Comprehensive Ultrasound Research Platform

Project Proposal

Emma Muir

Sam Muir

Jacob Sandlund

David Smith

Advisor: Dr. José Sánchez

Date: December 9, 2010

Project Summary

The goal of this project is to create a prototype ultrasound research platform that will test theoretical developments on methods that use coded pulses to excite ultrasound transducers. The prototype platform shall support transmission of eight arbitrary waveforms with configurable delays between channels to excite a transducer. After transmission, return signals from the transducer shall be processed and captured by an analog front end device. A PC will then filter and process the data using pulse compression and delay and sum beamforming. The resulting image will be displayed for analysis.

Goals

The main goal of our project is to create a functional, user friendly ultrasound research platform. This can be divided into several goals.

- Create a system for a PC to encode waveforms with sigma-delta encoding and transmit them over Ethernet to be stored on a field programmable gate array (FPGA).
- Design FPGA hardware to store and transmit eight arbitrary waveforms with configurable delays.
- Design and implement amplification circuitry for high frequency waveforms.
- Configure an analog front end device and embedded device to process, capture, and send received data from the transducer.
- Write C code to filter and process the received data and display the results.

System Block Diagram

First, the excitation waveforms must be initially transformed to sigma-delta modulation form to synthesize a digital-to-analog converter. As shown in Figure 1, the sigma-delta waveform will be stored in a memory device that is connected to an FPGA. The FPGA will be used to excite the transducer with the arbitrary waveform. Analog circuitry will be required to amplify and filter the digital output from the FPGA pins. After exciting the transducer with a voltage waveform, a pressure wave will propagate through the medium being imaged. The echo received by the transducer will be processed by the analog front end and stored on an embedded device. Finally, the stored data will be sent to a PC, processed, and displayed.

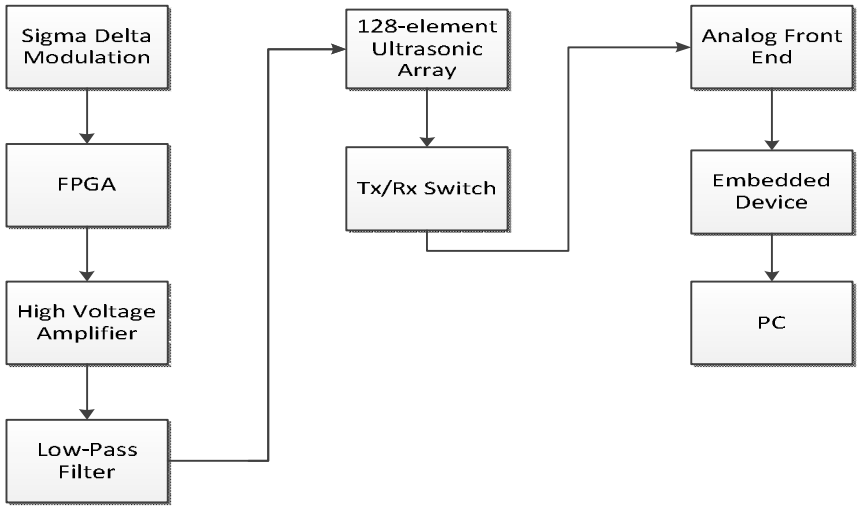

Figure 1: System Block Diagram

Functional Description

Sigma Delta Modulation

Sigma delta modulation is an analog to digital conversion technique that includes quantization error compensation. This technique will be used to store, in digital form, arbitrary waveforms that will be sent to the ultrasound transducer. A software-based one-bit analog to digital converter is initially used to round the input to either one or negative one. This value is then subtracted from the initial input creating an error value. The error is subtracted from the next input value, providing conversion error compensation. The adjusted input is again pushed through a software-based analog to digital converter to round to one or negative one. The overall output expresses an input of analog values as a series of ones and negative ones. This process can be seen in Figure 2 [1].

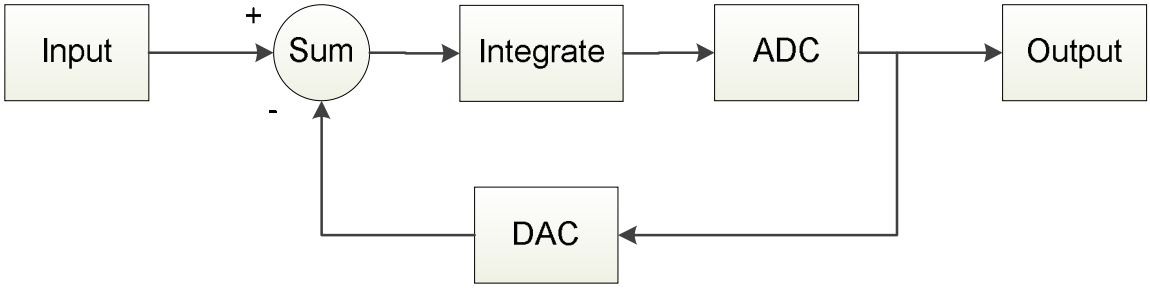

Figure 2: Sigma Delta Modulation Subsystem Block Diagram[1]

FPGA

The parallel task of transmitting waveforms to multiple ultrasound transducer pins is well suited to an FPGA. The FPGA excitation block encompasses three basic steps. A high level flowchart in figure 3 below shows the process. First, the input waveform data from the PC is transmitted and stored to the DDR2 memory on the FPGA development kit. This step allows the user to store many multiple waveforms on the FPGA, and allows for outputs to be delayed relative to each other. Next, the waveform data is retrieved from the DDR2 memory and parallelized. As shown on the diagram, this is only done after a signal commands to start the transmission. Once the data has been retrieved from the memory, it is transmitted to the FPGA pins at a high speed.

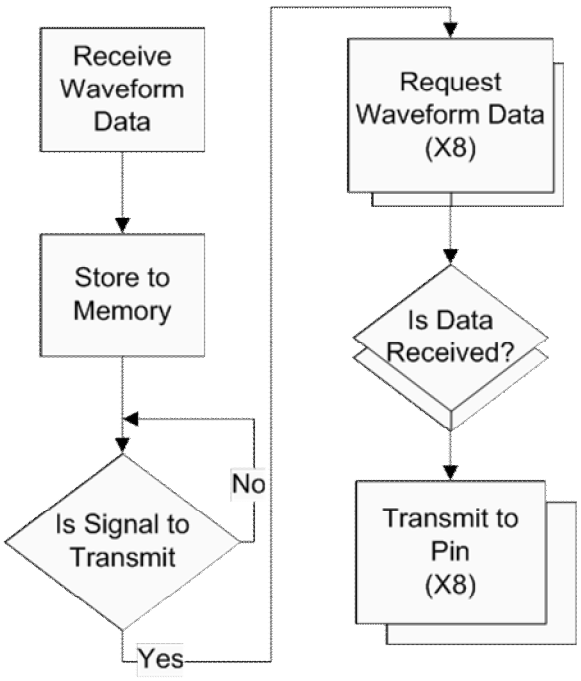

Figure 3: High Level FPGA Flowchart

High Voltage Amplifier

The purpose of this subsystem is to amplify the output from the FPGA, which is approximately 1V peak-to-peak, to approximately 100V peak-to-peak. This increase in voltage will allow the transducer to produce more pressure since the pressure produced is proportional to the input voltage. For amplification a Zetex ZXMHC10A07N8 or similar device will be used.

Low-Pass Filter

The low-pass filter is used to convert the sigma delta signal into its analog representation, which is then sent to the 128-element ultrasonic array. A simple RC low-pass circuit will be used to apply this filtering. This subsystem may not be necessary as the bandpass nature of the source could potentially be used as the filtering mechanism. Additional research will determine if this is the case.

