The Resistance – Digital Theremin

Landon Greene, Michael Bliss, and Elmo Alexander,

December 16, 2019

Capstone Design ECE 4440 / ECE4991

Signatures

Statement of work:

Landon served as the team's main hardware lead. In the beginning of the semester, he spent a lot of time researching various components and combing through their datasheets to find specific parts that would work within the bounds of the project, as well as creating unique footprints and packages for said parts. He also created and maintained all schematics used throughout the project in Multisim, including the resistor and capacitor network needed for the voltage regulators, power amplifier, and Butterworth filters used to filter the output of the DAC among others. In addition, he designed and populated all PCB's used in the project and was responsible for making any changes to the PCB connections (i.e. cutting any obsolete connections and replacing them by soldering a jumper wire to the correct pins). Finally, he took the lead on testing the various subsystems of the project, including the power supplies, sound actuation subsystem, and summing amplifier.

Landon also took a secondary role in the software test process. He assisted Michael in troubleshooting and debugging faulty code, especially in regard to the ADC/DAC software. In addition, he assisted both Michael and Elmo in the final assembly of the product.

As the team's software lead, Michael's main responsibilities this semester included the code design, methodology, and testing of the MSP430. This process was handled incrementally, starting with the code for reading inputs from the MSP430's ADC, then the code for producing outputs via a DAC, and eventually combining it all together into one program. The testing consisted largely of timing analysis to make sure that with the inclusion of the ADC, the DAC would still be able to produce the desired frequencies. In order to produce a sinusoidal output using the DAC, Michael calculated Timer A capture compare register values along with table lookup values, which are described later in the Microcontroller Subsystem section of this report.

Michael also assumed a secondary role in the circuit design/testing process. He worked closely with Landon in the design and testing of the voltage dividers (for the IR sensors), summing amplifier, and power amplifier. Additionally, Michael procured the materials for the final enclosure, built it, and painted it.

Elmo was responsible for the physical design of the encasement. This involved various tasks from learning Softworks 3D CAD modeling to learning about 3D printers to physical assembly. The design and research conducted during this continuous effort helped decide on the final encasement as well as how to display the sensors and contain the wires within the box in an aesthetically pleasing manner.

Table of Contents

Table of Figures

Table of Tables

Table of Equations

Abstract

The Theremin is one of the world's first electronic instruments and it is unique in that it is played without making physical contact. Traditionally, a Theremin is fully analog and consists of two antennae. To play the Theremin, a user holds their hands in space near these antennae. The proximity between the user's hand and the vertical antenna determines the pitch of the note being played, and the proximity between the user's hand and the horizontal antenna determines the volume. As a result, it can only play one note at a time. This project involved the design and production of a "digital" Theremin using infrared (IR) proximity sensors along with embedded technology. The use of two sensors to determine two distinct pitches converts this Theremin from a monophonic to a polyphonic instrument, meaning the user can play two notes simultaneously. Additionally, for increased ease of use, the frequency spectrum of possible output is discretized into the nine notes of the D flat major scale.

Background

We chose to build a Theremin for a few different reasons. First, the Theremin is a relatively unknown instrument with a rich and interesting background. It was originally developed by physicist Leon Theremin in 1920 while doing research on proximity sensors for the Russian government. His first prototype was a great success in the eyes of Vladimir Lenin, so much so that the Russian leader sent Theremin all throughout Europe to demonstrate his incredible invention. Eventually, after a couple of years in Europe, Theremin found himself in America, where he eventually filed a patent for his design in 1928. Despite this development, the instrument did not initially see great commercial success as fewer than 500 units were sold. However, much later in the life of the instrument, Robert Moog, best known for his electronic synthesizers, began producing Theremin's invention. With Moog's name behind it, the Theremin started to garner a cult following and to this day its eerie tune can now be heard in a large selection of horror and sci-fi films [1].

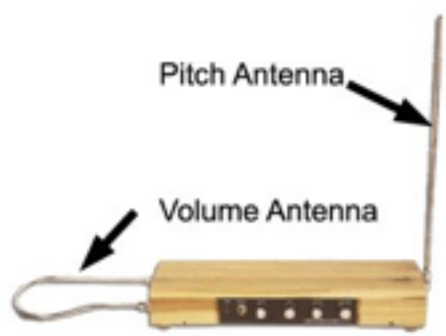

Figure 1: Classic Fully-Analog Theremin

Additionally, this project incorporated a wide range of Computer and Electrical engineering topics so it has served as a good encapsulation of our time here at the UVA Engineering school. Multiple facets of our previous coursework can be found in this project - there are embedded components, PCB design and production, circuit design, and more. All members of our group have experience with these elements from our previous coursework, but certain members have more interest/experience in certain parts of the design/production process. The different levels of interest/experience helped divide the workload in a manner that allowed each group member to focus on what he was interested in and what he possessed expertise in.

One similar project that was done in the past was a fully-analog Theremin. The project was done by ECE students at MIT and their spin on the classic Theremin was to add LED indicators that illuminate when certain frequencies are played. The problem that they saw with the classic Theremin was that it was hard to learn because there are no physical reference points for the user to know which note they are playing. In order to accomplish this, they created 9 Sallen-Key bandpass filters with center frequencies corresponding to the notes C1, C2, C3, C4, A4, C5, C6, C7, and C8. Each filter was then connected to an LED strip, and the number of individual LEDs in the strip that would illuminate was proportional to the amplitude of the signal outputted in that given frequency range [3]. Another similar project was done in the past by ECE students at Cornell University. This project involved creating a Theremin using infrared proximity sensors. This project much more closely resembles the project that we executed. The students used two IR sensors and the ATMEGA1284P microcontroller to create a monophonic Theremin device. They also included a push-button that allows the user to toggle between four different instrument voices, providing interesting effects/distortions to the sound created. All of this was packaged inside of a professional-looking enclosure that mimics the shape and design of a classic Theremin device [4].

Our version of the Theremin is different than past versions in a couple of ways. One major difference is the means through which we determine the distance between the user's hands and the device. The classic Theremin effectively creates a variable capacitor between the user's hands and the antennae. The measured capacitance determines what note will be played and at what volume. Instead of using these antennae, we use infrared proximity sensors. We use two infrared proximity sensors to determine two pitches in order to allow the user to play multiple notes at once (polyphonic). The traditional Theremin is monophonic due to the nature of the antenna it uses. The pitch antenna lies along the z-axis and its field radiates outwards along the x and y axes. So, if a second antenna were to be placed with the same orientation (along the z-axis), their fields would interfere with each other resulting in an inability to accurately detect the position of the performer's hand. This same concept is why the amplitude antenna is looped and lies in the x-y plane. This specific geometry results in a field that radiates along the z-axis which, when placed the proper distance away, will not interfere with the pitch antenna's field. This is why we plan on using IR sensors. The much narrower field of "vision" of the sensors will allow the performer to angle their hand to produce two distinct notes.

Constraints

Design Constraints

For our project, we elected to use the MSP430G2553 [5] microprocessor as it was determined that both the clock speed and the ADC10 sampling speed were fast enough to satisfy timing requirements. It possesses a 16 MHz clock and 16 KB of flash memory, which were more than enough to satisfy our space and timing requirements. The main reason that a 16 MHz clock was used was to generate precise periodic interrupts so that the output sinusoidal frequencies produced by the DAC were as close as possible to the desired frequency values. Additionally, the majority of the utilized flash memory was dedicated to storing the lookup tables that contained values to send to the DAC.

Additionally, the MSP430G2553 contains an internal 10-bit 8-channel analog-to-digital converter (ADC) which was used to sample the voltage outputs generated by the three IR sensors. One major constraint when using this microcontroller was that the internal reference voltage it can produce for sampling is a maximum of 2.5V. In order to accurately sample the expected incoming voltages of around 3V maximum, we had to send the sensor outputs through voltage dividers. However, this issue is adequately addressed and explained in the Technical Description of the project.

For schematic and board design, we utilized Multisim and Ultiboard [6], which are provided via the University for free. For interfacing with/debugging the MSP430, we used Code Composer Studio [7], which is another free, robust software provided by Texas Instruments. And finally, for 3D modeling, we used SolidWorks [8], which can also be obtained for free through the University.

Over the course of the semester, we were delayed by unforeseen manufacturing limitations and parts/PCB availability. Right off the bat, we ran into issues getting our parts and boards sent in properly. We did not receive our first order of parts until well into the semester, and we waited even longer before we obtained our first board. Part of this can be contributed to the fact that we were not as organized as we should have been when submitting parts orders and board send outs. However, part of it can also be attributed to the fact that there were only a limited number of both parts and board send outs. If we were developing this project independently and were not constrained by the limited send out dates, we would have had much less trouble getting the materials we needed on time.

Economic and Cost Constraints
As far as electronic instruments go, this was a relatively inexpensive project. The vast majority of parts that we used were very cheap and easy to procure. Only the DAC, at $18.42 per chip, and the IR sensors, at $11.18 apiece, cost more than $3. And this price could be further reduced in the future if we were to use a two-channel DAC instead of a four-channel DAC as the extra overhead is unnecessary. In addition, the use of a cheap microprocessor like the MSP430G2553 which is only $2.70 on Digi-Key makes this an extremely scalable design (see Costs section for a more in-depth analysis).

