GLOBAL ACTIVE NOISE CANCELLATION FOR CELL PHONE PRIVACY

Ву

Joel Godard

Hershed Tilak

Final Report for ECE 445, Senior Design, Spring 2013

TA: Justine Fortier

01 May 2013

Project No. 27

Abstract

In this project, we explore the feasibility of using Active Noise Cancellation (ANC) to cancel out a cell phone user's voice in the far field. We first built a circuit to read in a user's voice signal, invert it, and output the processed wave. We then performed experiments to determine the spatial intensity pattern of the human voice and analyzed speaker array patterns to determine which ones best matched the determined intensity pattern of the human voice. Finally we connected the speaker array to the circuit and measured the achievable noise cancellation provided by our design.

It was found that the cancellation was negligible across most of the frequency spectrum. Certain frequencies could be attenuated, however the level of cancellation was dependent on many non-static variables. It was ultimately determined that using ANC to cancel out a cell phone user's voice in the far field is unfeasible with the implemented design.

Contents

1. Introductio	n1
1.1 Block D	escriptions1
2 Design	5
2.1 Design	Procedure5
2.2 Design	Specifications9
3. Design Veri	fication14
3.1 Testing	Procedures14
3.2 Failed V	erifications19
3.3 Toleran	ce Analysis19
4. Costs	
4.1 Parts	
4.2 Labor	
5. Conclusion	
5.1 Accomp	lishments
5.2 Experim	nent Results
5.3 Uncerta	inties
5.4 Ethical	considerations
5.5 Future	work
Appendix A	Characterization of Acoustic Radiation Patterns25
References	
Appendix B	Requirement and Verification Table
Appendix C	Source Code
Appendix D	Schematics

1. Introduction

Cell phones have allowed for a great level of connectivity giving us the ability to communicate with each other whenever and wherever we so chose. However, this ability to speak to anyone at any time has essentially put our lives on display for anyone within earshot to hear. The goal of this project was to determine the feasibility of using Active Noise Cancellation (ANC) to cancel out a cell phone user's voice in the far field. If viable, the methodology could ultimately be incorporated into modern cell phones to provide increased privacy to cell phone users.

The function of the circuit component of this project is to cancel out a speech signal at its source. It does this by producing an output wave that is 180° out of phase with the initial sound wave so that the two waves cancel when they are superimposed. In the final implementation, it was found that the circuit is only capable of achieving 180° phase shift for select frequencies. As a result, only certain frequencies can be actively cancelled by this implementation.

The function of the experimental component of this project is to acquire an accurate spatial intensity pattern for the typical human voice. It was found that the results obtained in the experiment match the results that are quoted in literature [1]. The details of the experimental component of this project can be found in Appendix A.

1.1 Block Descriptions

Figure 1 shows a top level block diagram for the circuit component of the project.



Figure 1: Top Level Block Diagram



Figure 2: Audio Input block diagram

Audio Input

The Audio Input block, shown in Figure 2, consists of two subcomponents: a microphone to acquire the initial input speech signal and a preamplifier to raise the input signal to the appropriate amplitude levels. The microphone transduces the audio signal of the person's voice which is then fed into the preamplifier. The preamplifier increases the voltage amplitude of this signal and outputs it to the Audio-DSP.

Audio DSP Interface

The Audio-DSP Interface, shown in Figure 3, consists of a passive low pass anti-aliasing filter and an Audio Codec. The low pass filter has a cutoff frequency of 3 kHz, corresponding to the maximum frequency of the human voice [1]. The signal from the low pass filter is fed to the Analog to Digital Converter (ADC) of the Audio Codec. The digitized signal is passed on to the DSP block, which manipulates it and passes it back to the Audio Codec. This processed signal is then passed through the Digital to Analog Converter (DAC) of the Audio Codec before being sent to the Audio Output block. The Audio Codec communicates with the DSP through a Serial Peripheral Interface (SPI) communication protocol.



Figure 3: Audio-DSP Interface Block Diagram



Figure 4: Audio Output Block Diagram

Audio Output

The audio output block, shown in Figure 4, consists of eight pairs of speaker amplifiers and speakers. Each speaker amplifier reads in the analog signal from the DAC in the Audio-DSP Interface and amplifies it. The amplified signal is fed into the corresponding speaker which converts it into a physical signal.

DSP

The DSP block, shown in Figure 5, consists of a single digital signal processor. The DSP receives an incoming digital signal from the Audio CODEC. It processes the signal and produces an inverted signal. The inverted signal will cancel out the original signal when the two are superimposed. The inverted signal is sent to the Audio CODEC, which converts it into an analog signal before passing it on to the Audio Output block. In our final design, it was not possible to acquire the JTAG Emulator needed to program our DSP and so the Development Starter Kit (DSK) for the chip was used instead.



Figure 5: Digital Signal Processing Block Diagram

Power Supply

The power supply, shown in Figure 6, consists of a 9 V source, a 9 V-5 V DC-DC converter, a 9 V-3.3 V DC-DC converter, a 9 V-1.5 V DC-DC converter, and a 9 V-1.25 V DC-DC converter. The DC-DC converters are used to convert the 9 V source into the 5 V, 3.3 V, 1.5 V, and 1.25 V supplies required by the rest of the circuit blocks. The outputs of the DC-DC converters power the preamplifier, the Audio CODEC, the DSP chip, and the output speaker amplifiers. In the original design, a 9 V battery was used for the 9 V source. However, it was found that the battery could not supply the necessary current for the circuit and so a DC power supply was used instead in the final design.



Figure 6: Power Supply Block Diagram

2 Design

The schematics for all of the components discussed in this section are included in Appendix D for reference.

2.1 Design Procedure

Microphone

The microphone that is implemented in this circuit is the Analog Devices ADMP404 Microphone. This microphone was chosen for its omnidirectional reception pattern, the flatness of its frequency response across the band of interest (-3/+2 dB), high sensitivity of -38 dBV at 1 kHz and 94 dB SPL, and its small size of 3.35 mm × 2.50 mm × 0.88 mm [2]. The small size of this microphone was particularly attractive as the goal of the project was to examine the feasibility of implementing global active noise cancellation for cell phones. Thus, choosing a small microphone of the type that is used commonly on cell phones was of great import. Larger, separate condenser microphones were considered initially, but dismissed for the reason mentioned above. The sensitivity of this microphone was the final point that set it apart from many other choices in small microphones. The sensitivity at -38 dBV/Pa was 3 dB above most of the other small microphones that were considered.

Microphone Preamplifier

The microphone amplifier implemented in this design is the Texas Instruments INA333 Instrumentation Amplifier. This amplifier was chosen because it is an instrumentation amplifier and has a low noise figure of 50nV/vHz [3].

The gain of this amplifier is set to 4.78 V/V. This gain was chosen due to the low expected levels of the voice signal that will be output from the microphone and the desire to utilize the full input voltage range of the Audio CODEC. This gain was calculated considering the typical human voice pressure levels in conversation, the sensitivity range of the microphone, as well as the expected distance between the person speaking and the microphone. Typical voice levels at 1 m are quoted in [4] as ranging from 45 dB SPL to 85 dB SPL with 65 dB SPL being considered the nominal value. This range was limited to 55 dB SPL to 75 dB SPL for this experiment as there was some degree of control over the expected sound levels that were generated in the testing procedures. Assuming that the person speaking into the microphone will distance the microphone from his mouth anywhere from 1 cm to 8.85 cm (one fourth of a wavelength of sound in air at 1 kHz), the expected sound pressure level at the microphone is

$$P_{\rm mic,dB} = 20 \log \left(\left| 10^{\frac{P_{\rm 1m,dB}}{20}} \cdot \frac{1}{d} \right| \right) \tag{1}$$

where $P_{1m,dB}$ is the sound pressure level 1 m away from the sound source in dB and d is the distance of the microphone from the sound source in meters. The maximum, minimum, and typical operating condition of the microphone were calculated using Equation (1) to apply 75 dB SPL at 1 m to the microphone placed 1 cm from the mouth, 55 dB SPL at 1 m to the microphone placed 8.85 cm from the mouth, and 65 dB SPL at 1 m to the microphone placed 5 cm from the mouth. This resulted in expected pressure levels at the microphone ranging from 76 dB SPL to 115 dB SPL with a nominal condition of 91 dB SPL. Utilizing this expected pressure level range along with the specified microphone sensitivity tolerance, the range of voltage levels out of the microphone under normal conditions was calculated. The voltage output from the microphone with sensitivity S experiencing a sound pressure level P_{mic} is

$$V_{out} = S \cdot P_{mic} \tag{2}$$

where P_{mic} and S are expressed in linear magnitude. The range of expected output voltage levels from the microphone along with the nominal expected output voltage are summarized in Table 1.

Pressure Level at 1 m	Microphone Distance from Mouth	Microphone Sensitivity	V _{out}	Pressure Level at Microphone
55 dB SPL	8.85 cm	-41 dBV/Pa	1.168 mV	76 dB SPL
65 dB SPL	5.00 cm	-38 dBV/Pa	8.948 mV	91 dB SPL
75 dB SPL	1.00 cm	-35 dBV/Pa	200.400 mV	115 dB SPL

Table 1: Expected Microphone Output Voltage

Given these expected output voltages along with the $1V_{RMS}$ maximum voltage input of the Audio CODEC, the gain of the amplifier was specified as 5 V/V so that the maximum expected audio input to the microphone would not cause clipping at the input of the Audio CODEC.

In order for the amplifier to be used with a single power supply, the amplifier inputs had to be biased at half the supply voltage. Combining this criterion with the maximum output current from the microphone set the input impedance of the microphone at 12 k Ω . The microphone can supply a maximum output voltage of 1.6 V and a maximum output current of 90 μ A.

Low Pass Anti-Aliasing Filter

The low pass anti-aliasing filter is a first order filter with a cutoff frequency of 3 kHz. The reasoning for this design decision is that the human voice is generally limited to 3 kHz. Therefore, frequencies above this threshold are likely coming from outside sources which should not be allowed to pollute the voice signal.

Audio Codec

The Audio Codec used is the Texas Instruments TLV320AIC23 Stereo Audio Codec. This Audio Codec was chosen because it is the same one that is used by the TMS320C6713B Development Starter Kit (DSK). Using the same codec allowed for quick software development on the DSK and little design modification in the transition to the final implementation of the codec. The input signal of the Audio Codec is sampled at 8 kHz. This sampling rate was chosen because it is the minimum sampling rate on the Audio Codec which is still above the Nyquist rate needed to sample the 3 kHz bandlimited signal without aliasing [5].

