AquaSense

November 2, 2009

Dr. John Bird
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 Design Specifications for a Qualified Sonar System for the Masses

Dear Dr. Bird:

The enclosed document “Design Specifications for a Qualified Sonar System for the Masses” details and justifies the design choices of both the AquaScan sonar system and several candidates for a qualified, inexpensive analog receiver. The AquaScan is a testbed for the research, development, and evaluation of analog receiver schemes, for the obtaining of a receiver which optimizes the signal-to-noise ratio and coherent diffuse ratio within cost constraints.

It should be noted that our design specifications address only the essential functionalities detailed in our previously submitted functional specifications document. Upon completion of the initial project, should time and budget constraints allow, the remaining non-essential functionalities will also be pursued.

If any questions or concerns arise regarding our design specifications, feel free to contact me by phone at (604) 807-9823 or by email at ksw3@sfu.ca.

Sincerely,

Wu Shi Quan

Ken Wu
President and CEO
AquaSense System

Enclosure: *Design Specifications for a Qualified Sonar System for the Masses*

Design Specifications for a Qualified Sonar System for the Masses

Project Team: Ken Wu
Joseph Wong
Mincong Luo
Hamed Madani

Contact Person: Ken Wu
ksw3@sfu.ca

Submitted to: John Bird – ENSC 440
Steve Whitmore – ENSC 305
School of Engineering Science
Simon Fraser University

Issued Date: November 2, 2009

Revision: 1.0

EXECUTIVE SUMMARY

This document presents the design choices made for the AquaScan sonar system and their justifications in terms of theoretical arguments, practical considerations, and supporting references. It begins with a high-level overview of the entire system in terms of its major functional blocks, and provides a brief description of each. It then provides an in-depth discussion into the design of the AquaScan in terms of its four major aspects: the testbed hardware, the candidate receiver modules, the graphical user interface, and the firmware. The results of an initial cost and noise performance evaluation, which are the underlying tenets in the development of the AquaScan, are presented at the end of this discussion. Finally, test plans are detailed to ensure a reliable product can be released into the marketplace.

The design details and their justifications cannot be done justice here; therefore, instead of attempting to summarize them here, we would present the results of our development up til this present time. The pertinent points are detailed briefly as follows:

1. The testbed is nearing completion, and one candidate receiver has already been researched, developed, and its evaluations begun.
2. The testbed is capable of varying the carrier parameters within all its required ranges while simultaneously providing a synchronized clock for coherent demodulation. The ADC modules are also fully functional at this time.
3. The candidate receiver is a Tayloe detector, and has achieved an SNR of 34.89 dB. According to N. Neretti, N. Intrator, and L.N. Cooper in 2005 (this is [32] in the main document), a high SNR for a receiver is above 20 dB. Further design optimizations are have already been identified to boost the SNR above 40 dB.
4. The AquaScan is able to recover the in-phase and quadrature components of the received signal without the timing issues plaguing the previous design which we are replacing.
5. The user interface has been completed, and is capable of producing a quasi real-time plot of the in-phase and quadrature components of the received signal.
6. The estimated per-unit cost of a six-channel sonar system based upon the Tayloe detector, not including the cost of the supplied URL transducer driver, comes to $223.32; this is well under the $300 mark estimated for the system.
7. Several other promising candidate receivers have been identified, and further investigations will commence shortly.

It must be emphasized again that the cost and noise performance are the main criterion of the AquaScan sonar system, and pointed out that current development has already achieved these goals. With many major obstacles already surmounted, the road to a qualified, inexpensive sonar system for the masses appears very smooth and clear.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

A/D	Analog-to-Digital
ADC	Analog-to-Digital Converter
BPF	BandPass Filter
COM	Component Object Model
CDR	Coherent Diffuse Ratio – the ratio of the power in the coherent part of a signal to the power in the noise part of the signal; for a variable x, this is defined in dB as $20\log(\mu_X / \sigma_X)$
DAC	Digital-to-Analog Convertor
EIA	Electronic Industries Alliance
GUI	Graphical User Interface
I	In-Phase
IF	Intermediate Frequency
ISR	Interrupt Service Routine
LPF	LowPass Filter
MCU	MicroController Unit
MIPS	Million Instructions Per Second
PWM	Pulse-Width Modulation
Q	Quadrature
RF	Radio Frequency
RJ-11	Registered Jack 11
RS-232	Recommended Standard 232
SFU	Simon Fraser University
SNR	Signal to Noise Ratio – the ratio of desired signal power to noise power; in dB this is defined as $10\log(P_{signal}/ P_{noise})$
UART	Universal Asynchronous Receiver/Transmitter
URL	Underwater Research Lab
USB	Universal Serial Bus
VFA	Voltage Feedback Amplifier
VGA	Variable Gain Amplifier

1. INTRODUCTION

AquaSense systems has been commissioned with the development of the AquaScan testbed, and the research, development, and evaluation of proof-of-concept analog receivers with an optimal Signal-to-Noise Ratio (SNR) and Coherent Diffuse Ratio (CDR) subject to cost constraints. Its aim is to demonstrate that a qualified 3D sidescan sonar system for the masses is attainable using analog techniques for the receiver module, as opposed to field-programmable gate arrays and digital signal processors which can be more expensive, albeit better performing. The implementation of such a system is the ultimate goal of this project, and will complement software already developed by Dr. Bird of the Underwater Research Lab (URL) at Simon Fraser University (SFU) for 3D sidescan sonar technology.

1.1 SCOPE

This document addresses the implementation of its essential functionalities previously described in the document entitled “Functional Specifications for a Qualified Sonar System for the Masses” in [1]; that is, only the implementation of specifications marked by a level of important of ‘*E*’ will be proposed and justified. In addition to this, relevant theoretical background will be provided, and device limitations will be discussed. In the case of the candidate analog receivers, where several potential avenues of approach exist, the merits and drawbacks of each will be compared and contrasted in addition to the aforementioned items.

1.2 INTENDED AUDIENCE

The design specifications are intended primarily for the members of AquaSense Systems and its client, Dr. Bird of the URL at SFU. AquaSense Systems will use this document as a reminder of the design details determined through previous consensus, and will expect each of its members to adhere to it unless further changes are unanimously approved. By means of providing the supporting justifications through theoretical and practical considerations, AquaSense Systems also aims to persuade its client of the design choices that have been made.

2. SYSTEM OVERVIEW

The complete AquaScan sonar system in terms of its major components is detailed below in Figure 1:

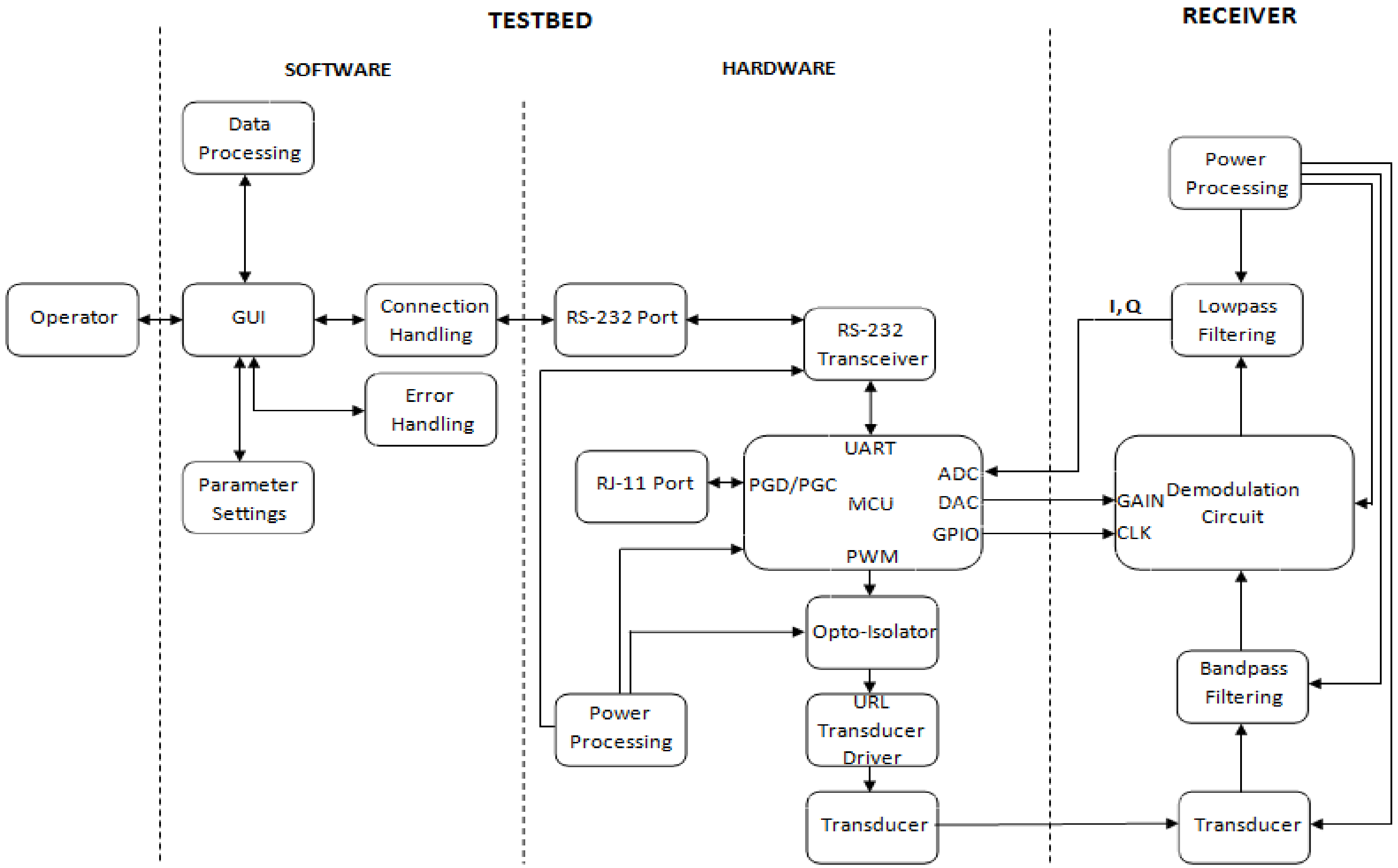


Figure 1: Block Diagram of Complete AquaScan System

The AquaScan sonar system consists of a testbed for the evaluation of candidate analog receivers, and the receivers themselves; we now proceed to summarize the various components mentioned in Figure 1.

The software of the testbed is built around a Graphical User Interface (GUI) through which the operator may vary the parameters of the carrier signal (including frequency, pulse length in carrier cycles, maximum detection distance), and the receiver gain (to boost small incoming signals); process received data; manage connection settings; and receive error notification. The hardware of the testbed consists of an Recommended Standard 232 (RS-232) port and transceiver, through which the parameters from the GUI are relayed to the MicroController Unit (MCU), and data obtained from the MCU are relayed to the GUI. The MCU implements the parameter settings in driving the URL transducer driver circuit, which is connected via an opto-isolator to isolate it from the noise of the testbed. A Registered Jack 11 (RJ-11) port interfaces with the MCU to support future firmware upgrades, and power processing circuitry conditions the input supplies before they are fed to the rest of the entire system. At the receiver, an input transducer directs the acoustic signal to the front-end BandPass Filter (BPF), which isolates the desired signal before passing it on to the demodulation circuit for further processing. The desired In-phase (I) and Quadrature (Q) components of the received signal are recovered at the output of the LowPass Filter (LPF) and then appropriated at the MCU through its internal Analog-to-Digital Convertor (ADC). This data is then plotted in real-time for the operator, who may also choose to export the sampled data for further processing.

The design choices corresponding to the AquaScan components and their justifications are detailed in the sections immediately following; items not addressed in Figure 1 will be dealt with thoroughly there.

3. TESTBED HARDWARE

3.1 ARCHITECTURE

Figure 2 presents the design of the complete AquaScan testbed:

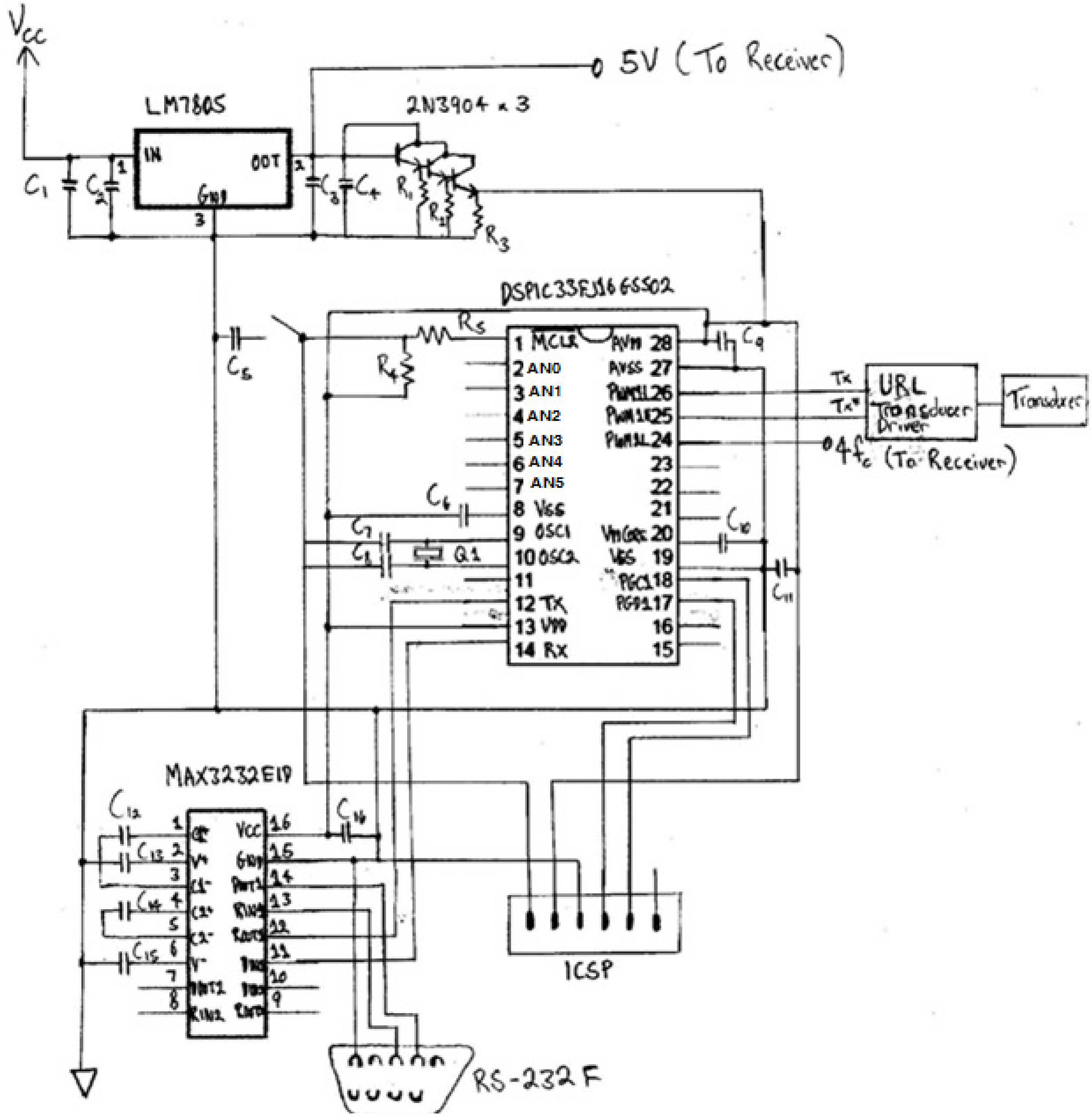


Figure 2: Design of Complete AquaScan Testbed

The design and function of these components are discussed in the subsections which follow.

3.2 MICROCONTROLLER UNIT

The function of the MCU is to drive the URL transducer driver circuit at a carrier frequency f_c and to provide the stable $4f_c$ clock to the receiver (these are both done with Pulse-Width Modulation (PWM), which is detailed in 4.2.3), vary the carrier as per the user interface (detailed in section 4.1.7), perform high frequency analog-to-digital (A/D) conversion on demodulated signals (detailed in 4.2.4) at a resolution sufficient for sonar applications, provide precise control of the variable receiver gain (to be detailed in 5.2.6), and to transmit large volumes of data back to the user interface (detailed in 3.5.2, 4.2.4, and 5.2.5).

The dsPIC33FJ16GS502 is one of the most powerful 16-Bit MCUs that Microchip Technology Inc. offers for control applications. The high speed ADC with its 10-bit resolution (high relative to that of other MCUs) makes it ideal for our sonar application. This is complemented by the high speed Universal Asynchronous Receiver/Transmitter (UART) connection to facilitate the need for large data transfers, and the 40 Million Instructions Per Second (MIPS) which the MCU can achieve. Other relevant features are a 10-bit resolution digital-to-analog convertor (DAC) to be used to control the gain of the Variable Gain Amplifier (VGA) block detailed in Figure 1, multi-channel high frequency PWM for providing the control signals to the receiver module, and the ability to drive each PWM channel with its own source to provide more flexibility in generating control signals. Alternative MCUs were considered - the dsPIC30F4011 was used in the preliminary stages of the project; however, inflexibilities in generating control signals due to PWM modules that were primitive relative to that of the chosen MCU, along with an slower overall instruction rate eventually proved it to be insufficient for our needs. 32-bit MCUs were also considered, but were ultimately discounted because of their cost and their packaging, which made them impossible to test on a typical breadboard and a limited budget. For more information of the dsPIC33FJ16GS502, refer here [2].