External Standards

IPC-6012 Standard
One engineering standard that was taken into account was IPC-6012 [9]. This standard deals with the Qualification and Performance Specification for Rigid Printed Boards. Specifically, our board is considered a "Class 1" board as it is not designed for an "extended life" and is a general-purpose PCB. IPC-6012 contains rules regarding conductor width and spacing, soldermask width, via size, etc. Our board layout was indirectly influenced by these standards, as when it was run through the standard DRC on Ultiboard, it was being checked with IPC-6012.

Tools Employed
There were a number of tools used throughout this project. In terms of hardware and circuit design, the two major tools used were Multisim and Ultiboard [6]. Multisim was used to design the project and create packages for various components, as well as to perform simulations which were compared to experimental results while testing. Ultiboard was used to design the physical layout of the boards for our design, as well as to create footprints for various parts. We have been

using these tools since we began studies under the ECE curriculum so there was little learning required, although their extended use has certainly improved our proficiency.

In terms of software for programming, the main tool used was Code Composer Studio (CCS) [7], in order to program and debug the MSP430 using the C programming language. Additionally, Git/GitHub [10] were used for version control management and remote file storage.
Finally, designing an encasement required the use of SolidWorks [8], a CAD software with which the team had no previous experience. Using SolidWorks, we were able to design prototypes for the device's encasement and convert these models into a format that could be printed by a MakerBot Replicator+ 3D printer [11].

Ethical, Social, and Economic Concerns

Environmental Impact

There was negligible environmental impact for this project and very little energy consumption associated with our device. The only sustainability concerns are the raw materials that needed to be cultivated in order to produce the components that we utilized, as well as the environmental overhead induced in the process of shipping the needed parts to Charlottesville.

Sustainability

Since the Theremin can be effectively powered by a single 9V battery and doesn't require much material to produce the encasement, this project is fairly sustainable. The indirect carbon emissions from producing the Theremin are negligible and result mostly from using the 3D printers to print the encasement and the power that is stored within the battery. Therefore, this model is incredibly sustainable.

Health and Safety

The only moderate health and safety concern for this project is to ensure that volume does not exceed what ASHA considers to be dangerous bursts of sound (approximately 91 dBA) [12]. This was taken into consideration in the process of testing the power amplifier. In order to ensure that no dangerous bursts of sound were to occur, we first configured the power amplifier to receive input from a function generator and to output it to a speaker. We slowly increased the amplitude of the input signal until determining that the maximum input signal amplitude that would generate a non-distorted audio output was approximately 0.9V. Using Decibel X [13], a free iOS application developed by SkyPaw, we were able to determine that the maximum output sound was well below 91 dBA, thus satisfying this safety constraint.

Manufacturability

The physical components of this project were fairly easy to manufacture, as there are no moving parts. The main area of concern is the manufacturing of the device's encasing, as the final encasement was custom-made using a wood-like material. If the device were to undergo mass production, an extensive 3D model would likely need to be made for reproducibility reasons. However, the use of a different material could affect the acoustics of the output sound, so further testing would need to be done before mass-producing a 3D printed enclosure.

Ethical Issues
We encountered no ethical concerns regarding the design and production of our digitally-implemented Theremin.

Intellectual Property Issues
US patent number US10395630B1 [14] patented a touchless effects pedal that adjusted the audio/visual signal processing circuit so that a musician could simultaneously adjust the effects pedal while playing the instrument. Although our device does adjust the output sound based on a touchless sensor, the adjustments change the frequency of the note being played and does not change the shape of the waveform itself. This leads us to believe that, while this patent exists, we would not be infringing upon it.

US patent number US10199022B1 [15] patented a Theremin with a touchless, modifiable power system that changes the power supply to the actual Theremin. One of the examples provided for power supply in this patent is the use of a 9V battery, which our device uses. The patent uses the phrase "alterations substantially beyond the alterations typically produced by the signal processing circuit" to describe its patentable idea which we believe could also apply to our digital Theremin with dual pitch sensors. Depending upon the interpretation of this vague description, we believe that our device could potentially be infringing upon this patent.

US patent number US6066790 [16] patented a display and measuring system that isolates frequencies and is able to display them simultaneously or individually. This patent utilizes a local transducer with the display system. This idea is similar to our original plan of isolating different frequencies for the output and displaying the notes being played on LED matrices. However, our idea is different in that our device does not actually measure any frequencies, it produces them. Therefore, we do not believe that we would be infringing upon this patent.

Detailed Technical Description of Project
There are a number of primary subsystems in this design. In this section, they will be discussed individually and in great detail.

Power Subsystem
The first main subsystem of our design is the Power subsystem. This is responsible for supplying power to all of the active components present in our design. These active components either ran on a 3.3V or 5V supply, and as a result, we needed two separate voltage regulators to satisfy these requirements.

The first voltage regulator used was the BA3259HFP-TR chip made by Rohm Semiconductors (figure 2) [17]. With an input of 9V, it outputs a fixed 3.3V signal with a maximum current output of 1A. This regulator was in charge of powering the MSP430 microprocessor and requires two 4.7uF bypass capacitors in order to eliminate any unwanted voltage drops on the power supply by storing charge which is released in the event of a voltage spike.

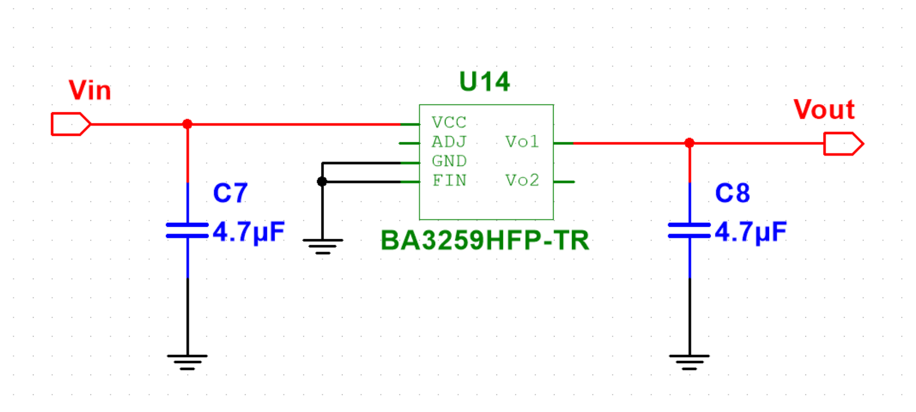

Figure 2: 3.3V Regulator Schematic

The second voltage regulator used was the L6932D1.2TR chip made by STMicroelectronics [18]. With an input of up to 14V, it can be configured to output a max voltage of 5V using an external system of resistors, with a maximum current output of 2A. The following equation was used to set the output of the regulator:

$$V_{out} = \frac{1.2}{R_{15}} * (R_{15} + R_{16})$$

Equation 1: Resistor Calculation for 5V Supply

In this equation, the component values correspond to those shown below in figure 3. Specifically, R16 was chosen to be 22kΩ while R15 was chosen to be 6.8kΩ. In addition, two bypass capacitors were required for the same reasons as the first regulator. The specific values, 1uF and 47uF were provided by the device's datasheet.

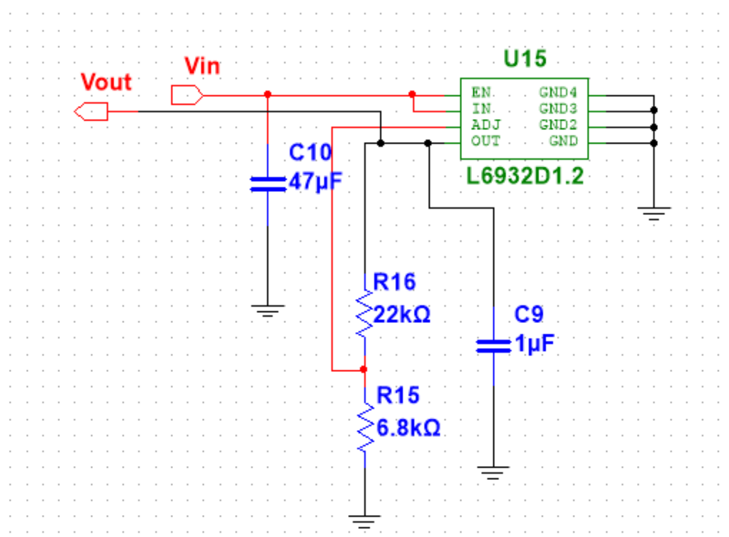

Figure 3: 5V Regulator Schematic

This regulator powered the sensors, the DAC, multiple Op-Amps, the Power Amplifier, the MAX LED driver ICs and LED Matrices. Due to the number of components, we needed a regulator with enough current to power all of these devices simultaneously. In hindsight, the 2 amps provided by this regulator were slightly overkill due to us not actually using the LED matrices in the final product. However, it is better to be safe than sorry and this regulator served us well.

Sensor Subsystem

The next subsystem of our design dealt with the infrared sensors used to measure the position of the user's hand(s). Specifically, we used GP2Y0A41SK0F sensors, produced by Sharp/Socle Technology [19]. These proximity sensors have a range of 3 - 30 cm and take an input voltage of 5V. They output a voltage that is inversely proportional to the distance of the measured object. Figure 4 below, taken from the sensors' datasheet illustrates the expected voltage output characteristic.