Digital Signal Processor

The DSP used is the Texas Instruments TMS320C6713B Digital Signal Processor. The DSP was chosen because a DSK for the chip was already available for checkout in the Senior Design Laboratory. We were

unable to obtain the JTAG Emulator needed to program the DSP on our final circuit board and so we used the DSK instead for our final demonstration.

Figure 7 shows a flow chart of the DSP algorithm. After the DSP has been properly powered up and initialized, it starts reading in signal samples from the Audio Codec ADC. Once the buffer in the DSP has been filled, the Fast Fourier Transform (FFT) is taken to convert the signal into the frequency domain. While inside the frequency domain, the signal is time-shifted to try to compensate for latency in our circuit. The signal is also frequency scaled to compensate for the frequency response of the output speakers. Next, the Inverse Fast Fourier Transform (IFFT) is taken to convert the signal back into the time domain. At this point, the first sample in the buffer is sent to the Audio Codec DAC, the buffer is shifted, and a new sample is read in. The source code for the DSP can be found in Appendix C.

The length of the buffer used in the program is 32 samples. This choice was made based on past experience with a similar DSP which had a slightly slower processor. Since the current DSP has a faster processor, it was easier to meet runtime specifications for the algorithm.

The DSP algorithm has to run in real time. The DSP has a processor speed of 225 MHz [6]. In addition, we are sampling at 8 kHz. Therefore, to meet runtime we have to output our processed sample within one eight thousandth of a second = 0.125 ms of reading an input sample. This means all of our processing needs to be done in less than 0.125 ms × 225 MHz = 28,125 processor cycles.

After the algorithm implementation was finished, the size of the buffer was increased to see if the program could still meet the necessary runtime specification. It was found through experimentation that the algorithm could not meet runtime with a buffer length greater than 32 samples.

Since the length of our buffer was limited to 32 samples, the FFT gave us only 32 distinct frequencies to manipulate in the Fourier Domain. When we attempted to implement frequency scaling, we found that it produced undesirable results due to the limited number of coefficients that could be scaled and so we decided not to use it in our final design.

Speaker Amplifier

The speaker amplifier that is implemented is the Texas Instruments LM4818 Audio Power Amplifier. This amplifier was chosen because of its high input impedance well as the fact that it was designed specifically to supply 120 mW of power (the typical power draw of each of the chosen speakers) to an 8 ohm load when supplied with 3.3 V [7]. This audio amplifier will supply this power necessary to drive each speaker through its class-AB output stage. The speaker amplifiers provide a high input impedance allowing up to 8 speakers to be driven by 8 separate speaker amplifiers without overdrawing the Audio CODEC DAC. The number of speakers that would be implemented in the final speaker array was unknown during the design of this amplifier stage. For this reason, the speaker amplifier circuit was designed to accommodate 8 speakers in order to allow for flexibility later on in the speaker array design stage of the project. The gain of each amplifier was set to 0.89 V/V allowing the full output voltage range of the DAC (1V_{RMS}) to correspond to the maximum 0.89 V_{RMS} that can be supplied across the terminals of each speaker.

The input impedance of each amplifier was set so that all 8 amplifiers in parallel would not draw more current from the Audio CODEC DAC than it was capable of supplying. The minimum load that the DAC can drive is 10 k Ω , thus the minimum input impedance of each of the amplifiers was specified as 80 k Ω .



Figure 7: DSP Algorithm Flow Chart

Speaker

The speaker that is implemented in this design is the CUI Inc. CLS0271M-L152 Speaker. This speaker is capable of producing sound levels greater than 72 dB SPL at 1 meter across the entire voice band when driven at 0.1 W [8]. This sound level is within 3 dB of the expected maximum voice level allowing the speaker to output an audio signal of equal magnitude to that of the person's typical voice level which is critical for noise cancellation. The constraining factors in the speaker selection were size and maximum output pressure levels. Larger speakers were considered because of their ability to provide larger pressure levels as well as having lower frequency responses. However, these speakers were dismissed due to the fact that speakers greater than roughly an inch in diameter would not fit on modern handheld devices and so would be ill fit for this experiment. Of all speakers that were examined having a diameter less than 30 cm, the CUI Inc. CLS0271M-L152 speakers had the highest low frequency response as well as the largest output pressure levels at frequencies within the 300 Hz to 3 kHz range.

DC Power Supply

The DC power source that was originally specified in this experiment design was a simple 9 V battery. However, it was found during verification that the common 9 V battery could not meet the requirements specified for the power source. Namely, the 9 V battery could not supply the approximately 1.5 A that were drawn by the entire circuit while maintaining a voltage output above 6.25 V. For this reason, an Agilent E3631A Triple Output DC Power Supply was used as the power source for Power Supply block and the rest of the circuit. This DC power supply was chosen due to its availability in the Senior Design Lab.

DC-DC Converters

All of the DC-DC converters that were implemented used Texas Instruments LM317 Adjustable Voltage Regulators. These regulators are capable of supplying currents from 10 mA to 1.5 A and allow for adjustable voltage regulation using two external resistors [9]. These voltage regulators were chosen in particular because of their availability in the ECE Parts Shop as well as their flexibility of use. The output voltage of each regulator is controlled by two external resistors allowing for the design and layout of a single circuit that can be used to create all four necessary voltage supplies by simply changing resistor values. Fixed output voltage regulators were considered, but would have required a different PCB layout for each one which would have taken more time to design in Eagle. In the final implementation, the 9V to 5V DC-DC converter supplies power to the microphone preamplifier; the 9V to 3.3V DC-DC converter supplies power to the Audio CODEC, the DSP, and the speaker amplifier; the 9V to 1.5V DC-DC converter supplies power to the Audio CODEC; the 9V to 1.25V DC-DC converter supplies power to the DSP.

2.2 Design Specifications

Microphone

The microphone bias voltage is supplied by the Audio CODEC, thus no calculations were needed for the implementation of the microphone. All component values surrounding the microphone were specified in its data sheet.

Microphone Amplifier

The gain of the microphone amplifier is specified as

$$G = 1 + \left(\frac{100 \,\mathrm{k}\Omega}{\mathrm{R}_{\mathrm{G}}}\right) \tag{3}$$

where R_G is the resistance of the gain set resistor. Setting this gain to 5 V/V gives a resistor value of 25 k Ω for R_G [3].

To bias the input of the microphone amplifier and set the input impedance, a resistor from the 5 V power supply to the input terminal was used along with a resistor from the input terminal to ground. Assuming that the input impedance of the amplifier is infinite, these resistors were set to have equal values so that the quiescent bias point of the input of the amplifier was 2.5 V. The input impedance of the amplifier is

$$R = \frac{V_{MIC} - 200i_{MIC} - Z_{L}i_{MIC} - 2.5}{-\frac{1}{2}i_{MIC}}$$
(4)

where V_{mic} is the maximum ac output voltage of the microphone, i_{mic} is the maximum current that the microphone can supply from its output terminal, and Z_L is the impedance of the coupling capacitor at 1 kHz. Solving for R gives a bias resistor value of approximately 12 k Ω . This value simply serves as a minimum specification. In the final implementation of the circuit, 20 k Ω resistors are used due to their availability.

Low Pass Anti-Aliasing Filter

The resistance value of the filter was chosen to be small in comparison with the input resistance of the Audio Codec, which is listed as $14 \text{ k}\Omega$ [5]. This allows the input signal of the filter to reach the input of the Audio Codec without being significantly attenuated. It was found using Equation (5) that a standard resistance value of $1.1 \text{ k}\Omega$ resulted in 93% of the input signal being passed to the Audio Codec which was determined to be sufficient.

% of signal kept =
$$\frac{\text{Audio Codec Resistance}}{\text{Filter Resistance} + \text{Audio Codec Resistance}} * 100\%$$
(5)

To determine the value of the capacitor in the filter, Equation (6) was used, plugging in the desired values of f and R. It was found that for f = 3,000 Hz and R = $1.1 \text{ k}\Omega$, we need that C = 0.048 μ F. A standard capacitor value of 0.047 μ F capacitor was used instead, meaning that we had a cutoff frequency of f = 3.08 kHz using Equation (6).

$$C = \frac{1}{2\pi fR}$$
(6)

Figure 8 shows a simulation of the filter with the chosen resistor and capacitor values. As can be seen, the simulation shows a filter cutoff frequency of 3.06 kHz.



Figure 8: Low Pass Anti-Aliasing Filter Simulation

Digital Signal Processor

For proper operation, the DSP needs its RESET pin to be held low for 100 ns [6]. In addition, the RESET pin needs a voltage of 2 V to be considered on [6]. To be safe, we decided to hold it low for one second. Equation (7) calculates the voltage at the output of the circuit after one time constant has passed where the time constant is equal to R × C. As can be seen, if V_{DD} = 3.3 V, then V = 2 which is the desired value. Therefore, after R × C seconds, the RESET pin will reach 2 V and be considered active. In order to achieve R × C = 1 we decided to choose R = 100 k Ω and C = 10 μ F.

$$V = V_{DD} \left(1 - \frac{1}{e} \right)$$
⁽⁷⁾

Speaker Amplifier

The gain and input impedance of each of the audio amplifiers are specified as

$$G = 2\left(\frac{R_{f}}{R_{i}}\right)$$
(8)

$$Z_{in} = \frac{R_i R_f}{R_f + R_i}$$
(9)

where R_f is the feedback resistor value and R_i is the input resistor value [7]. Setting the gain to 0.89 V/V and the input impedance to 80 k Ω as specified in Section 2.1 and solving this system of equations yields and input resistor value of 116 k Ω and a feedback resistance value of 260 k Ω . The final circuit that is implemented utilizes an input resistance value of 120 k Ω and a feedback resistance value of 270 k Ω as these are the nearest common resistor values to those calculated above.

Choosing these resistor values set the constraints for the input capacitor value which set the high pass cutoff frequency for the amplifier input. The cutoff frequency of the input of the amplifier is

$$f_{\rm C} = \frac{1}{2\pi R_{\rm i} C_{\rm i}} \tag{10}$$

where C_i is the input capacitor value and R_i is the input resistor value [7]. Because the frequency content of the human voice had not yet been characterized, the value of C_i was chosen to be 0.027 μ F giving a cutoff frequency of 40 Hz in which is well below the 300 Hz cutoff frequency of the human voice popularly referenced by the telecommunication industry.

Power Supply

The output voltage of each DC-DC converter used for the power supply is

$$V_{out} = V_{ref} \left(1 + \frac{R_2}{R_1} \right) + \left(I_{adj} \cdot R_2 \right)$$
(11)

where V_{ref} is 2.5 V, I_{adj} is typically 50 μ A, R_2 is the value of the resistor from the Adjust terminal to ground and R_1 is the value of the resistor from the Adjust terminal to the regulator output [9]. Solving Equation (11) for R_1 and setting R_2 to a common value yields the R_1 and R_2 resistor values summarized in Table 2.