3.3 TRANSDUCER DRIVER CIRCUIT

In order to drive the URL water transducers, a transducer driver circuit which includes opto-isolators has been provided by the URL and does not need to be designed; the opto-isolators in Figure 1 transfer signals from the MCU to the URL circuit while keeping them electrically isolated. The AquaScan needs only to drive this circuit with the correct waveforms, namely T_x and T_x^*; these are complementary waveforms having a deadband and a 50% duty cycle, which is according to specifications given by [3]. The reason for the inclusion of the deadband in seen in the URL transducer driver schematic in section 10.1 of the appendix. At the output of the circuit are two pairs of power MOSFETs, and the MOSFETs in each pair must be turned on in complementary fashion otherwise a short would occur between X3-1 and ground destroying the circuit. Since the switching time of these MOSFETs is at most 42 ns [4], a deadband of 50 ns has been selected; this is the shortest deadband obtainable with the 80 MHz external clock which drives the testbed, detailed in 4.2.5.

3.4 TRANSDUCER

The testbed must be able to operate correctly at carrier frequencies of between $f_c \epsilon [23, 600]\mathrm{kHz}$. Corresponding to these frequencies are several types of air and water transducers that have been provided by the URL for testing, and do not need to be designed. The AquaScan is responsible for being able to drive these transducers, and it does so through the PWM module detailed in section 4.2.3, and the transmitter driver circuit detailed in 3.3.

3.5 CONNECTION INTERFACES

3.5.1 Microcontroller Unit to Computer

The dsPIC33FJ16GS502 includes a high speed UART connection to support the RS-232 communication standard. Unfortunately, RS-232 has been phased out of most computers available today in preference of more modern standards, among which is the Universal Serial Bus (USB) standard used to communicate with the AquaScan user interface. To address this issue the AquaScan includes a RS-232 to USB adaptor to facilitate such communication; the availability of such an adaptor was a factor in choosing USB over other commonly used standards such as IEEE 1394.

Another issue in communicating between the MCU and the computer was that RS-232 runs on 12V, whereas the MCU is only capable of driving 3.6 V into its output pins (or 5 V with an open drain configuration including a pull-up resistor). Fortunately, RS-232 transceivers are commonly available in different speed grades; these ICs are capable of translating a serial binary bit stream from commonly used logic voltage levels to those used by the RS-232 standard, and vice versa. In addition to this, as will be detailed later in 4.2.4, the AquaScan requires at a minimum 1.2 Mbps for its transfer rate. These requirements lead to the selection of the MAX3232EID transceiver for inclusion in the AquaScan testbed; its pin configuration, the additional external components required, and further information can be found in [5].

3.5.2 Microcontroller Unit to Debugger

An RJ-11 cable is used by the Microchip ICD2 debugger to interface between the MCU and the circuit; the correspond to the pins marked “PGD/PGC” in Figure 1. Thus to program the MCU during development and provide possible firmware updates, an RJ-11 port must be integrated into the testbed, as shown in Figure 2. The pin configurations of this port are according to the RJ-11 specification, detailed in [6].

3.6. MISCELLANEOUS

3.6.1 Power Supply Regulation

So that the testbed may be used with a variety of power supplies, linear regulators have been integrated into a power processing module, which also includes several transistors, to couple power into the rest of the sonar system. The power processing module is shown in Figure 3 below:

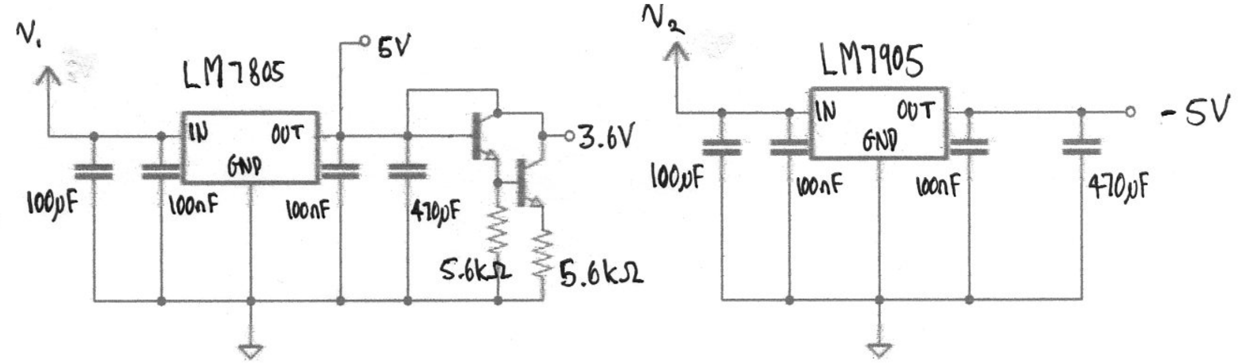


Figure 3: Power Processing Module

The left circuit consists of an LM7805 5V linear regulator which provides up to 1A of current. At its output is a cascade of 2N3904 NPN BJTs to provide a 3.6V output. The 3.6V is used by the MCU, while the rest of the entire circuitry including the receiver runs on 5V. The resistances were chosen arbitrarily save for the fact that loading effects should be avoided, and the transistor should have quiescent current enabling it to function correctly. The large electrolytic capacitors in the µF range are present to prevent sudden changes in output voltage, and provide board level capacitive decoupling. Finally, the voltages v_1 and $v_2$are typically at least 2V larger in magnitude than the output of the regulators to turn them on.

As an added advantage, the power processing module allows the AquaScan to be run on two common 9V alkaline batteries; the lifetime of such a setup has been tested to be around an hour.

3.6.2 Capacitive Decoupling

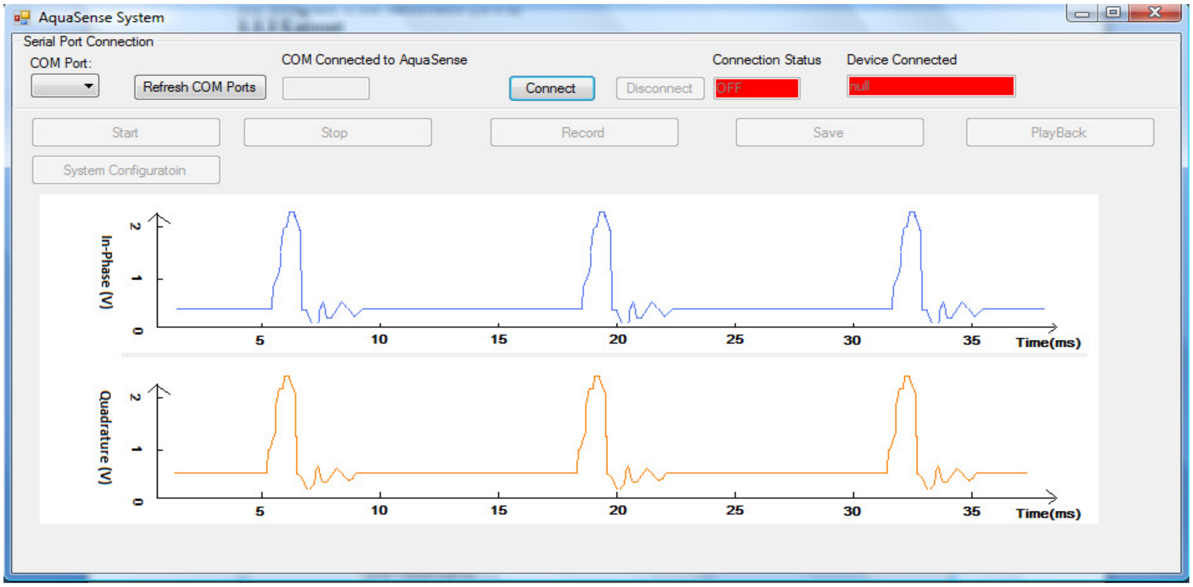
Capacitive decoupling should be provided at both the board and chip levels in the sonar system. The board level decoupling was discussed in 3.6.1, the chip level decoupling is detailed in 5.2.4 and 5.2.6.

4. TESTBED SOFTWARE

4.1 USER INTERFACE

4.1.1 Layout and Operation Overview

The arrangement of the user interface is shown in Figures 4 and 5; the former entitled “AquaSense System” is the main window, including the connection management and graphical output sections, while the latter is the “System Configuration” window which contains settings affecting the carrier signal.



Connection Status: OFF

Device Connected: null

Figure 4: GUI Main Window

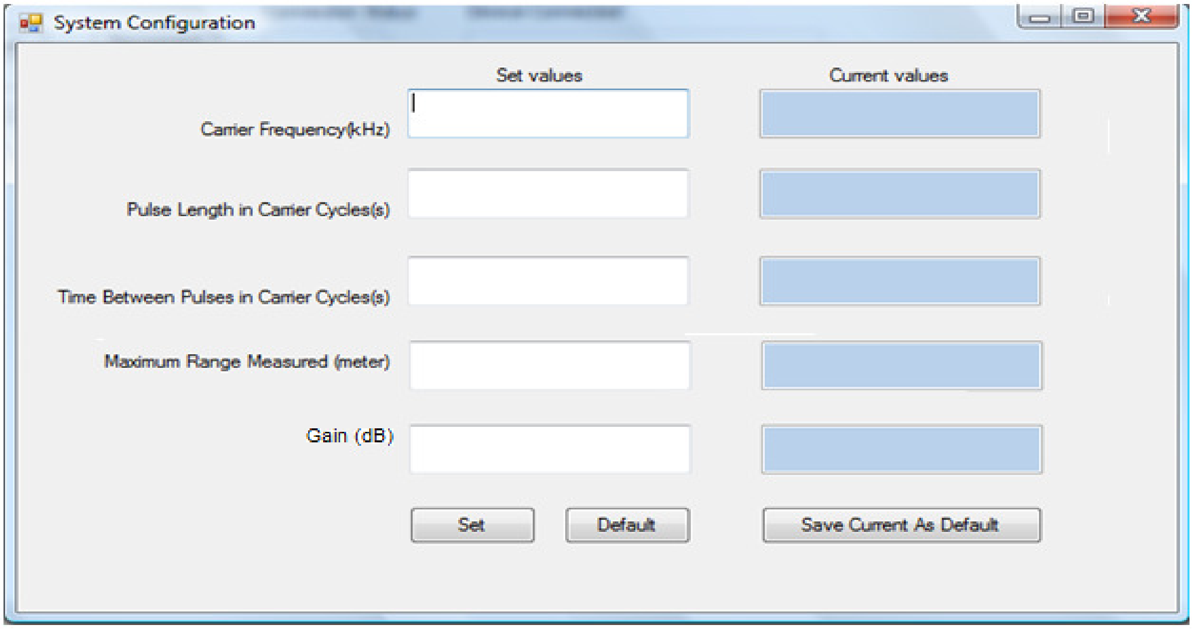


Figure 5: Carrier Settings

The "Serial Port Connection" section enables the user to choose an available Component Object Model (COM) port, connect or disconnect to the MCU, and be informed of the status of a connection. Below this is the graphical output section, which plots the waveforms of the I and Q components in quasi real-time using the data received from the MCU via the serial port; here the user also has the option to record a waveform, save it, as well as to play it back. When the "Start" button is clicked, a dialog appears which gives users the option to choose to use the previous carrier settings or to create new ones. To create new carrier settings, users must click the "system configuration" button, and the parameters window appears which shows all the existing system settings and allows users to configure the settings, such as carrier frequency, pulse length in carrier cycles, time between pulses in carrier cycles, maximum detection distance, and receiver gain. When the "Set" button is pressed, the GUI sends the parameters values to MCU, configuring the system.

4.1.2 Algorithm

The following discussion aims to highlight both our algorithm and the design choices which have been made.

The GUI is written in the C# programming language, which is a higher level language than C++ or C, which translates into more rapid development at the cost of performance [7]; however, since the AquaScan application is relatively simple, performance issues are not a concern.

In addition to the main thread which handles the user commands detailed above, the operation of the GUI can be broken into three threads with each handling a given responsibility: receiving data, writing data or graphical outputs. Due to the high speed at which data is received and the nature of the features in the GUI, an extremely large buffer must exist to store the data temporarily before it is written into a file and shown on the graph concurrently, otherwise data may be lost. The memory mapping of files is the best technique to implement this large buffer because it is particularly useful in manipulating in-memory data for large data structures. To take advantage of this, the recently released Microsoft .Net Framework 4.0 is used which introduces memory mapped files.

The advantage of memory mapped files lies in their ability to be shared concurrently between different processes, threads or programs; furthermore, each process, thread or program can perform multiple operations on the memory mapped file simultaneously. As an illustration of this, in the GUI program data is captured by the receiving thread, written by it into the buffer implemented by the memory mapped file, and then used by both the graphical output thread and the writing thread to draw the graph and write the data into a permanent data export file at the same time. Instead of using large strings or typical files as buffers, the GUI threads can concurrently access the memory mapped files directly to improve application performance. In Figure 6, which is a snapshot in time, both thread 1 and thread 2 are accessing the memory mapped file, and both threads are reading and writing from multiple locations at the same time; it is even possible for both threads to overlap each other in the locations that are accessed.

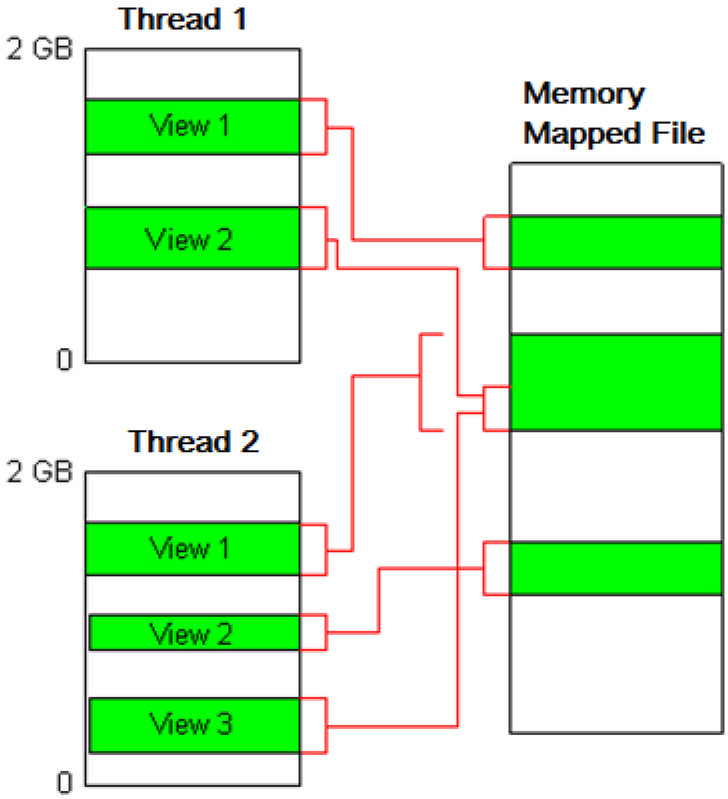


Figure 6: Memory-Mapped Files allow Multiple Views from Multiple Threads Simultaneously

Figure 7 shows the three threads used in the GUI program. The receiving thread remains in an infinite while loop to keep track of all the data coming from serial port. All transmitted data from the MCU will be first stored into the serial data buffer, and then written into the buffer implemented with the memory mapped file immediately after. As detailed before, this prevents data from being lost, and the culprit behind this is a buffer overflow in the small serial port buffer. Since the transmission speed and ADC sampling rates are high, this small buffer will overflow almost immediately once the program starts. The writing thread retrieves all the data from the large buffer and then immediately writes them to the export data file. The I and Q data contained in this file will be permanently saved after user clicks "Stop" button; otherwise, it will be discarded upon program exit. The drawing thread reads the I and Q data from the memory mapped file and then plots their graphs on the main window concurrently with the operation of the writing thread.