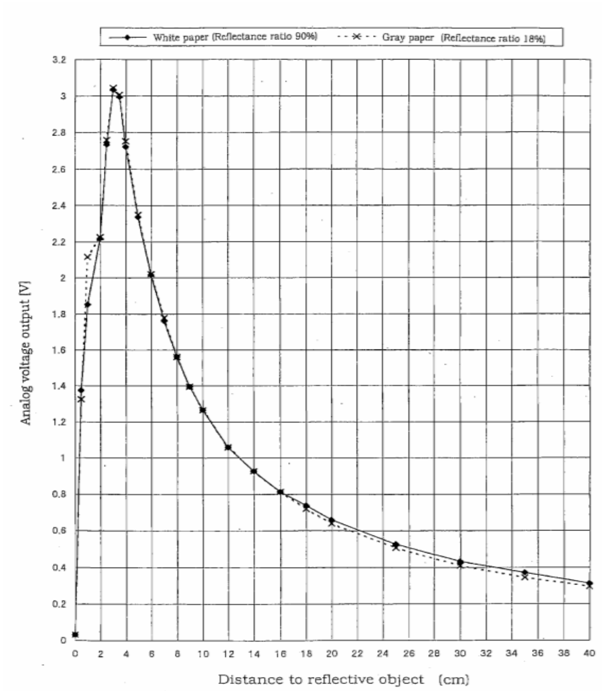

Figure 4: Sensor Voltage Characteristic from Datasheet

There were a couple of issues that we ran into. The first was that we used the internal ADC reference of our MSP430 microprocessor which has a max reference voltage of only 2.5V (see next section for more information regarding the MSP430). As shown in the figure above, the sensors can potentially output a maximum voltage of approximately 3.1V. So, in order to have the MSP's ADC function properly, we needed to drop the sensors' outputs accordingly so that they would have a maximum at around 2.5V. This was accomplished using simple voltage dividers comprised of two resistors.

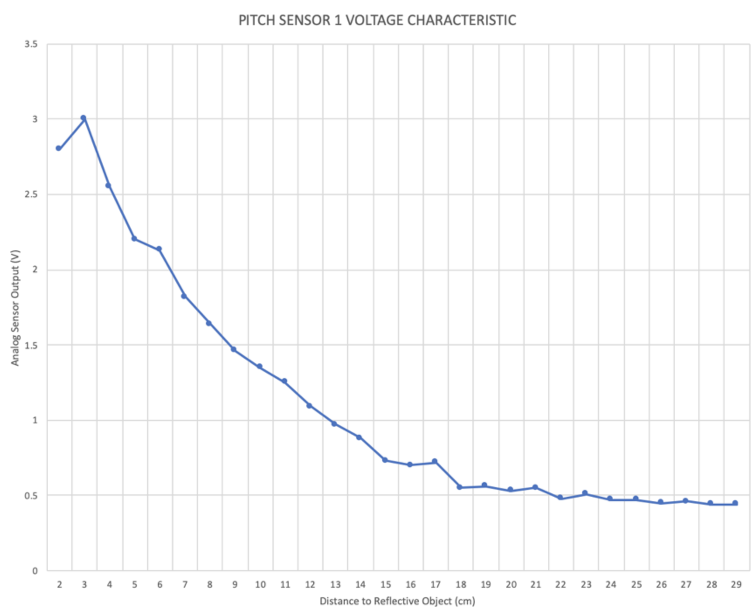

Figure 5: Sensor Voltage Characteristic from Measurement

The other issue we ran into was that the IR sensors we chose to use for this project are not incredibly accurate or precise. A voltage characteristic made from measured values of the sensor we called “Pitch 1” can be seen in figure 5 and it is noticeably different from that which is specified in the datasheet (figure 4). Once we began testing the sensors individually, we found that they all had variations in their respective outputs at similar distances. For example, while one sensor might have output 2.95V at 3 cm, another would only output 2.8V at that same distance. For this reason, we had to design the voltage dividers specifically for each sensor. The overall subsystem can be seen in figure 6 below. It illustrates the sensors and their inputs, as well as the different voltage dividers used for each sensor.

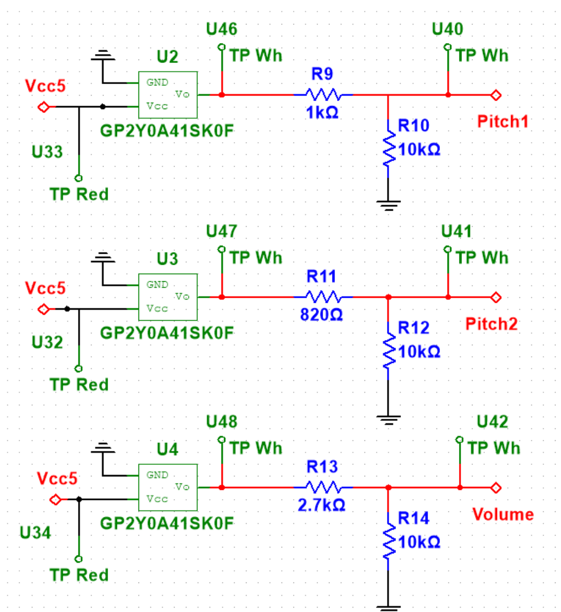

Figure 6: IR Sensor Subsystem Schematic

Microcontroller Subsystem

The next subsystem is the microcontroller subsystem. It is responsible for reading the outputs from the IR sensors in to the ADC, processing the readings, and producing outputs to be sent to the DAC. In this section, the processes/algorithms for doing such will be described. A high-level flowchart of the processes and algorithms can be seen in figure 7 below:

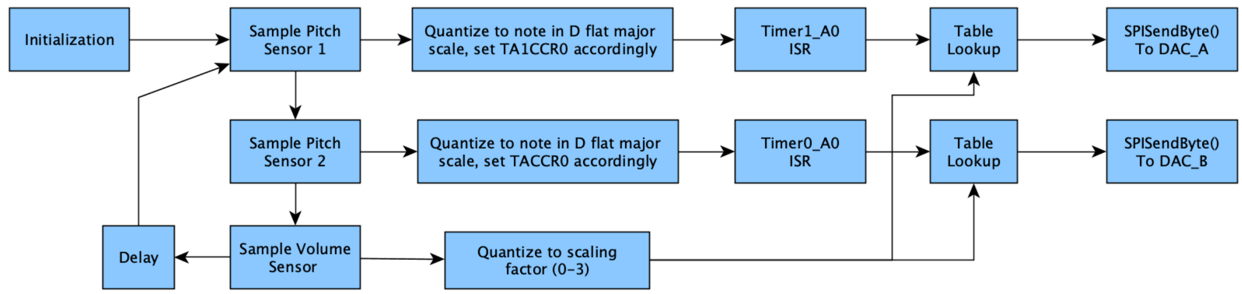

Figure 7: Software Flowchart

After powerup, the MSP430 performs numerous initializations for different modules used. First, the clock module is configured to run at 16 MHz. This is the maximum clock frequency for the MSP430G2553 and it is used to generate precise timer interrupts (described later). Next, the Universal Serial Communication Interface B (USCI_B) module is configured for four-pin serial-peripheral interfacing (SPI) mode. In this initialization, pin 1.5 of the MSP430 is configured to receive the clock output, pin 1.6 is configured to receive master-in, slave-out (MISO) input, and pin 1.7 is configured to receive master-out slave-in (MOSI) output. Next, the pins used to control the DAC are configured. It was selected that pin 2.0 of the MSP430 would serve as the DAC's chip select/load enable, and that pin 1.4 would control the DAC clear, so these pins are set as outputs in software.

After configuring the DAC port pins, the ADC is initialized. The ADC is configured to use the MSP430's internal reference voltage of 2.5 V when making conversions. Due to the shakiness of the sensor readings described previously, the ADC10's maximum sample and hold time of 64 clock cycles is used. We found using this large sample and hold time to be very effective in stabilizing the ADC readings to eventually produce less noisy, more consistent output.

The timer A0 and A1 modules are configured next during the initialization phase. For both timers, the clock source used is SMCLK with an input divide of 1. The timers are both interrupt-enabled and are configured in count-up mode. This means that when the clock's counter reaches the value stored in its respective capture-compare register, an interrupt is generated and the counter is reset to zero.