Output voltage	R ₁	R ₂
V ₀ = 5.00 V	R ₁ = 1428 Ω	$R_2 = 5 k\Omega$
V ₀ = 3.30 V	R ₁ = 2778 Ω	$R_2 = 5 k\Omega$
V ₀ = 1.50 V	R ₁ = 5 kΩ	$R_2 = 1 k\Omega$
V ₀ = 1.25 V	R ₁ = 5 kΩ	$R_2 = 0 \Omega$

Table 2: Power Supply Resistor Values

Figure 9 shows a simulation of the power supply using the resistor values listed in Table 2. The simulation shows that all four of the DC-DC converters should function as desired.



Figure 9: Power Supply simulation (from top down: 6.25V to 12V input voltage sweep, output of 9V to 5V DC-DC converter, output of 9V to 3.3V DC-DC converter, output of 9V to 1.5V DC-DC converter, output of 9V to 1.25V DC-DC converter)

3. Design Verification

A detailed list of all requirements and verification procedures, as well as quantitative results, can be found in Appendix B.

3.1 Testing Procedures

Audio Input

The first test performed was to check the power pins of both the microphone and the microphone preamplifier to verify that they were all being supplied the appropriate voltages. This was done by connecting the Audio Input block to the Power Supply and measuring all of the power pins with a multimeter. The values from the multimeter were all verified to be within the tolerances specified in Table 6.

For the microphone, it is important that the output voltage levels are in the correct range before reaching the amplifier. The voltage levels of the microphone output were tested by attaching an oscilloscope across the OUTPUT and GND pins of the microphone and speaking into it. The RMS voltage levels were measured using the oscilloscopes measure function and were verified to be within tolerance. Figure 10 shows the output of the oscilloscope.

The other crucial requirement for the Audio Input block is that the microphone preamplifier provides a gain of 5 ± 0.25 V/V. This was verified by feeding a 1 kHz sine wave with a peak to peak voltage of 100 mV into the preamplifier from a function generator. The output of the preamplifier was displayed on the oscilloscope and verified to have a peak to peak voltage of 4.78 V/V, which is within the ±0.25 V/V tolerance specified. Figure 11 shows the output of the oscilloscope.



Figure 10: Microphone Output Voltage Level Verification



Figure 11: Microphone Preamplifier Voltage Gain Verification

Audio-DSP Interface

For the Audio Codec, all of the power pins need to be supplied with the correct voltages. The procedure for verifying this is the same one that was used for the Audio Input block. All of the pins were found to be supplied with voltages within the tolerances specified in Table 7.

The Audio Codec also needs to be sampling at 8±0.25 kHz for our algorithm to run in real time. To verify this, the Audio Codec was initialized and the oscilloscope was connected across the LRCIN and DGND pins of the Audio Codec. The frequency of the signal on the LRCIN pin was measured using the oscilloscope's built in function and verified to be 8.0000 kHz, which is within the tolerance specified. Figure 12 shows the output of the oscilloscope. The LRCIN pin acts as a synchronization pulse that needs to be asserted every time the DSP reads a value from the codec. Therefore it corresponds directly to the sampling frequency of the codec.



Figure 12: Audio Codec Sampling Rate Verification



Figure 13: Anti-Aliasing Filter Cutoff Frequency Verification

The Low Pass Filter needs to have a 3dB cutoff frequency at 3±0.3 kHz. This was verified by connecting an oscilloscope across the capacitor in the filter. A 2.01 V peak to peak wave was produced on a function generator and fed into the filter. The frequency of the input wave was varied until the output wave on the oscilloscope reached 1.41V peak to peak. The output of the oscilloscope is shown in Figure 13. The frequency at which this happened corresponds to the 3 dB cutoff frequency. It was found that for our filter, this frequency occurs at 3.2 kHz which is within the specified tolerance.

Audio Output

The audio amplifier needs all of its power pins supplied with the correct voltages. The procedure for verifying this is the same as before. All of the pins were tested and verified to have supply voltages within the tolerance range specified in Table 9.

The audio amplifier needs to have a voltage gain of 0.89±0.05 V/V. This was verified by feeding a 1 kHz sine wave with a peak to peak voltage of 1.4 V into the amplifier from a function generator. The output of the amplifier was displayed on the oscilloscope and verified to have a peak to peak voltage of 1.27 V, meaning that the amplifier has a gain of 0.91 V/V, which is within the ±0.25 V/V tolerance specified. The output of the oscilloscope is shown in Figure 14.



Figure 14: Speaker Amplifier Voltage Gain Verification

It is also necessary that the speakers can be driven at 1 W input power from 300 Hz to 3 kHz. If this was not the case, it would mean that our speakers were defective. The requirement was verified by applying a 1.25 V peak to peak sine wave across the input terminals of each of the speakers. The frequency of the wave was varied from 300 Hz to 3 kHz and it was verified that the speakers produced sound across all of these frequencies.

For global noise cancellation, it is important that the spatial intensity pattern of the speaker array matches that of the human voice. This was verified by performing the experiments outlined in Appendix A on five people and on the speaker arrays. The results of the two experiments were then compared. It was found that the final speaker array chosen was able to match the spatial intensity pattern of the human voice within 3 dB for frequencies below 500 Hz.

DSP

The DSP needs all of its power pins supplied with the correct voltages. The procedure for verifying this is the same as before. All of the pins were tested and verified to have supply voltages within the tolerance range specified in Table 8.

The DSP needs its RESET pin to be held low for 1±0.5 and then switched to high voltage for proper operation of the DSP. [6] To verify that this was happening, an oscilloscope was connected between the RESET pin on the DSP and ground with its time scale set to one second per division. The DSP was then connected to the power supply block. The oscilloscope was paused after five seconds. The resulting scope image was examined and it was verified that the requirement was met. The output of the oscilloscope is shown in Figure 15.



Figure 15: DSP RESET Verification

The Audio CODEC needs to be configured to properly run upon startup of the DSP. This was originally going to be verified using a logic analyzer. However, it was extremely inconvenient to connect a logic analyzer to the circuit so a different verification method was devised. One oscilloscope probe was connected between SCLK on the codec and ground. A second oscilloscope probe was connected between SDIN on the codec and ground. The DSP and codec were powered on and the signals along the SCLK and SDIN lines were captured. The data was examined and it was verified that the correct signals were being sent to the codec. The output of the oscilloscope for one of the control signals is shown in Figure 16.



Figure 16: Audio Codec Initialization Verification

The final requirement for the DSP is it should output a signal that is $180^{\circ} \pm 18^{\circ}$ out of phase with the input signal. We attempted to verify this requirement by displaying the input signal to the Audio Codec ADC and the output signal from the DAC on two separate channels on the oscilloscope. If the requirement was met, these two signals would be out of phase by $180^{\circ} \pm 18^{\circ}$ for all frequencies between 300 Hz and 3 kHz. However, this was not the case and the requirement could not successfully be met.

Power Supply

For the supply, it was necessary that the 9 V source could supply greater than 6.25 V across a 4 Ω load. When we were still using a battery, we connected a 4 Ω load across the terminals of the battery and measured the current through the resistor on an oscilloscope. We found that the 9 V battery could only supply 5.4 V across such a small load. We then decided to move to a DC power supply. However, we did not want to connect a 4 Ω load across the power supply so we instead verified it by looking at the user guide for the power supply [10].

All of the DC-DC converters need to output the correct voltages for input voltages between 6.25 V and 12 V. This was verified by connecting the inputs of the converters to a DC power supply and measuring the output voltages on a multimeter. The input voltage was varied over the range of 6.25 V to 12 V and it was verified that the output voltage remained within the tolerable ranges for all of the DC-DC converters.

3.2 Failed Verifications

The DSP is the only component within the project design that did not meet all of its requirements. The requirement that was not met is that the output wave must be within 180° ±18° out of phase with the wave read into the DSP. Due to the processing time of the DSP, the outgoing wave had a 3.26 ms delay. This delay caused all signals above 30.67 Hz to have a phase error greater than 18° from being perfectly out of phase with the original signal after inverting the signal in the DSP algorithm. Applying a negative time shift to the processed signal was the original proposed solution to this problem. However, it was found that this only exacerbated the issue causing an even greater delay and greater phase error. This error made it so that only pure sinusoids with fundamental frequencies at integer multiples of 153 Hz could be cancelled while sinusoids at all other frequencies and non-periodic signals exhibited some if not complete constructive interference.

3.3 Tolerance Analysis

The goal of the tolerance analysis is to verify the correct behavior of the filter circuit within the Audio-DSP Interface. This is important for the proper function of our project because the filter removes high frequency noise, improving the signal to noise ratio as well as limiting aliasing in the frequency analysis of the signal in the DSP chip. The filter is a first order RC low pass filter. The correct values of the resistors and capacitors within this circuit needed to be chosen in order to achieve the correct cutoff frequency. If the cutoff frequency was too low, we would have lost important vocal data, while if the cutoff frequency was too high, high frequency noise will not have been attenuated as much as possible. A tolerance analysis of the resistor of the filter was performed in order to find a tolerance range for the resistance that ensures that the filter's cutoff frequency is within ± 10% of the chosen 3 kHz cutoff frequency. In order to verify that the resistor has a resistance value that is within the acceptable tolerance range, the -3 dB cutoff frequency of the filter was measured using a function generator and an oscilloscope. More specifically the function generator was used to input signals of known frequency and amplitude into the filtering circuit. The output was displayed on the oscilloscope and the measure function was be used to acquire the amplitude of the output wave. The frequency at which the amplitude of the output wave was 3 dB less than the amplitude of the input wave was be located. The resistor value was varied to find the range of resistance for which the measured -3 dB cutoff frequency was between 2.7 kHz and 3.3 kHz. The capacitor value in the filter was held constant at 0.047 μ F. Table 3 shows the results of the Tolerance Analysis. As can be seen, in order to be within the specified tolerance, the resistor in the filter needs to be between 1.05 k Ω and 1.31 k Ω .