128-element Ultrasonic Array

The ultrasonic transducer receives a high voltage input signal that is converted into a pressure waveform that will propagate throughout the medium. As the pressure waveform encounters objects, reflections that contain a fraction of the transmitted energy will be received by the transducer. A transmit/receive (T/R) switch will clamp high voltages to voltages that are sensible to the analog front end. The device currently specified is the TX810 by Texas Instruments.

Analog Front End
The amplitude from the received signal will be small for far away objects. Consequently, amplification of the echoes is required. The noise amplification must be limited to produce a clear received signal. A low noise amplifier (LNA) will amplify the signal. The analog to digital (A/D) converter will sample the analog signal to be recorded. The signal will be quantized and encoded into digital data by the device. The device we will be using is the AD9276-80KITZ by Analog Devices.

Receiving End Embedded Device
The receiving end embedded device will capture and store the data from the A/D converter. This data will then be sent to a PC over a USB connection so that the ultrasound data can be processed.

PC
The steps performed on the PC are shown in Figure 4. This block receives the raw data from the FPGA as an input. It also receives settings from the GUI for the data processing. First, pulse compression is performed on the signal. To achieve this, a Wiener filter is used. This filter compares the received signal to the excitation signal. This coded excitation and filtering technique is used to reduce noise in the signal and increase the resolution of the image [2]. Figure 5 displays the software flowchart for this process. Next, delay and sum beam-forming is performed to focus the different signals received from each channel of the ultrasonic transducer. Figure 6 displays the software flowchart for delay and sum beamforming. Time-gain compensation is then performed to compensate for attenuation of the sound waves due to the depth of the echoing substance. Next, envelope detection and log compression are performed to prepare the data to be displayed. This data is presented to the user through a graphical user interface.

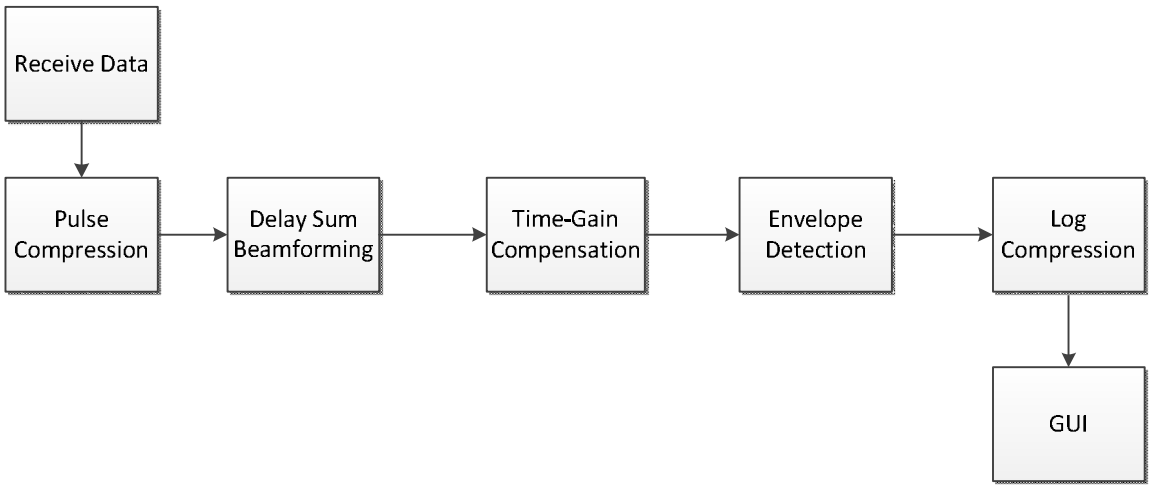

Figure 4: Processing Data Subsystem Block Diagram

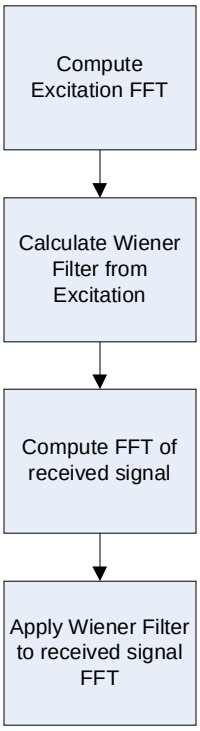

Figure 5: Software Flowchart for Pulse Compression

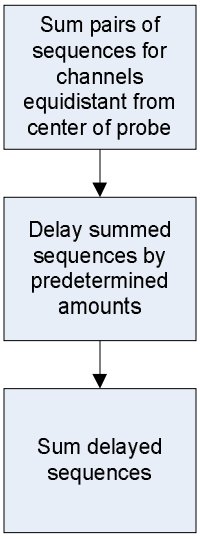

Figure 6: Software Flowchart for Delay and Sum Beamforming

Graphical User Interface

The graphical user interface allows the user to manipulate the displayed data. The inputs include dynamic range and depth settings. Depth allows the user to view all data from a start range to a stop range. Although the data is still available for all depths, this option allows the user to zoom in on areas that require a closer observation. Dynamic range sets the maximum compactness to be observed. All objects with compactness greater than the maximum appear black, while objects with less compactness range from gray to white based on the degree of their compactness. This allows the user to view objects of both high and low compactness clearly. The signal processing block accepts these inputs of depth and dynamic range and presents the data in the form requested as the user changes the data display settings.

Functional Requirements

System Requirements

- The research platform will transmit and receive signals on up to eight channels on the transducer.
- Excitation waveforms shall be 3 μs or less in duration to maintain a time-bandwidth product of 40.
- Circuitry will be designed to achieve a signal to noise ratio greater than 50 dB.
- Resolution enhancement will create excitation signals capable of enhancing the bandwidth of the transducer to 12.45 MHz (150% of the center frequency of 8.3 MHz).

Transmitting FPGA

- The FPGA shall connect to the PC via Ethernet. The connection must be able to transmit 24 kbits of data to the FPGA in roughly 10 seconds (2.4 kbits/s) or less.
- The waveform data after being retrieved from the DDR2 shall be parallelized to prepare sending out on the FPGA pins. After parallelization, each pin shall change at a rate of 1 GHz.
- The waveform data to transmit must be stored in DDR2 memory when it is brought in from PC. Then it must be retrieved when the transmission occurs. The memory speed to support the 8-pin requirement is 62.5 MHz.
- The FPGA shall store independent sigma-delta encoded excitation waveforms of a maximum of 3072 samples for each of the eight channels.
- Each channel shall have a configurable delay of up to 5 μs relative to the other channels prior to transmission.
- The transmitting FPGA shall output a signal to the receiving FPGA when all excitations have been transmitted to start data collection.

Signal Conditioning

- Waveforms from the transmitting FPGA shall be amplified from ±1 V (or 0 to 3.3 V) to ±10 V with a high speed op-amp. Slew rate for this op-amp shall be adequate to avoid distortion of the waveform.
- A second stage amplifier shall amplify the signals to ±100 V to increase energy transmitted by the probe.
- A low pass filter shall decode the sigma-delta encoded waveforms into a chirp signal. The chirp shall match the desired waveform to within 10% mean squared error.

Receiving FPGA

- To simulate a pulse repetition frequency of 1 KHz, each channel will be recorded for 997 μs after all excitation waveforms have been transmitted for a maximum depth of 76 cm.
- Received signals shall be recorded using an A/D converter at a rate of 65 MHz and a resolution of 14-bits.

- The recorded data will be stored in DDR2 memory. The DDR2 memory speed shall be at least 65 MHz.
- Total data for 8 channels will be approximately 8 Megabytes.
- Data shall be transmitted to the PC in less than 10 seconds after full data collection using Ethernet (bit rate greater than 5.8 Mbit/s).

Data Processing

- All data processing shall be performed in less than 2 minutes.
- The image created will display an image for depths between 0.25 cm and 30 cm.

