHM</td>
<td>DTA143ZUAT106</td>
<td>1</td>
<td>$0.021</td>
<td>$0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total Cost:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$57.584</td>
<td>$28.79</td>
</tr>
<tr>
<td><strong>Battery Protection</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P Mosfet</td>
<td>SW1</td>
<td>FAIRCHILD SEMICONDUCTOR</td>
<td>FDG330P</td>
<td>2</td>
<td>$2.78</td>
<td>$1.390</td>
</tr>
<tr>
<td>Voltage Reference Diode</td>
<td>D3</td>
<td>Linear Technology</td>
<td>LT1389BCS8-2.5#PBF</td>
<td>2</td>
<td>$18.34</td>
<td>$9.170</td>
</tr>
<tr>
<td>Operational Amplifier</td>
<td>AR1</td>
<td>Linear Technology</td>
<td>LT1495CS8#PBF</td>
<td>2</td>
<td>$14.20</td>
<td>$7.100</td>
</tr>
<tr>
<td>Fuse</td>
<td>F1</td>
<td>Littlefuse</td>
<td>0458003.DR.</td>
<td></td>
<td>$4.60</td>
<td>$2.300</td>
</tr>
<tr>
<td>Reverse Polarity Diode</td>
<td>D2</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total Cost:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$39.920</td>
<td>$19.96</td>
</tr>
<tr>
<td><strong>Overall Cost:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$384.41</td>
<td>$216.43</td>
</tr>
<tr>
<td><strong>Overall Cost Minus Shipping:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$335.92</td>
<td>$167.74</td>
</tr>
</tbody>
</table>
**GNU Radio Code**

The following is the Python code generated from the GRC’s. The library blocks are not included and are standard.

**Transmitter**

```python
#!/usr/bin/env python

# This is the Python code generated from the GRC’s. The library blocks are not included and are standard.

from gnuradio import audio
from gnuradio import blocks
from gnuradio import digital
from gnuradio import eng_notation
from gnuradio import gr
from gnuradio import window
from gnuradio.eng_option import eng_option
from gnuradio.gr import firdes
from gnuradio.wxgui import constsink_gl
from gnuradio.wxgui import fftsink2
from gnuradio.wxgui import histosink_gl
from gnuradio.wxgui import scopesink2
from grc_gnuradio import wxgui as grc_wxgui
from optparse import OptionParser
import wx

class transmitter(grc_wxgui.top_block_gui):
    def __init__(self):
        grc_wxgui.top_block_gui.__init__(self, title="Transmitter")
        _icon_path = "/usr/share/icons/hicolor/32x32/apps/gnuradio-grc.png"
        self.SetIcon(wx.Icon(_icon_path, wx.BITMAP_TYPE_ANY))

        # Variables
        self.samp_rate = samp_rate = 44100

        # Blocks
        self.notebook = self.notebook = wx.Notebook(self.GetWin(), style=wx.NB_TOP)
        self.notebook.AddPage(grc_wxgui.Panel(self.notebook), "Real Histo")
        self.notebook.AddPage(grc_wxgui.Panel(self.notebook), "Real Scope")
        self.notebook.AddPage(grc_wxgui.Panel(self.notebook), "Complex Constellation")
        self.notebook.AddPage(grc_wxgui.Panel(self.notebook), "Complex FFT")
```
self.Add(self.notebook)
self.wxgui_scopesink2_1 = scopesink2.scope_sink_c(
    self.notebook.GetPage(3).GetWin(),
    title="Scope Plot",
    sample_rate=samp_rate,
    v_scale=0,
    v_offset=0,
    t_scale=0,
    ac_couple=False,
    xy_mode=False,
    num_inputs=1,
    trig_mode=gr.gr_TRIG_MODE_AUTO,
    y_axis_label="Counts",
)
self.notebook.GetPage(3).Add(self.wxgui_scopesink2_1.win)

self.wxgui_scopesink2_0 = scopesink2.scope_sink_f(
    self.notebook.GetPage(1).GetWin(),
    title="Scope Plot",
    sample_rate=samp_rate,
    v_scale=0,
    v_offset=0,
    t_scale=0,
    ac_couple=False,
    xy_mode=False,
    num_inputs=1,
    trig_mode=gr.gr_TRIG_MODE_AUTO,
    y_axis_label="Counts",
)
self.notebook.GetPage(1).Add(self.wxgui_scopesink2_0.win)