Lastly in the initialization process, two different struct objects created for this project are initialized. The first of which is a struct called a NotePlayer. The NotePlayer type has three attributes which consist of CurrentNote1, CurrentNote2, and CurrentVolume (figure 8). CurrentVolume is an unsigned integer corresponding to the volume of the sound being played, and CurrentNote1 and CurrentNote2 are an enumerated type called a Note (figure 9). The enumerated type Note has elements that correspond to the 9 different notes that can be played by our device, along with a tenth element called NoNote. The two CurrentNote attributes of the NotePlayer are initialized to NoNote and the CurrentVolume field is initialized to 0.

```c
typedef struct{
    Note CurrentNote1;
    Note CurrentNote2;
    unsigned int CurrentVolume;
}NotePlayer;
```

Figure 8: NotePlayer Struct Type

```c
typedef enum {
    D1_flat, E_flat, F, G_flat, A_flat, B_flat, C_, D2_flat, NoNote
}Note;
```

Figure 9: Note Enumerated Type

The second struct type initialized is called a DAC (figure 10). It has a field called DACAddress which is an integer that corresponds to the desired channel to be used on the four-channel DAC. It also has fields called CurrentArrayIndex and ArrayLength that are used when performing table lookups. The last fields are unsigned int arrays called ArrayValuesPtr1, ArrayValuesPtr2, and ArrayValuesPtr3. These fields are pointers to the lookup tables stored in flash memory, and their use will be described further.

```c
typedef struct {
    int DACAddress;
    int CurrentArrayIndex;
    int ArrayLength;
    unsigned int * ArrayValuesPtr1;
    unsigned int * ArrayValuesPtr2;
    unsigned int * ArrayValuesPtr3;
} DAC;
```

Figure 10: DAC Struct Type

After performing all of this initialization, the ADC begins sampling. The ADC is configured to perform successive single-channel, single-conversion samples on pins 1.0, 1.1, and 1.3 and store the measured values in global variables called CurrentSampleS1, CurrentSampleS2, and CurrentSampleVol. Due to unsolved issues when sampling pin 1.2, we were forced to use this successive single-channel single-conversion approach instead of the successive sequence-of-channels, single-conversion approach that was planned on. The sampling and conversion rate is set to approximately 8 Hz. This rate was determined by trial and error. We found that faster sampling rates produced shakier, less consistent output and that slower sampling rates were not fast enough to detect much of the user's hand motion that occurs during use.

Each time a new sample is taken and new values are stored to CurrentSampleS1, CurrentSampleS2, and CurrentSampleVol, methods called SetPitchOutput1, SetPitchOutput2, and SetVolumeOutput are called to quantize the raw ADC values. In the SetPitchOutput methods, the raw ADC values are compared to different threshold values which correspond to different distances from the sensor. Based on this comparison, the timer A capture control registers are set such that the periods of the Timer A interrupts correspond to the desired notes to be played. The SetPitchOutput1 method is used to alter the TA1CCR0 value while the SetPitchOutput2 method is used to alter the TACCR0 value. The calculations for the timer A capture control register values will be explained further. Additionally, in the SetPitchOutput methods, the CurrentNote1 and CurrentNote2 fields of the NotePlayer struct are set. These values are used so that the Timer A capture control register values are not updated if the note being played has not changed.

The SetVolumeOutput method works similarly to the SetPitchOutput methods. It quantizes the raw ADC value read by the volume sensor into one of four different volume levels: 0, 1, 2, or 3. It then sets the CurrentVolume field in the NotePlayer to this value and its utility is described later.

In order to produce sinusoidal outputs, table lookups are performed in the interrupt service routines for the Timer A0 and A1 modules, and these outputs are sent to the DAC via SPI. Based on the CurrentVolume field of the NotePlayer, one of three different tables are used for the lookup. Each lookup table has a different amplitude but each consists of 16 16-bit unsigned integers that form a sinusoid when viewed in succession. Therefore, in order to produce a sinusoid at a desired frequency, the Timer A capture compare registers are set to trigger at 16 times the desired frequency, sending the subsequent value in the lookup table to the DAC at each interrupt. The calculations for the TAxCCR0 values are shown in equation 2 and table 1 below. The calculations for the lookup values are shown in equation 3 and figure 11 below.

$$TAxCCR0 = \frac{f_{clk}}{16 * f_{note}} - 1 \qquad f_{clk} = 16\ MHz$$

Equation 2: TAxCCR0 Calculation

Note	Desired Freq (Hz)	TAxCCR0 Value	TAxCCR0 Value (Rounded)	Actual Freq (Hz)	Percent Error (%)
D flat	1108.73	900.9328421	901	1109.88	0.1034339787
C	1046.5	954.566173	955	1047.12	0.0591850671
B flat	932.33	1071.581597	1072	932.83	0.0541533641
A flat	830.61	1202.934458	1203	831.25	0.0775772131
G flat	739.99	1350.369613	1350	740.74	0.101352721
F	698.46	1430.721215	1431	698.81	0.0502993377
E flat	622.25	1606.071113	1606	622.66	0.0665943851
D flat	554.37	1802.849415	1803	554.63	0.0470111370

Table 1: TAxCCR0 Calculation

$$Lookup\ Value = (2^{16} - 1) * \frac{A * \sin\left(\frac{2\pi n}{16}\right) + \frac{V_{REF}}{2}}{V_{REF}}$$
$$A = amplitude\ ||\ n\ = lookup\ index\ ||\ V_{REF} = 5V$$

Equation 3: Lookup Table Calculation

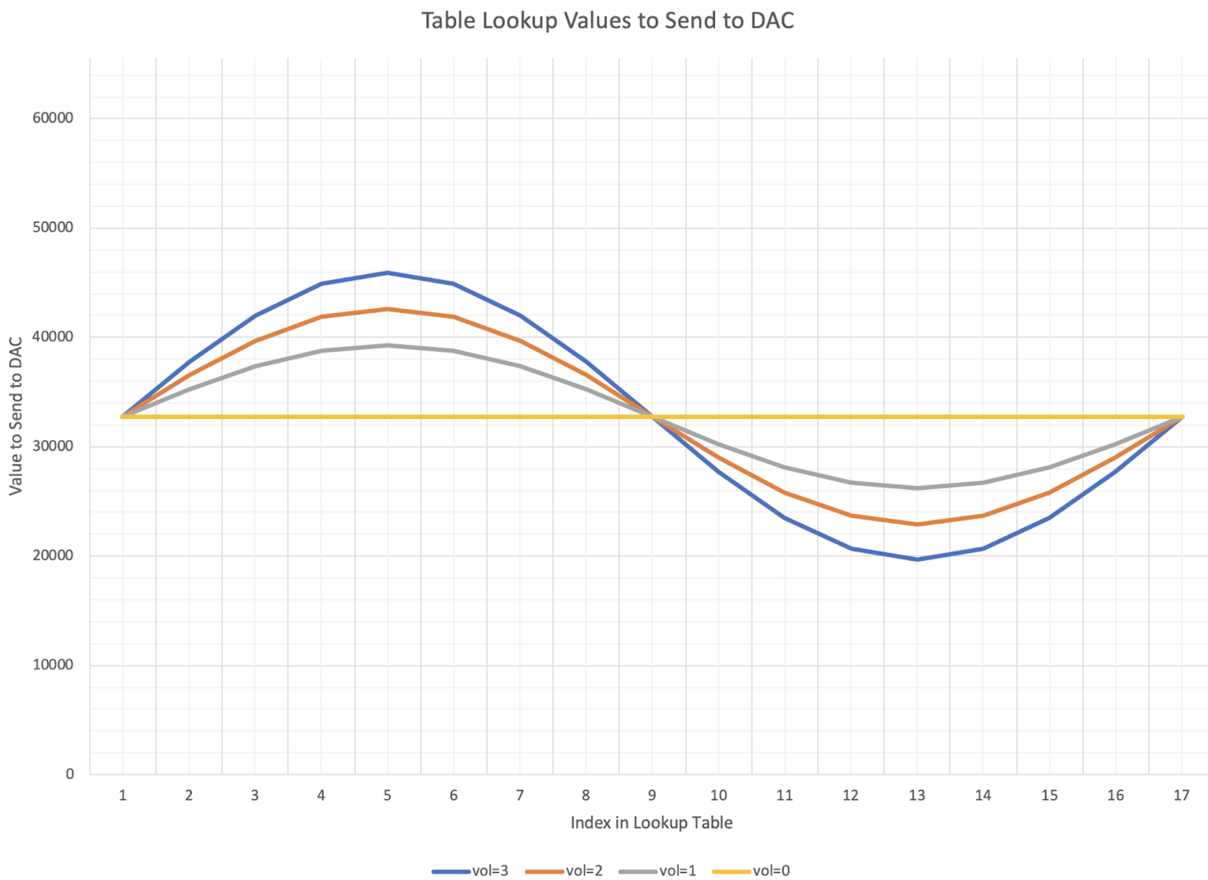

Figure 11: Lookup Table Values

DAC Subsystem

The next subsystem is the DAC. This is where the digital signal output from the MSP430 is sent to be converted back into an analog signal. There are two main components to this subsystem. The first is the DAC itself. For this project, we used the LTC2604 chip made by Linear Technology [20]. It contains a total of four 16-bit DACs, each with their own reference voltages. It operates on anywhere from 2.5V to 5.5V, although in our project, we powered it with our 5V supply. Since the DAC is an active component, we needed to be sure that we included a bypass capacitor, just as with the voltage regulators. The components datasheet recommended the use of a 0.47uF capacitor, however we did not have any capacitors of that denomination and used a 0.33uF instead. Because we were producing up to two separate notes, we only needed two of the possible four DACs. The same 5V supply that powered the device was used as the reference voltage for each DAC. These components take in a number of inputs as seen in figure 12 below. The first three, as mentioned above, are the power supply, Vcc, and the reference voltages, Ref-A and Ref-B. In addition, there is a chip select or load input, CS-LD, a clock input, SCK, a clear input, CLR, and a serial data in input, SDI. This last input is the data itself that is sent from the MSP's ADC. There are also three outputs of the DAC. SDO is the serial data out which is sent back to the MSP. Finally, there are Vout-A and Vout-B, which are the outputs of the first and second DACs, respectively. These outputs are then sent to the next part of the subsystem, the filters.