Preliminary Results

Resolution Enhancement Compression (REC)

REC is a technique to improve the bandwidth of an ultrasound transducer by creating a pre-enhanced chirp signal. Using convolution equivalence, a chirp can be constructed which, when passed through the actual transducer, will yield the same result as a linear chirp passed through a desired transducer with greater bandwidth than the actual transducer [2].

So far, promising results have been achieved in simulating a pre-enhanced chirp generated with REC. The first step was to create a model for the actual transducer's frequency response, based upon the average frequency response for the 128-element transducer to be used.

The impulse response of the transducer was constructed by windowing a sinusoid in the sequence domain with a Hann window to produce a frequency response with a center frequency of 8.4 MHz and fractional bandwidth of 101.9%, which correspond to those of the transducer elements. This impulse response is shown in Figure 7a. Next, a desired transducer impulse response was created using the same method, but with a smaller window to produce a fractional bandwidth of 150%. The windowed sinusoid was shifted to create a symmetric impulse response, as in Figure 7c. A symmetric impulse response reduces the low frequency content of the desired frequency response.

Next, a linear chirp was created with frequencies in the range of 1.14 times the desired bandwidth, as shown in Figure 7d. Extending the bandwidth of the chirp by a factor of 1.14 minimizes the ripples in the frequency response of the chirp passed through the transducer. By minimizing these ripples, side-lobes will be reduced after pulse compression. Convolution equivalence was then used to find a chirp which, when convolved with the actual transducer frequency response model, would yield the same output as the linear chirp convolved with the desired frequency response. The calculated chirp is shown in Figure 7b, and the equivalent output is shown in Figure 8. This simulation provides a pre-enhanced chirp to be tested on the finished system.

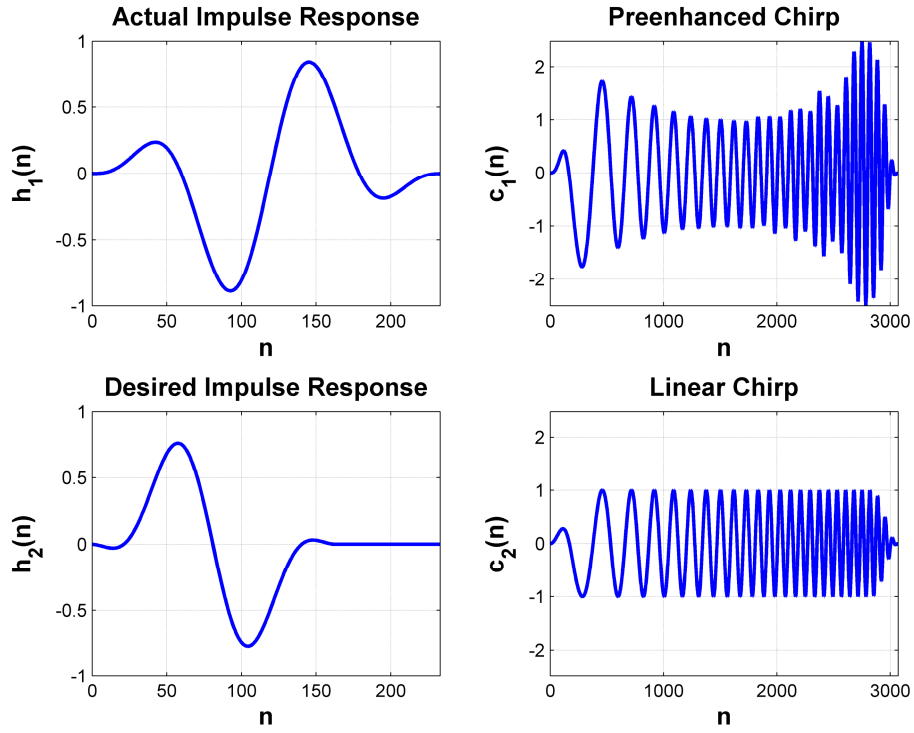

Figure 7: REC Chirp Calculation

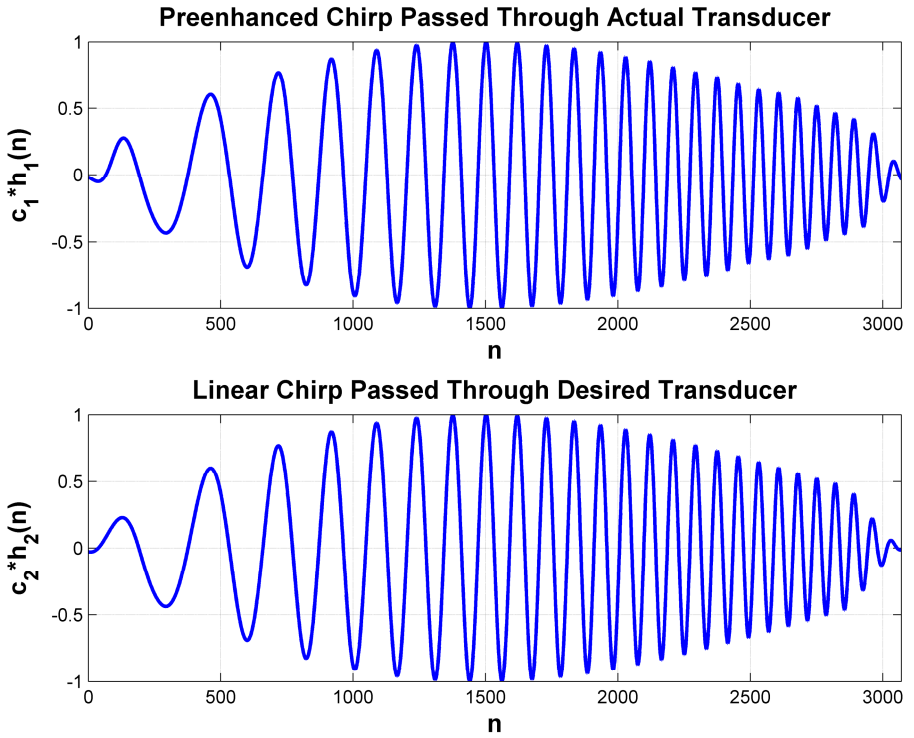

Figure 8: REC Simulation Results

Sigma Delta Modulation

As shown in Figure 2, sigma delta modulation uses oversampling and error compensation to convert an analog signal into a binary sequence (+1, -1) at a significantly higher data rate. The higher the order of the modulation, the more accurate the signal compares to the original analog signal, but a higher order modulation has a higher risk of saturating to a fixed string of only 1's or only -1's. Based on these factors, preliminary testing has shown that a second order sigma delta modulation will suffice.

The frequency of the modulation will be based on the specifications of the FPGA. Higher data rates produce higher accuracy; consequently, the data rate will be dictated by the limits of the FPGA. In addition, the data rate must be a power of two as specified by MATLAB (Mathworks, Natick, MA) sigma delta toolbox [3]. The culmination of all these contraints resulted in a data rate of 1.024 GSamples/sec. Figure 9 shows the desired analog chirp signal and its corresponding second order sigma delta sequence. Note that the chirp modulates the frequency linearly from 4MHz to 12MHz as this is the simulated bandwidth of the transducer.