Resistor Value	3dB cutoff
1.05 kΩ	3.3 kHz
1.08 kΩ	3.2 kHz
1.13 kΩ	3.1 kHz
1.18 kΩ	3.0 kHz
1.21 kΩ	2.9 kHz
1.28 kΩ	2.8 kHz
1.31 kΩ	2.7 kHz

Table 3: F	-ilter Circ	uit Toler	ance An	alysis

4. Costs

Table 4 outlines the cost of parts while Table 5 outlines the cost of labor. The total cost of the project including parts and labor comes to \$36,314.26

4.1 Parts

Table 4: Parts Costs				
Item	Vendor	Quantity	Unit Cost (\$)	Total Cost (\$)
TMS320C6713BGDP225 (DSP)	Digikey	1	\$45.64	\$45.64
25J20RE (20 Ω Resistor)	ECE Store	2	\$0.96	\$1.92
25J500 (500 Ω Resistor)	ECE Store	2	\$0.91	\$1.82
MFR-25FBF-52-1K00 (1 kΩ Resistor)	Digikey	4	\$0.10	\$0.40
MFR-25FBF-52-2K00 (2 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
MCCFR0W8J0332A20 (3.3 kΩ Resistor)	Newark	2	\$0.03	\$0.06
PR01000101801JR500 (5 kΩ Resistor)	Newark	8	\$0.09	\$0.72
MFR-25FBF-52-10K0 (10 kΩ Resistor)	Digikey	6	\$0.10	\$0.60
MFR-25FBF-52-20K0 (20 kΩ Resistor)	Digikey	4	\$0.10	\$0.40
MFR-25FBF-52-24K9 (25 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
MFR-25FBF-52-40K2 (40 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
CFR-25JB-52-100K (100 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
CFR-25JB-52-120K (120 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
MFR-25FBF-52-200K (200 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
CFR-25JB-52-270K (270 kΩ Resistor)	Digikey	2	\$0.10	\$0.20
150D104X9035A2 (0.1 μF Capacitor)	ECE Store	16	\$1.06	\$16.96
C0603C104J4RACTU (1 uF SMD Cap)	Digikey	104	\$0.11	\$11.44
P5305-ND (1 μF Capacitor)	ECE Store	12	\$0.23	\$2.76
P10766-ND (2.2 μF Capacitor)	ECE Store	2	\$0.26	\$0.52
4066PHCT-ND (10 μF Capacitor)	ECE Store	14	\$0.75	\$10.50
VJ1206Y473KXAMT (47 nF Capacitor)	ECE Store	2	\$0.52	\$1.04
2C37Z5U474MO5OB (470 nF Capacitor)	ECE Store	2	\$0.23	\$0.46
1N4004 (1A 400V Diode)	Newark	8	\$0.11	\$0.88
LM317KCSE3 (Voltage Regulator)	Mouser	8	\$0.54	\$4.32
INA333AIDGKR (Mic Amplifier)	Digikey	2	\$4.67	\$9.34
LM4818MX/NOPB (Speaker Amplifier)	Digikey	8	\$0.87	\$6.96
TLV320AIC23BPWR (Audio Codec)	Digikey	2	\$7.78	\$15.56
ADMP404ACEZ-RL7 (Microphone)	Digikey	2	\$3.11	\$6.22
CLS0271M-L152 (Speaker)	Digikey	8	\$4.15	\$33.20
1294 (Keystone 9 Volt PCB Mount	ECE Store	2	\$1.65	
Holder)				\$3.30
PC16049V (Duracell 9V Battery)	ECE Store	4	\$1.46	\$5.84
PCB	ECE Store	4	\$33.00	\$132.00
Tot	\$314.26			

4.2 Labor

Hourly Wage Multiplier Total Employee Hours Worked Joel Godard \$50 per hour 144 hours \$18,000 2.5 Hershed Tilak \$50 per hour 144 hours 2.5 \$18,000 Total \$36,000

Table 5: Labor Costs

5. Conclusion

5.1 Accomplishments

During the course of this project, we successfully built a circuit capable of reading an input signal, processing it, and outputting a new signal. We were also able to meet all of our original design requirements except for one. Although the circuit that we built cannot cancel out a user's voice, it is still capable of blocking out certain frequencies using active noise cancellation..

In addition, we were able to characterize the spatial intensity of the human voice. We found that our results were comparable to those quoted in literature [1]. We were also able to characterize the spatial intensity pattern of multiple speaker arrays and choose a speaker array whose radiation pattern closely resembled that of the typical human voice.

5.2 Experiment Results

The goal of this experiment was to assess the feasibility of using active noise cancellation to globally cancel the voice of a person speaking into a cell phone. It was determined that in its current implementation, this goal is infeasible. Several factors contributed to this conclusion. The primary finding that supports this claim is the inherent design as an open feedback loop. The amplitude and phase of the superposition of the initial and generated signals cannot be characterized without the use of microphones or other transducers in the far field. Furthermore, placing such devices in the far field would defeat the purpose of designing such a system for a portable electronic device. Because of this open loop feedback configuration, the distance from the user's mouth to the microphone must be known a-priori in order to match the amplitude of the incoming and output signals in the far field. Another large obstacle to global active noise cancellation for cell phone users is the high frequency response of speakers that are small enough to fit on portable electronic devices. Accounting for this uneven frequency response adds another level complexity to the overall problem. Finally, the acoustic radiation pattern of the human voice is highly variable from person to person. Although each person does demonstrate greater directivity at higher frequencies, the directivity pattern of each individual is different. Thus the aggregate of many different people appears almost omnidirectional. This makes matching the acoustic radiation pattern of the speaker array to that of a person a unique task that would have to be done separately for every individual. This is a very impractical and unrealistic solution. Each of these major issues must be addressed for any global noise cancellation system to be viable.

5.3 Uncertainties

Being an experimental project, there are a number of uncertainties that remain in the experiment design. Firstly, the anechoic chamber in which acoustic radiation pattern measurements were taken exhibited a large low frequency noise figure. It is unclear if we were able to remove all of the background noise from the anechoic chamber when we were collecting data for our experiment. We know we removed some noise, but we cannot be certain that we removed it all. In addition, we are uncertain about how this noise could have affected our data collection.

Another uncertainty in the experiment is whether or not the small sample size used for collecting data accurately represents the average human voice. We collected data on only five individuals. A more accurate pattern would have been obtained had we collected data on a much larger sample group.

In terms of our circuit, we are uncertain whether or not the DSP on our board would have actually worked had we been able to program it. We were able to verify that all of the power pins were supplied with the correct voltages and that the reset pins were handled appropriately at startup. However, without ever loading code onto the DSP, it is impossible for us to know if it would have worked. On a similar note, we are uncertain whether or not the Audio Codec on our board would have worked. The Audio Codec cannot be tested unless it is initialized by the DSP. Since we could not program our DSP, we could not initialize the Audio Codec on our board. However, we were able to verify that the Audio Codec on the DSK worked, since we were able to program it using the DSP chip on the DSK.

5.4 Ethical considerations

All group members adhered to the IEEE Code of Ethics throughout this project.

Since the project aimed to determine the feasibility of a product that may at some time be commercialized, it was important to adhere to code 1 of the IEEE Code of Ethics: "to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment" [11].

In addition, since the project was research oriented, it specifically demanded strict compliance with codes 3 and 5 of the IEEE Code of Ethics: "to be honest and realistic in stating claims or estimates based on available data" and "to improve the understanding of technology; its appropriate application, and potential consequences" [11].

Finally, since the project required technical work, it was important to adhere with code 7 of the IEEE Code of Ethics: *"to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others"* [11].

Each group member upheld the above standards along with all of the other standards outlined in the IEEE Code of Ethics.

5.5 Future work

Possible future work on the project involves replacing the DSP with a Field Programmable Gate Array (FPGA) and seeing if the reduced latency in our circuit can help us to achieve better results. Another possibility is to use inverting amplifiers instead of the DSP to reduce the latency even further than can be done using an FPGA. Both of these solutions should help to minimize or even eliminate the phase error that caused the failed verification in the DSP block. Other possibilities for future work include implementing the system using only inverting amplifiers to reduce latency as well as developing a closed loop feedback system that would allow for dynamic control of the phase and amplitude of the cancellation signal.

Appendix A Characterization of Acoustic Radiation Patterns

The acoustic radiation patterns of five individuals as well as several speaker arrays were characterized using the following experiment. The entire experiment was performed inside of an anechoic chamber in order to minimize the effects of reflections and outside noise sources. A single reference microphone was placed 1m away from the subject under test at a fixed location. A second microphone was moved in a semicircle around the subject under test in increments of 22.5°. A diagram of the test setup is shown in Figure 17 below. At each position of the mobile microphone, a 20 s audio recording was taken of the subject on both the stationary and mobile microphones. During each recording, if the subject under test was a speaker array, a sine sweep of frequencies ranging from 20 Hz to 20 kHz on a logarithmic interval was played from the speakers. If the test subject was a person, the following passage was read aloud.

"Please call Stella. Ask her to bring these things with her from the store: Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob. We also need a small plastic snake and a big toy frog for the kids. She can scoop these things into three red bags, and we will go meet her Wednesday at the train station."

This passage was designed to contain many of the English "consonants, vowels, and clusters" [12]. This passage took most of the participants approximately 18 s to read aloud fitting within the 20 s recording interval. For each recording performed in the manner described above, the Short Fourier Transform of the signal was taken with 50% overlap over a 200 ms interval. This interval was chosen upon the suggestion of Professor Jont Allen as being the approximate length of the typical English phoneme. The frequency content of each recording from the mobile microphone was then normalized to that of the reference microphone for the same sound sample allowing for radiation patterns in angle ϕ (in the plane parallel to the earth's surface with the direction in front of the person being taken as $\phi = 0$) to be constructed with just two microphones.



Figure 17: Voice Characterization Diagram

In order to verify the validity of the experiment outlined above, the subject under test was rotated 90 degrees so that the reference microphone sat directly in from of the individual or the speaker array. Acoustic radiation patterns were conducted as outlined above and left to right symmetry was checked in order to ensure that the room in which the recordings were taken as well as other sources of error were not dominant in the collected data. Figure 18 and Figure 19 below show the results of this characterization over frequency and angle ϕ from -90° to +90°.



Figure 18: Acoustic Radiation Intensity over Frequency and Angle for Joel Godard



Figure 19: Acoustic Radiation Intensity over Frequency and Angle for CUI Speaker

As can be seen in Figures 18 and 19, the symmetry about $\phi = 0^{\circ}$ substantiates the claim that the signals being received at each microphone location are not affected by reflections in the room. This is made particularly true by noting that the microphones were not placed in the center of the room, therefore reflections would have not acted on the radiation pattern in a symmetrical manner. For the speaker acoustic radiation pattern shown above along with all speaker patterns that were taken in this experiment, the sine sweep that was played through the speaker was repeated three times at each microphone position in order to allow for averaging to reduce the noise floor of the experiment and improve the dynamic range of accurate results.