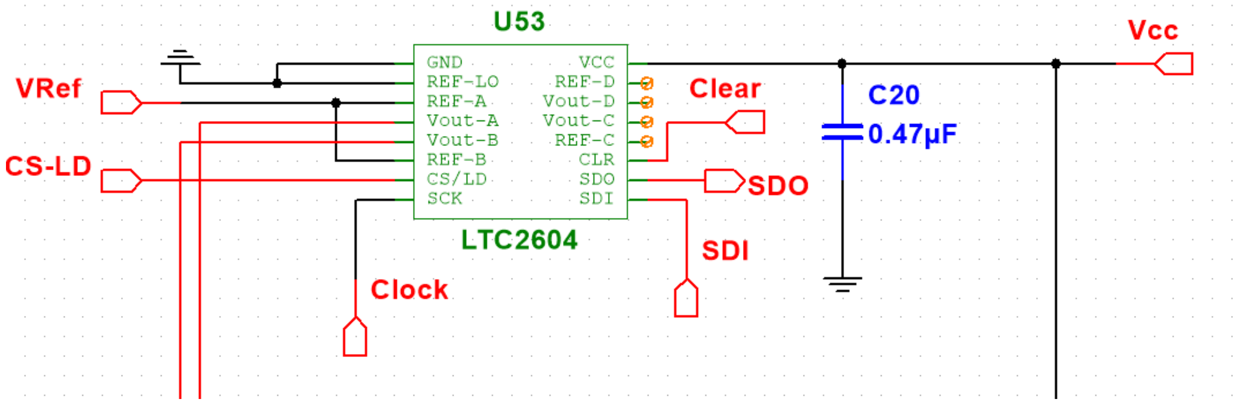

Figure 12: DAC Schematic

The next portion of this subsystem are two Butterworth filters that filter the output of each DAC. This step is necessary because the signals directly out of the DACs are not yet sin waves, Instead, they are quantized and are comprised of a number of discrete steps. This signal is very similar to the red signal in figure 13 below. If this signal were to be played through a speaker, it would not sound good. However, once the quantized wave is sent through a filter, the pure sine wave can be acquired (the blue wave in figure 13). The filter itself is a 2nd Order Low-Pass Butterworth filter and uses the MCP6L92T-E/SN Op-Amp made by Microchip Technology [21]. The specific combination of resistors and capacitors used can be seen in figure 14 below and is responsible for a corner frequency of around 550 Hz. It is also worth noting that we had originally anticipated that the signals coming out of the filters would need to be regulated in some way before being fed into the next stage of the design. For this reason, there are two voltage dividers in the image below that have resistors marked TBD. However, once we actually tested the filtered DAC outputs, we realized there was actually no need for these, and as a result, they were left blank on our board. Finally, on closer examination of the filters in figure 14, there is a noted absence of a bypass capacitor for the op-amp used. This would have caused problems and as a result, we had to solder a bypass capacitor directly to the pins of the chip in order to circumvent this issue.

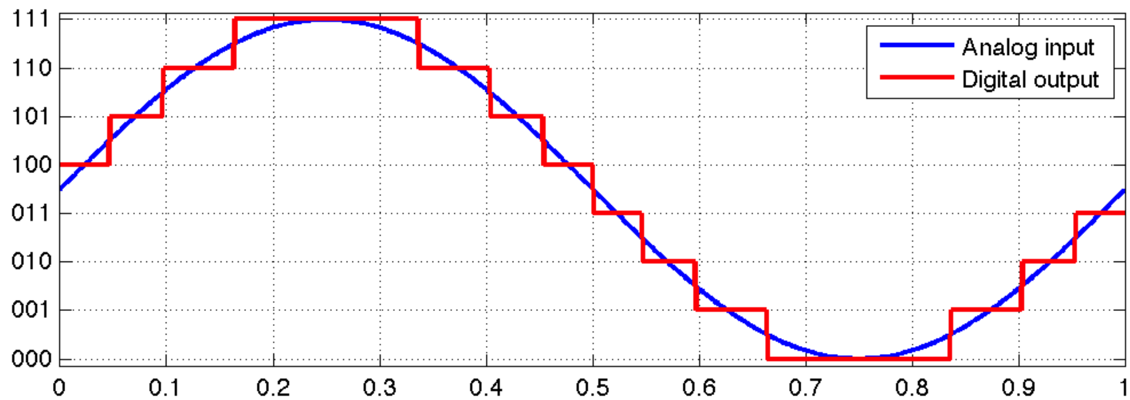

Figure 13: Raw DAC Output vs. Filtered DAC Output

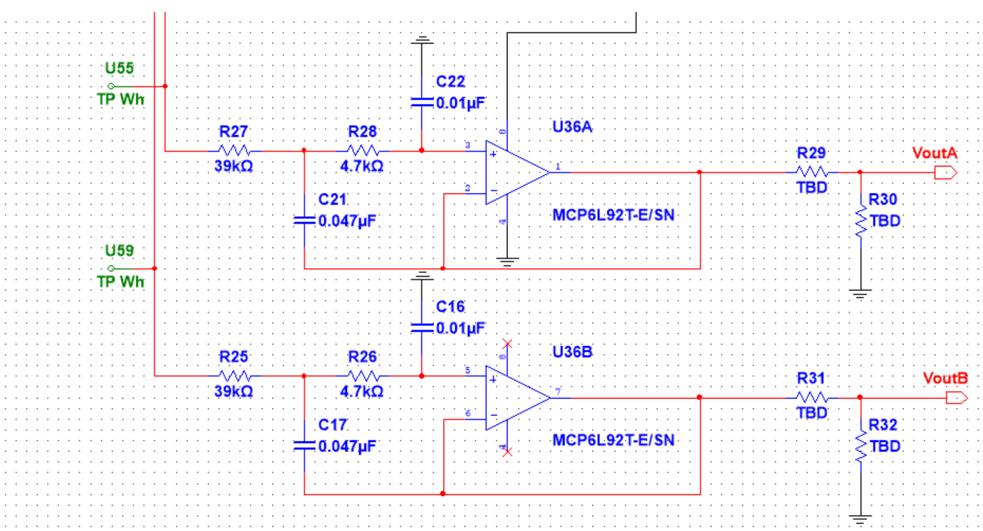

Figure 14: DAC Low-Pass Filters Schematic

Adder Subsystem

After the DAC subsystem is the Adder. This stage is where the two separate filtered DAC outputs are combined in order to create one composite waveform, which is then played through the speaker in the last subsystem. The adder was created using a TL072ACP op-amp made by Texas Instruments [22] and a network of resistors which can be seen in figure 15 below. R18 and R19 were both chosen to have the same value so that the two incoming notes would be weighted the same when added together. R20 and R21 were chosen to minimize the gain of the amplifier as we did not want the output of the amplifier to be too much for the next subsystem to handle. It is also worth noting that we needed a bypass capacitor for this circuit, like with the other active components present in our project. In the schematic below, it shows a 1uF capacitor but in reality, we used a 0.1uF cap.

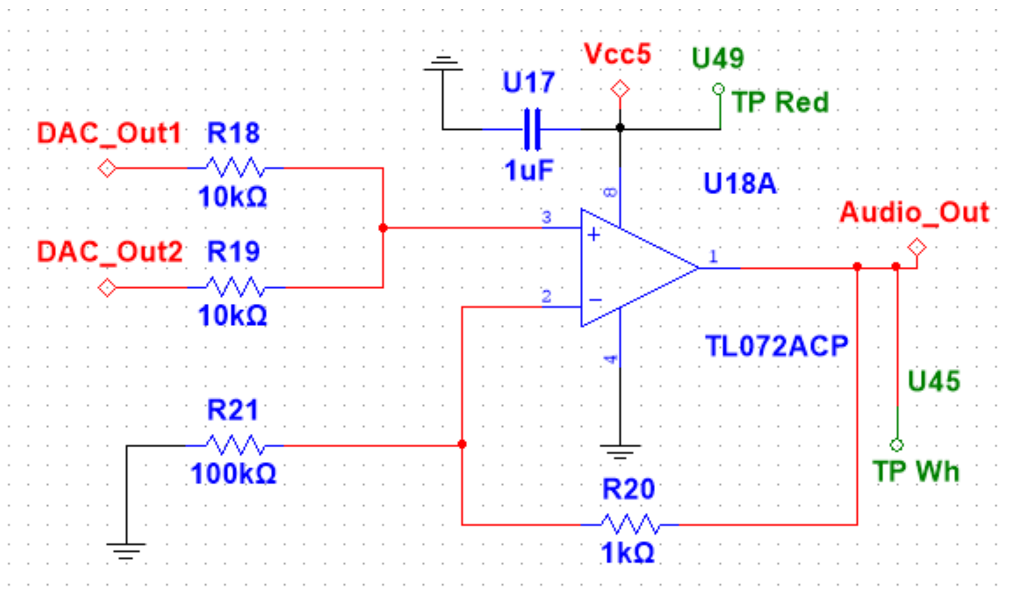

Figure 15: Adder Schematic

Power Amplifier/Speaker Subsystem

The final, and most complicated hardware subsystem of the project is the power amplifier. This component is what actually makes it possible to produce an output from a speaker. In this project, the speaker we used was a SP-3605-1Y 1.5W 8Ω speaker [23]. It was chosen due to its relatively small form factor and its low output power (approximately 80 dBm). Once this component was selected, a power amplifier was found that matched these characteristics. Specifically, the Texas Instruments LM4991MAX/NOPB chip was selected [24]. It is capable of delivering 1.5W to an 8Ω load when powered by a 5V supply which fit perfectly within the bounds of our project. This particular chip is also a differential power amplifier, which signifies that it is capable of achieving up to four times the power output of a single-ended (i.e. non-differential) amplifier.