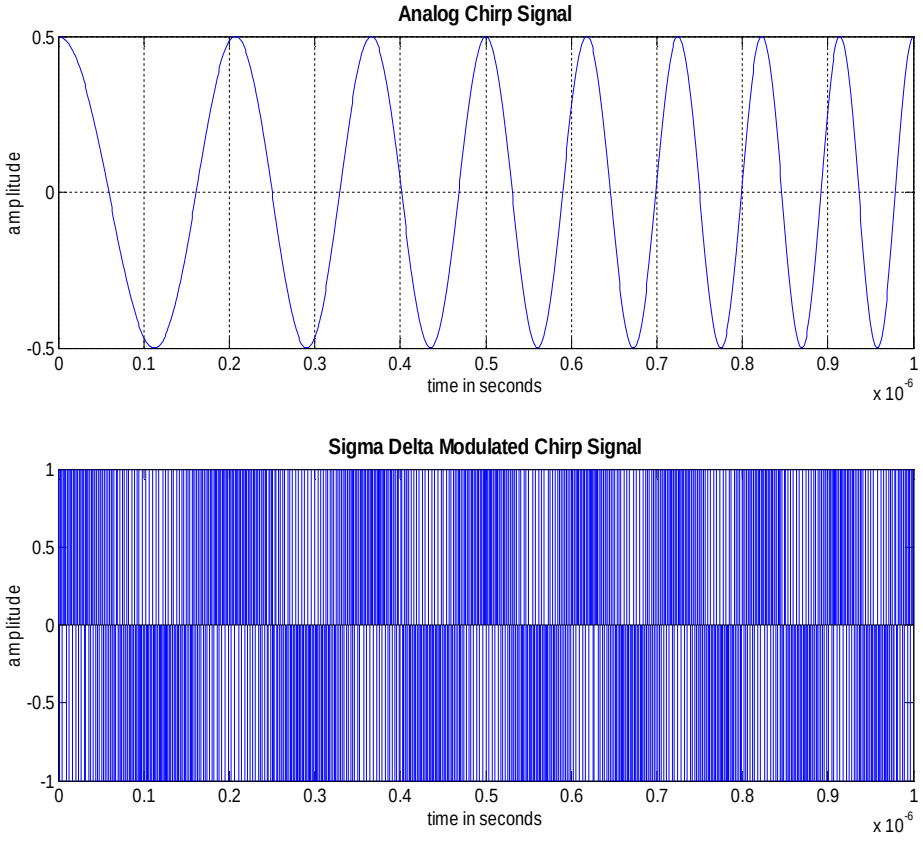

Figure 9: Sigma Delta Modulation for a Chirp Signal

To test the validity of the technique, the sigma delta sequence was reconstructed using a Parks-McClellan equiripple finite impulse response (FIR) filter. Using this method, the signal was reconstructed; however, with a linear decay as the frequency increased. Figure 10 shows the analog signal over four periods and the resulting sigma delta modulated signal after reconstruction with this filter. The resulting sigma delta modulated signal has a correlation of 0.9888 to the analog signal.

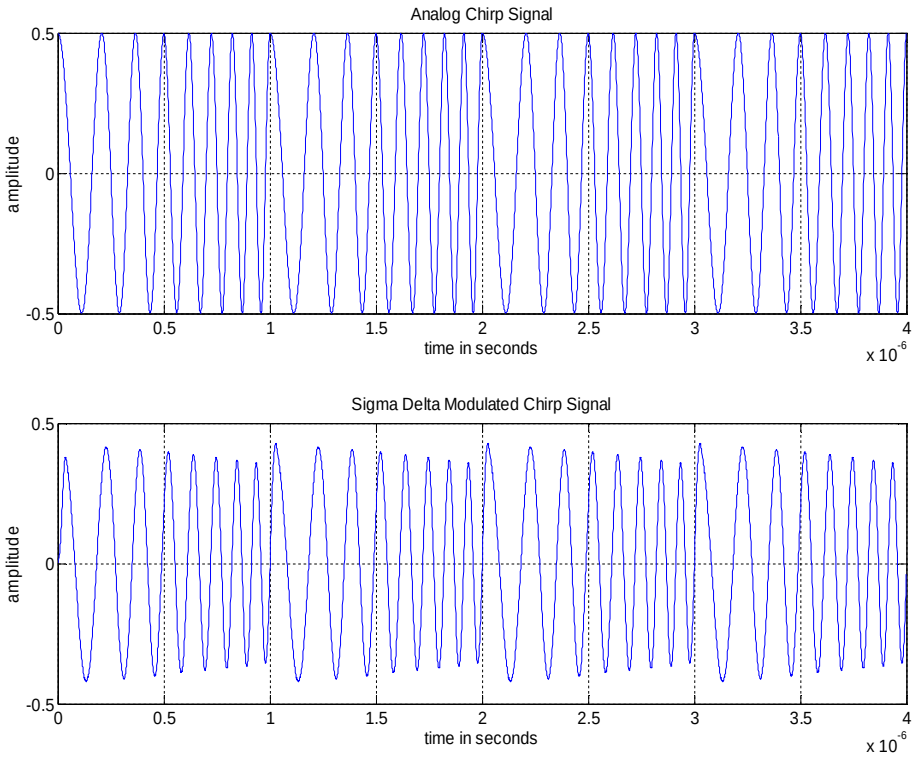

Figure 10: Reconstructed Sigma Delta Modulated Signal

Due to the linear nature of the decay, compensation with a proportional gain was performed by predistorting the linear chirp. The resulting reconstruction after taking into account the predistortion of the chirp is shown in Figure 11. The resulting sigma delta modulated signal has a correlation of 0.9900 to the analog signal which is greater than the 0.9888 correlation of the previous sigma delta modulated signal.

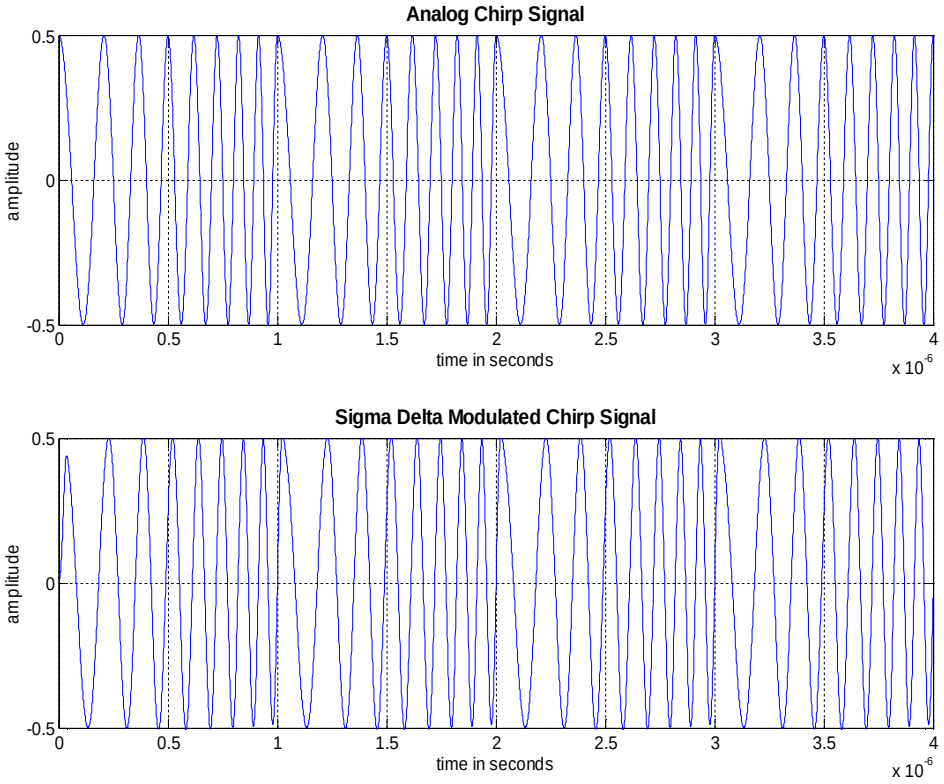

Figure 11: Reconstructed Modulated Sigma Delta Signal with Gain

To assess the performance that would be expected in a laboratory setting, the signal was reconstructed using a first order RC low pass filter. Using a 1 KΩ resistor and a 10 pF capacitor, a cut off frequency of 100 MHz was created to reconstruct the signal shown in Figure. This design shall be tested to determine if the accuracy shown is enough to meet specifications. The resulting sigma delta modulated signal has a correlation of 0.9129 to the analog signal, which is less than the 0.9888 and 0.9900 correlations of the previous two sigma delta modulated signal. This signal also has a mean squared error of 0.0264 which is less than the 0.10 requirement.

