UNIVERSITY OF OSLO
Physics Institute

Development of a system for mapping the sound field in shallow water

Master thesis

Adam Olson

1. August 2011

Contents

Preface

With the finishing of this thesis, I have concluded two challenging years of living in Norway. I am thankful that the Norwegian government is so generous to allow foreign students like myself to learn at its universities without requiring tuition.

The main aim of this thesis is to further develop a project that has been worked on by a few masters students. The project never has been ready to make measurements in the sound field, but it has served as a place to focus technical ideas and practices for the students. My thesis material has been a continuation of this project, and I hope that the work serves future students well. I tried to gain a better understanding of hydroacoustics and how shallow water sound fields behave, while also taking on some engineering challenges.

I had almost no knowledge of hydroacoustics before beginning this thesis, but my advisor, Dr. Helge Balk, was kind to offer a semester course to myself and two fellow students. Although the original goal of this thesis project was to take sound field measurements, I appreciate that Helge allowed me to direct my thesis towards my own interests in electronics. His suggestion to design a hydrophone driver let me visit a lot of new ideas that I had not learned before. I thank Helge for offering his broad knowledge of electronics and hydroacoustics, as well as his wisdom and conventionality to keep me focused in the project.

I also would like to thank two fellow grad students, Morten Huseby and Johan K. Jensen, who were always helpful to collaborate with and were always enthusiastic to introduce everything about Norway to me.

Lastly, I want to thank some people who helped me to take my journey to Norway. I am grateful to my parents, Chris and Beth, and brother, Eric, for supporting me across the ocean whenever I needed them, as well as Dr. Brian D. Huggins and Thomas J. O'Donnell who both helped me so much before I came to Norway.

Blindern, 1 August 2011

Chapter 1

Introduction

1.1 Motivation

Hydroacoustics has a wide range of applications. For most of the last century, many of the applications were oriented towards activity in the ocean or in deep lakes. In these environments, the boundaries of the body of water are so large that many times the sound does not interact with them between a source and a target. Reflections can be much weaker from the sea surface and sea bottom with the large distances, and refraction of the sound can be predicted with speed-depth profiles and ray-tracing. When the boundaries of the body of water are constrained to shallow water, the interaction of the sound waves are less predictable. Temperature gradients from the water surface to the floor can vary greatly over a few meters, and the ray-tracing models do not always predict how the sound travels through these environemnts. For this reason, a closer examination of how sound is propagating in the sound-field in these shallow bodies is desired to gain a better understanding.

An experiment had been devised to measure the sound-field. In the experiment, a transmitting transducer projector is to be placed on a stand that varies the height of the hydrophone in the water. At each height, the transducer must send out a pulse that propagates across the soundfield, and this pulse is then measured at a known distance away by a stationary hydrophone. This measurement is stored and corresponds to the height that the pulse was sent from. The process is repeated by lowering the transducer by a known increment of height each time until pulses have been sent from the top of the stand near the water surface to the bottom of the stand near the floor. The pulses can then be analyzed to find echos and reflections to see where the sound had traveled at each height. The experiment setup requires customized equipment for such a system.

1.2 Background

This project is a continuation of previous masters projects to develop this system for measuring and characterizing the sound field. Design decisions from the previous masters projects are carried over, but also improvements and new ideas are explored to continue the growth of the project. At times, questions were asked why the previous projects made some decisions, such as the sampling rate for the acoustic signal, and as a result, a new topic to study was pursued as in chapter 2 The close relationship of the hydroacoustics and electronics disciplines means that a lot of the work in this project is more related to electronics than actual hydroacoustics. An overview of the system is shown in figure 1.1. This general overview has several variations.

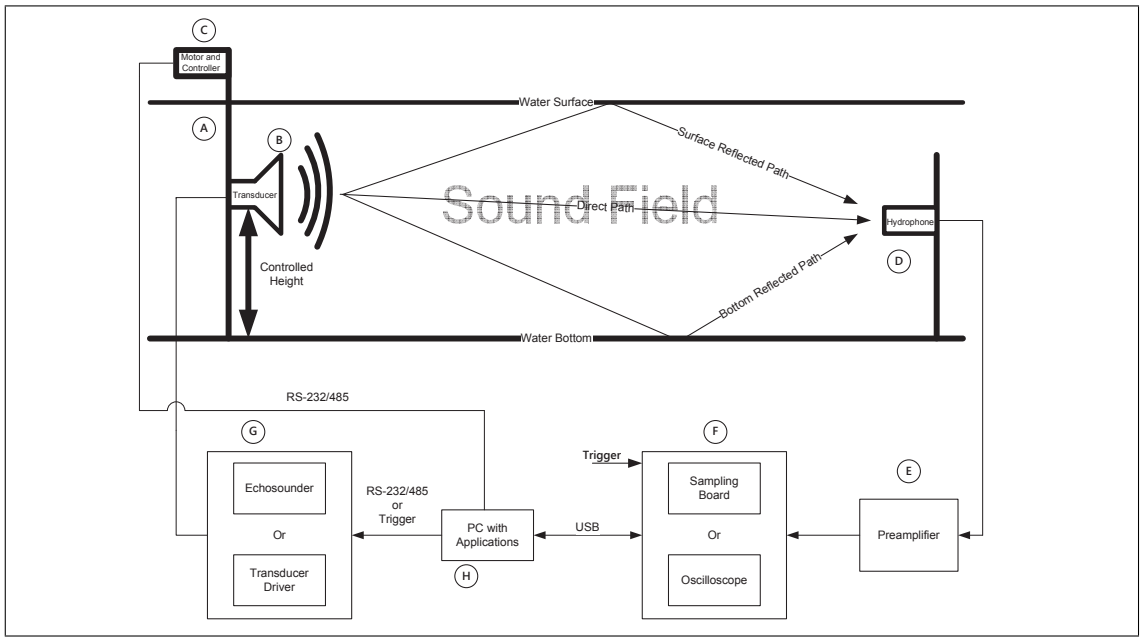


Figure 1.1: Overview of the system components needed to measure the sound field.

The components of the figure are described below:

- A. The transducer positioning stand that consists of a worm gear which lifts a platform across two rails. A motor turns the worm gear which controls the position of the platform.
- B. The transmitting transducer, the Simrad ES 120-4x10
- C. The motor and its controller that controls the position of the platform.
- D. The hydrophone that receives the transmitted acoustic signal and its echos. The device is a Reson TS4034.
- E. VP1000 preamplifier that adjusts the gain of the signal so it is measureable, also has a selectable high-pass filter.

- F. Sampling system to convert the hydophone voltage to a digital signal and send onto a PC. There are two methods considered in the thesis. The samplying system will be sent a trigger with a "travel time" delay so it is turned on as the sound ray arrives.
- G. Transducer driver. A transducer driver is designed in the thesis but not built, so an already existing EY500 must be used. The transducer driver will be sent a trigger when it is to transmit a signal.
- H. A laptop or PC that has the motor and aquisiton applications installed to control the system. Stores the transducer measurements in a file.

The tasks of this thesis are:

- Research signal analysis of the experiment and sampling requirements for shallow water reflections and echos.
- Select a BLDC motor and design a controller to vary the height of the transmitting transducer in the water.
- Design high voltage drive electronics to drive a 500W hydrophone.
- Find a method for measuring the reflections to be analyzed on a PC.

Chapter 2

The Acoustic Signal

2.1 Introduction

I needed to characterize the acoustic signal so that I could determine the specifications needed for measuring the signal. The bandwidth of the analog-to-digial converter as well as the data transfer rate and memory requirements of the data aquisition module all depend on the nature of the signal that will be measured. Also, an understanding of the signal properties is useful for later analysis of the sound-field measurements.

When measuring the sound-field of a shallow body of water, the received acoustic signal at the hydrophone will be made of the original transmitted signal that is sent by the transducer as well as reflected echoes and reverberation. These two components to be measured are shown in figure 2.1. These reflected echoes will be similar in waveform to the original signal aside from a few factors that change their amplitude and phase, and so they can be difficult to pick out if they arrive at the hydrophone while the transmitted signal is still hitting the hydrophone. The goal of the experiment is to measure the echoes in the signal in relation to the transmitted signal because they give information on where the sound is traveling and how it is being reflected in the body of water. When referring to the transmitted signal and the echo signals in this chapter, the transmitted signal is the direct sound ray as it travels from the transducer to the hydrophone. The echos are the indirect sound rays as the direct sound ray bounces off of the water surface, water bottom, or any objects between the transmitting transducer and the hydrophone before reaching the hydrophone. Echos in this chapter should not be confused with echos bouncing off a target and back to the transmitting transducer as is the case in other echo applications.

The main characteristics of the echoes that could be useful are:

- The time delay of when the echo reaches the hydrophone after the original signal

has reached the hydrophone: this can be used to estimate the time and distance that the echo has traveled.

- The phase of the echo with respect to the original signal: this phase information will tell where the echo came from, since reflection from the water surface has a different acoustic impedance than from the bottom of the water floor. The acoustic impedance determines if there will be a phase shift in the signal, and for the water surface there is while the bottom floor does not have one.
- The amplitude of the echo: this will give information of the attenuation the echo has received as it passed through the sound-field.

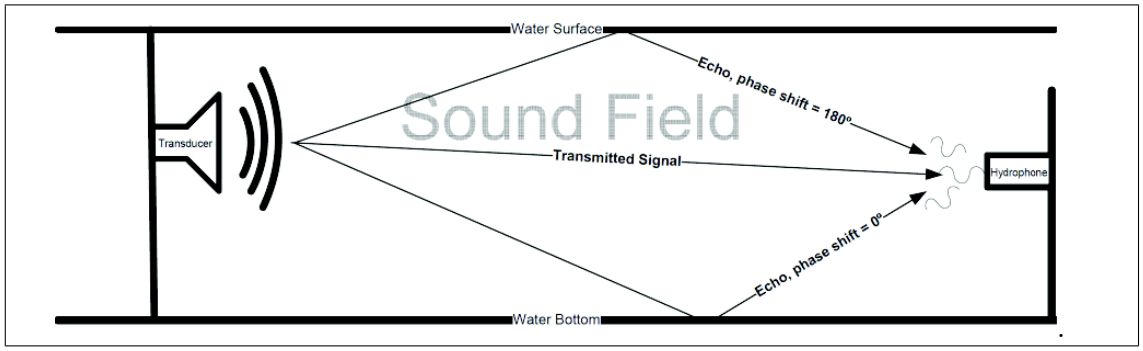


Figure 2.1: A diagram of a simplified scenario of a transmitted signal arriving at the hydrophone, and its echos generated by the sound-field arriving shortly after.

The echo signals will have time-delays associated with them since they take a longer time to reach the hydrophone because of the difference in travel path and distance. The time to distance relationship is simply determined by the speed of sound in the water c and the distances that the direct and echo signals traveled, r_{direct} and r_{echo}, as expressed in equation 2.1. This travel time delay will create an impulse change in the received signal at the time that the echo arrives at the hydrophone, and so the time that this impulse is measured tells the extra distance the echo traveled.

$$r_{echo} = (c)(t_{delay}) + r_{direct} \tag{2.1}$$

Similarly, if the echo is the result of a reflection of the original signal from a medium with a lower acoustic impedance than the water, the echo will have an additional phase shift of 180 degrees that also gives information of where the echo came from.

Also, the echoes will have attenuation factors that are different from the original signal. There are a few reasons for this difference. First, the reflected echos will be attenuated more as a function of the extra distance that they traveled. The amount of intensity lost by the signal as it passes through the sound field is called the transmission loss, TL, and the attenuation of the medium is given by equation 2.2 from Urick [24][p. 111], where the absorption factor α is in units of dB/m, and r is in units of meters. The absorption factor is dependent on the shear viscosity of the water, where

energy is lost from the movement of the water molecules, and also dependent on the salinity of the water from a process of sulfate ionic relaxation [24, p. 104-108]. Both of these absorptions transfer the acoustic energy into heat energy.

$$TL = \underbrace{20log(r)}_{\text{spherical spreading loss}} + \underbrace{\alpha r}_{\text{absorption loss}} \tag{2.2}$$

The other type of transmission loss in equation 2.2 is called the spreading loss. As the sound wave travels further from the source(assuming an omnidirectional source) and expands unbounded, it is spreading its energy radially over a larger area. Since intensity is defined as the acoustic power per a unit area, and the energy is conserved, the same amount of original power must be spread across a larger area as the sound wave propagates and so the intensity decreases by a factor of $\frac{1}{r^2}$ for spherical spreading.

Also, as the original signal is reflected from the water surface or the water bottom, the reflection coefficient, $\mathcal{R}$, determined by the angle of incidence and acoustic impedances between a two medium interface will determine how much energy is reflected in the echo. A reflection coefficient of less than 1 will result in lost energy and attenuation of the signal. If an echo experiences one or more reflections, equation 2.2 can be modified to add the reflection losses in equation 2.3, where N is the number of reflections an echo experienced and n is an instance of a reflection. For example, if an echo bounces from the water surface, travels to the water bottom, and then bounce back up to the hydrophone, it will have experienced two reflections, each with its own reflection coefficient. This summation accounts for the number of reflections that an echo may experience before it reaches the hydrophone, and this only applies to the echoes; not the transmitted signal.

$$TL = \underbrace{20log(r)}_{\text{spherical spreading loss}} + \underbrace{\alpha r}_{\text{absorption loss}} - \underbrace{\sum_{n=1}^{N} 10log\left(\mathcal{R}_n\right)}_{reflectioncoefficientlosses} \tag{2.3}$$

The attenuation from the absorption factor will be greater for the reflected signals since they travel longer distances than the direct signal. However, the distance of the experiment will be much less than a kilometer, and also the lower salinity of freshwater means that the absorption factor α is most likely negligible, which can be seen in figure 5.2 from Urich [24, p. 104], where the 120kHz frequency has an absorption factor of a little over 1dB per 914 meters distance traveled in distilled water. Thus, the transmission loss will be mostly a function of spreading loss and reflection losses, but even so, refraction of the sound wave can be considerable in shallow water with large temperature gradients, and so the spreading loss will not be perfectly spherical.

So by measuring and extracting the time delays and measuring their phases shifts and attenuation, we can then begin to trace back and make calculations of where these reflected echos have traveled in the sound-field. With the more variables available to be

measured (time delay, phase shift, attenuation), the echos paths can be more accurately determined. The variations in these echos can be correlated to the position of the the transducer in the water column, which will help determine where in the water column that the sound field is experiencing echo creation and the direction these echos are taking. With a carrier frequency of $120kHz$, the sound waves sent out have an approximate wavelength of 12.5mm, depending on the depth-velocity profile of the water. To sample a wavelength spatially requires the same Nyquist sampling rate concept, so that it must be sampled at a height resolution of at least twice this height of 6.25mm. By sampling at this high resolution, the waves can be more closely approximated as rays, which allows the data to be compared to the ray tracing models that try to predict their behavior.

The signal received at the hydrophone can be analyzed in both the time-domain and the frequency-domain. In the time domain, peak detection can be used to find the time delay of an echo. One such method is performed using the cross-correlation of two measurements. The maximum peak of the cross correlation between these measurements gives the estimated time delay [7]. Frequency-domain analysis can help determine how effective the time-domain methods will be depending on the bandwidth of the equipment used to measure the signal. Also, the information of time delay and phase change of echoes can be found with frequency-domain methods, where the sound-field is treated as a comb filter for each time delayed echo. These ideas are discussed in the next three sections. In this discussion, the assumption is made that, when looking at these signals, non-linear effects of the acoustics are ignored. This means that as the signal propagates through the water, new frequencies being created are neglected, which may ultimately have a role.

2.2 Time Domain

The signal transmitted by the transmitting transducer is a 120KHz pulse train that lasts for 0.1ms, which results in 12 pulsed cycles. Because the transducer has a resonance frequency at 120KHz from figure 4.4 and attenuates other frequencies like a band-pass filter, the acoustic signal can be approximated as a sinusoidal pulse. The signal is generated with no DC bias as would be the . A DC bias is relevant with regards to the electrical signal and its ground reference. As the piezoelectric behavior of the hydrophone only responds to AC voltage, there can be no DC bias in the acoustic signal. The signal can be considered as an amplitude modulated pulse.

This transmitted signal is expressed in equation 2.4 and its waveform is in figure 2.2. Where $\Pi(t)$ is a unit pulse function, τ is the pulse width of the pulse component, A is the amplitude of the sine component, and f_0 is the frequency of the sine wave. The sine wave is expressed in its complex form from Euler's theorem to facilitate fourier transform calculations with phasors in section 2.3. The signal can be represented either in its

electrical or acoustic form, but units are ignored for now.

$$x_{transmitted}(t) = \underbrace{\Pi\left(\frac{t-\frac{\tau}{2}}{\tau}\right)}_{x_{pulse}} \underbrace{\left[A\frac{e^{j(2\pi f_0 t)} - e^{-j(2\pi f_0 t)}}{j2}\right]}_{x_{sine}} \tag{2.4}$$

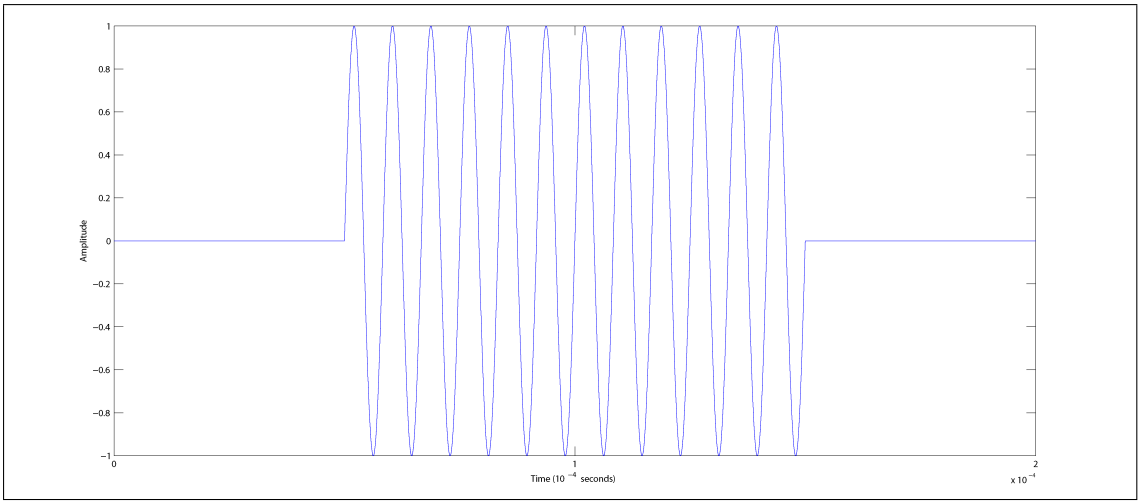


Figure 2.2: The time domain of the original transmitted signal, a 0.1ms wide pulse modulated by a 120kHz sine wave. Note: the pulse begins at time t = 0.05ms.

Similarly, the reflection echo is given in equation 2.5, where α is an attenuation factor, t_0 is the time delay of the signal, and θ is the possible phase change of the echo depending if there was a phase change upon reflection from the water surface.

$$x_{echo}(t) = \alpha x_{transmitted}(t - t_0)$$
or
$$x_{echo}(t) = \alpha\Pi\left(\frac{t-\frac{\tau}{2}-t_0}{\tau}\right)\left[A\frac{e^{j(2\pi f(t-t_0)+\theta)} - e^{-j(2\pi f(t-t_0)+\theta)}}{j2}\right] \tag{2.5}$$

And when these two signals meet at the hydrophone, the received signal will be the superposition of the transmitted and echo signals as shown in equation 2.6. In a real sound-field, there will be multiple echoes from different reflections, so in general the received signal can be described in equation 2.7 with x_n being an instance of an echo with an optional phase shift and with N being the number of echoes in the soundfield. In the project's experiment to measure the sound-field, the n, N, α_n, and t_n parameters will have unique values at each height of the hydrophone projector.

Plots of this superposition of a transmitted and an echo signal are shown in figure 2.3 and figure 2.4, with a time delay $t_0 = 0.0191ms$, and with an attenuation factor of $\alpha = 0.7$ as an example. In figure 2.3 the echo has no phase change. We see that when

the pulse envelope of the second echo begins, there is a sharp change in amplitude. In figure 2.4, a phase change $\theta = 180°$ is used to simulate a reflection from the water surface, and we see that there is a change, but the sharp change is hard to find visually.

$$x_{received}(t) = x_{transmitted}(t) + x_{echo}(t) \tag{2.6}$$

or in the general case

$$x_{received}(t) = x_{transmitted}(t) + \sum_{n=1}^{N} x_n(t - t_n) \tag{2.7}$$

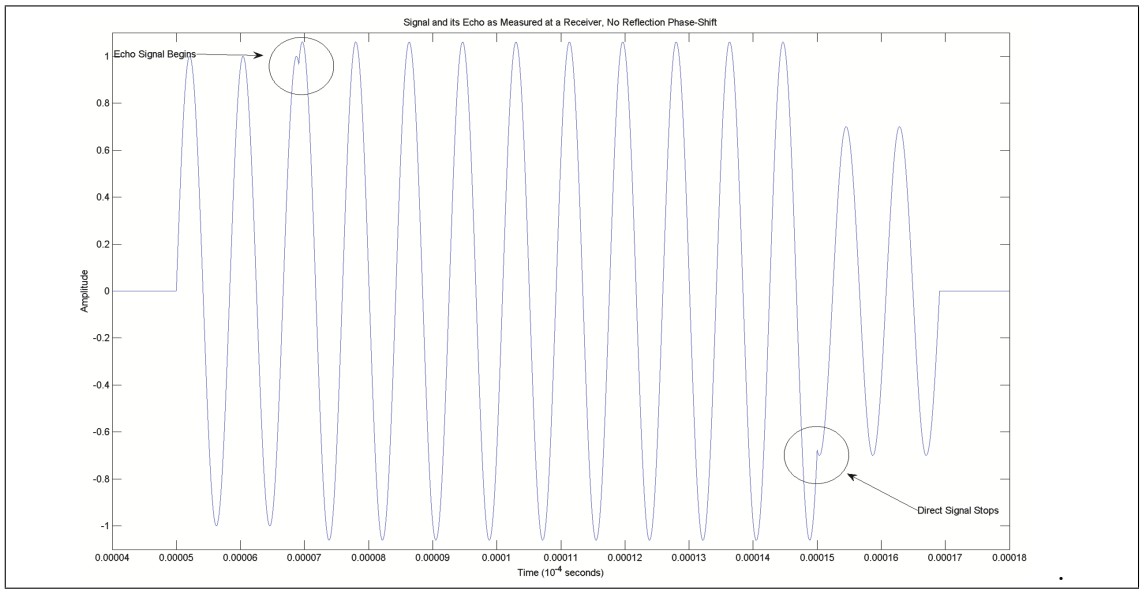


Figure 2.3: The received signal as a superposition of the transmitted and echo signals. The transmitted signal and the echo are in phase. The received signal shows sudden changes in amplitude from the time delay when the echo is added.

A closer look at the received signal at the point in time when the echo is added to the transmitted signal is shown in figure 2.5. The time between the first peak and the second peak should be measured so that a double peak can actually be detected. If the sampling frequency is too low, this time can be skipped over and only one peak will be detected. The datapoints on figure 2.5 can give an estimate on the bandwidth required to successfully capture this time delay. By approximating the drop in signal between the two peaks as a pulse and using the the time-bandwidth product, a rule-of-thumb minimum bandwidth in this instance is given in equation 2.8. This time-bandwidth product is different in each case, depending on when the echo arrives and what its phase is, and so it cannot be relied on to choose the appropriate sampling rate for the system. Also, the necessary bandwidth can be lower since a perfect reconstruction is not required to find a peak. The peak will be smaller if less bandwidth is used, but if the amplitude resolution is high enough it can still detect the smaller peaks. If the resolution of the

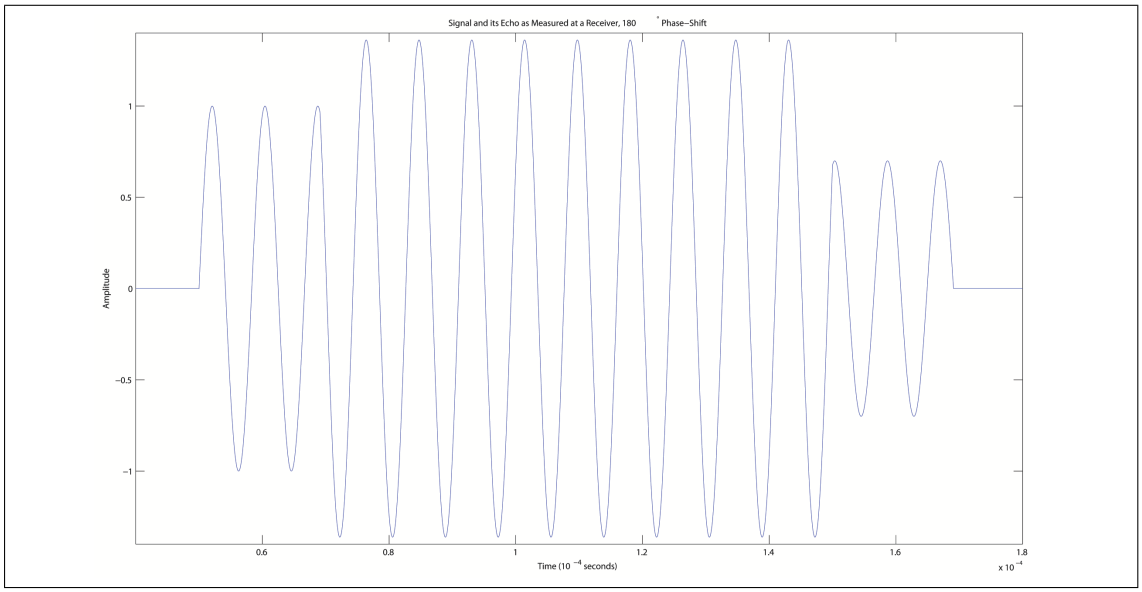


Figure 2.4: The received signal as a superposition of the transmitted and echo signals. The echo has a 180 degree phase shift to simulate a reflection from the water surface. The waveform appears smooth even though there is a sudden change in amplitude from the pulse envelope.

ADC is not enough or the noise is too large, then the maximum bandwidth is necessary to see these peaks. At some time delay values, there will not even be a second peak if the echo hits the hydrophone at a time when the original signal is not near a peak, like was shown in figure 2.4. This makes it even more difficult to accurately find the time delay. As was mentioned earlier, this is why cross-correlation techniques are used to detect the peaks when analyzing the system in the time domain.

$$\Delta f \geq \frac{1}{0.45\mu s} = 2.2MHz \tag{2.8}$$

2.3 Frequency Domain

While the time-domain description can give insight into where phase change and echo time delays occur in the signal, the spectra of the signal can yield more information, and besides this, it is important to know the relevant frequencies of the signal when designing a digital sampling system.

To find the frequency content of the signal, the fourier transform of the transmitted and echo signals in equation 2.9 are found separately, and, since the transform is linear and superposition holds, they can then be added together. The case of only one echo is considered, but the transform will be generalized for multiple echoes later. The variables used to describe the signals are the same as in equations 2.4 and 2.5. In the echo signal, θ can be modified for any value that the echo may have shifted from the original signal,

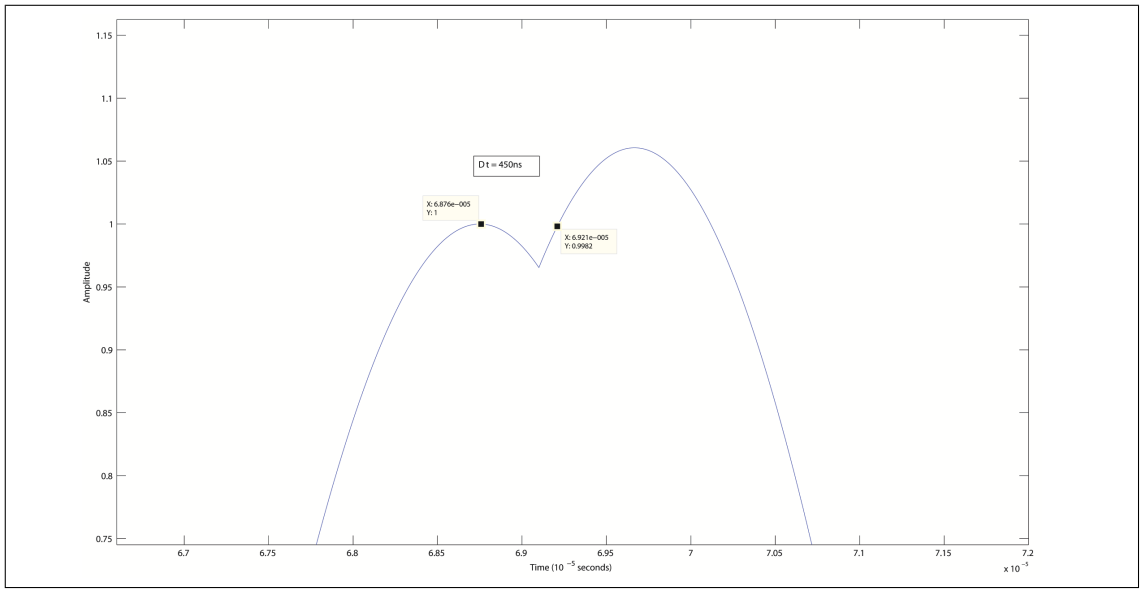


Figure 2.5: The received signal at the time that the echo arrives. This creates a secondary peak that must be measured to see the phase information in the time-domain. The time width of this peak gives an estimation for the bandwidth required to measure it.

but it will have a 180° phase shift if reflected from the water surface. Also, for the echo signal of equation 2.11, the attenuation factor, α and time delay t_0 are applied.

$$X_{received}(f) = X_{transmitted}(f) + X_{echo}(f) \tag{2.9}$$

where

$$X_{transmitted}(f) = \int_{-\infty}^{\infty} \left[\underbrace{\Pi\left(\frac{t-\frac{\tau}{2}}{\tau}\right)}_{x_{pulse}} \underbrace{\left[A\frac{e^{j(2\pi f_0 t)} - e^{-j(2\pi f_0 t)}}{j2} \right]}_{x_{sine}} \right] e^{-j2\pi ft} dt \tag{2.10}$$

and by the time delay theorem [27, p. 158]:

$$X_{echo}(f) = X_{transmitted}(f)\alpha e^{-j2\pi f t_0} e^{j\theta} \tag{2.11}$$

To solve the transform of equation 2.10, first the pulse and sine functions are transformed seperately, and by use of the multiplication theorem [27, p. 158] the transform becomes the convolution of the two spectra.

$$X_{transmitted}(f) = X_{pulse}(f) * X_{sine}(f) \tag{2.12}$$

X_{pulse} is found by equation 2.14 and X_{sine} is found by equation 2.17. The convolution of the pulse spectra with the sine wave's impulse functions results in the frequency

translation theorem [27, p. 158] where the pulse's spectra is shifted by the sine wave's frequency as seen in equation 2.18.

$$X_{pulse}(f) = \int_0^{\tau} 1e^{-j2\pi ft}dt \tag{2.13}$$

$$= \tau sinc(2\tau f) - j\tau^2\pi f sinc^2(\tau f) \tag{2.14}$$

$$X_{sine}(f) = \int_{-\infty}^{\infty} A\frac{e^{j(2\pi f_0 t)} - e^{-j(2\pi f_0 t)}}{j2}e^{-j2\pi ft}dt \tag{2.15}$$

where the phase shift can be taken out of the integral as a constant

$$X_{sine}(f) = A\int_{-\infty}^{\infty}\left[\frac{e^{j2\pi f_0 t}}{j2}e^{-j2\pi ft}\right]dt - A\int_{-\infty}^{\infty}\left[\frac{e^{-j2\pi f_0 t}}{j2}e^{-j2\pi ft}\right]dt \tag{2.16}$$

$$= \frac{A}{j2}\delta(f - f_0) - \frac{A}{j2}\delta(f + f_0) \tag{2.17}$$

$$X_{transmitted}(f) = \left[A\left(\frac{\tau sinc\left(2\tau(f - f_0)\right) - j\tau^2\pi(f - f_0)sinc^2\left(\tau(f - f_0)\right)}{j2}\right)\right.$$
$$\left. - A\left(\frac{\tau sinc\left(2\tau(f + f_0)\right) - j\tau^2\pi(f + f_0)sinc^2\left(\tau(f + f_0)\right)}{j2}\right)\right] \tag{2.18}$$

The magnitude plot of the $X_{transmitted}(f)$ spectra is shown graphically in figure 2.6, where $\tau = 0.1ms$ and $f_0 = 120kHz$. As can be seen in the figure, most of the energy in the signal is present around the modulated $120kHz$ center frequency and its side lobes. The lobes repeat every $\frac{1}{\tau}Hz$. The spectra is infinite because the pulse is finite in time, but the lobes drop in energy siginicantly at each successive lobe, and so bandlimiting the signal at an appropriate frequency should not remove much information from the signal. It was questioned if the time delay and phase shift of the echo were contributing to the sharp change of the received signal in figure 2.5 and if this sharp change added more frequency components, but when the magnitude of the echo spectra was found, it had exactly the same frequency components of the transmitted spectra, so it does not make sense to sample at a bandwidth higher than the bandwidth of where most of the transmitted signal's power is at . It is only necessary to have, ideally, infinite bandwidth if we wished to capture the sharp change of the pulse envelope in the time domain. So, the question is, if the time delay and phase changes of the echo are not shown as extra frequency components, then where are they found in the spectra?

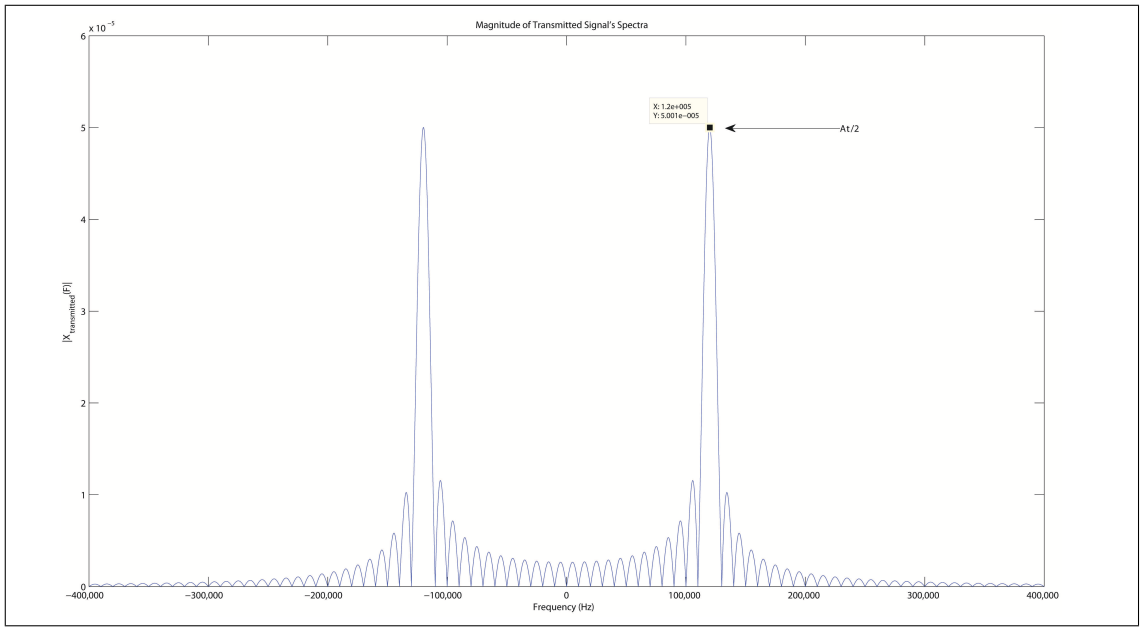


Figure 2.6: Magnitude plot of the the transmitted spectra. The pulse sinc function has been shifted to the modulation frequency.

Even though the magnitudes of the transmitted and echo spectra are the same, their phases are actually different. Where the phase changes periodically over the entire spectra. The phase of the transmitted signal spectra is shown in figure 2.7, where it is fit to show only the positive spectrum. The phase changes between -2π rad and 2π rad at $\frac{1}{\tau}$ intervals, and there is a change in sign of the phase at the center frequency of $120kHz$.

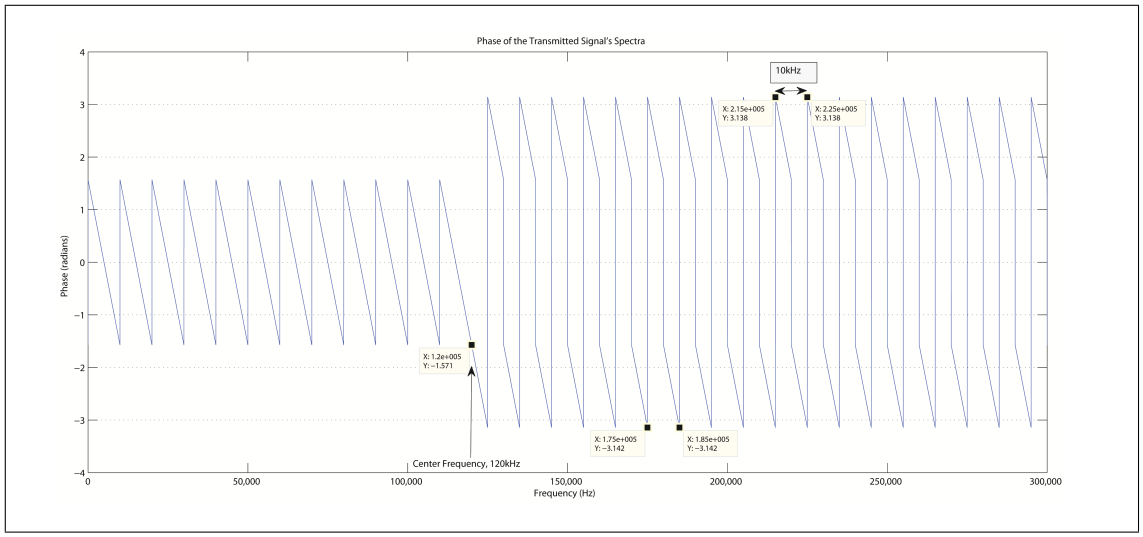


Figure 2.7: Phase plot of the spectra of the transmitted signal

When phase changes and time delays are introduced into the echos, the phase information changes, but the phase is still periodic over the spectrum. For example, by

observing the phase plots of echos in figure 2.8, the time delay is varied and we see that the phase is 0° at a period equal to $\frac{1}{2t_0}$, remembering that t_0 is the time delay. So, as the time delay becomes smaller, the phase information of this time delay is spread out further across the spectrum. If the spectra of the echo is bandwidth limited to a maximum of BW, then the minimum time delay that has all of its phase information inside this bandwidth is given in equation 2.19. Note this is only considering the case of one time delay. This periodicity in the frequency spectrum is an important idea for section 2.4.

$$t_{0_{min}} \geq \frac{1}{BW} \tag{2.19}$$

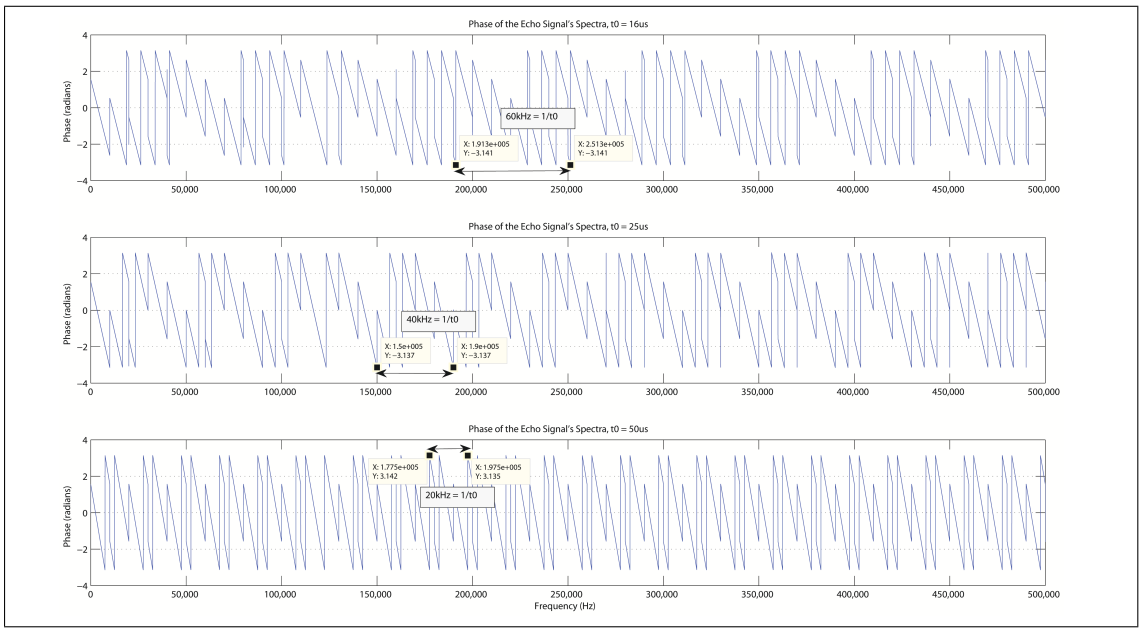


Figure 2.8: Phase plots of the spectra of the echo signal with different time delays for each plot. The frequency measurements show how much bandwidth is between frequencies of 0° phase.

From this observation, we can see in equation 2.20 that the minimum distance difference between the transmitted signal and its echo that they can travel, r_{min}, before time delay information is lost and thus the abilitiy to differentiate between the two signals is limited by the bandwidth, BW, where c is the speed of the wave in the water. For example, if the bandwidth is limited to 500kHz, the minimum distance that an echo must travel to be distinguished as a distinct impulse is 3mm further than the transmitted signal traveled. If the echo travel distance is less than 3mm longer than the transmitted travel distance, the received signal will look like a single transmitted signal with a larger amplitude.

$$r_{min} = t_{0_{min}} c = \frac{c}{BW} \tag{2.20}$$

2.4 Comb Filtering and the Cepstrum

2.4.1 The Filtering Effect at the Receiving Hydrophone

In acoustics and electronics, a comb filter is a type of filter that has a periodic attenuation. It can be thought of as having many passbands and stopbands across a spectrum. The name comb filter comes from the idea that the magnitude response looks like a hair comb, where each tooth selects the its frequency band, and the frequencies between the teeth are rejected. The comb filter effect is generated by a time delayed copy of an input signal added onto the original input signal with no time delay. We will see that the echos created in the sound-field are acting as copies of the original signal with time delays, and that they are being added onto the transmitted signal.

When the spectra of the transmitted and the echo signal are added together, as is the case when the two signals arrive at the hydrophone, the magnitude of this received signal's spectra, $X_{received}(f)$, has a filtering effect compared to the original sinc function. A plot of this is shown in figure 2.9, where the echo was given a time delay of $t_0 = 0.02ms$, an attenuation factor of 1, and no phase change from surface reflection.

If we compare figure 2.9 to the spectra of just the transmitted signal, we see that adding the echo has created an interference pattern on the transmitted signal. The soundfield is creating these echos, and so the sound field is the actual filter acting on the transmitted signal. To show that the sound-field is behaving as a filter, the expression for the received signal of equation 2.9 can be arranged using the echo's equivalent expression of equation 2.11 to show the transfer function of the sound-field in equation 2.21. The general case for multiple echos is given in equation 2.22 where N is the number of echos that the sound-field produced at a particular height and n is each instance of an echo.

$$\frac{X_{received}(f)}{X_{transmitted}(f)} = 1 + \alpha e^{-j2\pi f t_0} e^{j\theta} \tag{2.21}$$

$$\frac{X_{received}(f)}{X_{transmitted}(f)} = 1 + \sum_{n=1}^{N} \alpha_n e^{-j2\pi f t_{0_n}} e^{j\theta_n} \tag{2.22}$$

What is left, after the transmitted spectra has been divided out of the received spectra, is a periodic waveform dependent on frequency and the time delay of the echo. This is shown graphically in figure 2.10 by dividing out the transmitted spectra from figure 2.9 and what is left is the filter spectra of the sound-field. The periodicity of the filter is equal to the time delay of the echo. This comb filter also includes the phase shift information that the sound channel introduces as waves are reflected off of the water surface. This phase simply shifts the phase of the comb filter, as seen in figure 2.11. So, with this comb filter, the time delay and phase of the echo can be extracted quite easily just from visual inspection; however, this becomes difficult when there are multiple echos and the filtering is now a superposition of comb filters. There are a few methods to extract the time delay and phase of these more complex cases which take advantage of the periodicity of the filters.

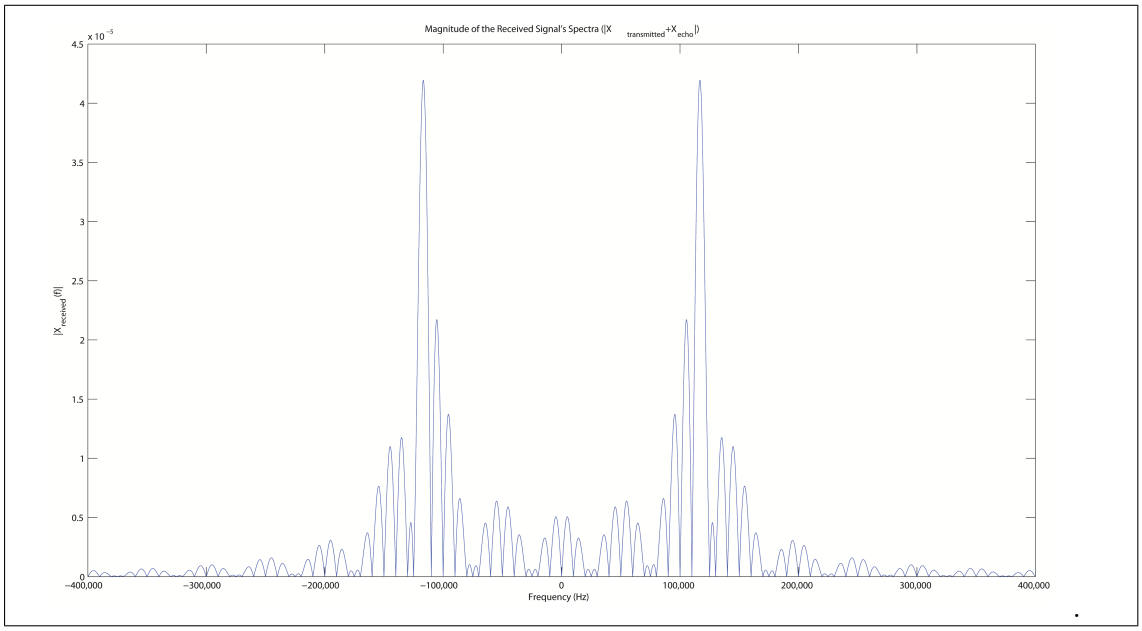


Figure 2.9: Magnitude plot of the received signal's spectra at the hydrophone. The signal shows that the transmitted signal has been filtered as it passed through the sound channel by the echos that the sound-channel creates.

2.4.2 Inverse Fourier Transform Technique

One method to find the time delays contained within the comb filter is simply to take the inverse fourier transform of the comb filter. The inverse fourier transform of the comb filter is given by equation 2.23 which uses the $\delta(t - t_0) \leftrightarrow e^{-j2\pi f t_0}$ transform pair [27, p. 171]. As the bandwidth becomes limited, these delta functions become spread out over the time axis. The equation is the impulse response of the sound field, and it shows that the time delays of the echos are given as impulses over time, and they have phase information attached to them as well. If a signal is measured with a digital sampling system, its DFT(discrete fourier transform) must be taken to analyze the signal's spectra, and then a model of the transmitted signal's spectra must be divided out of this DFT, which gives the comb filter's impulse response. Then, the IDFT(inverse DFT) of the comb filter spectra is found. The absolute value of the IDFT gives the impulses at the time delays, and the arg function of the IDFT, gives the phase angles of the filter. This process is outlined in figure 2.12.

$$\mathcal{F}^{-1}\left(1 + \alpha e^{-j2\pi f t_0} e^{j\theta}\right) = \delta(0) + \alpha\delta(t - t_0)e^{j\theta} \tag{2.23}$$

A simple example was made to verify that this technique is possible and to see what the results would look like. The spectrum of a comb filter was defined in MATLAB by 3 echos in equation 2.24, each with a different time delay. The first and third echos have a 180° phase shift to simulate reflections from the water surface, and the second echo has an attenuation factor of $\alpha = 0.5$. The inverse fourier transform of this spectra was

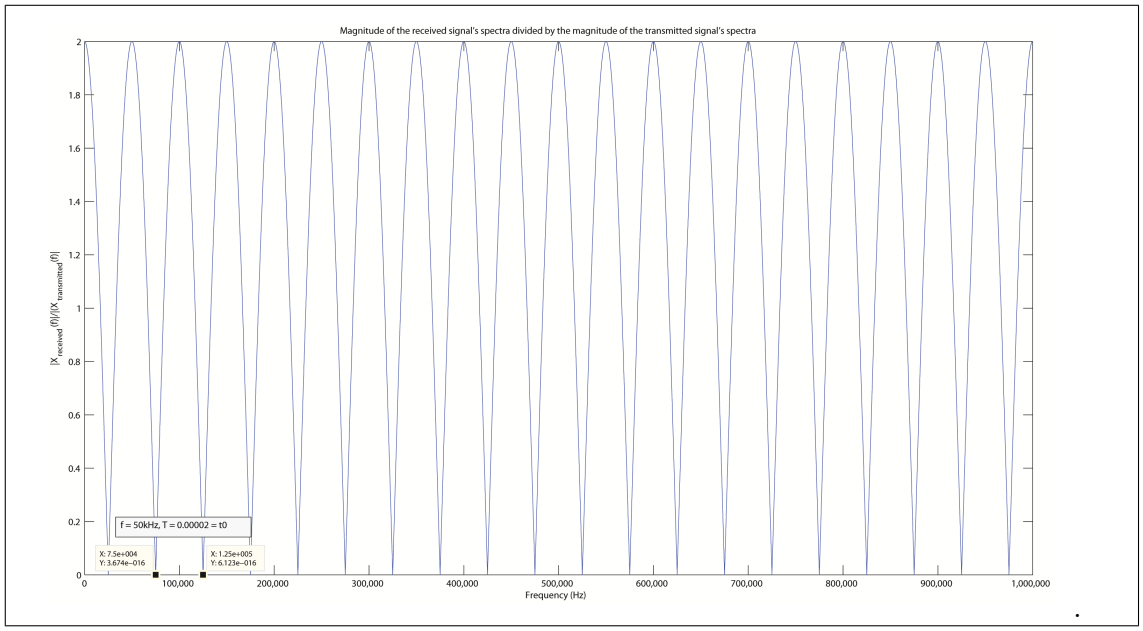


Figure 2.10: The comb filter spectra of the sound channel. The periodicity that the comb filter repeats is the inverse of the the time delay of the echo that created the filter effect.

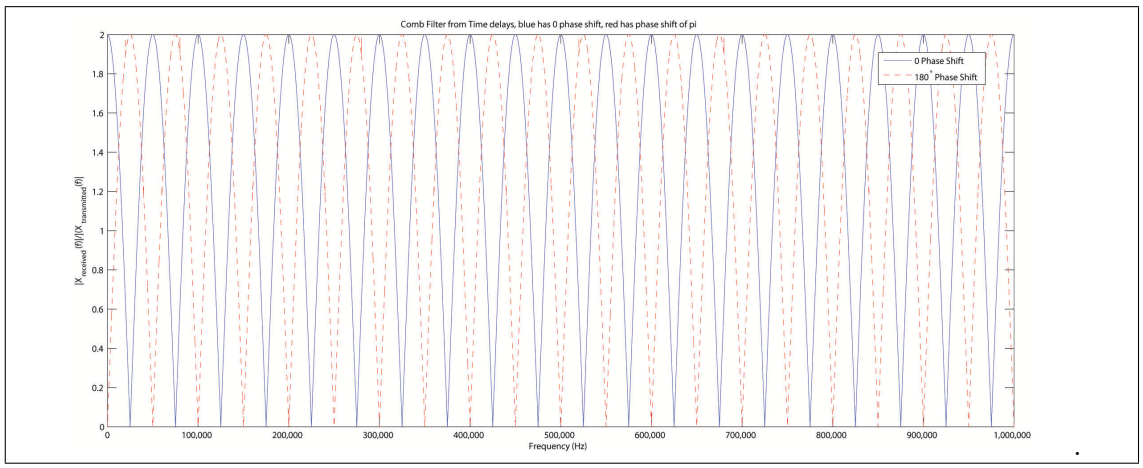


Figure 2.11: Two comb filter spectra overlayed on eachother. The solid line spectra is from an echo with no phase change, while the dashed line is from an echo with a phase change of π. This shows that not only the time delays, but also the phase changes are also contained within the sound channel filter.

taken using the ifft() function in MATLAB. The absolute value was taken and plotted in figure 2.13 and the phase was taken and plotted in figure 2.14. We see that this process reveals the time delays in both figures, where the height of the impulses represents the attenuation factor, and their position along the time axis shows how much time the echo is delayed by. Similarly, the phase changes between 0 and π at each positon along the time axis that the echos arrive.

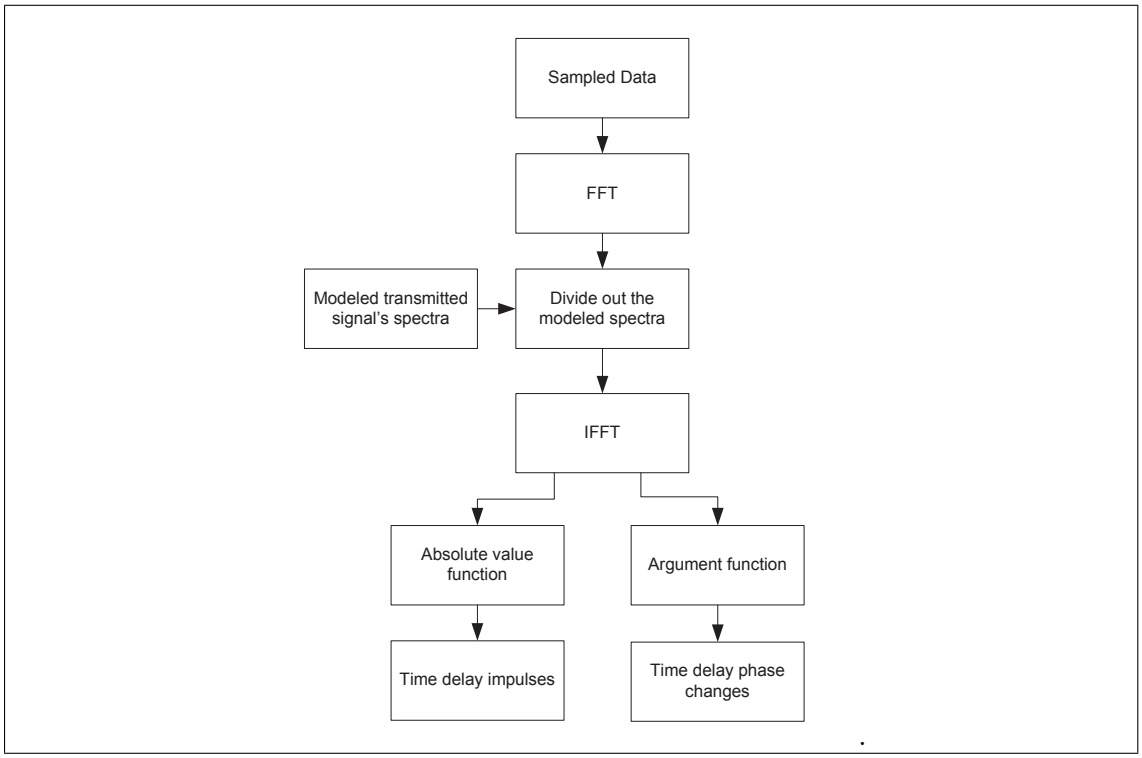


Figure 2.12: Process for finding the time delays and phase information of the echos by using the IFFT

$$X_{comb}(f) = 1 + e^{-j2\pi f(.00002)}e^{j\pi} + 0.5e^{-j2\pi f(.00005)}e^{j(0)} + e^{-j2\pi f(.00009)}e^{j\pi} \tag{2.24}$$

The main disadvantage of this technique is that an accurate model of the transmitted signal must be modeled in order for it to be extracted from the received signal. This could be easily done for the experiment under consideration if the source level and transmission loss of the acoustic signal are known, since they can be used to modify equation 2.18. Another practicality would be to align and scale this modeled spectra to the FFT data sets, but this is possible to automate in a program. If this process is applied to cases where the transmitted signal is unknown or modeled very poorly, it would not work at all. This disadvantage is avoided by the other technique which uses a tool called the cepstrum.