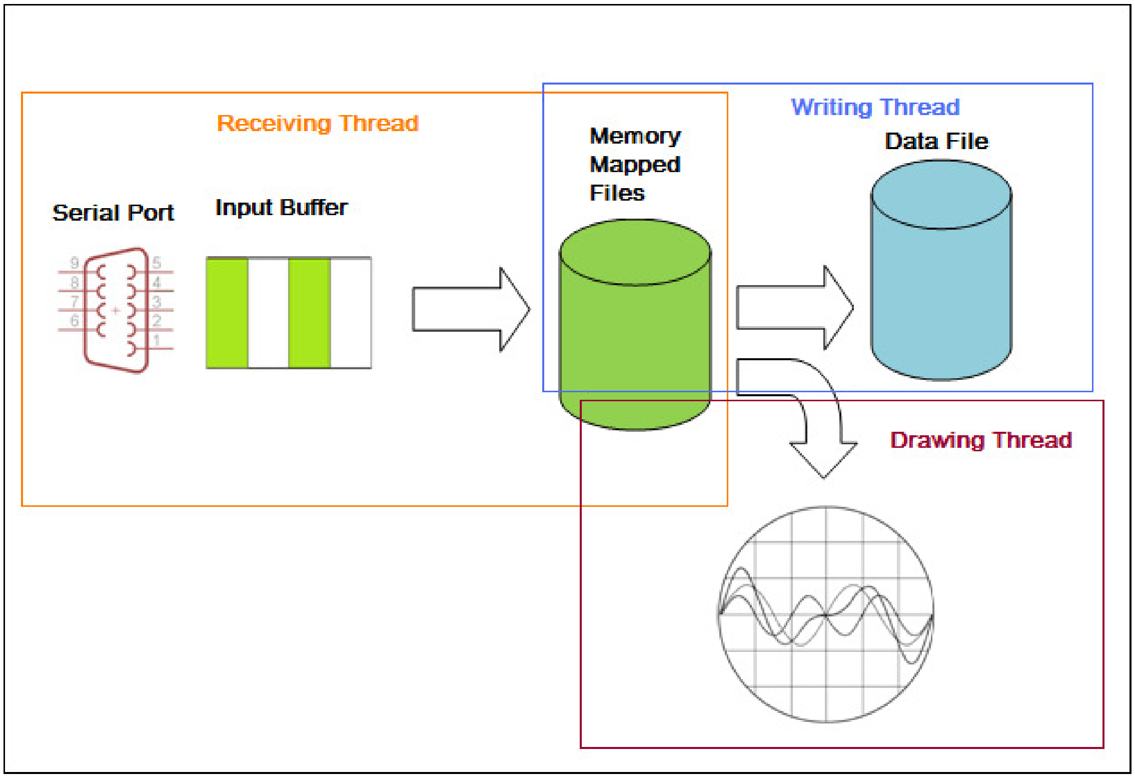


Figure 7: Threads used in the GUI program

4.1.3 Communications

The GUI and MCU communicate in full-duplex manner; that is, both can send and receive data at the same time. The serial port uses a baud rate of 1.25 Mbps, and transmits eight data bits per packet; most of this data is due to the high speed ADC, which by itself nearly depletes this allocated bandwidth, as detailed later in 4.2.4. Furthermore, the serial read buffer must be as large as possible to prevent overflow conditions, which lead to the loss of data almost immediately after the program starts; the size needed depends on the baud rate. In the case of our transmission speed, a serial buffer of size 50 Mbits was needed.

4.1.4 Data Transfer

Both the GUI and MCU follow the same conventions for their data transfer to facilitate ease of communication. Each piece of data contains a header and its associated contents. By using the data header, both the GUI and MCU are able to identify which type of data is being transferred. For example, a "?" might identify the data in a given piece as the carrier frequency. When the MCU receives the "?" in the header, it extracts the data and varies the carrier frequency accordingly.

4.1.5 Connections

The GUI provides both a "Connect" and "Disconnect" button. Once "Connect" is clicked, the connection between the GUI and serial port will be established on the selected COM port.

Immediately after that, the GUI will send out handshake string to the MCU. Upon receiving the string, the MCU will reply with another string to complete the handshake. If the connection completes successfully, the “Device Name” field will show that the AquaScan device is connected to GUI; otherwise, an error is indicated to the user. Handshake protocols are common, and their reasons are well-known among those skilled in the art, so they will not be elaborated here. A “Disconnect” terminates the connection between the GUI and the serial port, and release all the resources dedicated to the serial port.

4.1.6 Error Detection and Handling

The GUI is able to detect user inputs error and system errors. For users input errors, such as when users provide characters instead of numbers for the system settings, a warning window appears. For system errors, the GUI uses “try and catch” coding to implement the error detection. For example, when user tries to connect a busy serial port, “try and catch” coding will show an error message to the user. There are many other kinds of system errors, such as killing a thread before performing a required operation. The GUI considers all the illegal operation cases and brings up the corresponding error messages.

4.1.7 Parameter Selection

The GUI has four parameter settings to enable users to configure the transmitted carrier, as required by our client. These are the carrier frequency, pulse length per carrier cycle, carrier cycles between pulses, and receiver gain; another setting, the maximum detection distance, is dependent on the carrier cycles between pulses. For the sake of convenience, the application includes a default configuration file which stores all the default system settings, and allows the user to define their own default values via the GUI. The user can then recall the default values at any time by pressing the “Default” button. In addition to this, the GUI will automatically save and recall parameters used the last time the program was run. When users start the program next time, the GUI displays a message window for user to choose between loading the default or previous system parameters.

4.1.8 Graphical Output

The GUI uses .Net Framework commands to implement the output I and Q plots, since the same is already used for the memory mapped file buffers which have already been detailed, and open source packages are readily available to provide some complex functionalities; the use of open source packages for this specific purpose has already been approved by our client. The horizontal axis of the graphs represents time, and the vertical axis of the graphs represents the amplitudes of both the I and Q data. Color coding and textual annotations are also used to distinguish I and Q from one another. Once the GUI gets the I and Q data from the MCU, it will plot their graphs in the main window of the GUI in quasi real-time; a delay must be included due to the high ADC sampling rate and the high data rate of the serial port transmission.

4.1.9 Data Output

There are two types of output data: one is the current system setting parameters (carrier frequency, pulse length per carrier cycle, time between pulses, and receiver gain), which are retrieved from the MCU and displayed in the "System Configuration" window; the other is the recovered I and Q data which will be output to a file. The former is to provide the status of transmission to the user, and the latter is so that the user can then pass this data on to other programs for further analysis. The former has three bytes per data piece, with the first being the header, and the remainder the contents; the latter has two bytes per data piece, and this is detailed in the section on the ADC in 4.2.4.

4.2 FIRMWARE

4.2.1 Flowchart

Figure 8 presents a flowchart detailing the operation of the MCU firmware:

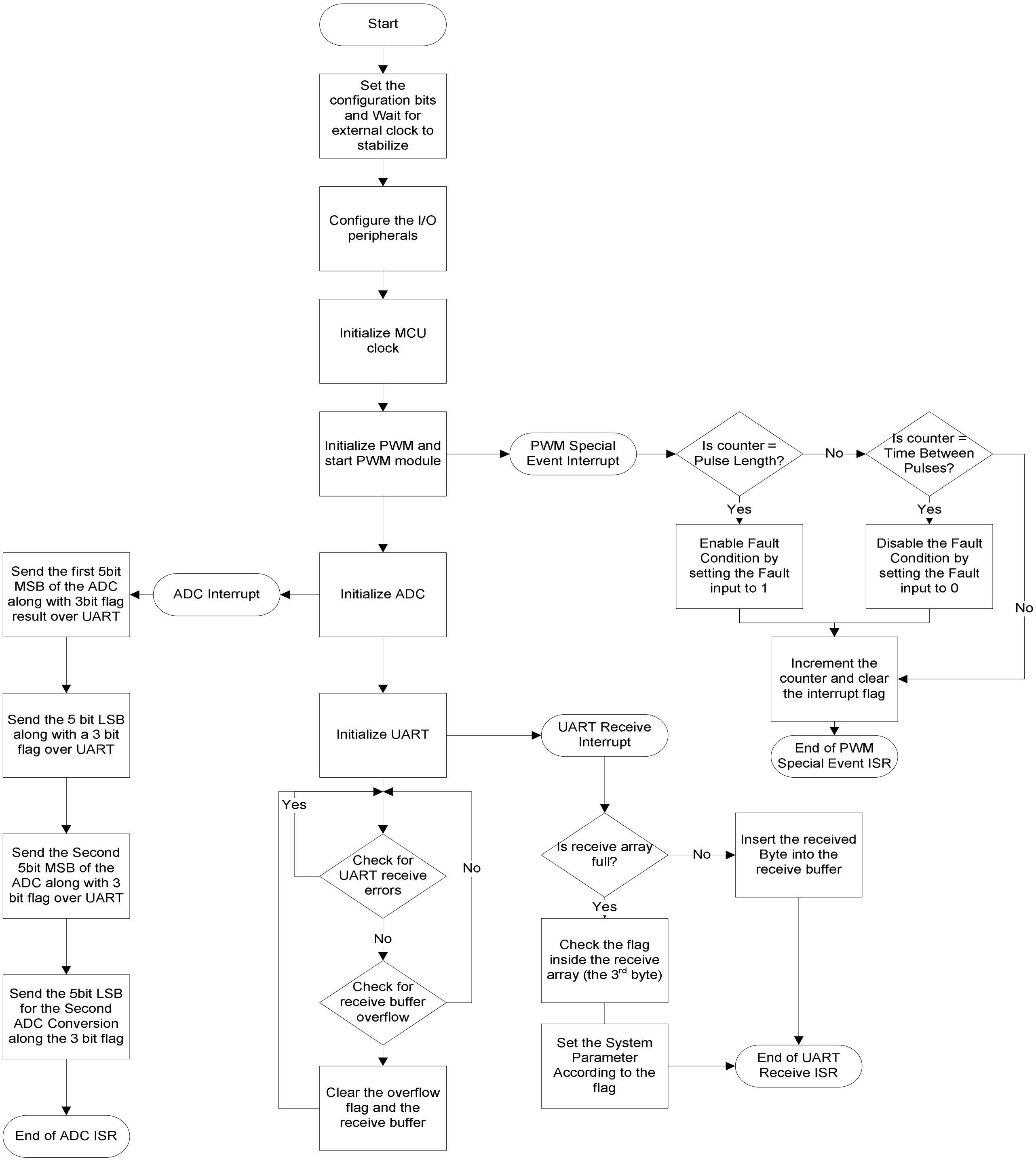

Start
Set the configuration bits and Wait for external clock to stabilize
Configure the I/O peripherals
Initialize MCU clock
Initialize PWM and start PWM module
PWM Special Event Interrupt
Is counter = Pulse Length?
No
Is counter = Time Between Pulses?
No
Yes
Enable Fault Condition by setting the Fault input to 1
Yes
Disable the Fault Condition by setting the Fault input to 0
Increment the counter and clear the interrupt flag
End of PWM Special Event ISR
Initialize ADC
ADC Interrupt
Send the first 5bit MSB of the ADC along with 3bit flag result over UART
Send the 5 bit LSB along with a 3 bit flag over UART
Send the Second 5bit MSB of the ADC along with 3 bit flag over UART
Send the 5bit LSB for the Second ADC Conversion along the 3 bit flag
End of ADC ISR
Initialize UART
Yes
Check for UART receive errors
No
No
Check for receive buffer overflow
Clear the overflow flag and the receive buffer
UART Receive Interrupt
Is receive array full?
No
Insert the received Byte into the receive buffer
Yes
Check the flag inside the receive array (the 3rd byte)
Set the System Parameter According to the flag
End of UART Receive ISR

Figure 8: Flowchart for MCU Firmware

As seen from Figure 8, the firmware can be divided into three main functions: that of the UART, the PWM, and the ADC. These modules and their design choices are thoroughly detailed in the following subsections.

4.2.2 Communication

The communication on the side of the MCU firmware is necessary to receive parameters from the user interface, and transfer ADC data back to it. To facilitate this, the UART operates in high speed mode due to the high volume of data which needs to be transferred back to the user interface, mainly because of the high speed ADC as detailed in subsection 4.2.4; to handle this, the selected baud rate is 1.25Mbps no parity, 8 bit and one stop bit. From the user interface, the user can update the system parameters, and then inside the receive Interrupt Service Routine (ISR) of the MCU the received data is processed immediately, and the system parameters get updated as soon as new valid values are identified; this is to prevent errors due to invalid parameters. The details relevant to the transfer of ADC data back to the user interface are dealt with in subsection 4.2.4, which deals with the ADC.

4.2.3 Pulse-Width Modulation

PWM1L and PWM1H are the two signals that drive the URL transducer driver circuit at f_c; these are shown at pin 25 and 26 in Figure 2. The two PWM signals belong to the same channel and are complementary with both having a dead band of 40 ns and a 50% duty cycle; the reasons for both parameter values are detailed in section 3.3.

In order to create a sonar ping, the programmer has to control the number of rising edges of these two pulses for its duration and the time between the pings; these two pulses are always active so that the time between pulses can be counted in terms of rising edges, but are only output when the ping is active. Special event interrupts fire at the end of each ping, and inside the ISR a counter handles the aforementioned counting to determine when to start the next ping. By enabling a fault input and toggling its logical value inside the special event interrupt we are able to selectively pass the PWM signals to create the sonar pings.

Another PWM channel provides the stable $4f_c$ clock for the demodulator; this corresponds to the pins marked “GPIO” and “CLK” in Figure 1. Because the frequency of this channel is faster than that of the complementary PWM pair, the two channels cannot be in-phase with respect to each other, due to inherent limitations in the MCU [8]. To obtain an in-phase demodulator clock, a third PWM channel is thus utilized to synchronize the first two channels together. By setting the appropriate bits for this third PWM channel in the PWM control register, setting a current limit input, and directing the third PWM channel to the current limit input for the first channel, the third PWM channel has the same frequency as the complementary pair PWM1H and PWM1L, and the two channels are perfectly in phase. The only difference between this channel and the first PWM channel is that this channel never turns off therefore it can provide a synchronizing signal for the high speed clock that is produced by the second channel.

The fourth PWM channel was used to trigger an interrupt that controls the A/D sampling. By controlling the frequency of this PWM channel we are able to control the A/D sampling rate; the ADC is detailed in the following subsection.

Another pertinent point is that in a six-channel sonar system, all these PWM control signals can be shared among the channels, so that the number of MCUs required does not need to increase.

4.2.4 Analog-to-Digital Conversion

The dsPIC33FJ16GS502 provides three 10-bit A/D conversion pairs; these are found at pins 2 through 7 in Figure 2. Only the first pair AN0 and AN1 is used to sample I and Q since the AquaScan possesses only a single sonar channel; the remaining pairs could be utilized in six-channel sonar systems, which means we would require at least 2 MCUs. Because we used the 8-bit UART mode detailed earlier, we have to split the result of the 10-bit A/D conversion into two bytes Inside the A/D ISR. The result of the 10-bit conversion is divided to two pairs of 5 bit data, with the 3 most significant bits in each byte being the identifier for the contents of the byte and the rest of the bits being the A/D results. Inside the A/D ISR the result of the conversion is also sent to the GUI. The sampling frequency of the analog to digital converter is controlled by the frequency of PWM4H and PWM4L. As will be seen in section 5.2.5, the maximum sampling rate that the AquaScan must handle is 60 kHz per signal. This translates into 600 kbits per second per signal, or 1.2 Mbits per second per channel. As seen in subsection 4.2.2, the existing RS-232 connection is barely able to handle this requirement, and a six-channel system will need a definite upgrade to a faster interface, such as USB.

4.2.5 MCU Clock Initialization

One of the distinguishing features of the dsPIC33FJ16GS502 is the high speed PWM and ADC. The use of an external clock source ensures an accurate PWM pulse generation and ADC sampling interval, which are essential to our application. However, the external clock requires some time to stabilize, and the dsPIC33FJ16GS502 must utilize its internal clock upon power-up. Thus as soon as the MCU is powered on, the MCU will run over the internal clock for a short period of time (several μs), and then the MCU will initiate the clock switch. The clock for these two features can be separated from the rest of the MCU as needed by the application. For the AquaScan testbed, an external clock was used for the ADC and PWM, which differed from that of the MCU since the optimal clock rates were not the same; the former is 120 MHz, while the latter is 80 MHz.

5. RECEIVERS

5.1 BASIC DEMODULATION THEORY

A simplified block diagram of a superhet receiver system is shown below, and is taken from [9]; this forms the basic structure for many modern receiver systems.

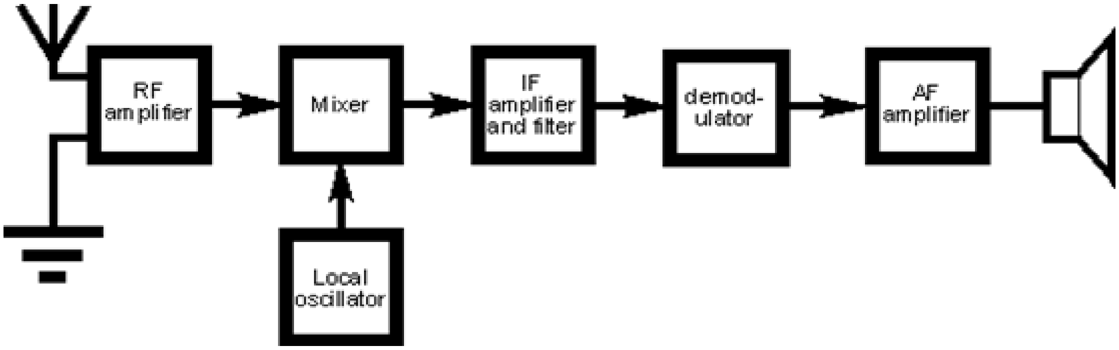


Figure 9: Superhet Receiver System

The superhet receiver system is used to demodulate incoming signals of varying carrier frequency f_c. The Radio Frequency (RF) amplifier block includes both an amplifier and a wideband BPF. The RF BPF is wideband because it needs to be able to capture the entire bandwidth of all desired signals over the range of all possible f_c; the RF amplifier is used to boost the SNR at the front end of the receiver, which makes the predominant noise contribution, as we will later prove. The mixer and the local oscillator bring the spectrum of the received and filtered signal down to the *fixed* Intermediate Frequency (IF) filter frequency, in a process called heterodyning (the multiplication of signals to generate signals at new frequencies). Here at the IF stage, the signal is passed through a BPF and amplified again. This filter can be narrower in bandwidth than the RF filter because it is meant for signals with a fixed f_c, as opposed to the latter which is supposed to capture signals of varying f_c; the IF filter typically determines the noise bandwidth of the receiver because it has the narrowest bandwidth, and a narrower bandwidth limits the noise power [10]. The signal is then demodulated and brought back to baseband, and in the final stage it passed through a LPF to isolate the desired signal. For a more in-depth discussion of superhet receivers, please refer to [9].