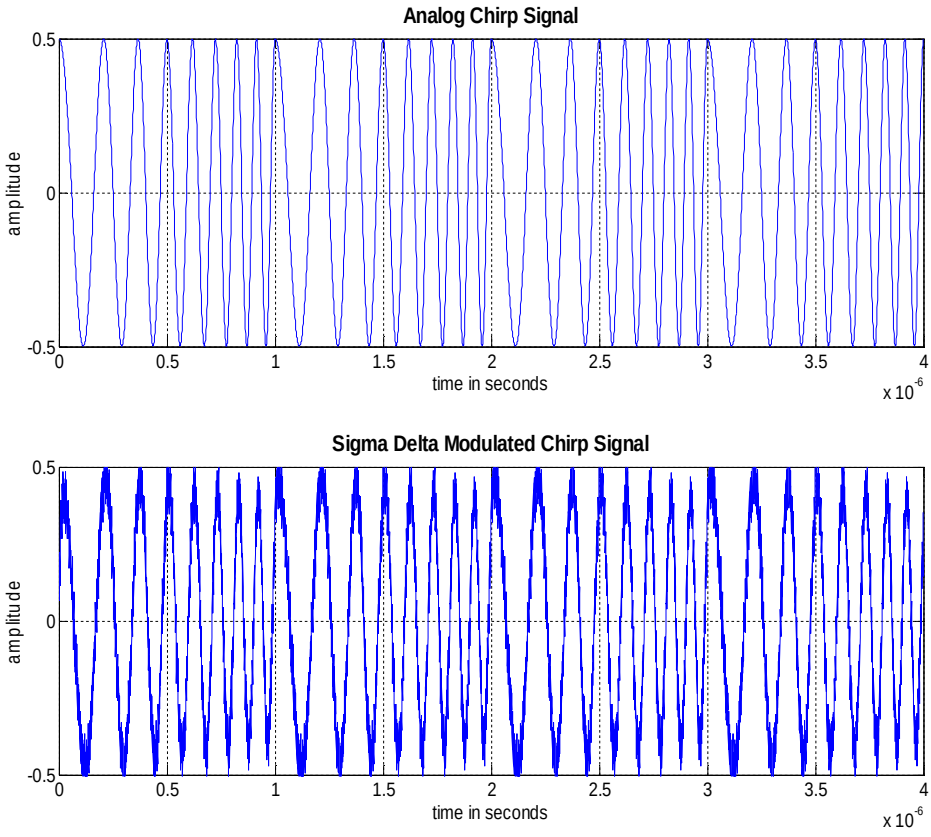

Figure 12: RC LPF Reconstructed Modulated Sigma Delta Signal

FPGA

The functional description for the FPGA sub-system described three operations the FPGA must accomplish. These were the storing of waveform data on the DDR2 memory, the parallelizing of the data to the pins, and the high speed transmission of that data. In the preliminary work described below, each of these steps have been partially implemented.

DDR2 Interface

For the storing of data on the DDR2 memory, the Memory Interface Generator (MIG) version 3.1, a Xilinx Core Generator package, was used to do the physical interfacing. An interface to the MIG block was constructed to simplify the communication of the memory to other components. This interface was tested and verified up to the maximum rate of the DDR2 memory of 266 MHz.

Arbitration System

To integrate the DDR2 interface with the other FPGA functionality, a system was constructed to arbitrate access to the memory. The importance of this system is to allow multiple components to connect to the memory. The waveform data, which must be first stored onto the memory, must be accessible from multiple blocks that will output the data to the pin. Unfortunately, only one block, also called a socket, can retrieve data from the memory at a time.

In constructing an arbitration system, special consideration was needed to ensure that more important sockets had priority to access the memory. Also, a scheme was developed that reduces the possibility for starvation by cycling access in a round robin fashion. The arbitration system follows a process outlined in figure 14 below. Multiple sockets may request access to the memory at a time. Only after the block that currently has access completes its list of reads and writes are the requesting sockets considered for access. Next, the requesting sockets are compared to determine the one that should receive access next. This is done in two steps. First, all requesting sockets of a lower priority are eliminated. Second, the history of when the remaining sockets last had access is compared. Access is then given to the socket that had its previous access furthest in the past.

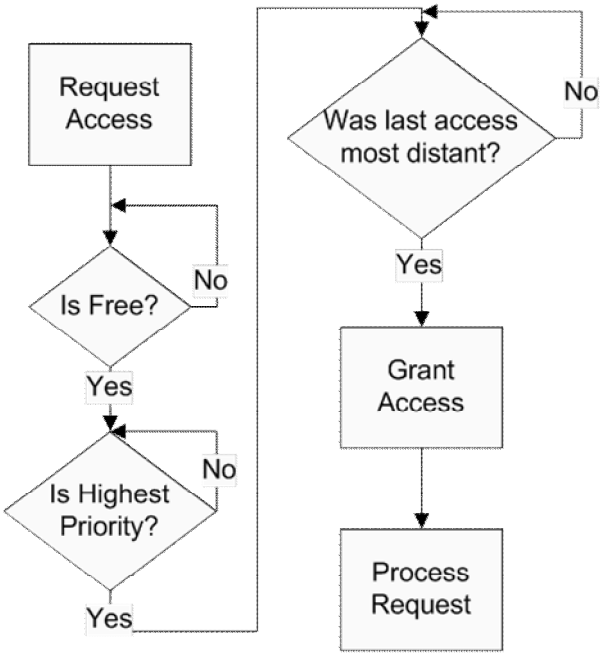

Figure 14: Arbitration System Flow Chart

High Speed Transmission

A method to quadruple the speed of the output from the maximum DDR2 frequency of 266 MHz to 1.07 Gsamples/s was developed. Using two clocks at 266 MHz, offset by 90 degrees, this method interleaves the data in four different signals later combined by xor gates to create the output. This is shown in figure 15 below.

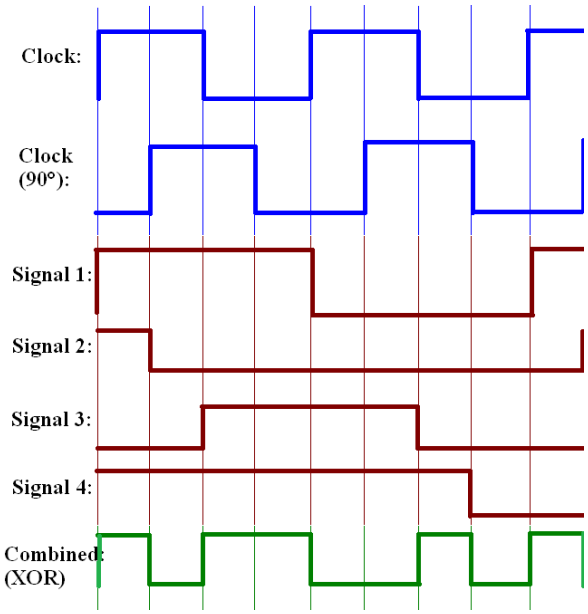

Figure 15: High-Speed Transmission Method

Controls for Excitation

To offer more flexibility to the user of the platform, two controls were added to the system. One was a method designed to establish which waveforms are sent to what pins. Another was the ability to add individualized delays to each pin.

Before implementing the flexibility of assigning waveforms to different pins, functionality was added to remember the location of the waveforms in the DDR2. The record that stores this information also remembers an identifying number associated with each waveform. Then, the control to assign waveforms to pins uses this record to determine the location of the data. This method was verified to work on two different waveforms assigned to arbitrary pins.