2.4.3 Cepstrum Technique

The cepstrum technique was originally developed for characterizing seismic echos by Bogart, Healy, and Tukey [4], but it is now applied to many fields where echos are either desired or must be removed such as in speech processing, acoustic restoration, and noise canceling [10]. The name cepstrum is just a play on words of spectrum because the process to get the cepstrum is very similar to getting the spectrum. The concept is the same as the previous technique, which is to extract the periodicity of the filtering effects to

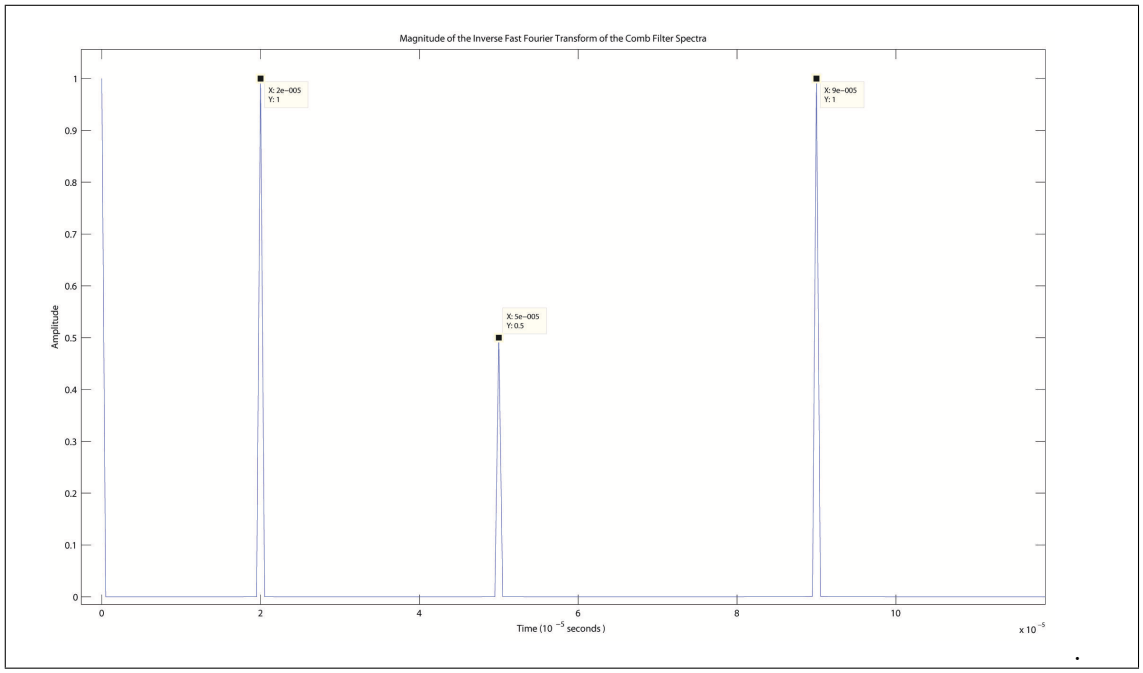


Figure 2.13: The impulses of the comb filter in the time domain after the transmitted spectra was divided out and the ifft of the spectra was taken. The amplitudes are proportional to the attenuation that the echo experienced as it traveled through the sound channel. Each impulse represents a time delayed echo.

determine the time delays of the echos. Unlike the previous technique, this method does not require a modeling of the original signal's spectra. Instead, it assumes the original signal's spectra is not periodic or at least not as strongly as the time delay's comb filter periodicity, and so it takes a second fourier transform of the first fourier transform to show where the strongest periodicity lies in the frequency spectrum, and this is called the cepstrum. There are different cepstrum approaches, where just the real information or the complex information can be transformed. Only the complex approaches preserve the phase information, and so, for this application, the complex cepstrum should be used so that the reflection phase changes can be found.

The complex cepstrum is defined by equation 2.25 [10], where $\mathcal{F}(x(t))$ represents the fourier transform of the received signal.

$$C(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[log\left(\mathcal{F}(x(t))\right) + jarg\left(\mathcal{F}(x(t))\right) \right] e^{j\omega n} d\omega \tag{2.25}$$

2.4.4 Conclusion

This chapter gives a good summary of why this experiment is useful for characterizing the sound-field. When we want to find the ray paths that the sound takes through the

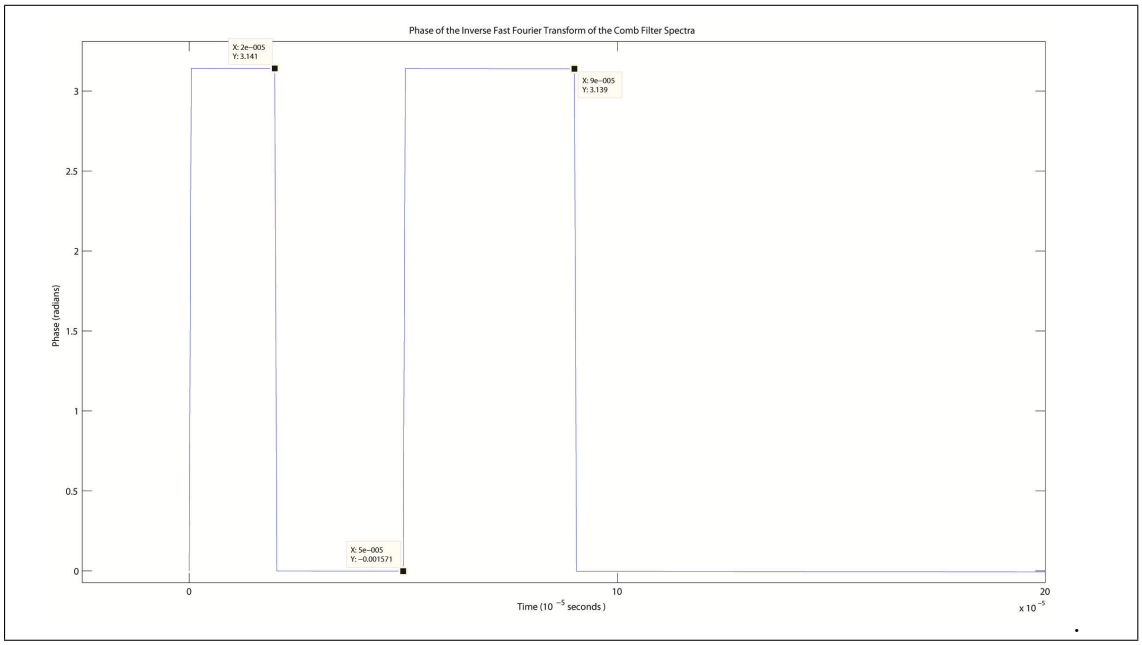


Figure 2.14: The phase shifts of the impulse response in the time domain. The phase changes at each time a new echo arrives that has a different phase than the current phase.

sound field, the time delay, phase information, and amplitude all give clues as to where the sound has traveled.

The question of what the sampling rate should be to measure the signal was the driving question for analying the signal and its spectra. By looking at the spectra plots, and by using equation 2.19 and equation 2.20, a safe estimate of the bandwidth needed is chosen as 500kHz, which includes the main lobe and the first 37 side lobes. So, the system should sample the received signal at a minimum of 1MHz by the Nyquist sampling theorem. For this reason, any part of the spectra of the received signal that is past 500kHz should be filtered out to eliminate aliasing. If any sampling rate is chosen higher than 500kHz, it will only be capturing the extra lobes of the signal with very low energy, and will not give much extra information.

Analysis of the signal not only provides the important frequencies in the received signal, but it also leads to interesting spectra analysis tools with the comb filtering and cepstrum technique. We see that the time delays are creating interference patterns in the spectra of the transmitted signal, and these patterns are periodic across the spectrum. These methods could be useful for analyzing the data that the experiment will collect.

Chapter 3

BLDC Motor Controller

3.1 Introduction

At the beginning of the project, the plan was to use the motor control system developed by Halvor Strøm in his thesis. His method used a stepper motor with an off-the-shelf motor driver, and he built a control circuit and PC interface to send PWM and control signals to the motor driver. With his help, I was able to get his equipment and software functioning, but I found there were some drawbacks to the implementation.

First, unlike most motors, a stepper motor's position can be commanded very accurately in open-loop. The previous stepper motor had no rotary encoder or other feedback sensor, and so it was run in open-loop. However, there is no guarantee that the control program commanding the position can know the motor's true position without some form of feedback. Errors in the motor position can result from inherent properties of the motor described in Halvor's thesis [21] such as torque dead zones,gear slip, and the rotor's momentum.

The other drawback to using this stepper motor was that it was very slow and created a lot of grinding noise and vibration or slip in the motor. I did not investigate the cause of these grinding noises and vibrations, but reducing the frequency of the PWM signal helped to remove some of this at the cost of lowering the top speed of the motor. The motor took approximately 40 minutes to move the transducer platform from the top of the stand to the bottom while operating at the maximum stepping frequency of 1300Hz.

Because the equipment will be measuring bodies of water outdoors, environmental variables in the body of water can change over the course of a few hours, such as water temperature, and this could effect the measurements of the sound-field. Thus, taking more than one measurement sample at each height in the water would take hours to complete and could be vulnerable to unwanted variables in the experiment.

I designed a new motor system to eliminate these drawbacks, which would gaurantee more reliable position control and reduce the time needed to move the transducer platform. I chose a motor with higher torque and RPM capabilities so that it could move the platform much more quickly. Since the system is to operate remotely outdoors, I chose a DC motor so that I could power it with batteries. I chose to use a BLDC motor because it offers higher torque than a brushed DC motor at the same size. It is also more efficient and reliable since there is no commutation brush to create friction losses. There was also a motivation to learn BLDC motor technology because it has become a popular motor technology in many applications on both large and small scales, such as their use in modern electric vehicles, hard drives, controlled fans, and pumps.

3.2 BLDC Motor Basics

3.2.1 Construction and operation

A BLDC motor's physical design is actually closer to that of a permanent-magnet-syncronous-machine (PMSM) AC motor with multiphase windings than it is to a typical brushed DC motor. A simplified diagram of the components of a BLDC motor is given in figure 3.1. This figure also represents the motor chosen, because the QBL5704 has 4 poles and 3 windings distributed over 6 slots. The rotor of a BLDC motor is made of a ferromagnetic material with permanent magnet poles distributed around the rotor, where each pole is the opposite polarity of its adjacent poles. The air gap represents the location where the flux linkage* of the windings and of the rotor's permanent magnets interact with eachother. The rotor is usually made of a soft magnetic core to shape the flux field in the air gap so that it is radial and perpendicular to the rotor. The airgap is a factor for determining the behavior of the motor because it defines the overall permeability of the medium that the magnetic flux flows through, and so the reluctance (the reciprocal of permeability) of the motor should be minimized so that the flux can flow more freely, and this is done by making the air gap as small as possible.

The stator of the motor has the windings distributed around it, where each winding is wound in concentrated coils in the slots of the stator, and these coils operate as electromagnets when current is passed through them. Some literature refers to these coils in the stator slots as poles as well, since they have a magnetic pole when energized; however, when looking at the chosen motor's datasheet, it only refers to the rotor's magnetic poles which can cause confusion. The windings are distributed such that each phase is adjacent to the two other phases, and the phases in the figure have been labeled U,V,

*The term flux linkage, usually denoted λ, is commonly used in motor literature, but it is just the total flux that passes through a winding with many turns. If the individual turns of the winding are not uniform and do not have the same amount of flux through their area, then the flux linkage is a sum of the individual flux through each turn, and if all the turns have the same flux, then the flux linkage is just the flux that passes through the winding times the number of turns in the winding.

and W to match the naming of the phases in the motor datasheet.

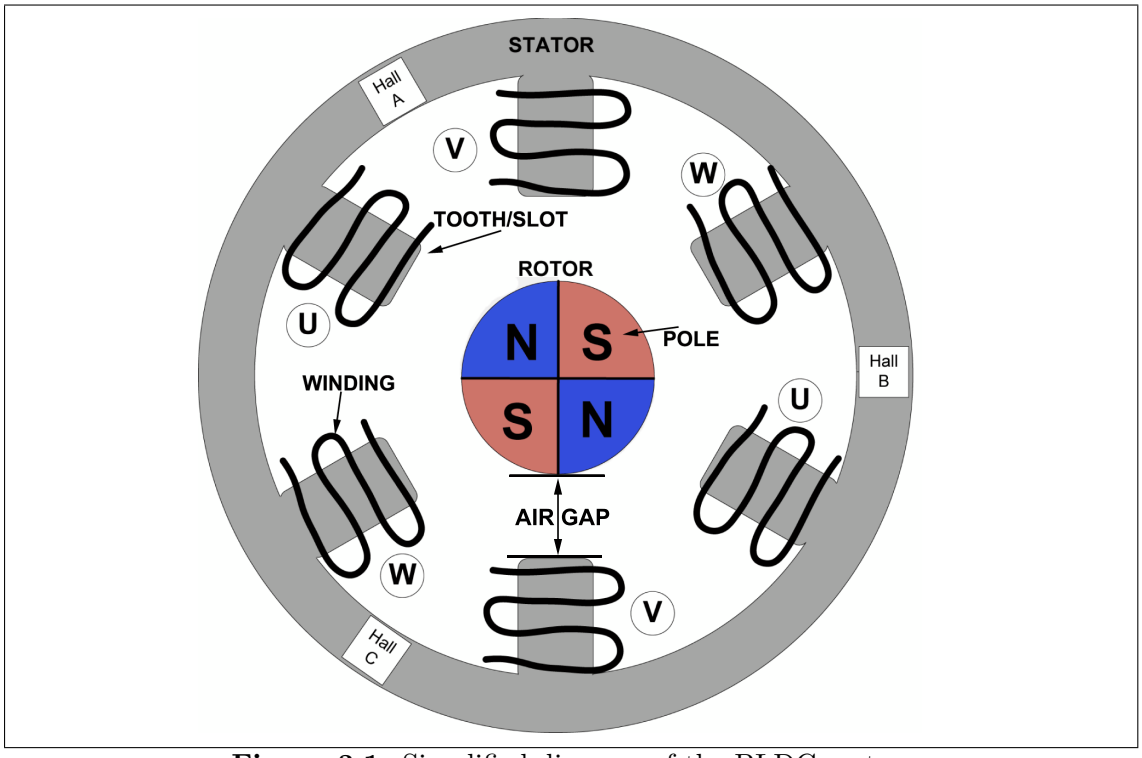


Figure 3.1: Simplified diagram of the BLDC motor.

When each winding of the motor is energized with a current, a magnetic field is created, and the sum of the windings' magnetic fields results in a magnetic field vector. As current is switched between the windings in a predetermined pattern, the magnetic field vector is rotated. This rotating flux field as a function of the winding currents is shown in figure 3.2. The rotating magnetic flux field interacts with the magnetic field of the rotor to produce a torque on the rotor. The goal of the commutation is to keep the windings' flux field perpendicular to the rotor's magnetic field, which ensures that all of the rotor's magnetic field's magnitude is in the vector component that cuts through the windings to give a maximum torque on the rotor, as described by the lorentz force on a current carrying wire in equation 3.1, where $\vec{F}$ is the force of the rotor field on the winding, L is the length of each turn in a winding that the magnetic field cuts through, N is the number of turns in the winding, and θ is the angle between the winding's flux field and the rotor's magnetic field.

$$\vec{F} = NLsin(\theta) \tag{3.1}$$

The number of electromagnetic windings depends on the number of phases of the motor, and so a 3-phase BLDC motor will have 3 windings. The windings are distributed over the number of slots that the stator has, and so with 6 slots, each winding will occupy 2 slots. The number of pole pairs(a north and south) is simply the number of

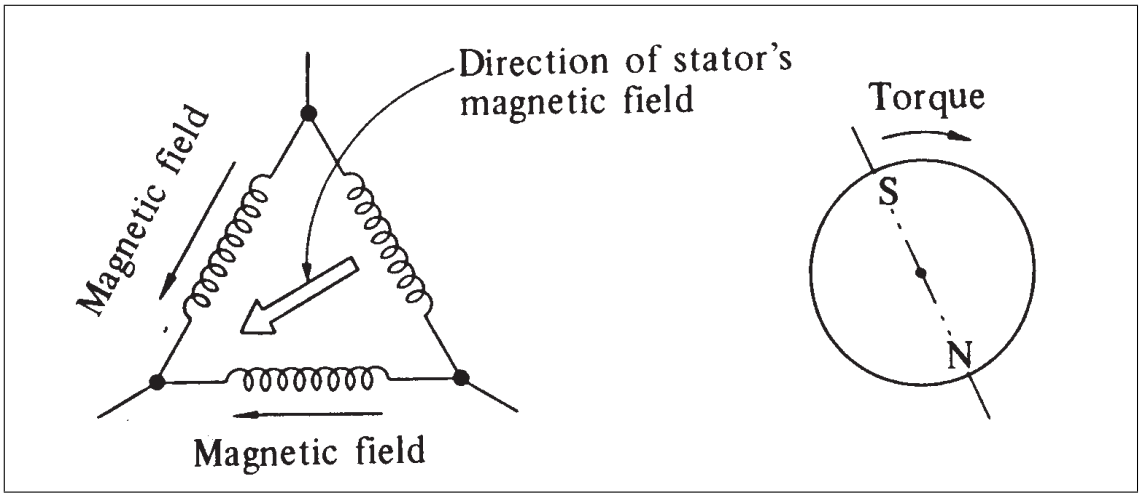


Figure 3.2: Rotating magnetic field vector created from the winding current commutations [14, p. 62]

poles divided by 2. If a motor has one pole pair, then its electrical angle and mechancial angle are the same. This means that one rotation of the magnetic field vector of the windings causes the rotor to make one complete revolution. As the number of pole-pairs is increased, the electrical angle must rotate a multiple number of time more to get one rotation in the rotor. If the pole-pair number is N, and the electrical and rotor angles are θ_e and θ_r, then their relationship is given in equation 3.2. This means that if the rotor has 2 pole-pairs, its controller must perform the entire commutation sequence twice to get one revolution.

$$\theta_e = N\theta_r \tag{3.2}$$

A BLDC motor is considered a synchronous motor because the magnetic field vector generated in the windings of the motor rotates at the same frequency as the rotation of the rotar's magnetic field, such that these fields are in sync with eachother. There are at least two fundamental distinctions between a BLDC motor and a similar AC permanent magnet syncronous machine (PMSM). First, the windings of a BLDC motor are usually distributed around the stator uniformally, while in a similar permanent magnet AC motor, the windings are distributed sinusoidally across the stator. This is shown in figure 3.3. These geometries determine how much of the magnetic field of the rotor crosses the windings at each position, where some positions have much more magnetic flux where the winding distribution is larger, and less flux where the distribution is lower. As the rotor is turning, this flux linkage through the winding is changing depending on this distribution, and by faraday's law, a change in the flux through a winding loop creates a back EMF voltage across the windings. The uniform distribution of a BLDC motor gives the windings a back-EMF that has a trapezoidal shaped waveform, while the sinusoidal distribution of an AC motor gives a sinusoidal back-EMF. The other difference between a BLDC motor and an AC motor is that the phases of the BLDC motor coils

are commutated electronically with a DC voltage source instead of being driven directly by a multiphase sinusoidal source.

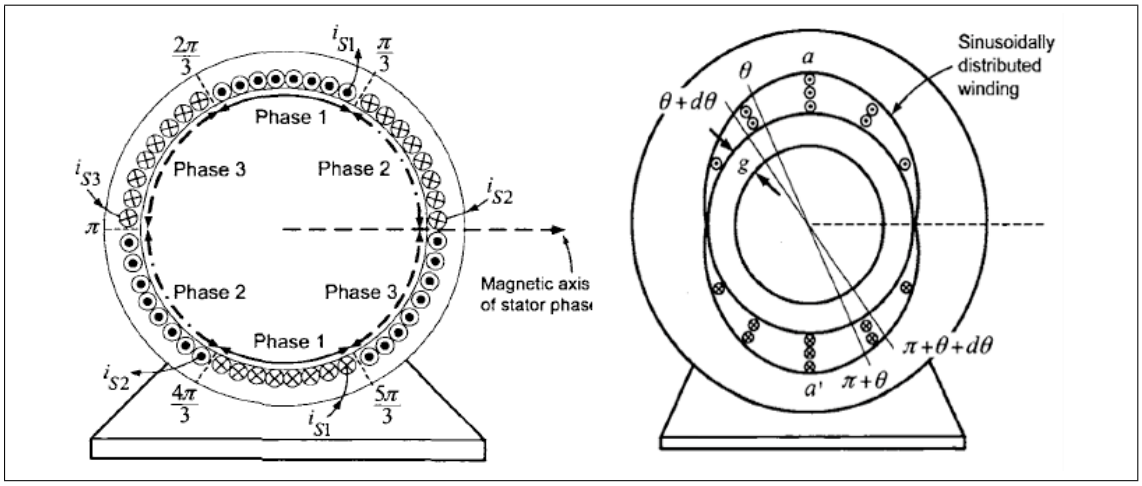


Figure 3.3: The motor on the left is a BLDC motor with uniform distribution of its windings, while the motor on the right is an AC motor with sinusoidal winding distribution. These give the motors their characteristic back-EMF waveforms.

The relationship between current and torque, and back-EMF and angular velocity are defined as constants K_t and K_v. Basically, when the motor is moving with a given current, its torque can be calculated with the K_t constant, and when the rotor is rotating at a specific angular velocity, the back-EMF in the windings can be calculated with the K_v constant. These relationships are what helps characterize the BLDC motor operation as equivalent to brushed motor operation, since both types of motors have these constants.

3.3 Commutation Strategy

As we saw, one of the main advantages of a BLDC motor compared to a brushed DC motor is that there is no physical commutator brush to supply current to armature windings(windings in the rotor) as there is on a brushed DC motor. Instead, the commutation is made in the stator electronically. As a consequence, the BLDC motor requires a more involved commutation strategy, and this is one of the challenges of designing a BLDC motor controller.

The goal is to generate torque in the motor and this is done by constantly adjusting the voltage into the windings to ensure that the stator flux of the windings is always at a 90° angle with the rotor flux of the permanent magnets. Because the flux is a function of the position of the rotor, it follows that the commutation of the windings must be synchronized with the changing position. A diagram of the commutation sequence is shown

in figure 3.4. This commutation sequence is often used in Y-connected motors, where one of the phases floats in each stage, but it can also be used on a Δ-connected(delta) motor as shown in the figure. There are other possible commutation sequences for a Δ connected motor that offer better performance, but the one used has wider compatibility with Y-connected motors. In general cases, a motor controller can often drive either connected type motor with the same commutation sequence. As was noted in section 3.2.1, the changing rotor flux in the motor's windings as the rotor moves creates a back-EMF in the windings. The currents of figure 3.4 are plotted along with the back-EMF voltages for each phase of the delta-connected motor. This figure shows that between electrical angles of 60°, the current in each phase is a distinct value, and the back-EMF of the phases have a constant value except one phase is changing from high to low at any 60° sequence. These sequences must be synchronized with the position of the rotor at all times, and so position sensing is required.

There are two very common position sensing methods used in commutation implementations. The most simple and straightforward is using 3 hall effect sensors located in the motor at positions where the winding flux must be changed to keep the flux angles at 90°. These positions are usually at 120° spacings, but they can be spaced closer together if the electrical angle is smaller than the mechanical angle. The hall sensor is simple because the controller simply switches to the next winding combination when it detects a change from one of the hall sensors, and so it can be done easily in both hardware or software. However, the hall sensors add extra cost to the overall price of the motor, and they also reduce reliability since failure of the hall sensor will leave the motor controller inoperable. Another issue is that alignment of the hall sensors must be as close to perfect as possible. Without proper alignment, the commutations will not happen at the correct rotor position. This misalignments causes some phases of the windings to be energized for a longer time than they should be, while other phases are energized for shorter times than they need to be, where the difference between ideal and misalignment times can be considered as timing offsets. When these phases are not energized at the right times, torque ripple is generated in the motor, where the phases are actually opposing the momentum of the rotor during these small timing offsets. This can lead to less reliability, lower efficiency, and audible noise coming from the motor.

The second common position sensing method is called back-EMF sensing, often referred to as sensorless commutation because there is no "direct"[†] transducer required to convert the position sensing to a voltage. As seen in figure 3.5, the back-EMF waveforms of a commutated BLDC motor alternate between high and low, where one phase is always transitioning between these levels. The figure shows that these back-EMF slopes of each phase cross the 0V point only twice, during a rising and once during a falling, and so there are 6 distinct positions that a back-EMF voltage crosses the zero point. This is what one back-EMF strategy called zero-crossing detection uses. It senses the three

[†]The winding itself is acting as the transducer.

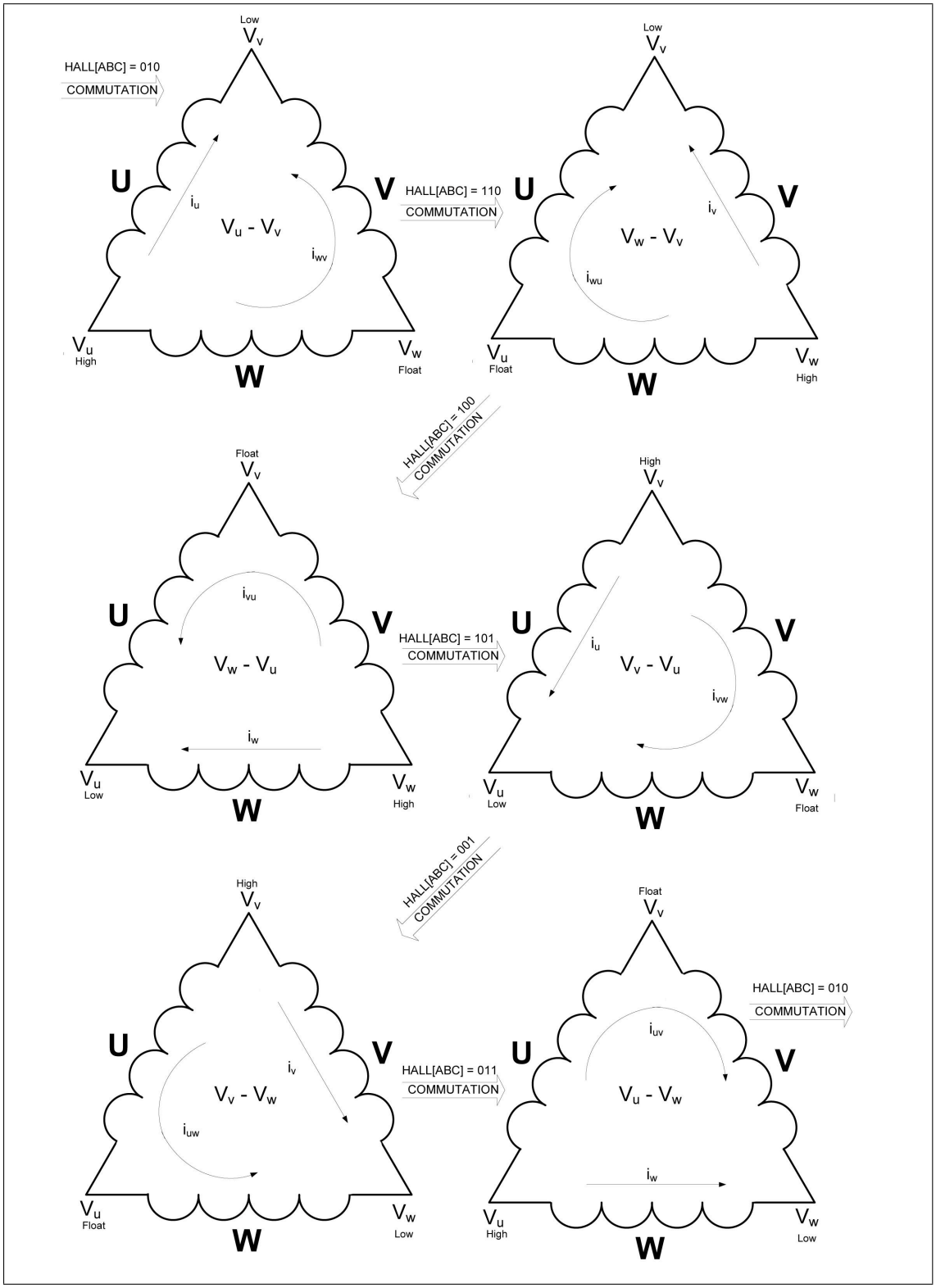


Figure 3.4: The commutation strategy used for a delta motor. If the electrical and motor angle are the same, each commutation moves the motor 60°. The arrows with hall combinations are valid for the final motor controller design discussed later.

phase voltages, and looks for when one of the phase voltages has crossed the zero-point, and knows that soon it must commutate to the next sequence. The zero-point crossing and the next commutation sequence are offset by 30°, and so the design of the back-EMF sensing commutation must compensate for this [9]. A back-EMF sensing method is more reliable and cheaper because fewer components are required, and it dominates many applications; however, there is a distinct disadvantage. Because the back-EMF is only generated when the rotor is moving, a stationary motor will have no back-EMF on any of the phases, and so the controller does not know which commutation sequence to use. As a result, a start-up sequence must be initialized to get the motor moving again, where the controller tries each commutation sequence and tests if the motor has begun moving, and then begins commutating from the successful sequence. Therefore, hall sensors are better suited for low-speed and positioning applications in the simplest designs. The motor that was selected comes with hall sensors, and because it will be positioning the transmitting transducer, which will be at rest most of the time, the hall sensors are a good fit for this application.

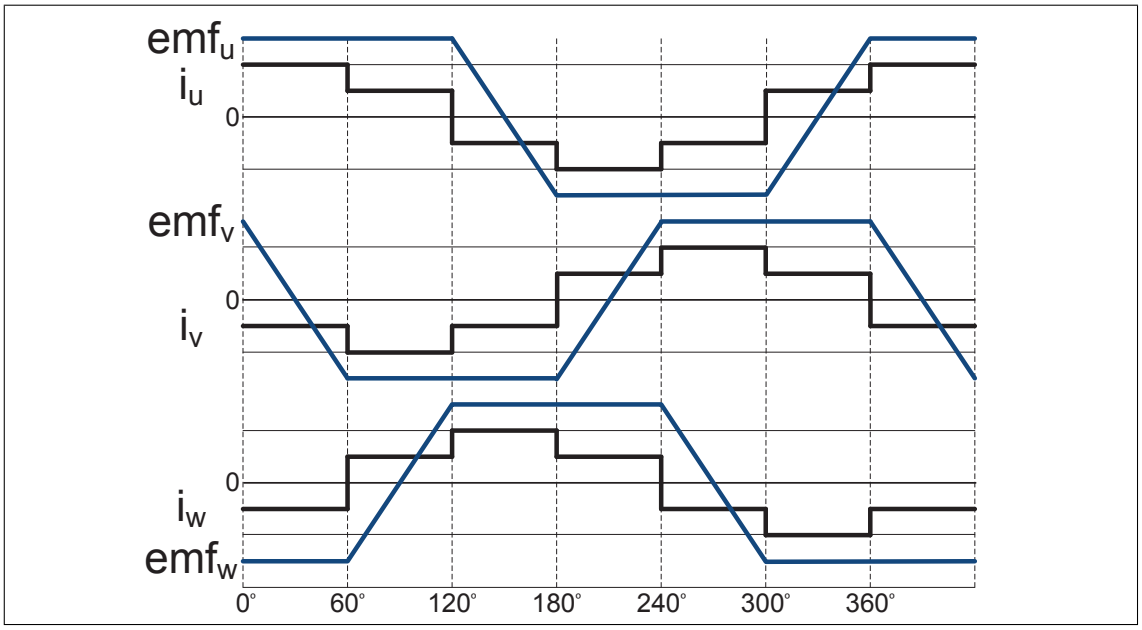


Figure 3.5: A plot of the current and back-EMF in each of the motor phases as it is commuted one electrical cycle.

With a known commutation sequence, the controller needs some way to actually control the voltage into each phase individually. The basic method to do this is to connect each phase of the motor to a half-bridge, as shown in figure 3.6. There are many application notes and literature that describe this method. Each half-bridge consists of a high-side MOSFET and a low-side MOSFET. The motor controller controls which gates are on at any time, and by choosing any combination of one high-side on and one low-side on, it is able to control the direction of current in each of the phases.

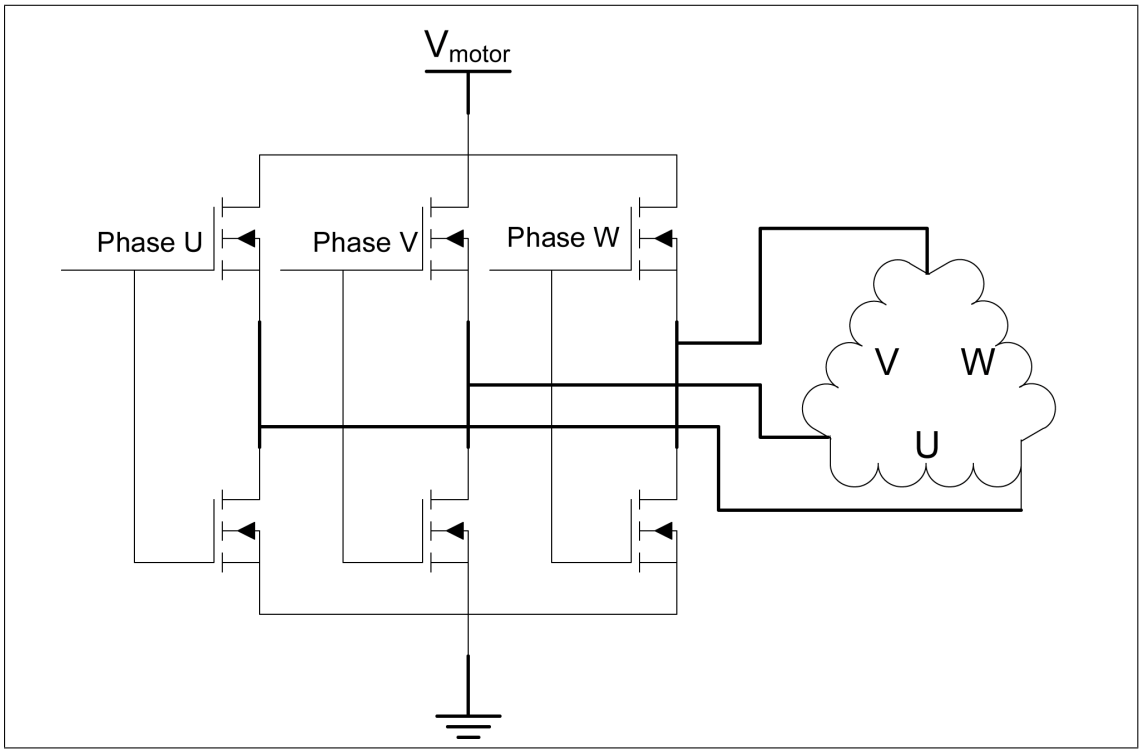


Figure 3.6: General 3 half-bridge circuit used to apply the commutations to the windings phases.

Turning on these half-bridge MOSFETs will connect the phases to the supply voltage V_{MOTOR} and ground and will allow control of the motor, but there is more that can be done with this setup to give even more control over how the motor operates. By varying the voltage to the MOSFET gates with a pulse-width-modulation(PWM) signal, the overall voltage into the winding phases will be the average of the time the MOSFET is on to the time it is off, known as the duty cycle, where the duty cycle(D) of the voltage to the phases can be defined in equation 3.3. The reason this method works is that the series resistance and inductance of the winding phases acts as a low pass filter with a time constant $\tau = \frac{R}{L}$ that filters out the PWM modulation frequency, and leaves only the time average DC voltage. The switching frequency should be chosen so that it is much higher than the bandwidth determined by this time constant. For small BLDC motors, their inductances are relatively larger, and so they prefer low PWM frequencies around 200Hz, while larger BLDC motors have smaller inductance values for their windings and can run optimally at PWM frequencies in the tens of kHz. So, by varying the voltage into the windings with a PWM signal, it follows that the velocity of the motor can be controlled, where the motor constant K_m gives the relationship between motor angular velocity and back-EMF present on the coils.

$$D = \frac{t_{on}}{t_{on} + t_{off}} \tag{3.3}$$

3.4 Motor Selection

It was important to select a motor that had a high enough RPM and torque rating necessary to move the transmitting transducer platform. It was difficult to measure the torque needed to move the platform at the desired RPM, so an indirect method was used to find the correct torque and RPM values. There was an available brushed DC motor at the lab, the Pittman 92365009, which has very detailed specifications in its datasheet. Among the specs were the motor's K_t and K_v constants. These constants can be used to measure how much torque the motor is applying to the transducer platform as well as how fast the motor is moving.

To make these measurements, the old stepper motor was removed from the clutch and the Pittman motor was attached. Voltage was applied to the motor and the current draw and voltage were measured. The voltage was increased until the transducer platform began moving at a much faster speed than what the stepper motor was driving it at. The Pittman motor was not rated for a high enough torque to consistently move the platform, and it was leaking fluid from being overstressed by this experiment, so it could not be used as the motor. When the platform was moving at an acceptable rate, the voltage in the Pittman motor was at 13.3V and it was drawing 2.6A of current. The torque constant of the Pittman motor K_t is 4.5x10^{-2}N-m/A, its back EMF constant K_v is 4.8V/1000RPM, and its armature resistance is 2.49Ω. So by using these constants, the required motor speed was found in equation 3.4. The voltage drop across the winding's resistance is subtracted from the input voltage to give the back-EMF voltage, and then this is divided by the K_m constant. The required torque was found in equation 3.5 by using the known current into the motor.

$$\frac{(13.3V - (2.5\Omega)(2.6A))}{.0048\frac{V}{RPM}} = 1417RPM \tag{3.4}$$

$$(4.5x10^{-2}\frac{N\text{-}m}{A})2.6A = 0.1099N\text{-}m \tag{3.5}$$

This torque value was also assumed to be the torque required to overcome the coulomb friction of the mechanical system, which is a constant value. The static friction required to start moving the transducer platform from rest was not considered although the motor would need a higher torque rating than what was measured for this case. Since the velocity of the transducer platform should move relatively constant, the viscous friction of the mechanical system was assumed to be a part of the coulomb friction.

These RPM and torque values guided the motor selection. A search at vendor websites showed that the cheapest motor that could drive these requirements was the QBL5704 from Trinamic. The velocity vs. torque curve of the motor is shown in figure 3.7 on the square marked line. An image of the motor is shown in figure 3.8

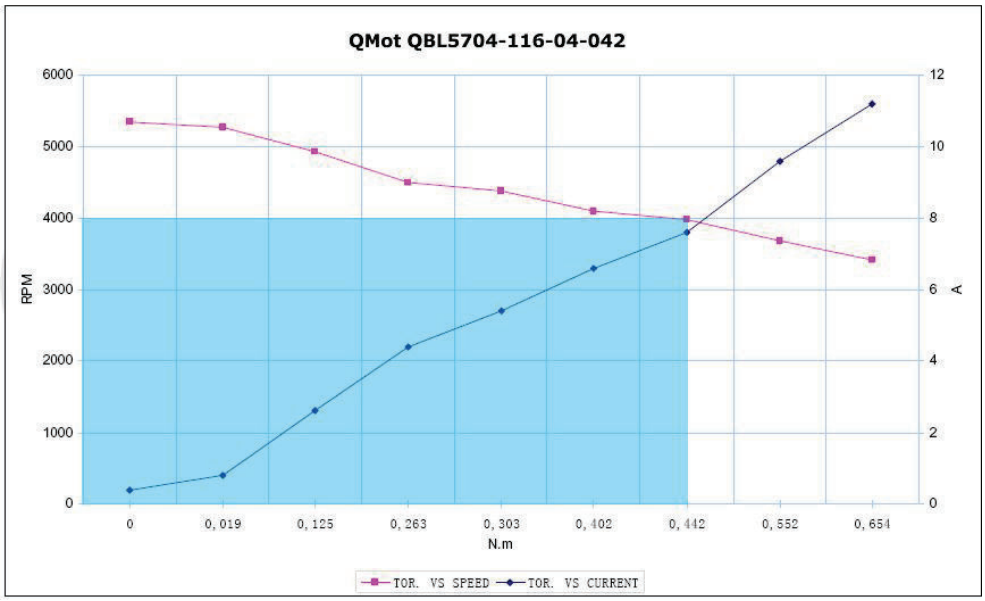


Figure 3.7: Velocity vs. Torque curve of the Trinamic QBL5704-116-04-042 motor.

Figure 3.8: The QBL5704 BLDC motor selected for the positioning system.

3.5 BLDC Motor Board Implementation

A circuit board was designed to drive and control the QBL5704. A block diagram of the design is shown in figure 3.9. The motor can run with a voltage up to 36V, while the rest of the system operates at 3.3V and 5V digital levels, and so the voltage had to be stepped down. A buck converter is used to do this, and then it supplies 10V to the digital supples, the 3.3V and 5V LDOs. The micrcontroller used is the ATXMEGA128A1, and it provides the timing and commutation sequencing to the gate drivers which turns on the half-bridges connected to the motor phases. It also reads the hall sensor states from the motor. An opamp was placed across the battery and gives a voltage to the the microcontoller's ADC port, and similarily, a current sense resistor was placed between the half-bridges and ground and a differential amplifer gives a voltage across this resistor

to the ADC port. These opamps were meant to allow the microcontroller to monitor the motor voltage and torque, but were not necessary for the positioning application. The next subsections will give some insight into the design of all the hardware. The following sections refer extensively to the schematics in appendix A.0.1.

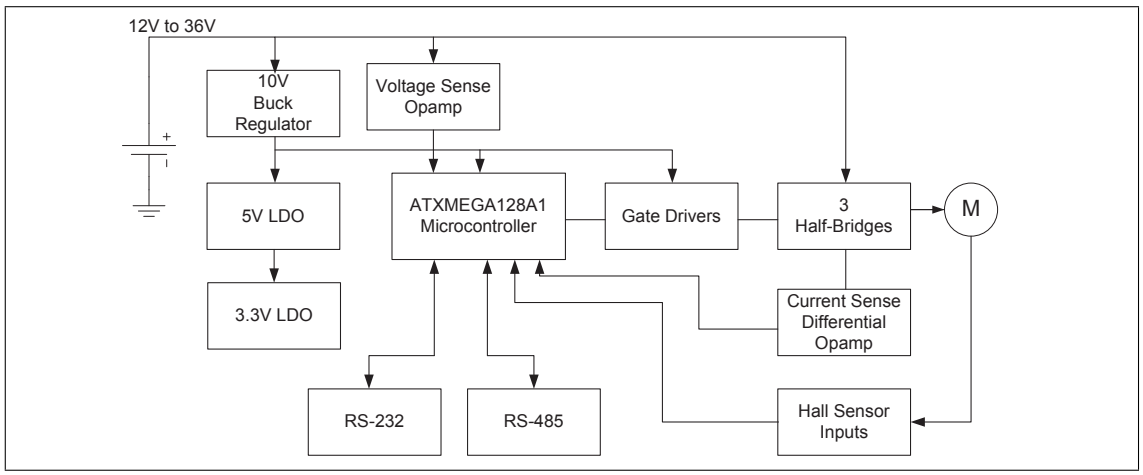


Figure 3.9: Block diagram of the BLDC motor controller hardware.

3.5.1 Voltage Regulators

Because the motor is rated to operate up to 36V, the supply voltage had to be stepped down for the other electronics. By looking at the datasheets of all of the components, an estimate on the current draw of the system was made. The 3.3V supply needs to source about 70mA and the 5V supply needs to source about 20mA. Low drop-out(LDO) regulators were used to regulate the 3.3V and 5V supplies. To reduce the power dissipation of the 3.3V LDO, and thus reduce its voltage drop, the 5V LDO supplies the input voltage to the 3.3V LDO, which means that the 3.3V LDO must only drop 1.7V across it. This dictates that the 5V LDO must supply about 90mA.

The LM3480 was chosen as the 3.3V LDO because it can source 100mA with a drop-out voltage of 1.2V. Also, the datasheet recommends applications for this chip to convert a 5V supply to 3.3V, so this LDO did exactly what was needed. The LM2941S was chosen as the 5V LDO and it can supply 1A, well above the 90mA it must supply, so it is safely oversized to give it headroom. The LM3480 has a predefined output regulation voltage of 3.3V, while the LM2941S has a feedback pin that allows the voltage to be adjusted by a voltage divider. Tantalum capacitors are used at the output of both of these regulators because they have a higher ESR resistance than ceramic capacitors, and the regulators require this resistance to maintain stability in their output regulation.

Power dissipation calculations were made to ensure that these devices would not get too hot in the operating temperatures. For the LM2941S, the maximum junction

temperature, $T_{J(max)}$, is $150°C$, the junction-to-ambient thermal resistance,Θ_{JA}, of the TO-263 package is $73°C/W$, and so the maximum ambient temperature, $T_{A(max)}$ for the LDO is given by equation 3.6, which yields $117.2°C$. Likewise, the LM3480 has a Θ_{JA} of $300°C/W$ and a $T_{J(max)}$ of $150°C$, so it can operate at a maximum temperature of $114.3°C$ under the design's load conditions.

$$T_{A(max)} = T_{J(max)} - (V_{dropout})(I)(\Theta_{JA}) \tag{3.6}$$

These calculations assume a 10V supply into the 5V regulator. If this 10V supply was not regulated, it could go up to 36V, and, by using equation 3.6, it shows that the 5V LDO would burn up under this condition. Another reason that a 10V supply is needed is that the gate drivers for the half-bridges require at least 10V as one of its supplies, which is discussed further in the gate driver section. So, to solve this, a buck regulator was used to bring the voltage from the unregulated battery voltage to 10V. The LM20333MH has a rated input range of 4.5V to 36V and is capable of sourcing 3A. The datasheet of this device was used extensively for choosing the components and connections.

Some of the simple design decisions were to pick the resistor values to set the output voltage. The output voltage is set by a voltage divider circuit that feeds the output voltage back to the internal regulator. There is also a bootstrap capacitor that must be placed between the switcher output of the regulator to a bootstrap pin. This bootstrap capacitor provides the voltage to the gate driver that controls the inverter switches. The SS/Track pin is connected to a capacitor to ground which gives the device a soft start, where the regulator reaches regulation voltage before 5ms of being powered. The sync pin was left unconnected, which defaults the switching frequency to 200kHz.

The output inductor is selected based on the allowable ripple current in the inductor, the duty cycle of the switching, the switching frequency, and the drop out voltage of the regulator. This is given in equation 3.7. The datasheet recommends that the ripple current not exceed 30% of the maximum rated output current(3A). The duty cycle is given as: $DutyCycle = V_{out}/V_{in}$. A $47\mu H$ inductor gives a ripple current of 770mA, well below the 30% limit. We see that if the switching frequency is increased, the inductor value can be decreased, but then the efficiency of the internal inverter switches are decreased.

$$L = \frac{(V_{in} - V_{out}(DutyCycle))}{I_{ripple} f_{sw}} \tag{3.7}$$

Similarily, the output capacitor, C_{out} must be selected based on the allowable ripple voltage that the output will have. This depends on the capacitor value, its equivalent series resistance(ESR), and the inductor ripple current, given in equation 3.8 from the datasheet. Using, a $4.7\mu F$ capacitor with $3.5m\Omega$ ESR, the voltage ripple will be 106.3mV.

$$V_{ripple} = I_{ripple}\left(R_{SSR} + \frac{1}{(8)(f_{sw})(C_{out})}\right) \tag{3.8}$$

Lastly, an RC compensator, with values R_{C1} and C_{C1}, must be connected to the regulator to add a zero for its internal control loop which cancels out a pole created by the output inductor and capacitor. The datasheet gives equation 3.9 to calculate the compensator values. The device should only need to supply about $90mA$, aside from when the gate drivers are firing, and since the device is rated for $3A$, the output current was decided to be $800mA$ to give the device extra current capability for overhead. Using this value, the compensator components were calculated to be 6.8nF and $1.96k\Omega$.

$$R_{C1} = \left[\frac{C_{C1}}{C_{out}}\left[\frac{I_{out}}{V_{out}} + \frac{(2)(DutyCycle)}{(f_{sw})(L)}\right]\right]^{-1} \tag{3.9}$$

3.5.2 Microcontroller

The microcontroller chosen to perform the commutation, position control, and PC interface was the ATXMEGA128A1. This is an 8-bit microcontroller that can operate up to 32MHz. It has 9 8-pin ports, each attached to various peripherals such as 16-bit timers, ADCs, and USARTS. A summary of the port connections is given in table 3.10. Also, the microcontroller can be operated with an external crystal, or with an internal RC oscillator. The RC oscillator is acceptable for many applications, but the crystal clock source can provide greater precision and stability over operating temperatures. This design includes a crystal to give the programming of the chip more options, but the internal oscillator is precise enough for anything needed in this motor control application. The crystal oscillates at 16MHz, and 22pF bypass capacitors were selected to balance the crystal drive circuit interal to the microcontroller.

Pin \ Port	0	1	2	3	4	5	6	7
PORTA	ADC - Motor Voltage	ADC - Motor Current	-	-	-	-	-	-
PORTB	Push-button/Reed Sensor	Push-button	-	-	JTAG-TMS	JTAG-TDI	JTAG-TCK	JTAG-TDO
PORTC	USART - RS-232 RTS (GPIO)	USART - RS-232 CTS (GPIO)	USART - RS-232 RXD	USART - RS-232 TXD	USART - RS-232 DTR (GPIO)	USART - RS-232 DCD (GPIO)	USART - RS-232 DSR (GPIO)	USART - RS-232 RI (GPIO)
PORTD	-	-	USART - RS-485 RXD	USART - RS-485 TXD	-	-	-	-
PORTE	AWEX 16-bit Timer - HIN1	AWEX 16-bit Timer - LIN1	AWEX 16-bit Timer - HIN2	AWEX 16-bit Timer - LIN2	AWEX 16-bit Timer - HIN3	AWEX 16-bit Timer - HIN3	AWEX 16-bit Timer - SD1	-
PORTF	Hall Sensor 1	Hall Sensor 2	Hall Sensor 3	-	-	-	-	-
PORTH	LED0	LED1	LED2	LED3	LED4	LED5	LED6	LED7
PORTJ	Transmitting Transducer Trigger	Measurement Trigger	-	-	-	-	-	-
PORTK	-	-	-	-	-	-	-	-

Figure 3.10: Port connections and peripherals used for the ATXEMGA128A1.

PORTA has an ADC peripheral that can be configured to read the voltages coming from the opamps that measure the motor voltage and the voltage across a current sense

resistor in in series with the motor windings. Port B is configured as an input port, with pushbuttons connected to them. Pin 1 of port B also has a reed sensor soldered to the ends of the pushbutton, so that when the reed sensor is actuated, the pin can interrupt the microcontroller. Port C is used as the serial interface to the PC. All of the RS-232 control lines were added to the port; however, the USART of the ATXMEGA128A1 only provides hadware support for the RXD and TXD signals, and so the other lines must be provided as GPIOs and customized software must be written to perform the functions of these control lines. This is not a problem because the simple serial communication used for this application does not require handshaking or any other advanced functions of the RS-232 standard, and so these extra signals are not used. Port D is used in the same setup as port C, where only its RXD and TXD signals are needed to transmit and receive on RS-485.

Port E is an important port used for controling the motor because the timer associated with this port has an advanced waveform extension(AWeX) peripheral. This port is used to supply the highside and lowside signals to the gate drive that controls the half-bridges of the motor phases. The port is used as a normal 16-bit timer to generate a PWM signal, but it also uses the AWeX peripheral to configure dead-time-insertion(DTI) periods for the highside and lowside switches. Basically, when the motor is commutated from one state to another, there is a possiblity that the highside gate and the lowside gate of the same half-bridge both need to be switched, and during the turn-on time of the highside and the turn-off time of the lowside transistion, or vice versa, both switches can be on. This state is known as shoot-through because current from the highside supply has a direct path to ground through the MOSFETs, and it causes damage to the MOSFETs as well as reduces the efficiency of the controller. The AWeX addresses this problem by routing the PWM signal from the 16-bit timer into the AWeX output register that goes to the port pins. A DTI value loaded into a register tells the output AWeX register to wait a certain amount of clock cycles inbetween changing its outputs to high and low. Additionally, the PWM signal generated by the timer that goes to these port pins is used to control how much voltage the half-bridge supply to the phases. 33Ω resistors are placed in series to these ports to prevent ringing on the traces as the PWM signal changes constantly at 30kHz.

Port F is configured as an input port for the signals from the hall sensors. The hall sensors are supplied with 5V, but they have open-collector outputs, and so they must be pulled up to the system's digital signal when high. Since these signals are provided on a header that could be improperly connected or subjected to ESD spikes, the ports pins are protected with zener resistors that limit the voltage to 3.3V if they are accidentally connected to the battery supply. Also, a low-pass filter was connected on the input to these pins to filter out any noise that may couple onto the signal from other electronics or from the motor since the motor's inductive spikes can be very noisy, which can happen during a commutation but also at the PWM switching frequency of 30kHz. The motor

rotates at a maximum of 4000RPM, and each hall sensor pulses twice(once per electrical angle) in a revolution, so the maximum hall sensor frequency is given in equation 3.10. A cutoff frequency was chosen at 14.5kHz to avoid distorting the hall pulse's higher harmonics, while still cutting off the higher PWM frequency.

$$f_{hall} = \left(\frac{2pulses}{rev}\right)\left(\frac{4000rev}{min}\right)\left(\frac{1min}{60sec}\right) = 133Hz. \tag{3.10}$$

Finally, port J is configured as an output and can provide time delayed trigger signals to a transmitting tranducer and a measurement device. The time delay allows the measurement device to begin measuring right as a sound signal reaches it, after having traveled a distance from the transducer.

3.5.3 Serial Interface

One of the simplest ways for the ATXMEGA128A1 to communicate with a PC is by using its universal synchronous-asynchronous receiver/transmitter (USART) to send bytes out serially on an RS-232 link. RS-232 is a large standard, but the important concerns for using it with this microcontroller is that the voltage levels must be shifted and the frequency that the bytes are sent out serially must match a baud rate that the PC com port is set to communicate at. To shift the levels, an RS-232 transceiver chip, the MAX3245E, is used. This chip uses an internal charge-pump to boost the digital voltage levels of the microcontroller to swing the transmit serial signal from -5.4V to +5.4V, and it can receive signals driven by a sender from -25V to +25V. This chip is capable of 200kbps transmission, and so the 19200 baud rate of the ATXMEGA128A1 can be handled by this easily.

Another serial link was added so that if the motor controller is a far distance away from the device that it communicates with, it could still communicate without problems. The RS-232 link only has a range of about 15m, and so an RS-422/485 transceiver is used to communicate over longer distances. The main difference between RS-422 and RS-485 is that RS-485 can have more driver and receiver devices on the bus, but both can work between two devices. Their range is extended by stronger drivers, and the transmit and receive signals are converted into differential pairs that improve noise immunity. The transceiver chosen is the MAX489, but it operates at 5V, while it must interface to the 3.3V logic-level microcontroller. The SN74LS07D is a hex inverter(6 inverters), with open-collector outputs which allows the output to be pulled up to the desired voltage level. Normally an open-collector inverts the signal operating it, but the built in inverters cancel out this inversion such that they act as cheap logic-level converters. Lastly, this serial link can only be used with other devices with an RS-422/485 transceiver, and so it cannot communicate to most PCs directly without a converter.