What made this subsystem complicated was that a network of resistors and capacitors had to be designed by the user, although the datasheet was very helpful in explaining the guidelines behind making these design decisions. The schematic for this subsystem can be found below. The datasheet strongly suggested that both C4 and C5 be 1uF capacitors as that value minimizes any pops or clicks when the speaker turns on. R7 and R8 were chosen to yield a differential gain, Avd, of 3, using the formula R8/R7 = Avd/2. C6 was chosen in agreeance with R7 to create a high-pass filter with a corner frequency of 5*18.5 or approximately 93 Hz.

Once these values were selected, the speaker was connected to the power amplifier via two soldered on wires. When a simple 440 Hz sine wave was inputted via Audio_In, we heard a slightly distorted output. However, this was likely due to a poor connection to the speaker itself, because when we took a new speaker and took special care to ensure that we had a solid solder connection, the difference in sound was very noticeable with much less distortion and noise.

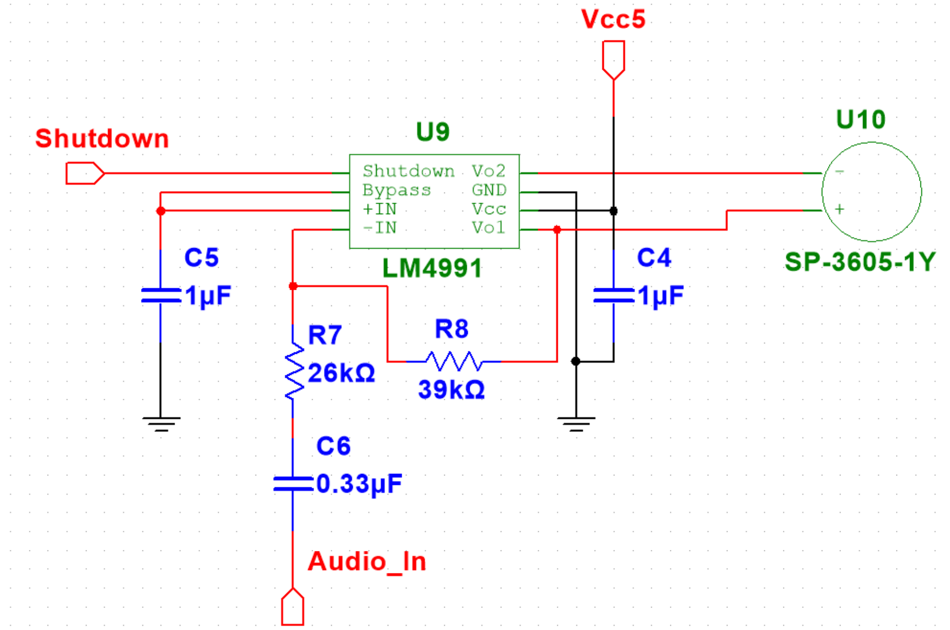

Figure 16: Power Amplifier Schematic

Board Layout

Below (figure 17) is the final layout of our board. One noticeable feature of the board is that it utilizes a copper bottom power plane for Net 0 or "ground". As a result, the majority of traces are kept to the copper top layer, and vias are only used when necessary, as seen comparing figure 18 to figure 19. This is to ensure that the ground plane is kept as open as possible so that no electrons flowing on this plane are cut off from one another.

Another noticeable feature is the board's size, measuring in at 7x4 inches. Obviously, this is quite large for a PCB. However, during production, size was not a major constraint for this project. 7x4 inches was the limit we set on the size of the board so it was originally designed to be that size. Looking at the board, it is clear that it could easily be condensed into something much more compact. Unfortunately, we did not have time to send out one last more compact board once we knew that our design worked. However, taking into account the fact that we don't actually need the two LED Matrices or their connecting headers, the board would have been much smaller. In addition, we opted to use primarily through hole resistors and capacitors as we had an abundance of those leftover in our lab kits. However, if we switched to surface mount components and 0805 packages, the design would have been even more compact. We are confident that if these changes were made, we could reconfigure the board into a much more compact and manageable size.