The other control, the adding of individual delays, was also successfully implemented. For testing purposes, a delay of up to 1 second long was allowed. The delay was implemented by keeping a run-time counter that begins counting when the signal to transmit is raised. The delays are kept in a table with units of the number of counts from the beginning of transmission. When a particular pins start time matches the counter, the transmission for that pin begins. This method was verified by oscilloscope using a one second delay. Further verification at a finer time scale will be done after the delay method is finalized.

PC to FPGA Communication

Initially, the proposed method to connect the FPGA to the PC was to use Ethernet. However, work that was done with the goal of implementing this showed that this interface is more difficult than necessary for this project. Additionally, the convenient USB support on the analog front end lowered the data rate requirements of the connection. Therefore, UART was chosen to make the connection to the PC. This protocol was tested on the FPGA in a stand-alone project, and was shown to function correctly. More work will be done to integrate this communication to the entire system.

Pulse Compression

Pulse compression is a technique which can be used to improve the penetration depth and signal to noise ratio (SNR) of an ultrasound device [2]. An encoded signal is sent to the transducer instead of a short pulse. The longer signal increases the energy transmitted, and provides a code which can be detected. Filtering the sampled reflected signal to detect the transmitted code can compress the pulse.

A MATLAB simulation has been written to test coded excitation and pulse compression. The encoded signal used in this simulation was the REC chirp discussed in the previous section. A Wiener filter is used for pulse compression. If there is a lot of noise in the output, the matched filter is the best solution. The matched filter correlates the input and output signals to compress the pulse. A disadvantage of the matched filter is that side lobes, or clutter occur as a result of cross correlation. An inverse filter compresses the pulse perfectly for a system with no noise, but is not accurate in the presence of noise. The Wiener filter aims to find an optimization between the two filters which accurately compresses the pulse but reduces clutter [4]. The Wiener filter equation contains a smoothing factor, which adjusts the operating point of the filter on a continuum between a matched and an inverse filter. Figure 16 shows the simulation results for the Wiener Filter, with varying smoothing factors. The SNR for the simulation was 60 dB.

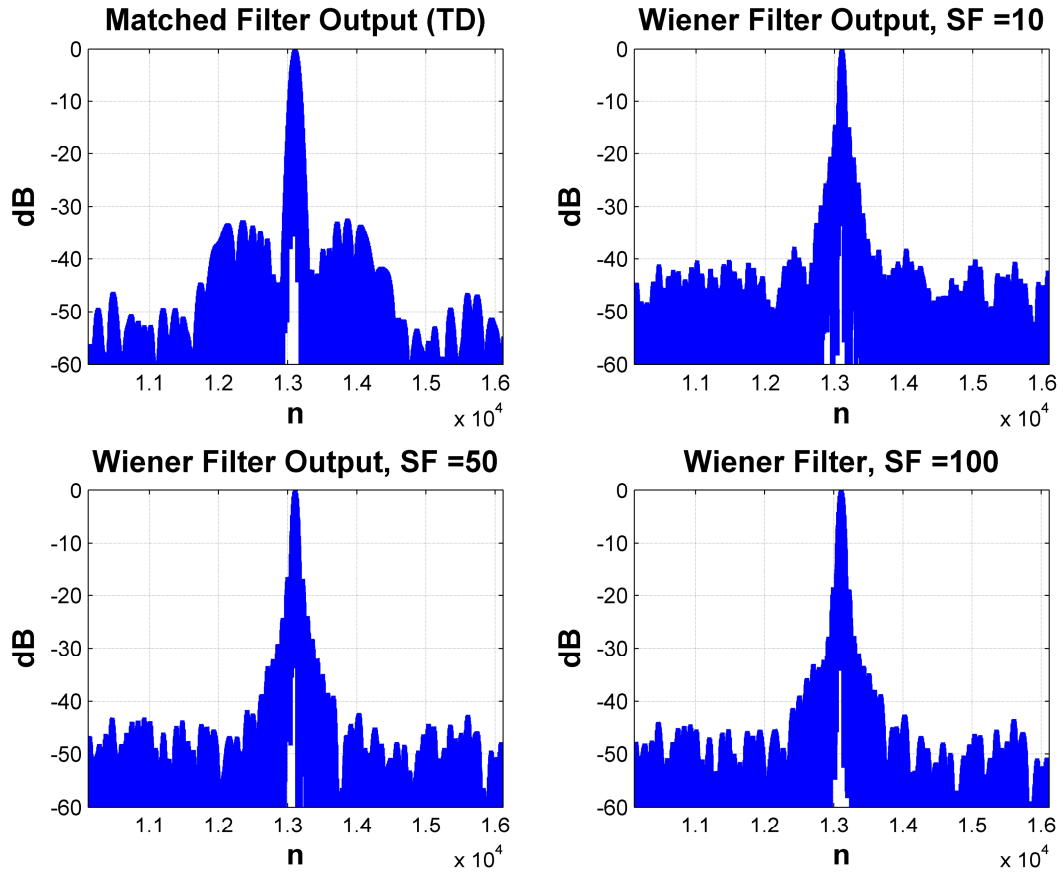

Figure 16: Pulse Compression Results

High Voltage Amplifier

The H-bridge was tested using the recommended switching time test circuit provided on the data sheet for the N-Channel. The following calculations were computed:

$P = I^2 \times R;$

$V_{DD} = 15$ V;

$V_{DS} = 10$ V;

$V_{GS} = 5$ V;

$\therefore V_d = 5$ V;

$V = I \times R;$

$P_{Rd}=I_d^2 \times R_d \rightarrow$ According to the graphs given, at those values, $I_d \cong 0.3$ A, $\therefore R_d \cong 20\ \Omega$. $P_{Rd}= (0.3)^2 \times (16.6667) = 1.5$ W. This is a little high for the resistors currently being used, but because the input signal is pulsed, the resistors should be able to dissipate the extra power.

The H-bridge supplied was determined not to be fast enough to operate at the speeds required for this project. It was decided that the H-bridge would either be constructed from selected MOSFETs or another part would be used.

T/R Switch

A simulation of the T/R switch in OrCAD was performed using the Spice model and test circuit that Texas Instruments provided through their website [5]. To simulate the T/R switch in OrCAD, B3, B2, B1 were set to high level logic (+5 V). According to the datasheet, the clamp voltage should be 1.9 V. The circuit was simulated for inputs at 1, 2, 5, 10, 20, 50, and 90 V_{in} amplitude (at 2MHz). At 1, 2, 5, and 10 V, the clamp voltage was below 1.9 V, at about 1.88 V. But when simulated at 20, 50 and 90 V, the clamp voltage was above 1.9 V. This spec is acceptable because this includes overshoot, and the datasheet is not specified for overshoot. Figure 17 shows the simulated Spice model.