3.5.4 Voltage sensors

The following two opamp designs were made, but only to give the circuit board more versatility later down the road if it were to ever be used for another application. The first opamp circuit shown in figure A.4 with U12 is designed to scale the motor supply voltage, and convert it down to a voltage range that the microcontroller's ADC port can accept, which is 3.3V to 0V. The circuit assumes that the supply voltage can be no lower than 10V, which is true because the electronics it supplies needs more than 10V to operate, so this sets the minimum that the voltage reading will be. The maximum is set by the motor's maximum supply voltage of 36V.

R34 and R56 act as a voltage divider that brings the 36V maximum down to 4.06V, and the negative feedback sets the gain on this 4.06V as well as subtracts the 10V by the same gain of R57/R33, or 0.127. The output voltage of the opamp is given in equation 3.12. The circuit was simulated, and a DC sweep of V_{motor} was used to verify the voltage scaling was correct. The simulation is shown in figure 3.11, and the circuit used for the simulation is in appendix F.1. Finally, some capacitors were added to give the circuit some noise immunity and filter out any transient voltage noise that the motor might be creating on the motor supply voltage.

$$V_{out} = V_{motor}\left[\left(\frac{R56}{R56+R34}\right)\left(\frac{R57}{R33}+1\right)\right] - 10V\left(\frac{R57}{R33}\right) \tag{3.11}$$

$$= V_{motor}\left[\left(\frac{12.7k}{12.7k+100k}\right)\left(\frac{12.7k}{100k}+1\right)\right] - 10V\left(\frac{12.7k}{100k}\right) \tag{3.12}$$

The second opamp circuit, shown in figure A.4 is an instrumentation amplifier that measures the voltage across the sense resistor R44 in figure A.5, between the lowsides of the half-bridges and ground. The sense resistor should be as small as possible so that it does not hinder the current flow in the motor windings, but this means that the voltage drop across it will be small. Additionally, the switching and commutation of the motor can create noise on this resistor, and so an instrumentation amplifier was used to reject the common-mode voltage across the resistor since instrumentation amplifiers have a very high common-mode rejection ratio(CMRR) property. The AD623ARZ was chosen because it is a single supply, rail-to-rail instrumentation amplifier with a high CMMR of over 80dB at frequencies up to nearly 10kHz. The gain value was determined by the fact that the motor is specified to have a continuous current of 6.67A. If the sense resistor is 0.01Ω, then the maxmum voltage across the sense resistor will be 0.0667V. To scale this for the ADC port on the microcontroller, it needs a gain of 49.47. The AD623ARZ gain is determined by a single resistor, and a value of 2.2k gives the amplifier a gain of 46.45. During peak currents from the motor, the current can be as high as 20A. This is not a problem because the AD623ARZ is supplied a voltage from the 3.3V supply, and so it would saturate rather than amplifying the voltage above the voltage limit on the microcontroller. If the motor never runs at its maximum torque, the current will not get

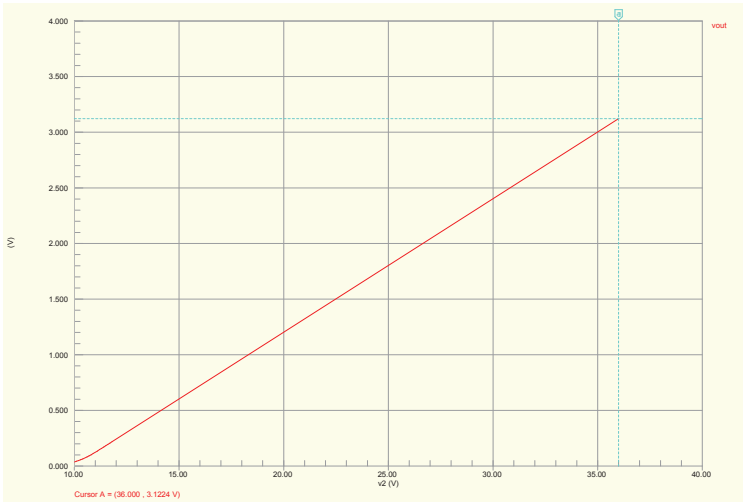


Figure 3.11: The simulation of the voltage scaling circuit used for measuring the motor supply voltage. The supply voltage V2 was swept from 10V to 36V, and V_{out} shows the opamp scaled the voltage from about 0V to 3.12V.

up to 6.67A and so the gain resistor can be changed to give the amplifier to give better resolution on lower currents.

3.5.5 Gate-Drivers and Half-Bridges

The half-bride circuits and their gate drivers are shown in figure A.5. The half-bridges transfer current from the motor voltage supply, through the motor phases, and then to ground. In any half-bridge, only the highside or the lowside will be on at any time. The power MOSFETs that control the current path were selected to have high enough ratings for the selected motor. The MOSFET chosen for these ratings was the IRFR3806. The motor voltage can be as high as 36V, and so the drain-to-source voltage rating, $V_{DS_{max}}$, of the MOSFETs was selected to be 60V, well above the maximum voltage. Also, external flyback diodes were placed in parallel with the drain-to-source nodes because, during motor switching, the inductive voltage spikes of the motor could exceed the drain-to-source rating, and the diodes provide a current path back to the supply voltage.

Also, the MOSFETs must be sized appropriately for current and power dissipation. A motor can have current spikes up to 20A according to the datasheet, and so the MOSFETs had to be selected to handle these surge currents; they are rated for 43A. Finally, during operation of the motor, the MOSFETs will be switching on and off and so they will dissipate heat across their drain-to-source resistance R_{DS}. The power dissipation is dependent on the conduction loss when the MOSFET is fully on, and also the switching loss as the MOSFET goes from high to low. The total average power dissipation, $P_{D_{max}}$ is given by equation 3.13, where D is the duty cycle that the switches are being modu-

lated at, f is the switching frequency, t_r and t_f are the rise and fall times of the MOSFET, I_{motor} is the maximum average current into the motor, and V_{motor} is the supply voltage to the motor. The switching loss is approximated by assuming that during the switch, the voltage and current are dropping between on and off, and so the average voltage and current during this switch are half of their maximum values [11]. Since each MOSFET is only switching for 1/3 of the time during a revolution of the motor, the average power is only a third of what a single MOSFET always switching would dissipate. Using the values from the datasheet and assuming a 50% duty cycle and 30kHz PWM switching signal, the power dissipation is 170mW. This is much less when the motor is operated below its maximum current and voltage ratings. Since the MOSFETs are taking very little power from the motor, this hardware configuration is a very efficient method for driving the motor.

$$P_{D_{max}} = \frac{1}{3}\left((D)\left(I_{motor}^2\right)\left(R_{DS_{on}}\right) + \left(\frac{I_{motor}}{2}\right)\left(\frac{V_{motor}}{2}\right)(f)\left(t_r + t_f\right)\right) \tag{3.13}$$

MOSFETs with high voltage and current ratings generally have higher gate capacitance, and so they require more charge at their gate to turn on. Also, the half-bridge setup causes the highside MOSFETs' source terminals to be floating. For theses reasons, driving the MOSFETs directly with the port pins of the microcontroller at 3.3V is not acceptable. So, to interface the port pins to the MOSFET gates, gate driver ICs are used. The gate driver selected was the IR2112. The gate driver is capable of switching its outputs at high frequencies, enough for the 30kHz PWM signal. To turn on the high-side FET, the gate driver's VS pin is used as the reference for a bootstrap circuit that pushes the HO pin 10V(the VCC supply) above the highside MOSFET's source voltage. The two important considerations for this design are, first, to choose a bootstrap capacitor large enough to supply the charge that must turn on the highside gate. Secondly, the bootstrap diode must have a fast enough reverse recovery time so that, when the highside's source node floats and the diode is reverse biased, the boostrap capacitor does not discharge back through the diode and damage the 10V supply.

3.5.6 Reed Sensor

A reed sensor was glued to the top of the positioning stand. A magnet was glued onto the moving platform, and aligned with the reed sensor. A reed sensor is a switch that is actuated when a magnetic field is in proximity. This sensor is used to signal to the microcontroller that the positioning platform is at the top of the stand. A picture of the reed sensor used and how it was setup is shown in figure 3.12.

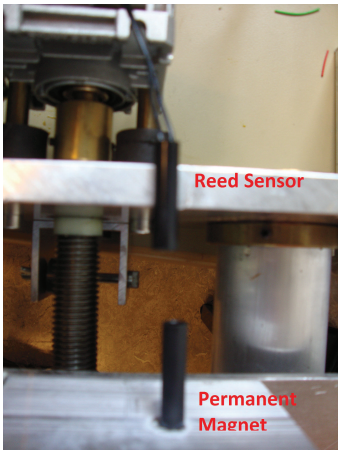


Figure 3.12: Reed sensor setup on the positioning stand to sense when the platform is near the top.

3.5.7 Board Layout

The circuit board was designed for 4-layers and can be see in figure A.6, A.7, A.8, and A.6. The power voltages: 3.3V, 5V, and 10V are placed on the top layer and flooded where possible. The bottom layer is a ground plane that is flooded across the entire board. The ground plane serves a few useful functions. First, it provides immediate paths for current to flow to ground. Also, it acts as a ground shield to keep EM radiation from leaving the board and from entering the board. Also, it can serve to disperse heat generated in the board. Additionally, a ground ring is placed around the edges of all four layers, and a via fence is placed in this ring. The via fence acts like a faraday cage around the board and prevents EMI. The spacing of the vias determines the maximum wavelength of EMI that can pass through. The signal traces are placed on inner layers so that the ground and power plane can shield them.

The final board is shown in figure 3.13. Some of the components were removed and modified as the board was tested on over time.

3.6 BLDC Board Software

The BLDC motor software is presented in appendix B. Flow charts of the software are shown in figures 3.14 and 3.15. The microcontroller begins by initializing its internal oscillator to the 32MHz frequency. Then ports are configured for their I/O functions, and then the timers are configured.

3.6.1 Reed Sensor Detection

A reed sensor is attached to port B pin 1, and the pin is pulled high. When the reed sensor actuates, the port is pulled low and generates an interrupt. This interrupt sets a flag that the rest of the main program always checks to make sure that the position has

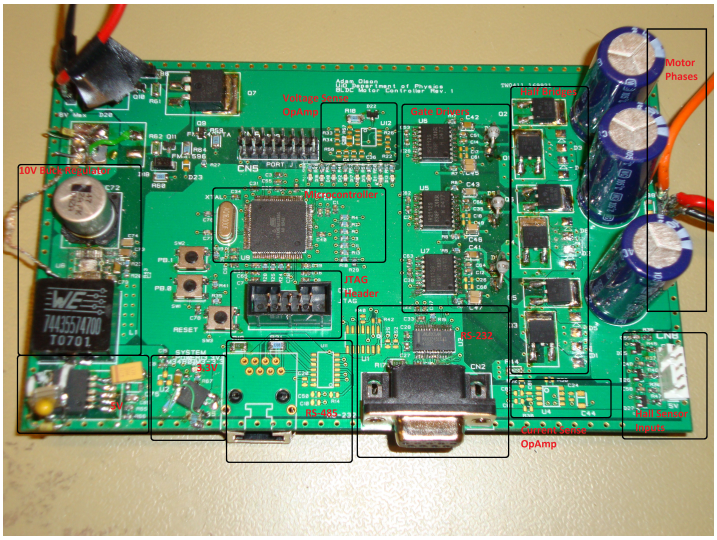


Figure 3.13: A picture of the BLDC motor controller after being built and tested.

been initialized. This serves to important functions. First, if the software is not allowed to run its initialization routine, it can only move and send relative positions, which is not useful. Secondly, if the reed sensor does not set a flag to tell the software that the transducer platform is at the top of the stand, the motor controller could keep moving the platform upwards and damage the stand and the motor.

3.6.2 RS-232 Interface

The UART is configured to send and receive messages at a baud rate of 9600. The software has a switch case that decides what to do based on the message received. The bytes that the UART ISR stores are not acted upon until a read message function is called by the main loop. If the message received starts with a "#" character, the UART interrupt service routine(ISR) begins storing the next 5 bytes received into a position value array, which represents a command signal from the PC to move the transducer platform to this height. The position value can be from 0 to over 13000, and so ASCII bytes are used to represent each digit. For the ASCII value to be used in 16-bit integer math, the ASCII digits are convertered to binary coded decimals(BCD) and then converted to a position value. Also, the UART is constantly sending out the motor position in 10ms intervals as a 5-byte ASCII value with a start bit of "#", which serves as feedback to the user operating the PC what the current position of the transducer platform is.

3.6.3 Timers

Timer E is configured for the AWEX mode of operation, where it generates a PWM signal based on the fraction of its 16-bit wide period register that is high. The period register is used to set the PWM duty cycle, and the AWEXE.OUTOVEN register determines the pattern of the 6 half-bridge MOSFETs. Timer C is configured to overflow when its counter has counted up to a time delay that is set on RS-232. This overflow creates an interrupt that determines the time delay between two trigger signals for a transmitting transducer and a measurement device so that the measurement device can be synchronized to begin measuring when a soundwave that has been transmitted reaches the measurement device. Timer D is configured to interrupt every 2ms to update the position error between the desired position and the feedback position.

3.6.4 Position Feedback and Commutation

Port F is configured as an input that interrupts from both rising and falling edges caused by the hall sensors. In the ISR, the hall combination at port F is saved and then it is checked in an index to see where in the hall sequence the motor is at. It also stores a previous hall sequence, and so it compares the current hall sequence index to the previous hall sequence, and if the current sequence is foward of the previous, it means that the motor has moved in one direction, and if it is backwards, it has moved in the other direction. The ISR keeps a running total of how many times the hall sensors have interrupted, and increments or decrements this total depending on the direction. Each time the number of hall sensor interrupts rolls over 11 or under 0, a rotation is added or subtracted to the previous number of rotations variable. This is how the system integrates the hall pulses to give a feedback position.

After the ISR has determined the direction and position of the motor, it then checks the hall combination and chooses the next commutation sequence that corresponds to this hall combination. This sequence is applied onto port E pins that switches the half-bridges. This is only one part of making the motor move. If the PWM is 0%, the commutation does nothing, and so the motor won't move until another part of the code has set the PWM to 100%.

3.6.5 Control Loop

The control loop ensures that the command position is the same as the feedback position. While this is commanded in revolutions of the motor, the actual position of the transducer stand is stepped down with the gear system. The speed reduction gear has a 20:1 turn ratio, and the worm gear thead spacing is 2.5mm. This works out to 1mm of platform position change for 8 revolutions of the motor. This conversion is taken care of in the user application software, and so the motor controller still only works in units

of revolutions. As was said previously, the control loop is activated every 2ms, and so the position feedback is sampled and the error signal is updated at this period. This was a requirement for a digital PID controller, where the control loop should update at a constant period; however, a PID controller was not designed.

Rather a simple state-based "bang-bang" controller was implemented. In the control loop, the current position is sampled and then subtracted from the command position. If the error is 0, the PWM output is set to 0% duty cycle, but a positive error means the motor must move the platform up and a negative error means the motor must move the platform down. A state-based system can be known to be unstable and exhibit "chatter" behavior where the error oscillates back and forth as the system overshoots each time it moves. These control systems can be stabilized by adding hysteresis or a deadzone to the error signal. This positioning system has a natural deadzone, which is the angle of the motor position that is between the hall sensors. This is a 30° angular displacement, which corresponds to 0.0104 mm along the length of the worm gear. The friction torque of the system is so high that even at $12V$ driving the motor, if the motor stops applying power when its error is 0, the friction of the system will stop the movement from the motor's inertia before it reaches the next hall sensor state.

3.7 BLDC Motor Controller Problems

When the motor was built and tested, there were a number of issues that needed to be resolved. The most simple issue was that the LEDs were connected to 10V and sunk current through the microcontroller, which destroyed the 3.3V regulator; the board was modified quickly for this error. Also, the RS-232 transceiver was wired to the connector as a master (data terminal), and PCs are also wired as a master, and so the TXD and RXD signals were swapped on an adapter connector.

A more difficult problem to find was that the original bootstrap diodes used on the gate-drivers did not have a fast enough recovery time, and as a result some of the gate drivers were being destroyed as well as the 10V buck regulator that supplied the bootstrap was damaged. These diodes were replaced, and also larger bootstrap capacitors were added on the gate drivers, and then the gate drivers began working better.

Another complicated problem was discovered as the unloaded motor was pushed to above 20V under certain software versions. When the commutations were configured to happen after hall-sensor interrupts, the motor began to experience large torque ripple at high RPM. The torque ripple was so high that it began current limiting the power supply. A secondary commutation method was programmed that had the microcontroller sampling the hall sensors in a high frequency timer interrupt, rather than interrupting from the hall sensors directly. This method reduced the torque ripple greatly. Unfortunately,

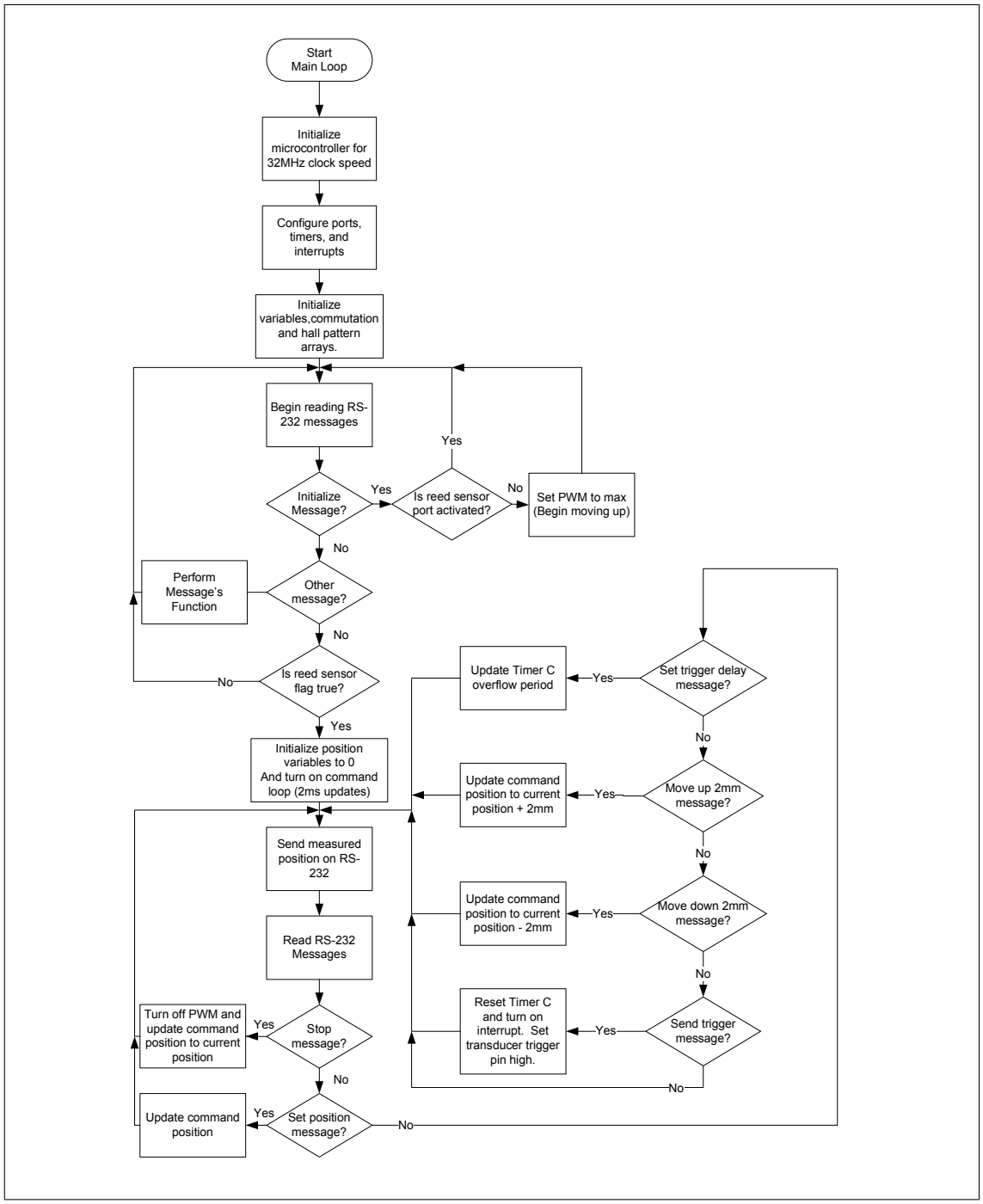


Figure 3.14: Flow chart of the motor controller's main loop.

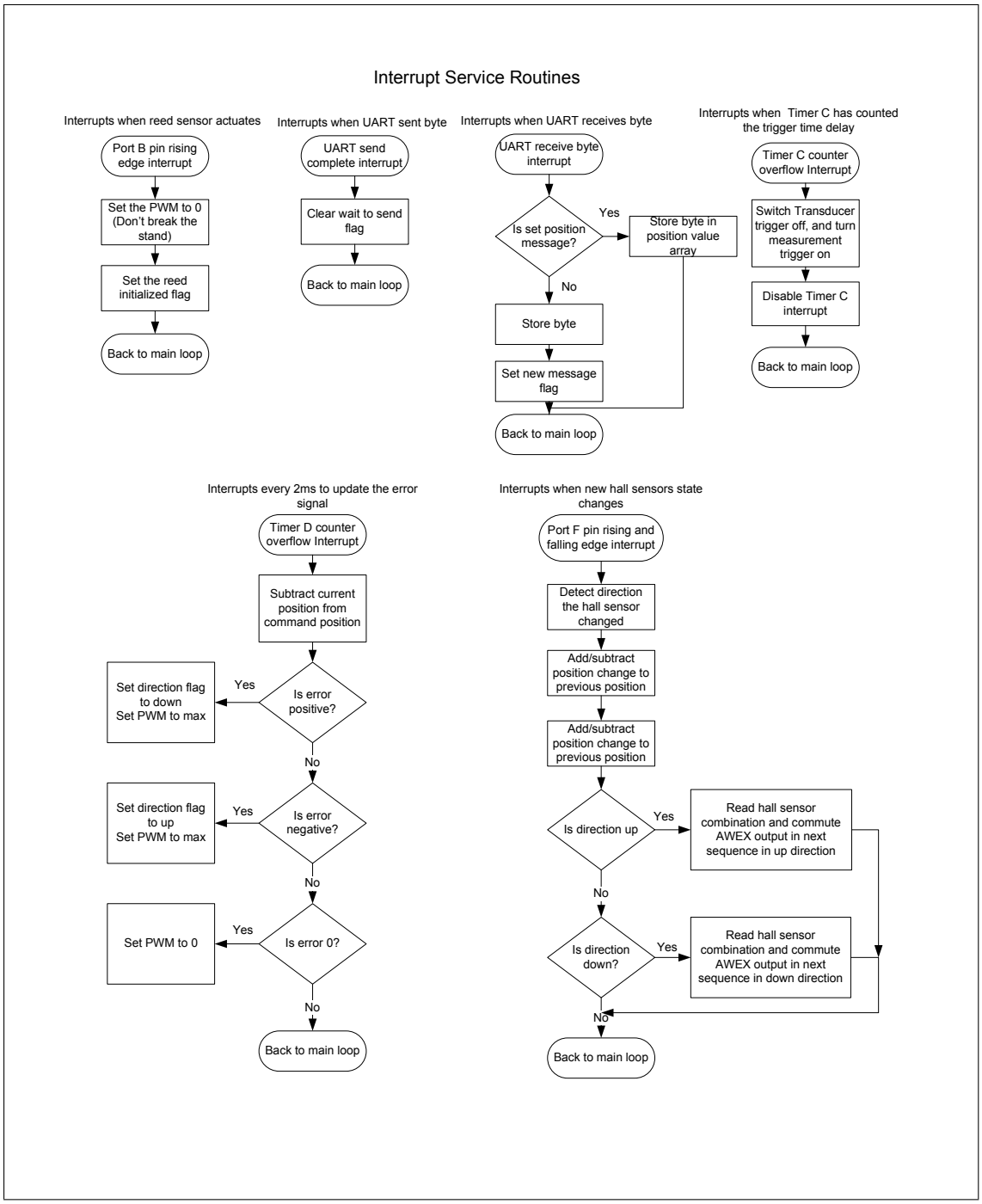


Figure 3.15: Flow chart of the motor controller's interrupt service routines.

the hall sensor interrupt version was tried again, and when the motor supply voltage was at 24V the 10V buck regulator exploded off of the board. There are a few explanations for this, but nothing is certain. First, the difference of torque ripple between the commutation methods is possibly due to hall sensor misalignment in the motor. It was discussed in section 3.3, a common downside to using hall sensor as feedback for commutation is that their alignment with the rotor magnetic field is not always perfect when manufactured. Interrupting off of misaligned hall sensor signals could create increased torque ripple as the velocity is increased because the motor controller has less time to force the rotor field back in sync with the phase field. So, the sampling method of commutation could possibly be resolving this issue by not always commutating exactly when the hall sensor tells it to, but rather is averaging out the commutations so that part of the misalignment offset time is skipped over during successive samples. A possible explanation for why the 10V buck regulator suffered the damage from this torque ripple is that the input of the 10V buck regulator is connected to the motor voltage supply, and during torque ripple, current surges could be creating inductive voltage spikes on the motor voltage supply that destroyed the buck regulator.

Unfortunately, as a result of the damages to the board, and the troubleshooting process, some of the original intended functions of the board were removed. The RS-485 transceiver and the voltage sensing opamps were removed to eliminate them as possible source of failures on the board during trouble shooting. These functions were eliminated as possible culprits, but there was not time to replace them on the working board. Also, development needed to continue on the board and there were not spare buck regulator ICs, and so the board was rewired to bypass the buck regulator section of circuitry. The motor supply voltage was wired directly to the 5V LDO regulator as well as to the VCC pins on the gate drivers. This means that the motor supply voltage needs to stay less than 20 volts, as that is the maximum rating for the VCC pins. Also, the 5V regulator should not have a large voltage drop across it, and so the motor controller should only operate at 12V-15V until the buck regulator circuitry can be repaired. This is acceptable because during testing it was found that the stand's gear system cannot really move much faster than when the motor is operated at 12V.

Some future guidelines learned from these problems are that more care should be observed when connecting a voltage regulator to noisy voltage sources, and protection circuitry should be carefully designed if it is necessary to supply the regulator from the such sources. Also, when diodes are used in both forward and reverse biases, they should be selected with a fast enough reverse recovery time to avoid voltage spikes back through the diode.

3.8 Test and Results

The motor was running, and one way to verify that it was operating as expected was to measure the back-EMF of the phases. This measurement is shown in figure 3.16. The figure shows the expected trapezoidal waveform, where each phase is 120° out of phase with the next. The spikes seen are due to the PWM signal modulating the voltage into the phases, and so the effective back-EMF is less(and the motor turns slower) as the PWM duty cycle drops.

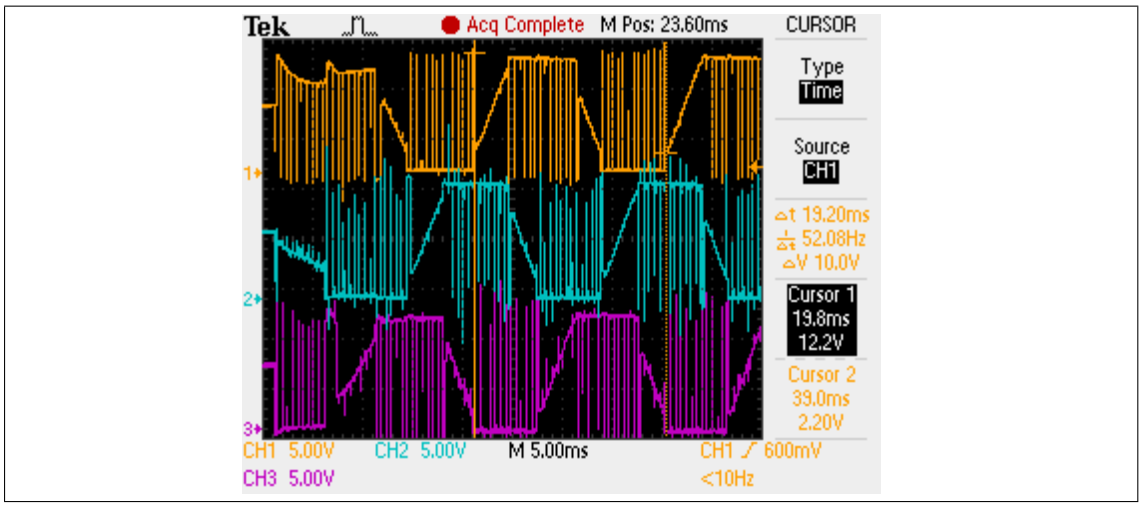


Figure 3.16: The BLDC motor's back-EMF voltages during operation.

The transducer platform was remodified by the mechanical department to fit the BLDC motor onto the transducer platform. With this setup, some initial results were found on the performance of the system. These test were made using the PC application from chapter 6.

First, it was found that the torque friction between the worm gear and the transducer platform is not constant along the entire shaft. With the motor running at 12V, the current draw from the motor varies from 1.5A to 1.9A, but as the motor reaches positions that are past 1.3 meters from the top of the stand, the current draw spikes as high as 2.2A at some sections of the worm gear. This is not a problem for the motor, but the speed reduction gear at the top of the stand began leaking fluid whenever it was pushed past 1.3 meters, and the housing of the speed reduction gear was very warm after this run. For this reason, the maximum distance from the top position on the stand is set as 1300mm in the software.

A quick method to test the positioning accuracy of the system was to measure along one of the stand poles with a tape measure and to mark 10cm increments across the pole. Additionally, a ruler was used to mark 5mm and 1mm increments for the first 40cm across a strip of tape that was adhered to the pole. This is shown in figure 3.17. A position command profile was created to run the motor through both some short position

increments, and also some long position increments to try to find errors that would add up to large offsets over a large enough distance.

The command positions were sent from the PC application, and then after the motor po-

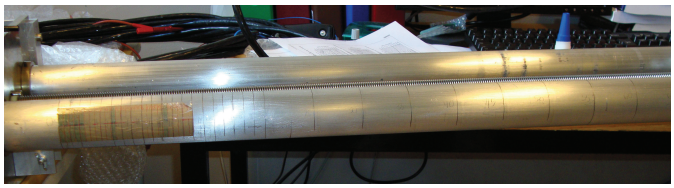


Figure 3.17: 1mm,5mm, and 10 cm markings placed along the pole of the stand to measure real position of the platform vs. commanded position.

sitioned the platform, the position was read off of the markings on the pole and recorded. The plot of the commanded position and the measured position position are shown in figure 3.18. The motor command sequence was: 0-10,10,15,20,25,30,35,40,100-105,5-10,500,1000,45-41,0. It shows that the motor positioned the platform at the commanded increments, except for two points. When the PC application commanded 35mm, the motor positioned the platform to 36mm, and again on the way down from 1000mm to 45mm, the motor controller positioned the platform at 46mm.

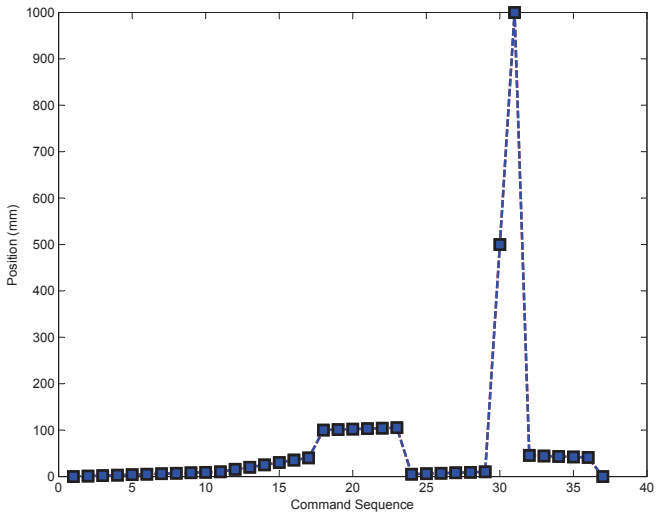


Figure 3.18: Data collected of measured position and a position profile commanded to the hydrophone stand. The commanded and measured curves are almost identical, so it is hard to see that there are two curves.

Chapter 4

Transducer Driver

4.1 Introduction

In this chapter, the development and analysis of circuitry to drive a hyrdoacoustic transducer setup as a transmitter are discussed. For the rest of this chapter, the term transducer refers to the hydroacoustic transducer in a transmitting mode. This became a part of the project because the old echosounder was inconvenient to use. I designed a transmitter driver with an electrical signal that is powerful enough to drive the transducer with a source level of $230.27dB$ with a $0.1ms$ long pulse train at $120kHz$. A charge-pump design was implemented in schematics and a PCB layout was routed to supply the electrical power to the transducer. There is little to no literature about driving hydroacoustic transducers with charge-pump voltage sources, but charge-pumps are commonly used in LED drivers, solid-state memory devices, and other applications. Applying a charge pump drive to this application has challenges and limitations.

The transducer that was already available in the physics department is the Simrad ES 120-4x10. The department also already had an echo-sounder available to source the required electronic signal, the EY500, however there are a few limitations to using this equipment, and the decision to design a new transducer echo sounder was made upon these limitations:

- The EY500 is more complex than the project needed and has many features that are not used. The EY500 is able to supply the signal needed, but time and knowledge of its operation are required to use the equipment and to ensure it is applied correctly to the experiment. The EY500 runs under rms which demands a separate PC to be started in order to control the transmission of the sound. This is time consuming, difficult, and not good with respect to protection against damage from water to use PCs in the field. The field experiments should use as few PCs

as possible. This applied not only to the current project, but also to any related projects in the future with new users.

- The EY500 must be sent a digital trigger pulse in order to transmit the transducer signal, and there is little control of the transducer signal after this trigger pulse is sent. Concerns with time delays necessary for triggering the sampling window of the transducer hardware can be simplified when a custom source is designed with full control over timing of the hardware and software.

- The EY500 is rather large and bulky, as shown in figure 4.1. Dimesions for this unit are L340 x W350 x H141 mm, and 7 kg.[1, p. 12] The project's application requires the equipment to be used either in a boat or on a platform attached to the transducer projector stand. Because the EY500 is a fullscale echosounder with many unneeded functions, the size of the device can be reduced greatly be using only the hardware needed for the project.

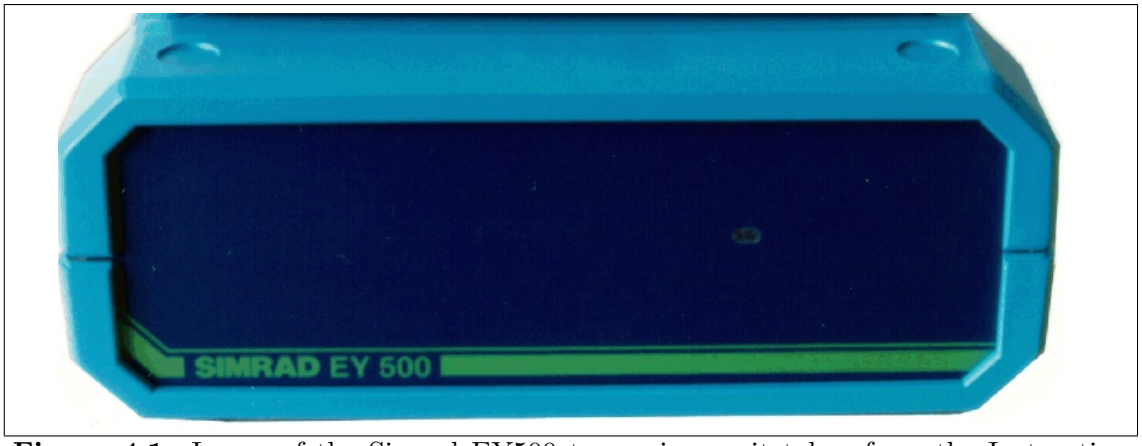


Figure 4.1: Image of the Simrad EY500 transceiver unit taken from the Instruction Manual. The unit is considerably larger than the prototype circuit board designed in the project.

A new transducer driver design will have a trigger input to send a pulse, and a trigger output as feedback that the pulse has been sent. This output can also be programmed with a delay so that it can trigger an oscilloscope or sampling device to sample the signal at the correct time. This is shown in figure 4.2.

4.2 Transducer Drive Requirements

As seen in figure 4.3, the transducer's specifications show the important parameters: resonant frequency, maximum pulse power input, maximum continuous power input, and

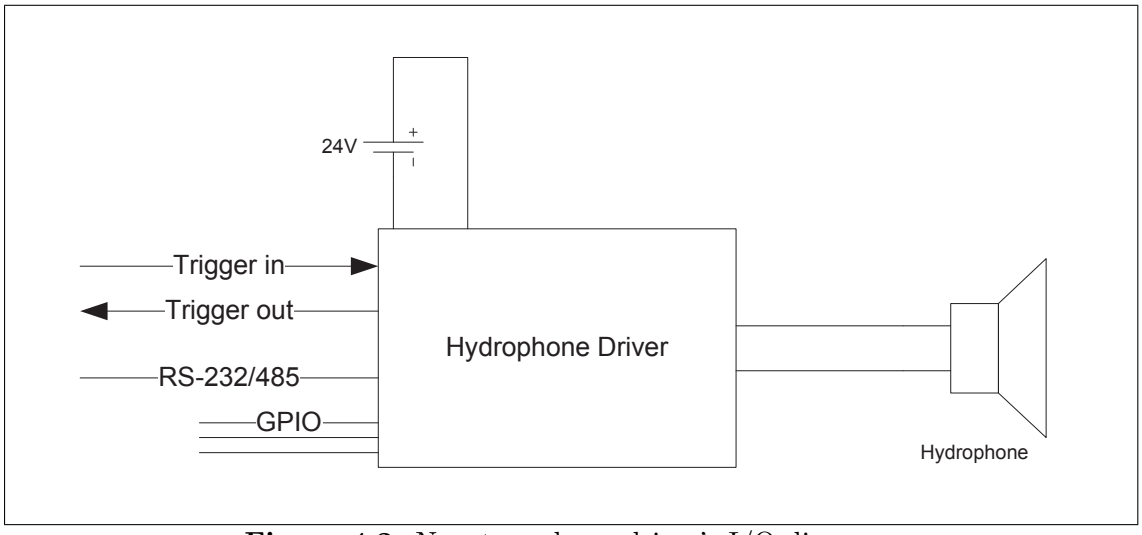


Figure 4.2: New transducer driver's I/O diagram.

nominal impedance.

The Simrad ES 120-4x10 is rated for 500W electrical pulses and the nominal load

Technical specifications

Resonant frequency	120 kHz
Beamwidth:	
Vertical	4.4 deg
Horizontal	9 deg
Directivity:	
D	850
DI=10logD	29.5
Equivalent two-way beam angle:	
Ψ	0.007 steradian
10 logΨ	-21.5 dB
Side lobes:	
Vertical	less than -25 dB
Horizontal	less than -18 dB
Back radiation	less than -25 dB
Angle sensitivity:	
Vertical	36
Horizontal	18
Impedance:	
Nominal	19 ohms
Max. variation in |Z|	15 - 24 ohms
Max variation in phase angle	±30 deg
Transmitting response	184.5 dB re 1µPa per V
Receiving sensitivity,	
open circuit	-186 dB re 1V per µPa
Electroacoustic efficiency	0.50
Maximum pulse power input	500 W
Maximum continous power input	20 W
Maximum transducer depth	20 m
Cable:	
Length	20 m
Diameter	11 mm
Weight without cable	6 kg
Storage temperature	-20 to 70 °C

Figure 4.3: Simrad ES120-4x10 Hydrohphone's Electrical Specifications. Maximum pulse input, Maximum continuous power input, and nominal impedance are the values needed for the design calculations.

impedance is 19Ω at the resonant frequency of 120kHz. However, this nominal load impedance is the complex impedance, but the average power is only dependent on the real impedance. Figure 4.4 shows the admittance characteristics of the transducer. At $120kHz$, the real part of the admittance, the conductance, is about $0.085mS$ which is equivalent to about 11.76Ω. Note that the datasheet says this impedance is only valid when all four elements of the transducer array are connected in parallel. This is the configuration to be used in the experiment, since a single beam is desired. Therefore each element individually has a higher impedance, and different calculations would need

to be made to operate this transducer in split-beam modes.

In general, a transducer's power rating is given as an RMS value, even when not specified[5]. Since the datasheet only provides an input power and the electrical impedance of the transducer, the maximum peak-to-peak voltage and current of the pulses were calculated by first converting the RMS power to a peak-to-peak value which is denoted as pk-pk in the following equations.

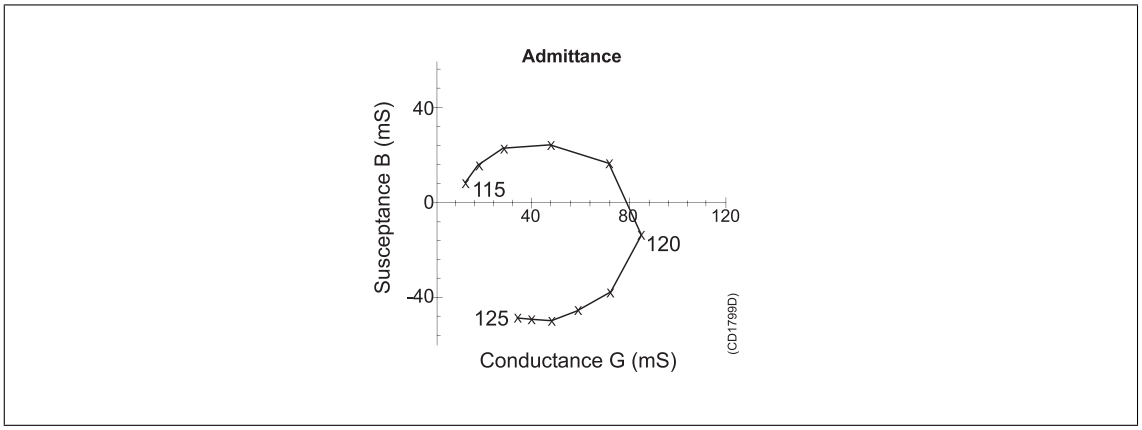


Figure 4.4: Admittance characteristics of the Simrad ES 120-4x10.

The RMS power is given as:

$$P_{rms} = V_{rms} I_{rms} = \frac{V_{rms}^2}{R} \tag{4.1}$$

where, for a sinusoidal waveform

$$V_{rms} = \frac{V_{pk-pk}}{2\sqrt{2}} \text{ and } I_{rms} = \frac{I_{pk-pk}}{2\sqrt{2}} \tag{4.2}$$

and so

$$V_{pk-pk} = \sqrt{(2\sqrt{2})(2\sqrt{2})P_{rms}R} = \sqrt{8P_{rms}R} \tag{4.3}$$

and using the values from the datasheet:

$$V_{pk-pk} = \sqrt{(8)(500\text{W})(11.76\Omega)} = 216.88V \tag{4.4}$$

and

$$I_{pk-pk} = \frac{216.88V}{11.76\Omega} = 18.442A \tag{4.5}$$

These values are the maximum that the transducer is rated during a pulse; however, the device that was replaced, the EY500, only operates the transducer at power levels shown in table 4.1.

So, the transducer driver's design is adjusted below the maximum to ensure safe operation, but designed such that it can source more power than the EY500 does if a more

Table 4.1: Operating values from the EY500 Instruction Manual. At 119kHz, the power transmitted is only 60W, while the transducer is designed for a maximum of 500W

Frequency (kHz)	Beam type	Power (W)	Pulse duration (ms)			Bandwidth (kHz)		Resolution (cm)
						Narrow	Wide	
37.878	Single	250	0.3	1.0	3.0	0.38	3.8	10
70.422	"	50	0.2	0.6	2.0	0.7	7.0	5
119.047	"	60	0.1	0.3	1.0	1.2	12.0	3
200.000	"	60	0.06	0.2	0.6	2.0	20.0	2
714.286	"	50	0.06	0.2	0.6	7.1	71.4	2
37.878	Split	250	0.3	1.0	3.0	0.38	3.8	10
70.422	"	50	0.2	0.6	2.0	0.7	7.0	5
119.047	"	60	0.1	0.3	1.0	1.2	12.0	3

powerful signal is desired. The chosen output power and resulting voltage and current requirements are shown in table 4.2

Table 4.2: Power output design parameters for the transducer driver

P_{rms}	P_{pk-pk}	V_{pk-pk}	I_{pk-pk}
425 W	3400 W	200 V	17 A

4.3 Voltage and Current Converter Methods

With these power requirements set as the maximum voltage and current pulses required, a voltage driver was designed. Automotive batteries were chosen as the power source for the transducer because the design was meant to be portable and operate in the field without easily accessible power. Car batteries are designed with a very low output impedance, and, when starting a car, the battery must be be able to supply enough current to rotate a crank shaft while maintaining voltage high enough for ignition electronics. This means that the batteries are designed to source hundreds of amps for over half a minute. The limitation of automotive batteries is that they are only manufactured in cells of a standard voltage of 12V, far below the 200V required by the design parameters of table 4.2. So the voltage must be converted with a transformer or DC-DC converter from 24V,using two batteries in series, up to 200V. There are several options for generating a regulated voltage at this magnitude:

4.3.1 Transformer Method

A common method in many transducer projector circuitry designs is to use a step-up transformer. A voltage signal is varied on the primary winding of the transformer, and the voltage is then transferred to the secondary winding at a higher voltage proportional to the ratio of number of windings on each side. Because a transformer can only transfer AC signals, the pulses should be generated on the primary side of the transformer. A simplified example of this type of circuit is shown in figure4.5. Although this is a practical method, there are several drawbacks to using it.

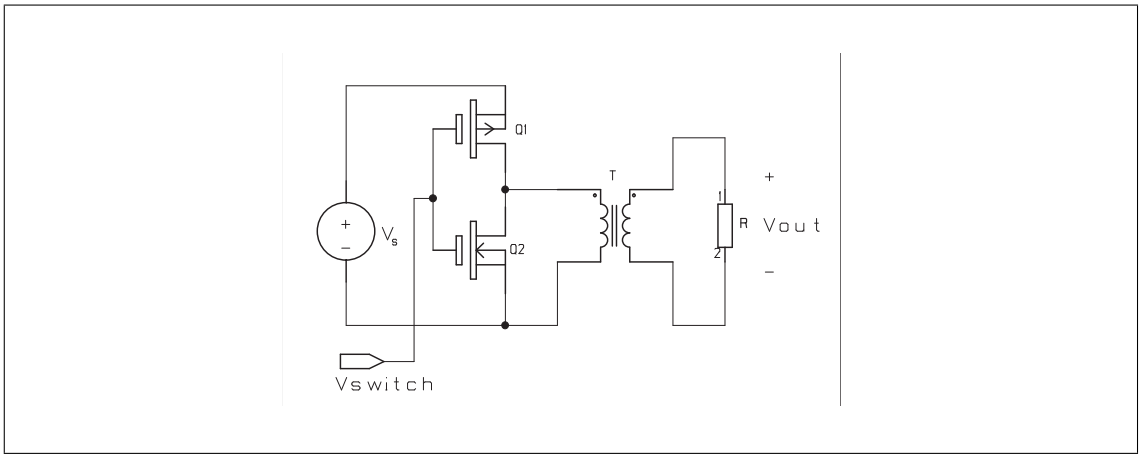


Figure 4.5: A simplified transformer drive circuit. A pulsed voltage is applied to the primary windings of a step-up transformer to generate the high voltage pulse on the secondary side.

First, transformer tolerances are not as precise as other available components for this application, and their properties change as load and signal change. Leakage flux from one winding to the next can cause voltage regulation to drop with varying loads [16, p. 68-74]. Secondly, as the signal goes from high to low, the magnetic field is changed, and the hysteresis of the core causes energy to be lost as heat. This hysteresis limits the frequencies that the transformer will operate under as well as distort the signal from the losses. Also, if the current pulse required is too high, the winding's magnetic fields may saturate so that no more current can be transferred linearly as expected. Another possible issue is when switching the windings with a square-wave at currents as high as 17 Amps, the inductive voltage spikes may be high enough to require more expensive switching components such as MOSFETs with better Vds ratings. Finally, it is hard to implement regulation in the design because it would require an adjustable transformer or tuning of the input voltage after the load has changed. All of these factors should be considered when designing a pulse generator with a transformer, although some may be non-applicable to the specific application and the non-ideal properties may be negligble.

There are other drawbacks to a transformer design not directly related to voltage step-up. One is size constraint, as transformers tend to be bulky and weigh a lot be-

cause of the cores used. Transformers are also electrically noisy, as the inductive voltage spikes from the switching of a transformer can couple to other parts of the system such as transducer amplifiers that will be sensitive to this noise, or the noise may be transmitted into the transducer and converted into excess heat dissipation and acoustic energy.

The main advantages of this method are simplicity of design, because a single transformer with the required windings can generate voltages many times larger than the voltage into the primary windings. Also, since the transformer can be designed with larger coils, they can be chosen for high current pulses without much resistive power dissipation.

4.3.2 Boost Converter Method

Another method is to use a boost converter switch-mode voltage regulator, also referred to as a SMPS or a flyback converter. This circuit has many applications such as step-up power supplies and spark-ignition circuits to start petrol enignes. A simplified circuit is shown below in figure 4.6. A DC voltage is supplied at the input of the SMPS device. The device switches the connection of an inductor to ground, resulting in a change of current in the inductor. The inductor's voltage is defined as: $V_L = L\frac{di}{dt}$. When the open-switch stage and close-switch stage current changes in the inductor are equal to each other for steady-state operation, voltage across the inductor is increased as the switch stays on for a higher duty cycle. This results in a higher voltage at the output, where the output voltage is dependent on the duty cycle as given by equation C.1. Some calculations for an appropriately sized boost converter needed to drive the transducer are given in appendix C.

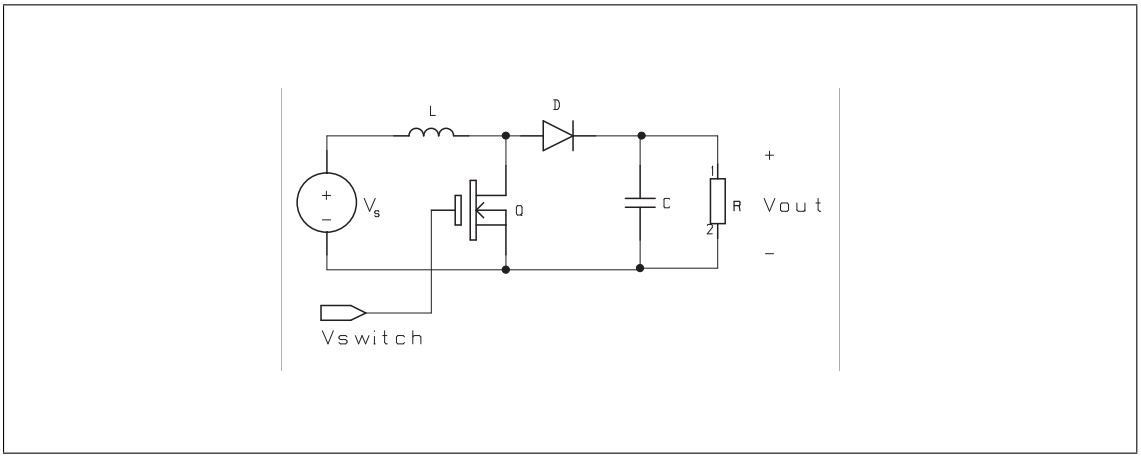


Figure 4.6: Simplified boost converter circuit shows that a switch varies the duty cycle to increase the voltage across the inductor which is passed through the diode to the output capacitor.

From appendix C, it is shown that the average current in the inductor would be $87.7A$

and this current must be switched at $57kHz$. For this application, it would be impractical to find a controller capable of swtching such high currents. A boost converter voltage regulator may be an option in this application, but the requirements are too high for most commercially available ICs. Another problem is that, like the transformer method, the switching on the inductor has the possibility to generate inductive voltage spikes that can be a source of noise that couples onto the signal as well as on neighboring circuitry.

The difficulty for a practical SMPS regulator IC to source $17A$ continuously at $200V$ from a 24V supply leads to a simpler solution. Placing large enough resevoir capacitors at the output of the regulator and then reducing the load requirements of the regulator greatly can make the components of the regulator more practical. With this simpler solution, the SMPS boost converter method will not allow unlimited pulses to be generated without a down time to recharge the output resevoir capacitor. Another issue is that, even with the resevoir capacitors, the voltage regulation of an SMPS may be insufficient because the controller can reach its regulation saturation whenever the resevoir capacitor voltage drops from the regulated output. In saturation, the duty cycle approaches 100 percent,and so the SMPS regulation is insufficient to ensure square wave pulses without distortion. These issues are also valid for the charge-pump method discussed next. Since the charge-pump method is the main method discussed, the issues are analyzed more closely there.

4.3.3 Charge Pump Method

A charge-pump design is the method chosen for the project. Charge pumps use switched capacitors to push the charge in the capacitors to higher voltages. Charge pumps can be regulated or unregulated. In a regulated design, one control method is to bias the switching FETs so their $R_{DS_{on}}$ changes the level the capacitors are charged to [25]. Charge pumps are known to have efficiencies of over 90% in some designs [18]. In many applications such as voltage translators in RS-232 transceiver ICs, output capacitors are added to the charge pump output to reduce the droop in the voltage level. A disadvantage is that a large multiplication of voltage requires many more circuit stages and components than the boost converter and transformer methods, and the stages at the high voltage levels require high voltage rated capacitors. Like the boost converter, but unlike the transformer method, the output voltage of a charge pump is controlled tightly with the tolerances of the components, and such a direct-drive of the hyrophone reduces the need for tuning the transducer[22].

One implementation of a charge pump is the Dickson charge pump. An example of a single Dickson charge pump stage is figure 4.7. This type of circuit doubles the voltage, and is discussed further in section 4.4 to give some background of the device that I designed.

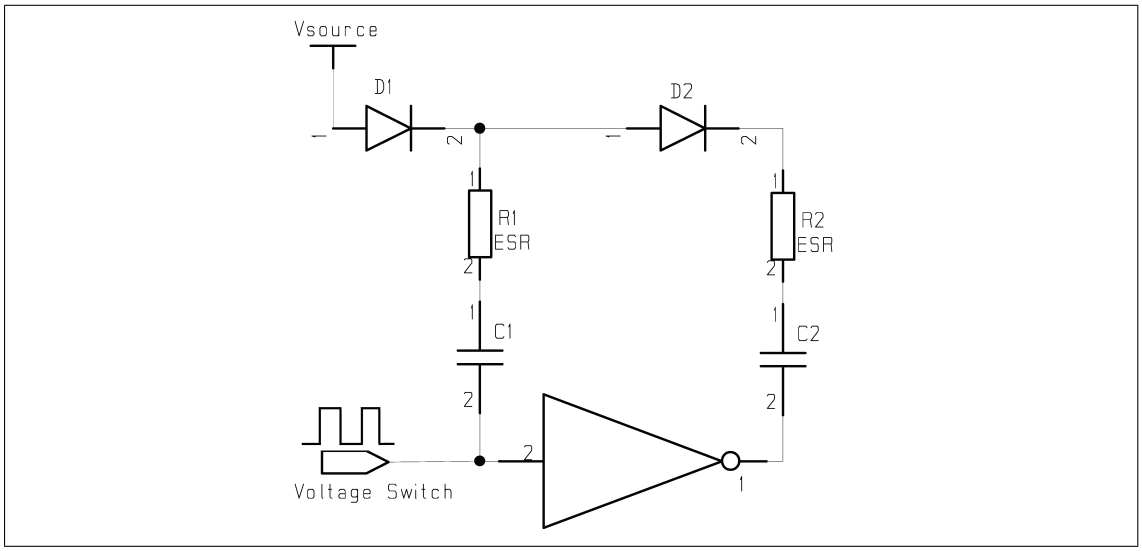


Figure 4.7: Single stage of a Dickson charge pump. The voltage at the output is close to twice the voltage at the input after voltage drops from the diodes.