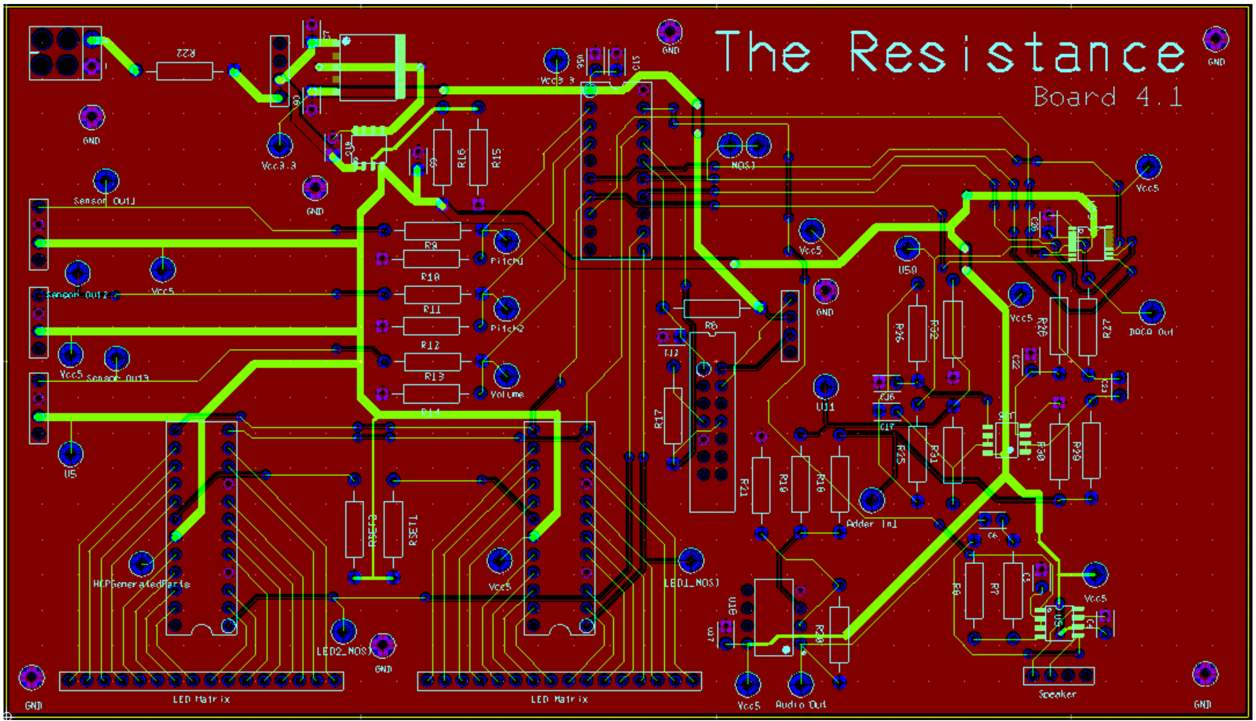

Figure 17: Final Board Layout

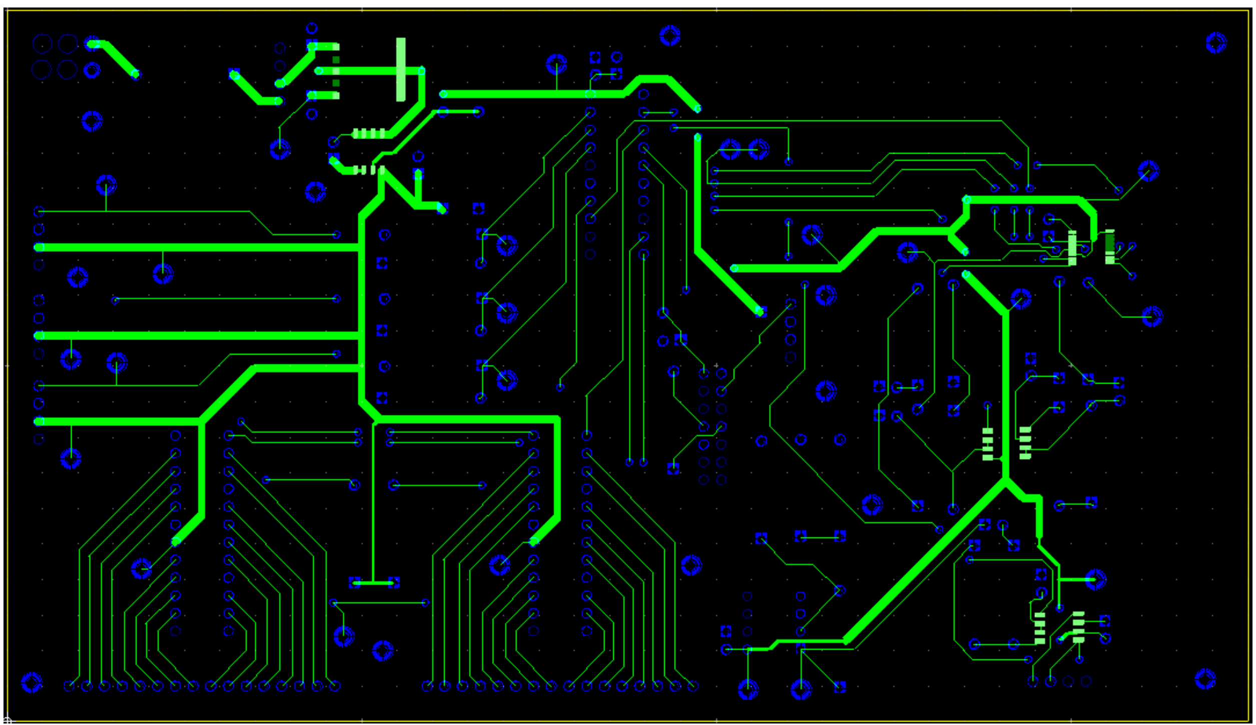

Figure 18: Final Board Layout – Copper Top

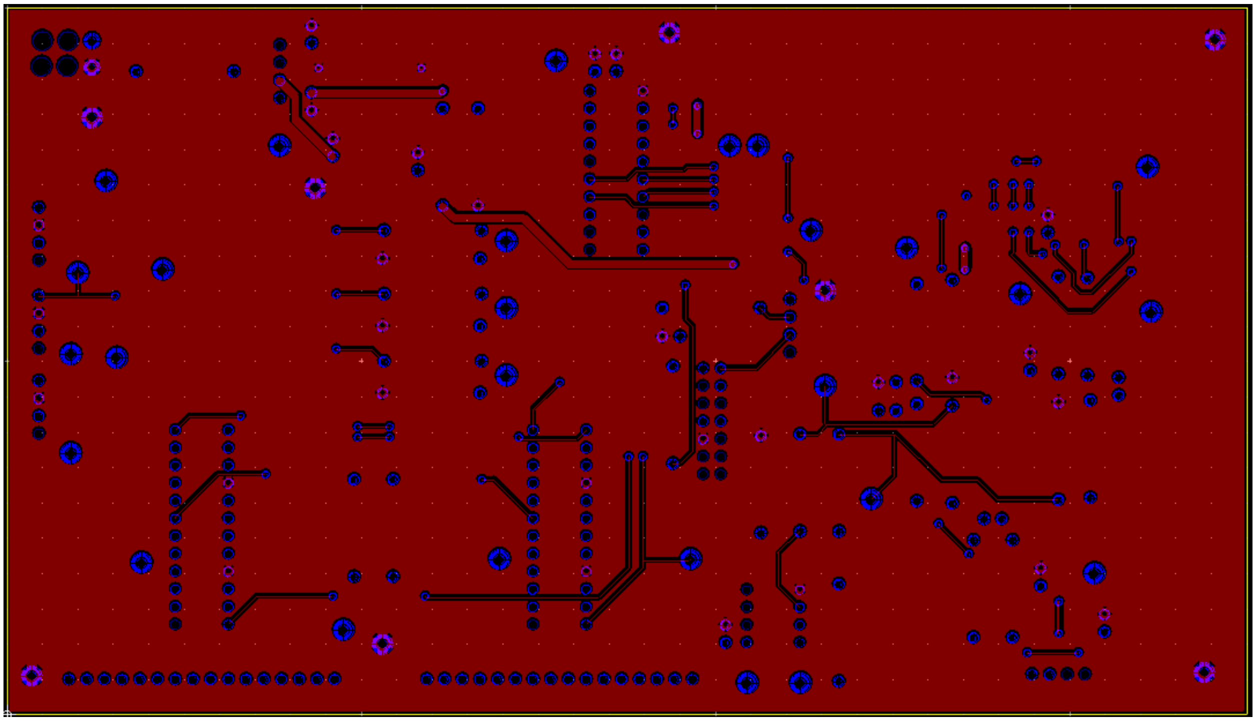

Figure 19: Final Board Layout – Copper Bottom

Final Assembly

Once the subsystems were all assembled, they all had to be soldered onto the board and the external components needed to be connected. This was accomplished using header pins and jumper wires. Figure 20 below shows how the three sensors were connected to the board using grey wires for ground, green wires for Vcc, and blue wires for Vout connections. It also shows two male-to-female wires connected to the speaker. The male ends of the black and red wires are soldered onto the back of the speaker, while the female ends are connected to a set of header pins on the other side of the board.

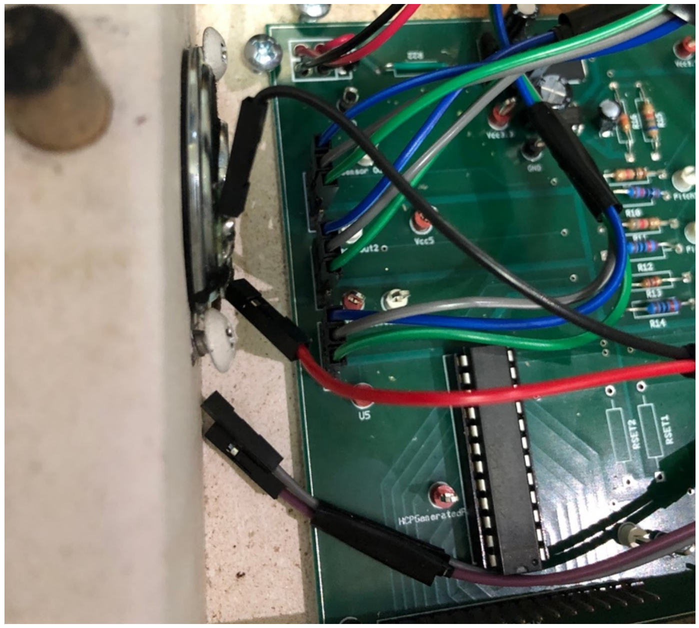

Figure 20: Sensor Connections to PCB

Design Modifications

There were a couple of last-minute design modifications that we made to our project. One dealt with the problem we ran into when trying to sample three separate channels. Originally, we had the volume sensor connected to pin 1.2 of the MSP. However, due to difficulties using that pin, we switched to pin 1.3 in software. However, on the PCB the MSP was still connected to P1.2. As a result, we had to sever this connection, and then solder a jumper wire along the bottom of the board to connect the volume sensor output to its proper pin (P1.3).

Another modification we made involved the shutdown pin of the power amplifier. This issue is discussed in more depth later on in the Test Plan section, but essentially, pin one of the power amplifier needed to be grounded or the sound produced by the speaker would randomly shut off. To fix this issue, like before, we soldered a wire to the bottom of the board in order to directly connect the shutdown pin to ground, and stop the sound from cutting off.

A final modification that we needed to make to the board involved the filters that dealt with the DAC outputs. Originally, we were unsure as to how large the signal sent out of the DAC would be, and as a result, we included two voltage dividers directly following the filters to regulate the signals. However, after testing, we determined that these voltage dividers were not needed, and as a result, we needed to jump the outputs of the filter directly to the summing amplifier. For one output, we did the same as the last two modifications - soldered a wire on the underside of the board connecting the filter to the summing amplifier. For the second output, we recognized the opportunity to add some added functionality to our device.

Instead of soldering on another wire, we found a switch in the NI Lab and connected it to the board via two male-to-female jumper wires. This gave us the ability to toggle between playing one or two notes. When the switch is turned off, the second DAC output is not sent to the summing amplifier, and only the first signal reaches the power amplifier and speaker. As a result, only one note is played. But when the switch is turned on, it creates a connection between the summer and the second DAC output, and two notes are sent to the sound actuation subsystem.