self.wxgui_histosink2_0 = histosink_gl.histo_sink_f(
    self.notebook.GetPage(0).GetWin(),
    title="Histogram Plot",
    num_bins=27,
    frame_size=1000,
)
self.notebook.GetPage(0).Add(self.wxgui_histosink2_0.win)

self.wxgui_fftsink2_0 = fftsink2.fft_sink_c(
    baseband_freq=0,
    y_per_div=10,
    y_divs=10,
    ref_level=0,
    ref_scale=2.0,
    sample_rate=samp_rate,
    fft_size=1024,
    fft_rate=15,
    average=False,
    avg_alpha=None,
    title="FFT Plot",
    peak_hold=False,
)
self.notebook.GetPage(4).Add(self.wxgui_fftsink2_0.win)

self.wxgui_constellationsink2_0 = constsink_gl.const_sink_c(
    self.notebook.GetPage(2).GetWin(),
    title="Constellation Plot",
    sample_rate=samp_rate,
    frame_rate=5,
    const_size=2048,
M=4,
theta=0,
loop_bw=6.28/100.0,
fmax=0.06,
mu=0.5,
gain_mu=0.005,
symbol_rate=samp_rate/4.,
omega_limit=0.005,
}

self.notebook.GetPage(2).Add(self.wxgui_constellationsink2_0.win)
self.digital_psk_mod_0 = digital.psk.psk_mod(
    constellation_points=4,
    mod_code="gray",
    differential=True,
    samples_per_symbol=4,
    excess_bw=0.35,
    verbose=True,
    log=False,
)

self.digital_bytes_to_syms_0 = digital.bytes_to_syms()

blocks.throttle(gr.sizeof_gr_complex*1, samp_rate)
self.blocks_file_source_0 =
blocks.file_source(gr.sizeof_char*1, "/home/keelan/Honours/Hello World!", True)

self.blocks_complex_to_real_0 = blocks.complex_to_real(1)
self.audio_sink_0 = audio.sink(samp_rate, ",", True)

# Connections

self.connect((self.blocks_file_source_0, 0),
(self.digital_bytes_to_syms_0, 0))
self.connect((self.digital_bytes_to_syms_0, 0),
(self.wxgui_scopesink2_0, 0))
self.connect((self.wxgui_scopesink2_0, 0),
(self.blocks_file_source_0, 0))
self.connect((self.digital_psk_mod_0, 0),
(self.wxgui_histosink2_0, 0))
self.connect((self.wxgui_histosink2_0, 0),
(self.blocks_file_source_0, 0))
self.connect((self.digital_psk_mod_0, 0),
(self.blocks_throttle_0, 0))
self.connect((self.blocks_throttle_0, 0),
(self.wxgui_scopesink2_1, 0))
self.connect((self.wxgui_scopesink2_1, 0),
(self.blocks_throttle_0, 0))
self.connect((self.blocks_throttle_0, 0),
(self.blocks_complex_to_real_0, 0))
self.connect((self.blocks_complex_to_real_0, 0),
(self.audio_sink_0, 0))
self.connect((self.blocks_throttle_0, 0),
(self.wxgui_fftsink2_0, 0))
self.connect((self.blocks_throttle_0, 0),
(self.wxgui_constellationsink2_0, 0))

def get_samp_rate(self):
    return self.samp_rate

def set_samp_rate(self, samp_rate):
    self.samp_rate = samp_rate
from gnuradio import audio
from gnuradio import blocks
from gnuradio import digital
from gnuradio import eng_notation
from gnuradio import gr
from gnuradio.eng_option import eng_option
from gnuradio.wxgui import scopesink2
from grc_gnuradio import wxgui as grc_wxgui
from optparse import OptionParser
import wx

class Receiver(grc_wxgui.top_block_gui):
    def __init__(self):
        grc_wxgui.top_block_gui.__init__(self, title="Receiver")
        _icon_path = "/usr/share/icons/hicolor/32x32/apps/gnuradio-grc.png"
        self.SetIcon(wx.Icon(_icon_path, wx.BITMAP_TYPE_ANY))

        self.samp_rate = samp_rate = 44100

        self.wxgui_scopesink2_0_0 = scopesink2.scope_sink_f(
            self.GetWin(),
            title="Scope Plot",
            sample_rate=samp_rate,
            )

if __name__ == '__main__':
    parser = OptionParser(option_class=eng_option, usage="%prog: [options]")
    (options, args) = parser.parse_args()
    tb = transmitter()
    tb.Run(True)
v_scale=0,
v_offset=0,
t_scale=0,
ac_coupled=False,
xy_mode=False,
num_inputs=1,
trig_mode=gr.gr_TRIG_MODE_AUTO,
y_axis_label="Counts",
)