4.4 Charge Pump Analysis

In section 4.3, I discussed and evaluated three different options for converting the 24 voltage DC up to 200V. The conclusion of the discussion is that a charge pump is best suited. Hence, the following sections discuss the charge pump and the construction of it. The basic operation of such a charge pump is shown in figure 4.8 and figure 4.9. The charge pump switches between capacitors in the two stages. First C1 is charged by switching it between the voltage source and ground. Then, once C1 has been charged to the voltage source level, it is switched to be in series with the source, and C2 is switched to ground which discharges C1 into C2. The diodes are used as switches, and they prevent the capacitors from discharging back down to the voltage source or to ground. The voltage source impedance is neglected in the analysis, but this ultimately limits the maximum current that can charge the capacitors.

4.4.1 Circuit Analysis

The first stage of operation is to charge C1, shown in figure 4.8, where $\frac{V_o}{s}$ is the initial condition voltage on the capacitor. The diodes are assumed to be ideal diodes. If the capacitor is fully discharged, then $\frac{V_o}{s} = 0$. Because D2 is reverse biased, the mesh loop with C2 and R2 has no current. The current flow is simply dependent on the RC time constant between C1 and its series resistance R1. The KVL equation for the circuit is:

$$-V_{source} + V_{D1} + R_1 + V_{o1} + \frac{1}{C_1 s} = 0 \tag{4.6}$$

The voltage and current of the capacitor are found with the KVL equation and are given in equation 4.9 and equation 4.11.

$$V_{C_1}(t) = \mathcal{L}^{-1}\left[(V_{source} - V_{D1} - V_{o1})\left(\frac{1}{s} - \frac{1}{s + \frac{1}{RC}}\right) + \frac{V_{o1}}{s}\right] \tag{4.7}$$

$$= (V_{source} - V_{D1} - V_{o1})\left(1 - e^{-\frac{t}{R_1 C_1}}\right) + V_{o1} \tag{4.8}$$

$$= (V_{source} - V_{D1})\left(1 - e^{-\frac{t}{R_1 C_1}}\right) + V_{o1} e^{-\frac{t}{R_1 C_1}} \tag{4.9}$$

$$i_{C_1}(t) = \mathcal{L}^{-1}\left[\left(\frac{V_{source} - V_{D1} - V_{o1}}{R_1}\right)\left(\frac{1}{s + \frac{1}{R_1 C_1}}\right)\right] \tag{4.10}$$

$$= \left(\frac{V_{source} - V_{D1} - V_{o1}}{R_1}\right) e^{-\frac{t}{R_1 C_1}} \tag{4.11}$$

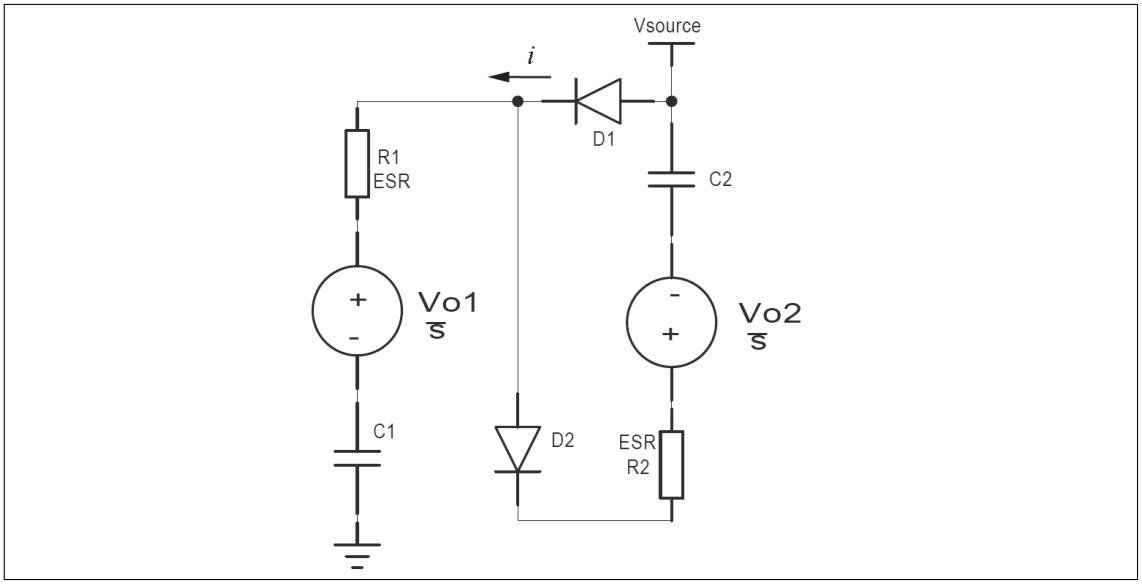


Figure 4.8: Equivalent circuit of a charge pump with C1 switched to ground and C2 switched to the voltage source. Stage 1 of operation.

The second stage of operation, in figure 4.9, transfers the charge from C1 to C2. This is done by switching the ground connection of C1 up to the supply voltage, which shifts the other node of C1 to twice the voltage source potential. D1 prevents current flow back to the voltage source, and because C2 is now switched to ground, C1 discharges into C2. Assuming $R = R1 = R2$ and $C = C1 = C2$, the voltage and current of C2 are given in equation 4.12 and equation 4.13. At the time that the circuit switches between stage 1 and stage 2 with a switching frequency f and with a duty cycle of 50%, the new V_{o1} is a constant equal to equation 4.9, where $t = \frac{f}{2}$. Note, equation 4.12 only applies when $V_{source} + V_{01} > V_{02}$; when this is not true D2 is reverse biased and blocks discharging C2 back into C1.

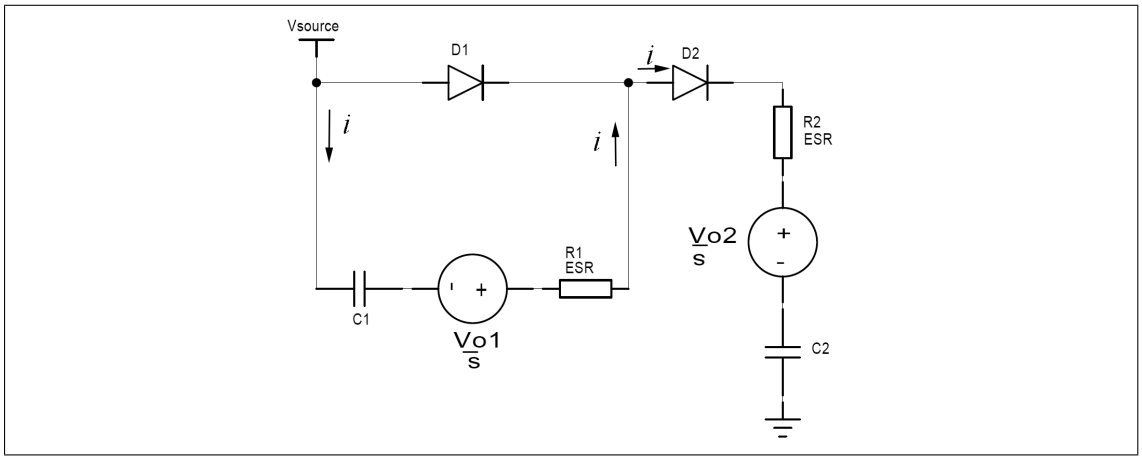


Figure 4.9: Equivalent circuit of a charge pump with C2 switched to ground and C1 switched to the voltage source. Stage 2 of operation.

$$V_{C_2}(t) = (V_{source} + V_{o1} - V_{D2})\left(1 - e^{-\frac{t}{RC}}\right) + V_{o2}e^{-\frac{t}{RC}} \tag{4.12}$$

$$i_{C_2}(t) = \left(\frac{V_{source} + V_{o1} - V_{D2} - V_{o2}}{2R}\right) e^{-\frac{t}{RC}} \tag{4.13}$$

The above equations can be computed to find the transient voltage and current in each capacitor of a circuit iteratively as the transistors switch between stages. As more circuit blocks are cascaded in series, the initial conditions of the previous capacitors pass on to the next circuit blocks. If the switching frequency is selected low enough to fully charge and discharge the capacitors in each cycle, then the intial conditions of equation 4.12 can be assumed as $V_{o1} = V_{source} - V_{D1}$ and $V_{o2} = 0V$. Also, assume $V_{D1} = V_{D2} = V_D$. In this case, at the end of the cycle, the output voltage is given in equation 4.14.

$$V_{C_2} = 2\,(V_{source} - V_D) \tag{4.14}$$

The equations show that the charge throughput of the pump is dependent on the capacitance value, series resistances, and switching frequency. The amount of current that a charge pump is able to transfer from the source voltage to its output voltage is dependent on the size of the capacitors in each stage as well as the switching frequency. A switched capacitor has an equivalent resistance, which is described in appendix D. Equation D.4 shows that increasing the capacitance value or the switching frequency lowers the equivalent resistance of the switched capacitor. Smaller capacitors can transfer the same amount of charge as larger capacitors if the switching frequency is increased for the smaller capacitors. However, if the switching frequency is too high, the capacitor voltage level will not reach near the maximum doubling of the voltage source during each switch, and so more switching cycles are needed to finally double the voltage.

4.5 Transducer Driver Design

The design for a charge pump to drive the transducer is implemented with a Dickson charge pump. Because the transducer pulses can be driven at 17 amps at 200 Volts, it was necessary to size the design for large current throughput and high voltage. A system block diagram of the driver is shown in figure 4.10. The full hardware schematics for the design are in appendix A.0.2.

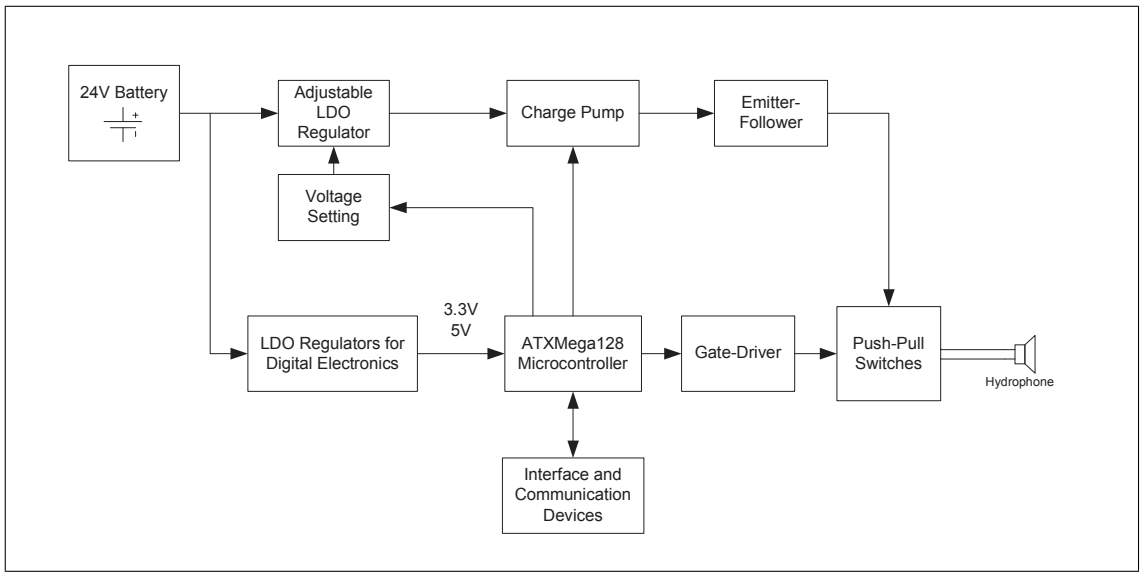


Figure 4.10: Block diagram of the transducer driver design.

As seen in figure 4.10, the 24V battery powers the LDO regulators that provide the digital 3.3V and gate-driver 15V and also feeds into an adjustable LDO regulator. The adjustable LDO regulator can be turned on/off and its output voltage can be set by the microcontroller. The output of this regulator is fed into the charge pump circuit. The charge pumps multiplies the voltage up to the drive voltage for the transducer by charging the resevoir capacitors of the emitter-follower circuit. The voltage from the charge pump is passed through an emitter-follower transistor to maintain a constant voltage at the high side of the push-pull switches. The charge in the resevoir capacitors is discharged into the transducer by the push-pull switches. The pulse signals to the gate drivers are generated by the microcontroller. The microcontroller also has the option to communicate to other devices serially, or to just interface with a trigger input and output.

4.5.1 Charge Pump Design

The charge pump circuit design is shown in figure 4.11. The charge pump consists of 10 stages of the circuit of figure 4.7 that are cascaded to multiply the voltage. The number of stages was determined by modifying equation 4.14 to find the output voltage for N cascaded stages, which is expressed in equation 4.15 from Rodes [19]. The

diode voltage drop is dependent on the current through each diode. The LM317 LDO regulator that supplies the voltage V_{DCDC} can source a maximum current of $1.5A$, and so using this current maximum with the V_D vs. I curve from the BYV 27-150 diode data sheet, the maximum forward voltage drop is approximately $0.8V$. With a regulated $V_{DCDC} = 20.8V$, a diode voltage drop of $0.8V$, and a desired $V_{Hydrophone\_Supply} = 220V$, equation 4.15 gives:

$$N = \frac{V_{Hydrophone\_Supply}}{(V_{DC\_DC} - V_D)} - 1 = 10 \tag{4.15}$$

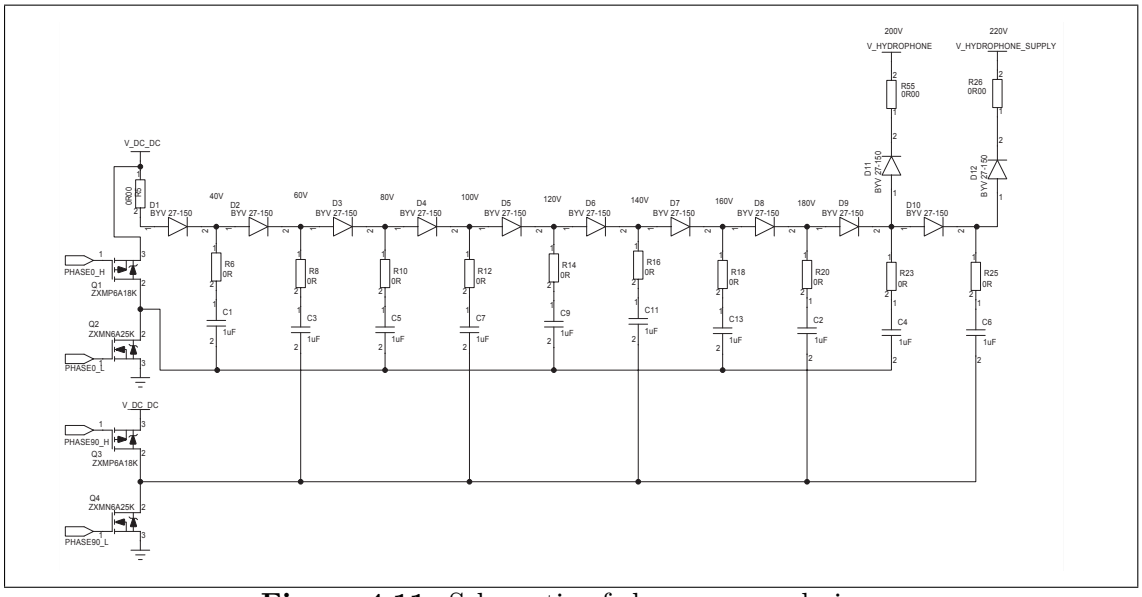


Figure 4.11: Schematic of charge-pump design.

From section 4.4.1, it was shown that increasing the value of the capacitors in the charge pump allowed more charge to be transfered per a switching cycle, so 1uF capacitors were chosen. Also, to maximize the switching frequency, the series resistance should be as low as possible so that the charging time is only limited by the source impedance of the LDO regulator. For these reasons, low ESR ceramic capacitors are selected. Finding the right capacitors was important because, as the voltage is multiplied, the energy storage of the capacitors must increase, and so their voltage rating has to be high enough for 220V. The largest surface mount capacitors that were reasonably cheap and available were $1\mu F$ at $250V$.

There is caution in using larger capacitances because the in-rush currents that must pass through the other discrete components become larger. The capacitors can be replaced with smaller values, or the LDO can be replaced with a higher current voltage regulator if necessary. Also, 0 ohm resistors are placed in series with the capacitors, and can be replaced with larger values to reduce the in-rush current. Another reason to add series resistance, as mentioned in a Texas Instruments application note [13, p. 5], is to reduce current spikes during the switching; however, they also limit the current

throughput and switching frequency of the charge pump.

The diodes chosen are ultra fast rectifiers so that, as the charge pump switches back and forth, the short reverse recovery time for the diode to turn off, $t = 25ns$, will prevent the capacitors from losing charge back into the previous stages. Also, the diodes were chosen to have a repetitive peak forward current rating of $14A$ to prevent damage during in-rush currents in each switching cycle. Because the component ratings are higher than the current that the LM317 LDO can supply, the charge-pump can be fed with a different voltage source with higher current capabilities if more current throughput is desired.

As was shown in figure 4.7, the capacitors are switched by an inverter. If the design used individual inverters, it would require 11 inverters. Instead, only two inverters with greater current capabilities are chosen, and the inverters are made with discrete power MOSFETs. Each inverter consists of one P-channel and one N-channel MOSFET. The MOSFET pair must be controlled with 2 separate signals with different voltage levels because the maximum gate-to-source voltage(V_{GS}) rating on the MOSFETs is $20V$, and if the N-channel and P-channel gates were controlled by the same signal, then when $V_{DCDC} > 20V$, the V_{GS} swing would be too high. Q1 and Q2 from figure 4.11 make up an inverter circuit, where Q1 is a P-channel MOSFET that is turned on when its gate voltage drops $V_{GS_{th}}$ below V_{DCDC}. Similarly, Q2 turns on when its gate voltage rises $V_{GS_{th}}$ above ground. This also applies to the Q3 and Q4 inverter pair.

The MOSFETs were selected to have a maximum drain-source resistance of 0.5 mohms and a maximum drain current, $I_D = 10.4A$ so that their series resistance in the charging of the capacitors are minimal. The MOSFETs were also specified to have a drain-source voltage of 60V, safely above the maximum drain-to-source voltage (V_{DS}) that the design will have of $V_{DS} < 24V$.

A timing diagram for the 4 switching signals is shown in figure 4.12. The signals PHASE0_H and PHASE90_L are opposite of each other, and PHASE0_L and PHASE90_H are opposite of each other. These signals are created by the microcontroller through open-collector BJTs that shift the voltage from digital $3.3V$ to V_{DCDC} and $15V$ so that they are high enough to turn on/off the MOSFETs. The microcontroller's corresponding signals must be inverted because of the open-collectors. An advantage of using four signals is that it enables each switch state to be controlled individually by the microcontroller. Individual control allows the dead-time-insertion features of the AWG peripheral in the XMEGA timers to be used to ensure that only one switch in each inverter is on at a time to avoid shoot-through current. To ensure that the charge-pump is operating as fast as possible, the dead-time-insertion period t_{DTI} should be much less than the overall period of the square-wave. Since the microcontroller controls the switches, the software determines when the voltage multiplication takes place, and it can also shut down the charge pump at any time.

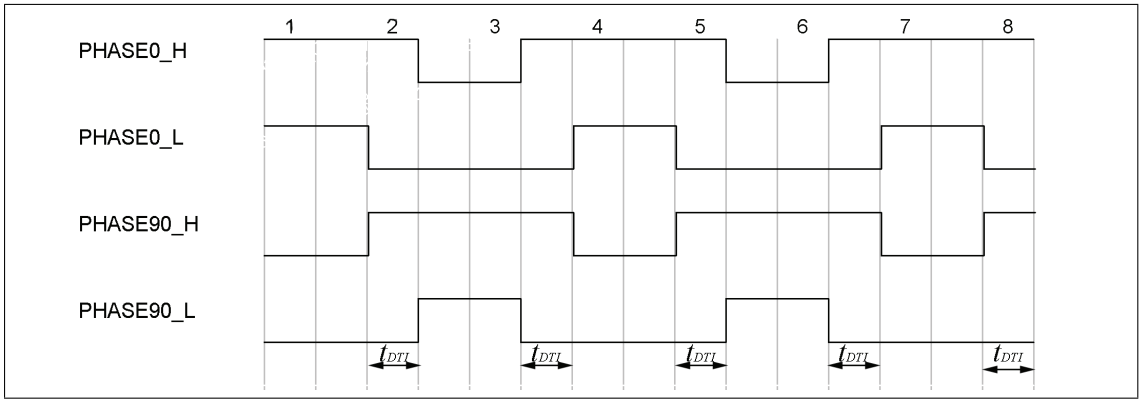


Figure 4.12: Timing diagram for switching the charge pump's MOSFETs.

4.5.2 Charge Pump PSPICE Simulation

Simulations of the charge pump were performed to verify that the circuit will give an output voltage as predicted in section 4.4.1. The simulation circuit is shown in figure F.2. SPICE models were downloaded from the vendor websites.

The circuit was simulated with V_{GS} at $5V$ and at $10V$. Also, the switching frequency was varied from $50kHz$ up to $300kHz$. The simulations at $5V$ are shown in figure 4.14 and the simulations at $10V$ are shown in figure 4.15. The simulations show that the final voltage reaches about $221 - 222V$, which is very close to the voltage predicted by the equations. The possible reason for the slight deviation is that schottky diodes were used in the simulation, which means that the diode drops were lower.

The simulations also show how the switching MOSFET drain-source resistance and switching frequency affected the charge-througput of the charge pump. It was found that the $R_{DS_{on}}$ of the MOSFETs introduces a decrease in the charge throughput of the charge pump. Also, higher switching frequencies increase the charge througput of the charge pump until a certain frequency maximum, and then the throughput begins to decrease again as frequency is increased higher.

The switching frequency at $100kHz$ for the two gate voltage cases in figure 4.14b and figure 4.15b shows that driving the gate to a higher voltage, well above the gate threshold voltage, forces the charge pump to charge much faster, where the effective time constant is less than half. The explanation is that when the gate is driven to $10V$, the MOSFET is in full enhancement, and so its R_{DS} is lower. The decrease in the R_{DS} means that the series resistance with the capacitors is lowered, and so more charge is able to enter the capacitor during each switching cycle. For this reason, the design drives the gate voltage as close to the V_{GS} max of the MOSFET as possible at $15V$.

In both figure 4.15 and figure 4.14, the charge pump charges the slowest at the lowest switching frequency. The charging time decreases as the frequency is increased, but as the frequency goes above $100kHz$, the charging time actually starts to increase again. From section 4.4, it was shown that increasing the frequency that a capacitor is switched should increase its charge throughput. So, a possible reason for this limit in switching frequency is that as the MOSFETs begin switching faster, they are dissipating more power and this begins to be a factor for how much charge is transferred at higher frequencies. From Elbanhawy [8], equation 4.16 shows that a term in the power dissipation is dependent on frequency, where I is the drain current, R_{on} is the MOSFET source-drain resistance, Q_G and V_G are the gate voltage and charge, t_r and t_f are the rise and fall time of the switching, D is the duty cycle, and f is the switching frequency. So, from the simulations, it is decided to switch at a frequency of 100kHz.

$$P = I_{RMS}^2 R_{on} D + \underbrace{f\left[Q_G V_G + \frac{1}{2}(t_r + t_f) I_{RMS} V_s\right]}_{\text{frequency dependent power loss limits the upper frequency for charge throughput}} \tag{4.16}$$

While the unloaded charge pump simulations help to determine operating parameters of the charge pump like the optimum switching frequency and capacitor values, a loaded charge pump simulation shows how long it would take to charge the capacitor banks on the emitter-follower circuit. This simulation was done with a $47\mu F$ load, and it is shown in figure 4.13. A problem with simulating this is that the time steps necessary for simulating the switching are in the nanosecond range, while the time to charge the $47\mu F$ is in the millisecond range. This resulted in a simulation requiring many data points, and the simulator runs out of memory before the capacitor can be fully charged. A way to reduce the memory needed was to collect only datapoints for the node voltages during the simulation. With the time that the simulation ran, it shows that the load capacitor is taking tens of milliseconds to charge, which is much slower than in the unloaded simulations. Recall that a switched capacitor has an equivalent resistance, and so the circuit can be considered as an RC circuit, and the time constant of the charge pump and capacitor load can be estimated in equation 4.17, where τ is the time constant, N is the number of switched capacitors, C is the capacitance of each switched capacitor, f is their switching frequency, and C_{load} is the load capacitance.

$$\tau \approx \frac{N}{fC}\left(C_{load} + NC\right) \tag{4.17}$$

The graph in figure 4.13 shows the cursor measurements at the estimated 2τ and 3τ times, and these cursor points show that the load capacitor has charged quite close to what the time constants dictate, but not exactly. This could be because of the nonlinearities in the diodes and MOSFETs. After 5τ, the load capacitor should be close to its maximum voltage, which is after about 30 to 40 ms. Of course, this charging time

could be longer because of the source resistance of the LDO supplying the current. A conclusion from this simulation is that the transducer should only be driven about once a second at its maximum source-level, so that the charge pump has plenty of time to recharge the capacitor banks. If the transducer is driven at a lower source level than the maximum, the circuit should be able to drive it in much shorter time intervals.

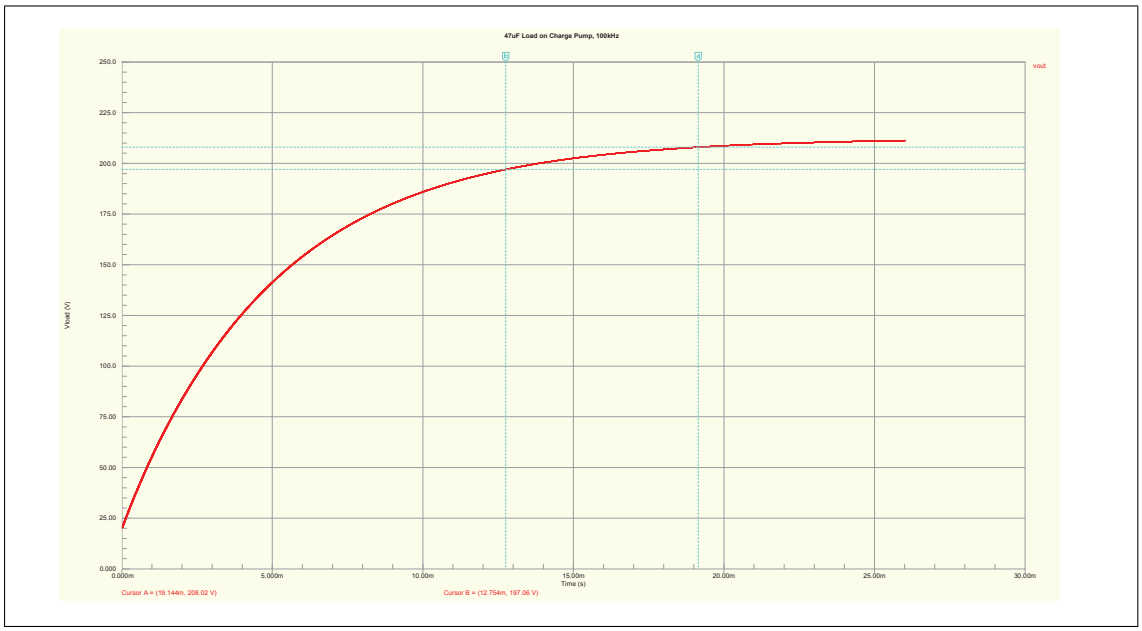
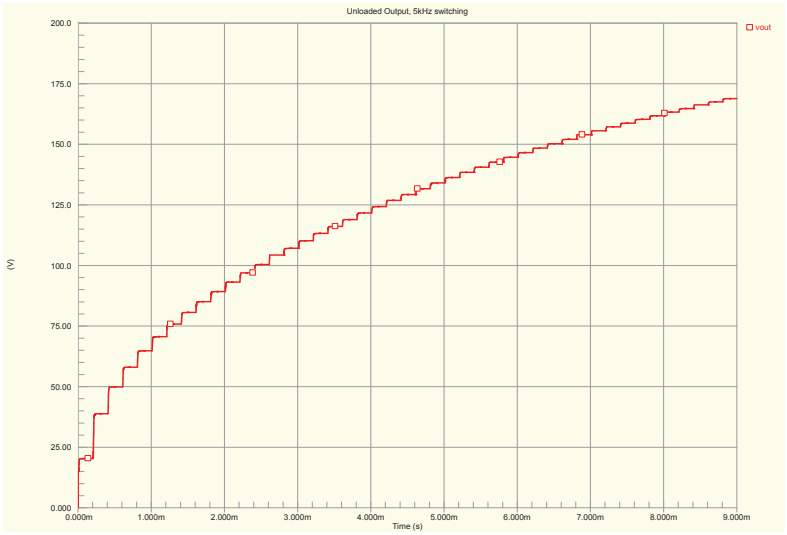


Figure 4.13: Simulation of the charge pump charging a 47uF capacitor.

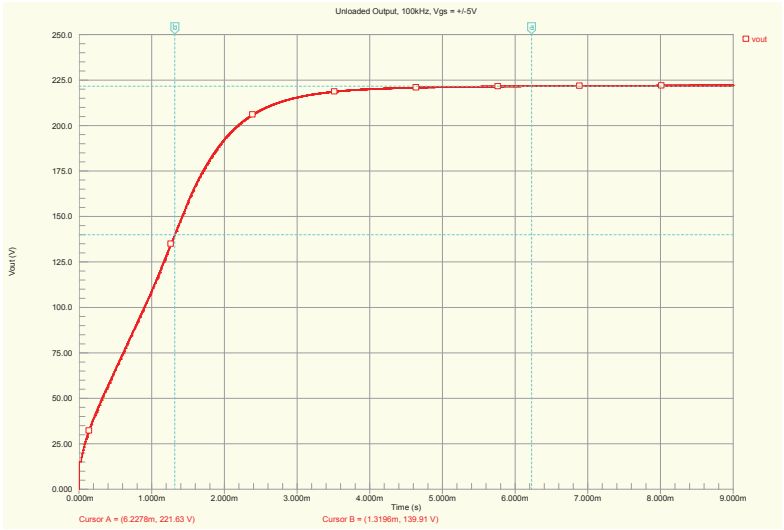
4.5.3 Emitter-Follower Output Regulation

The charge-pump's output current is limited by its switching frequency, capacitor size, and by the source impedance of the LDO regulator. The LDO regulator's constant power output of $(20.8V)(1.5A) = 31.2W$ means that it could never continuously source the $425W_{RMS}$ that the transducer draws during its pulse output. To be able to source the power, the energy is stored in resevoir capacitors and then discharged during the pulse. The charge pump is tapped at two voltage levels from figure 4.11. Each tap goes into capacitor banks that must store enough energy to supply the pulse.

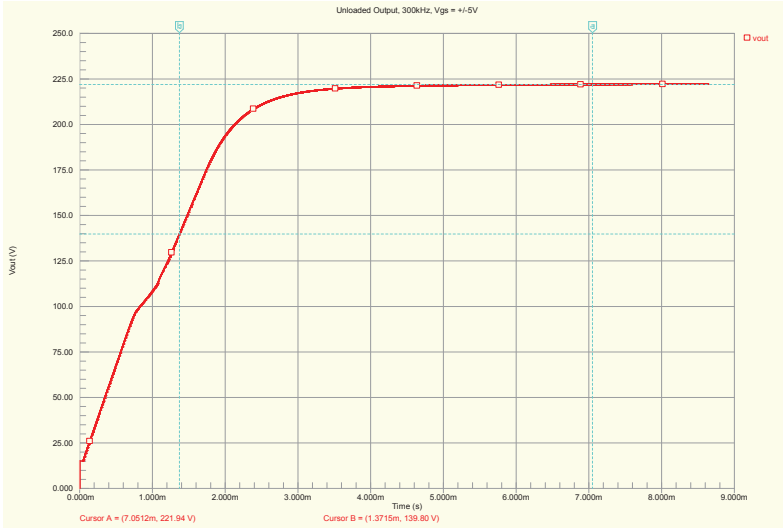
The reason for two banks of capacitors at different voltage levels is because the lower voltage bank is used as a reference voltage, while the higher voltage bank supplies the majority of the current. If only one capacitor bank were discharged from the desired voltage without regulation, then each successive pulse of the 12 pulses into the transducer has a decay in its voltage level as the voltage of the capacitor drops by its RC time constant. In order to maintain a constant voltage and current level at the output as the capacitors are discharged, the banks are connected to an emitter-follower of figure 4.16.



(a) 5kHz, $V_{GS} = 5V$

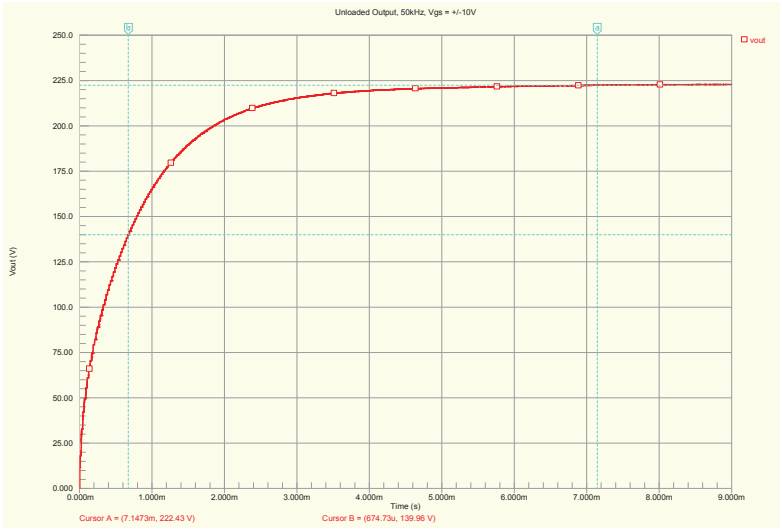


(b) 100kHz, $V_{GS} = 5V$

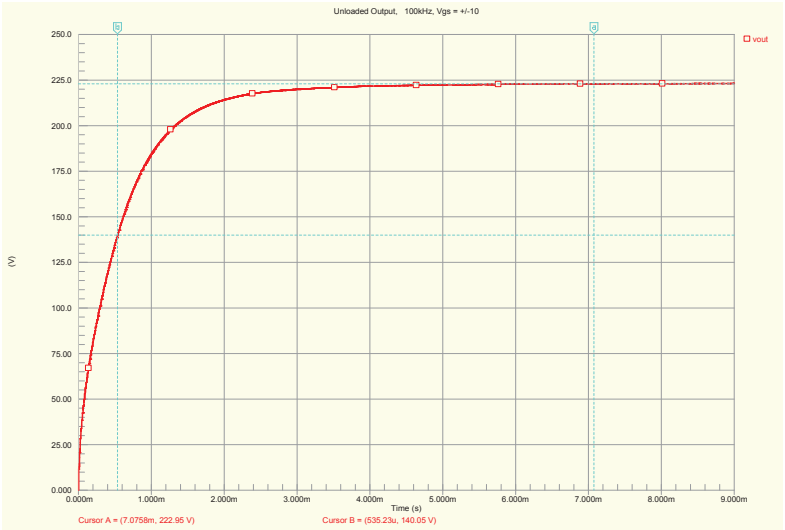


(c) 300kHz, $V_{GS} = 5V$

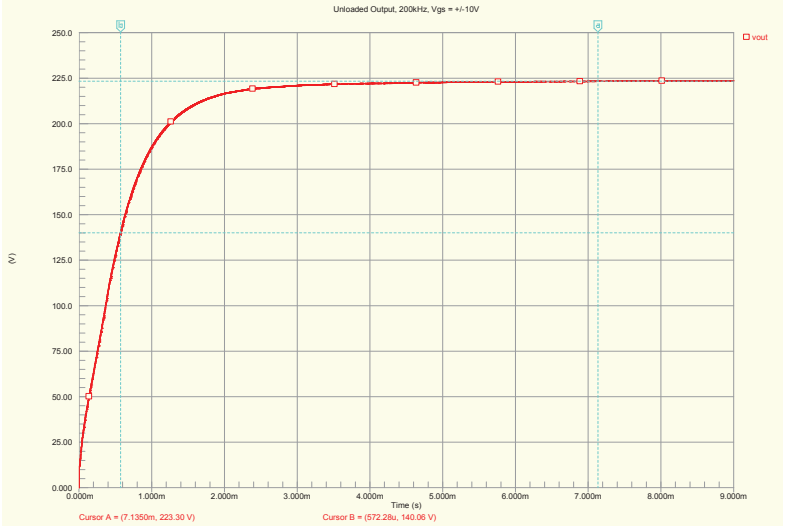
Figure 4.14: Charge pump simulations with MOSFETs driven with 5V. Fastest time constant is 1.319ms at 100kHz.



(a) 50kHz, $V_{GS} = 10V$



(b) 100kHz, $V_{GS} = 10V$



(c) 200kHz, $V_{GS} = 10V$

Figure 4.15: Charge pump simulations with MOSFETs driven with 10V. Fastest time constant is 535μs at 100kHz.

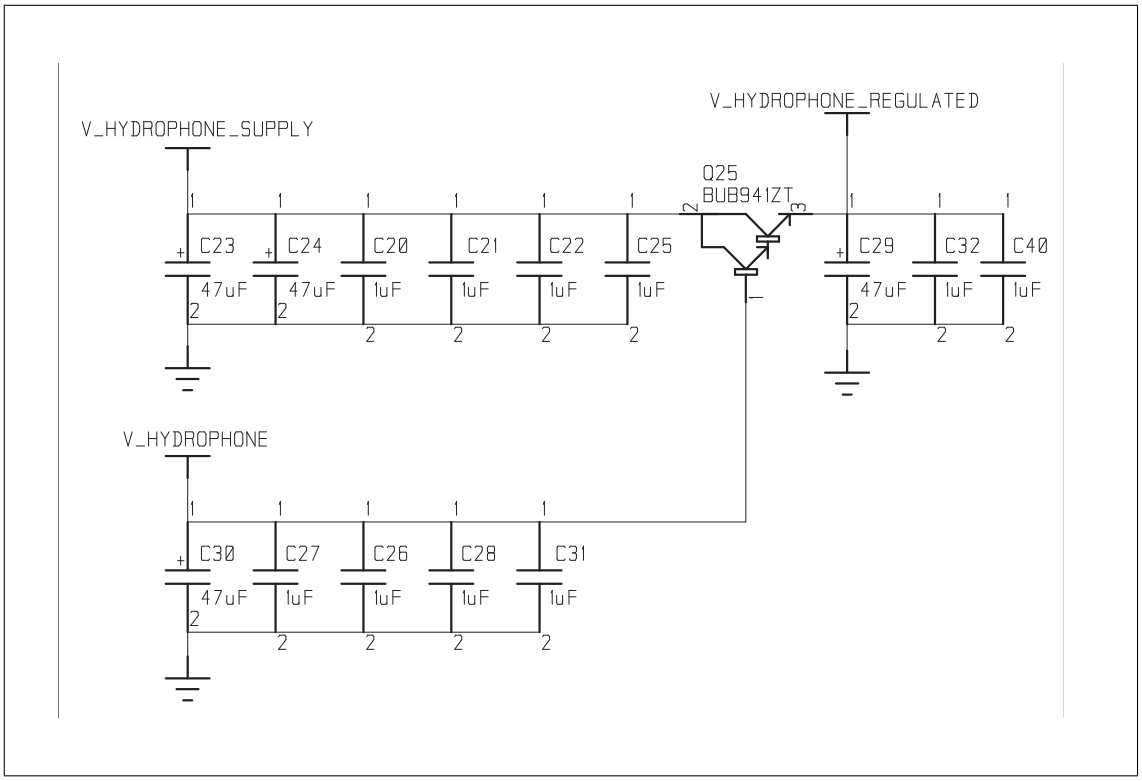


Figure 4.16: Schematic of the emitter-follower regulator with the high voltage capacitor banks.

The emitter-follower regulator is similar to an emitter-follower amplifier; however, DC biasing is not necessary in this application since only a DC level is required at the output. When $V_{Hydrophone}$ is applied at the base of the NPN transistor, the current flows to the emitter, and then the emitter voltage is equal to the base voltage minus the base-emitter saturation voltage drop, $V_{be_{sat}}$*. If the voltage of the capacitor bank at the collector, $V_{Hydrophone\_Supply}$, is not charged above the emitter voltage, the transistor acts as an open-collector until the capacitor banks have been filled at to the voltage $V_{Hydrophone}$. Once $V_{Hydrophone\_Supply}$ is higher than the emitter voltage, the majority of the current out of the emitter comes from the collector source.

The BJT requires a high current gain h_{FE} because the voltage source at the BJT base is the voltage reference, and any load on it will force its voltage level to drop, which brings the emitter-follower out of regulation. As h_{FE} is increased, more current is drawn from the higher voltage capacitor banks than from the lower voltage banks. A darlington-pair transistor was chosen for this reason, which increases the current gain greatly compared to a single transistor. The BUB941ZT darlington-pair's original application is to be a high voltage iginition coil driver, and so it has high voltage and current ratings necessary for this design. It has a minimum current gain of $h_{FE} = 300$. This transistor was also

*The voltage drop can actually be less than $V_{be_{sat}}$ because the transistor is operating in a negative feedback mode.

chosen for having a high $V_{CE_{off}}$ of $350V$, well above the maximum possible voltage.

The currents drawn by the BJT are given in equation 4.18 and equation 4.19 with a known current gain of the BJT, and assuming the emitter current is $I_e = 17A$.

$$I_b = \frac{I_e}{h_{fe} + 1} = \frac{(17A)}{(300 + 1)} = 56.5mA \tag{4.18}$$

$$I_c = I_b h_{fe} = (34.9mA)(300) = 16.95A \tag{4.19}$$

When the transducer load is pulsed, 16.95A is passed through the BJT collector. At the beginning of the pulse, V_{ce} is at approximately $20V$, and lowers as the capacitor bank is discharged. This initial voltage difference at 16.95A gives a considerable amount of power dissipation across the collect-emitter, but figure 4.17 shows that even at the $1ms$ pulse curve, the collector current can be as high as $30A$ when the voltage drop $V_{ce} = 20V$.

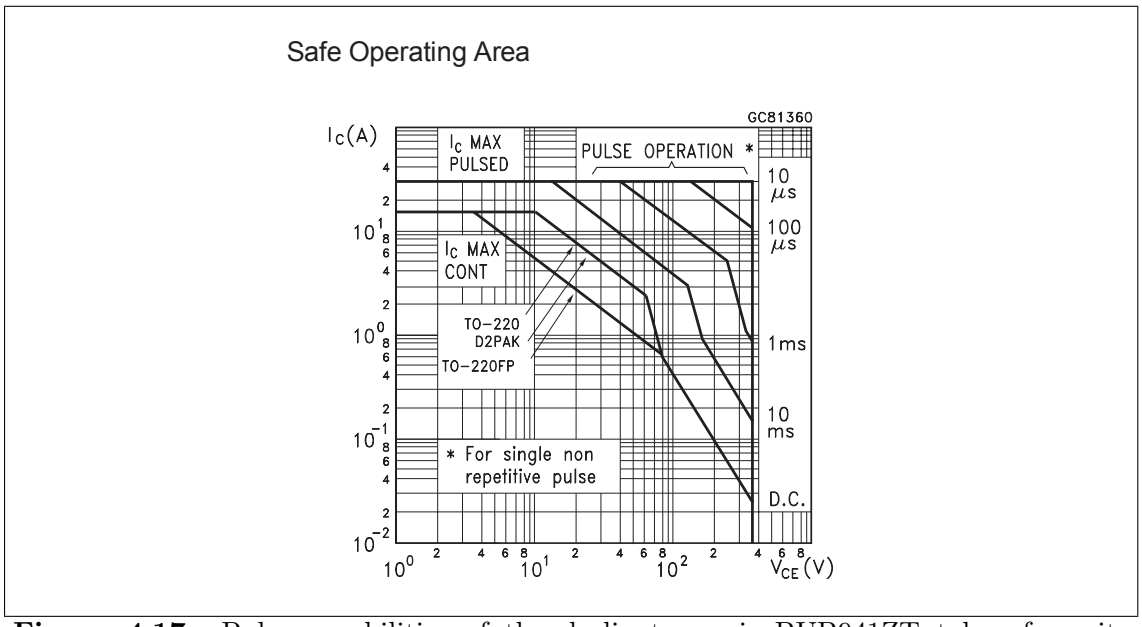


Figure 4.17: Pulse capabilities of the darlington pair BUB941ZT taken from its datasheet.

The capacitor banks must be selected to have enough charge for the transistor to draw current from them without dropping below the regulation voltage. The capacitor bank at $V_{Hydrophone\_Supply}$ will be charged to 220V, and should not drop below $V_b - V_{be} + V_{ce_{sat}}$ during the duration of the pulses. Change in voltage and change in time calculations are given in equation 4.20 and equation 4.21. The discharge time is the time that the $120kHz$ transducer pulses at a period T for 12 periods.

$$V_2 - V_1 = (V_b - V_{be} + V_{ce_{sat}}) - 220V = 199.3 - 220V \tag{4.20}$$

$$t_2 - t_1 = 12\frac{T}{2} = (12pulses)\frac{(4.167\mu s)}{(pulse)} = 50\mu s \tag{4.21}$$

From appendix E it is shown how to find the required capacitance when a constant current is drawn from a capacitor bank. Equation E.5 is used to find the minimum capacitor bank value in the design, and the value is found in equation 4.22.

$$C = -\frac{I(t_2 - t_1)}{V_2 - V_1} = -\frac{16.95A(50\mu s)}{199.3V - 220V} = 40.9\mu F \tag{4.22}$$

This capacitance is a minimum, and so extra capacitance is added to ensure enough current is available during switching losses. Also, extra capacitors reduces the total ESR of the capacitor bank. The capacitor bank for the base of the BJT is chosen similarly, and finally some bypass capacitors are also included on the emitter of the transistor to allow even more current capabilities during the pulse as well as protect the transistor from any current spikes that the pulse switching may create. The capacitors used are a mix of aluminum electrolyic and ceramic capacitors, and the $47\mu F$ are sized for a $400V$ rating because as capacitors are charged to near their maximum voltage rating, their capacitance is derated and they can only store about half of their energy capacity.

4.5.4 Pulse Generation

A circuit must generate the pulses to the transducer terminals now that the charge pump and the emitter-follower circuitry provide a constant high voltage source. This is implemented using a push-pull circuit. The push-pull circuit is similar to the bridges circuits used in the BLDC motor controller of chapter 3. The choice of the push-pull switches was guided by the high voltage requirement, the high current requirement, and the switching frequency. The two technologies considered for the switches were the MOSFET and the IGBT. A comparison of the devices are given by Blake and Bull [3]. An IGBT can be represented as a switch that has the high voltage and current capability of a BJT with the high input impedance of a MOSFET. Between the two technologies, MOSFETs are usually better for higher frequency applications, where modern IGBTs start to drop off in performance in the lower hundreds of kHz. IGBTs become the better choice as voltage across the switch gets up to about 600V. Also, at high currents, the losses between the two technologies depend on the $R_{DS_{on}}$ resistance of the MOSFET versus the diode drops of the IGBT. Because the supply voltage is only 200V, and the pulse frequency is $120kHz$, the MOSFET is a better fit for the design.

A BJT push-pull circuit was also considered; however, it is rather impractical because constant current must be drawn at the base while the switches are turned on, which can

discharge the capacitor banks too much. The BJT push-pull circuit and the IGBT switches were added in the schematic design so that the three technologies could be compared, and also if the board should drive different transducers ever, it could have the option to use the other technologies if the voltage and frequency requirements changed.

The MOSFET selected, IRFR12N25D, has an operation region shown in figure 4.18, and with a total pulse high of $50\mu s$, the operation is just within the safe limits.

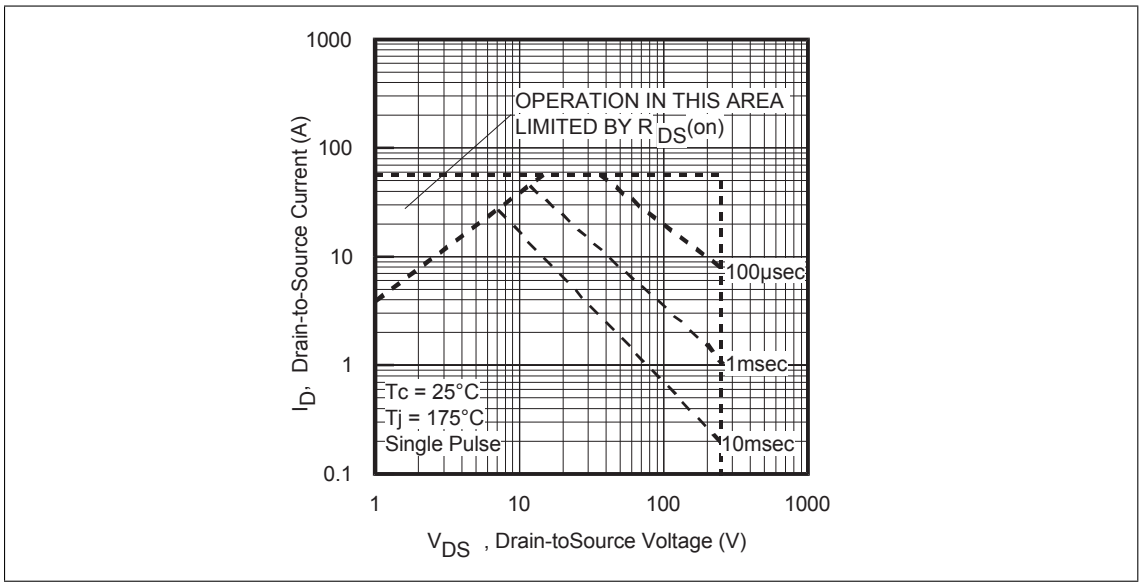


Figure 4.18: Safe area of operation of the MOSFET switch.

The MOSFET gates must be driven from a low impedance source capable of sourcing and sinking sufficient current to provide for fast insertion and extraction of the controlling charge at their gates [2] . Using a gate-driver allows the MOSFETs to be switched on with fully enhanced conduction channels for maximum current. The gate-driver chosen is the same as in the BLDC motor controller, the IR2112. This driver is controlled by the microcontroller, which gives the high-low outputs for the $120kHz$ pulse.

4.5.5 Pulse PSPICE Simulations on the Transducer Load

The capacitor banks, emitter-follower regulation, and push-pull FETs were simulated in PSPICE to verify that the parts could provide a 220V pulse at 120kHz with 17A without the voltage dropping after 12 pulses. The simulation circuit is shown in figure F.3. SPICE models were downloaded from the vendor websites. The emitter-follower darlington transistor did not have an available spice model, so a similar transistor model was used, the BUB323ZT4. An 11.76 ohm load is used to represent the impedance of the transducer. The capacitor banks are charged through diodes by pulsed ideal voltage sources for $200\mu s$ to represent the initial charge pump steady state voltage, and then the

MOSFETs begin to switch at $200\mu s$.

The pulsing simulation is shown in figure 4.19. The pulses have a voltage amplitude of about $200V$. The current across the load is approximately $17A$, and the power in the pulses is just over $3.4kW$. This closely matches the desired characteristics of the electrical pulses to the transducer which were specified in table 4.2. From figure 4.20, the pulses begin to drop in amplitude after about $100\mu s$. This is because the voltage in the capacitor banks at the emitter-follower's collector terminal has dropped below the regulated voltage at this time. The amplitudes of the pulses are maintained constant for 12 consecutive pulses before the drop becomes noticeable.

The simulation of figure 4.21 was made to validate that the emitter-follower is regulating the output voltage and that the capacitor banks are working as expected. It shows the voltage level of the collector, base, and emitter of the BJT. The collector immediately begins to drain charge out of its bank capacitors, and goes from $220V$ to $200V$ in $100\mu s$, which confirms the predicted capacitance needed from equation 4.22. Once the collector voltage has dropped to $200V$, the simulation shows that the base and emitter voltages begin to drop because now those capacitor banks are required to source the current that the collector no longer can supply. Also, during the pulses, the emitter voltage is fluctuating slighty, and this is because the bypass capacitor on the emitter terminal is charging and discharging slightly for each pulse before the BJT responds with more current from the collector.

4.5.6 Microcontroller Interface and Control

A microcontroller provides the signals for the switching FETs of the charge pump as well as the pulses to the gate drive chip that controls the push-pull circuit to make the pulses on the transducer. These signals can be generated with the AWG timer peripherals of PORTE and PORTC.

The microcontroller can also control the source level, SL, of the hyrophone by controlling the voltage output of the LDO regulator that supplies the charge pump. The regulator has an adjustment pin on it that determines the output voltage. In figure A.10, it shows that there are three options for determining the regulation level of this output. The first is simply to use the resistor divider that sets a fixed voltage output, and so R38 is populated with an appropriate resistor value, while R41 and R42 are depopulated. This is the recommended method in the LDO datasheet.

If software control of the source level is needed, two additional methods are available in the design. The second method is to use a PWM signal from the microcontroller. The opamp U6 is configured as a low pass filter with an output that can be driven by a $30kHz$ PWM signal from the micrcontroller. The low-pass filter removes the $30kHz$,

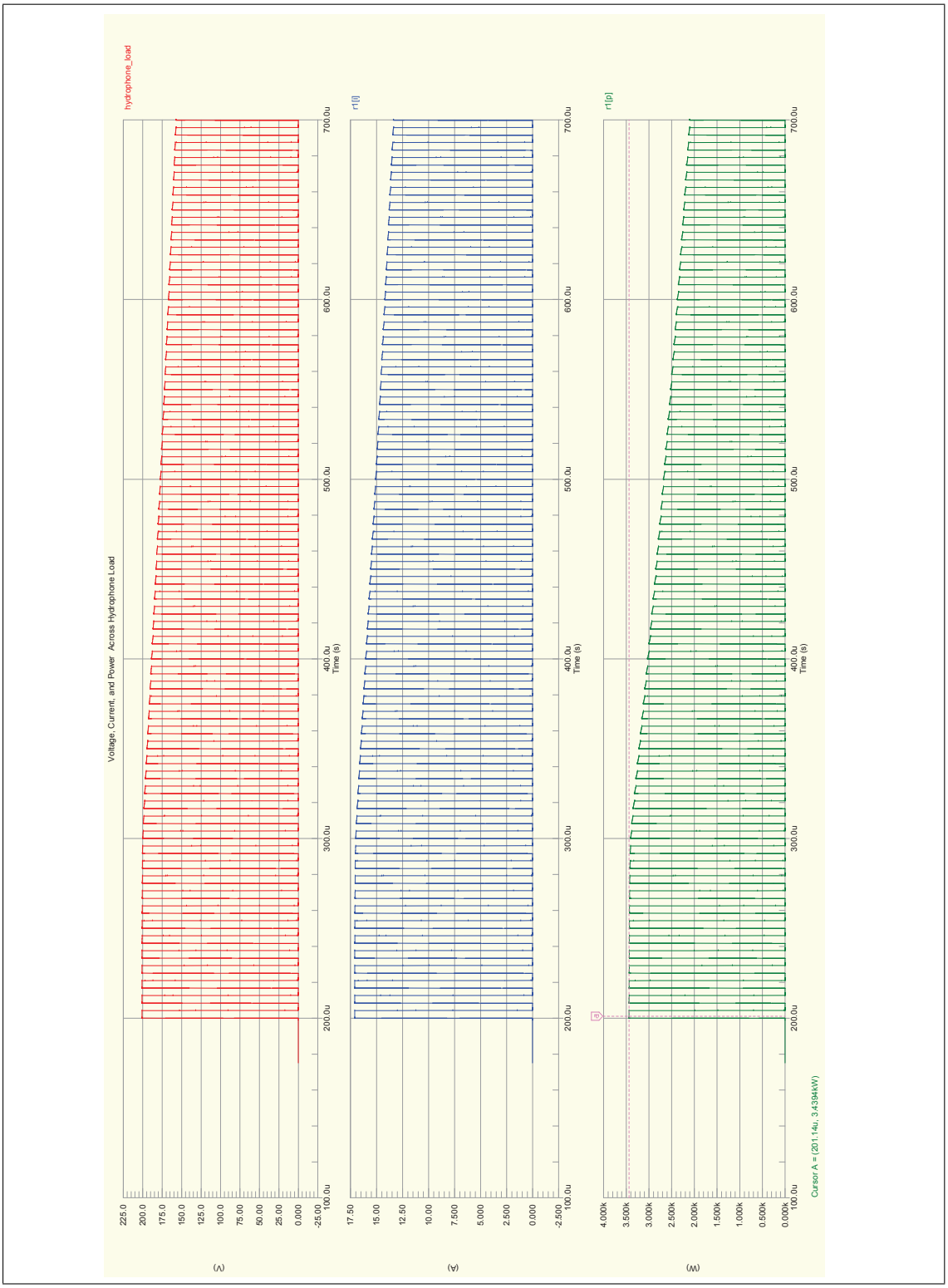


Figure 4.19: Simulation of the transducer being driven, showing the voltage, current, and power into the load.