5.2 TAYLOE DETECTOR

The Tayloe detector exploits the following observation regarding bandpass signals. Any bandpass signal may be expressed in terms of its I and Q components as:

$$s(t) = I(t)cos(2\pi f_c t) + Q(t)\sin(2\pi f_c t), \tag{1}$$

where $I(t)$is the I component of the bandpass signal, $Q(t)$ its Q component, and f_c is the carrier frequency. If our received signal is bandpass, we may sample the signal at the phases of 0°, 90°, 180°, and 270°, and get $I(t)$, $Q(t)$, $-I(t)$, and $-Q(t)$ respectively. Sampling at precisely these phases may be difficult in hardware however, so we take advantage of another fact that so long as the phase separation between samples is 90°, we will still get $I(t)$, $Q(t)$, $-I(t)$, and $-Q(t)$ albeit scaled by a constant. So, in terms of our Tayloe detector, this means we just need to sample at $4f_c$, which is not as rigid a constraint.

A simplified block diagram for the Tayloe detector is shown in Figure 10. Note that only components directly contributing to the function of the detector are shown; thus buffers, regulators, and the like are not detailed here.

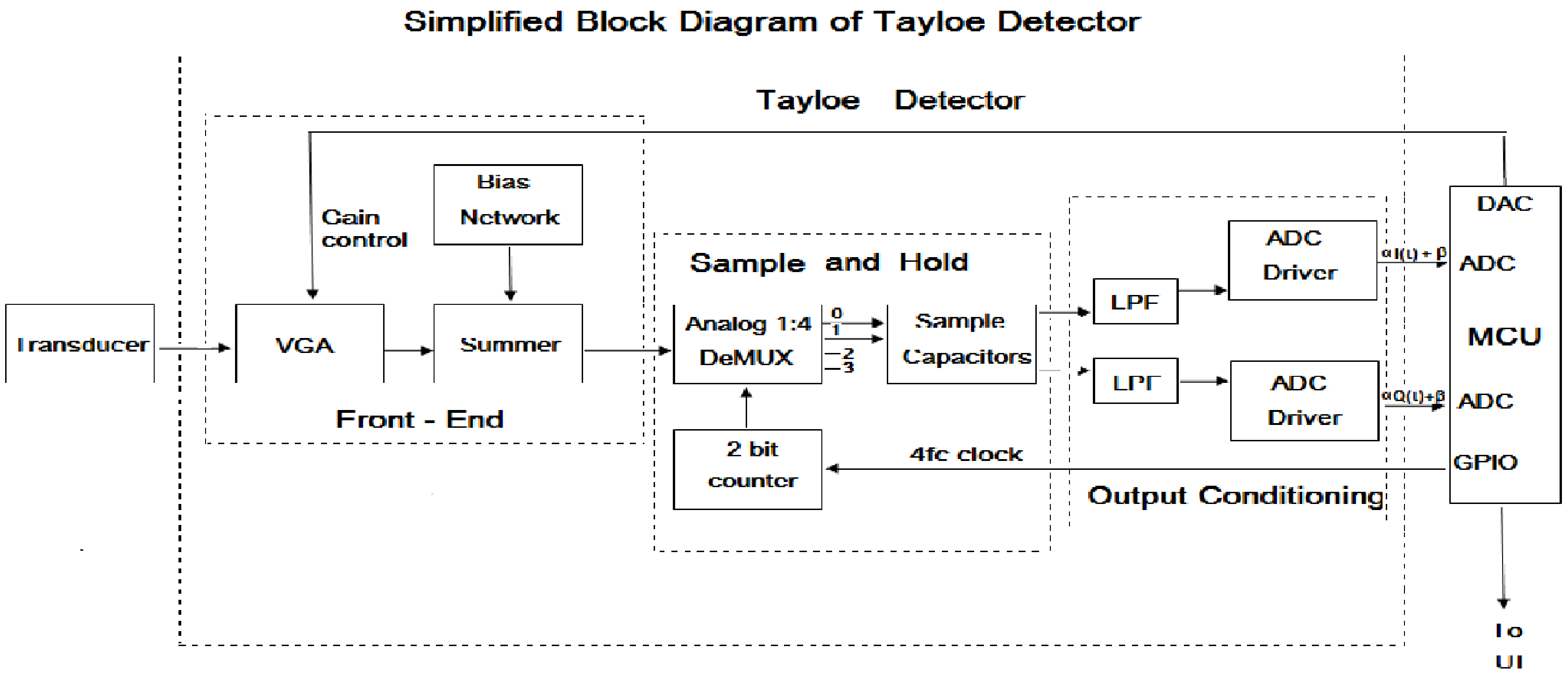


Figure 10: Simplified Block Diagram for Tayloe Detector

This block diagram will be thoroughly discussed in the following sections, so we shall not comment on its specifics here.

The first thing to note is in contrasting our Tayloe detector to the superhet receiver scheme is that we are using a transducer instead of an antenna; however, this is merely a different means of receiving the transmitted signal, and the general idea is still valid – the Tayloe detector works just as well with an antenna. The second thing to note is that we appear to have no RF amplifier and filter stage. According to [11], this is because the input resistance of the summer sub-stage (assuming that other resistances in the front-end stage are negligible in comparison) and the sampling capacitors form a BPF already centered at the detection frequency. This is advantageous, because this amounts to an inherently variable frequency filter, whose bandwidth we can be more selective with than with the superhet RF filter which needs to encompass all possible received signal spectra. The mixer and local oscillator of the superhet receiver are analogous to the following stage of the Tayloe detector which switches the input to four possible outputs using a quadrature track and hold sampling detector [12] at a rate of $4f_c$ (and f_c can change, just as with the superhet), so that each output is connected to the input at a rate of f_c. This can be implemented with several techniques, including 1:4 demultipexing and a 4-way commutating switch. This same stage with the sampling capacitors forms a sample and hold circuit, whose outputs are the sampled versions of our four desired demodulated quantities $I(t)$, $Q(t)$, $-I(t)$, and $-Q(t)$; so it is analogous to more than just the mixer and local oscillator – the analogy also includes the IF and demodulation stages of the superhet receiver. Finally, just as in the superhet scheme, there is a LPF at the output, yielding the smoothed versions of our desired quantities. Note that the Tayloe detector could also be used as just the IF and demodulation stage of a superhet receiver scheme; this would amount to a fixed switching frequency. A more detailed discussion can be found in [11], [12], and [13].

In addition to the inherently variable, more selective BPF mentioned earlier, there are several other advantages the Tayloe detector possesses; these are again found in [11],[12],and [13] – because this is primarily a design document, please refer there for more detailed information:

1. Less than 1dB of conversion loss (the power loss from the input to the output), while typical designs have at least 7 dB
2. At least a 6 dB improvement in noise performance over typical designs; because of this increase in noise performance, pre-amplification may become unnecessary, and removing this stage improves large signal performance
3. Able to handle a higher dynamic power range of input signal than typical designs
4. A high 3rd order intercept (a measure of linearity)

5. A compact, simple, inexpensive, and high-performing design compared to simple direct conversion and image reject receivers
6. Does not suffer from the “image problem” which plagues the superhet receiver scheme
7. A maximum useful frequency which readily extends to 10 GHz

5.2.1 Front-End Stage

We now provide general considerations for each of the functional blocks of the Tayloe detector leading up to the high-level design in Figure 10. These considerations will dictate the design of the Tayloe detector, detailed in sections 5.2.6 through 5.2.8. The front-end block performs a variable amplification of the received signal, and then sums it with a DC offset.

According to [10] and [14], the front-end stage must be the cleanest in a receiver because it makes the predominant noise contribution; this argument can also be made for the sub-stages in the front-end stage itself. This is demonstrated by Frii’s formula for noise shown below:

$$F_{cascade} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \frac{F_4 - 1}{G_1 G_2 G_3} + \cdots \quad (2)$$

where *F* is the noise figure, which is the ratio of the SNR at the input of a stage to that at its output, *G* is the gain of the stage, and the front-end stage corresponds to F_1 and G_1. The formula remains valid so long as the stages are correctly matched, the noise bandwidth is the same throughout, the temperature is constant, and there are no other subtleties involved [15]. The noise floor of a receiver is important because, according to [10], it determines its sensitivity to low level signals and its capability of detecting and demodulating those signals. Also we see from the equation that in order to decrease the relative noise contributions of the following stages, and thus ensuring its noise contribution is predominant, the initial stage should have a moderate gain – this is the reason we amplify the received signal in our front-end sub-stage of the front-end stage; however, the gain should not be too high because this could limit large signal performance due to saturation. The amplification is variable because the strength of the received signal varies with detection distance. Thus, a VGA should be used for the front end of the entire system, with a high quality DAC to control the variable gain of the VGA to the finest resolution possible. The VGA will have lowest possible input noise density, and will in the production model have its gain controlled by the MCU depending on the desired maximum detection distance input by the user. The VGA amplifier should employ the maximum gain allowed without saturating itself or the following sub-stages; this is due to the noise considerations detailed above. The input noise density is determined by the short-circuit voltage noise density and the open-circuit current noise density, and are found on most amplifier datasheets; a thorough discussion of the quantities is found here [16].

The offset is required because the vast majority of analog demultiplexers run on unipolar supply voltages. This is problematic because the received signal oscillates about a DC level of 0V, which would cause both saturation of the amplifiers and a loss of information were the signal not allowed to swing between its peak values. A query into the database of Digikey and Newark, which are major electronics distributors, reveals no analog demultiplexers running on bipolar supplies nor any commutating switches, and should be indicative of the majority of all electronics available. The effect of this correction is to cause a DC offset in the final values of I and Q, which can be easily removed within the software. This final offset, the signal amplification, and the scaling factor caused by sampling not precisely at the 0°, 90°, 180°, and 270° phases are the reasons behind the final values of $\alpha I(t) + \beta$ and $\alpha Q(t) + \beta$ going into the MCU instead of their pure counterparts.

5.2.2 Sample and Hold Stage

Ideally, we want to sample at precisely the 0°, 90°, 180°, and 270° phases so we will get $I(t)$, $Q(t)$, $-I(t)$, and $-Q(t)$ respectively. To minimize the phase shift between the clock and the received signal, the clock signal driving the counter must come from the transducer driver circuit which supplies the carrier frequency to the transmitter; this is also necessary so that any change in the carrier frequency there will be reflected here. Also, an accurate clock source is essential to the obtaining of samples spaced 90° apart; a high quality clock crystal must thus be used.

5.2.3 Output Conditioning Stage

The ADC driver should consist of an amplifier with low output referred noise, and a rail-to-rail output with a dynamic range matching that of the ADC input; this is to maximize our acquired data resolution. Output referred noise is the noise which is transferred to the input of the next stage, and is critical here because it determines the dynamic input range of the MCU ADC [17].

5.2.4 General Considerations for the Design of the Tayloe Detector

Before proceeding with the design of the Tayloe detector, several other pertinent points need consideration:

1. Desirable qualities for amplifiers relevant to our application would include: good input and output impedances to prevent loading, which obviates the need for additional buffers; high open loop gain at all frequencies of interest, to enhance the benefits of negative feedback; slew rates high enough to handle all signals within the amplitudes and frequencies of interest; low noise characteristics; and voltage supply ranges large enough so as to not limit signal swings throughout the system.
2. Different types of resistors suffer from different types of noise effects [18], [19]; however, the thermal noise which all resistors suffer from is the predominant noise, and

thus the type of resistors that we use does not matter [3]. Values chosen must not be either too large (around 200 kΩ) or too small (several hundred ohms) to prevent loading and because large resistors are noisy; the resistors must also have a low tolerance because accuracy is desired in low noise applications – this will become more apparent when the design is detailed.

3. Decoupling capacitors need to be included at all IC power supply pins to deal with the associated noise; to deal with both high and low frequencies, ceramic capacitors should be placed in shunt with their electrolytic counterparts [20], [21]
4. The targeted \$300 per-unit cost of the six-channel system must be taken into account, as per [3].
5. The noise at the ADC input is of primary interest and must be kept to a minimum, as per the targeted figures later detailed in their relevant sections; a tutorial on the calculation of op amp noise, which cause the predominant noise contribution according to Frii's equation, is found in [22].
6. Since the AquaScan sonar consists of both digital and analog circuitry, their power supplies must be properly isolated from each other [3], [23]. A suitable technique includes the use of ferrite beads [24]; though the vast majority of the AquaScan circuitry does not involve frequencies high enough to necessitate their use, at least one component does – the MCU runs on internal clocks in the tens of MHz. However, given that the proof-of-concept is to be demonstrated on a breadboard, which is notorious for its high frequency performance, a ferrite bead is not practical unless the decision is made by our client to migrate to a PCB design.

5.2.5 Theory of Operation

Having explained the Tayloe detector in terms of its functional blocks, we now proceed to provide the mathematical basis for the basic operation of this circuit; that is, we seek to detail the time and frequency domain analysis of the received waveform throughout the Tayloe detector where it aids to explain its operation. For the sake of argument, let's say that this is the received, noiseless waveform:

$$y(t) = \sin(2\pi f_m t)\cos(2\pi f_c t)\, rect\big(2f_m(t - 0.75T_m)\big), \tag{3}$$

where f_m is the frequency of the message signal (the envelope), f_c is the carrier frequency, and T_m is the message period. For the sake of simplicity, we shall not include a DC offset, which is equivalent to the received signal bypassing the non-inverting summer; its inclusion would yield similar results. This also requires that we assume an analog demultiplexer with bipolar voltage supplies. The resulting waveform is plotted on the following page:

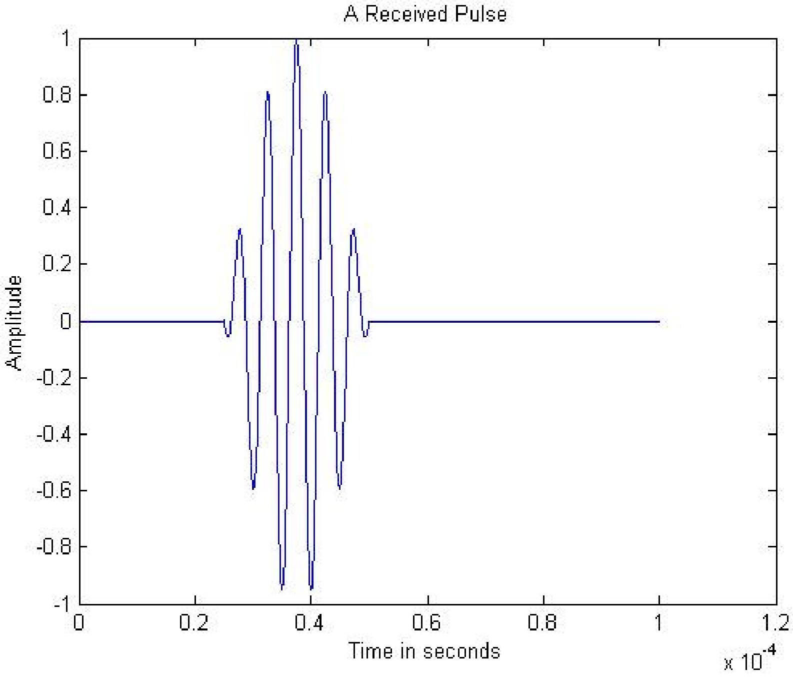


Figure 11: Model of Received Waveform

This is a reasonable model of the actual received waveform in air, which is shown next as an example:

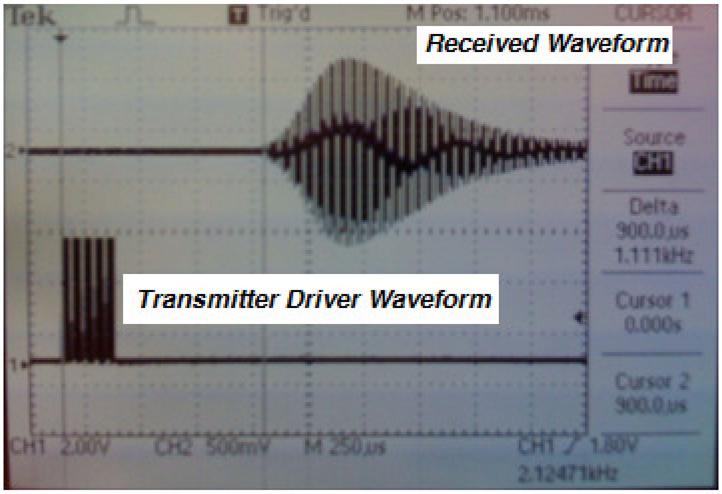


Figure 12: Actual Received Air Waveform

According to [12], the Tayloe detector functions as a quadrature track and hold sampling detector, and thus it adheres to discrete time sampling theory. Therefore we can model the ideal operation of the sampling circuit (implemented by our analog demultiplexer) as a multiplication of a scaled impulse train and *y(t)* as follows [25]:

$$y_\delta(t) = y(t) f_s comb(f_s(t - \tau)), \tag{4}$$

where $y_\delta(t)$ is the sampled form of $y(t)$, the *comb(t)* function is the impulse train with impulses of unit weight, τ translates into a general phase offset, and f_s is the sampling frequency, which for a given demultiplexer output is equal to f_c, because in doing so we will obtain either $I(t)$, $Q(t)$, $-I(t)$, or $-Q(t)$ multiplied by a constant scaling factor which depends on the phase offset.