The frequency response of the average human voice was also used to assess the ability of the experiment to accurately characterize frequency content. Each of the five individuals whose voices were tested in this experiment repeated the passage mentioned above directly into the reference microphone three times from a distance of 1 m. The microphones used were Dayton EMM-6 microphones that came with calibration files allowing for absolute pressure levels seen at the reference microphone to be calculated. All maximum pressure levels of the speech content containing frequencies between 100 Hz to 500 Hz fell within the expected pressure level range of 55 dB SPL to 75 dB SPL. Furthermore, the calculated frequency response of the average human voice exhibits many characteristics similar to that found by [1], an often quoted source in this subject area. A comparison of the results obtained in this experiment and those obtained by [1] is shown below in Figure 20. The data collected in this experiment shown in Figure 20 was converted from dB SPL to Dynes/cm² and scaled by 1m/30cm in order to account for the difference in distance from the microphone to the human mouth used in both experiments. Furthermore, the data presented below is the calculated result after removing low frequency noise that was consistently present throughout the experiment. This noise was sampled in a quiet room three times on two separate occasions separated by several weeks. On both occasions, the frequency content and amplitude of the noise sampled were remarkably similar. For this reason, the noise was simply subtracted from the data after it had been converted to the frequency domain. The frequency content of the noise experienced during the experiment is shown in Figure 21.



Figure 20: Comparison of Average Frequency Response of Human Voice at 30 cm [1]



Figure 21: Frequency Content of Noise Floor

It can be seen in Figure 21 that the noise level below 100 Hz begins to exceed that of most data collected from the subjects under test. Thus data below this frequency should be disregarded.

Having verified that the voice characterization experiment produces reliable frequency content and spatial intensity data, the acoustic radiation patterns of five individuals were measured and averaged in order to find an aggregate human voice radiation pattern. For each person, half of the radiation pattern was measured ranging from $\phi = 0^{\circ}$ to $\phi = -180^{\circ}$ in 22.5°. Based on the symmetry between positive and negative ϕ shown above, this radiation pattern was reflected over the $\phi = 0^{\circ}$ axis in order to provide full -180° to 180° degree radiation pattern. Figures 22-27 show the radiation patterns obtained for each of the individuals tested normalized to the magnitude of the pattern at $\phi = 0^{\circ}$ for 200 Hz, 500 Hz, and 1500 Hz.

The mean radiation pattern of all five individuals was then calculated. Several speaker arrays consisting of four to eight speakers were then constructed and their acoustic radiation patterns were measured. Each speaker array's directivity pattern was then compared to that of the average human voice and the speaker array that had the most similar radiation pattern to that of the average human was selected as the speaker array to be implemented with the final circuit. A side by side comparison of the acoustic radiation pattern of the final selected array and the average human voice is shown in Figure 28. For frequencies below 1 kHz, the two radiation patterns were found to be within 5 dB of each other at all angles of ϕ for which measurements were taken.



Figure 22: Acoustic Radiation Patterns of Erica Daily and Maggie Su at 200 Hz



Figure 23: Acoustic Radiation Patterns of Hershed Tilak, Joel Godard, and John Outwater at 200 Hz



Figure 24: Acoustic Radiation Patterns of Erica Daily and Maggie Su at 500 Hz



Figure 25: Acoustic Radiation Patterns of Hershed Tilak, Joel Godard, and John Outwater at 500 Hz



Figure 26: Acoustic Radiation Patterns of Erica Daily and Maggie Su at 1500 Hz



Figure 27: Acoustic Radiation Patterns of Hershed Tilak, Joel Godard, and John Outwater at 1500 Hz



Figure 28: Acoustic Radiation Patterns Average Human Voice and 8 Element Vertical Linear Speaker Array at 200 Hz, 500 Hz, and 1500 Hz

The array that was constructed and used in the final product of the experiment consisted of eight individual speaker elements. These elements were placed in a linear array stacked vertically. Thus the radiation pattern in ϕ is equivalent to that of a single speaker element, but producing sound pressure levels that are greater than that of a single speaker by 9 dB.

References

- H. K. Dunn, S. D. White, "Statistical Measurements on Conversational Speech", J.A.S.A, vol. 11, pp. 278-288, Jan. 1940.
- [2] ADMP404 Data Sheet, Analog Devices, Norwood, MA, 2012.
- [3] INA333 Data Manual, Texas Instruments Incorporated, Dallas, TX, 2008.
- [4] M. Valente, H. Hosford-Dunn, R. J. Roeser, "Room Acoustics," in *Audiology: Treatment*, 2nd ed. New York, Thieme, 2008, ch. 18, pp. 419.
- [5] *TLV320AIC23 Data Manual,* Texas Instruments Incorporated, Dallas, TX, 2002.
- [6] *TMS320C6713B Floating-Point Digital Signal Processor,* Texas Instruments Incorporated, Dallas, TX, 2006.
- [7] *LM4818 Data Manual*, Texas Instruments Incorporated, Dallas, TX, 2004.
- [8] CLS0271M-L152 Data Manual, CUI Inc., Tualatin, OR, 2007.
- [9] *LM317 Data Manual*, Texas Instruments Incorporated, Dallas, TX, 2012.
- [10] Agilent E3631A Triple Output DC Power Supply, Agilent Technologies, Santa Clara, CA, 2012.
- [11] *IEEE Code of Ethics* [Online]. Available: http://www.ieee.org/about/corporate/governance/p 7-8.html
- [12] Weinberger, Steven. (2013). *Speech Accent Archive*. George Mason University.
- [13] R. Benveniste, B. Sırmaçek, C. Ünsalan, A Quick Start to Texas Instruments TMS 320C6713 DSK, 2010.
- [14] TMS320C6000 CPU & Instruction Set Reference Guide, Texas Instruments Incorporated, Dallas, TX, 2006.
- [15] *TMS320C6201/6701 EVM: TMS320C6000 McBSP toMultimedia Audio Codec Interface,* Texas Instruments Incorporated, Dallas, TX, 2001.

Appendix B Requirement and Verification Table

The following tables list Requirements, Verifications, and Results for the five main components of the circuit. Table 6 covers the Audio Input, Table 7 covers the Audio-DSP Interface, Table 8 covers the DSP, Table 9 covers the Audio Output, and Table 10 covers the Power Supply.

Requirement	Verification	Verification status (Y or N)
Microphone:		
1) The microphone V _{DD} should be supplied with 2.45V±0.1225V with respect to the GND conductor on the microphone	 Check microphone power supply Using DC multimeter, place positive voltage terminal on microphone V_{DD} and negative voltage terminal on the GND of the microphone Verify that 2.45±0.1225V are supplied across these two terminals. 	1) Y (2.379 V)
2) The output of the microphone should carry RMS voltage levels between 0.3694 mV and 634.2 mV when spoken into from a distance of 1cm to 8.54cm at normal voice levels.	 2) Check microphone output voltage Attach oscilloscope probe ground to GND of microphone and the probe's positive voltage terminal to OUTPUT of the microphone. Measuring distance from the microphone to the mouth with a ruler, speak into the microphone from a distance of 5cm at normal speaking levels. Although the use of "normal" is quite subjective, the dynamic range of voice level that will generate an acceptable voltage at the OUTPUT pin of the microphone is quite large (40 dB). Thus, this test should be able to verify whether or not the microphone is working despite differences in what "normal" levels are considered to be. Check oscilloscope to verify that the peak voltage levels are within the range of 0.5224 mV to 896.9 mV (0.3694 mV_{RMS} to 634.2 mV_{RMS}) using peak-hold on the oscilloscope. 	2) Y (27 mV)
Microphone Preamplifier:		
 The microphone should be supplied with 5V±0.25V at its V₊ terminal. 	 Verify microphone amplifier power supply Using DC multimeter, place positive voltage terminal on amplifier V₊ and negative voltage terminal on the V₋ of the amplifier. 	1) Y (4.97 V)

Table 6: Audio Input Requirements and Verifications

		•	Verify that 5±0.25V are supplied across these two terminals.		
2)	The V _{IN+} terminal of the amplifier should have a DC offset of 2.5V±0.125V.	2) • •	Verify V_{IN+} terminal supply of amplifier Using oscilloscope, place positive voltage terminal of probe on amplifier V_{IN+} terminal and negative voltage terminal on the V ₋ of the amplifier. Make sure that the oscilloscope is DC coupled. Verify that there is a 2.5±0.125V dc offset at this terminal	2)	Y (2.548 V)
3)	The V _{IN-} and V _{REF} terminals of the amplifier should supplied with 2.5V±0.125V.	3) • •	Verify microphone amplifier V_{IN} and V_{REF} terminals voltage supplies Using DC multimeter, place positive voltage terminal on amplifier V_{IN} and negative voltage terminal on the V. of the amplifier. Verify that 2.5±0.125V are supplied across these two terminals. Using DC multimeter, place positive voltage terminal on amplifier V_{REF} and negative voltage terminal on the V. of the amplifier. Verify that 2.5±0.125V are supplied across these two terminals on the V. of	3)	Y (2.441 V)
4)	The amplifier should provide a gain of 5 ± 0.25 from its differential inputs V _{IN-} and V _{IN+} to its output V _{OUT} .	4) • •	Verification of amplifier gain Ensure that amplifier is disconnected from microphone by removing the DC blocking capacitor between the microphone OUTPUT and the V_{IN+} terminal of the amplifier. Verify that amplifier is powered correctly by checking all of the above amplifier verifications. Apply function generator positive voltage to V_{IN+} and function generator negative terminal to V Set the function generator to 100mV peak to peak sine wave with no DC offset at 1 kHz. Attach oscilloscope probe across V_{OUT} and V. and verify that the output waveform has a peak to peak voltage of 500±25mV.	4)	Y (4.78 V/V)

Requireme	nt	Verification	V	erification status (Y or N)
Low Pass Anti-Aliasi 1) The low pase aliasing filte provide a co frequency of kHz that is 3 from the in power.	ing Filter: ss anti- er should orner of 3±0.3 3dB down put •	Verifying cutoff frequency of filter Ensure that the low pass filter is disconnected from the microphone amplifier by removing the DC blocking capacitor between the input to the filter and the V_{OUT} terminal of the amplifier. Using the function generator, apply a 2V peak to peak sine wave across the input terminal of the filter from the resistor input to ground. Attach the two terminals of an oscilloscope probe across the output of the filter from the output of the resistor to ground. Use the function generator to sweep from 300 Hz to 4 kHz verifying on the oscilloscope that the point at which the output voltage is 1.41V peak to peak is at 3±0.3 kHz.	1)	Y (3.2 kHz)
Audio CODEC: 1) The Audio (AV _{DD} , BV _{DD} , HPV _{DD} shou supplied a v 3.3±0.165V	CODEC 1) and ild be voltage of	Verification of Audio CODEC power supply at 3.3V Using DC multimeter, place positive measurement voltage terminal on Audio CODEC AV _{DD} terminal and negative voltage measurement terminal on the AGND of the Audio CODEC. Verify that 3.3±0.165V are supplied across these two terminals. Using DC multimeter, place positive measurement voltage terminal on Audio CODEC BV _{DD} terminal and negative voltage measurement terminal on the DGND of the Audio CODEC. Verify that 3.3±0.165V are supplied across these two terminals.	1)	Y (AV _{DD} = 3.293 V) (BV _{DD} = 3.294 V) (HPV _{DD} = 3.294 V)