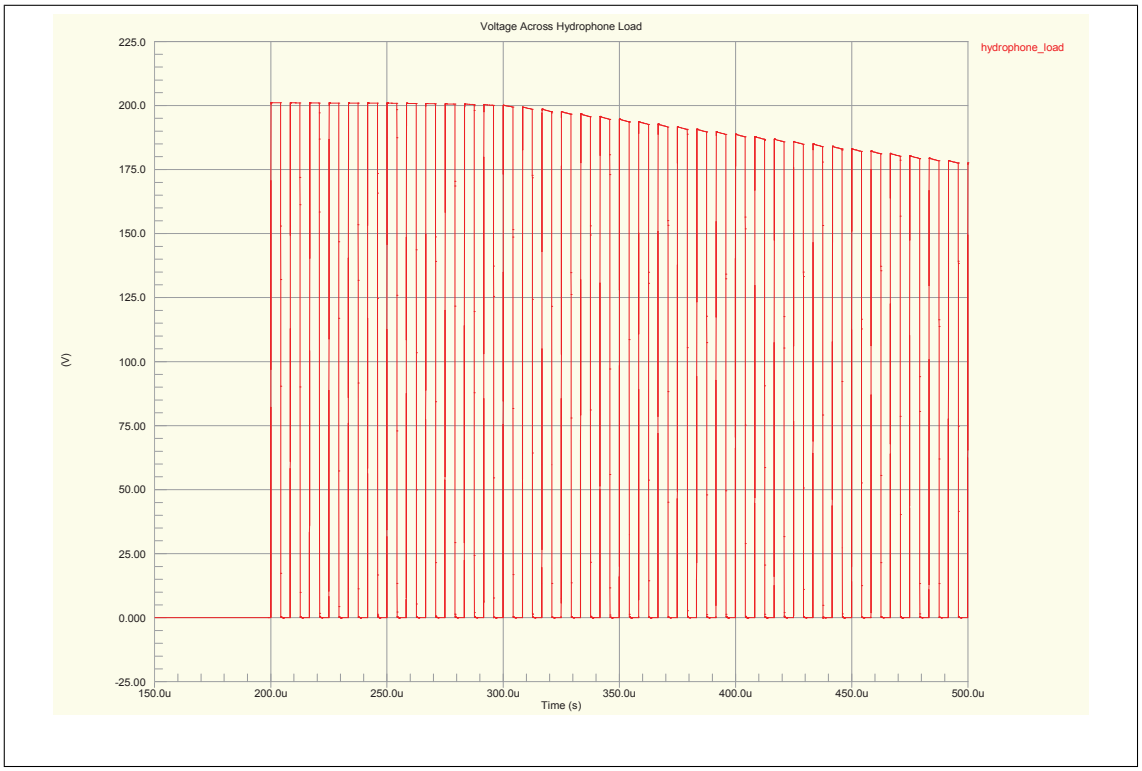


Figure 4.20: Simulation of the transducer being driven, showing that the voltage level begins to drop after the 13th pulse.

and has an output of an averaged DC voltage level proportional to the duty cycle of the PWM signal and the $3.3V$ amplitude. The opamp is powered directly from the battery and has a gain of 5.87, so that a 100% duty cycle of $3.3V$ from the microcontroller gives a voltage output V_{opamp} of $19.371V$ as determined by equation 4.23, where the duty cycle is given as a percentage.

$$V_{opamp} = (DutyCycle)(3.3V)(5.87) \tag{4.23}$$

The LM317 LDO maintains a reference voltage of $1.25V$ across resistor R39, and the regulated output voltage changes to keep this reference voltage constant. So the output voltage is equal to the $1.25V$ reference plus whatever the voltage drop is between R39 and the adjustment pin. Instead of using a resistor to create this drop, the output of the opamp creates the voltage drop, and so the output of the LDO, V_{DCDC}, is determined in equation 4.24. If the duty cycle is 0, the voltage drops too low for regulation and the LM317 shuts off, effectively turning off the charge pump.

$$V_{DCDC} = V_{opamp} + 1.25V \tag{4.24}$$

The third method is to use a digital potentiometer as the adjustment resistor. The problem in finding a suitable digital potentiometer was that the available parts were only

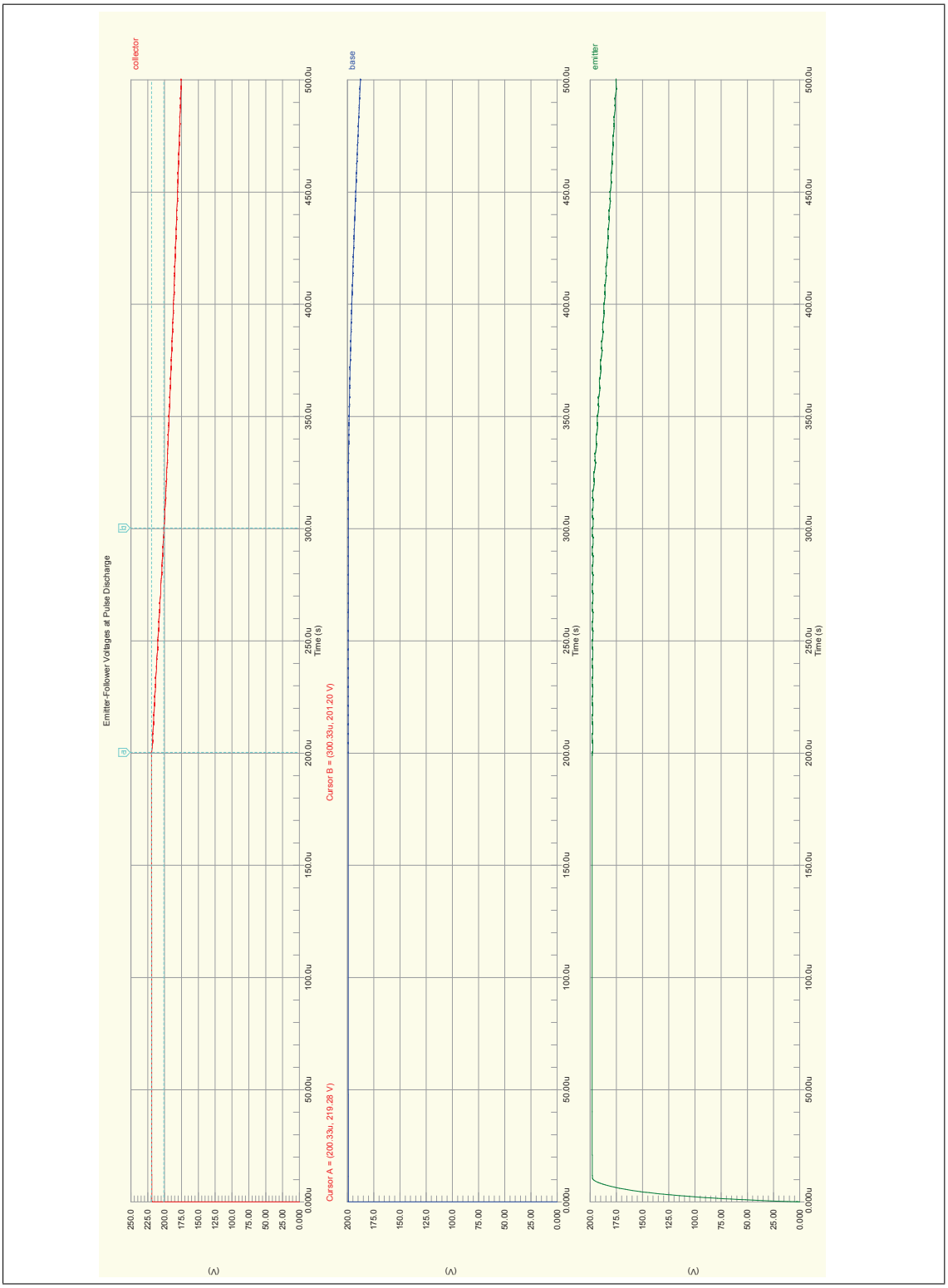


Figure 4.21: The voltage levels of the collector, base, and emitter of the BJT as it discharges the capacitor banks while driving pulses to the transducer.

rated for the digital voltage across the terminals. Since the voltage across the adjustment resistor can be over $19V$ while the digital signal is only $3.3V$, these parts could not be used, and so discrete components were used to make a digital potentiometer. The MOSFET pairs Q5,Q6,Q8,Q9 are switched in combinations from the microcontrollers POT1-POT8 signals. The combinations determine the overall resistance from the LM317's adjustment pin to ground, and, in turn, determine the regulated output of the LDO.

There are 64 valid combinations with the digital potentiometer, and the resistors values were selected by trial and error in a spreadsheet formula to give a linear voltage range over the 64 combinations. These combinations were used to show the range that the charge pump output would have in figure 4.22. In practice, more standard resistor values would need to be chosen, and so the desired output curve will not be as linear. This method is rather complex and was not simulated, so it was only added as a backup method. The first two methods should be used instead, but the third method served as an example of designing a digital potentiometer for voltages greater than the digital voltage.

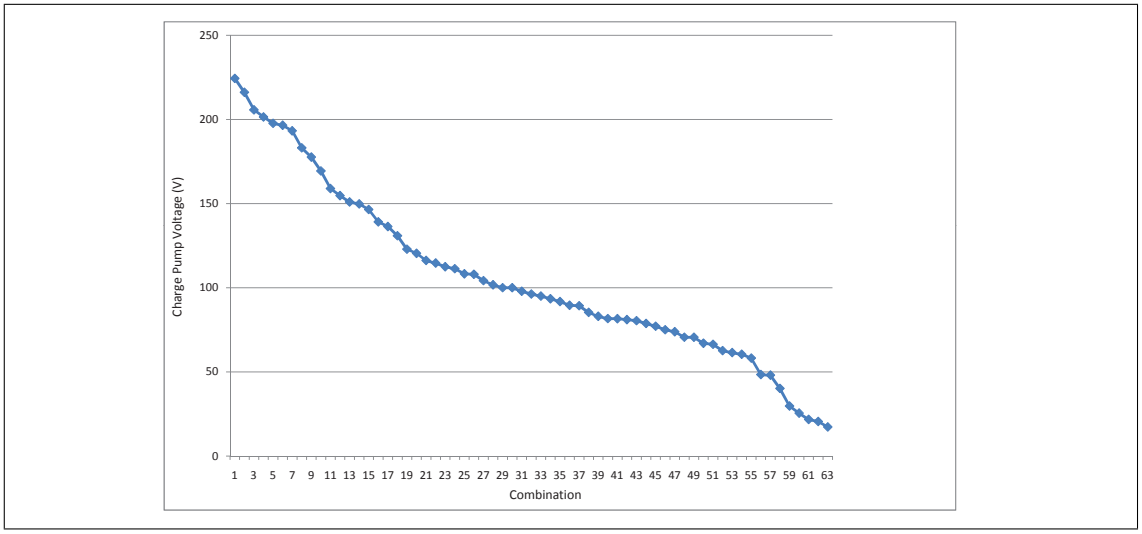


Figure 4.22: The output voltage of the charge pump as the resistor combinations of the digital potentiometer vary.

All three methods were designed in the schematics and layed out on the gerber files of the PCB to give the design flexibility, but none have been tested other than the first method, where just a resistor is used, since this circuit was used in the BLDC motor controller circuit design successfully and is also recommended in the datasheet. The opamp was simulated in PSPICE to verify the low-pass filter passed only an average value of the PWM signal, but it has not been tested on a circuit board.

4.5.7 Setting the Transducer Source Level

The source level of the transducer can be calculated as a function of the the resistor value of R38 for method one in equation 4.29, and as a function of the duty cycle for method two in equation 4.33. The real impedance R, efficiency E, and directivity DI ratings of the ES 120-4x10 specification in figure 4.3, and equation 4.25 from Tucholski [23] are used to make the calculations. Note that R38 cannot not be made any higher than what is dictated by the LM317 LDO's minimum drop out voltage of $3V$.

$$SL = 171.5 - 10log(R) + 20log(V_{rms}) + 10log(E) + DI \tag{4.25}$$

$$\begin{aligned} SL &= 171.5dB - 10log(11.76) + 20log(V_{rms}) + 10log(0.5) + 29.5dB && (4.26)\\ &= 187.3dB + 20log\left[\frac{(10)(V_{DCDC} - 0.7)}{\sqrt{2}}\right] && (4.27)\\ &= 187.3dB + 20log\left[\frac{(10)\left(1.25\left(1 + \frac{R38}{1.47k}\right) - 0.7V\right)}{\sqrt{2}}\right] && (4.28)\\ &= 187.3dB + 20log\left[(R38)(0.0085) + 3.89\right] && (4.29) \end{aligned}$$

$$\begin{aligned} SL &= 171.5dB - 10log(11.76) + 20log(V_{rms}) + 10log(0.5) + 29.5dB && (4.30)\\ &= 187.3dB + 20log\left[\frac{(10)(V_{DCDC} - 0.7)}{\sqrt{2}}\right] && (4.31)\\ &= 187.3dB + 20log\left[\frac{(10)\left((DutyCycle)(19.371V) + 1.25V - 0.7V\right)}{\sqrt{2}}\right] && (4.32)\\ &= 187.3dB + 20log\left[(DutyCycle)(136.97) + 3.89\right] && (4.33) \end{aligned}$$

4.5.8 Transducer Driver Implementation

The design from section 4.5 was implemented as two separate board designs to isolate the microcontroller related functions from the charge pump and push-pull circuit so that software can be run on the microcontroller while isolated from the the high voltage components. The boards are connected through a header that connects the supply voltages, switching signals, and pulse signals to the high voltage circuits. The circuit board layouts are shown in appendix A.0.2. The boards are layed out with two routing layers, and, as a result, many vias were placed under the microcontroller for fanout of all the signals. This was not ideal for routing, and a four layer board would have been a better approach. Additional copper spacing was used in the high voltage circuit areas to avoid voltage creepage or any possible arcing. This board has warning labels on it in the silk screen layer because voltages higher than $200V$ could cause injury to someone if they contact the components. When working with this circuit, care should be made not to

touch the high voltage capacitor terminals and their connected circuits. The layout is finished, but boards were not ordered and the design has not been built. The gerber files have been generated and the board is ready to be ordered, built, and programmed in the future.

The I/O ports of the micrcocontroller that go out to headers, such as CN4 in figure A.12 are isolated with zener diodes to prevent ESD or over voltage conditions from damaging the microcontroller. Also, the device is dependent on having the battery supply at 24V. As the battery voltage drops, the device will not give the expected voltage levels at the charge pump. A zener-protected resistor divider connected between the battery voltage and the ADC port of the microcontroller was included in the design so that a voltage monitoring feature could be added to adjust the regulated voltage or to completely shut down the charge pump if the battery voltage was too low or high.

Lastly, the sections of this chapter gave numbers for maximum values, so before the driver is tested on a transducer, the appropriate values should be lowered greatly to avoid possible damage to the transducer. Operating the device at the same power level as the EY500 of $60W$ would mean that the resistor R38 should be chosen to be $7.37k\Omega$ or that the duty cycle to the opamp should be about 43% . The driver will have better performance characteristics when the power supplied to the transducer is less than its maximum, such as a faster charge time of the resevoir capacitors which reduces the wait time between sending pulses.

4.5.9 Conclusion

This chapter shows what kind of electrical requirements are required to drive the Simrad transducer for the pulsed signal. DC-DC converter methods were discussed to select a way to generate this electrical power, and the dickson charge was chosen. Some analysis of the dickson charge pump circuit showed the important component selection for the operation and performance, and a design was created out of real components and then simulated.

We saw that the source level of the transducer can be controlled by varying the voltage into the charge pump, and this can be controlled by the microcontroller. The microcontroller gives the design a lot of versatility, because the amount of pulses, frequency of pulses, and source level can all be configured and adjusted in software. There are little limitations on the pulse frequency, only limited by the switching frequency of the pulse MOSFETs and their gate driver, and the power output should not vary by frequency, unlike when a transformer is used, so that the source level should remain very predictable at different frequencies. Thus, less tuning and calibrations need to be made for this design compared to a transformer design.

The simulations show the feasibility of the design, and circuits boards were designed to implement this; however, there was not time to build the design and test it on a transducer. The circuit boards could probably be redesigned as four-layer boards, and the high voltage board also could be made smaller because, during the time of the layout, the schematic had twice the amount of charge pump stages needed, and so extra space was taken up before these extra stages were removed. One last uncertainty is that the LDO used to regulate the voltage at the input of the charge pump will be current limited, specified at $1.5A$, and so the true time to charge the resevoir capacitors is unknown. Also, as the source level is reduced, the voltage drop across the LDO must get higher, and it will dissipate more heat. Power dissipation calculations for the LDO were only considered with an input voltage of 24V and an output voltage of 20V, and so they may need to be redone if lower source levels are desired with a 24V battery used as the input (if only a 12V battery is used, the LDO will work better for the lower source levels).

Chapter 5

Sampling and PC Interface

5.1 Introduction

A method to measure the hydrophone transducer's signal from the sent pulses and to transfer this data to a PC was needed so that the experimental data could be stored and analyzed later. Additional functions are needed to interface to the sampling hardware and to position the hydrophone with the BLDC motor controller card. The requirements are summarized:

PC Software Interface

- Communicate with the PC with a standard data link: RS-232, USB, Ethernet, etc.
- Relay the transmitting transducer height to the motor controller and the trigger signals to a transducer driver.
- Sample and transfer the hydrophone signal measurements to the PC software.

There were a couple options that were considered in how to implement the above tasks. The first option was to design a custom hardware measurement board to perform all of the required functions. This method was chosen in the previous masters project by Halvor Strøm. This task is quite big, and requires a lot of hardware design as well as programming for all of the functions. His board was not fully functioning, and could not communicate on ethernet, so it was only used as a starting point to design a new board. Figure 5.1 shows how this new board would fit in with the other system components. It would be the communication link to the PC, while it also could send commands to the motor controller, send trigger pulses to the hydrophone card, and sample the acoustic signal from the hydrophone.

Another option considered is to the interface the PC software directly to the motor controller and then to measure the hydrophone signal with a separate data aquisition tool such as a portable oscilliscope. The physics department owns a Tektronix TDS2004B

oscilliscope with a USB interface that can be accessed by PC software using MATLAB or LabVIEW with the use of a driver. There are also off-the-shelf USB portable oscilliscopes with sampling speeds up to $1\frac{Gs}{s}$. These scopes come with internal memory for storing the samples and customized software that allows saving of the measurements to a hard drive. The disadvantage of these options is that more manual work must be done to manage the different software tools on the PC. Also the TDS2004B oscilloscope uses an AC power source, so an inverter must be used in the field. The main other disadvantage of using the TDS200B oscilloscope is discussed further in section 5.5. Also, interface between the PC and the motor controller must still be made with RS-232 which has limited range and many new laptops do not have RS-232 ports, so a USB to RS-232 adapter may be needed.

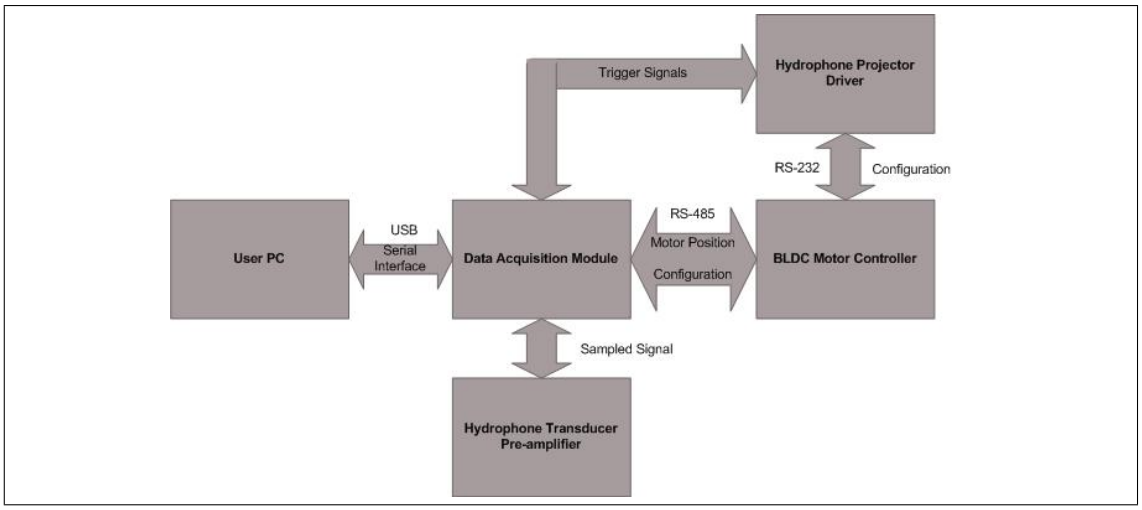


Figure 5.1: The block diagram of the system as it related to the data aquisition module shows the required functions of the module.

Both options were considered. First, the custom hardware was designed, but the task of building the board and programming all of the functions was too large for the scope of this thesis. So after the hardware was designed, the second option of using an oscilloscope was explored and found to be a much faster solution, although there are drawbacks. First I will discuss the custom hardware that was designed in section 5.2, as it provides a reference design and some of the considerations that can be made when designing a sampling system. Then I will discuss the oscilloscope method and how this was implemented.

5.2 Hardware Design

The hardware design is composed of the functions in figure 5.2. The hardware schematics are shown in appendix A.0.3. The ATXMEGA128 microcontroller was used again, like it was in the other hardware designs. An additional microcontroller, the ATMEGA32U2 is included as a SPI to USB bridge. The ATXMEGA128 is programmed through JTAG,

while the ATMEGA32U2 is factory programmed with a USB bootloader and can be programmed directly on USB using a program called Atmel Flip.

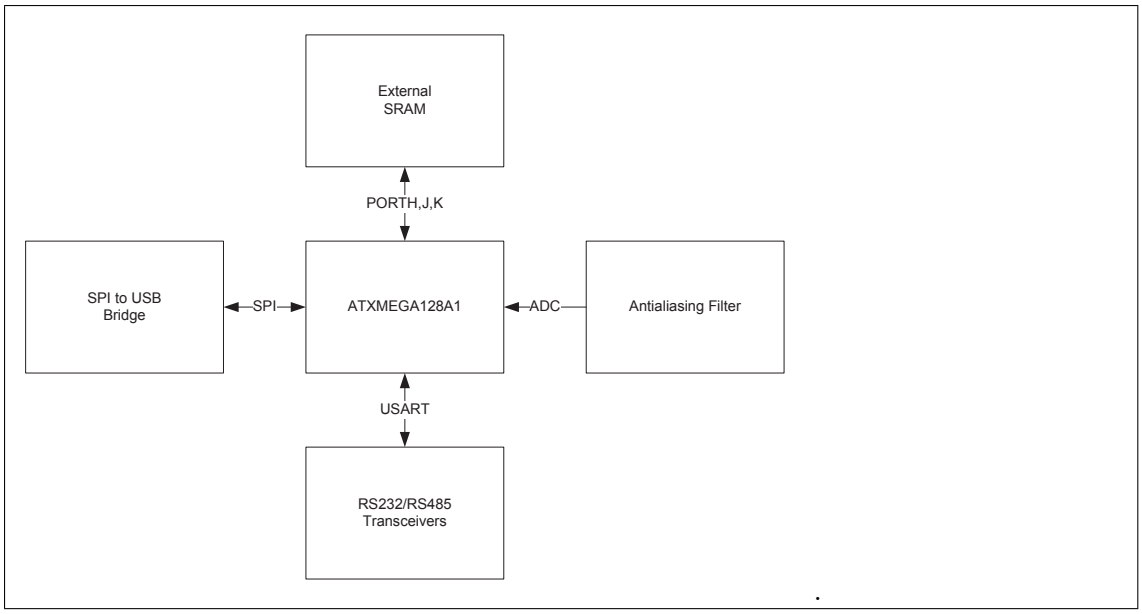


Figure 5.2: Block diagram of the sampling and interface hardware designed

5.2.1 USB

USB was chosen as the interface to the PC instead of ethernet because of its simpler hardware design as well as its growing standard as the serial interface for PC peripherals. A simple comparison of USB and ethernet are given in table 5.1 The ATXMEGA128 does not have a USB peripheral available, so it uses a SPI bridge to communicate to a second microcontroller, the ATMEGA32U2. The ATMEGA32U2 is one of the Atmel family 8-bit microcontrollers with an internal USB PHY, so it can communicate directly on USB. The reason this USB microcontroller could not be used alone was that it does not have the additional peripherals that the ATXMEGA218 has, and its ADC does not have a fast enough sampling rate.

The SPI bus that the two devices communicate on consists of two unidirectional data lines, an output and an input, as well as a clock signal and optional chip select pins if more than one device is on the SPI bus. Because the input and output data lines are on separate signals, the interface is able to communicate in full duplex mode. Each byte is sent serially, so it takes 8 clock cycles to send a byte. From the ATXMEGA128 datasheet, the chip can run with a clock frequency of 32MHz, and the SPI clock can operate at a maximum of half this frequency, so the SPI bus is capable of 16MHz. With 8 cycles per a byte, the SPI can transfer data at a maximum of 2MBytes/second, neglecting any overhead. The USB of the ATMEGA32U2 is USB 2.0 full-speed compliant, which means that it is capable of a data rate of 12Mbits/second, or about 1.5Mbytes/second.

Table 5.1: Basic Comparison of Ethernet and USB

Parameter	Ethernet	USB
Speed	100BASE-TX: 100Mb/s Gigabit: 1000Mb/s	USB 2.0: 12Mb/s High Speed: 480Mb/s
Typical Applications	Networking	PC Peripherals
Advantages	Simpler software interface, drivers not necessary, more robust.	Simpler hardware, powered over cable.
Disadvantages	More complex hardware	Requires drivers, requires VID/PID, lower cable range

5.2.2 ADC

The ATXMEGA128 samples the acoustic signal from the hydrophone with its internal ADC peripheral. This ADC is capabable of 2 MSPS, where each sample is 12 bits wide. Without any kind of compression, each sample requires two bytes, one for the lower order byte, and another for the 4 most significant bits of the sample. So at 2MSPS, and two bytes per sample, this gives a data rate of 4MBytes/second. With an ADC reference voltage of 3.3V, this gives a resolution of 806μV.

The USB is only capable of 1.5Mbytes per second, so there is no way for the the data to be transferred in real time without the ADC conversions overflowing. A solution to allow all of the samples to be transferred to the PC is to create a buffer of memory to write the samples into immediately before sending out the entire sampled data on the USB link.

5.2.3 Memory

The pulse from the hydrophone is only 0.1ms wide, but echos must also be sampled, and so a sampling window of 8ms [21] is used. This means that 32KBytes are required to store one pulse; however the ATXMEGA128 only has 8Kbytes of internal SRAM memory, so external memory is required. The CY62128EV30 was chosen as the SRAM chip to be used because it has 1024Kbytes of memory as well as a parallel data and address interface. Smaller memory sizes could be used, but the 1024Kbytes version is a standard size and so was the cheapest.

External memory can be accessed by the ATXMEGA128 on its external bus interface peripheral, called the EBI. There is 1Mbyte of address space available for each chip select. The timing diagrams of the write and read cycles are shown in figure 5.3. The EBI uses the microcontrollers port pins as the signals to drive the SRAM chip, and there are several configurations for the EBI to save the amount of pins needed. The EBI reduces

the needed pins by multiplexing its address lines. Address lines A[7:0] are multiplexed with address lines A[15:8], and the chip select signal CS0 is used as address line A16. The ALE signal latches the first address byte and then writes the second address byte directly to the memory chip before lowering the WE/RE signal to perform the write or read. 74HC573 latch chips are used to latch the address signals.

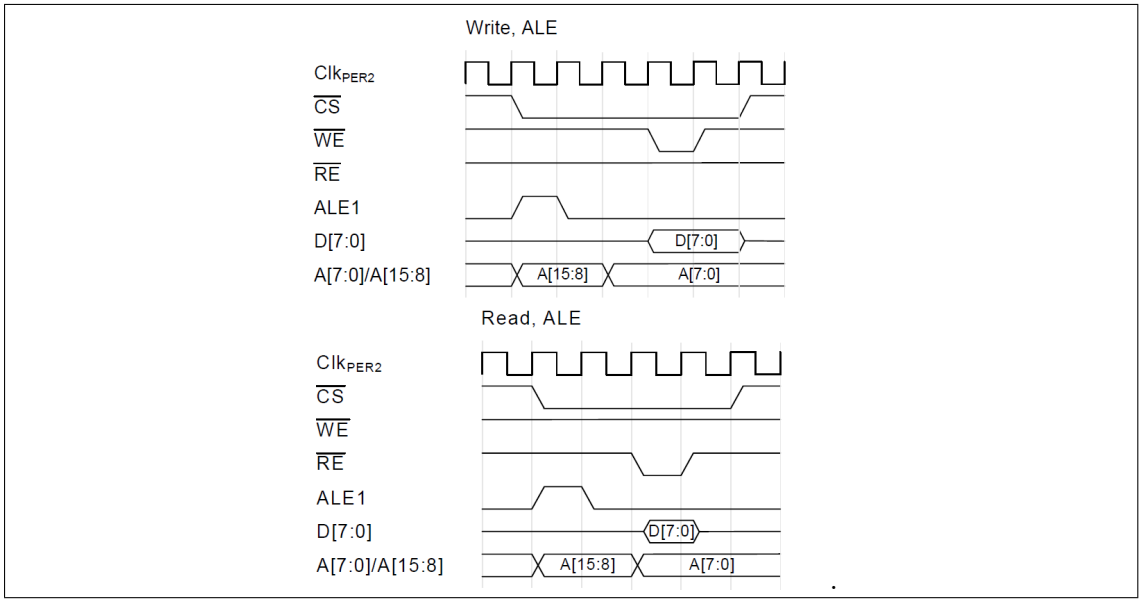


Figure 5.3: Timing diagrams for the EBI peripheral of the ATXMEGA128 to access external SRAM.

5.3 Filtering

5.3.1 Filter Design

With a method to sample the signal with the ADC, a way to store the samples in memory, and with USB to send the data to a PC, the other function needed for sampling the signal is an analog front-end. The front-end will receive the signal from a preamplifer, the Reson VP1000, that has an adjustable gain and high-pass filter adjustable cutoff frequency. From chapter 2, we saw that most of the acoustic signal's energy was centered around 120kHz and the nearest sides lobes. It was concluded that a 1MHz sampling rate should be sufficient to capture the information required, but the ATXMEGA128A can sample up to 2MHz, and so this was the sampling rate chosen. A 2MHz sampling rate of the ADC means that, by the Nyquist sampling, the maximum frequency that should have signal energy in it is 1MHz, or else the signal will experience aliasing. For this reason, a low pass filter was designed to eliminate the signal that lies out of this frequency limit. A 500kHz cut off was also considered, which would mean that oversampling could be taken advantage of, and also downsampling could be performed to fit the sample stream directly to the USB instead of storing it in a memory buffer; however, only the 1MHz

cut off without over/downsampling was designed.

The low pass filter must have a steep roll-off past the cutoff frequency to get as much of the signal spectrum without aliasing. If the signal was more bandlimited, to say 500kHz, then the oversampling at 2MHz would allow the antialiasing filter to have a less steep roll-off without worry of aliasing. While most of the acoustic signal is in fact in the 500kHz bandwidth, a 1MHz cutoff was chosen to give as much quality of the signal as possible. Regardless of the designed cutoff, the component values could be readusted for a 500kHz cutoff later anyway. Another important consideration was that low pass filters distort the phase information of the signal, and if the phase was distorted, then it could be difficult to find out if the echos came from surface reflections. If the group delay of the signal, which is the derivative of the phase response with respect to frequency, is constant over the entire frequency range, then there will be no distortion of the signal's phase.

The first filter considered was a bessel filter. Bessel filters have nearly linear phase response over their frequency range, and so they seemed like a good selection. However, a bessel's filter's roll off is not steep and its cutoff frequency is not directly controlled by pole placement, since the bessel polynomial must be maintained. Rather a scaling factor is used that is dependent on the order of the filter to determine the cutoff [17]. Because of the low roll off in the stopband, the bessel filter was not used.

Instead, the filter used is a butterworth filter. A butterworth filter has no ripple in its passband, unlike a chebychev, and it has less group delay than the chebychev also. A butterworth polynomial defines the poles of the filter, and it's poles are placed in the complex-plane along a circle who's magnitude is ω_0 which is given in equation 5.1, and the spacing between the poles is at angles of $\frac{\pi}{N}$ where N is the order of the filter. ω_p is the passband frequency of 1Mhz in this case, and ϵ determines the maximum variation in the passband [20]. ϵ is defined by equation 5.2 and a value of $\epsilon = 0.5088$ gives a maximum variation, A_{max} of 1dB in the passband at $N = 6$.

$$\omega_0 = \omega_p \left(\frac{1}{\epsilon}\right)^{\frac{1}{N}} \tag{5.1}$$

$$\epsilon = \sqrt{10^{\frac{A_{max}}{10}} - 1} \tag{5.2}$$

The poles were computed in MATLAB by finding the points on the complex-plane that had the approprate angle and magnitude, using equation 5.3. The poles are shown in figure 5.4 and the transfer function is given in equation 5.4. The complex-plane plot shows that there are 3 complex-conjugate pairs, where each pair reprsents a 2nd-order low-pass fiter with its own Q factor. The three 2nd order filters are plotted separately in figure 5.5 to show that one actually has a large gain at the cut off frequency, and

is reduced by the other two filters. The practical implication of this is that one of the opamps used to implement these filters must have a large enough unity-gain-bandwidth product specification to be able to have the the high Q factor gain at 1MHz, and also this filter stage should be cascaded at the end of the filter cascade so that only one amp has to have this gain active at this frequency. When the three stages are combined, they give the 6th-order filter behavior of figure 5.6, which is the theoretical magnitude bode response of the filter. It shows that the cutoff frequency is very close to the specified 1Mhz, and after the cutoff frequency, the magnitude begins to roll off at -120dB/decade.

$$P_N = \omega_0 \left(-cos\left(\frac{\pi}{N}\right) + jsin\left(\frac{\pi}{N}\right)\right) \tag{5.3}$$

$$T_f(s) = \frac{\omega_0^N}{(s-P1)(s-P2)(s-P3)(s-P4)(s-P5)(s-P6)} \tag{5.4}$$

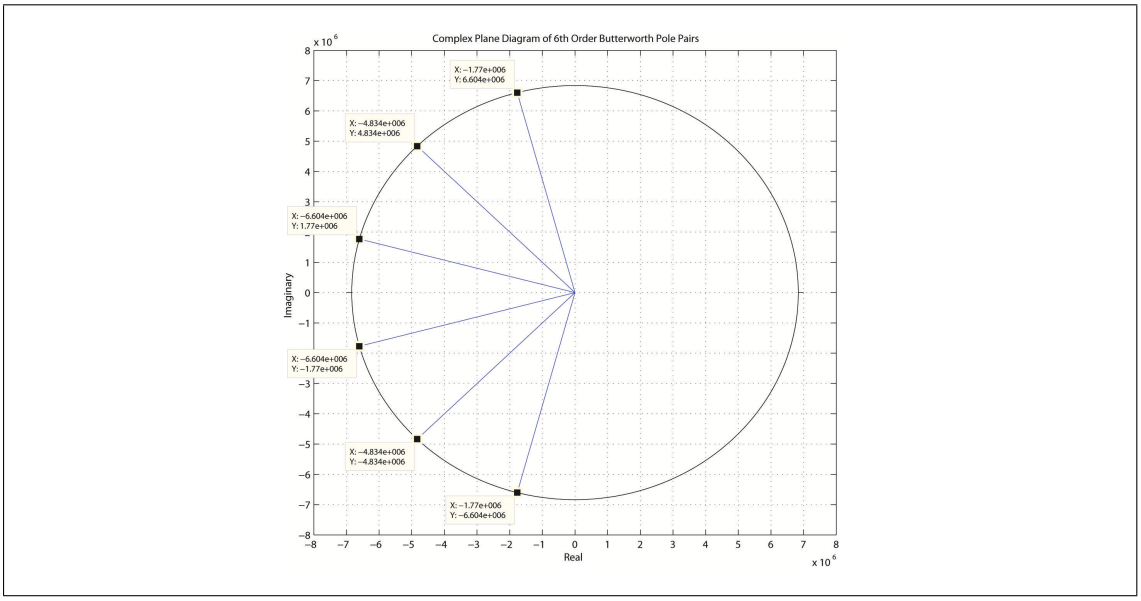


Figure 5.4: The poles of the 6th-order butterworth filter plotted in the complex plane.

With the poles known for the filter, the next step was to implement the filter with active elctronics components. A popular filter topology for 2nd-order active filters is known as the Sallen-Key topology of figure 5.7 taken from Kugelstadt [15]. The Sallen-Key filter has a transfer function given by equation 5.5. By setting this transfer function equal to the transfer function of equation 5.4, the component values could be calculated. A note in selecting the components is that as the cutoff frequency gets higher, the capacitors sizes tend to get smaller. To ensure that the capacitors stayed well above the parasitic capacitances of the opamp, the resistor values had to decrease to maintain the same cutoff frequency, and so there was a tradeoff during the component selection.

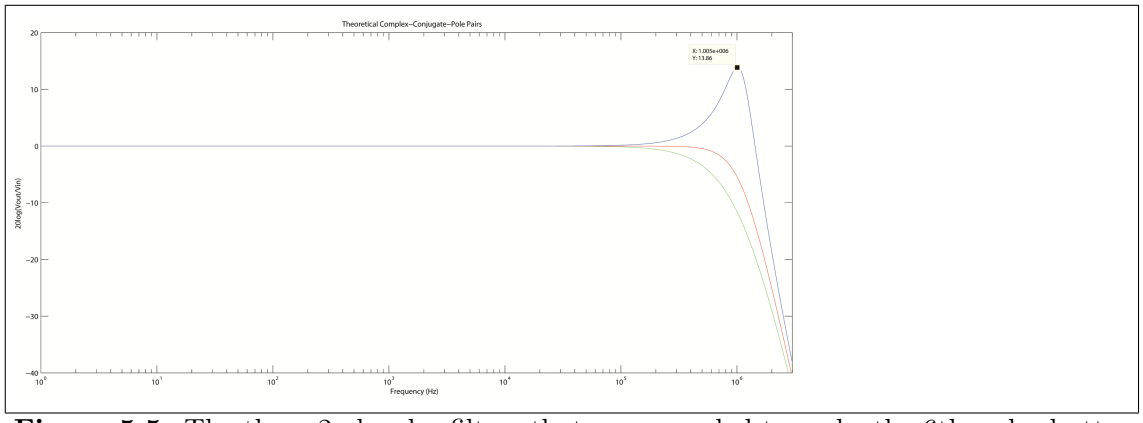


Figure 5.5: The three 2nd order filters that are cascaded to make the 6th order butter worth filter.

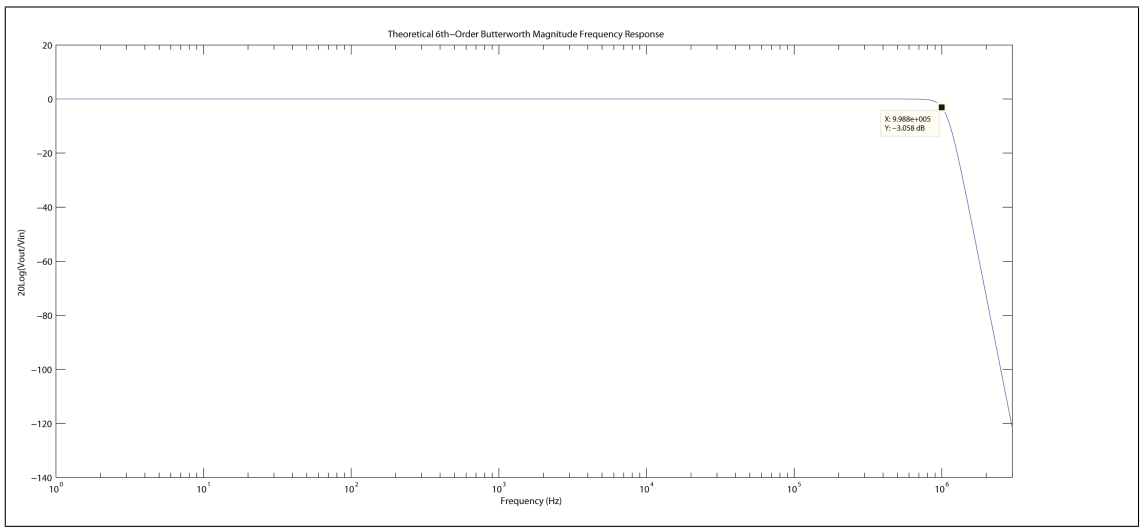


Figure 5.6: The theoretical bode magnitude response of a 6th-order filter as determined by the design equations.

$$T_f(s) = \frac{1}{s^2 R_1 R_2 C_1 C_2 + sC1(R_1 + R_2) + 1} \tag{5.5}$$

As was discussed before, the opamps have to be selected with enough bandwidth to filter the signal up to 1MHz. Kugelstadt [15] recommends that the peak gain of each filter stage should be 40dB less than the open-loop unity-gain-bandwidth of the selected opamp. The LMH6619 was selected as the opamp to be used because of its high frequency specifications as well as its rail-to-rail 3.3V or 5V single supply operation. The LMH6619 unity-gain-bandwidth is shown in figure 5.8. At 1MHz, it has a gain of about 37dB, and the highest gain factor of the three filters is 13dB, which only gives a gain margin of 24dB. This opamp is still used, and the simulations were made to verify that the circuit works as expected, even with less than ideal gain at the high frequency.

While the 6-th order filter gives a -120dB/decade rolloff to ensure that higher fre-

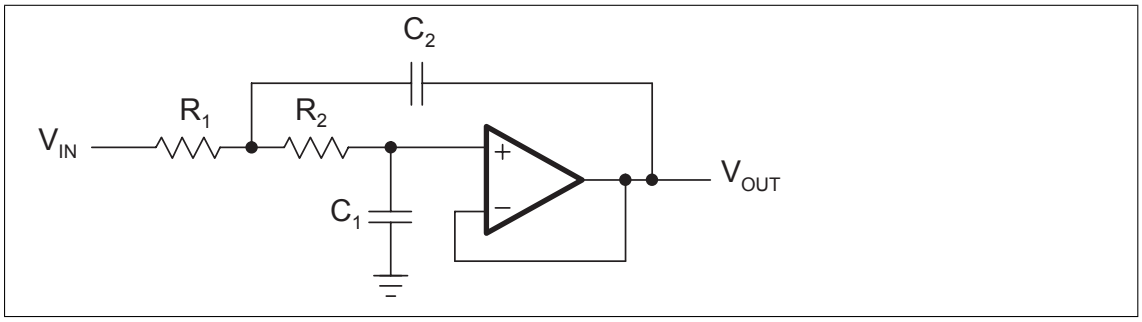


Figure 5.7: The sallen-key topology to implement a 2nd-order active filter

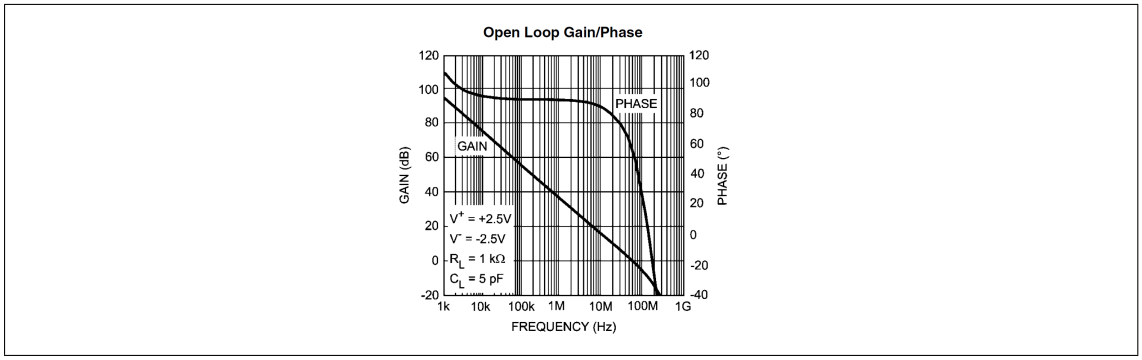


Figure 5.8: Open-loop gain-bandwidth curve of the LMH6619. The gain of the filters should be atleast 40dB lower than this gain at 1MHz.

quencies are not aliased into the sampled data, it also introduces a non-linear phase response that changes the group delay of the signal. To eliminate this group delay, a 2nd-order phase equalizer filter was cascaded after the butterworth filter to linearize the phase response and reduce the group delay spike of the filter near the cutoff frequency. An allpass filter has the transfer function of equation 5.6, where ω_0 is the cutoff frequency. The transfer function shows that it has unity gain over the entire spectrum. It only modifies the phase of the signal by adding a group delay.

$$T_f(s) = \frac{s^2 - \frac{\omega_0}{Q}s + \omega_0^2}{s^2 + \frac{\omega_0}{Q}s + \omega_0^2} \tag{5.6}$$

Different Q values were tested on the transfer function to find where the group delay would be flattened out most. The negative derivative of the phase of the transfer was computed in MATLAB using the -gradient() function to give the group delay of the butterworth filter after the phase equalizer filter had been applied. Another more exact method for choosing the appropriate Q factor is to differentiate the phase of the transfer function and then to modify the pole positions to adjust the group delay [26, p. 256-259]; however, this was not necessary as the graph in figure 5.9 shows quickly which Q values give the flattest group delay. A Q factor of 0.48 was chosen for the transfer function.

A circuit to implement a 2nd-order phase equalizer is shown in figure 5.10 from Deliyannis, Sun, and Fidler[6]. The transfer equation of this circuit is given in equa-

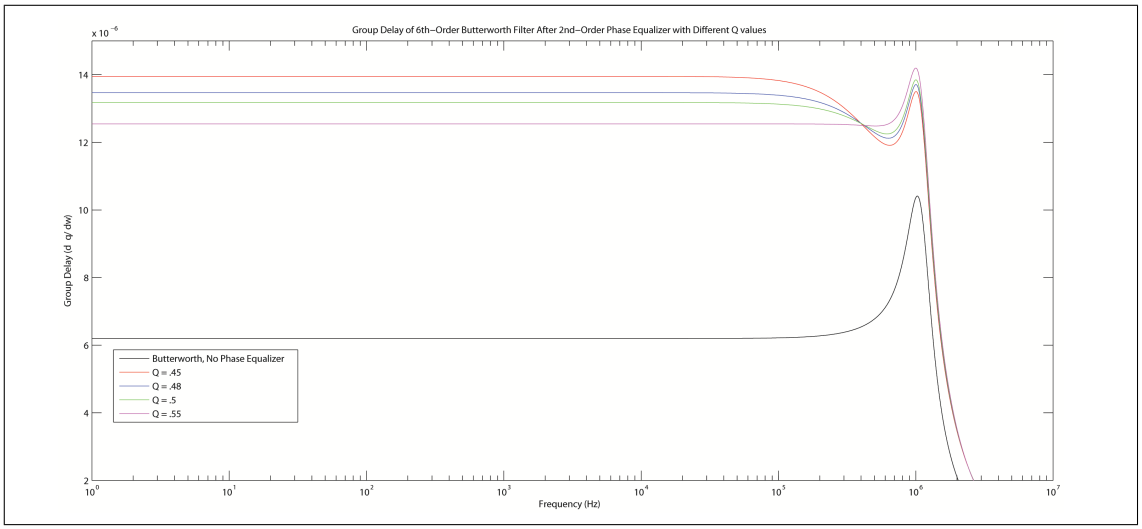


Figure 5.9: Group delays of the butter worth filter transfer function before and after a 2nd order phase equalizer. The Q factor is adjusted to find a minimum group delay.

tion 5.7, and it was set equal to the general equation with a Q factor of 0.48 to determine the component values. While the general transfer function has a gain of 1, this topology introduces a gain of $\frac{R_b}{R_b+R_a}$ that must be corrected for with a second gain stage.

$$T_f(s) = \frac{R_b}{R_a + R_b} \frac{s^2 - \left[\frac{R_a}{R_b}\frac{1}{R_1 C} - \frac{2}{R_2 C}\right] s + \frac{2}{R_1 R_2 C^2}}{s^2 + \frac{2}{R_2 C} s + \frac{1}{R_1 R_2 C^2}} \tag{5.7}$$

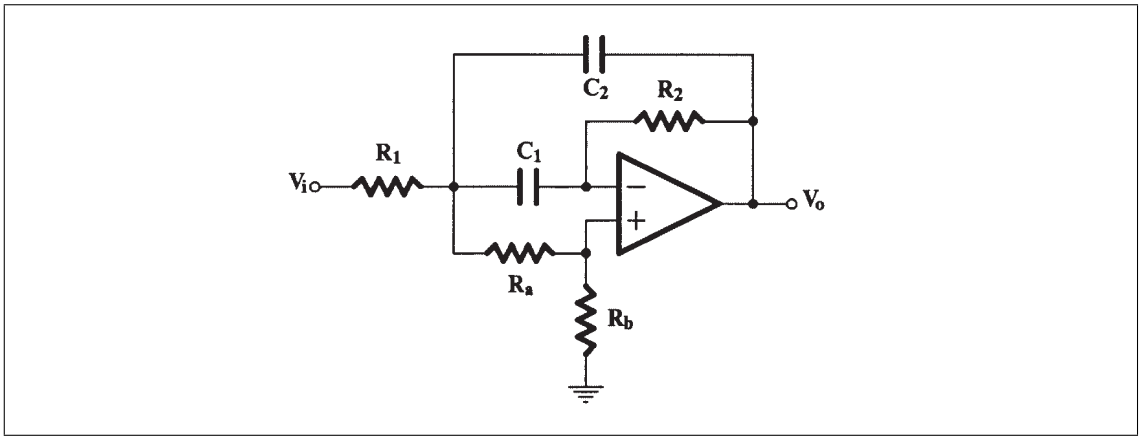


Figure 5.10: Filter topology for a 2nd-order phase equalizer circuit.

The design equations gave component values, but many of them were not readily available values, and so they were adjusted to more realistic values. These changes effect the transfer functions of the filters, but the simulations showed that the variations were acceptable. Also, the input to the butterworth filter has an AC coupling circuit as well as diode clamps to protect the input and bias resistors to keep the signal within the operating range of the opamp. The AC coupling capacitor acts as a high-pass filter with

the bias resistors so that low frequency components of the signal will not be measured.

5.3.2 Filter Simulation

PSPICE simulations were performed to verify that the filter design would perform as expected. The simulation circuit is shown in figure F.4. An AC analysis was performed to find the bode plots of the circuit. Figure 5.11 shows the frequency response of the filter. The cutoff frequency is measured to be 1.08MHz, so it is 80kHz off of the design. Also, the magnitude rises slightly at the cutoff, which indicates that the highest Q valued complex-pole pair filter of the butterworth has a higher gain than designed for, which is due to the component value selection. Also, the roll-off is less than the predicted -120dB/decade, but is still quite good at -110dB/decade.

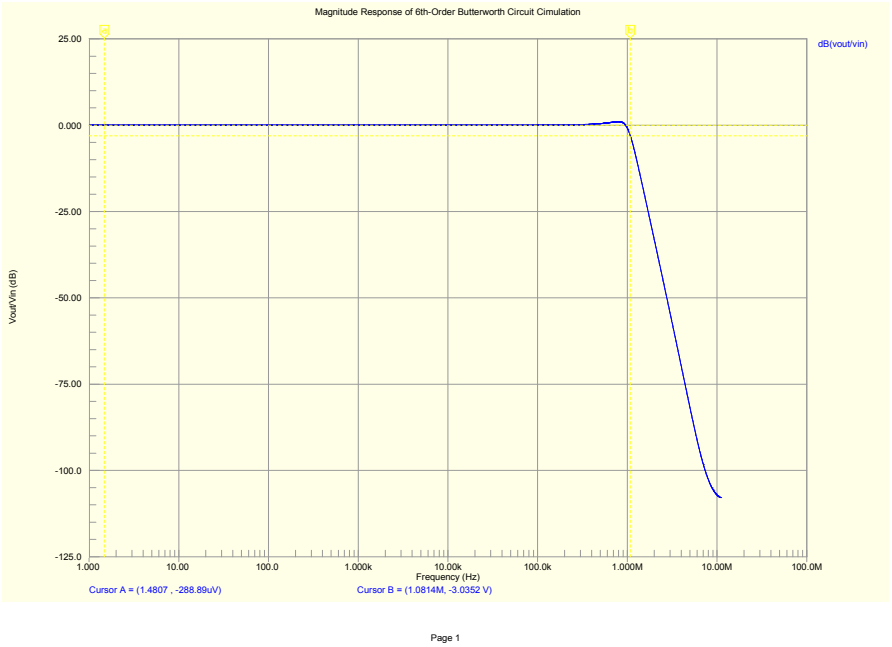


Figure 5.11: Bode plot of the 6th-order butterworth filter.

The phase equalizer was also simulated by looking at the group delay of the circuit at the butterworth output node and at the phase equalizer output node. A comparison is shown in figure 5.12, and it shows that the phase equalizer only has a group delay variation of $0.26\mu s$ while the butterworth alone has a group delay variation of $0.5\mu s$. The group delay is only a maximum of $0.5\mu s$ in the butterworth, and the acoustic signal's sine wave has a period of $8.6\mu s$, so the group delay is actually very small compared to

the time of the signal. The phase equalizer may actually not be necessary, especially since the frequencies around 120kHz have a constant group delay, however, it does give added improvement to the signal quality.

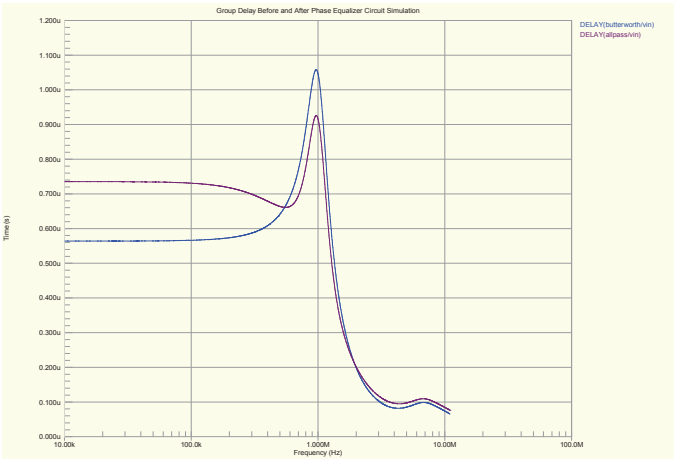


Figure 5.12: Group delay of the butterworth filter before and after phase equalization.