The resulting waveform is shown qualitatively in Figure 13:

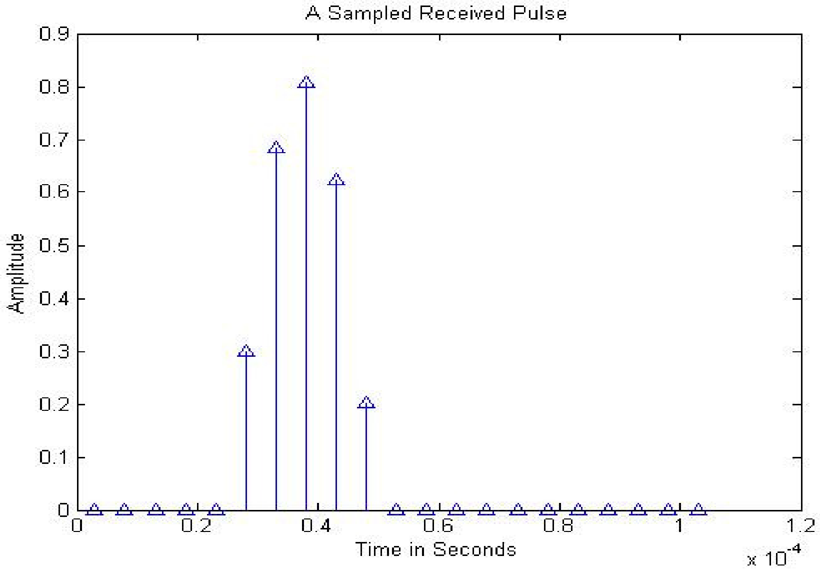


Figure 13: Sampled Waveform

The continuous-time Fourier transform of the ideal sampled waveform is given by:

$$|Y_\delta(f)| = f_s|\sum_{k=-\infty}^{\infty} Y(f - kf_s)|, \tag{5}$$

which produces the following qualitative frequency spectrum shown in Figure 14 below:

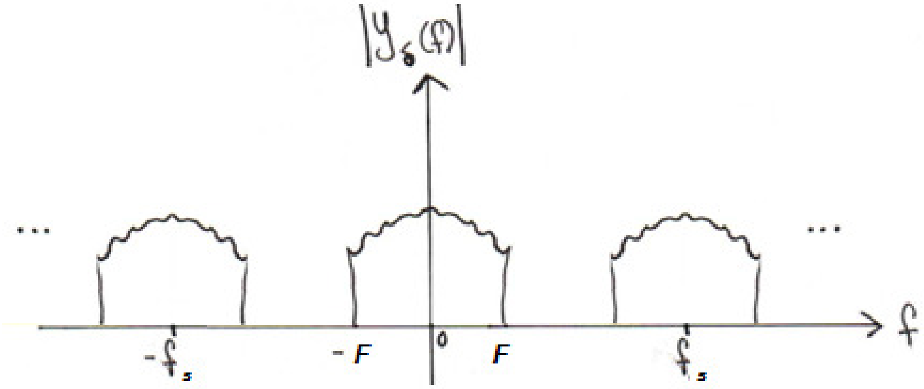


Figure 14: Spectrum of Sampled Waveform

Here the frequency *F* represents the highest frequency component in the received signal *y(t)*, and the spectrum repeats itself every f_s. To prevent aliasing we must satisfy the Nyquist criterion, which dictates that:

$$f_s \geq 2F \tag{6}$$

This equation simply states that the sampling frequency of the analog demultiplexer must be at least 2 times the highest frequency component in the sampled signal to prevent aliasing; note that this constraint will also govern sampling by the ADCs of our MCU,

The sampling capacitors complete the hold circuit, and according to [26], their operation can be approximated by the convolution of the sampled waveform with a single rectangular pulse of the form:

$$r(t) = rect\left(f_c\left(t - \frac{1}{2f_c}\right)\right). \tag{7}$$

The result of such a convolution between $y_\delta(t)$ and *r(t)* in the time domain is shown qualitatively in Figure 15 below; note that this resembles the staircase approximation to our I and Q waveforms obtained at the output of the sample and hold circuit:

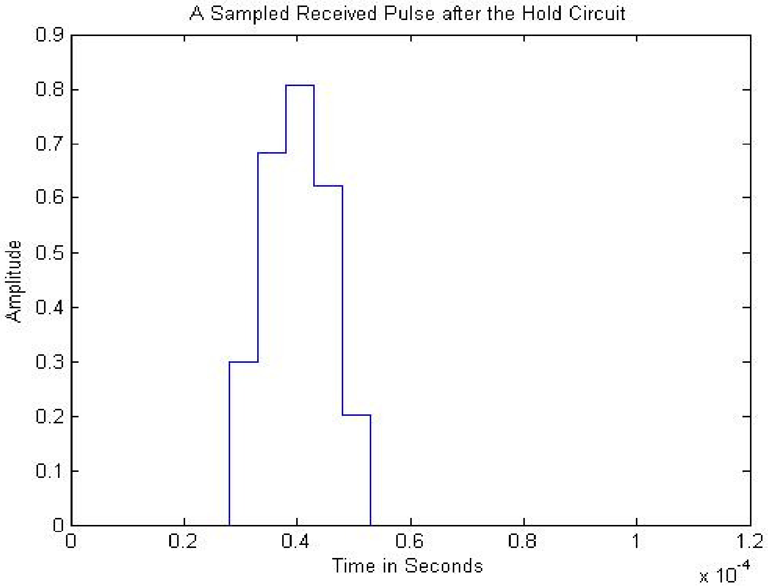


Figure 15: Form of Staircase Approximation to I and Q Components

We know from linear systems theory that convolution in the time corresponds to multiplication in the frequency domain. The magnitude of the Fourier transform of *r(t)* is given by:

$$|R(f)| = T_c|sinc(2\pi T_c t)|, \tag{8}$$

where T_c is the period corresponding to f_c. The central lobe of the un-normalized sinc function spans from $t \in [-1,1]$. Here however, the central lobe is expanded by a factor of f_c, and thus the spectrum of the received signal after passing through the sample and hold circuit is:

$$|Z(f)| = |R(f)||Y_{\delta}(f)| = f_s T_c |sinc(2\pi T_c t)||\textstyle\sum_{k=-\infty}^{\infty} Y(f - kf_s)|, \qquad (9)$$

which qualitatively it looks like the following:

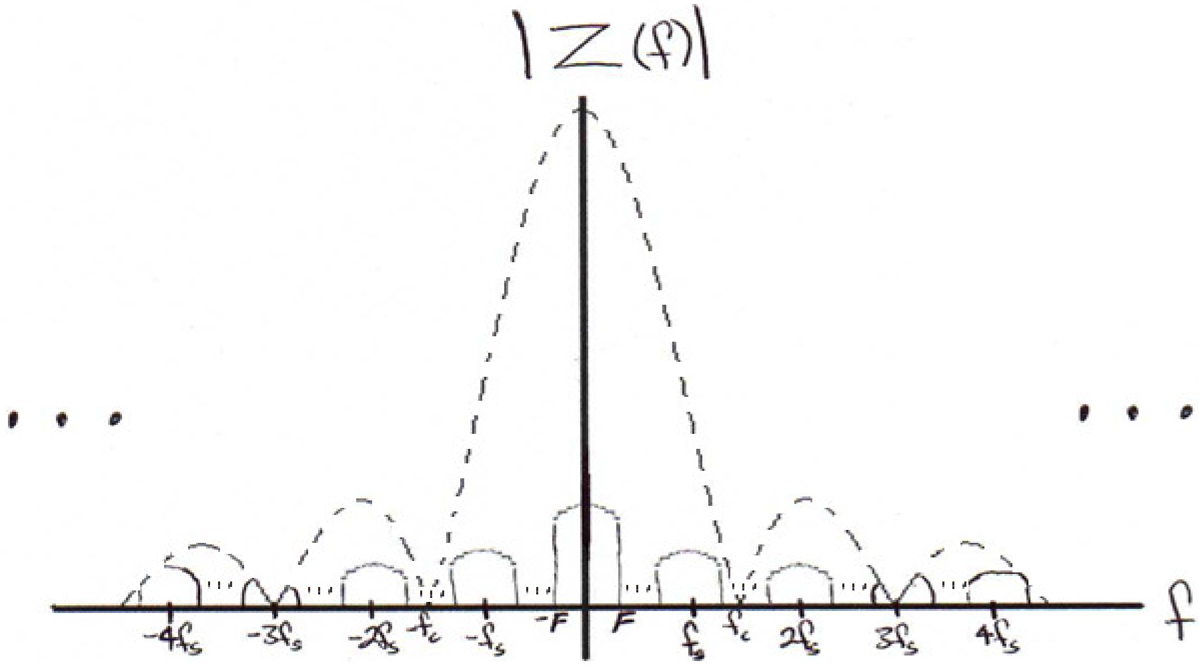


Figure 16: Spectrum of Waveform at Sample and Hold Output

The magnitude response at the output of the sample and hold circuit, $|Z(f)|$, is indicated by the plot in the solid lines; the sinc in the dashed lines represents$|R(f)|$.

This indicates that the LPF at the output of the Tayloe detector must have a bandwidth of at least *F*. According to [3], the bandwidth of a typical transducer is 10% of its operating frequency, which in our case is simply f_c. Thus, a worst case approximation of the received bandwidth is

$$Bandwidth = 600\text{kHz} \cdot 10\% = 60 \text{ kHz}. \qquad (10)$$

Once the signal has been demodulated to the baseband however, we have *F* = 30 kHz, which determines the bandwidth of our final LPF. After passing through this final LPF, the signal will be of the form shown in Figure 17 on the following page; this form resembles the recovered I and Q components of the acoustic signal received from the air:

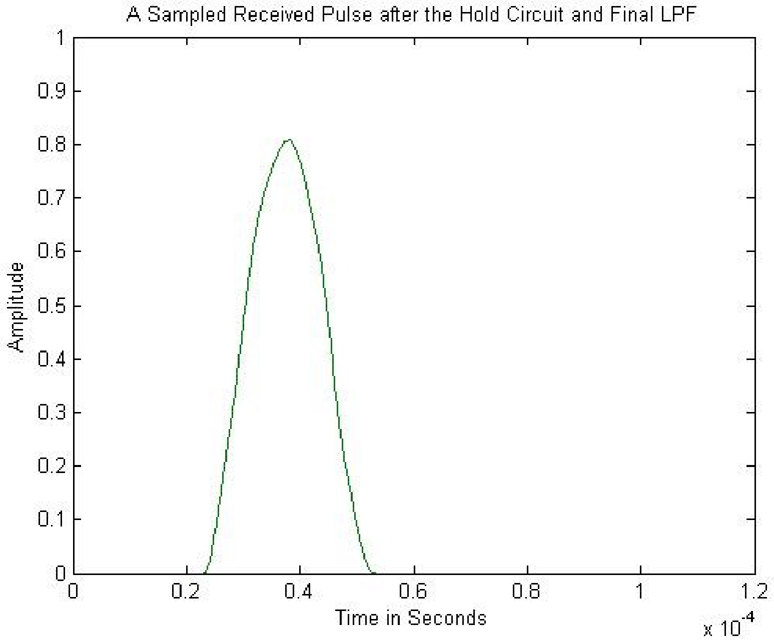


Figure 17: Form of Recovered I and Q Components

We note here that our $f_s = f_c$ satisfies the Nyquist criterion, and that furthermore, at a minimum the ADC sampling rate $f_{ADC-sample}$ of the I and Q components needs to occur at 60 thousand samples per second.

5.2.6 Design of Front-End Stage

Based upon a thorough understanding of the Tayloe detector and several physical phenomena which limit the performance of practical designs, research into components to be used in implementing our design has already been completed. Note that not all the physical phenomena have been covered as of yet; some issue naturally from the following discussion on our design considerations. We now proceed to present our final designs with their derivations, and provide further justifications where necessary.

The design for this circuit is shown on the following page in Figure 18:

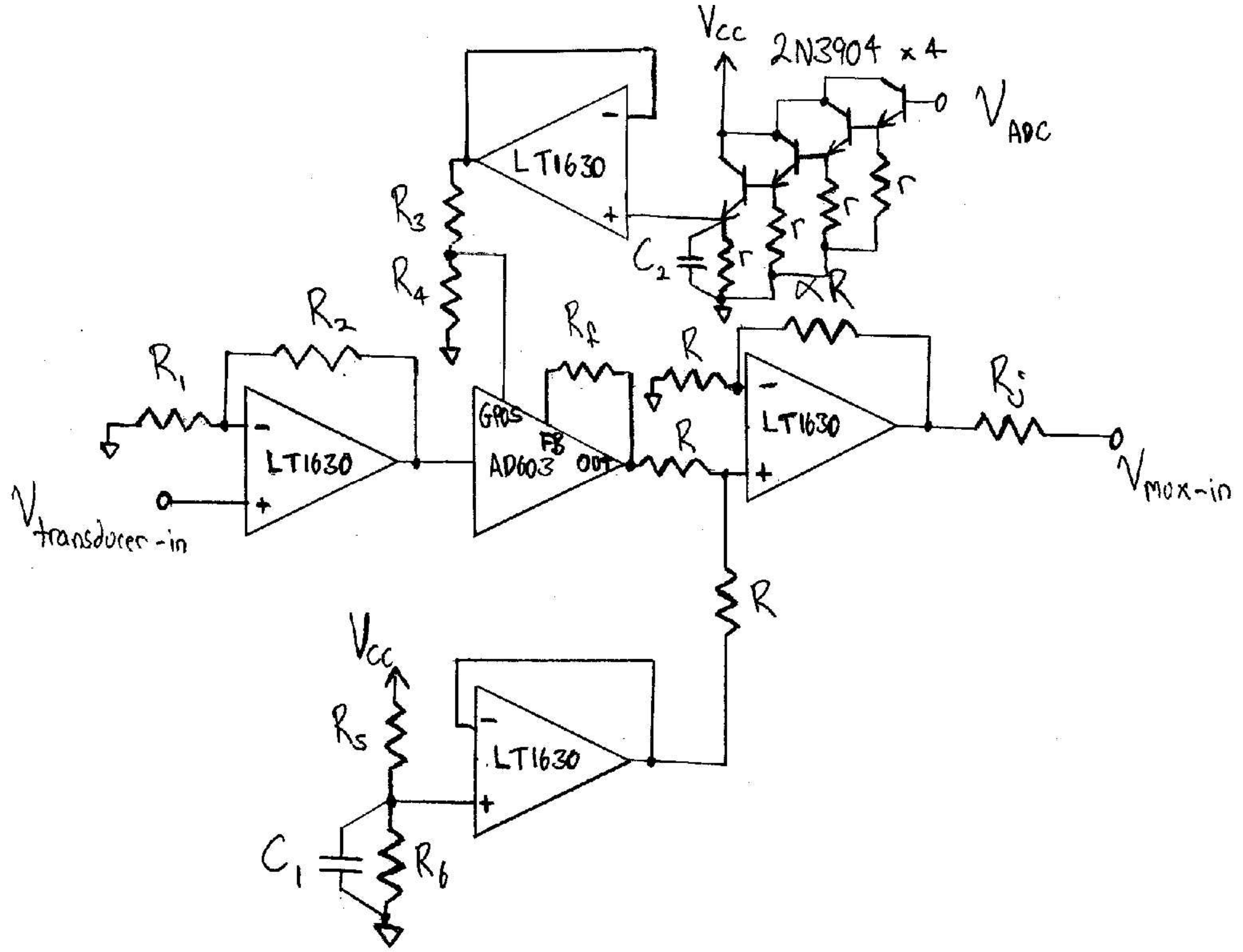


Figure 18: Front-End Stage of Tayloe Detector

The LT1630 is a low noise, rail-to-rail Voltage Feedback Amplifier (VFA) capable of running on low voltage supplies. The AD603 is a low noise VGA with 40 dB dynamic gain range over an interval which is variable. The NPN BJTs are generic 2N3904s.

Some preliminary considerations are in order before we continue:

1. All amplifiers are powered by bipolar 5V supplies, except for the bottom amplifier, which runs on only a single positive 5V supply voltage. The positive supply V_{CC} has a value of 5V, and the negative supply V_{SS} has a value of -5V.
2. The MCU ADC has an input signal range of 0 to 5V, therefore signal swings will be restricted to between 5V peak-to peak. Given that the highest frequency of received signal is 600 kHz, we then require at a minimum a slew rate of:
$$5\text{V}\cdot 600\text{kHz}\cdot 2 = 6\text{V}/\mu\text{s}, \tag{11}$$
where factor of two comes in because the phase difference between the minima and maxima of a sinusoid is 180°. Both amplifiers chosen meet this criterion.

The amplifiers are in their given order because of the following reasons:

1. Ideally, a gain of 20 dB is desired in the front-end amplifier, and the VGA is unable to provide this while maintaining its desired dynamic range [27]; thus, it must be preceded by another amplification sub-stage.
2. The top LT1630 is used to control the gain voltage input of the VGA using the MCU DAC. The cascade of BJTs is to provide DC level shifting so that voltage at the amplifier input ranges from -2.5 to 2.5 V.
3. Finally, the last 2 LT1630s form a summing amplifier to level shift the DC value of the processed signal to 2.5 V, so that the processed signal varies between the MCU ADC input voltage range.