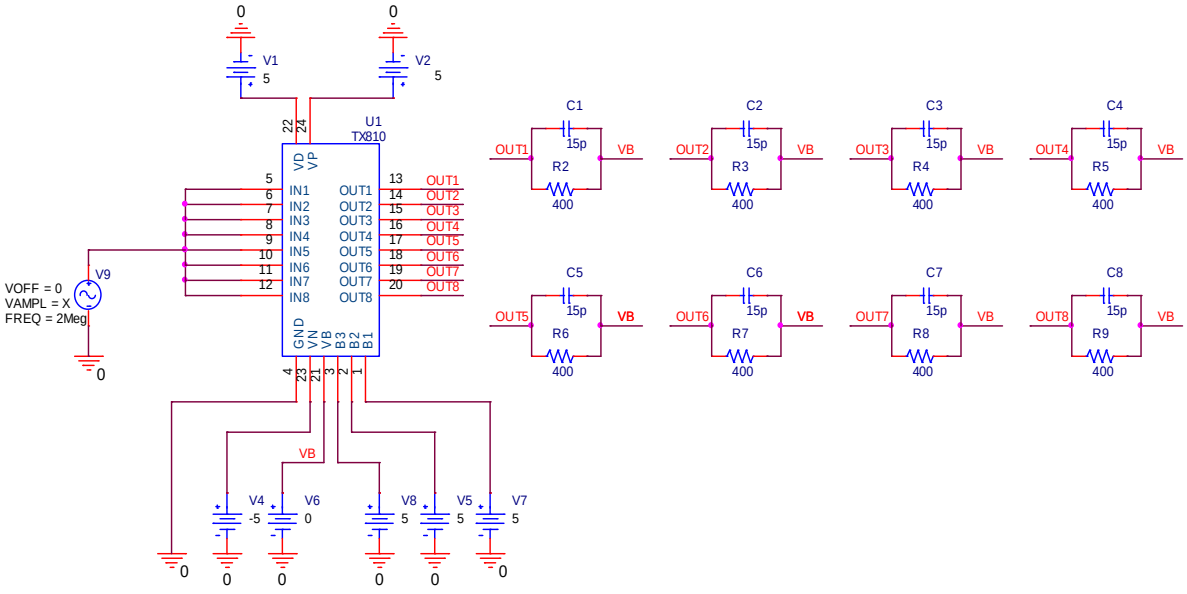

Figure 17: Schematic diagram of simulated circuit provided by Texas Instruments [5]

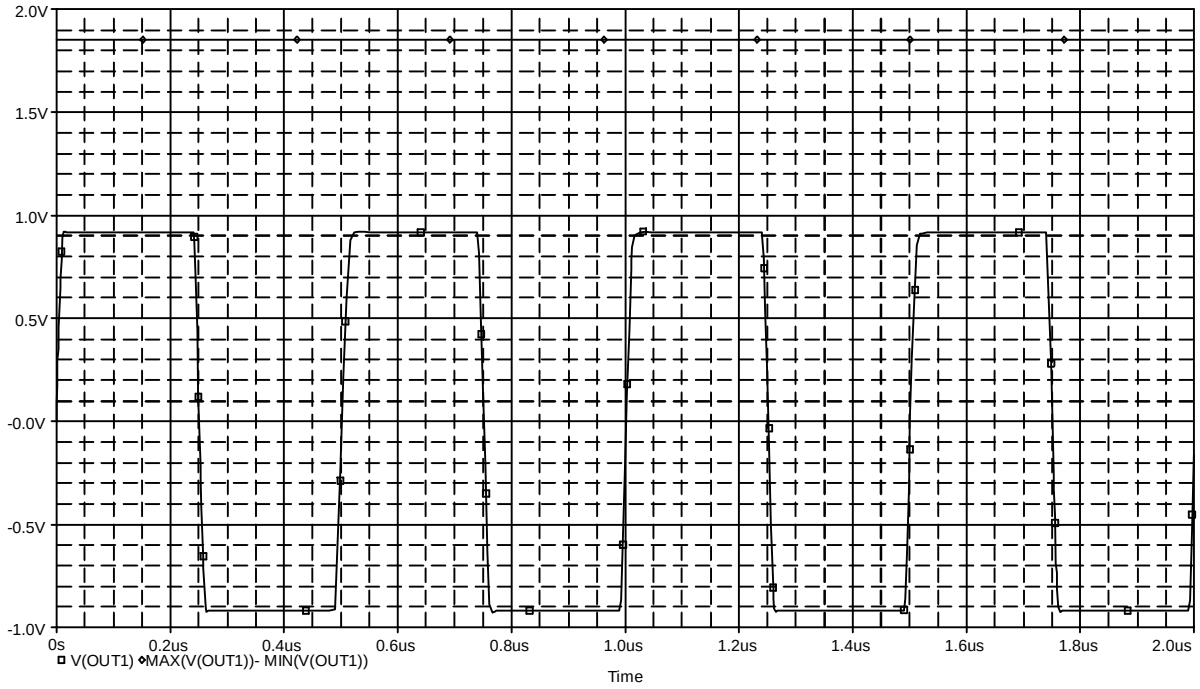

Figure 18: Simulation at $10V_{in}$ @ 2MHz. V_{outpp}= 1.85 V

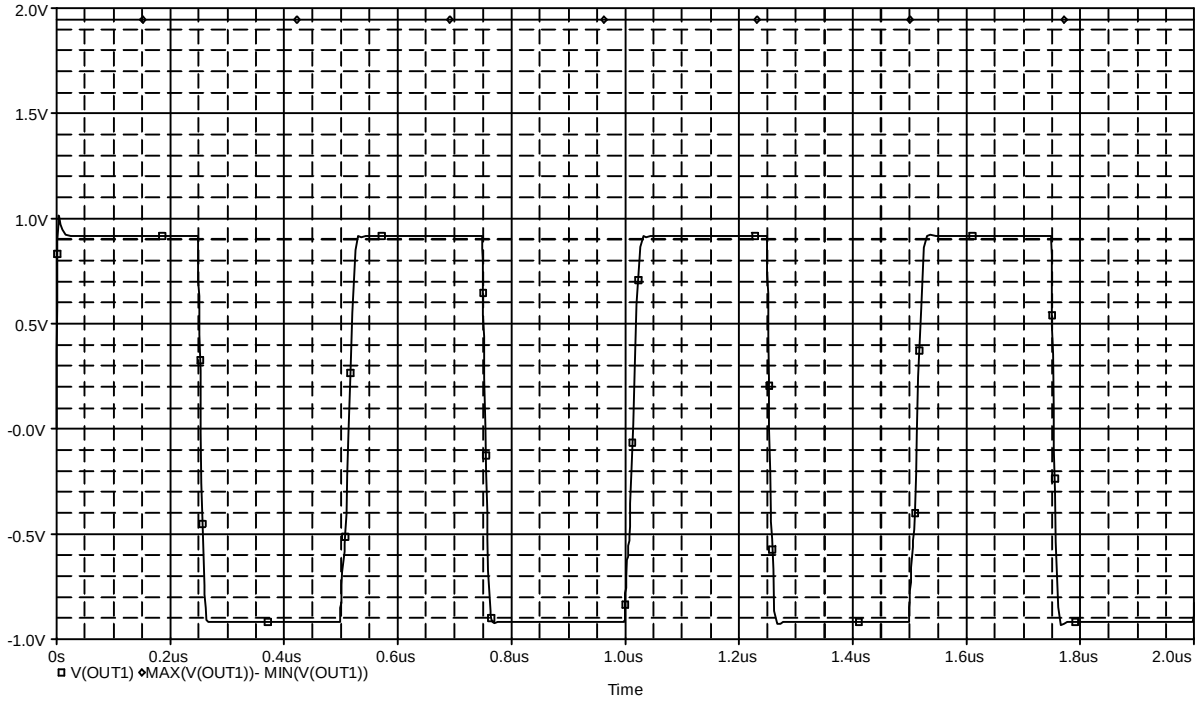

Figure 19: Simulation at $90V_{in}$ @ 2MHz. V_{outpp}= 1.94 V

Printed Circuit Board

In order to operate at the desired frequencies a printed circuit board (PCB) is a requirement. The PCB will contain the op amp, H-bridge, and Transmit/Receive switch, along with a ZIF connector that will connect the transducer to the transmit and receive circuitry.

OrCAD PCB designer is used for the PCB design. The book *Complete PCB Design Using OrCad Capture and PCB Editor* by Kraig Mitzner is being used to help understand how to utilize PCB design tools. The example at the beginning of Chapter 2 was used as a tutorial to get familiar with the design process. This was followed with using other parts to see if other footprints would transfer. Unfortunately, the footprints needed for the parts that will be used in the project are not available through the tool, so they will have to be designed using the pad and footprint designer provided by the tool.