self.Add(self.wxgui_scopesink2_0_0.win)

self.digital_psk_demod_0 = digital.psk.psk_demod(
    constellation_points=4,
    differential=True,
samples_per_symbol=4,
excess_bw=0.35,
    phase_bw=6.28/100.0,
timing_bw=6.28/100.0,
    mod_code="gray",
    verbose=False,
    log=False,
)

self.blocks_throttle_0 =
blocks.throttle(gr.sizeof_gr_complex*1, samp_rate)

self.blocks_null_source_0 =
blocks.null_source(gr.sizeof_float*1)

self.blocks_float_to_complex_0 = blocks.float_to_complex(1)

self.blocks_file_sink_0 = blocks.file_sink(gr.sizeof_char*1,
"/home/keelan/Honours/Received")

self.blocks_file_sink_0.set_unbuffered(False)

self.audio_source_0 = audio.source(samp_rate, ",", True)

# Connections

self.connect((self.blocks_null_source_0, 0),
(self.blocks_float_to_complex_0, 0))

self.connect((self.blocks_float_to_complex_0, 0),
(self.blocks_throttle_0, 0))

self.connect((self.blocks_throttle_0, 0),
(self.digital_psk_demod_0, 0))

self.connect((self.audio_source_0, 0),
(self.blocks_float_to_complex_0, 0))

self.connect((self.digital_psk_demod_0, 0),
(self.blocks_file_sink_0, 0))

self.connect((self.audio_source_0, 0),
(self.wxgui_scopesink2_0_0, 0))

def get_samp_rate(self):
    return self.samp_rate

def set_samp_rate(self, samp_rate):
    self.samp_rate = samp_rate
    self.blocks_throttle_0.set_sample_rate(self.samp_rate)
    self.wxgui_scopesink2_0_0.set_sample_rate(self.samp_rate)

if __name__ == '__main__':
parser = OptionParser(option_class=eng_option, usage="%prog: [options]")
(options, args) = parser.parse_args()
tb = Receiver()
tb.Run(True)
Hardware Schematics
The schematics for the hardware design are on the following pages.
Power_in

XC6216D

C11 100pF

C12 100pF

GND

5V_reg

PIC1101

PIC1201

PIXC01

PIXC02

PIXC03

COC11

COC12

COXC

5V Regulator

1.0.0

Keelan Burns
ADC

Title

Number

Revision

Size

A4

Date: 14/11/2013

Sheet

File: C:\Users\..\ADC.SchDoc

Drawn By: Keelan Burns

ADC

ADC1

3

VIN

VDD

5

SDATA

ADCS7476AIMF/NOPB

1

SCLK

GND

2

CS

BB in

5V reg

BB_CLK

BB Enable

GND

POADCIN

POBB0CLK

POBB0Enable

POBB0IN

PIADC101

PIADC102

PIADC103

PIADC104

PIADC105

PIADC106

COADC1

PIADC107

PIADC108

PIADC109

PIADC110
Wake-Up receiver Circuit

- **C14**: 0.1uF Capacitor
- **R31**: 15K Ohm Resistor
- **R32**: 15K Ohm Resistor
- **R33**: 15K Ohm Resistor
- **R34**: 15K Ohm Resistor
- **R35**: 68K Ohm Resistor
- **R36**: 10K Ohm Resistor
- **C15**: 0.1uF Capacitor
- **C16**: 220pF Capacitor
- **C13**: 0.1uF Capacitor
- **C17**: 220pF Capacitor
- **5V_reg**: 5V Reference
- **GND**: Ground

**Components**
- **RD1**: RMS-to-DC Converter
- **TL331**: Comparator

**Connections**
- RD1 connects to C14 and C15
- C14 is connected to 5V_reg and GND
- C15 is connected to 5V_reg and GND
- C16 is connected to R33 and GND
- R31 is connected to C15 and GND
- R32 is connected to C15 and GND
- R33 is connected to C16 and GND
- R34 is connected to R33 and GND
- R35 is connected to R34 and GND
- R36 is connected to R35 and GND
- Wake_Up receiver Circuit

**Labels**
- **PIEZO**: Piezo
- **Wake_Up**: Wake-Up receiver Circuit
- **5V_reg**: 5V Reference
- **GND**: Ground