Some other general design considerations are:

1. Decoupling capacitors will be required at all supply voltage pins. We chose 10 µF electrolytic and 100 nF ceramics for this purpose. The former is sufficient at the frequencies we are operating at, as shown by the following calculation:
$$Z = \frac{1}{j\omega C} = \frac{1}{j \cdot 40E3 \cdot 10E-6} = -2.5j\ \Omega, \qquad (12)$$
where 40 kHz is the lowest frequency signal in the system. This demonstrates that any noise on the supply lines that it causes would see almost a short to ground. The ceramic capacitor is required to improve the frequency response of the former, as discussed previously.
2. Differential input protection diodes are not included because the LT1630 already has them internally. These are needed because high differential input voltages are capable of destroying most opamp circuits. Even in a negative feedback topology, this could occur if the applied input voltage became too high, saturating the opamp. This is a plausible scenario in our case; for instance, if the sonar was up against an object, it would be possible for the received voltage to saturate the input amplifier of the front-end stage.
3. Due to the noise reduction of following stages modeled by Frii's equation, we choose to implement our gains entirely in the first sub-stages of our Tayloe detector. This means that we aim to amplify to a peak-to-peak value of 5V by the output of the VGA. Note that this will not saturate the VGA because its output swings within 2 V of its supply rails. This will be calibrated for the user in the software internally in a production model; for a proof-of-concept, it would be sufficient to just implement a volume control on the UI.

The calculation of values and their justifications follow. Let us first consider the resistor R_f, which belongs to the AD603. The purpose of this resistor, located between the terminals marked "FB" and "OUT", is to determine the interval of the dynamic gain range as seen in "Figure 3" of [27]. We choose to set this value to ∞ to yield a range of 10 dB to 50 dB gain. According to [3], typical noise voltage densities at the input amplifier in the front-end stage of a sonar system are 1 to 1.5 nV/$\sqrt{\text{Hz}}$ with a gain of 20 dB. In the non-inverting topology used by our amplifier, the latter would also be the noise gain, giving us a worst case output referred noise density of 15 nV/$\sqrt{\text{Hz}}$. The LT1630 has an input noise voltage density of 6 nV/$\sqrt{\text{Hz}}$ [28], which means that if we ignore all other noise contributions, to stay within 15 nV/$\sqrt{\text{Hz}}$ the closed-loop gain should be restricted to 2.5 V/V, or roughly 8 dB. Thus, the conditions on R_1 and R_2 are

$$\frac{R_2}{R_1} = 1.5. \tag{13}$$

However, this translates to a dynamic gain range of only 18 to 58 dB; so the question at hand is whether or not this issue is worth rectifying.

Consider if the dynamic gain interval of our final design was [18, 58] dB, which corresponds to a maximum gain of roughly 800 $\frac{\text{V}}{\text{V}}$ and a maximum received signal peak-to-peak voltage swing of 6.25 mV before saturation. This is as opposed to a desired gain of 1000 $\frac{\text{V}}{\text{V}}$ corresponding to a maximum received signal peak-to-peak voltage swing of 5 mV. Given that the pressure of the received signal is inversely proportional to the twice the maximum detection distance (because the signal travels to the target and back) [29] and this is related to the voltage [3], this means that the maximum detection distance has decreased by 20%; this is definitely a cause for concern. To make up the difference in gain while avoiding exceeding the target output noise contribution of the front-end sub-stage, we elect to transfer the remaining 2 dB of gain to the summation stage while abiding by the above conditions on R_1 and R_2.

Given that the noise performance of the system is one of the primary goals of this project, we wish to determine the noise at the input of the MCU ADCs. Our entire analysis only takes into account broadband noise, ignoring the pink noise (noise at low frequencies) because the normalized noise of the LT1630 at 1 Hz is relatively low at 30 nV/$\sqrt{\text{Hz}}$, so it's contribution pales in comparison to the broadband noise; this approximation can also be justified by examining the pink noise formulae in [30]. Also, the current noise is ignored because so long as the equivalent resistance of R_1 and R_2 looking out of the inverting input of the LT1630 is kept small (in the range of hundreds of ohms), the voltage noise it translates into will be small as well since it is 4 orders of magnitude smaller. Thus, we have the additional constraint that the equivalent resistance of R_1 and R_2 should be in the hundreds of ohms; this is a reasonable expectation since the same equivalent resistance is desired to be small for all gains to produce a low thermal voltage noise. Thus, we choose standard Electronic Industries Alliance (EIA) values of 100 Ω and 150 Ω, respectively. Thermal noise is given by the following formula detailed in [19]:

$$V_t = \sqrt{4k_B T R_{eq}}, \tag{14}$$

where V_t is the thermal noise in V/$\sqrt{\mathrm{Hz}}$, k_B is Boltzmann's constant, T is the temperature in Kelvin, R_{eq} is the equivalent resistance seen from the inverting terminal of the amplifier, and B is the effective noise bandwidth of the system. Using a worst case value of 1 kΩ for R_{eq} (since we aim for a value in the hundreds of ohms), we obtain 4 nV/$\sqrt{\mathrm{Hz}}$. The total noise voltage density at the amplifier input is then given by a root-sum-square of the input noise voltage density of the LT1630 (6 nV/$\sqrt{\mathrm{Hz}}$) and the thermal noise voltage density to yield 7.2 nV/$\sqrt{\mathrm{Hz}}$. This means that the output noise density will be greater than the targeted 15 nV/$\sqrt{\mathrm{Hz}}$; however, we will demonstrate in the calculations which follow that the performance obtained from our design is more than acceptable.

To determine the noise at the output of the front-end amplifier, we follow the calculations detailed in [30] for an in-depth discussion on noise analysis, refer there. First, the closed-loop noise bandwidth must be found; this is given by:

$$closed\ loop\ noise\ bandwidth = \frac{unity\ gain\ bandwidth}{noise\ gain} \tag{15}$$

$$= \frac{30\ \mathrm{MHz}}{2.5} = 12\ \mathrm{MHz}, \tag{16}$$

where the unity gain bandwidth is found in [28]. Then, the effective noise bandwidth of the amplifier over the entire frequency spectrum is given by:

$$effective\ noise\ bandwidth = 12 \cdot 1.58 = 18.96\ \mathrm{MHz},$$

where the factor of 1.58 is detailed in [22]. Since our received signal has at a worst case 60 kHz of bandwidth contained within the effective noise bandwidth but centered around the 600 kHz carrier, we use it instead for the noise bandwidth of the *front-end amplifier*; thus its noise is:

$$Noise\ added = \left(7.2\,\frac{\mathrm{nV}}{\sqrt{\mathrm{Hz}}}\right)(2.5\ \frac{\mathrm{V}}{\mathrm{V}})\sqrt{60\ \mathrm{kHz}} = 4.4\ \mathrm{\mu V} \tag{17}$$

The *total* input noise density (including current and voltage sources) of the AD603 itself at its own input is specified in [27] to be 1.3 nV/$\sqrt{\mathrm{Hz}}$, and its maximum gain of 50 dB corresponds to 316.2 $\frac{V}{V}$. Also, its unity gain bandwidth is at most 4 MHz, which means that its effective noise bandwidth is only:

$$effective\ noise\ bandwidth = \frac{unity\ gain\ bandwidth}{noise\ gain} \cdot 1.58 = \frac{4\mathrm{MHz}}{316.2\frac{V}{V}} \cdot 1.58 = 20\ \mathrm{kHz}, \tag{18}$$

This is outside the spectra of all possible received signals, given that $f_c \epsilon [40, 600]\mathrm{kHz}$ and the bandwidth of the receiver at a worst case is 10% of that; thus, we ignore this contribution. In other words, it is dwarfed by the noise contribution of the input amplifier stage, as predicted by Frii's equation. Therefore, the noise at the output of the AD603 is calculated to be:

$$(4.4\ \mu\mathrm{V})\left(316.2\frac{V}{V}\right) = 1.39\ \mathrm{mV} \tag{19}$$

Just as for the input buffer amplifier, if the values of R and α are kept small such that the equivalent resistance seen from the inverting input of the final amplifier is in the range of several hundred ohms, the current noise there can be neglected. Again, we neglect pink noise, and consider only the broadband voltage noise and the thermal noise of the resistors. The value of α to yield 2 dB is 1.5, and this reaches the requirement of 60 dB maximum gain. If we choose R to be 100 Ω, then the input noise voltage of the final LT1630 sub-stage due to itself is approximately again 1.76 $\mu\mathrm{V}$, which is the value found for the input amplifier, since the equivalent input resistance at the inverting input is approximately equal. We take this value multiplied by the gain from the non-inverting input of $2.5\frac{\mathrm{V}}{\mathrm{V}}$ as the output noise of this back-end amplifier due to itself, again yielding 4.4 µV, which is already seen to be dwarfed by the prior noise contributions, again as predicted by Frii's equation. The noise at the output of this amplifier due to the previous stages is thus approximately given by the output noise of the AD603 multiplied by the gain of this stage, and is:

$$(1.39\ \ \mathrm{mV})\left(1.25\frac{V}{V}\right) = 1.74\ \mathrm{mV}. \tag{20}$$

Given that there are no more gain stages in the rest of the design, this noise should be predominant until the MCU ADC inputs. We can thus estimate the effective ADC resolution, SNR, and CDR of our design. The DAC is 10 bits across a 5V span, therefore each of its levels occupies a voltage interval of 4.88 mV; this means that we have an effective ADC resolution of:

$$10 - log_2\left(\frac{1.74}{4.88}\right) = 11.48\ \mathrm{bits} \tag{21}$$

after the noise is accounted for, in the worst case first order approximation. This is of course impossible with a 10 bit ADC, and the interpretation of this result is that the noise is not enough to affect the resolution of the ADC operation. For the SNR, we assume a maximum signal swing on 5 V peak-to peak, which gives us:

$$SNR = 10\log\left(\frac{(5\mathrm{V})^2}{(1.74\ \mathrm{mV})^2}\right) = 69.15\ \mathrm{dB} \tag{22}$$

According to [30], the noise voltage calculated at the input of the ADC multiplied by a factor of 6 is 3 standard deviations away from the mean in the probability density function of the noise, which can be treated as Gaussian. In Figure 12 the received waveform in air was shown to lack a plateau in its envelope; thus the CDR would not be a good measure there. The received waveform in water does have a plateau, according to [3], and we can estimate the mean value there to be 2.5V since we maximize the signal swing to match the 5V dynamic range of the ADC to take full advantage of its resolution, but we have a DC offset of 2.5 V. Thus, the CDR can be estimated as:

$$20\log\left(\frac{\mu}{\sigma}\right) = 20\log\left(\frac{2.5\text{ V}}{\frac{1.74\text{ mV}}{3}}\right) = 72.7\text{ dB} \tag{23}$$

Now we choose values for R_3 and R_4 - these are located near the "GPOS" pin in Figure 18, which has an input impedance of 50 MΩ [27]. The gain can be varied by applying an input voltage of between -500 mV to 500 mV at this pin, corresponding to the minimum and maximum amplifier gains, respectively. To produce these voltages, the top LT1630 amplifier has been configured to run off bipolar 5 V supplies, and has a cascade of 4 2N3904 NPN BJTs at its input. The emitter resistance values are not of any consequence so long as they bias the transistors in forward active region; values of EIA values of 5.6 kΩ have thus been chosen for *r.* This level shifts the MCU DAC output range to a voltage within [-2.5, 2.5] V. Therefore, to obtain our desired values, we require a voltage division ratio of 1:5. Given that the output impedance of the LT1630 is around 0Ω [28], and high input impedance of the "GPOS" pin, we are at liberty to choose a wide range of values for R_3 and R_4 without suffering loading effects. Standard EIA values of $R_3 = 402\Omega$ and $R_4 = 100\Omega$ are chosen; and a noise analysis follows to confirm their usability.

There are two sources of noise here – the thermal noise of the equivalent resistance here, and the noise due to the LT1630 buffer; a third source of noise, that of the AD603 at the "GPOS" pin, cannot be accounted for given that manufacturers typically only specify noise densities at the signal input and outputs of amplifiers – thus we are forced to ignore it here. To obtain the noise due to the buffer, we need to obtain its effective noise bandwidth, which is:

$$effective\ noise\ bandwidth = \frac{unity\ gain\ bandwidth}{noise\ gain} \cdot 1.58 = \frac{30\text{MHz}}{1\frac{V}{V}} \cdot 1.58 = 47.4\text{ MHz} \tag{24}$$

This is contained the 60 kHz worst-case bandwidth centered on a carrier frequency of 600 kHz. This yields an output noise at the buffer of:

$$(6\frac{\text{nV}}{\sqrt{\text{Hz}}})(\sqrt{60\text{ kHz}}) = 1.46\ \mu\text{V}. \tag{25}$$

Note that the noise from the other source - the transistors - is bypassed to ground by means of the decoupling capacitor C_2. Thus, the noise just calculated is also the noise at the "GPOS" pin. Assuming clean power supplies, the lowest frequency signal in the system is that of a received signal with a 40 kHz carrier frequency. If we then aim for an impedance of 1 Ω at that frequency this, yields a capacitance of C_2 = 25 µF; note that to decouple higher frequency noise, a ceramic capacitor should be added in shunt here – again a typical value used is 100 nF. The same logic can be applied to C_1. The DAC is also 10 bits across a 5V span, therefore each of its levels occupies a voltage interval of 4.88 mV, which is 2 orders or magnitude higher than our noise voltage. Thus, we conclude that our choice of R_3 and R_4 is safe from a noise standpoint.

To complete our analysis, we also need to select values for R_5, R_6, and R_j. The DC value desired at the output of the bottom buffer is 2.5/1.25 = 2V. Thus, using the equation for the transfer function of a voltage divider, we choose values of 150 Ω and 100 Ω for R_5 and R_6, respectively. The selection of resistor R_j will be dealt with in the design section for the sample and hold stage, since it is more relevant there.

Of course, this assumes that issues such as adequate decoupling of power pins and analog from digital power supplies are all dealt with, in addition to other possible factors; several of these such as the latter are virtually impossible to implement on a breadboard, which itself has notorious high-frequency performance due to parasitic capacitances and lead inductances. Thus, we expect these figures to be significantly lower in the proof-of-concept version built on the breadboard.

5.2.7 Design of Sample and Hold Stage

The design for the circuit is shown below in Figure 19:

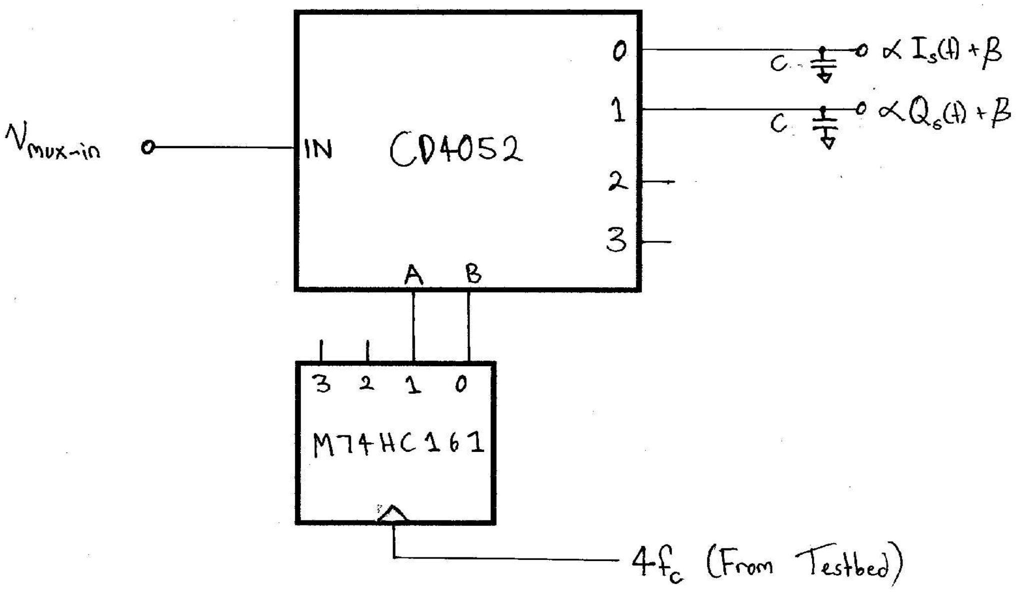


Figure 19: Sample and Hold Stage of Tayloe Detector

The M74HC161 is a high speed 4-bit counter with a typical maximum frequency of 62 MHz, far above the 2.4 MHz which we require. Using its two least significant output bits, it drives the CD4052, which is a 1:4 analog demultiplexer possessing a bandwidth of 40 MHz, again far above the 2.4 MHz we require. Both circuits are capable of running rail-to-rail on single 5V positive supplies to match the signal levels from the front-end stage of the receiver. As with the rest of the system, proper decoupling of power supplies will be essential.

We now proceed to calculate the values of C and R_j from the previous stage. From our mathematical discussion, we found that the bandwidth needed was 60 kHz. The bandwidth of the BPF is given in [12] as:

$$Bandwidth = \frac{1}{4\pi R_{in} C}, \tag{26}$$

where R_{in} is the input resistance looking back into the front-end stage, and C is the capacitance of the individual sampling capacitors. Note that this definition of bandwidth is from the center frequency to the cutoff frequency only, so the bandwidth used in this equation is 30 kHz. R_{in} consists of the resistance looking towards the front-end stage from the capacitor, and this is:

$$R_{in} = R_j + R_{mux-on}, \tag{27}$$

where R_{mux-on} is the on resistance of the demultiplexer, which is found in [31] to have a typical value of 270 Ω. If we then aim for a capacitance of 1 nF, we obtain a value of 2.65 kΩ for R_{in}, which means that R_j has a value of 2.4 kΩ; we choose the closest standard EIA value of 3.92 kΩ.