5.4 Hardware Design Implementation

The functions described above were implemented in schematics as well as some additional features such as transistors to drive a trigger signal to a hydrophone driver. The electronics are powered with LM317 LDO regulators that should be supplied a 12V power supply from a battery in the field. The design has an RS-232 and an RS-485 transceiver so that the ATXEMGA128's USART can communicate to a PC or to the BLDC motor controller serially also.

The hardware was designed as a two board system. and the schematics are shown in figures A.22 to A.26. The first board contains most of the necessary peripherals like memory, filtering, and I/O. This board layout is shown in figure A.27 and figure A.28. The second board is the SPI to USB bridge. The boards can connect with a SPI header that connects the SPI ports of the ATXEMGA128 to the ATMEGA32U2. The board layout for the SPI to USB bridge is shown in figure A.30 and figure A.31.

The same layout considerations were made as were for the BLDC motor controller and the hydrophone driver, but also the USB signals required special trace requirements. The USB signals need 90Ω impedance, and so the trace width and spacing were designed specifically for this impedance.

These circuit boards are ready to be built, but there was not time to build nor test this design, so the faster alternative of using an oscilloscope with PC software was considered.

5.5 Oscilloscope Measurement

Although a customized hardware solution would be useful, more portable, and easier to use in the experiment, the design time for such a device is very large. So, the other option was considered to take the TDS2004B oscilloscope as equipment in the experiment. The task with using the oscilloscope is storing all of its measurements on a PC. For one run of the experiment, the stand must make 750 measurements if the transmitting transducer is varied over a distance of 1.5 meters at 2mm increments, and this would be time consuming to do by hand.

The oscilloscope has a USB interface on it, and certain programs have drivers that allow it to interact with the oscilloscope, which include National's LabVIEW and Mathwork's MATLAB. The physics department has experience with LabVIEW, so LabVIEW was used to interface to the oscilloscope.

A library package with drivers for the TDS2004B was downloaded from National's website. LABView uses a high level API called a VISA that interacts with the instrument drivers and allows the LabVIEW VI programs to interact with the instrument. The drivers come with example programs, one of which is the edge-triggered aquisiton program. This program is almost already completely ready for aquiring signals from the hydrophone. The program was edited and merged with the motor controller application, as will be discussed in the next chapter. The program allows the user to save each measurement.

There was a problem found with this method because of the way that the oscilloscope is designed to work. Although the oscilloscope is specified to be capable of 1 GS/s, the actual sampling rate that the oscilloscope uses is dependent on the time divisions that the scope is set to. The oscilloscope has a record length, and this record length is constant at all time divisions. The record length is 2.5k, which means it takes 2.5k data points along the sample window determined by the time divisions, no matter how large or small the time divisions are. The scope must make 2.5k samples in 2.5ms to achieve a 1MHz samping rate. With 10 divisions making up the sampling window, the scope must be set at a time division of 250 μs to achieve 1MHz sampling. This 2.5ms sampling window is much less than the 8ms sampling window desired to give enough time for the oscilloscope to capture the transmitted signal and its echos. It also gives less margin for error in calculating the travel time of the signal, since the oscilloscope must be triggered just before the signal arrives at the transducer.

Additionally, as the time window becomes larger, the reconstruction of higher frequency waveforms is degraded. For example, the oscilloscope performs reconstruction of a pure $120kHz$ sine wave with a sample window of just $250\mu s$, which is equivalent to a sampling rate of 10MHz, as shown in figure 5.13. This figure was captured by the LabVIEW application. The figure shows the measurement expanded to show that reconstruction is not complete. This means that a reconstruction algorithm should be

performed on the data before it is analyzed since the oscilloscope fails to perform a perfect reconstruction. When using the oscilloscope instead of a custom hardware design,

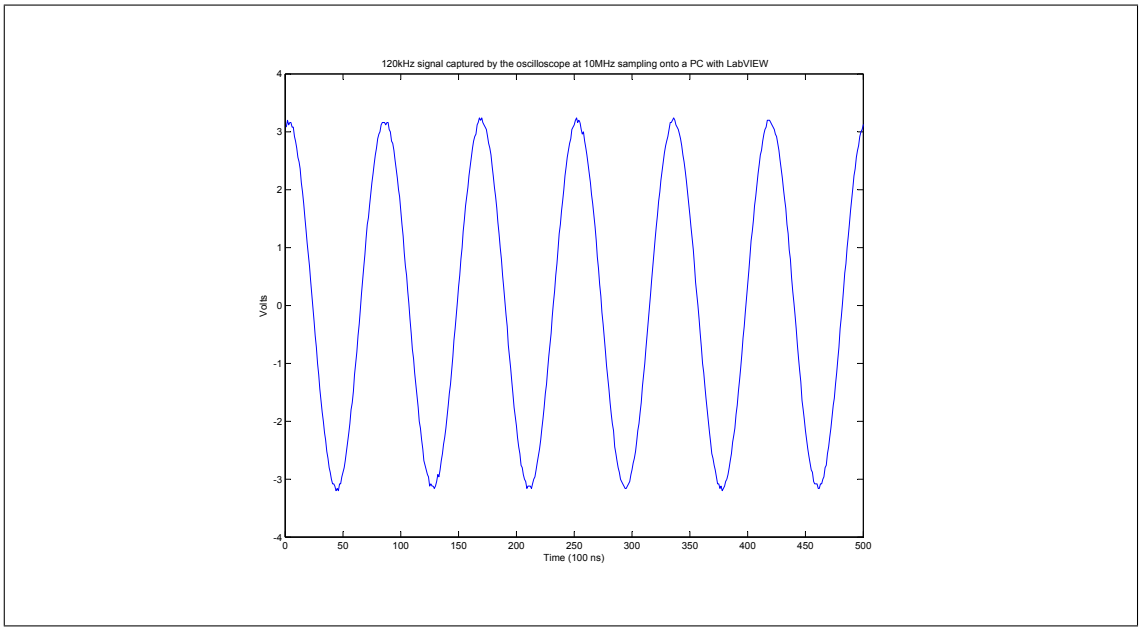


Figure 5.13: The oscilloscope reconstructing a 120kHz sine wave sampled at 10MHz.

trigger signals must be sent to the oscilloscope with a signal travel time delay after the transmitting transducer has been triggered to send an acoustic pulse. This function can be performed by the motor controller card, and so when substituting the custom measurement board described in section 5.2 with the oscilloscope, the new equipment setup becomes that of figure 5.14.

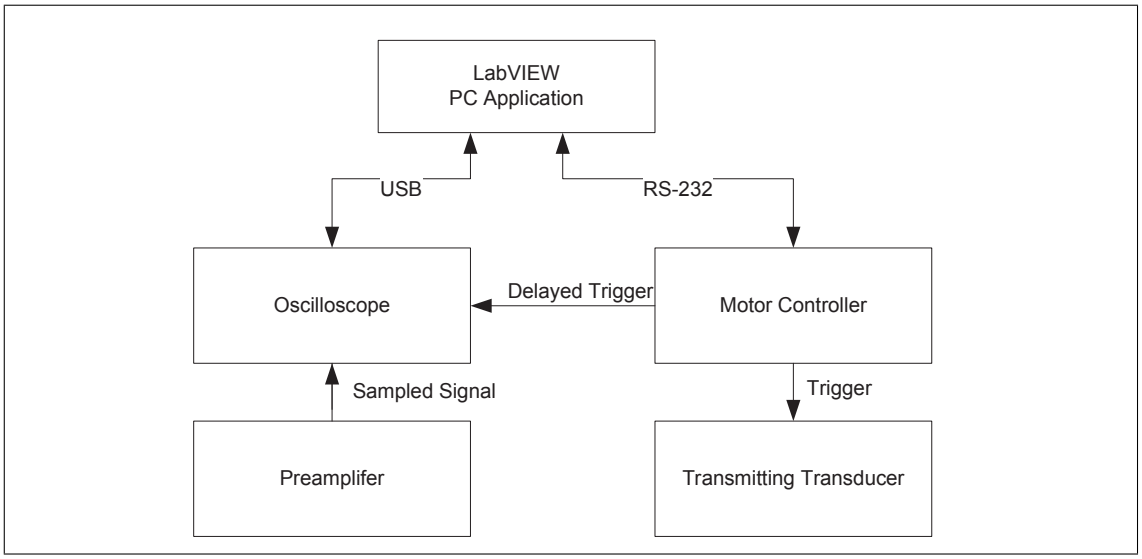


Figure 5.14: The final experiment setup realization.

5.6 Conclusion

Two methods for a system to transfer the measured acoustic signal to the PC were explored. The first option to design custom hardware was considered. A digital sampling system was designed, and analog filters were carefully designed to give the best signal quality without distorting the measured waveform. A hardware design with schematics and circuit board layouts was created and is ready to be built, but it was not built or tested yet; however, there was a simpler solution to use an oscilloscope that interfaces to LabVIEW on the PC. To make this solution work, the motor controller card is modified to provide the trigger signals necessary for timing the sampling of an acoustic pulse. The disadvantage of using the oscilloscope is that it can only take 2500 samples on any time scale, and so to sample at 1MHz, the oscilloscope can only have a maximum sampling window of 2.5ms, which may not be long enough to capture all of the echos and may also make it harder to time the triggers.

Chapter 6

PC Software Applications

From chapter 3, the design of a BLDC motor controller was discussed. The controller was built with an RS-232 interface to communicate to a PC to receive commands and to send position feedback information. For the PC to communicate with this board, a user application had to be created. Similarily, from chapter 5, a sampling system was layed out using an oscilloscope that must interface to the PC on LabVIEW using USB. To make the integration of the motor controller and oscilloscope as seamless as possible, both applications were created in LabVIEW.

6.1 LabVIEW basics

LabVIEW is a comprehensive instrumentation and data aquisition development platform. A distinction for programming LabVIEW compared to other languages is that all programming is done graphically using function blocks. LabVIEW uses a programing language called G for implementing this graphical approach. Wires connecting the blocks represent variables and data streams. Program flow is still controlled in the same way as a written language with loops and conditionals; however, the functions and variables behaving according to these flow structures by being placed inside them. A completed LabVIEW program is called a VI file, and sub VIs can be added into the hierarchy to give the program more modularity.

A big advantage for using LabVIEW is that many instrumentation equipment drivers are created for it. As was discussed in the previous chapter, LabVIEW uses a VISA API to interact with these drivers, and so to begin commmunicating with hardware, a VISA to the device must be opened. Instrument VISAs can be registered using the Measurement and Automation Explorer program that comes with LabVIEW.

6.2 The LabVIEW programs

6.2.1 Motor Controller Interface Application

The first program interfaces with the motor controller on RS-232. The program is shown in figure 6.1. First, a COM port VISA is initialized, and it is sent into a while loop. In the while loop, there are two sequences that happen sequentially. The first sequence is the event case. The event case is triggered from PC events like mouse-ups, clicks, and other GUI events. Each case in this event box is triggered by buttons on the GUI, and these events tell the application to send messages out on RS-232 to the motor controller. The application sends these messages by putting a string input into the VISA's write function block available in a functions menu. Some string functions are used in some of the event cases to format the data for how the motor controller expects it. Most commands are simply an ASCII character, except for setting the position. To set the position, the program must transmit a "%" followed by 5 ASCII digits that represents the command number of revolutions the motor should take. When the user sets the position, they give the position in mm units. The LabVIEW program automatically converts this into revolutions inside the while loop.

If there is no event within a timeout period, a timeout event case occurs that moves the program onto the next section, where a read function is performed on the VISA. This function takes all of the bytes in the COM port buffer. More string functions are used to sort through the buffer data and find the latest string that matches the format "% _ _ _ _ _", where each "_" represents a digit that the motor controller sent of its last known revolution count. The LabVIEW program converts this value into a string and displays it on the GUI.

6.2.2 Oscilloscope Interface Application

To interface to the oscilloscope, a library was downloaded that contained the subVI modules that give access to the oscilloscope's parameters. These subVIs are given the desired values, such as setting the scope to be edge triggered from an external source and setting the scope's time divisions. When a trigger on the scope happens, the scope captures the sample and sends it to the LabVIEW program, which is sent out in a waveform stream. This waveform is plugged into a graphing function block that plots the data on the GUI. A second sequence is added after this that gives the user the option to save the waveform or to discard it and make a new measurement. This sequence is controlled by an event case block, like in the motor controller program. When a user presses save, the event routes the waveform stream into a save measurement file function block. The block is configured to save to the same file every time, and if there is already a file with this name, it will append the new waveform data at the end of the file. The data is saved in a tab delimited format. Once the make new measurement button is pressed in the GUI, the event case will reset the loop and now the oscilloscope is brought back into aquisition mode,

waiting for a new trigger signal. The oscilloscope interface program is shown in figure 6.2

Both programs work individually, and so then a merge tool was used in LabVIEW to combine the programs into one GUI. The GUI is shown in figure 6.3. The 120kHz waveform was captured from a function generator, and was captured using the trigger sent from the motor controller. The motor controller trigger outputs were verified on the oscilloscope, where the output to the transmitting transducer goes rises and lasts for the specified duration in ms, at which point this trigger falls and the trigger to the oscilloscope rises. The motorcontroller is programmed with defaults of 1-10ms delays in 1ms increments.

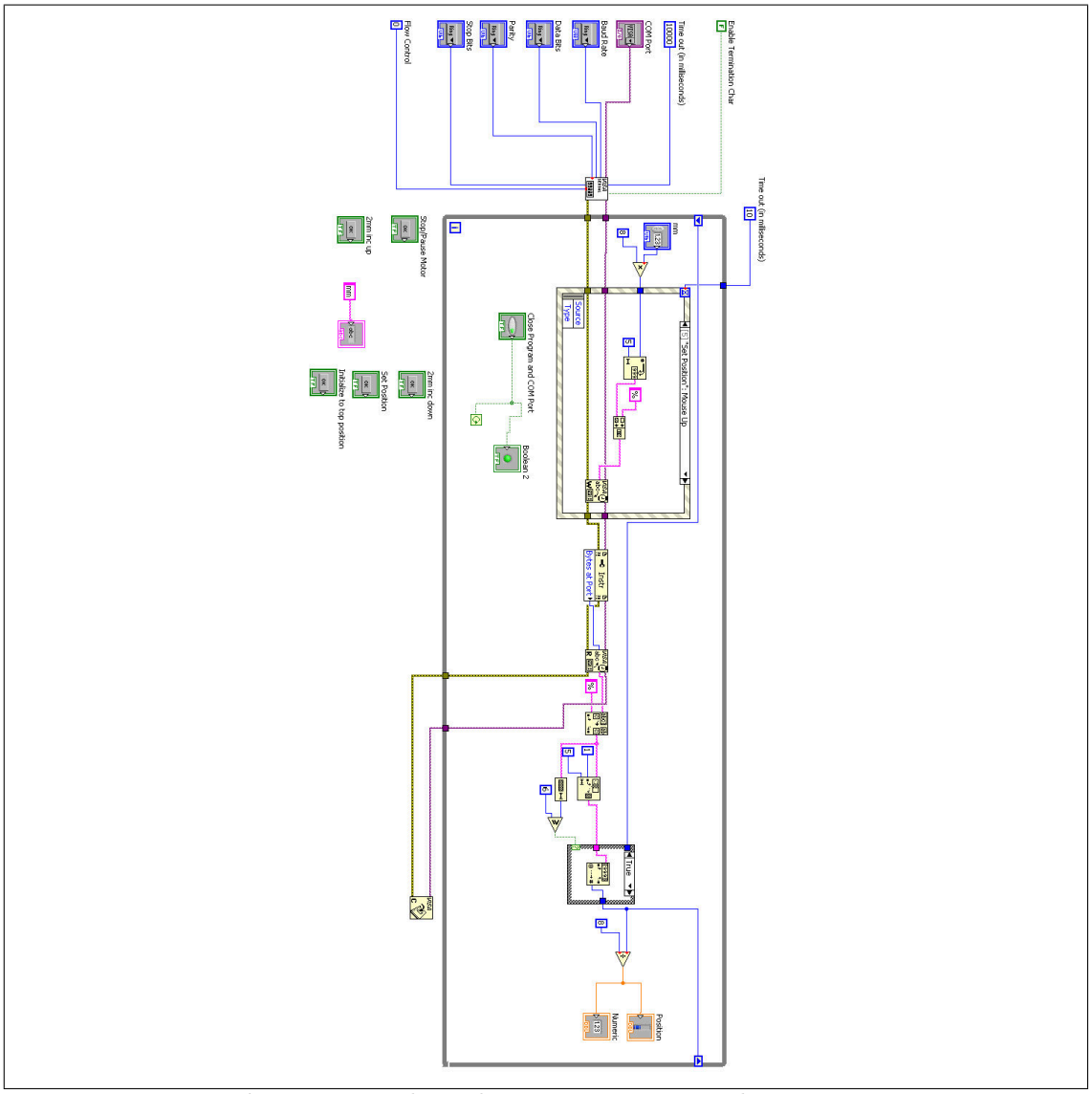


Figure 6.1: The LabVIEW motor control program.

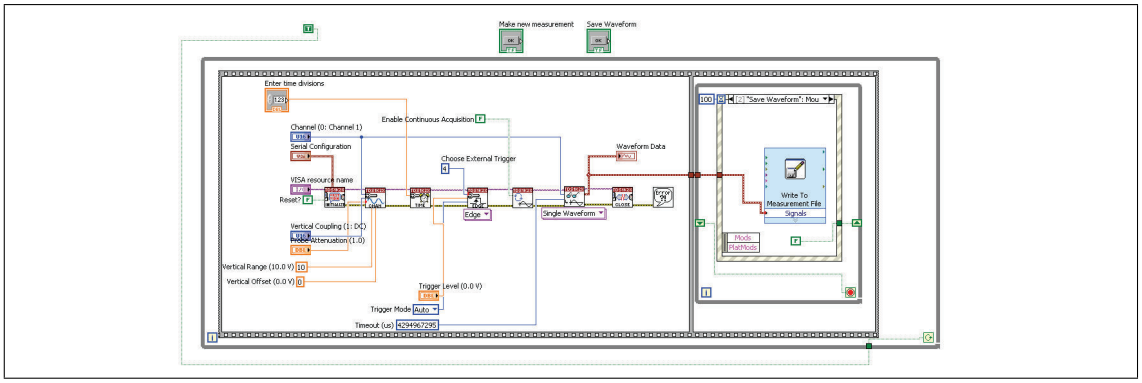


Figure 6.2: LabVIEW program for the oscilloscope aquisition.

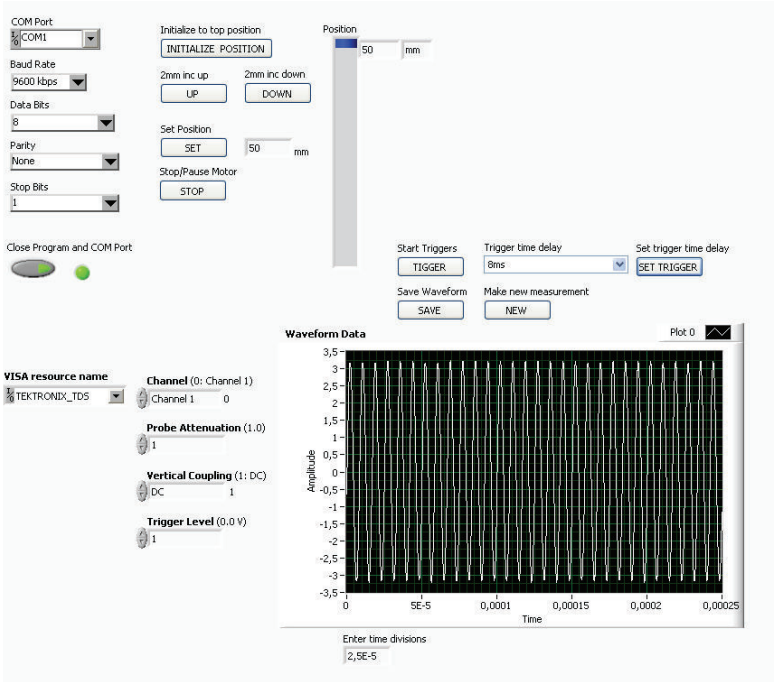


Figure 6.3: The LabVIEW GUI created to communicate to the motor and oscilloscope.

Chapter 7

Conclusions

7.1 Project Summary

This project concludes with a working system to make measurements in the sound-field. The system consists of the motor controller which positions the platform within 1mm accuracy, and an oscilloscope with LabVIEW software that enables storage of waveforms onto a file on the PC. Because the transducer driver was not built, the system will have to use the EY500 still.

In Halvor Strøm's thesis conclusion, one of his ideas for future work was for someone to come up with a new drive for the positioning system. This thesis accomplished this goal. The motor controller section of the project was the most time consuming and practical part, where most of the accomplishments took place. This design went from learning BLDC motor theory, choosing an appropriate sized motor, designing a motor controller from scratch(with many reference designs in application notes) to drive this motor, and then to developing software to run the motor and eventually to control the rotor position which allowed accurate positioning of the hydrophone platform. In addition, a PC application successfully interfaces to the motor controller and gives the user complete control of the hydrophone positioning. Performance of the system has improved as well, where the motor can move the platform much faster now. Also, the reed sensor addition guarantees calibrated positioning and gives the data a true reference position. Furthermore, the control was designed so that position is specified with an absolute position rather than relative positions.

Some of the failures of this section were that the board needed a lot of modifications on it, and some functions are not useable because of the modifications, like RS-485; however, they are not needed in the final setup created. If the motor is needed to operate at above 12V or if a new system setup is created, the current motor controller will not work, and either needs to be repaired or a new version needs to be made. Also, there was not

time to design a proper PID controller, and this system will have some challenges in implementing such a controller because of how easily the motor command saturates with the high friction of the gears and with how a large command position range can easily cause windup issues when using an integrator in the control loop. Another shortcoming is that the tests to verify the accuracy of the positioning were not very accurate themselves, where markings were drawn with a ruler and by hand. My advisor has a ranging laser tool, and this would have been a better tool to measure the positions more accurately. However, the positioning was always within 1mm of the the commanded position, and the system was specified to have a resolution of 2mm, so it should still be very useful as is.

The acoustic chapter involved learning about comb filters, echos, and describing the acoustic signal in the frequency domain. This helps give a good perspective on how the experiment can actually be used to describe the sound-field. It also gives guidance on selecting an appropriate sampling frequency. I offer some examples of using a frequency domain technique to extract the echo information from the received signal. I also provide relationships concerning sampling bandwidth to show the limitations the system will have for measuring echos.

The transducer driver is a promising method for creating the electrical signal to be transmitted. The simulations and equations all show the capabilities of the design, and also guide how to design such a system. A major shortcoming is that the design could not be built or tested in time, and so this is a design project that could never be validated.

Likewise, the sampling system designed in chapter 5 would be ideal for this sound-field experiment. The hardware provides the necessary ADC, memory, and datalink to a PC. Filtering techniques were explored, and simulations of the butterworth and phase equalizer validate the design decisions. The filters alone would be useful for any kind of sampling system trying to make the sound-field measurements. Again, there was not time to build this equipment either, but the oscilloscope system was created as a secondary solution so that sound-field measurements would be possible without the need to build and test the schematics that I designed. The only concern with using the oscilloscope is its fixed 2.5k data points in a sampling window.

7.2 Future Work

As this project came to me rather open-ended, it still remains open-ended for future work. The hydrophone positioning system can be considered complete now. There should be no need to spend any more time with this portion of the system, and so focus can be put on sampling the acoustic signal and interfacing this to a PC.

The oscilloscope method developed in this thesis should be tried first before more

effort is put into designing a sampling system. If a sampling system or a transducer driver are still needed from masters students after evaluating the oscilloscope method, these designs should be split into separate projects for one person to focus on each task from start to completion, unlike the error I made of trying to design too many things. I would highly recommend that this project is worked on in a group if possible in the future, because there are many subsystems used in this experiment. If the transducer driver is needed, the design can be built from the schematics and gerber files that I provided. It then has to be programmed, and tested. The sampling hardware I designed can be also built with gerber files that I provided, or the schematics can be used as a reference design along with the design work Halvor Strøm did before myself.

When sound-field measurements are actually aquired, an evaluation of the two frequency domain methods I described could be used to see if they are successful for analyzing the data from this experiment, and they can be compared to the peak-detection time domain methods. This assessment of analysis techniques would be very interesting.

Bibliography

[1] Simrad Subsea A/S. Instruction manual: Simrad ey500 portable scientific echo sounder, January 1998. [cited at p. 60]

[2] Laszlo Balogh. Design and application guide for high speed mosfet gate drive circuits. *Texas Instruments [Application Note]*. [cited at p. 83]

[3] Carl Blake and Chris Bull. Igbt or mosfet: Choose wisely. *International Rectifier [Application Note]*. [cited at p. 83]

[4] M.J. R. Healy B.P. Bogert and J.W. Tukey. The quefrency alanysis of time series for echoes: Cepstrum, psuedo-autocovariance,cross-cepstrum and saphe cracking. *Proceedings of the Symposium on Time Series Analysis, M. Rosenblat, Ed., Wiley, NY*, pages 209–243, 1963. [cited at p. 24]

[5] Airmar Technology Corporation. Understanding transducer specifications. May 2010. [cited at p. 60]

[6] T. Deliyannis, Yichuang Sun, and J.K. Fidler. *Continuous-Time Active Filter Design (second edition)*. CRC Press, 1999. [cited at p. 102]

[7] Sanjeev Dhull, Sandeep Arya, and O.P Sahu. Comparison of time-delay estimation techniques in acoustic environment. *International Journal of Computer Applications*, pages 29–31, October 2010. [cited at p. 10]

[8] Alan Elbanhawy. A simple guide to selecting power mosfets. *EDN*, pages 87–90, November 2001. [cited at p. 76]

[9] Charlie Elliott and Steve Bowling. An901: Using the dspic30f for sensorless bldc contro. [Microchip Website]. [cited at p. 36]

[10] Guido Gessl. Cepstrum analysis, January 2004. [cited at p. 24, 25]

[11] Dusan Graovac, Marco Purschel, and Andreas Kiep. Mosfet power losses calculation using the data-sheet parameters. *Infineon [Application Note]*, July 2006. [cited at p. 48]

[12] Daniel W. Hart. *Power Electronics*. McGraw-Hill, 2011. [cited at p. 167]

[13] Stephanie Johnson. Design for a discrete charge pump. *Texas Instruments [Application Note]*. [cited at p. 73]

[14] T. Kenjo. *Permanent Magnet and Brushless Dc Motors*. Oxford, 1985. [cited at p. 32, 176]

[15] Thomas Kugelstadt. *Op Amps for Everyone.* Texas Instruments. [cited at p. 99, 100]

[16] P.G. McLaren. *Elementary Electric Power and Machines.* Ellis Horwood Ltd, 1984. [cited at p. 65]

[17] Ray Miller. A bessel filter crossover, and its relation to others. [cited at p. 98]

[18] T. Myono, A. Uemoto, S. Kawai, E. Nishibe, S. Kikuchi, T. Iijim, and H. Kobayashi. High-efficiency charge-pump circuits with large current output for mobile equipment applications. *IEICE Transactions on Electronics*, pages 1602–1611, October 2001. [cited at p. 67]

[19] Francis Rodes. Build a transformerless 12v-to-180v dc/dc converter. *EDN*, pages 83–86, July 2004. [cited at p. 72]

[20] Adel S. Sedra and Kenneth C. Smith. *Microelectronic Circuits (fifth edition).* Oxford University Press, 2004. [cited at p. 98]

[21] Halvor Strøm. Utvikling av system for kartlegging av lydfelt p grunt vann. Master's thesis, Universitetet i Oslo, 2007. [cited at p. 29, 97]

[22] Devologic Subsea Systems. Ham.base compact hydro acoustic modem. [Product Brochure]. [cited at p. 68]

[23] Ed Tucholski. Active sonar equation and projector source level. [http://usna.edu/Users/physics/ejtuchol/Chapter16.pdf]. [cited at p. 89]

[24] Robert J. Urick. *Principles of Underwater Sound(3rd Edition).* McGraw-Hill Book Company, 1983. [cited at p. 9, 10]

[25] Vladimir Vitchev. Calculating essential charge-pump parameters. *Power Electronics Technology*, pages 30–42, July 2006. [cited at p. 67]

[26] Steve Winder. *Analog and Digital Filter Design (second edition).* Newnes, 2002. [cited at p. 101]

[27] Rodger E. Ziemer, William H. Tranter, and D. Ronald Fannin. *Signals and Systems: Continuous and Discrete (4th Edition).* Prentice Hall, 1998. [cited at p. 16, 22]

Appendices

Appendix A

Hardware Design Documents

A.0.1 BLDC Motor Controller

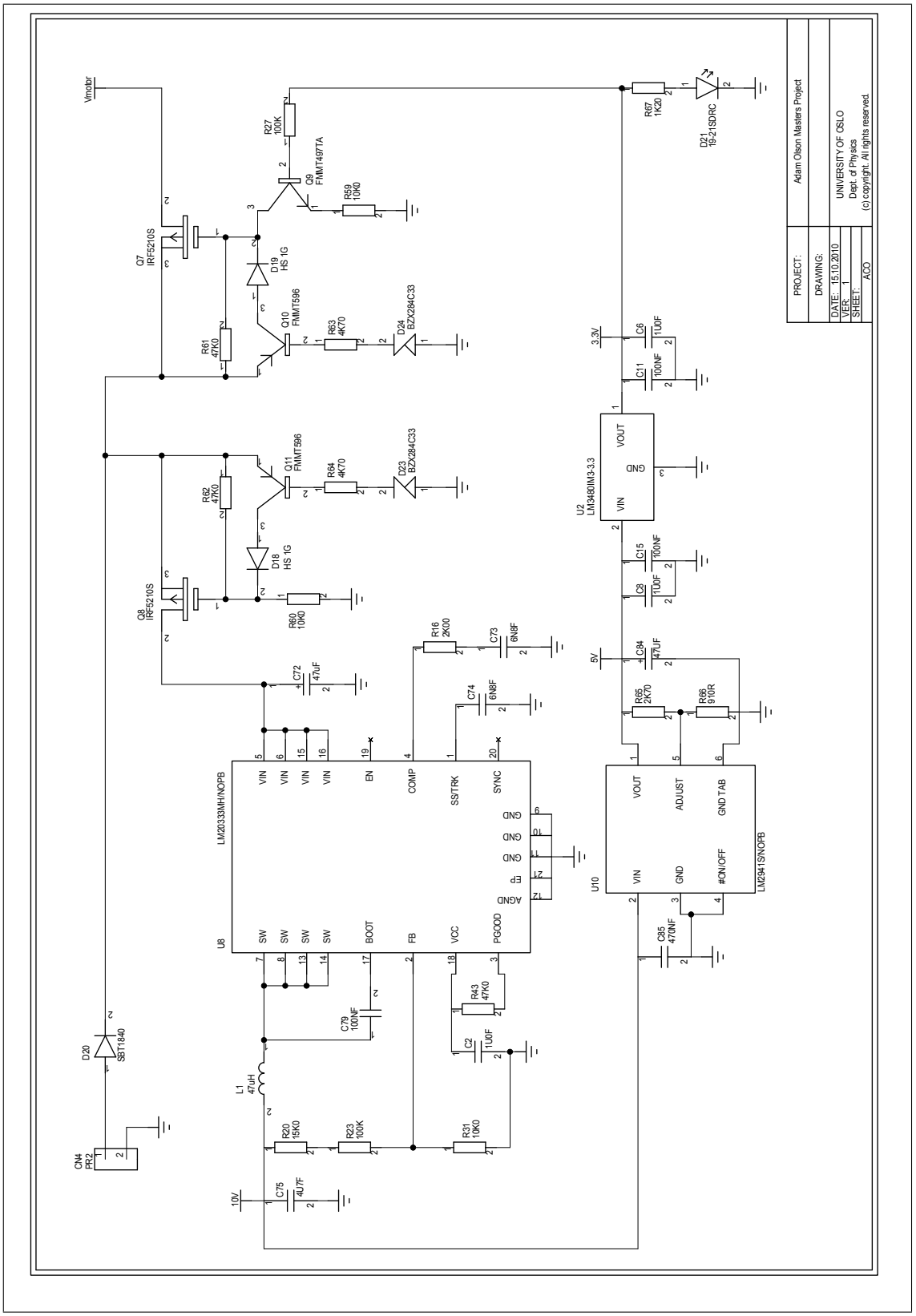


Figure A.1: BLDC motor controller, sheet 1, power supplies.

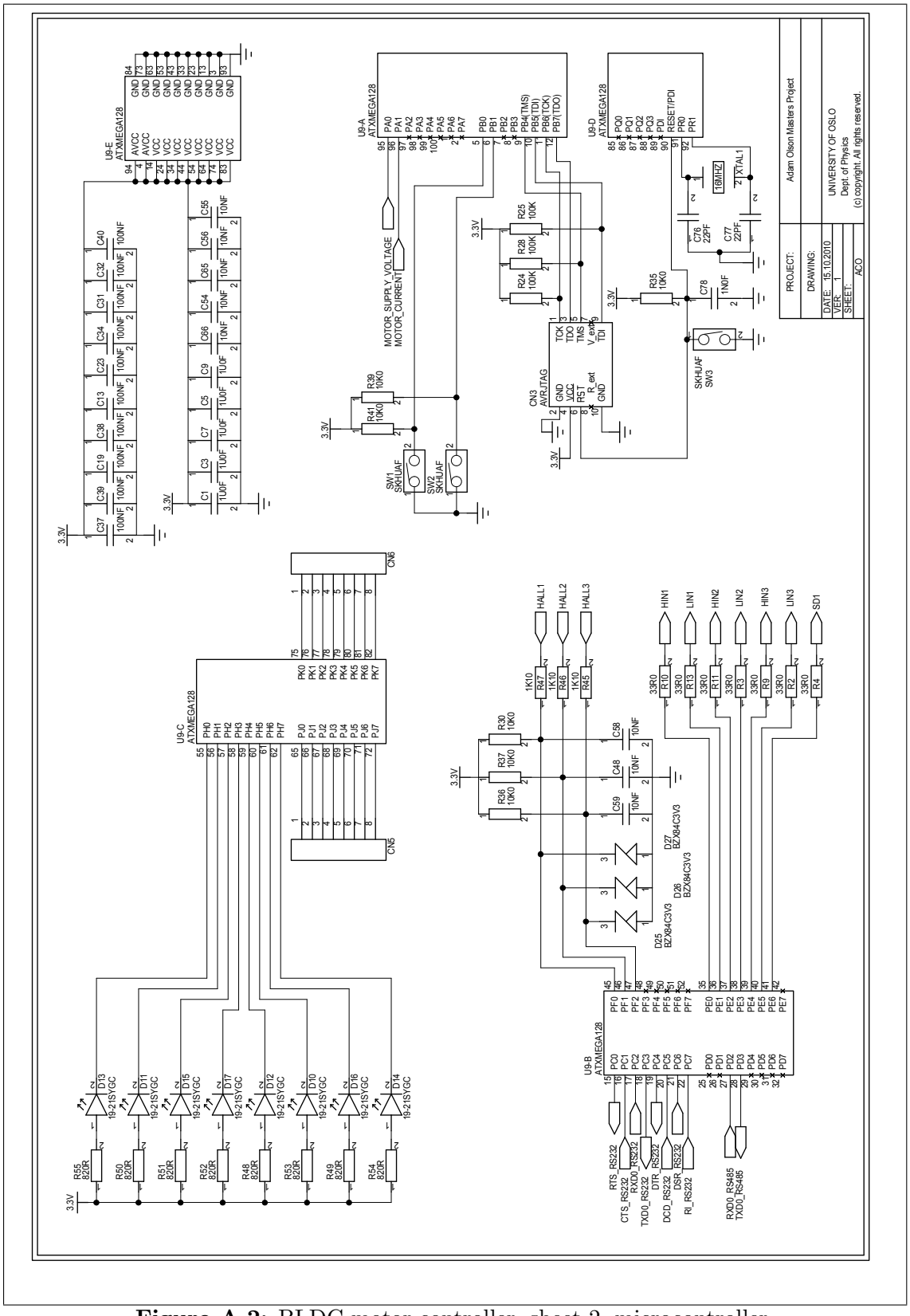


Figure A.2: BLDC motor controller, sheet 2, microcontroller.

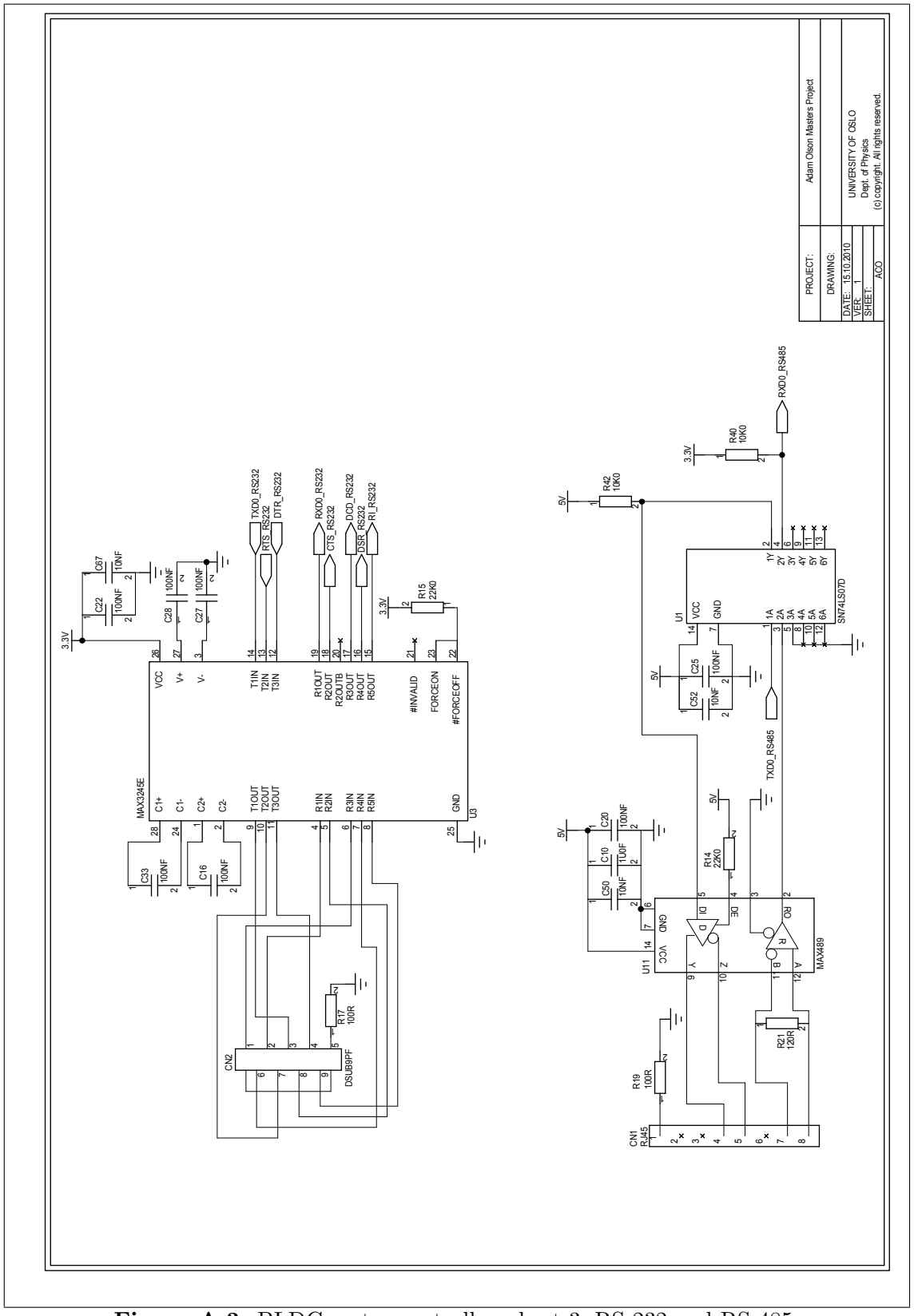


Figure A.3: BLDC motor controller, sheet 3, RS-232 and RS-485.

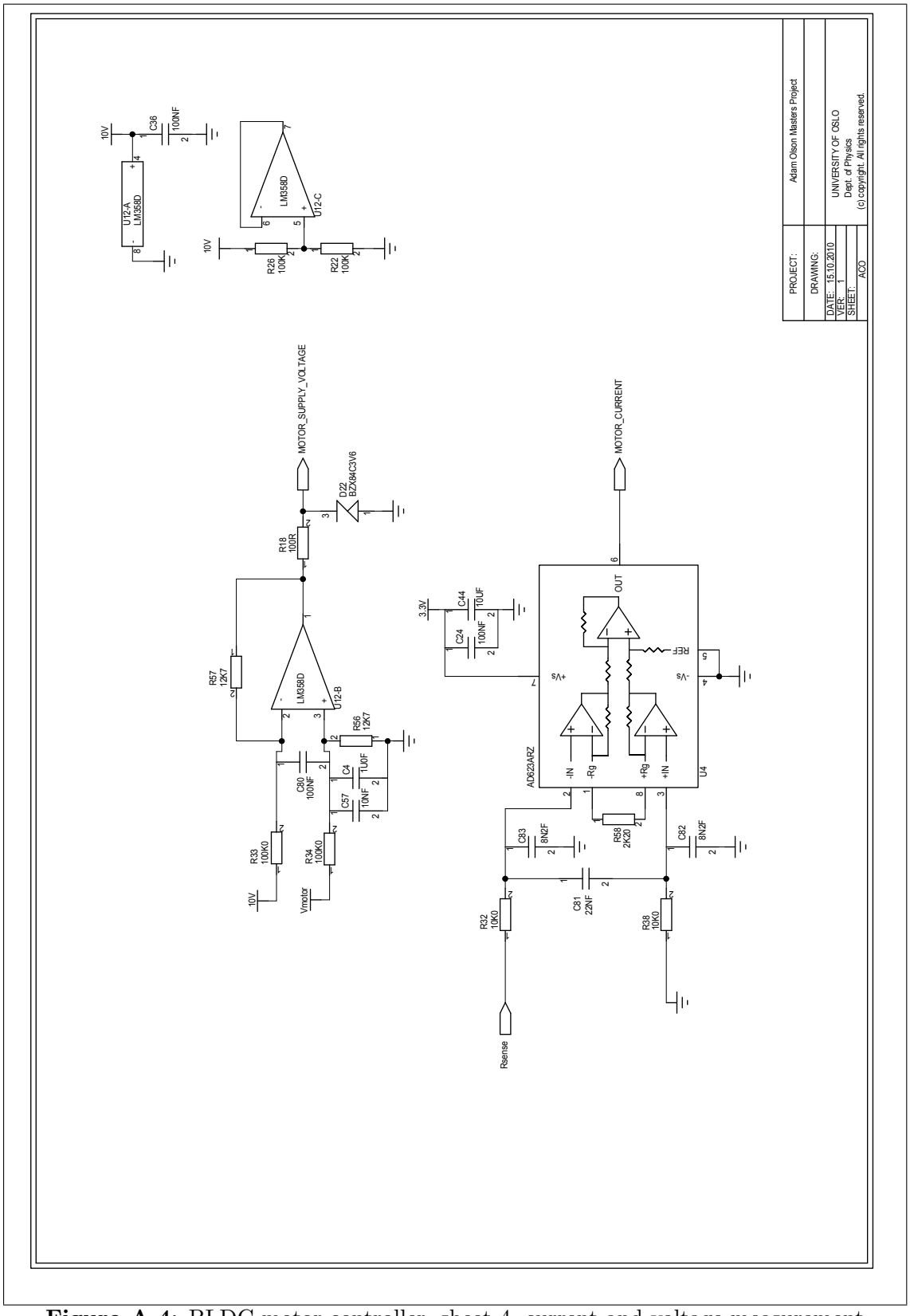


Figure A.4: BLDC motor controller, sheet 4, current and voltage measurement

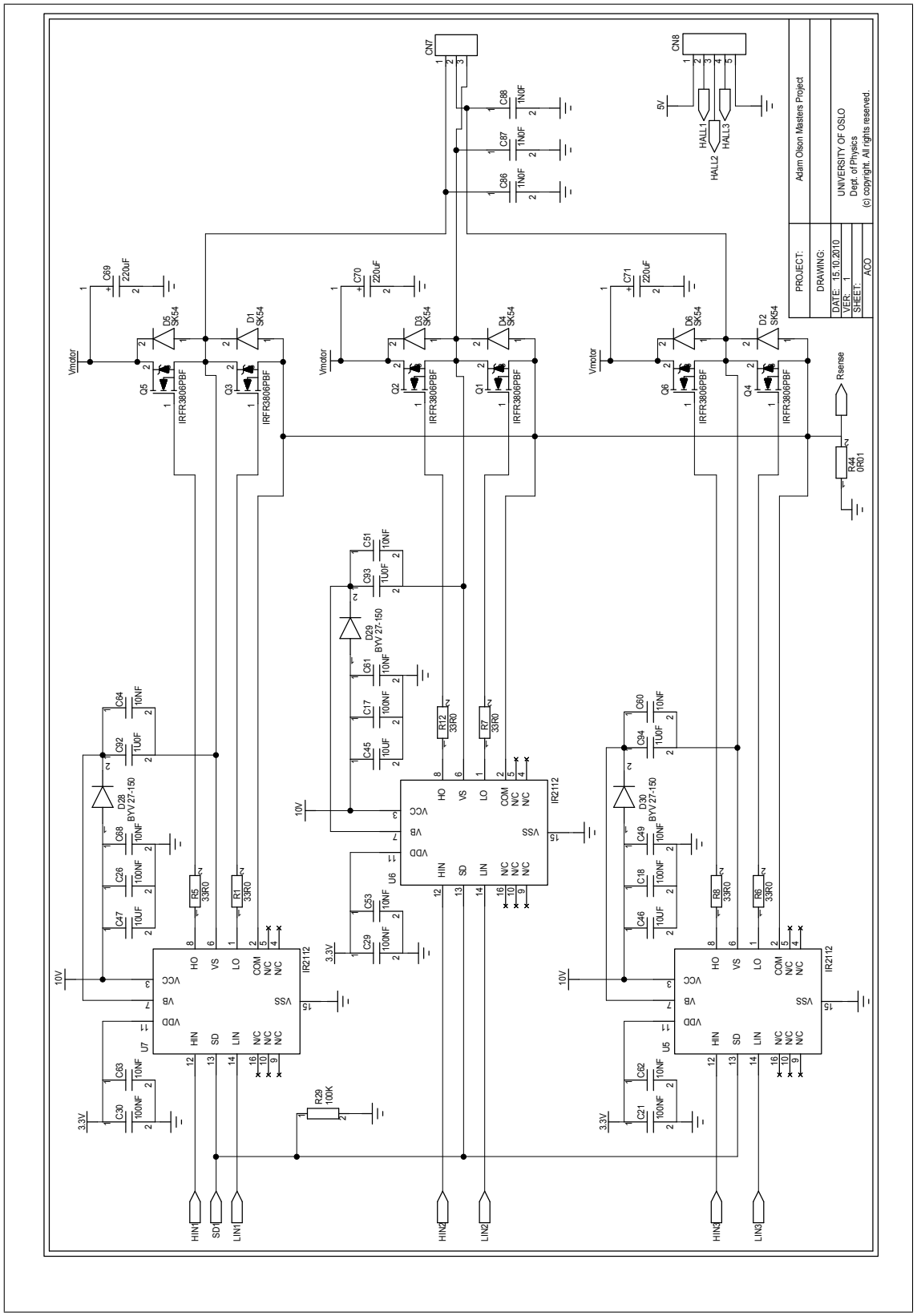


Figure A.5: BLDC motor controller, sheet 5, 3-phase half bridges and gate drivers.

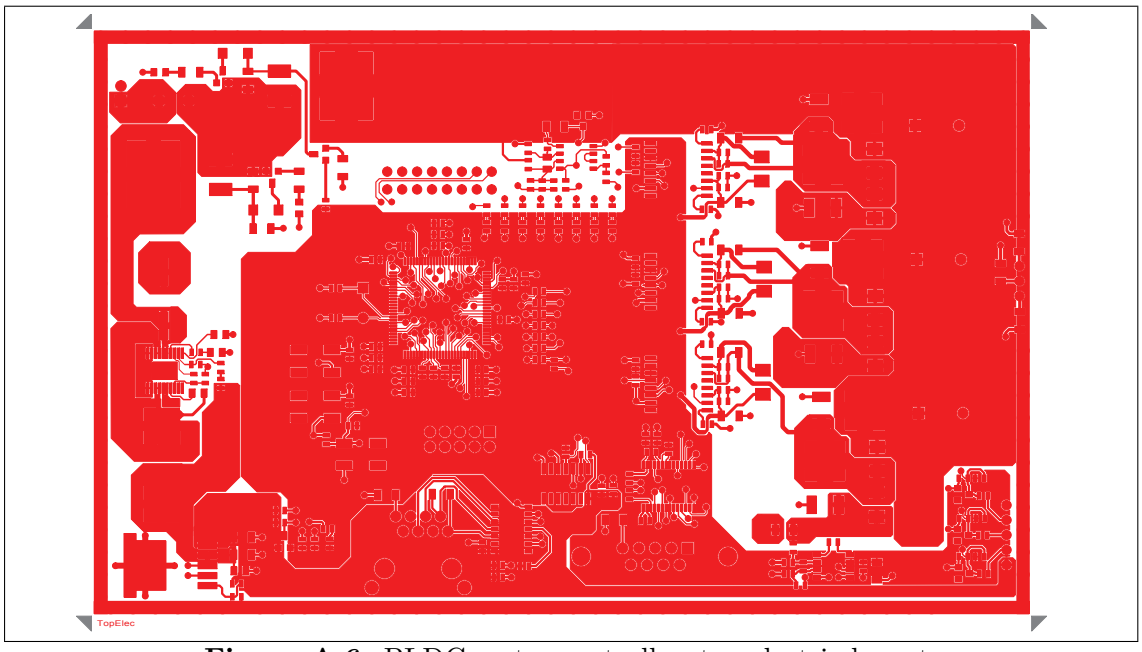


Figure A.6: BLDC motor controller, top electric layout.

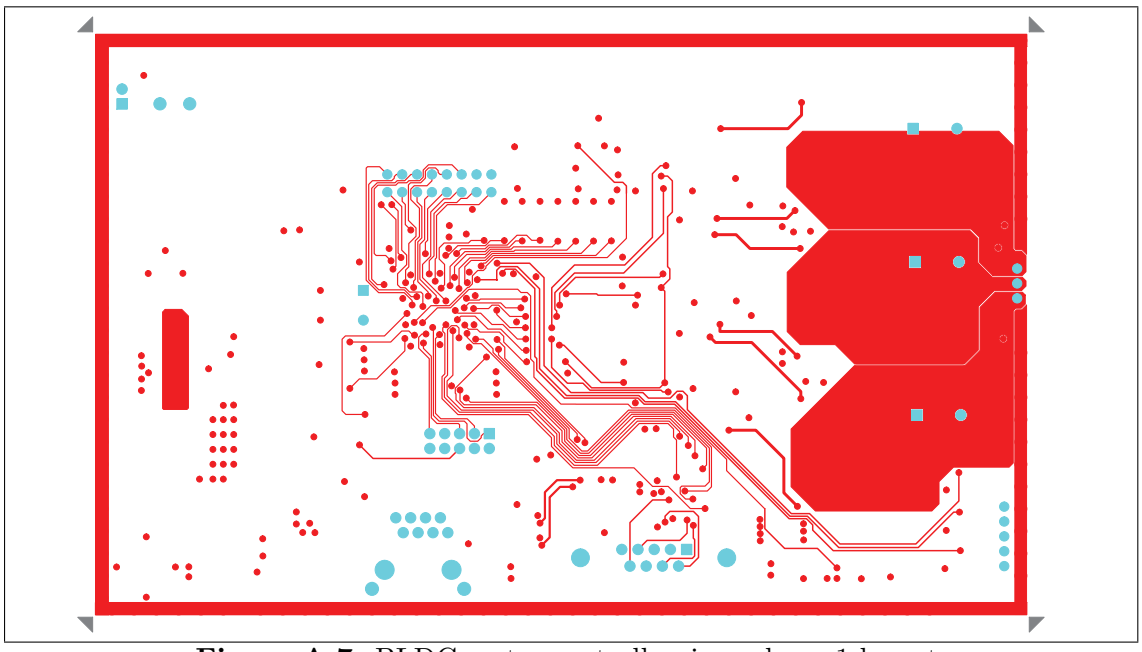


Figure A.7: BLDC motor controller, inner layer 1 layout.

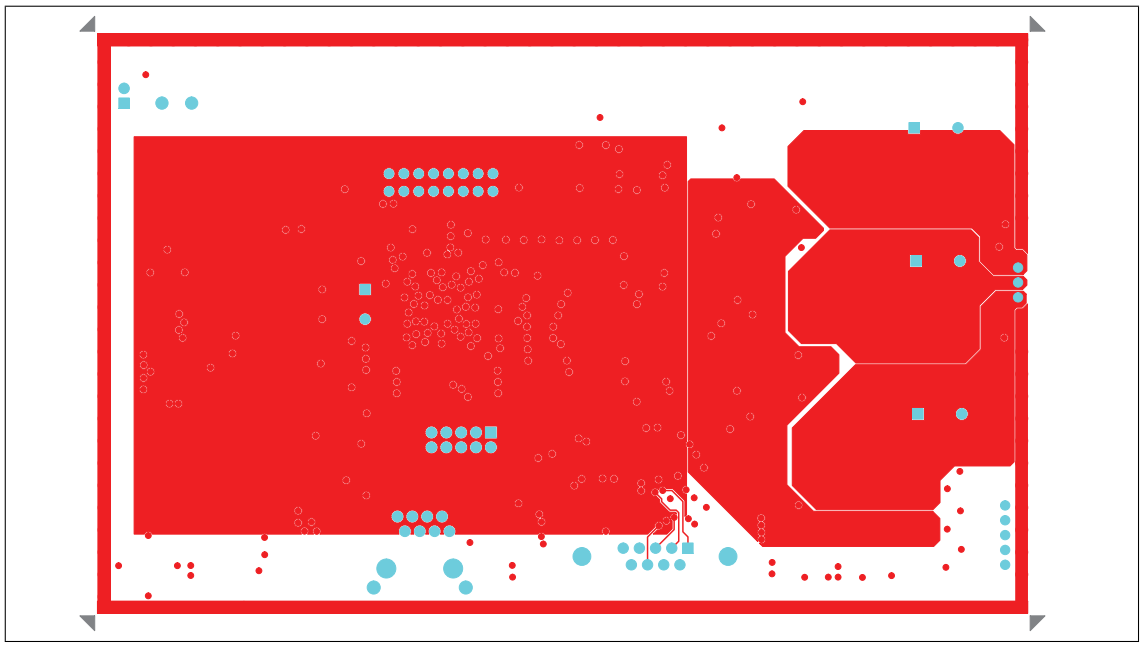


Figure A.8: BLDC motor controller, inner layer 2 layout.