Table 7: Audio-DSP Interface Requirements and Verifications

	 Using DC multimeter, place positive measurement voltage terminal on Audio CODEC HPV_{DD} terminal and negative voltage measurement terminal on the AGND of the Audio CODEC. Verify that 3.3±0.165V are supplied across these two terminals. 	
 The Audio CODEC DV_{DD} input should be supplied with 1.5±0.075V. 	 2) Verification of Audio CODEC power supply at 1.5V Using DC multimeter, place positive measurement voltage terminal on Audio CODEC ADV_{DD} terminal and negative voltage measurement terminal on the DGND of the Audio CODEC. Verify that 1.5±0.075V are supplied across these two terminals. 	2) Y (1.509 V)
3) The sampling rates of the ADC and DAC should both be 8±0.25 kHz	 3) Verification of 8 kHz sampling rate for ADC and DAC Use DSP chip to send initialization signals to Audio CODEC Using oscilloscope, place positive voltage terminal of probe on LRCIN pin of codec and negative voltage terminal on DGND pin. Verify that the frequency of the waveform on the oscilloscope is at 8±0.25 kHz 	3) Y (8.0000 kHz)

Table 8: DSP Requirements and Verifications

Requirement	Verification	Verification status
		(Y or N)
Digital Signal Processor		
 The DSP must be supplied with 3.3±0.165V to all DV_{DD} inputs. 	 DSP digital power supply verification Using DC multimeter, place positive measurement voltage terminal on each DSP DV_{DD} terminal and negative voltage measurement terminal on the V_{ss} of the DSP. Verify that 3.3±0.165V is supplied across each of these two terminals. 	1) Y (3.294 V)
The DSP must be supplied with	2) DSP core power supply verificationUsing DC multimeter, place positive	2) Y (1.250 V)

	1.25±0.0625V to all CV _{DD} inputs.	n D n tl • V a	measurement voltage terminal on each $PSP CV_{DD}$ terminal and negative voltage measurement terminal on the V_{SS} of the DSP. Verify that 1.25±0.0625V is supplied cross each of these two terminals.		
3)	The RESET Pin of the DSP must be held low on power-up for 1±0.5 seconds and then switched to high voltage for proper operation of the DSP	3) R • U v o o • S o • P tl s v V 1 g	ESET Pin voltage level verification Using oscilloscope, place positive oltage terminal of probe on RESET pin of DSP and negative voltage terminal on V _{ss} pin of the DSP et the oscilloscope to capture data over a 5 second window rower up the circuit and hit pause on the oscilloscope at the end of the 5 econds ferify that the signal is held low for ±0.5 seconds after power-up and then oes high	3)	Y (1 second)
4)	The Audio CODEC must be configured to properly run upon startup of DSP.	 4) C 4) C 6 6 7 8 7 8 4 10 	Configuration verification Using logic analyzer, connect analyzer GND to DGND on the Audio CODEC Connect logic analyzer probe pin 1 to DIN on the Audio CODEC Connect logic analyzer probe pin 2 to CLK on the Audio CODEC Connect logic analyzer probe pin 3 to CS' on the Audio CODEC et analyzer to record and reset the DSP Use the logic analyzer to verify that the ontrols 0x0097, 0x0297, 0x0430, x0630, 0x0814, 0x0A03, 0x0C01, x0E43, 0x100C, and 0x1201 are sent o the Audio CODEC	4)	Y
5)	Assuming correct configuration of Audio CODEC, signal input into ADC should be shifted in phase by 180°± 18° at output of DAC	5) V • P o ir L • P C • C	Verification of DSP program Place positive terminal of one escilloscope test probe on the MIC input to the ADC and another on the ine Out of the DAC. Place the negative terminal of both probes on the AGND of the Audio CODEC. Check that input signal and output	5)	Ν

signal are $180^{\circ} \pm 18^{\circ}$ out of phase on	
oscilloscope.	

Requirement	Verification	Verification status (Y or N)
Audio Amplifier:		(1.2.1.)
 The audio amplific should be supplie with 3.3±0.165V a V_{DD} terminal. 	 Audio amplifier power supply verification Using DC multimeter, place positive measurement voltage terminal on V_{DD} terminal and negative voltage measurement terminal on the GND of the amplifier. Verify that 3.3±0.165V is supplied across these two terminals. 	1) Y (3.284 V)
2) The audio amplifies should have a voltagain from its inpuout put terminal o 0.89±0.05.	 Audio amplifier gain verification Ensure that the amplifier is not connected to the output of the DAC by removing the DC blocking capacitor between the line out of the Audio CODEC and the –IN terminal of the amplifier. Using the function generator, apply a 1 kHz 1.4V peak to peak sine wave across the input of the amplifier from the –IN terminal to GND of the amplifier. Place an oscilloscope probe across the output of the amplifier from the V₀₁ to the V₀₂ terminal of the amplifier. Verify that the output signal is a 1.246±0.07V peak to peak sine wave on the oscilloscope with no distortion. 	2) Y (0.914 V)
Sneakers:		
 Each speaker show be able to be driv at 1 W of input power from 300 H 3 kHz. 	 Verification of individual speaker operation. Using power supply, apply 1.25V peak to peak sine wave at 1 kHz across the input terminals of each speaker. Make sure that sustained sound is emitted from each speaker. 	1) Y
 For global noise cancellation, the spatial audio 	 2) Test of the radiation pattern of the microphone configuration Using the same experimental scheme 	2) Y

Table 9: Audio Output Requirements and Verifications

intensity of the speaker array configuration should match that of the	that was employed in the characterization of the spatial intensity of the human voice, characterize the radiation pattern of the microphone
human voice found in	array.
the characterization	Compare this pattern to those of the
of the spatial	human voice at each frequency. The
intensity of the	closer the patterns match the greater
human voice.	the area of noise cancellation.

Requirement	Verification	Verification status
9V Source:		
 The 9 V source must be able to supply greater than 6.25V across a 4Ω load. 	 Confirmation of 9 V power supply Using multimeter, place low voltage probe on – terminal of source and high voltage probe on the + terminal of the source. Connect the + and – terminals of the source through a 4Ω load. Verify that the multimeter reads a voltage greater than 6.25V. 	1) Y (9 V)
9V-5V DC-DC Converter:		
 The 9V to 5V DC-DC converter must output 5V ±0.05V for all voltage inputs from 6.25V to 12V DC. 	 Verification of 9V to 5V power supply Using DC power supply, apply a positive voltage from the Input to ground. Use multimeter to measure the voltage from the Output terminal to ground of the converter. Vary the input voltage from the DC power supply from 6.25V to 12V and verify that the output voltage is 5V ±0.25V for all input voltages. 	1) Y (4.97 V)
9V-3.3V DC-DC Converter:	· · · · ·	
 The 9V to 3.3V DC-DC converter must output 3.3V ±0.033V for all voltage inputs from 6.25V to 12V DC. 	 Verification of 9V to 3.3V power supply Using DC power supply, apply a positive voltage from the Input to ground. Use multimeter to measure the voltage from the Output terminal to ground of the converter. Vary the input voltage from the DC power supply from 6.25V to 12V and verify that the output voltage is 3.3V 	1) Y (3.293 V)

Table 10: Power Supply Requirements and Verifications

	±0.165V for all input voltages.	
9V-1.5V DC-DC Converter:		
 The 9V to 1.5V DC-DC converter must output 1.5V ±0.015V for all voltage inputs from 6.25V to 12V DC. 	 Verification of 9V to 1.5V power supply Using DC power supply, apply a positive voltage from the Input to ground. Use multimeter to measure the voltage from the Output terminal to ground of the converter. Vary the input voltage from the DC power supply from 6.25V to 12V and verify that the output voltage is 1.5V ±0.0.075V for all input voltages. 	1) Y (1.509 V)
9V-1.25V DC-DC Converter:		
 The 9V to 1.25V DC-DC converter must output 1.25V ±0.0125V for all voltage inputs from 6.25V to 12V DC. 	 Verification of 9V to 1.25V power supply Using DC power supply, apply a positive voltage from the Input to ground. Use multimeter to measure the voltage from the Output terminal to ground of the converter. Vary the input voltage from the DC power supply from 6.25V to 12V and verify that the output voltage is 1.25V ±0.0625V for all input voltages. 	1) Y (1.249 V)