Note that the outputs of this stage are $\alpha I_s(t) + \beta$ and $\alpha Q_s(t) + \beta$; these are the sampled I and Q components which have been amplifier and level shifted. The presence of the DC offset means that we cannot add together the 0° and 180° outputs, as well as the 90° and 270° outputs to obtain 6 dB of gain from this stage as some implementations of the Tayloe detector do. However this is insignificant, because as Frii's equation demonstrates, it is always better to maximize the gain in the earlier stages of a receiver, and we have done so already. Also, according to [3] the addition of the outputs creates a ripple superimposed atop the I and Q components, adding unnecessary noise. Due to these factors, and emphasis on low noise, our present design is clearly superior.

5.2.8 Design of Output Conditioning Stage

The design of the output conditioning stage is shown below in Figure 20:

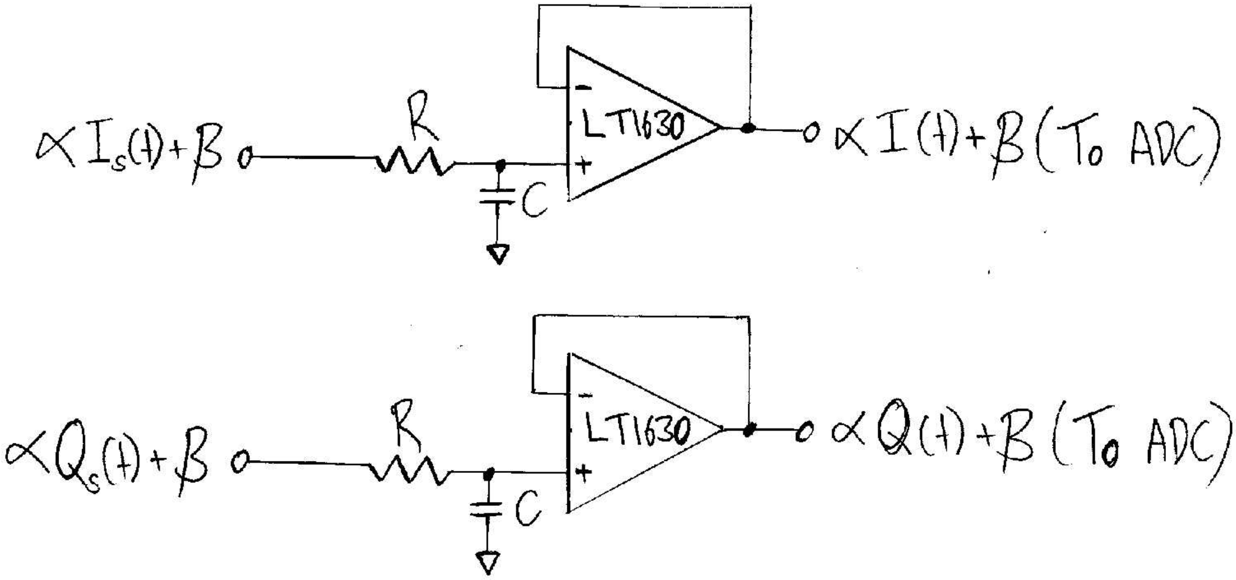


Figure 20: Output Conditioning Stage of Tayloe Detector

This stage consists of a simple first order LPF followed by an ADC driver circuit, which is a simple LT1630 buffer. The LT1630 runs on bipolar 5 V supplies so that its internal bias circuitry produces a common mode voltage of 2.5 V, allowing the signal from the sample and hold stage to maintain its full 0 to 5 V swing. The excellent characteristics of the LT1630 VFA, as well as the need for adequate decoupling have already been addressed.

We now proceed to determine the values of the components in this stage. The cutoff frequency of this LPF is given by:

$$f_{cutoff} = \frac{1}{2\pi RC}, \tag{28}$$

and f_{cutoff} was determined earlier to be 30 kHz. If we aim for a capacitance of 1 nF, then we require a resistance of 5.3 kΩ; we select a standard EIA value of 4.7 kΩ in series with 560 Ω.

5.2.9 Measured Results

The receiver detailed above has already been constructed and evaluated in an acoustic setting, and the results are presented here. Note that the following results are at a specified gain setting on the VGA. However, these are valid for all gains since the predominant noise originates at the front-end amplifier in the front-end stage of the system, as detailed previously in section 5.2.6 and Frii's equation; in other words the noise goes through the exact same stages as does the signal of interest, and because SNR are and CDR are ratios, these gains should cancel yielding the same ratios. Figure 21 presents the recovered I and Q waveforms below; note the resemblance to the mathematical model presented in Figure 17:

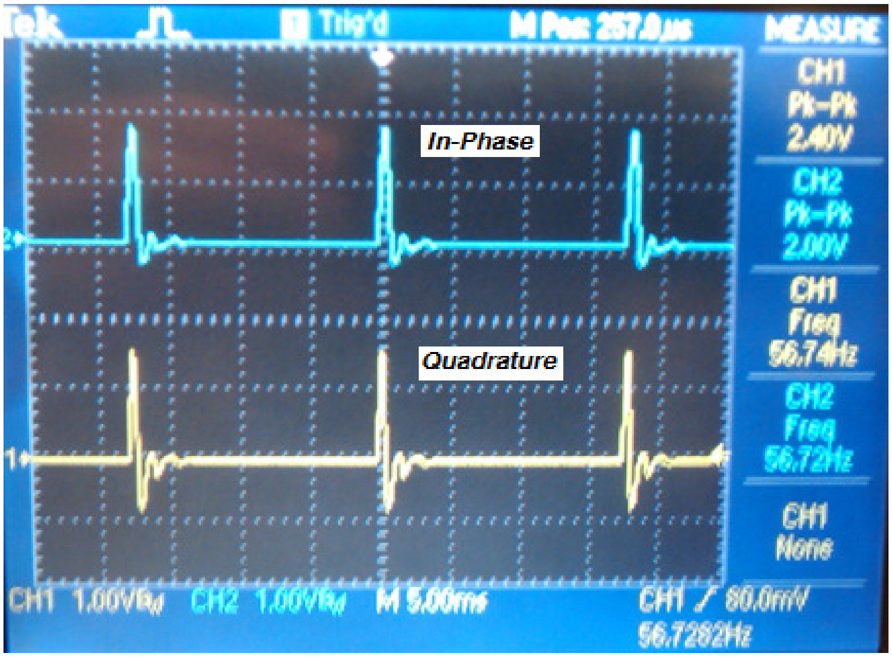


Figure 21: Recovered I and Q Waveforms

Figure 22 presents a screen capture of the noise at the input to the ADC:

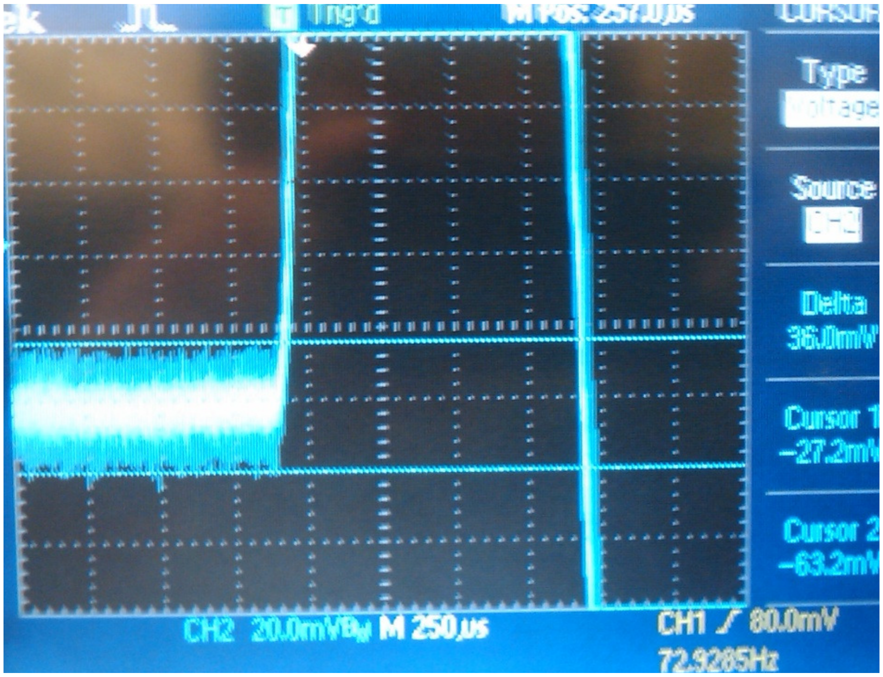


Figure 22: Noise at ADC Input

We note that there are two main regions of noise; these are higher amplitude 330 kHz spikes superimposed on lower amplitude noise. We include them both in the noise measurement, yielding 36.0 mV as seen in Figure 22. At the time of this writing, the team is investigating the removal of the 330 kHz noise; doing so would lower the noise measurement to just 16.0 mV as seen in Figure 23 below:

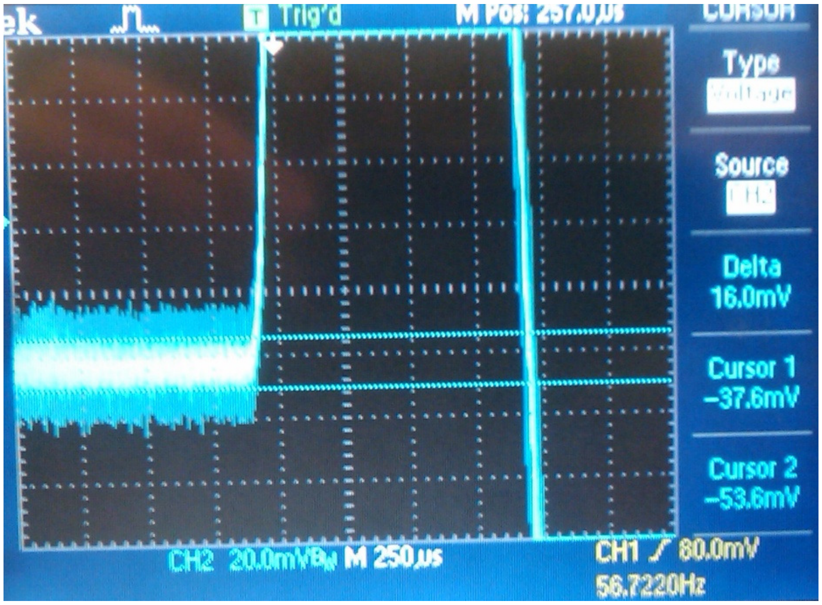


Figure 23: Noise Measurement without 330 kHz Component

Figure 24 presents the amplitude of the I and Q components at the ADC input for the same gain settings as the noise shown above.

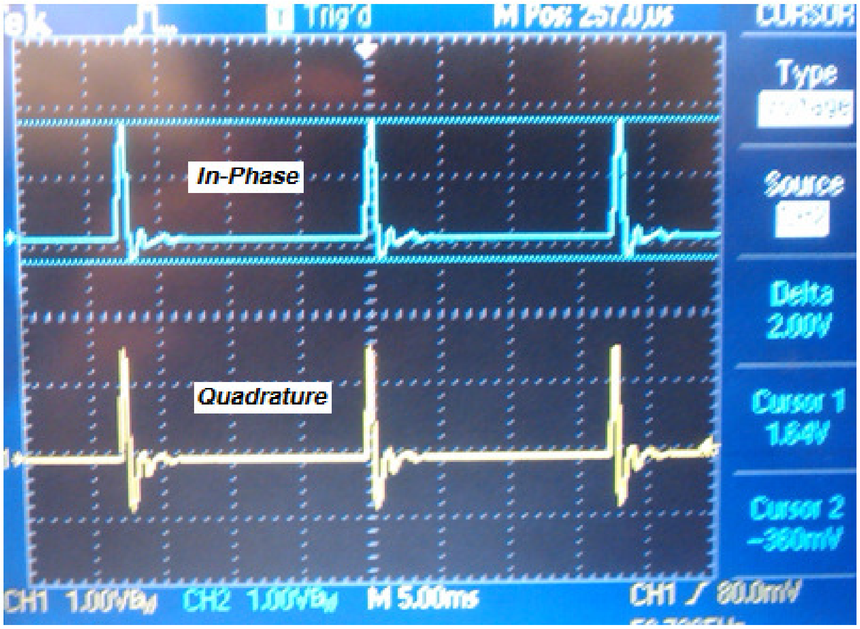


Figure 24: I and Q Amplitudes at ADC Input

From the information presented in Figures 22, 23 and 24, we can calculate the measured SNR; as can be seen in Figure 24, the topmost signal has the smaller amplitude, and we elect to go with that for a worst-case SNR. The measured SNR with all noise components is calculated to be:

$$10\log\left(\frac{(2.00\text{ V})^2}{(0.036\text{ V})^2}\right) = 34.89\text{ dB}, \qquad (29)$$

which is substantially lower than the idealized value calculated above, but should improve if the 330 kHz noise is removed. Doing this yields the following SNR:

$$10\log\left(\frac{(2.00\text{ V})^2}{(0.016\text{ V})^2}\right) = 41.94\text{ dB}. \qquad (30)$$

According to [32], an SNR of 20 dB for a receiver is considered as high; this seems quite promising, since noise performance is one of the tenets of the design of the AquaScan.

As mentioned before, the absence of a plateau in the received air signal makes the CDR a bad measure of performance here. Also, to measure the CDR requires sampled datasets from which the mean and variance may be calculated. At the time of writing, the functionality of the ADC has just been implemented, but the water transducers have not been tested.

5.2.10 Cost Estimates

In this section, we estimate the cost for both a single channel and six channel sonar system, the latter being the ultimate goal of this project; cost is the other tenet of our design. Estimates are based on current unit prices on Digikey Canada and Newark Canada, and do not include development costs, the URL transducer driver, transducers, or minor components such as resistors and capacitors. We start with a single channel design; from Table 1, this requires:

Table 1: Cost Estimate of Single-Channel Sonar

Component Name	**Per Unit Cost ($)**	**Quantity**	**Cost(S)**
BJT (2N3904)	0.059	7	0.413
VFA (LT1630) – Dual	6.49	6/2 => 3	19.47
VGA (AD603)	10.53	1	10.53
Counter (M74HC161)	0.86	1	0.86
Analog Demultiplexer (CD4052BCN) –Dual	0.64	1	0.64
MCU (DSPIC33FJ16GS502)	6.26	1	6.26
RS232 Transceiver (MAX3232EID)	1.85	1	1.85
Pin Adaptor (A751-ND)	8.17	1	8.17
Regulator (LM7805)	0.53	1	0.53
40 MHz Crystal	0.86	1	0.86
RS232 Connector	5.67	1	5.67
RS232 to USB cable (from NCIX Canada)	16.99	1	16.99
Total			72.24

The case of a six channel design is shown in Table 2; note that this is just a first approximation:

Table 2: Cost Estimate of Six-Channel Sonar

Component Name	Per Unit Cost ($)	Quantity	Cost($)
BJT (2N3904)	0.059	7	0.413
VFA (LT1630) – Dual	6.49	26/2 => 13	84.37
VGA (AD603)	10.53	6	63.18
Counter (M74HC161)	0.86	1	0.86
Analog Demultiplexer (CD4052BCN) -Dual	0.64	3	1.92
MCU (DSPIC33FJ16GS502)	6.26	2	12.52
USB Transceiver (SP5301CY-L)	1.88	1	1.88
Regulator (LM7805)	0.53	1	0.53
40 MHz Crystal	0.86	1	0.86
USB Connector	1.25	1	1.25
USB cable (from NCIX Canada)	5.54	1	5.54
Printed Circuit Board + Contingencies	50.00	1	50
Total ($)			223.32

Several components can be shared among the channels and do not need to be replicated; this includes the buffers in the front-end of the receiver, and the level-shifting cascade of BJTs in Figure 18. As well, the existing regulator, BJT level-shifting cascade, and MCU crystal of the testbed can support a six channel system. One analog demultiplexer chip is needed per two channels since it is a dual, and one M74HC161 counter can drive all the demultiplexer chips. One MCU is fast enough to handle 2 channels, and one VGA is needed per channel. RS-232 is not fast enough to handle 6 channels, so a USB transceiver, connector, and cable are needed instead. Lastly, the final product must be fabricated on a PCB. This yields the final quantities shown in Table 2; note that the total cost falls well within the $300 targeted earlier.

5.3 ALTERNATIVE RECEIVER DESIGNS

Several potential receiver designs are under consideration at the moment; however, no promising leads have been found as of yet, and this is also due to the amount of labor involved with the development of the testbed and the Tayloe detector. It is our intention to research, develop, and evaluate at least 2 more promising alternative designs in time for the completion of this project.

6. TEST PLAN

Both hardware and software test plans must be passed to make sure the entire AquaScan sonar system works properly in all specified cases before releasing the final product. For the sake of brevity, these are described only in a qualitative manner.

6.1 HARDWARE

The hardware test plan is mainly concerned with the performance of the various subsystems within the AquaScan for all carrier parameters within the required range. The hardware test plan on the side of the testbed is intimately related to the firmware testing, and thus tests related specifically to the testbed will not be described here. The test plan in this section deals mainly with the candidate receiver modules and the overall system, and is as follows:

1. Submersible parts will be submerged in water for reasonable periods of time and reasonable sets of extreme conditions, and should continue to operate correctly. An ideal data set will serve as a control group, and received data will be compared against it to verify correct operation.
2. Power will be suddenly removed from the system while it is in operation, and no electrical damage should occur.
3. For the diode protection circuitry at the input of the receiver, a large signal is applied, and the system should survive the encounter. In addition to this, the system must recover its normal operations within a short period of time after the removal of the signal. This simulates the case where the sonar accidentally comes in contact with an object at too close a distance for the current gain setting.
4. The system will be subjected to reasonable shock and vibration, and measurements will be verified to continue to be taken accurately in the manner described in 1; this test is appropriate because the sonar is commonly found in maritime vessels, and the turbid environment and even the motors of the vessel are sources of vibration. In addition to this, collisions and sudden starts/stops are reasonable scenarios where shock may occur.
5. The water in which the sonar is submerged will be disturbed, and measurements will be verified to continue to be taken accurately in the manner described in 1; this test is appropriate for the same reasons as in 3.