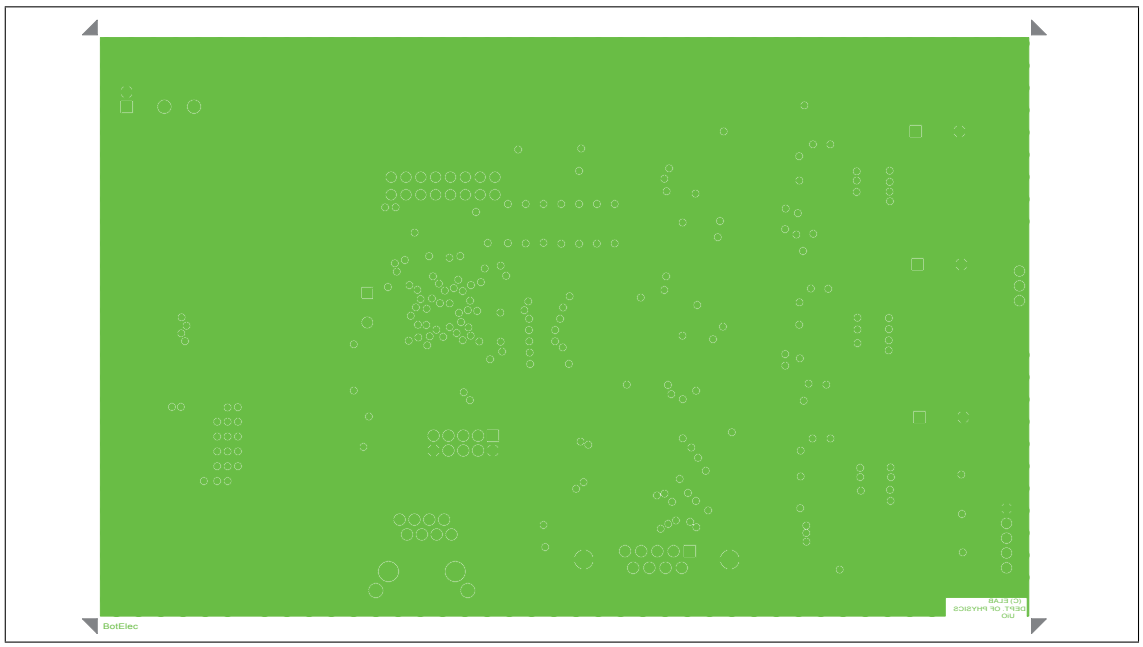


Figure A.9: BLDC motor controller bottom electric layout.

A.0.2 Transducer Driver

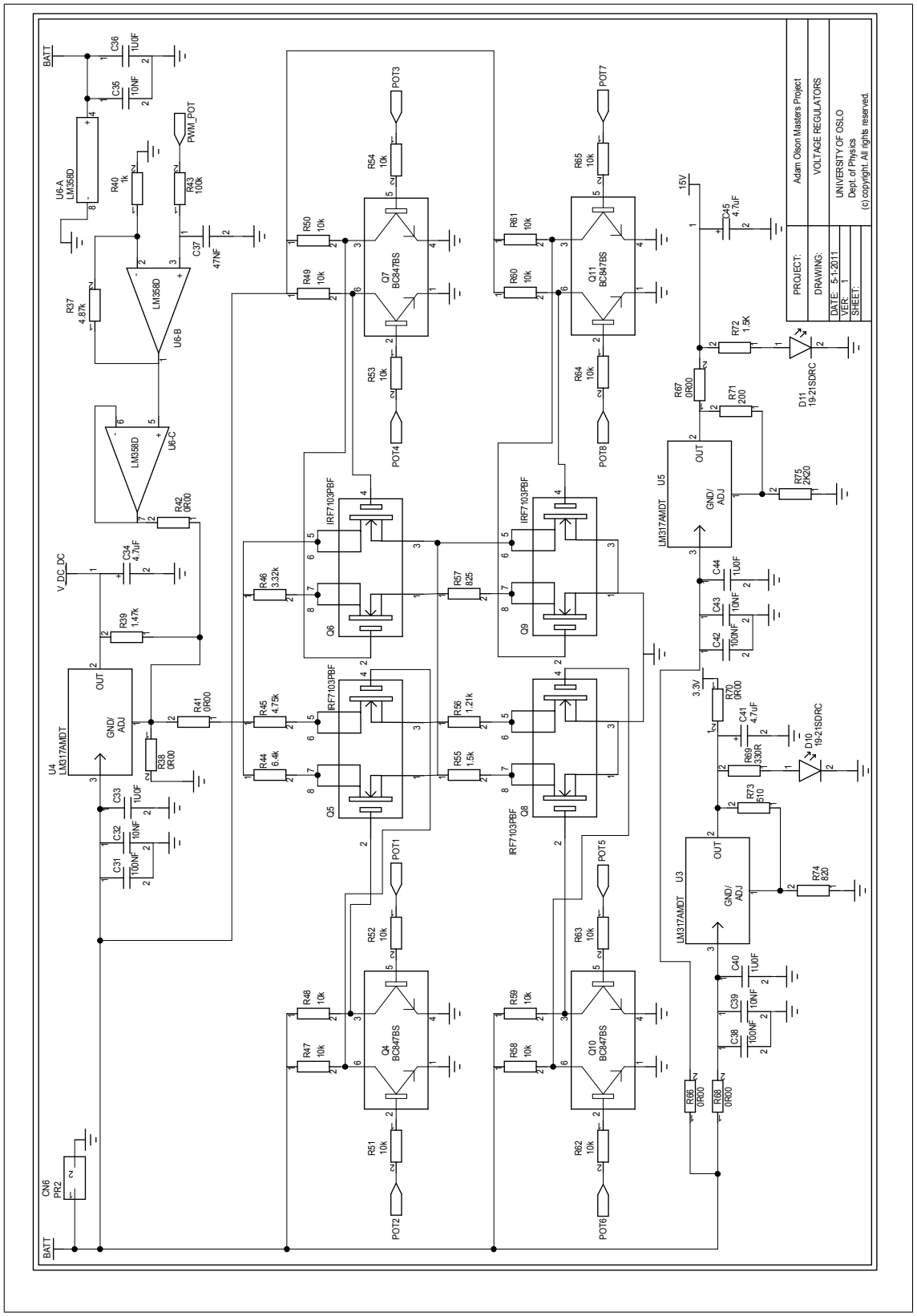


Figure A.10: Hydrophone driver, board 1, sheet 1, power supplies.

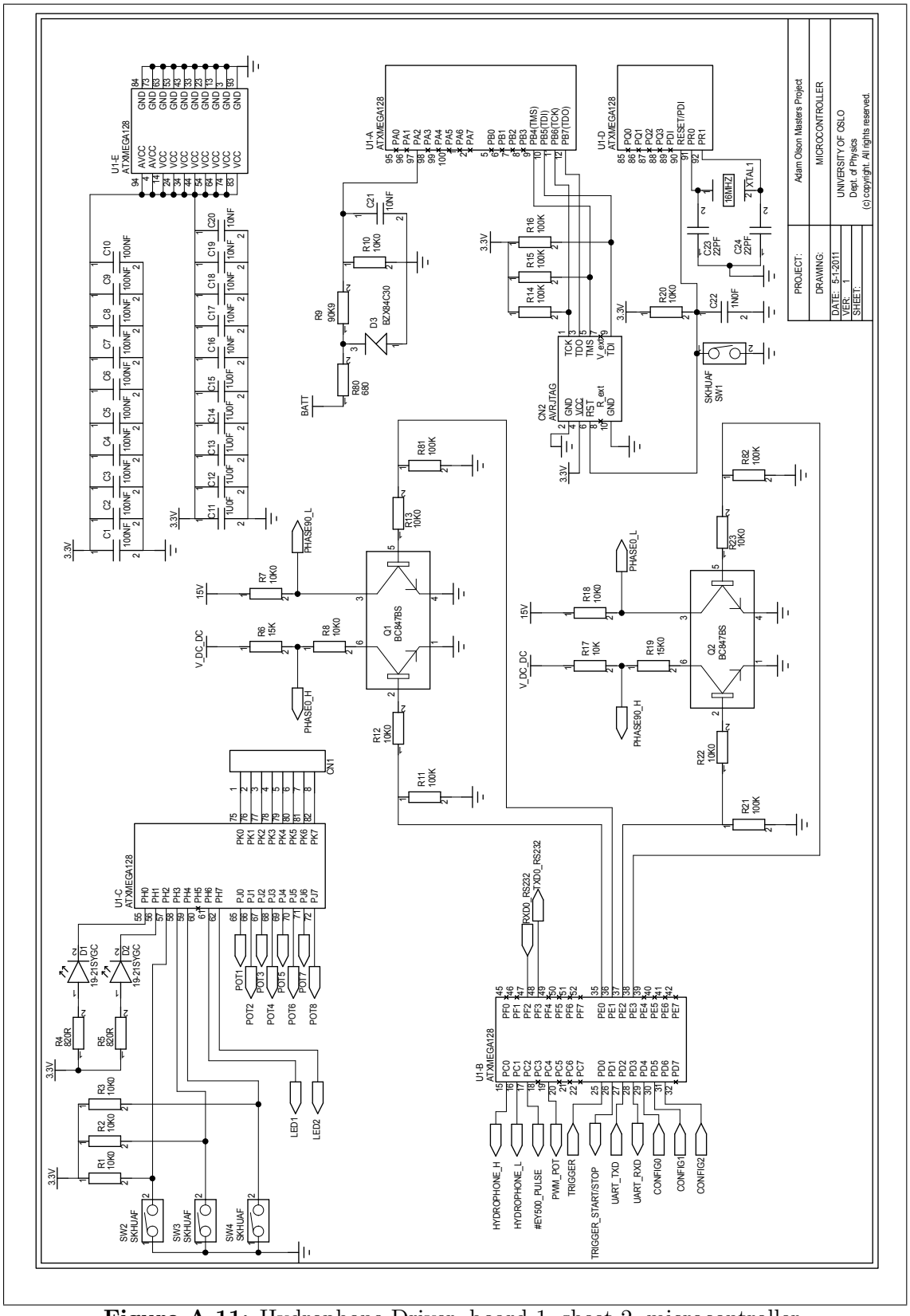


Figure A.11: Hydrophone Driver, board 1, sheet 2, microcontroller.

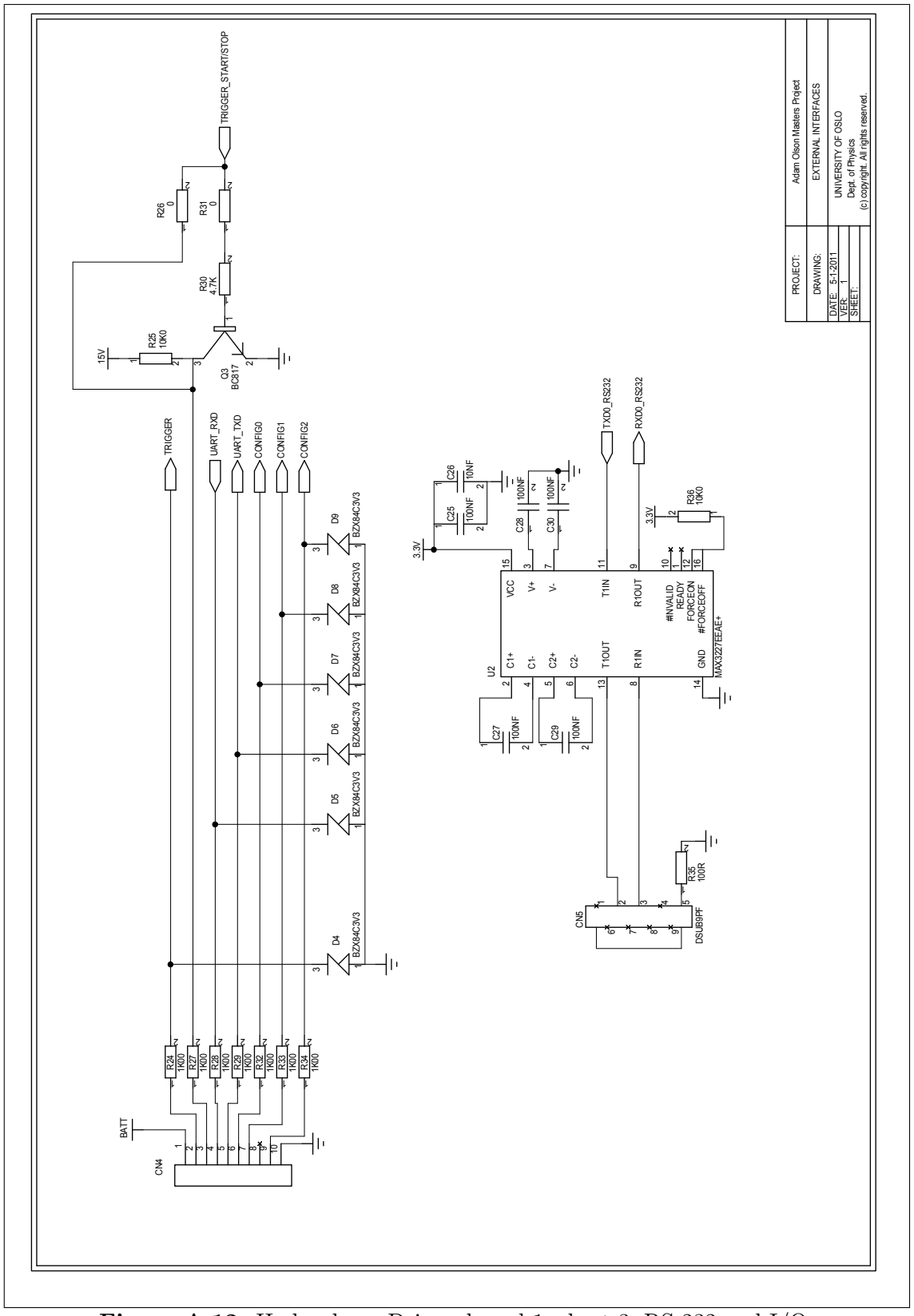


Figure A.12: Hydrophone Driver, board 1, sheet 3, RS-232 and I/O.

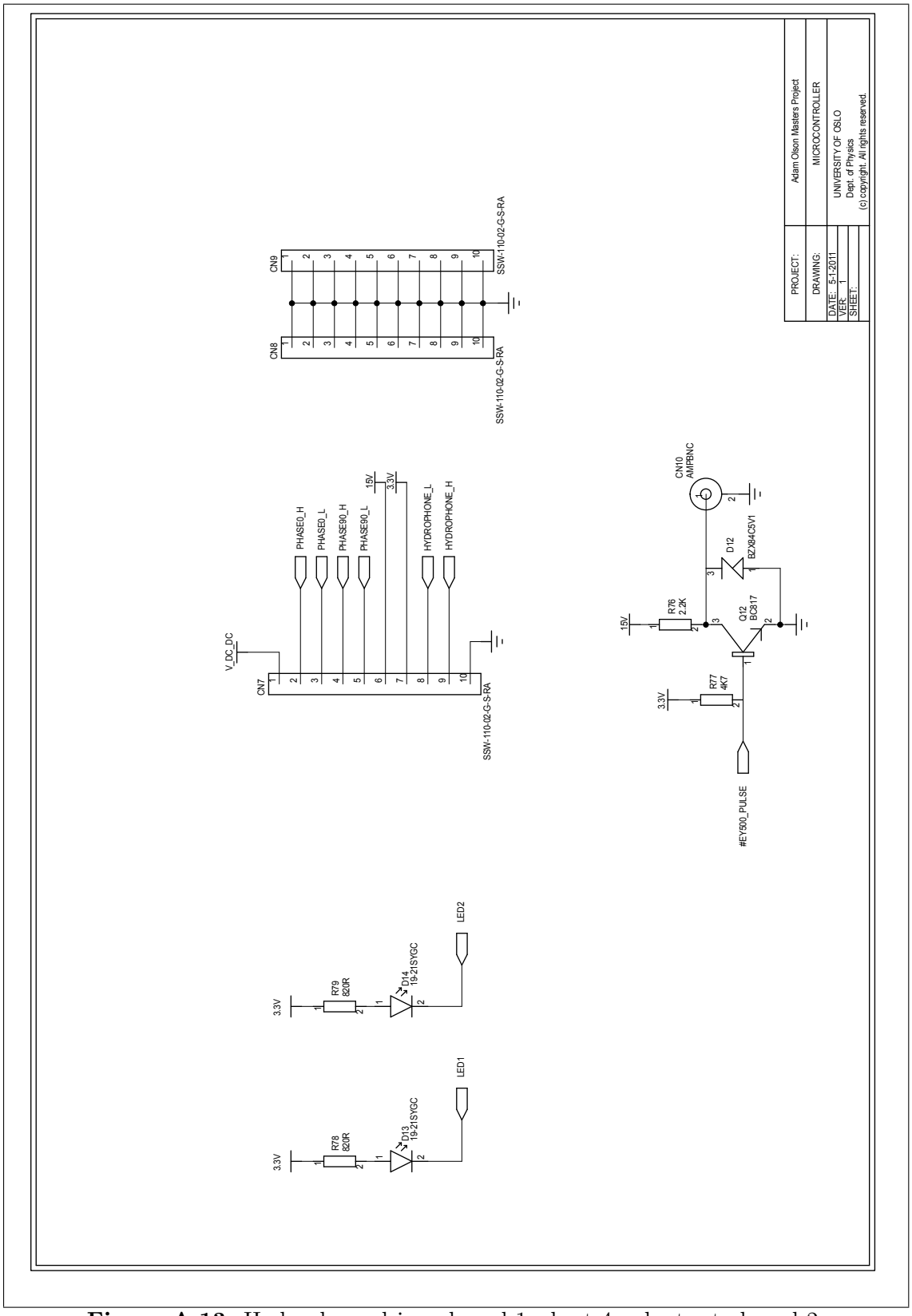


Figure A.13: Hydrophone driver, board 1, sheet 4, adapter to board 2.

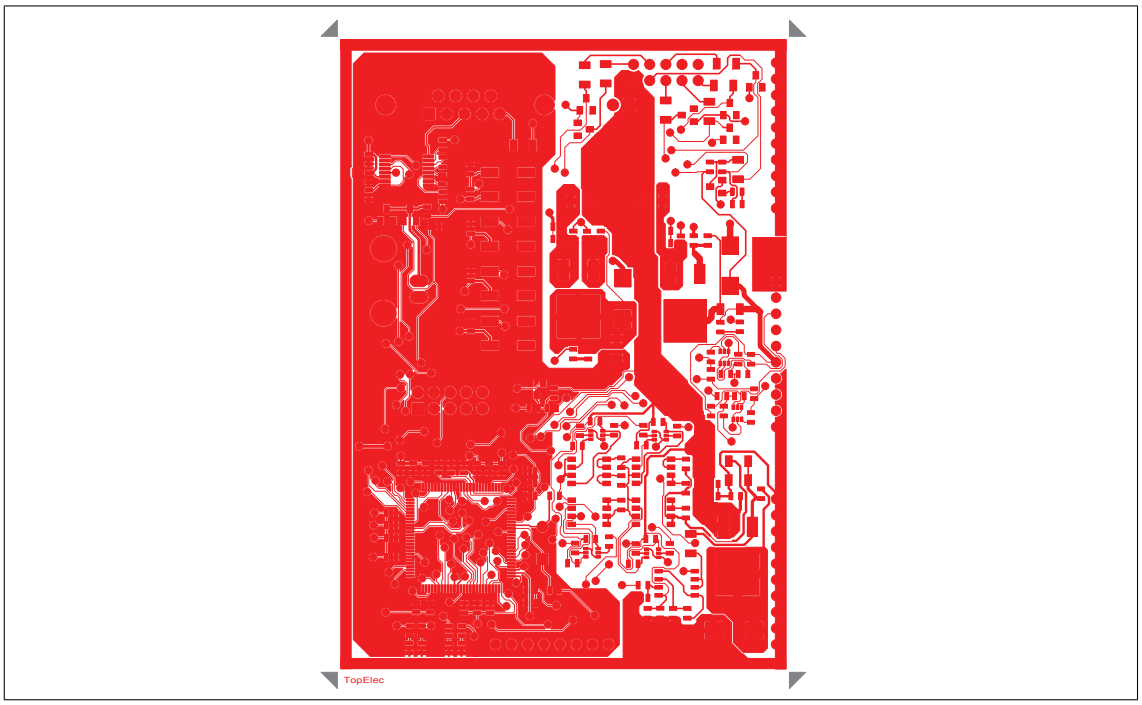


Figure A.14: Hydrophone driver, board 1, top electric layer

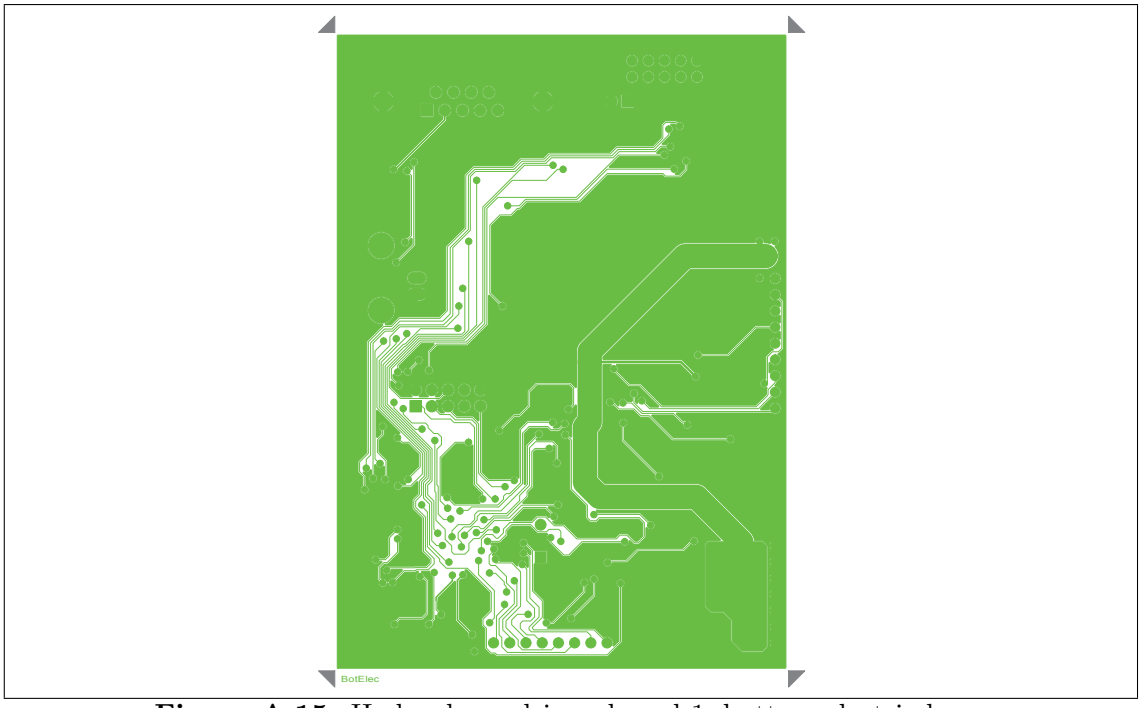


Figure A.15: Hydrophone driver, board 1, bottom electric layer

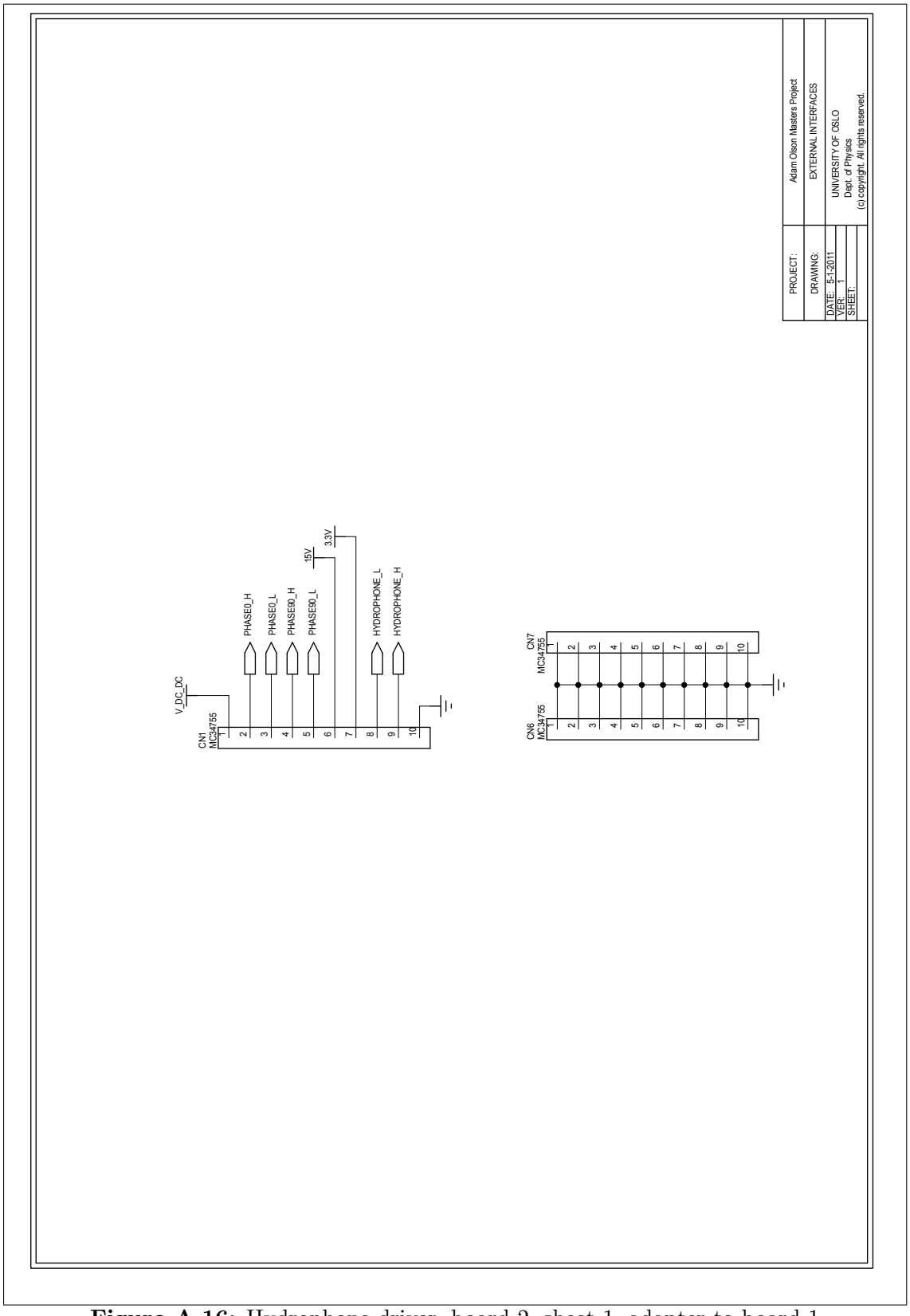


Figure A.16: Hydrophone driver, board 2, sheet 1, adapter to board 1

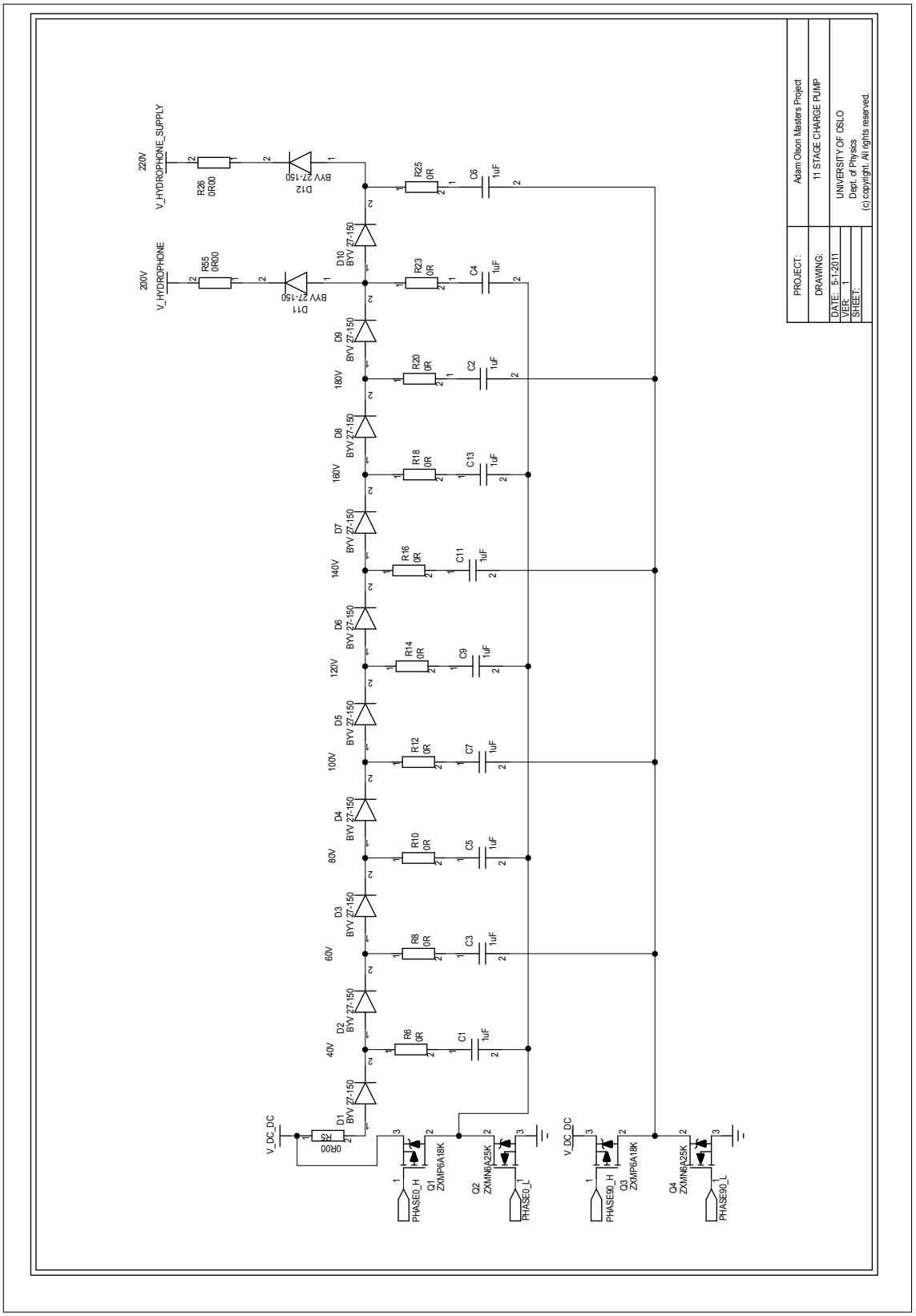


Figure A.17: Hydrophone driver, board 2, sheet 2, Dickson charge pump.

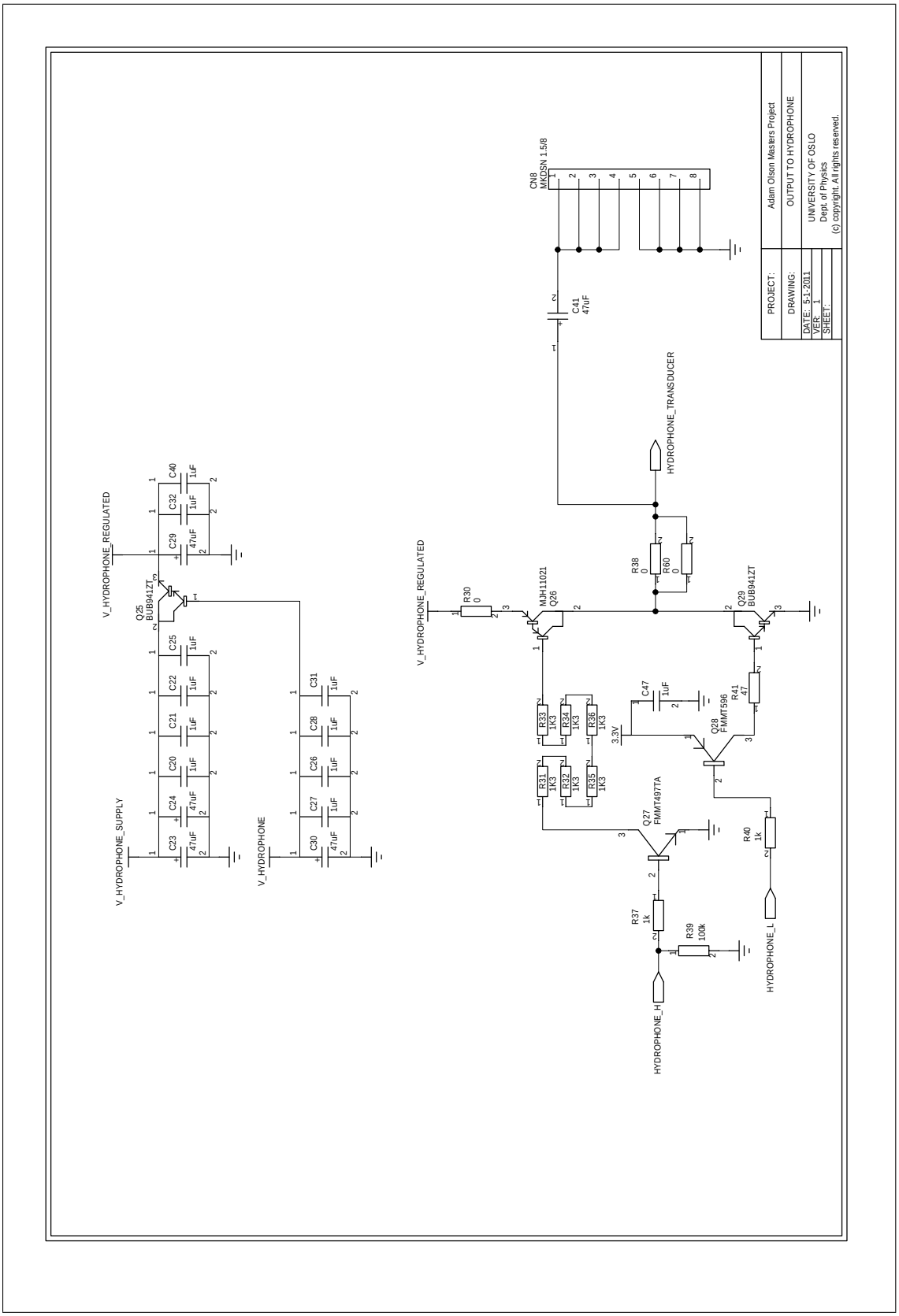


Figure A.18: Hydrophone driver, board 2, sheet 3, emitter-follower regulator with capacitor banks.

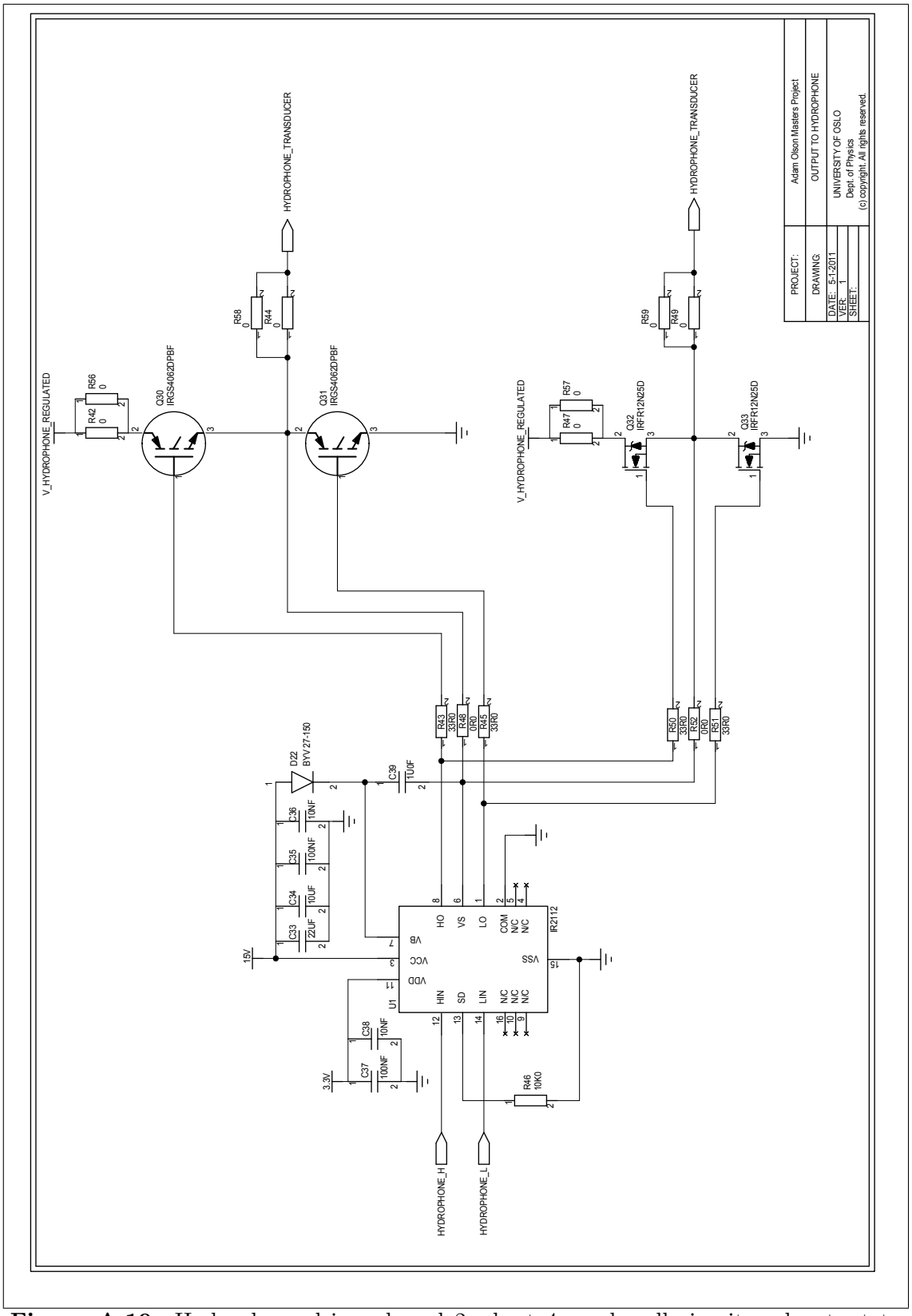


Figure A.19: Hydrophone driver, board 2, sheet 4, push-pull circuit and output to hydrophone.

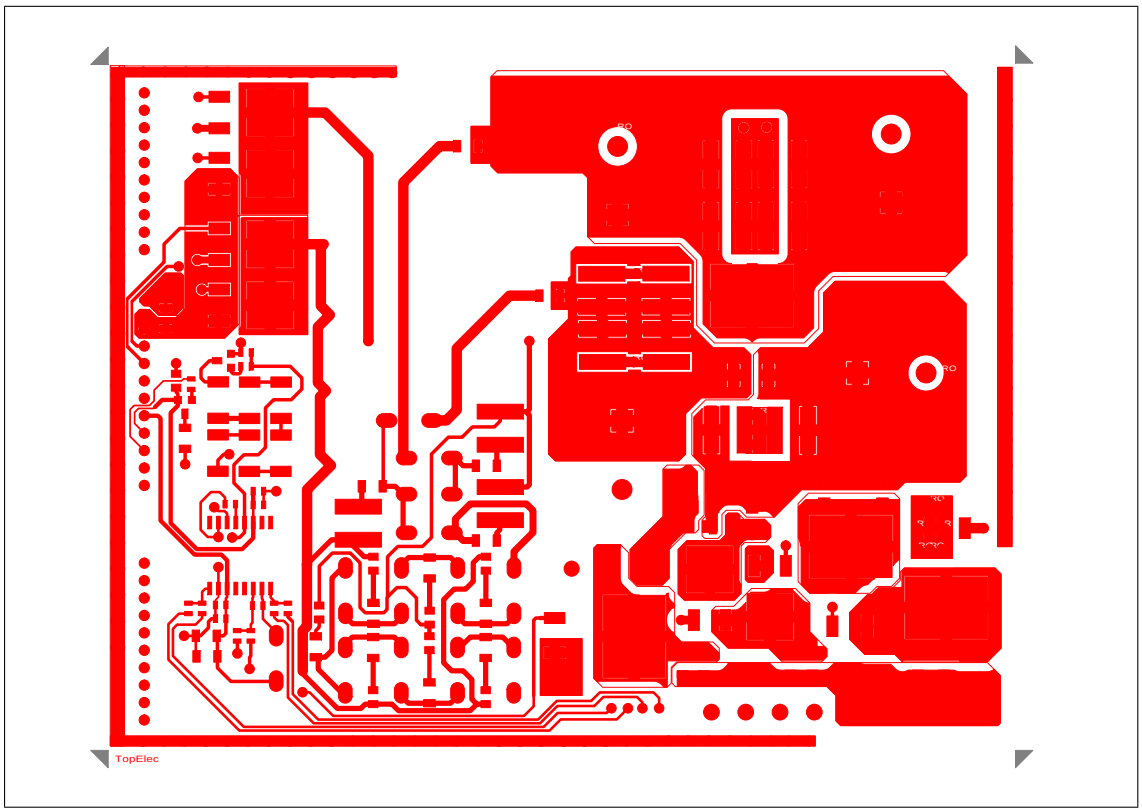


Figure A.20: Hydrophone driver, board 2, top electric layer

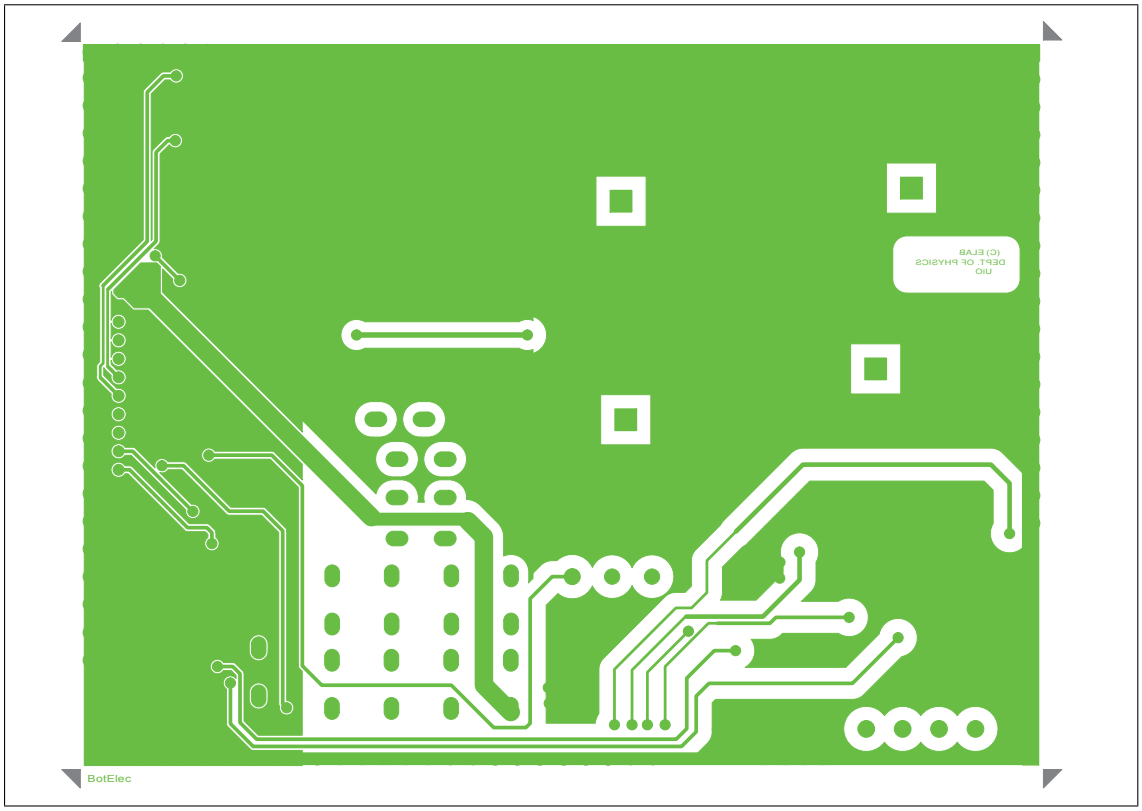


Figure A.21: Hydrophone driver, board 2, bottom electric layer

A.0.3 Measurement Board and USB Adapter

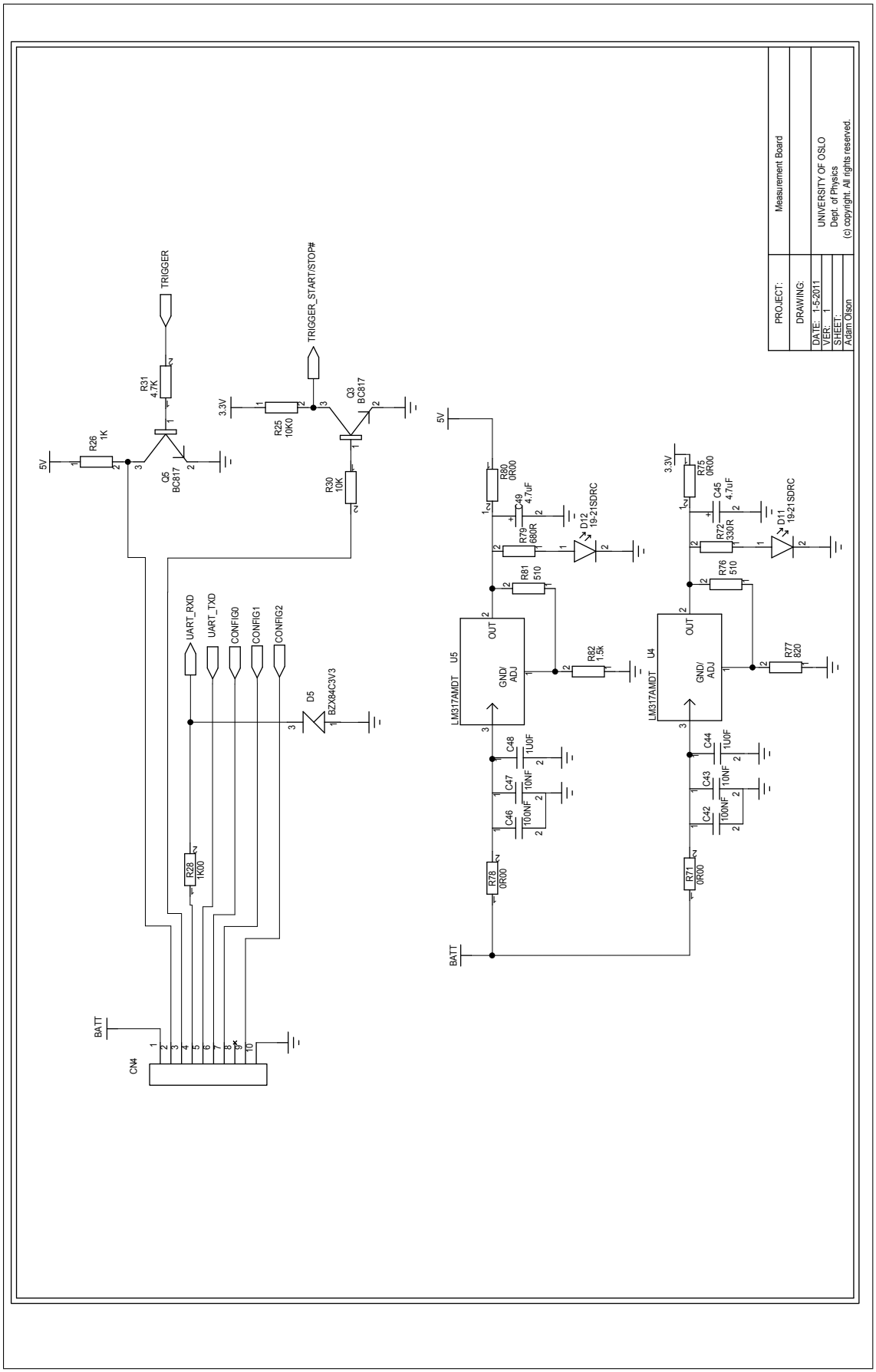


Figure A.22: Measurement Board, sheet 1, adapter to board 1

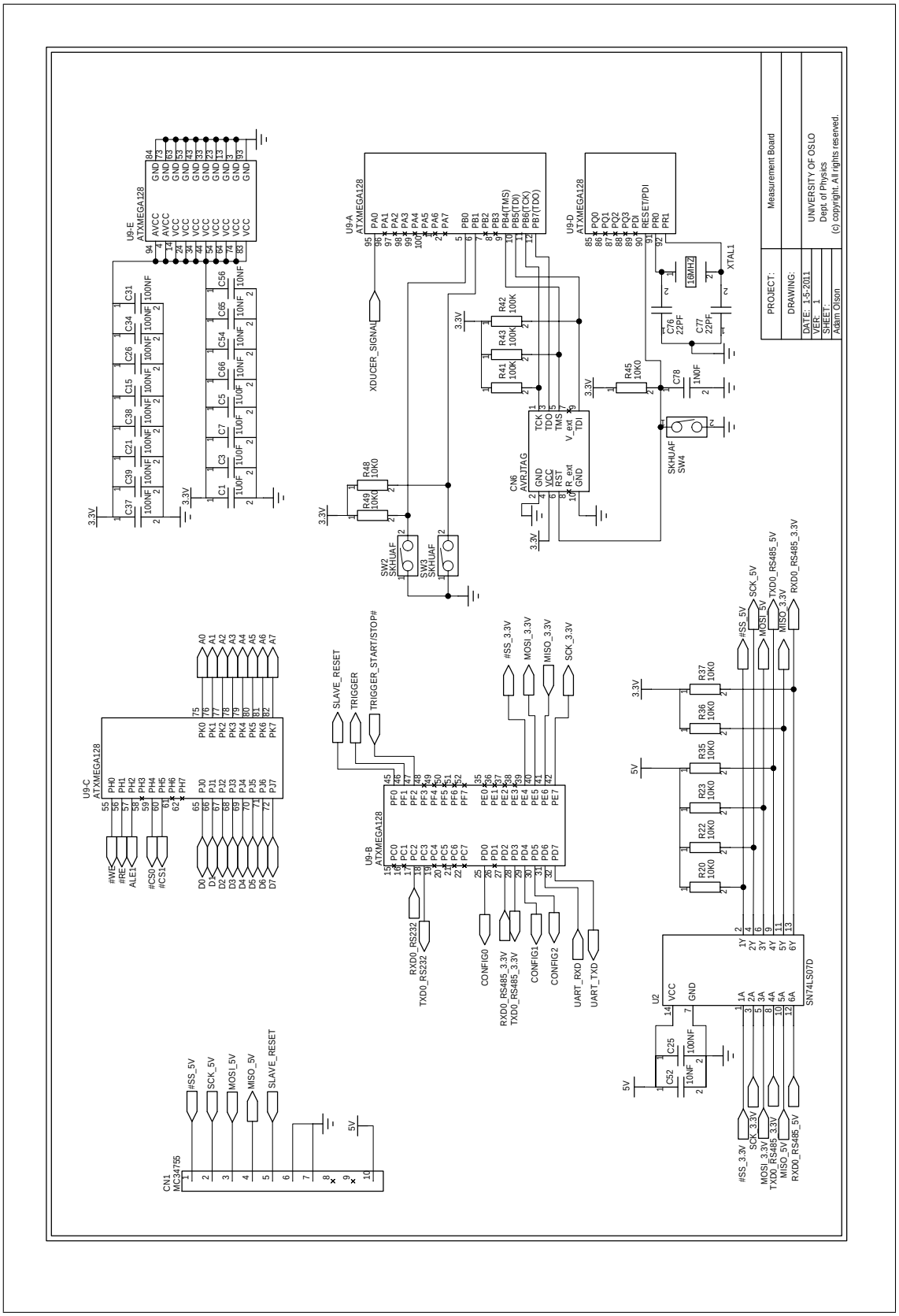


Figure A.23: Measurement Board, sheet 2, microcontroller and connector to USB adapter.

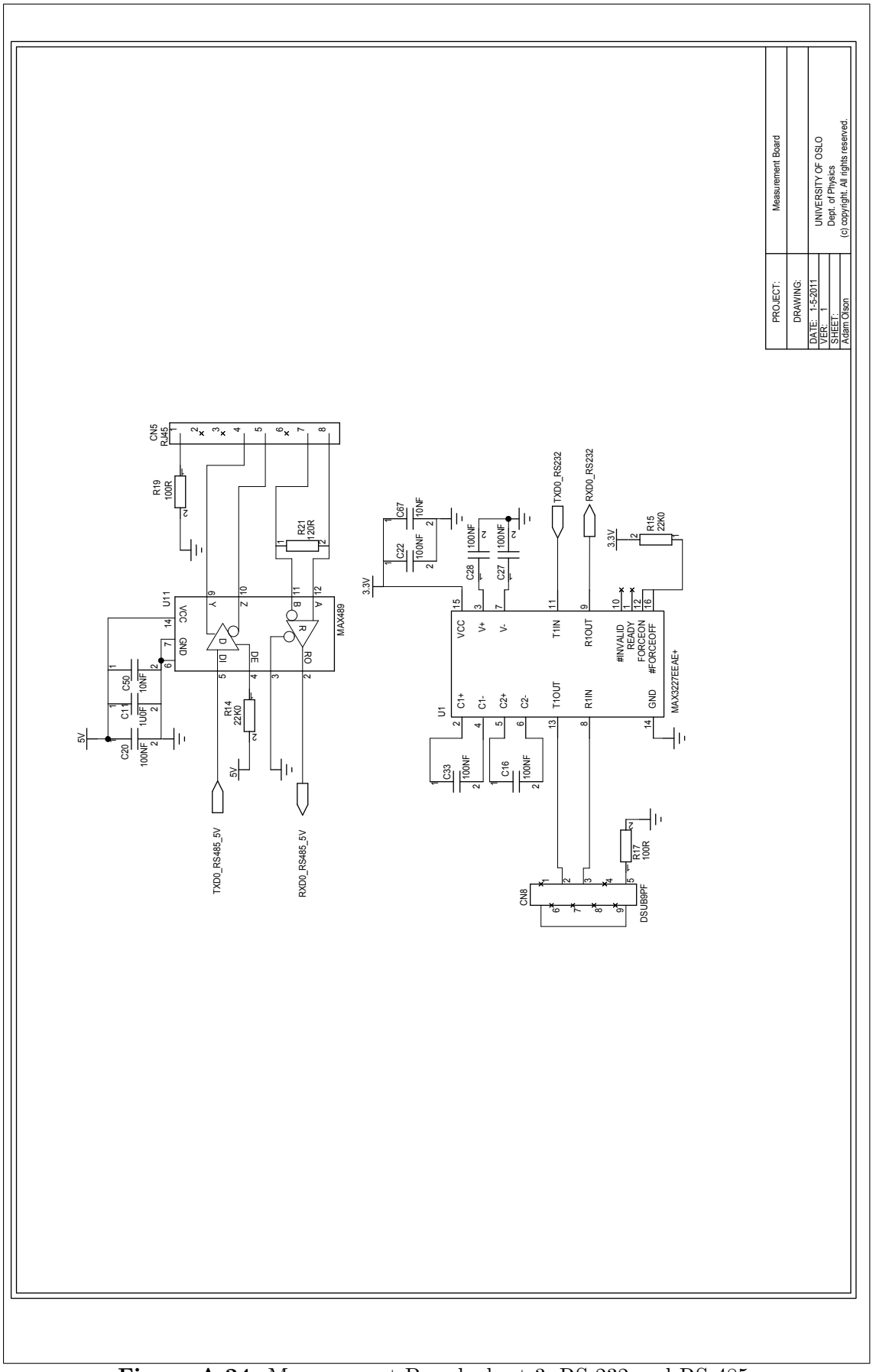


Figure A.24: Measurement Board, sheet 3, RS-232 and RS-485.