Appendix C Source Code

The following two files were written specifically for this project. The rest of the files needed to run the code are available on the TMS320C6713B DSK Manufacturer's Website.

```
/* Active Noise Cancellation Code
* Joel Godard and Hershed Tilak
* Spring 2013
* main.c file
*/
// Include Files
#include <tistdtypes.h>
#include "dsk6713.h"
#include "dsk6713_aic23.h"
#include <math.h>
#include "stdlib.h"
#include "utility.h"
#include "DSPF_sp_cfftr2_dit.h"
#include "DSPF_sp_icfftr2_dif.h"
#include "sin_tables.h"
// Definitions
#define PI 3.14159265359
#define N 32
// Global Variables
float input_buffer[2*N] _
                   ,0 ,0};
0,0,0};
float w[N] __attribute__((aligned(32)));
void main(void)
{
       // Declare local variables
       DSK6713_AIC23_CodecHandle hCodec;
       Int16 OUT DATA:
       DSK6713_AIC23_Config config = DSK6713_AIC23_DEFAULTCONFIG;
       Uint32 data = 0;
       float old_real_val = 0;
       // Generate Twiddle Factor Array
       tw_genr2fft(w,N);
       bit_rev(w,N>>1);
       // Initialize the board
       DSK6713_init();
       // Start the CODEC
       hCodec = DSK6713_AIC23_openCodec(0, &config);
       // Set microphone as input
       DSK6713_AIC23_rset(hCodec, 0x0004, 0x0015);
       // Set frequency to 8 kHz
       DSK6713_AIC23_setFreq(hCodec, DSK6713_AIC23_FREQ_8KHZ );
       while(1)
       {
              // Read in sample
              while (!DSK6713 AIC23 read(hCodec, &data));
              // Save sample in input buffer
              input buffer[0] = data;
              // read in garbage value so that it is not read on the next iteration
              while (!DSK6713_AIC23_read(hCodec, &data));
              // copy buffer into new buffer to perform fft on
              shifted_buffer[0] = input_buffer[0];
              shifted_buffer[1] = 0;
              shifted_buffer[2] = input_buffer[2];
              shifted_buffer[3] = 0;
shifted_buffer[4] = input_buffer[4];
              shifted_buffer[5] = 0;
```

```
shifted_buffer[6] = input_buffer[6];
shifted_buffer[7] = 0;
shifted_buffer[8] = input_buffer[8];
shifted_buffer[9] = 0;
shifted_buffer[10] = input_buffer[10];
shifted_buffer[11] = 0;
shifted_buffer[12] = input_buffer[12];
shifted buffer[13] = 0;
shifted_buffer[14] = input_buffer[14];
shifted buffer[15] = 0:
shifted_buffer[16] = input_buffer[16];
shifted buffer[17] = 0;
shifted_buffer[18] = input_buffer[18];
shifted_buffer[19] = 0;
shifted_buffer[20] = input_buffer[20];
shifted_buffer[21] = 0;
shifted_buffer[22] = input_buffer[22];
shifted buffer[23] = 0;
shifted_buffer[24] = input_buffer[24];
shifted buffer[25] = 0;
shifted_buffer[26] = input_buffer[26];
shifted_buffer[27] = 0;
shifted_buffer[28] = input_buffer[28];
shifted buffer[29] = 0;
shifted_buffer[30] = input_buffer[30];
shifted_buffer[31] = 0;
shifted_buffer[32] = input_buffer[32];
shifted_buffer[33] = 0;
shifted_buffer[34] = input_buffer[34];
shifted_buffer[35] = 0;
shifted_buffer[36] = input_buffer[36];
shifted_buffer[37] = 0;
shifted_buffer[38] = input_buffer[38];
shifted_buffer[39] = 0;
shifted_buffer[40] = input_buffer[40];
shifted_buffer[41] = 0;
shifted buffer[42] = input buffer[42];
shifted buffer[43] = 0:
shifted_buffer[44] = input_buffer[44];
shifted_buffer[45] = 0;
shifted_buffer[46] = input_buffer[46];
shifted_buffer[47] = 0;
shifted_buffer[48] = input_buffer[48];
shifted_buffer[49] = 0;
shifted_buffer[50] = input_buffer[50];
shifted_buffer[51] = 0;
shifted buffer[52] = input buffer[52];
shifted buffer[53] = 0;
shifted buffer[54] = input buffer[54];
shifted buffer[55] = 0;
shifted_buffer[56] = input_buffer[56];
shifted_buffer[57] = 0;
shifted_buffer[58] = input_buffer[58];
shifted_buffer[59] = 0;
shifted_buffer[60] = input_buffer[60];
shifted_buffer[61] = 0;
shifted_buffer[62] = input_buffer[62];
shifted_buffer[63] = 0;
// take FFT
DSPF_sp_cfftr2_dit(shifted_buffer,w,N);
bit_rev(shifted_buffer,N);
// add sample time delay
old real val = shifted buffer[2];
shifted_buffer[2] = (shifted_buffer[2]*cos_table_delay[2])+(shifted_buffer[3]*sin_table_delay[2]);
shifted_buffer[3] = (shifted_buffer[3]*cos_table_delay[2])-(old_real_val*sin_table_delay[2]);
old_real_val = shifted_buffer[4];
shifted_buffer[4] = (shifted_buffer[4]*cos_table_delay[4])+(shifted_buffer[5]*sin_table_delay[4]);
shifted_buffer[5] = (shifted_buffer[5]*cos_table_delay[4])-(old_real_val*sin_table_delay[4]);
old_real_val = shifted_buffer[6];
shifted_buffer[6] = (shifted_buffer[6]*cos_table_delay[6])+(shifted_buffer[7]*sin_table_delay[6]);
shifted_buffer[7] = (shifted_buffer[7]*cos_table_delay[6])-(old_real_val*sin_table_delay[6]);
old_real_val = shifted_buffer[8];
shifted_buffer[8] = (shifted_buffer[8]*cos_table_delay[8])+(shifted_buffer[9]*sin_table_delay[8]);
shifted_buffer[9] = (shifted_buffer[9]*cos_table_delay[8])-(old_real_val*sin_table_delay[8]);
old real val = shifted buffer[10]:
shifted_buffer[10] = (shifted_buffer[10]*cos_table_delay[10])+(shifted_buffer[11]*sin_table_delay[10]);
```

shifted_buffer[11] = (shifted_buffer[11]*cos_table_delay[10])-(old_real_val*sin_table_delay[10]); old real val = shifted buffer[12]; shifted_buffer[12] = (shifted_buffer[12]*cos_table_delay[12])+(shifted_buffer[13]*sin_table_delay[12]); shifted_buffer[13] = (shifted_buffer[13]*cos_table_delay[12])-(old_real_val*sin_table_delay[12]); old_real_val = shifted_buffer[14]; shifted_buffer[14] = (shifted_buffer[14]*cos_table_delay[14])+(shifted_buffer[15]*sin_table_delay[14]); shifted_buffer[15] = (shifted_buffer[15]*cos_table_delay[14])-(old_real_val*sin_table_delay[14]); old real val = shifted buffer[16]; shifted_buffer[16] = (shifted_buffer[16]*cos_table_delay[16])+(shifted_buffer[17]*sin_table_delay[16]); shifted_buffer[17] = (shifted_buffer[17]*cos_table_delay[16])-(old_real_val*sin_table_delay[16]); old_real_val = shifted_buffer[18]; shifted_buffer[18] = (shifted_buffer[18]*cos_table_delay[18])+(shifted_buffer[19]*sin_table_delay[18]); shifted_buffer[19] = (shifted_buffer[19]*cos_table_delay[18])-(old_real_val*sin_table_delay[18]); old_real_val = shifted_buffer[20]; shifted_buffer[20] = (shifted_buffer[20]*cos_table_delay[20])+(shifted_buffer[21]*sin_table_delay[20]); shifted_buffer[21] = (shifted_buffer[21]*cos_table_delay[20])-(old_real_val*sin_table_delay[20]); old_real_val = shifted_buffer[22]; shifted_buffer[22] = (shifted_buffer[22]*cos_table_delay[22])+(shifted_buffer[23]*sin_table_delay[22]); shifted_buffer[23] = (shifted_buffer[23]*cos_table_delay[22])-(old_real_val*sin_table_delay[22]); old real val = shifted buffer[24]; shifted_buffer[24] = (shifted_buffer[24]*cos_table_delay[24])+(shifted_buffer[25]*sin_table_delay[24]); shifted_buffer[25] = (shifted_buffer[25]*cos_table_delay[24])-(old_real_val*sin_table_delay[24]); old_real_val = shifted_buffer[26]; shifted_buffer[26] = (shifted_buffer[26]*cos_table_delay[26])+(shifted_buffer[27]*sin_table_delay[26]); shifted_buffer[27] = (shifted_buffer[27]*cos_table_delay[26])-(old_real_val*sin_table_delay[26]); old_real_val = shifted_buffer[28]; shifted_buffer[28] = (shifted_buffer[28]*cos_table_delay[28])+(shifted_buffer[29]*sin_table_delay[28]); shifted_buffer[29] = (shifted_buffer[29]*cos_table_delay[28])-(old_real_val*sin_table_delay[28]); old_real_val = shifted_buffer[30]; shifted_buffer[30] = (shifted_buffer[30]*cos_table_delay[30])+(shifted_buffer[31]*sin_table_delay[30]); shifted_buffer[31] = (shifted_buffer[31]*cos_table_delay[30])-(old_real_val*sin_table_delay[30]); old_real_val = shifted_buffer[32]; shifted_buffer[32] = (shifted_buffer[32]*cos_table_delay[32])+(shifted_buffer[33]*sin_table_delay[32]); shifted_buffer[33] = (shifted_buffer[33]*cos_table_delay[32])-(old_real_val*sin_table_delay[32]); old_real_val = shifted_buffer[34]; shifted_buffer[34] = (shifted_buffer[34]*cos_table_delay[34])+(shifted_buffer[35]*sin_table_delay[34]); shifted_buffer[35] = (shifted_buffer[35]*cos_table_delay[34])-(old_real_val*sin_table_delay[34]); old real val = shifted buffer[36]; shifted_buffer[36] = (shifted_buffer[36]*cos_table_delay[36])+(shifted_buffer[37]*sin_table_delay[36]); shifted_buffer[37] = (shifted_buffer[37]*cos_table_delay[36])-(old_real_val*sin_table_delay[36]); old real val = shifted buffer[38]; shifted_buffer[38] = (shifted_buffer[38]*cos_table_delay[38])+(shifted_buffer[39]*sin_table_delay[38]); shifted_buffer[39] = (shifted_buffer[39]*cos_table_delay[38])-(old_real_val*sin_table_delay[38]); old_real_val = shifted_buffer[40]; shifted_buffer[40] = (shifted_buffer[40]*cos_table_delay[40])+(shifted_buffer[41]*sin_table_delay[40]); shifted_buffer[41] = (shifted_buffer[41]*cos_table_delay[40])-(old_real_val*sin_table_delay[40]); old real val = shifted buffer[42]; shifted_buffer[42] = (shifted_buffer[42]*cos_table_delay[42])+(shifted_buffer[43]*sin_table_delay[42]); shifted_buffer[43] = (shifted_buffer[43]*cos_table_delay[42])-(old_real_val*sin_table_delay[42]); old_real_val = shifted_buffer[44]; shifted_buffer[44] = (shifted_buffer[44]*cos_table_delay[44])+(shifted_buffer[45]*sin_table_delay[44]); shifted_buffer[45] = (shifted_buffer[45]*cos_table_delay[44])-(old_real_val*sin_table_delay[44]); old_real_val = shifted_buffer[46]; shifted_buffer[46] = (shifted_buffer[46]*cos_table_delay[46])+(shifted_buffer[47]*sin_table_delay[46]); shifted_buffer[47] = (shifted_buffer[47]*cos_table_delay[46])-(old_real_val*sin_table_delay[46]); old_real_val = shifted_buffer[48]; shifted_buffer[48] = (shifted_buffer[48]*cos_table_delay[48])+(shifted_buffer[49]*sin_table_delay[48]); shifted_buffer[49] = (shifted_buffer[49]*cos_table_delay[48])-(old_real_val*sin_table_delay[48]); old real val = shifted buffer[50]; shifted_buffer[50] = (shifted_buffer[50]*cos_table_delay[50])+(shifted_buffer[51]*sin_table_delay[50]); shifted_buffer[51] = (shifted_buffer[51]*cos_table_delay[50])-(old_real_val*sin_table_delay[50]); old_real_val = shifted_buffer[52]; shifted_buffer[52] = (shifted_buffer[52]*cos_table_delay[52])+(shifted_buffer[53]*sin_table_delay[52]); shifted_buffer[53] = (shifted_buffer[53]*cos_table_delay[52])-(old_real_val*sin_table_delay[52]); old real val = shifted buffer[54]; shifted_buffer[54] = (shifted_buffer[54]*cos_table_delay[54])+(shifted_buffer[55]*sin_table_delay[54]); shifted buffer[55] = (shifted buffer[55]*cos table delay[54])-(old real val*sin table delay[54]); old_real_val = shifted_buffer[56]; shifted_buffer[56] = (shifted_buffer[56]*cos_table_delay[56])+(shifted_buffer[57]*sin_table_delay[56]); shifted_buffer[57] = (shifted_buffer[57]*cos_table_delay[56])-(old_real_val*sin_table_delay[56]); old_real_val = shifted_buffer[58]; shifted_buffer[58] = (shifted_buffer[58]*cos_table_delay[58])+(shifted_buffer[59]*sin_table_delay[58]); shifted_buffer[59] = (shifted_buffer[59]*cos_table_delay[58])-(old_real_val*sin_table_delay[58]); old_real_val = shifted_buffer[60]; shifted_buffer[60] = (shifted_buffer[60]*cos_table_delay[60])+(shifted_buffer[61]*sin_table_delay[60]); shifted_buffer[61] = (shifted_buffer[61]*cos_table_delay[60])-(old_real_val*sin_table_delay[60]); old real val = shifted buffer[62]: shifted_buffer[62] = (shifted_buffer[62]*cos_table_delay[62])+(shifted_buffer[63]*sin_table_delay[62]);