6.2 SOFTWARE

6.2.1 User Interface Test Plan

The test plan for the user interface includes both sanity tests and unity tests; only one case is tested for each function and feature in the sanity tests, while all extreme cases are must be passed for the unity test. The following section is a description of the sanity tests:

1. For the connection management, the GUI must connect to the COM port when the program starts and disconnect before the program exits.
2. For each carrier setting, the GUI sets one parameter and sends the corresponding data to the MCU. The oscilloscope is used to verify that the correct data has been sent.
3. For each change in default settings when the system is idle, the corresponding defaults file will be checked to ensure that that the changes are accurately reflected.
4. For the graphical plot, a constant value from the ADC port of the MCU is passed to the GUI, and must appear correctly in the GUI for the length of the test.
5. For each button, text box, and list box of the GUI, valid input must correctly register, and errors must be caught (but not necessarily handled).

The following section describes the unity tests:

1. For the connection management, the GUI must handle all reasonable connection and disconnection scenarios, such as a disconnection during transmission or an attempted connection to a busy COM port.
2. For each system setting, the GUI must still be able to correctly send out-of-bounds parameter values and give a warning to the user; the former will be verified on the oscilloscope.
3. For each change of default setting, regardless of when this change is made (during transmission, idle, or other status), the change must be reflected in the defaults file.
4. For the graphical plot, signals must be correctly displayed in quasi-real time for all desired ADC sampling rates, carrier frequencies, and gain ranges; the export data file must be written correctly.
5. For each button, text box, and list box of the GUI, valid input must correctly register, errors must be handled correctly, and warning messages must be displayed to the user.

6.2.2 Firmware Test Plan

The firmware test plan is mainly concerned with the accuracy of the MCU response to various system requests within the required range of carrier parameters; a brief description is as follows:

1. Change the frequency, time between pulses, and the pulse length using the GUI and observe PWM channel one and two on the oscilloscope. The shape and frequency of the complementary pair (PWM1H and PWM1L) and the frequency of the 4X clock observed by the oscilloscope must correspond to the values set in the GUI.
2. The ADC sampling rate is measured by setting a timer in ADC ISR. The timer measures the time between the two ADC samples and send the result to the GUI. This sampling time must conform to the ADC sampling rate requested by the GUI.
3. The accuracy of the UART connection is measured by sending a pre-defined array of integers in the ADC ISR and comparing the received array in the GUI with the original array in MCU.
4. The stability of the system is measured by changing the system parameters in GUI to extreme values (Frequency higher than 600 kHz or less than 20 KHz) and observing the MCU response by probing the MCU PWM channels using an oscilloscope.

7. DEVICE ENCLOSURE

We chose to use aluminum for our waterproof enclosure because it possesses several desirable characteristics [33], such as: widespread availability; malleability and ductility; lightness of weight; strength; good thermal conductivity; relative inertness in salt water compared with other metals [34]; non-toxicity; recyclability; and being relatively inexpensive [35]. An obvious disadvantage is that aluminum is also an excellent conductor of electricity; to deal with this issue, we propose to isolate the circuitry from the casing by fixing it upon poles formed from an electrical insulator, such as an inexpensive rubber, which would also provide some measure of shock absorption in contrast to more rigid insulators such as polyvinyl chloride. An illustration of the proposed enclosure is shown in Figure 25 on the following page:

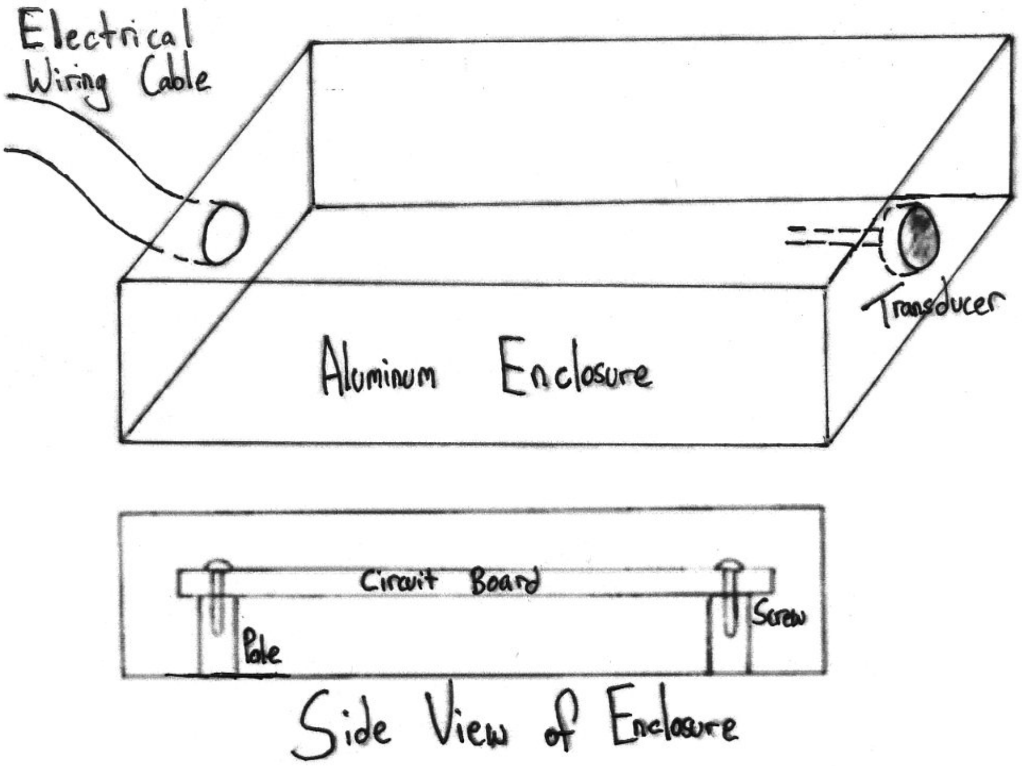


Figure 25: Illustration of Proposed Enclosure

There are four poles for each of the corners of the circuit board; the poles function to isolate the circuitry from the electrically conductive casing and to prevent movements inside the container. In addition to this, both sides of the container will have a drilled hole; one side for the wiring through a cable, and the other for the water transducer. The holes will be filled in with a glue gun to prevent any water from leaking in.

At the final stage of our design, we will apply a thin, transparent layer of parylene material to the AquaScan sonar system. This thin layer of parylene will provide protection for the AquaScan in case the waterproof container is penetrated; the parylene layer will not react with any common solvents and bacteria. Parylene is also a good electric insulator, which reduces the risk of electrocution to those handling the device. Moreover, parylene coatings radiate heat outward into the air and water thus providing a passive cooling for the AquaScan. Finally, parylene also possesses good thermal characteristics, and is able to withstand temperatures between -200°C to +200°C [36].

8. CONCLUSION

Development of the testbed is nearing completion, one candidate receiver has already been completed, and its evaluations have already begun. The two tenets of our design are cost and noise performance, and the results of the initial noise evaluations, detailed briefly in section 5.2.9, seem promising; further optimizations to the AquaScan are being considered to further improve its noise performance. Furthermore, a cost analysis has been performed for both a one and six-channel sonar system, and found to be well within the initial targets. Our initial research has also identified several other promising candidate receivers for further investigation, which should commence shortly. Since both research and development are well underway and conform to the tenets of our design, we are optimistic that by December 2009, we will have produced a worthy candidate to complement the 3D sidescan sonar technology already developed by Dr. Bird of the URL at SFU.

9. REFERENCES

[1] AquaSense Systems, “Functional specifications for a qualified sonar system for the masses.” Canada, October 19, 2009.

[2] Microchip Technology Inc., “dsPIC33J06GS101/X02 and dsPIC33J16GSX02/X04 data sheet,” March 26, 2009, http://ww1.microchip.com/downloads/en/DeviceDoc/70318D.pdf.

[3] J. Bird (private communication), October, 2009.

[4] Fairchild Semiconductor, “IRFR420, IRFU420,” September 17, 2004, http://www.datasheet catalog.org/datasheet/fairchild/IRFR420.pdf.

[5] Texas Instruments, “MAX3232,” September 16, 2009, http://focus.ti.com/lit/ds/symlink/ max3232.pdf.

[6] Company Seven, “Autoguider port RJ-11 connector pin assignments,” 2005, http://www. company7.com/library/astrophy/rj11pinouts.html.

[7] Cprogramming.com, “What advantages does C# have?,” August 1, 2007, http://cboard. cprogramming.com/csharp-programming/87269-what-advantages-does-csharp-have.html.

[8] MicroChip Technology Inc., “dsPIC33FJ16GS502,” March 26, 2009, http://www.microchip. com /wwwproducts/Devices.aspx?dDocName=en537164.

[9] Radio-Electronics.com, “Superhet or superheterodyne radio receiver,” n.d., http://www. radio-electronics.com /info/receivers/superhet/superhet.php.

[10] RF Engineer Network, “Radio architectures, pt 4: sensitivity, noise, front-end amplifiers,” June 18, 2009, http://www.rfengineer.net/1492/radio-architectures-pt-4-sensitivity-noise-front-end-amplifiers/.

[11] D.R. Tayloe, “Ultra low noise, high performance, zero IF quadrature product detector and preamplifier,” July 10, 2004, http://wb9ipa.qrpradio.com/60meter/exciter/Tayloe_mixer _x3a .pdf.

[12] G. Youngblood, “A software-defined radio for the masses, part 1,” *QEX,* pp. 13-21, August 2002.

[13] D.R. Tayloe, “Product Detector and Method Therefor,” US Patent 6,230,000, May 8, 2001.

[14] J.Bird (bird@cs.sfu.ca), "Re: ENSC 440 sonar project question," personal email (October 2, 2009).

[15] C. Angrove, “RF design,” 2009, http://www.chrisangove.com/ee_ref/rfdes.htm.

[16] “Notes set 7: first look at noise in circuits,” class notes for ECE 594, Department of Electrical and Computer Engineering, University of California Santa Barbara, May 15, 2004.

[17] Analog Devices, “AD8331/AD8332/AD8334,” April 29, 2008,http://www.analog.com/static/imported-files/data_sheets/AD8331_8332_8334.pdf.

[18] D.S Atmaja, “Types of resistors,” August 23, 2009, http://dsaprojects.110mb.com/electronics/data_book/resistors.html.

[19] R. Aiken, “Resistor types – does it matter?,” August 3, 2009, http://www.aikenamps.com/ResistorNoise.htm.

[20] Analog Devices, “Op amp power supply rejection ratio (PSRR) and supply voltages,” February 10, 2009, http://www.analog.com/static/imported-files/tutorials/MT-043.pdf.

[21] “Designing active filter circuits,” class notes for ENSC 425, School of Engineering Science, Simon Fraser University, Summer 2009.

[22] A. Kay, “Analysis and measurement of intrinsic noise in op amp circuits part 2,” October 29, 2006, http://www.analogzone.com/avt_1023.pdf.

[23] Actel, Appl. Note AC204, pp. 1-7.

[24] Signal Consulting Inc., “Ferrite beads,” October 12, 2000, http://www.signalintegrity.com /Pubs/edn/ferritebeads.htm.

[25] “Lecture 24,” class notes for ENSC 380, School of Engineering Science, Simon Fraser University, Spring 2008.

[26] S. Haykin and M. Moher, *Introduction to Analog and Digital Communications.* Hoboken: John Wiley & Sons, 2007, pp. 198-199.

[27] Analog Devices, “AD603,” May 21, 2007, http://www.analog.com/static/imported-files/data_sheets/ad603.pdf.

[28] Linear Technology, “LT1630/LT1631,” November 19, 2002, http://cds.linear.com/docs/Datasheet/16301fs.pdf.

[29] Seng Piel Audio, “Damping of sound level with distance,” n.d., http://www.sengpielaudio.com/calculator-distance.htm.

[30] A. Kay, “Analysis and measurement of intrinsic noise in op amp circuits part 3,” April 26, 2007, http://www.analogzone.com/avt_1204.pdf.

[31] Fairchild Semiconductor, “CD4051BC/CD4052BC/CD4053BC,” April 2002, http://www.fairchildsemi.com/ds/CD/CD4051BC.pdf.

[32] N. Neretti, N. Intrator, and L.N. Cooper, “ Improved noise tolerance for sonar applications in critical environments,” in *IEEE International Conference in Computational Intelligence for Homeland Security and Personal* Safety, April 1, 2005, pp. 94-96.

[33] Magnode, “Aluminum properties,” n.d., http://www.magnode.com/magnode/extrusion expert-terms.html#.

[34] Duroboat, “Aluminum boats and corrosion,” 2006, http://www.duroboat.com/experience/AluminumBoatsandCorrossion.htm.

[35] OnlineMetals.com, “Aluminum bare sheet 5052 H32,” n.d., http://www.onlinemetals.com/merchant.cfm?pid=4786&step=4&showunits=inches&id=240&top_cat=60.

[36] J. Salazar, J. Yoon, B. Azage, T. Whiteman, and S. Ohr, “Waterproof cellular phone design,” final report for EE15N, Stanford University, San Francisco, CA, USA, January 22, 2008.

10. APPENDIX

10.1 URL TRANSDUCER DRIVER CIRCUIT

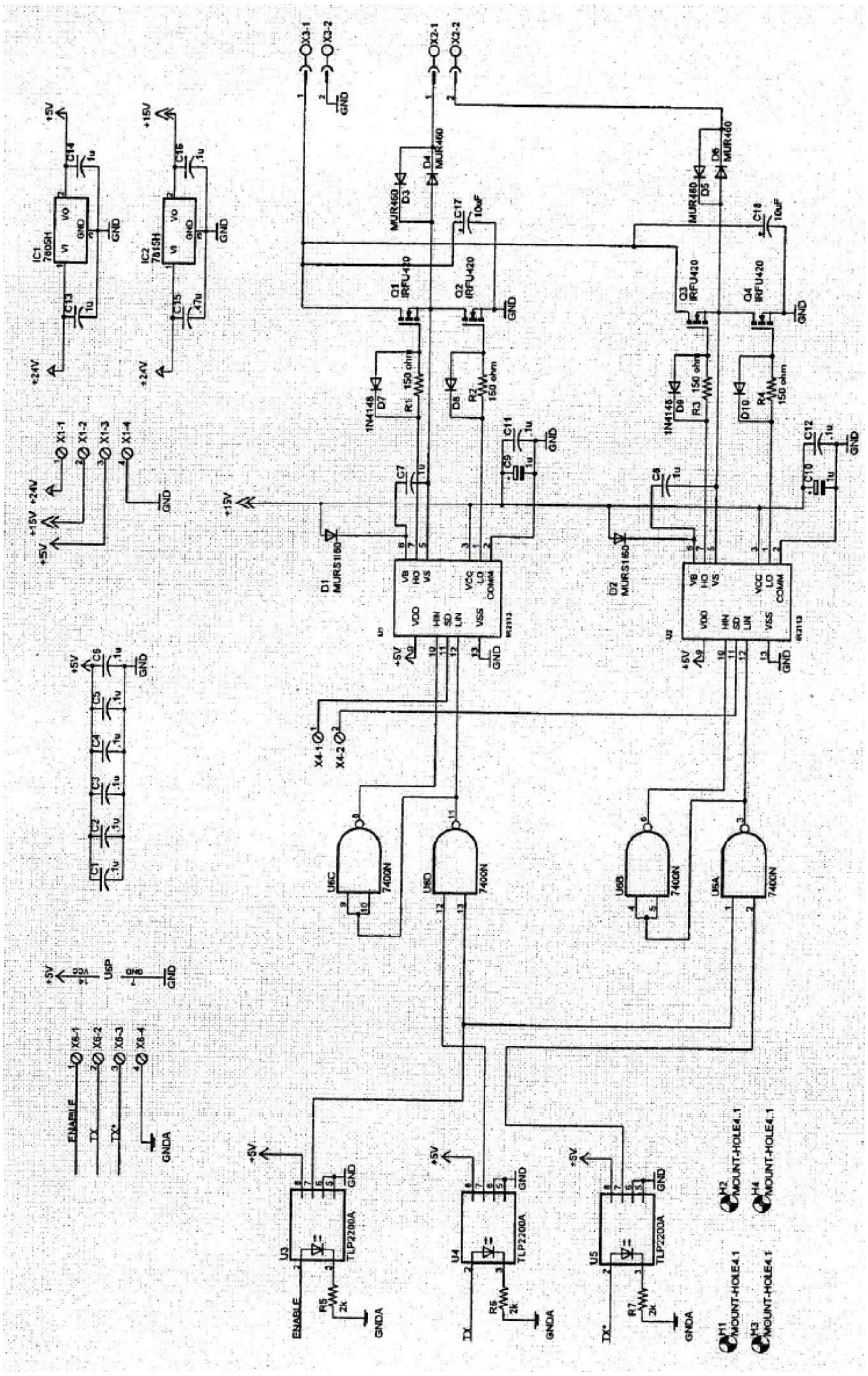


Figure 26: Schematic of URL Transducer Driver Circuit