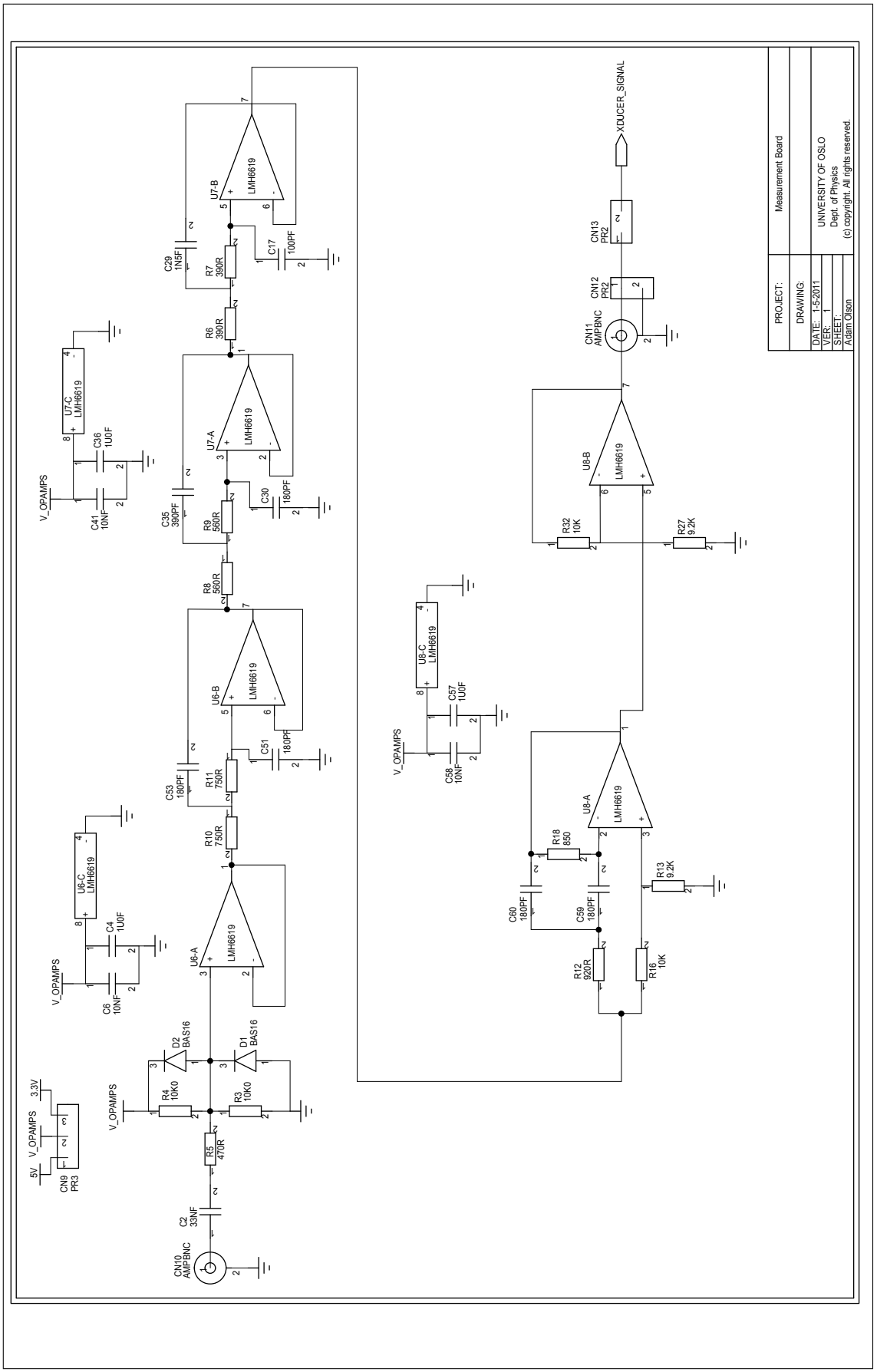


Figure A.25: Measurement Board, sheet 4, butterworth filter .

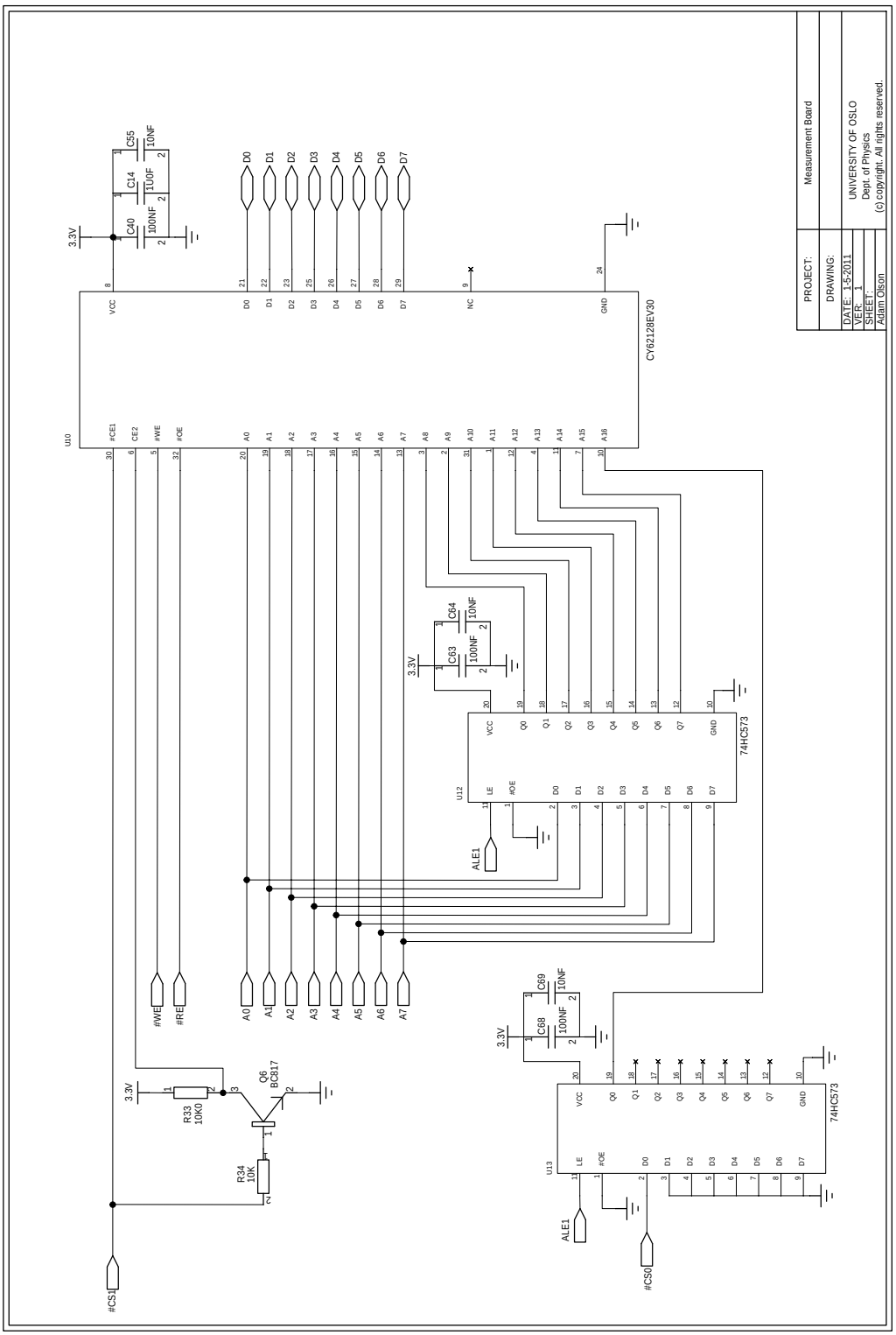


Figure A.26: Measurement Board, sheet 5, SRAM memory.

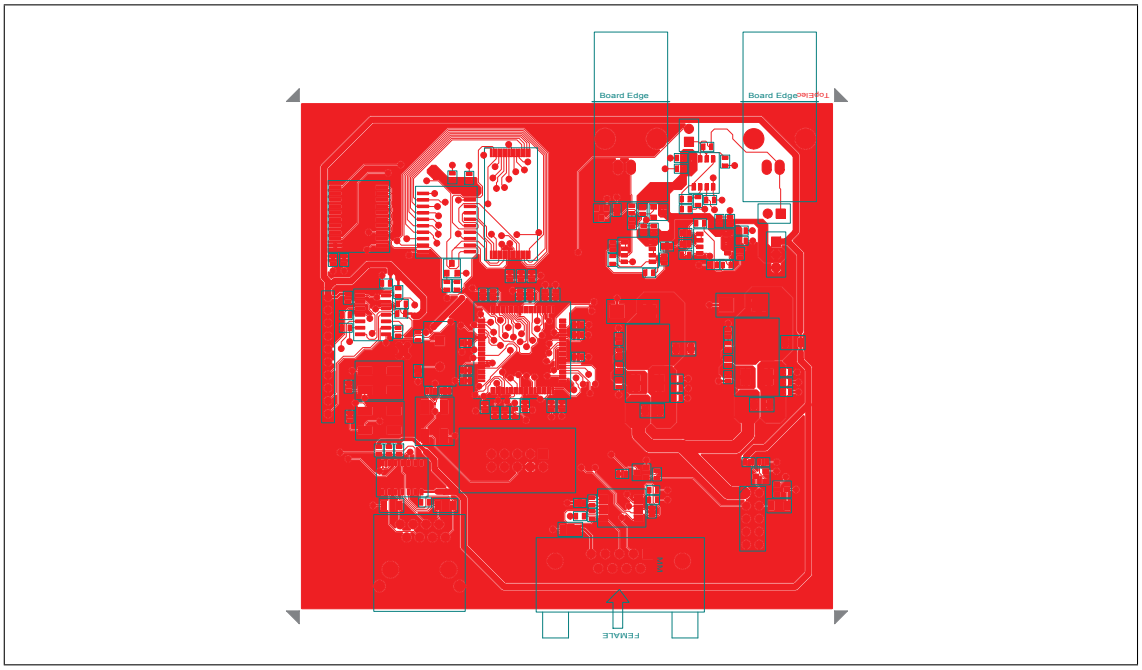


Figure A.27: Measurement Board, top electric layer

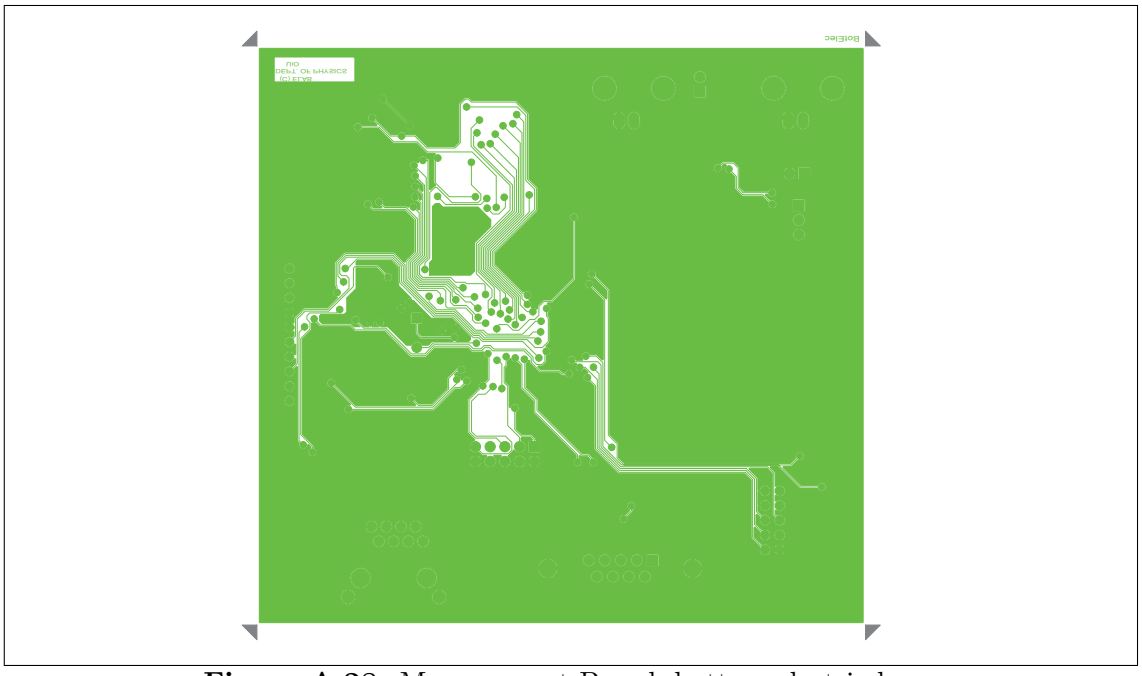


Figure A.28: Measurement Board, bottom electric layer

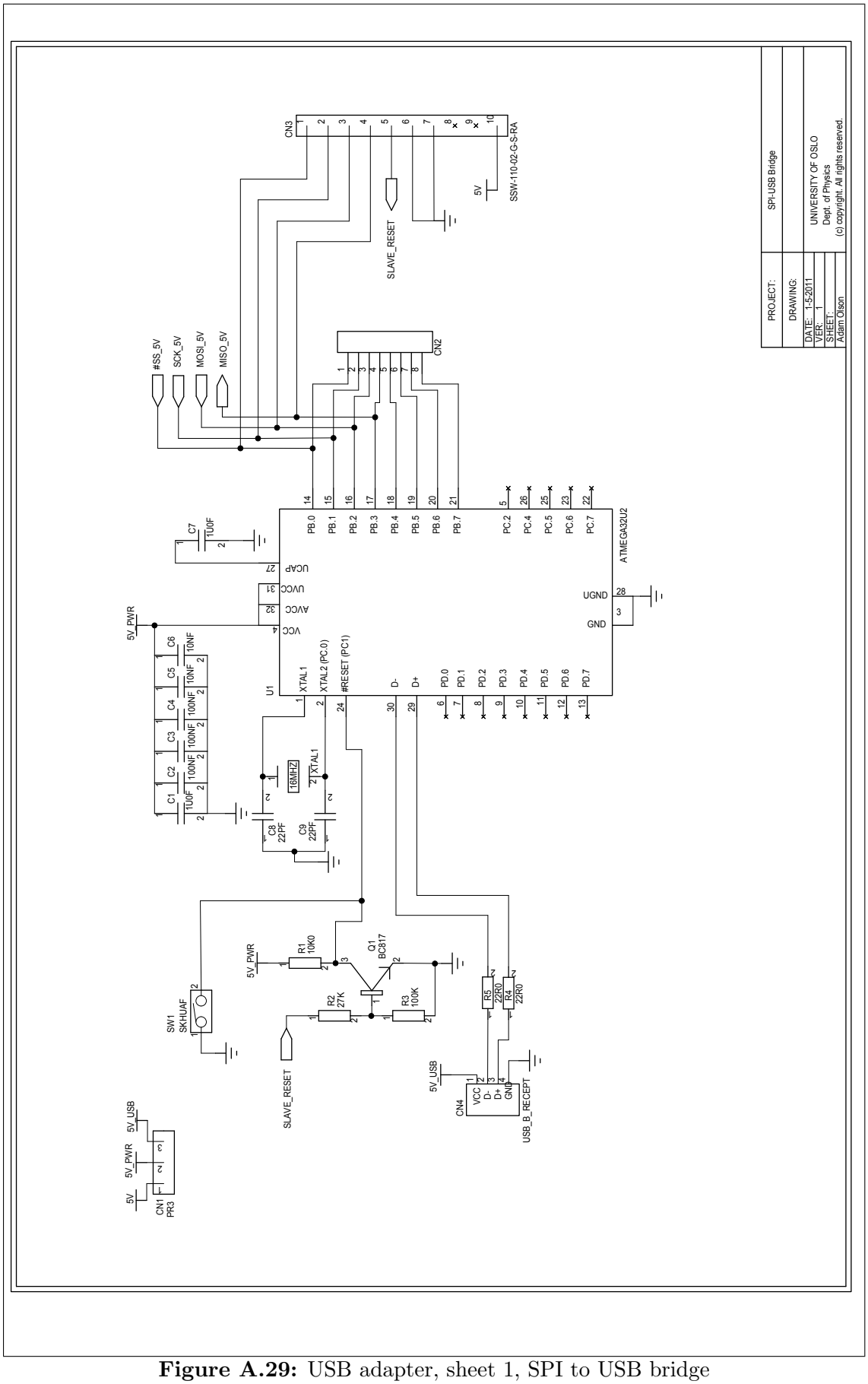


Figure A.29: USB adapter, sheet 1, SPI to USB bridge

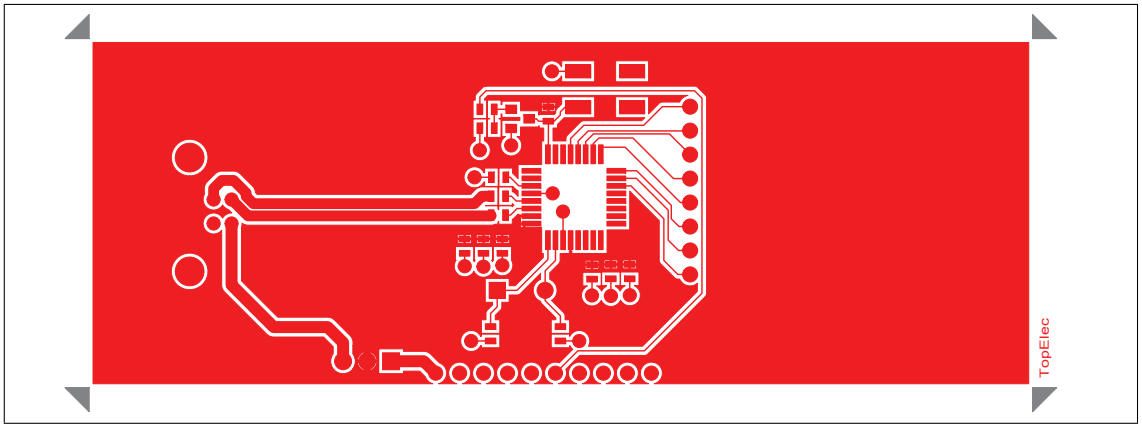


Figure A.30: USB adapter, top electric layer

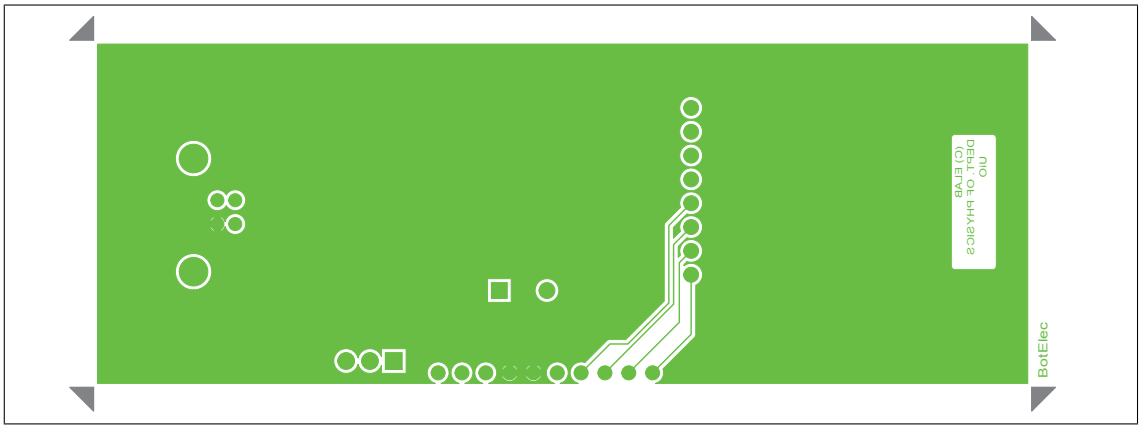


Figure A.31: USB adapter, bottom electric layer

Appendix B

C Code for ATXMEGA128A1 BLDC Motor Controller

```
/* Adam Olson
 4/10/2009
 UiO Masters program
 Motor controller program

*/


#define F_CPU 32000000UL

#include <avr/io.h>  //  ATXMEGA128A1 registers
#include <avr/interrupt.h>  // ISR signal vectors calls
#include <util/delay.h>   // System delay calls
#include <stdint.h>    // Standard integers
#include <stdlib.h>    // Standard library
#include <stdbool.h>   // Boolean variable
#include <math.h>  // math functions

#define TOPL 0xA5    // Define the PWM frequency
#define TOPH 0x01    // Define the PWM frequency
#define UP true   //  Up is down in this code
#define DOWN false  //  Down is up in this code
#define MAX_HEIGHT  12000 //  Sets the maximum command position


// Function prototypes
int initialize_system();   //  Initializes the 32MHz internal RC oscillator
int initialize_position(); //  begin moving the platform up
int USART_INIT();  //  Initializes the USART peripheral
int read_message();  //  check the new_message flag and read
int control_loop();  //  control loop that repeats every 2ms
int16_t BCD_to_revolutions(); //  Converts ASCII revolutions
int revolutions_to_BCD(); //  Converts the motors 16 bit integer
int move_increment();  //  Increments the command position
int move_decrement();  //  Decrements the command position
int commute_motor();  //  Reads the hall sensor and chooses
int set_PWM(uint16_t);  //  Reloads the Timer E period registers
void sense_position();  //  updates current position


volatile char rev_out[5];  // String for output of current position
volatile char message[5];  // String buffer to collect rev command
volatile int message_index = 0;  // index var to keep track of message buffer position
```

```
volatile char receive_char;
volatile char command;   // the message command from app
volatile bool start_message = false;
volatile bool new_message = false; // flag to see if the uart has received a new character
volatile bool start_revolutions; // flag to show a new position command has begun
volatile bool DIRECTION;  // Commanding direction state of the motor, commutation base on this
volatile bool control_update = false;
volatile bool reed_initialized = false;
volatile bool new_revolutions;  // current sampled position of motor

volatile bool transmit_wait = false; // flag to wait to send a new character to the uart


volatile int COMMUTATION[6];  // commutation array
volatile int HALL_COMB[6];  // hall combination array
volatile int HALL_COMB_PREV = 0; // variable to store the previous hall
volatile int hall_index;  //
volatile int hall_check;  //
volatile int16_t rev_value = 0;  // current revolution value to convert BCD/ASCII


volatile uint16_t PWM = 0;  //  PWM value to be loaded into time E at any time


volatile bool SENSED_DIRECTION = DOWN; // Direction that the controller detects the motor is moving
volatile int HALL_COUNT = 0x00;  //  Integrating value of hall counts to keep track of position
volatile int16_t COMMAND_ROTATIONS = 0; // Desired position given from the user application
volatile int16_t SAMPLED_ROTATIONS = 0; //  Sample of the current number of rotations for the control c
volatile int16_t ROTATIONS = 0;   // Raw Position of motor updated in real time.
volatile int16_t PREVIOUS_ROTATIONS = 0; //Previous sampled motor position
volatile int16_t error;   // error between current position and commanded position
```

```
ISR(PORTF_INT0_vect)
{ //  Interrupt from the hall sensor.
 // commutation sequence must be used to move again.
 cli();
 sense_position();
 commute_motor();
 sei();
}
```

```
ISR(TCD0_OVF_vect)
{ //  A ~2ms control loop  initiated by this timer interrupt
 TCD0.INTCTRLA = 0x00;//  Turn off the interrupt
 //control_update = true;
 SAMPLED_ROTATIONS = ROTATIONS;
   if (control_update == true)
   {
    control_loop();
   }
 TCD0.INTCTRLA = 0x02; //  Turn the interrupt back on
}
```

```
ISR(TCC0_OVF_vect)
{ //  A ~2ms control loop  initiated by this timer interrupt
 TCC0.INTCTRLA = 0x00;//  Turn off the interrupt
 //control_update = true;
 PORTJ.OUT = 0x01;
 _delay_ms(1);
 PORTJ.OUT = 0x00;
}

//////////////    REED SENSOR INTERRUPT     ///////
ISR(PORTB_INT0_vect)
{
 PORTB.INTCTRL = 0x00;
 set_PWM(0x0000);
 reed_initialized = true;
 PORTB.INTCTRL = 0x01;

}



//////////////////////   USART RECEIVE INTERRUPT    //////////////
ISR(USARTC0_RXC_vect)
{
 receive_char = USARTC0.DATA;


 if (receive_char == '%')
  {
   start_revolutions = true;
   message_index = 0;

  }
 else if (start_revolutions == true)
  { if (receive_char == 0x20)
   {
    receive_char = 0x30;
   }
   message[message_index] = receive_char;
   message_index = message_index + 1;
   if (message_index == 5)
    {
     new_message = true;
     new_revolutions = true;
     start_revolutions = false;
     message_index = 0;
    }

  }
 else
  {
   new_message = true;
   command = receive_char;
  }


}



////////////////////////////////////   USART TRANSMIT INTERRUPT    ////////////////////////////////////
ISR(USARTC0_TXC_vect)
{
transmit_wait = false;
}



int main()
{ initialize_system();
 USART_INIT();
```

```
// Pins to Motor FETs
PORTE.OUT = 0x00;
PORTE.DIR = 0xFF;

// Pins to LEDs
PORTH.DIR = 0xFF;
PORTH.OUT = 0x00;
_delay_ms(2000);

// Hall Sensor Pins
PORTF.DIR = 0x00; // don't interrupt until in control mode


// Reed Sensor and Push Button Pins
PORTB.DIR = 0x00;
PORTB.PIN0CTRL = 0x02;
PORTB.PIN1CTRL = 0x02;
PORTB.PIN2CTRL = 0x02;
PORTB_INT0MASK = 0x03;
PORTB.INTCTRL = 0x01;

 // Hall Sensor Pins
PORTF.DIR = 0x00;
PORTF.INT0MASK = 0x07;
PORTF.PIN0CTRL = 0x00;
PORTF.PIN1CTRL = 0x00;
PORTF.PIN2CTRL = 0x00;
PORTF.INTCTRL = 0x03;


 // Pins to 3,3V Triggers
PORTJ.DIR = 0xFF;  //set as output


// Define Hall Combinations
HALL_COMB[0] = 0b00000001;  // WU
HALL_COMB[1] = 0b00000011;  // WV
HALL_COMB[2] = 0b00000010;  // UV
HALL_COMB[3] = 0b00000110;  // UW
HALL_COMB[4] = 0b00000100;  // VW
HALL_COMB[5] = 0b00000101;  // VU

//      V  U  W Define Commutation Sequence
COMMUTATION[0] = 0b00010010;  // WU
COMMUTATION[1] = 0b01000010;  // WV
COMMUTATION[2] = 0b01001000;  // UV
COMMUTATION[3] = 0b00001100;  // UW
COMMUTATION[4] = 0b00100100;  // VW
COMMUTATION[5] = 0b00110000;  // VU

// enable all priority level interrupts
PMIC_CTRL = 0x07;
sei();


// configure commutation PWM signal
/* TCE0 - Timer/Counter E0 */
TCE0.CTRLA = 0b00000001; // using clock with prescaler of 1 (32 MHz)
//TCE0.CTRLB = 0b11110101; // all CCxEN bits set so port will output
TCE0.CTRLB = 0xF5;
//TCE0_CTRLC = 0b00000000; // R/W Compare output values
TCE0.CTRLD = 0b00000000; // events disabled
TCE0.CTRLE = 0x00;  // disable bytemode
TCE0.INTCTRLA = 0b00000000; // disable timer error and timer over/underflow interrupts
TCE0.INTCTRLB = 0b00000000; // disable timer compare or capture interrupts
TCE0.PERL = TOPL;
TCE0.PERH = TOPH;
set_PWM(0x0000);
// Patter generation mode, DTI enabled for all channels
AWEXE.CTRL = 0x2F;
AWEXE.DTLS = 0x01;
```

```
AWEXE.DTHS = 0x01;
  //  All ports off for now
AWEXE.OUTOVEN = 0x00;


// configure trigger delays
/* TCC0 - Timer/Counter C0 */
TCC0.CTRLA = 0x07;  // using clock with prescaler of 1024 (31,25 kHz)
TCC0.CTRLB = 0x00;
TCC0.CTRLD = 0b00000000; // events disabled
TCC0.CTRLE = 0x00;   // disable bytemode
TCC0.INTCTRLA = 0x00;
TCC0.INTCTRLB = 0b00000000; // disable timer compare or capture interrupts
TCC0.PERL = 0xBB;
TCC0.PERH = 0x00;


// control loop interrupt
TCD0.CTRLA = 0b00000001; // using clock with prescaler of 1 (32 MHz)
TCD0.CTRLB = 0x00;
TCD0.CTRLD = 0b00000000; // events disabled
TCD0.CTRLE = 0x00;   // disable bytemode
TCD0.INTCTRLA = 0x01;
TCD0.INTCTRLB = 0b00000000; // disable timer compare or capture interrupts
TCD0.PERL = 0x00;
TCD0.PERH = 0x7D;


 while (!reed_initialized)
  {


   if (new_message ==  true)
    { read_message();
     new_message = false;
    }
  }



ROTATIONS = 0;
HALL_COUNT = 0;
control_update = true;
while(1)
{


if (reed_initialized)
{
 //move_motor_down();
 _delay_ms(10);
 revolutions_to_BCD();
 read_message();

{









 }

}




};
```

```
return 1;
}
```

```
int initialize_system()
{

//  INTITIALIZATION STUFF
/*SLEEP - Sleep Controller */
SLEEP_CTRL = 0x00;  // sleep disabled

/* CLK - Clock System */
//CLK_PSCTRL = 0x00;    //  No prescalers
OSC_PLLCTRL = 0x10;  // PLL mult. factor ( 2MHz x8 ) and set clk source to PLL.
OSC_CTRL = 0x10;     // Enable PLL
while( !( OSC_STATUS & 0x10 ) );     // Is PLL clk rdy to go ?
CCP = 0xD8; //Unlock seq. to access CLK_CTRL
CLK_CTRL = 0x04;


return 1;
}
```

```
int USART_INIT()
{
 PORTC.OUT = 0x08;
 PORTC.DIR = 0x19;
 USARTC0.CTRLA = 0x28;    //  Set to medium interrupt after a send or receive
 USARTC0.CTRLC = 0x03; //Asynchronous, no parity, 1 stop bit, 8 bit
 USARTC0.BAUDCTRLA = 0xCF; //19200baud
 USARTC0.BAUDCTRLB = 0x00;
 USARTC0.CTRLB = 0x18; // Turn on the transmitter, turn off the receiver
 return 1;
}
```

```
int16_t BCD_to_revolutions()
{


rev_value = (int16_t)(message[4]-0x30) + 10*((int16_t)(message[3]-0x30)) + 100*((int16_t)(message[2]-0x
 return(rev_value);
}

int revolutions_to_BCD()
{
 int16_t rev_to_send = ROTATIONS;
 //int16_t rev_to_send = rev_value;
 rev_out[0] = (char)(rev_to_send%10L);
 rev_to_send = (((rev_to_send - (int16_t)rev_out[0])/10L));
 rev_out[1] = (char)(rev_to_send%10L);
 rev_to_send = (((rev_to_send - (int16_t)rev_out[1])/10L));
```

```
 rev_out[2] = (char)(rev_to_send%10L);
 rev_to_send = (((rev_to_send - (int16_t)rev_out[2])/10L));
 rev_out[3] = (char)(rev_to_send%10L);
 rev_to_send = (((rev_to_send - (int16_t)rev_out[3])/10L));
 rev_out[4] = (char)(rev_to_send);
 USARTC0.DATA = '%';
 for (int8_t b = 5; b>0; b--)
  {

   while(transmit_wait == true);
   USARTC0.DATA = (rev_out[b-1]+0x30);
   transmit_wait= true;

  }
 return(0);
}

int initialize_position()
{ COMMAND_ROTATIONS = 0;
 DIRECTION = DOWN;
 int i = 0;
 while(PORTF.IN != HALL_COMB[i])
  {
   i = i+1;
  }
 AWEXE.OUTOVEN = COMMUTATION[i];
 if((PORTB.IN & 0x03) == 0x03)
  {
   PWM = 0x01A0;
   TCE0.CCA = PWM;
   TCE0.CCB = PWM;
   TCE0.CCC = PWM;
   TCE0.CCD = PWM;
  }
 else
  {
   reed_initialized = true;
  }
 return(0);
}

int stop()
{ COMMAND_ROTATIONS = ROTATIONS;
 set_PWM(0x0000);
 AWEXE.OUTOVEN = 0x00;
 return(0);
}


int control_loop()
{
 error = COMMAND_ROTATIONS - ROTATIONS;
 if (error > 0)
  { DIRECTION = UP;
   set_PWM(0x01A0);
  }
 if (error < 0)
  {
   DIRECTION = DOWN;
   PORTH.OUT = 0x00;
   set_PWM(0x01A0);
  }
 else if (error == 0)
  {
   set_PWM(0x0000);
  }
return(0);
}


int read_message()
{
```

```
if (new_message == true)
 { if (new_revolutions == false)
   {
    switch(command)
     {
      case 'I': initialize_position();break;
      case 'R': set_PWM(0x01A0); break;
      case 'S': stop();break;
      case 'U':
          if (COMMAND_ROTATIONS >= 16)
           {
            COMMAND_ROTATIONS = COMMAND_ROTATIONS-16;
           }
          break;
      case 'D':
          if (COMMAND_ROTATIONS != MAX_HEIGHT)
           {
            COMMAND_ROTATIONS = COMMAND_ROTATIONS+16;
           }
          break;


      case 'E':  TCC0.PERL = 0x1F; // 1ms
         TCC0.PERH = 0x00;
         break;

      case 'F':  TCC0.PERL = 0x3E; // 2ms
         TCC0.PERH = 0x00;
         break;


      case 'G':  TCC0.PERL = 0x5D; // 3ms
         TCC0.PERH = 0x00;
         break;


      case 'H':  TCC0.PERL = 0x7D; // 4ms
         TCC0.PERH = 0x00;
         break;


      case 'J':  TCC0.PERL = 0x9C; // 5ms
         TCC0.PERH = 0x00;
         break;


      case 'K':  TCC0.PERL = 0xBB; // 6ms
         TCC0.PERH = 0x00;
         break;


      case 'L':  TCC0.PERL = 0xDA; // 7ms
         TCC0.PERH = 0x00;
         break;


      case 'M':  TCC0.PERL = 0xFA; // 8ms
         TCC0.PERH = 0x00;
         break;

      case 'N':  TCC0.PERL = 0x19; // 9ms
         TCC0.PERH = 0x01;
         break;

      case 'O':  TCC0.PERL = 0x38; // 10ms
         TCC0.PERH = 0x01;
         break;

      case 'T':  PORTJ.OUT = 0x02;  //  Begin Trigger Sequence
         TCC0.INTCTRLA = 0x00;
         TCC0.CTRLA = 0x00;
         TCC0.INTFLAGS = 0x01;
```

```
            TCC0.CNT = 0x0000;
            TCC0.CNT = 0x0000;
            TCC0.CTRLA = 0x07;
            TCC0.INTCTRLA = 0x02;
            break;

       }
      new_message = false;
     }
    else
     {
     COMMAND_ROTATIONS = BCD_to_revolutions();
     if (COMMAND_ROTATIONS < 0)
      {
       COMMAND_ROTATIONS = 0;
      }
     else if (COMMAND_ROTATIONS > MAX_HEIGHT)
      {
       COMMAND_ROTATIONS = MAX_HEIGHT;
      }
     new_message = false;
     new_revolutions = false;
    }
  }
return(0);
}


int commute_motor()
{


 if (PORTF.IN == HALL_COMB[0])
  {

   if (DIRECTION == UP)
    {
     AWEXE.OUTOVEN = COMMUTATION[0];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[3];
    }
  }
 else if (PORTF.IN == HALL_COMB[1])
  {

   if (DIRECTION == UP)
    {
     AWEXE.OUTOVEN = COMMUTATION[1];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[4];
    }
  }
 else if (PORTF.IN == HALL_COMB[2])
  {
   //PORTH.OUT = ~HALL_COMB[2];
   if (DIRECTION == UP)
    {
     AWEXE.OUTOVEN = COMMUTATION[2];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[5];
    }
  }
 else if (PORTF.IN == HALL_COMB[3])
  {

   if (DIRECTION == UP)
```

```
    {
     AWEXE.OUTOVEN = COMMUTATION[3];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[0];
    }
  }
 else if (PORTF.IN == HALL_COMB[4])
  {

   if (DIRECTION == UP)
    {
     AWEXE.OUTOVEN = COMMUTATION[4];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[1];
    }
  }
 else if (PORTF.IN == HALL_COMB[5])
  {

   if (DIRECTION == UP)
    {
     AWEXE.OUTOVEN = COMMUTATION[5];
    }
   else
    {
     AWEXE.OUTOVEN = COMMUTATION[2];
    }
  }
return(0);

}

int set_PWM(uint16_t value)
{
 if (value > 0x01A0)
  {
   PWM = 0x01A0;
  }
 else
  {
   PWM = value;

  }
 TCE0.CCA = PWM;
 TCE0.CCB = PWM;
 TCE0.CCC = PWM;
 TCE0.CCD = PWM;
 return(0);
}

void sense_position()
{

 hall_check = PORTF.IN;    // Store the current hall combination
 // determine the direction that the motor moved in.
 //  Search for what the hall combination is from the known combinations
 if (hall_check != HALL_COMB_PREV)
  { hall_index = 0;
   // Find the current hall combination's index
   while (hall_check != HALL_COMB[hall_index])
    { hall_index++;
    }
   int prev_index = 0;
   while (HALL_COMB_PREV != HALL_COMB[prev_index])
   {
    prev_index++;
   }
   if (hall_index > prev_index)
```

```
    {
     if (hall_index == 5 && prev_index == 0)
      {
       SENSED_DIRECTION = UP;
      }
     else
       SENSED_DIRECTION = DOWN;
    }
   else if (hall_index < prev_index)
    {
     if (hall_index == 0 && prev_index == 5)
      {
       SENSED_DIRECTION = DOWN;
      }
     else
       SENSED_DIRECTION = UP;
    }

   //  Update position
   if (SENSED_DIRECTION == UP)
    {
     if (HALL_COUNT == 11)       // complete rotation condition/overflow condition
      {
       HALL_COUNT = 0;
       ROTATIONS = ROTATIONS + 1;
      }
     else         // 30 degree rotation downwards
      {
       HALL_COUNT = HALL_COUNT+1;
      }
    }
   else if (SENSED_DIRECTION == DOWN)
    {
     if (HALL_COUNT == 0)     // back to previous rotation condition/underflow condition
      {
       HALL_COUNT = 11;
       ROTATIONS = ROTATIONS - 1;
      }
     else
      {
       HALL_COUNT = HALL_COUNT - 1;// 30 degree rotation upwards
      }
    }

  }

  HALL_COMB_PREV = hall_check;

}
```

Appendix C

Boost Converter Calculations

This appendix shows the parameters to design a boost converter with a $200V$ output, a 19Ω load from the hydrophone, and an input voltage of $24V$ from an automotive battery. Equations are given by Hart [12, p. 211-215]. The duty cycle of the switch is computed by equation C.1, where V_{out} is the output voltage, V_s is the voltage supplied at the input, and D is the duty cycle of the switching.

$$D = 1 - \frac{V_s}{V_{out}} = 1 - \frac{24}{200} = 0.88 \tag{C.1}$$

The average current in the inductor for this duty cycle is given in equation C.2.

$$I_L = \frac{V_s}{(1-D)^2 R} = \frac{24}{(1-.88)^2 19} = 87.7A \tag{C.2}$$

A search at vendor websites found that any inductor that was reasonable for PCB mounting and that had current ratings high enough had a maximum inductance of about L = $2.2\mu H$. With this inductance, the minimum switching frequency can be calculated with equation C.3

$$f_{min} = \frac{D(1-D)^2 R}{2L} = \frac{0.88(1-0.88)^2 19\Omega}{2(2.2\mu H)} = 54.720kHz \tag{C.3}$$

Appendix D

Switched Capacitor Equivalent Resistance

The switch between a capacitor C and two voltage nodes, $V1$ and $V2$, can be expressed as an equivalent resistance. The charge in a capacitor is given in:

$$q = CV \tag{D.1}$$

As the capacitor's voltage is switched between values V_1 and V_2, the charge changes as:

$$\Delta q = C(V_1 - V_2) \tag{D.2}$$

If the charge changes periodically at a period $T = 1/f$, the change in charge over time is:

$$\frac{\Delta q}{T} = \frac{C(V_1 - V_2)}{T} \tag{D.3}$$

Change in charge over time is equivalent to current, so that $I = \frac{\Delta q}{T}$. Because current and change in voltage are known, the equivalent resistance is found as:

$$R = \frac{V_1 - V_2}{I} = \frac{1}{fC} \tag{D.4}$$

Appendix E

Capacitance Calculation for Constant Current Discharge

The appropriate capacitor size can be found when a constant current is drawn from a capacitor bank. First, note the voltage of a capacitor is given in equation E.1, and the current of a capacitor is given in equation E.2.

$$V_c(t) = V_o e^{\frac{-t}{RC}} \tag{E.1}$$

$$I_c(t) = \frac{V_o}{R} e^{\frac{-t}{RC}} \tag{E.2}$$

The derivative of the voltage gives equation E.3, and by observing the current term of equation E.2 present in the derivative, it can be set equal to a constant current, I, since current discharge is constant over time. This is shown in equation E.4.

$$\frac{dV_c(t)}{dt} = -\frac{V_o}{RC} e^{\frac{-t}{RC}} \tag{E.3}$$

$$\frac{dV_c}{dt} = -\frac{I}{C} \tag{E.4}$$

So, with a linear voltage change over time, the appropriate capacitance can be solved with design constraints that the capacitor changes from V_1 to V_2 over a time t_1 to t_2 in equations E.5.

$$C = -\frac{I(t_2 - t_1)}{V_2 - V_1} \tag{E.5}$$

Appendix F

PSPICE Simulation Circuits

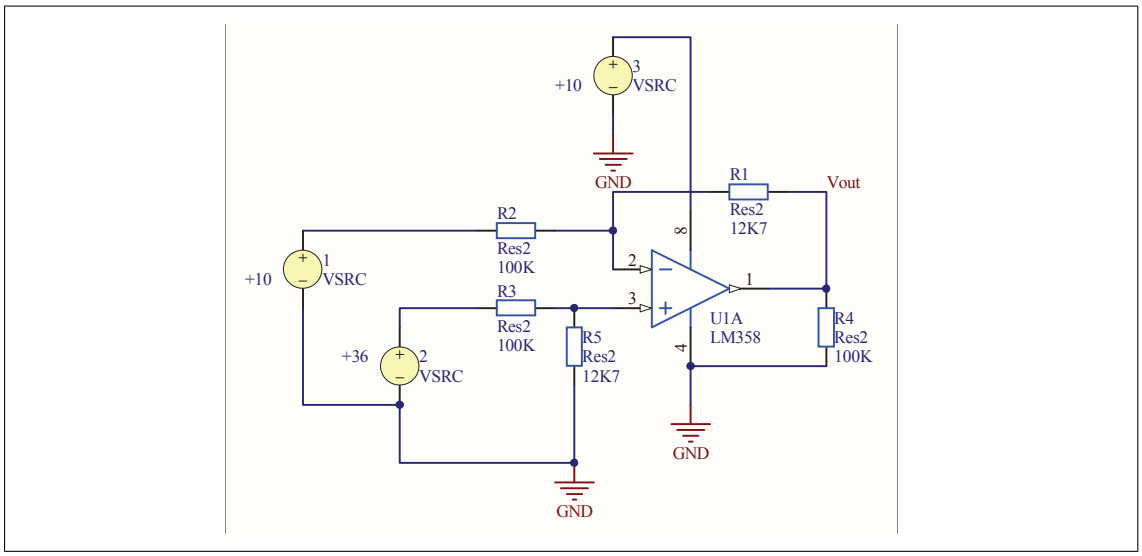


Figure F.1: PSPICE circuit used to simulate the voltage scaler for sensing the motor's battery voltage. V2 represents the motor battery voltage.

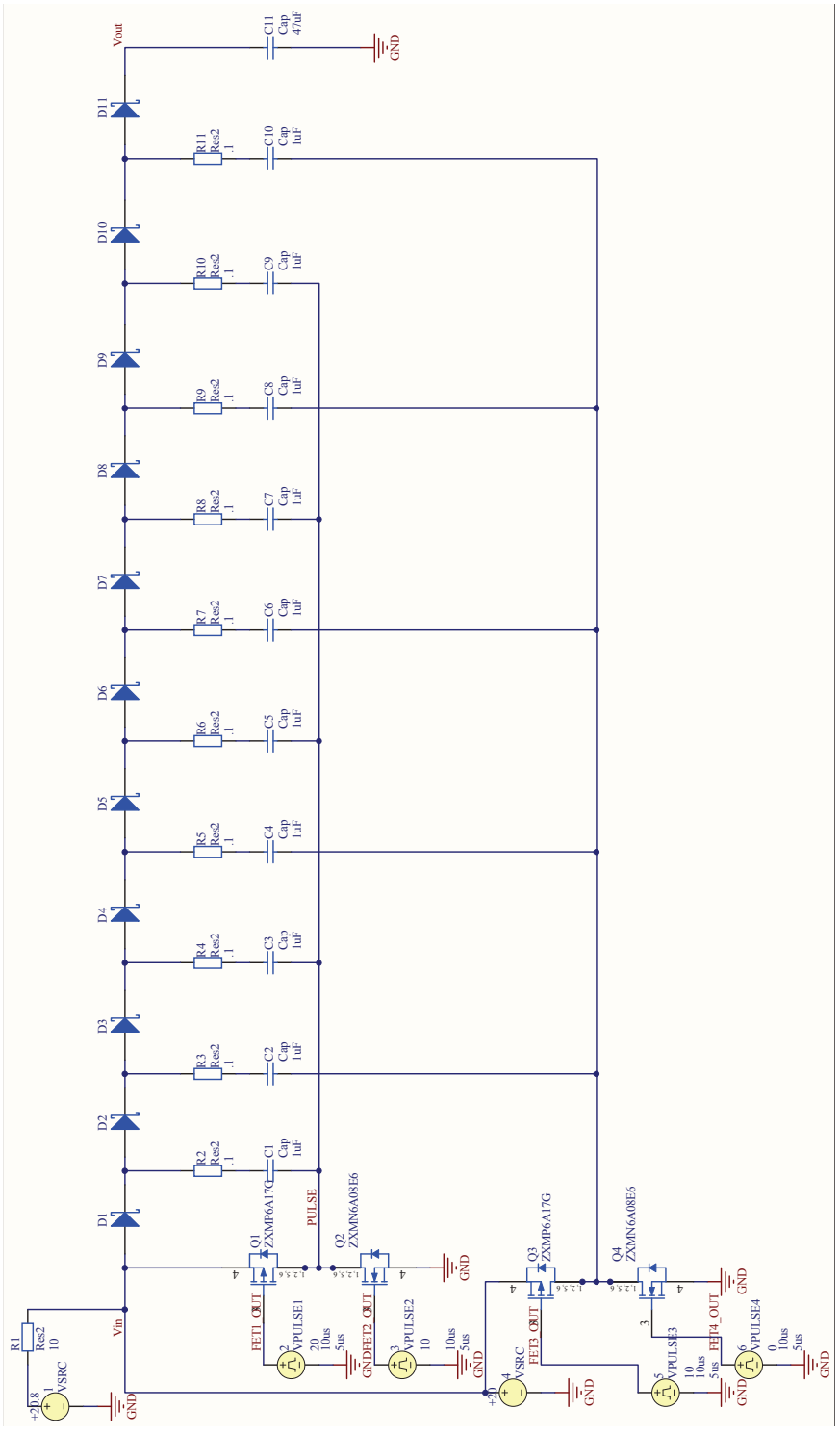


Figure F.2: PSPICE circuit used for charge-pump simulations.

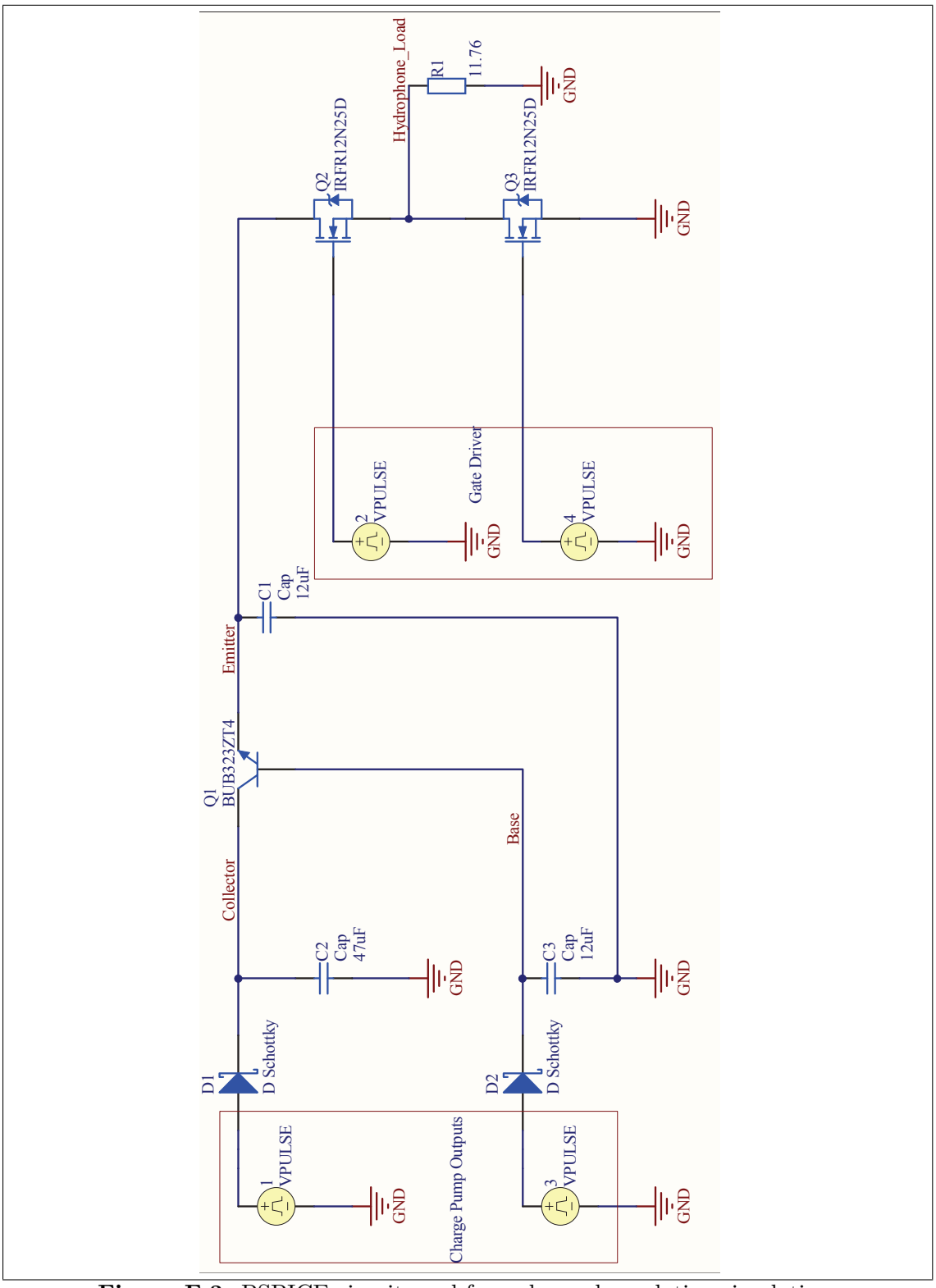


Figure F.3: PSPICE circuit used for pulse and regulation simulation.

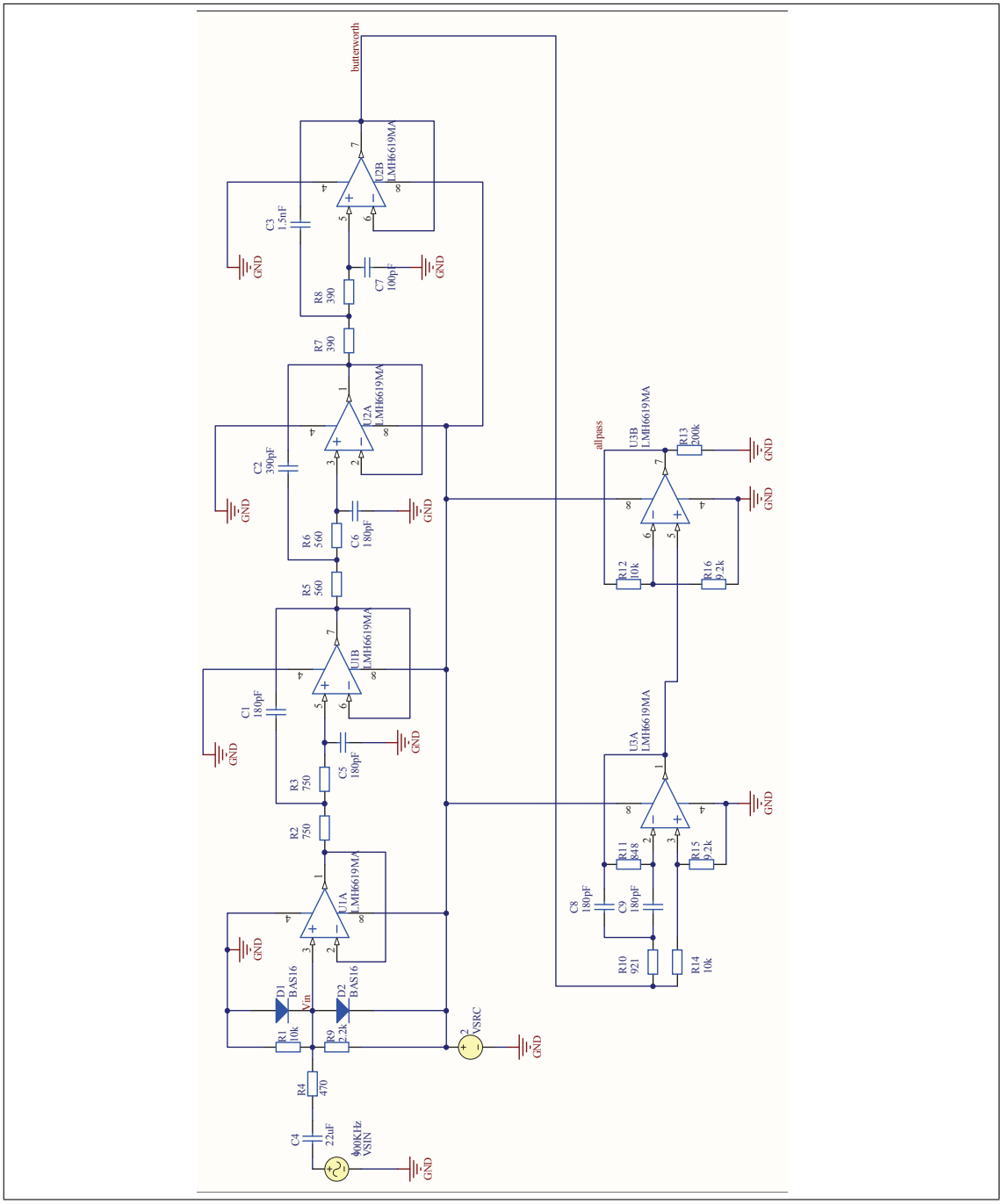


Figure F.4: PSPICE circuit used to simulate butterworth filter and phase equalizer.

List of Symbols and Abbreviations

Abbreviation	Description
BLDC	Brushless DC motor
PWM	Pulse-Width-Modulation
RPM	Revolutions-per-Minute
EMF	Electro-Motive-Force
ADC	Analog-to-Digital Converter
RMS	Root-Mean-Square
DC-DC	Conversion of one DC voltage level to another DC voltage level
SMPS	Switch-Mode-Power-Supply
IC	Integrated Circuit
ESR	Equivalent-Series-Resistance
MOSFET	MetalOxideSemiconductor Field-Effect Transistor
BJT	Bipoloar-Junction Transistor
IGBT	Insulated-Gate Bipolar Transistor
LDO	Low Drop-Out regulator
SPICE	Simulation Program with Integrated Circuit Emphasis
USB	Universal Serial Bus
SPI	Serial Peripheral Interface
DFT	Discrete Fourier Transform
IDFT	Inverse Discrete Fourier Transform
FFT	Fast Fourier transform
IFFT	Inverse Fast Fourier transform

List of Figures

List of Tables

Index