Another problem that arose with the board is the footprints themselves. The packages are very difficult to solder if a reflow oven is not available. So the parts may need to be sent to the manufacturer of the PCB to have them solder the components onto the board. This will increase the price of manufacturing the PCB, and will be considered according to need and if it is concluded that the soldering cannot be done at the University.

Tentative Schedule

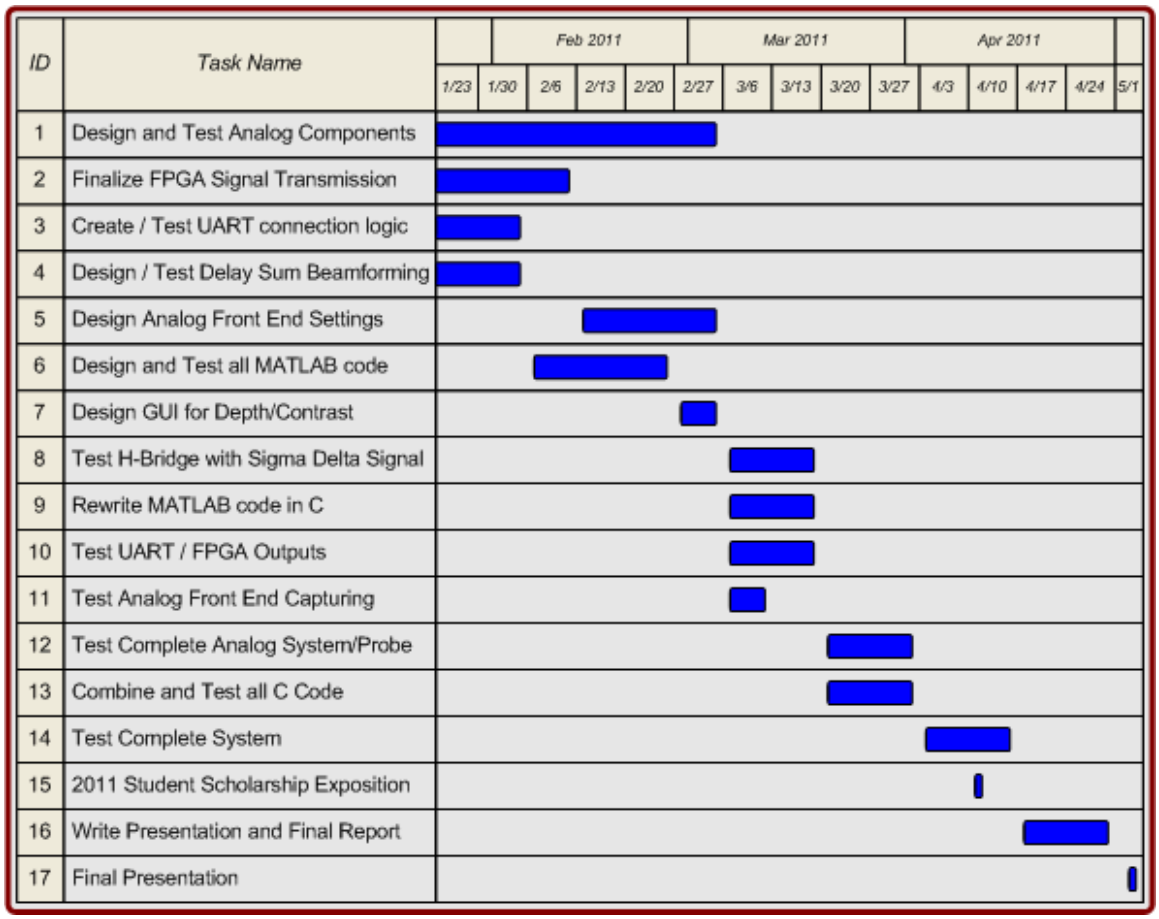

Figure 20: Task Schedule

The subsequent page shows a detailed breakdown of tasks and the group members responsible for that particular item.

Week	Stage	Task	Resource
01/24/11-02/28/11	1	Design and test • H-Bridge • Passive Filter • Excitation PCB • Wiring for ultrasound transducer • Transmit/Receive (T/R) switch	David
01/24/11-01/30/11	1	Create a UART connection between the computer and the FPGA board to transmit and receive on both ends	Sam and Jacob
01/24/11-02/07/11	1	Finalize the FPGA's processing of the waveform and high speed transmission to pins.	Jacob
01/24/11-01/31/11	1	Design and test a delay sum beamformer using Fields II MATLAB package	Emma
02/14/11-02/28/11	1	Design analog front end settings	Sam and Jacob
02/07/11-02/21/11	1	Design and test MATLAB code for • Time-gain compensation • Envelope detection • Log compression • Image display	Emma and Sam
02/28/11	1	Design a graphical user interface (GUI)	Emma
03/07/11-03/14/11	2	Transmit sigma-delta signal into the H-Bridge and test	Emma and David
03/07/11-03/14/11	2	Rewrite all MATLAB code in C	Emma and Sam
03/07/11-03/14/11	2	Test Ethernet and FPGA outputs for a generic signal	Jacob
03/07/11	2	Test analog front end signal capturing	Sam
03/21/11-03/28/11	3	Combine complete analog system and test probe	David and Jacob
03/21/11-03/28/11	3	Combine and test all C code	Emma and Sam
04/04/11-04/11/11	4	Combine and test complete system	All
04/11/11	5	2011 Student Scholarship Exposition	All
04/18/11-04/25/11	5	Write oral presentation and final report	All
05/03/11	6	Final Presentation	All

Equipment List

- LeCroy High Speed Oscilloscope 725Zi (Chestnut Ridge, NY)
- Blatek 128 pin Ultrasound Probe (State College, PA)
- Ultrasound Testing Phantom
- Xilinx Virtex 5 Development Kit ML509 (San Jose, CA)
- UART Null Modem Adapter
- Analog Devices Analog Front End AD9276-80KITZ (Norwood, MA)
  - Low noise amplifier
  - Variable control amplifier
  - ADC
- MOSFETS x4 for the H-Bridge – model to be determined
- Printed Circuit Boards (2)
- Texas Instruments –T/R Switch TX810 (Dallas, TX)
- Software
  - Matlab Version 7.5.0.342 R2007b (Natick, MA)
    - Sigma Delta Toolbox
    - Field II
  - Agilent Connection Expert (Santa Clara, CA)
  - Xilinx ISE (San Jose, CA)
- Hydrophone (University of Illinois, Champaign, IL)

Acknowledgments

The authors would like to thank Analog Devices and Texas instruments for their donation of parts.

The authors also thank Dr. Irwin, Dr. Lu, Mr. Mattus, and Mr. Schmidt for their assistance.

This work is partially supported by a grant from Bradley University (13 26 154 REC)

References

[1] R. Schreier and G. C. Temes. *Understanding Delta-Sigma Data Converters*, John Wiley & Sons, Inc., 2005.

[2] M. Oelze. "Bandwidth and Resolution Enhancement Through Pulse Compression," IEEE Trans. Ultrason., Ferroelectr. Freq. Contr., vol. 54, no. 4, pp. 768-781, Apr. 2007.

[3] R. Schreier, *The Delta-Sigma Toolbox Version 7.3*. Analog Devices, Inc, 2009.

[4] T. Misaridis and J. A. Jensen. "Use of Modulated Excitation Signals in Medical Ultrasound," IEEE Trans. Ultrason., Ferroelectr. Freq. Contr., vol. 52, no. 2, pp. 177-191, Feb. 2005.

[5] Texas Instruments. "TX810." Internet: http://focus.ti.com/docs/prod/folders/print/tx810.html, Jun. 11, 2010.