	<pre>shifted_buffer[63] = (shifted_buffer[63]*cos_table_delay[62])-(old_real_val*sin_table_delay[62]);</pre>
	// take IFFT
	<pre>bit_rev(shifted_buffer,N);</pre>
	DSPF_sp_icfftr2_dif(shifted_buffer,w,N);
	// drab output
	// Shift buffer
	input_buffer[62] = input_buffer[60];
	input_buffer[60] = input_buffer[58];
	input_buffer[58] = input_buffer[56];
	input_buffer[56] = input_buffer[54];
	input_buffer[54] = input_buffer[52];
	<pre>input_buffer[52] = input_buffer[50];</pre>
	input_bufter[50] = input_bufter[48];
	input_putter[48] = input_putter[46];
	input_buffer[46] = input_buffer[44];
	input buffer[43] - input buffer[42],
	input buffer[42] = input buffer[38]:
	input buffer[38] = input buffer[36]:
	input buffer[36] = input buffer[34];
	input_buffer[34] = input_buffer[32];
	input_buffer[32] = input_buffer[30];
	input_buffer[30] = input_buffer[28];
	input_buffer[28] = input_buffer[26];
	input_buffer[26] = input_buffer[24];
	input_buffer[24] = input_buffer[22];
	<pre>input_buffer[22] = input_buffer[20];</pre>
	input_bufter[20] = input_bufter[18];
	input_ourrer[18] = input_ourrer[16];
	input buffer[10] = Input buffer[14];
	input buffer[14] = input buffer[14].
	input buffer[10] = input buffer[8]:
	input buffer[8] = input buffer[6]:
	input buffer[6] = input buffer[4];
	input_buffer[4] = input_buffer[2];
	input_buffer[2] = input_buffer[0];
	// Output Sample
	// Vocput Sample
3	
}	
·	

* sin_tables.h * sine and cosine tables needed for 26 sample shift in DSP algorithm */ #ifndef SIN_TABLES_H_ #define SIN_TABLES_H_ float cos_table_delay[128] = {1, 0.290284677254462, -0.831469612302545, -0.773010453362737, 0.382683432365090, 0.995184726672197, 0.195090322016129, -0.881921264348355, -0.707106781186547, 0.471396736825997, 0.980785280403231, 0.0980171403295626, -0.923879532511286, -0.634393284163646, 0.555570233019602, 0.956940335732209, -9.80336419954471e-16, -0.956940335732209, -0.555570233019603, 0.634393284163644, 0.923879532511288, -0.0980171403295610, -0.980785280403230, 0.471396736825999, 0.707106781186546, 0.881921264348355, -0.195090322016128, -0.995184726672197, -0.382683432365091, 0.773010453362735, 0.831469612302547, -0.290284677254458, -1, -0.290284677254461, 0.831469612302545, 0.773010453362737, 0.382683432365089, -0.995184726672197, -0.195090322016131, 0.881921264348353, 0.707106781186551, -0.471396736825993, -0.980785280403230, -0.0980171403295606, 0.923879532511284, 0.634393284163647, -0.555570233019600, -0.956940335732208, -4.16441809773759e-15, 0.956940335732207, 0.555570233019601, -0.634393284163640, -0.923879532511287, 0.0980171403295665, $0.980785280403230, \ 0.471396736825994, \ -0.707106781186545, \ -0.881921264348361, \ 0.195090322016123, \ 0.995184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921264348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.881921266348361, \ -0.195090322016123, \ -0.95184726672197, \ -0.707106781186545, \ -0.707106781186545, \ -0.707106781186545, \ -0.881921266781, \ -0.707106781186545, \ -0.881921266781, \ -0.707106781186545, \ -0.881921266781, \ -0.707106781186545, \ -0.881921266781, \ -0.707106781186545, \ -0.881921266781, \ -0.707106781186545, \ -0.881921266781, \ -0.707107106781, \ -0.7071067$ 0.382683432365096, -0.773010453362737, -0.831469612302550, 0.290284677254460, 1, 0.290284677254466, -0.831469612302547, -0.773010453362741, 0.382683432365090, 0.995184726672198, 0.195090322016129, -0.881921264348358, -0.707106781186550, 0.471396736825988, 0.980785280403231, 0.0980171403295587, -0.923879532511284, -0.634393284163645, 0.555570233019596, 0.956940335732209, 9.30917261542965e-15, -0.956940335732208, -0.555570233019611, 0.634393284163653, 0.923879532511286, 0.0980171403295543, -0.980785280403230, -0.471396736826005, 0.707106781186536, 0.881921264348353, -0.195090322016125, -0.995184726672196, -0.382683432365094, 0.773010453362729, 0.831469612302541, -0.290284677254462, -1, -0.290284677254464, 0.831469612302540, 0.773010453362730, -0.382683432365092, -0.995184726672197, -0.195090322016141, 0.881921264348345, 0.707106781186548, -0.471396736825990, -0.980785280403228, -0.0980171403295567, 0.923879532511285, 0.634393284163655, 0.555570233019609, -0.956940335732209, -7.34849977552071e-15, 0.956940335732204, 0.555570233019622, -0.634393284163643, -0.923879532511291, 0.0980171403295704, 0.980785280403231, 0.471396736826003, -0.707106781186538, -0.881921264348352, 0.195090322016127, 0.995184726672196, 0.382683432365106, -0.773010453362721, -0.831469612302548, 0.290284677254450};

float sin_table_delay[128] = {0, -0.956940335732209, -0.555570233019602, 0.634393284163645, 0.923879532511287, 0.0980171403295600, -0.980785280403230, -0.471396736825998, 0.707106781186548, 0.881921264348355, -0.195090322016127, 0.995184726672197, -0.382683432365091, 0.773010453362737, 0.831469612302546, -0.290284677254461, -1, -0.290284677254462,
0.831469612302545, 0.773010453362738, -0.382683432365088, -0.995184726672197, -0.195090322016132, 0.881921264348354,
0.707106781186549, -0.471396736825998, -0.980785280403230, -0.0980171403295616, 0.923879532511286, 0.634393284163648, 0.555570233019599, -0.956940335732210, 1.96067283990894e-15, 0.956940335732209, 0.555570233019602, -0.634393284163645, 0.923879532511287, 0.0980171403295584, 0.980785280403230, 0.471396736826001, -0.707106781186544, -0.881921264348358,
0.195090322016129, 0.995184726672197, 0.382683432365097, -0.773010453362736, -0.831469612302547, 0.290284677254466, 1,
0.290284677254467, -0.831469612302546, -0.773010453362741, 0.382683432365088, 0.995184726672196, 0.195090322016130, 0.881921264348357, -0.707106781186550, 0.471396736825987, 0.980785280403232, 0.0980171403295597, -0.923879532511284, 0.634393284163641, 0.923879532511287, -0.0980171403295533, -0.980785280403232, 0.0980171403295597, -0.923879532511284, 0.881921264348360, -0.155090322016124, -0.995184726672197, -0.382683432365095, 0.773010453362737, 0.831469612302550, 0.290284677254461, -1, -0.290284677254465, 0.831469612302559, -0.471396736825093, 0.773010453362731, -0.382683432365091, -0.995184726672198, 0.950890322016128, 0.881921264348351, 0.707106781186559, -0.471396736826002, -0.980785280403231, -0.980171403295578, 0.950940335732208, -0.555570233019609, -0.95184726672197, 0.382683432365094, -0.773010453362779, 0.83146961230254, 0.797106781186547, -0.881921264348359, 0.19509322016140, 0.995184726672197, 0.382683432365094, -0.773010453362729, 0.831469612302541, -0.92084677254477, -0.831469612302532, -0.773010453362739, 0.37804833224163664, 0.995184726672196, 0.

#endif /* SIN_TABLES_H_ */

Appendix D Schematics

Included in this section are schematics for all of the components discussed in Section 2 of this report. Figure 29 is a top level schematic of the circuit. Figure 30 is a schematic for the microphone. Figure 31 is a schematic of the preamplifier. Figure 32 is a schematic of the low pass anti-aliasing filter. Figure 33 is a schematic of the Audio Codec. Figure 34 is a schematic of the DSP. Figure 35 is a schematic of the JTAG Header, which is used to program the DSP. Figure 36 is a schematic of the speaker amplifier and speakers. Figure 37 is a schematic of the power supply.



Figure 29: Overall Schematic



Figure 30: Microphone Schematic

Preamplifier





Low Pass Filter



Figure 32: Low Pass Filter Schematic



Figure 33: Audio Codec Schematic



Figure 34: DSP Schematic





JTAG Header



Figure 35: JTAG Header (used to program DSP) Schematic