Project Time Line

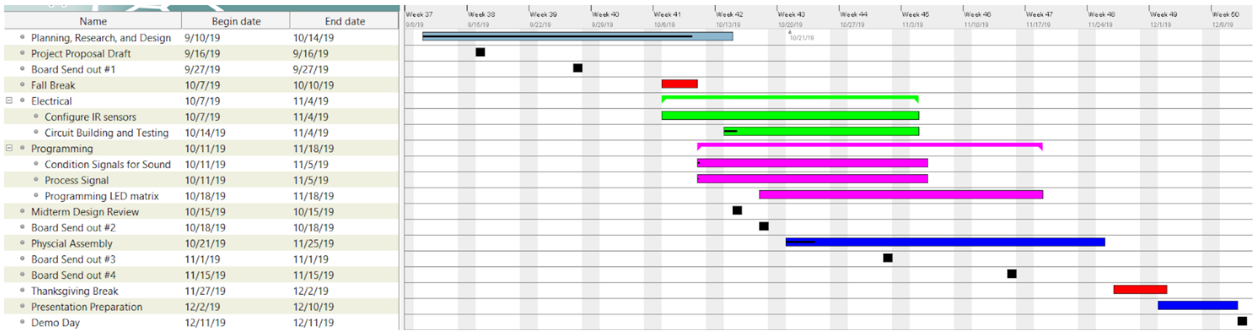

Figure 21: Initially Proposed Gantt Chart

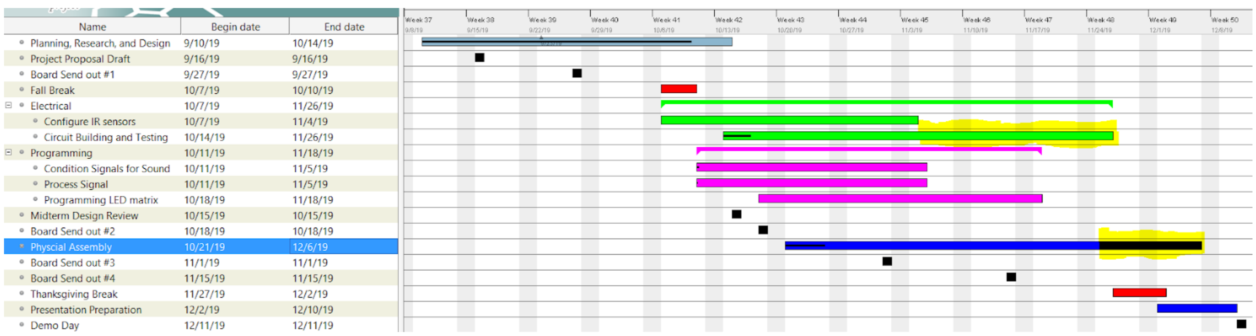

Figure 22: Final Updated Gantt Chart

The Gantt chart shows the timeline of events throughout the course of the semester including class deadlines, production deadlines, holidays, and work efforts. There were numerous components of the project that included (but were not limited to) building numerous filters, testing the filters, sampling/quantization of the input signals, embedded programming for the LED matrices, embedded programming for the output signal, and setting up the physical structure of our system. Many of the tasks were able to be completed in parallel so they could be worked on simultaneously. As discussed in more detail at the bottom of this section, to our surprise, there were some aspects of our project that had to be done in series, notably testing and physical assembly.

Elmo took the lead role in building the physical structure of the system. This included using 3D CAD software and 3D printers to meet the needs of the different components of the system to display them as one finished product. Elmo also took a secondary role in circuit building and testing to filter the signals received from the infrared sensors.

Michael took the lead role in the embedded programming of the device. This included the sampling and quantization of the input signals to synthesize output signals. He also built the final enclosure, in addition to assuming a secondary role in the circuit design, assembly, and testing, along with a secondary role in testing the infrared sensors.

Landon was in charge of the hardware design of the Theremin. This involved designing and laying out PCBs, designing the hardware filters and adders required for the project, and IR sensor interfacing. He also assisted with the development of the embedded software of the MSP's internal ADC.

Our timeline was pretty severely altered due to when we received both our parts and boards. Our first parts order came in well before our first board, and as a result we were not able to conduct any testing of our electrical components until after Thanksgiving break. We did stay relatively on schedule in regards to the writing of our software, as we had an MSP launchpad from the start, and were not really waiting on anything else. In that regard, the software progressed in parallel with the physical board assembly and testing. The physical assembly of the encasement was also pushed back later than we had anticipated for a couple of reasons. On one hand, we needed to have the board almost completely finished before we could fit it into the encasement, so the delay in our parts/board orders delayed our assembly. On the other hand, we had originally planned to 3D print the entire container, however, we ran into significant delays on that front as well. In the end, we changed our plans and ended up fabricating the final enclosure over Thanksgiving break and assembled the final project the night before demo day.

Test Plan

Figure 23 below depicts our original test plan form our proposal. For the most part, we followed this plan when constructing our project, although the order of certain tests was changed.

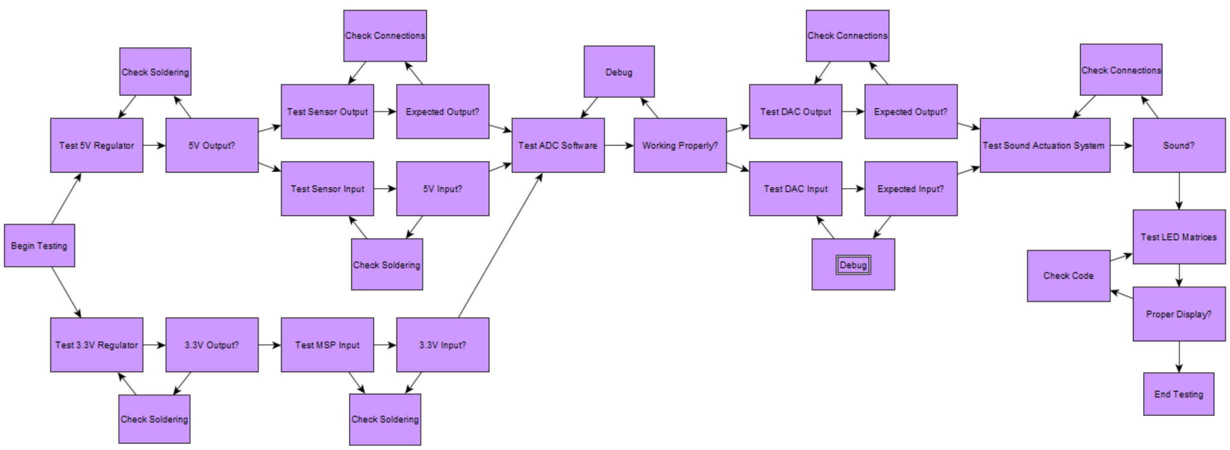

Figure 23: Initial Test Plan Flowchart

The first thing we tested in our project were the infrared sensors instead of the two voltage regulators. This is because we received our parts well before we received any boards, and as a result, we were not able to test any of the surface mounted parts. To test the sensors, we first connected the power supply of the virtual bench to a breadboard. From there, we connected the Vcc and Ground pins of the sensors to the breadboard, and put a DMM on the Vout pin. From there we tested the sensors by moving our hands within the specified range to get our own characteristic graphs. It was here that we realized that each sensor does not output the same voltage when an object is placed the same distance away. For that reason, we then worked on designing voltage dividers that were unique to each sensor.

By the time we had completed this step, our boards had still not arrived, so we moved on to testing the ADC/DAC code of the MSP. To do this, we used an MSP430 launchpad and DAC header board that we had left over from taking Intro to Embedded Systems. First, we connected one of the sensor's outputs to the bottom of the launchpad, and fitted the DAC header to the top. Then, we tested to see if the code we had written was functioning properly by connecting an oscilloscope to one of the output pins of the DAC header. We expected to see a sine wave whose frequency changed as the distance from the sensor to one of our hands was either increased or decreased. If we did not see the expected results, we went back and debugged our code before repeating this process again.

Once we had one sensor working, we moved to two. Like before, the sensor outputs were connected to the bottom of the launchpad, while the DAC header was connected to the top. The same steps as before were repeated until we saw two separate sinusoids whose frequency changed in response to the output of the sensors.

The last part of testing our code involved adding the volume functionality by adding a third sensor. The same steps as before were performed until we came across a roadblock. As mentioned above in the MSP subsystem explanation, for reasons still unknown, we could not get the sampling to function properly when using P1.2. As a result, we had to change the code functionality to allow us to sample from pins 1.0, 1.1 and 1.3 instead. To test this new pinout, we had to scratch out the connection between the volume sensor and pin 1.2 and solder a jumper wire between the volume sensor and pin 1.3. Once this was done, we were able to confirm desired functionality by confirming that we were able to produce variable-amplitude output based on the proximity between a user's hand and the volume sensor.

By the time we got the code working, we had received our first board. So, we went back to the original test plan and started testing the power supplies once we had soldered the necessary components on our board. From the start, everything worked as expected, so no debugging or correction of the system was necessary. When a 9V supply was provided to the board, both regulators were checked by connecting a DMM to ground and to various test pins across the board connected to either the 5V or 3.3V supplies.

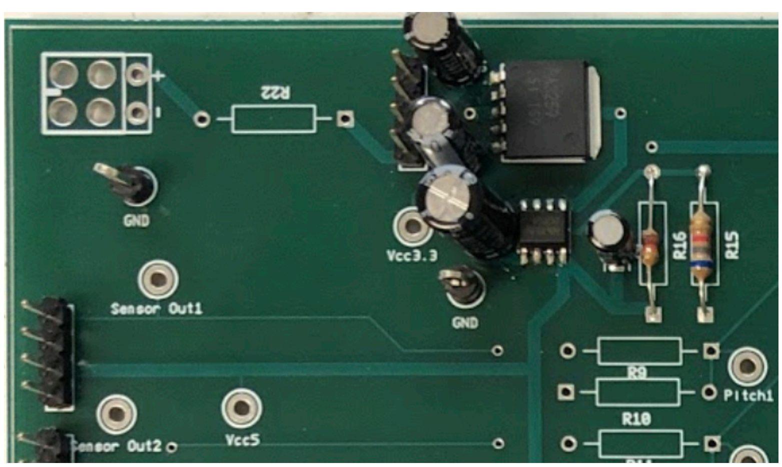

Figure 24: Power Subsystem on PCB

Once the power supplies and voltage regulators had been verified and the power amplifier and necessary resistors/capacitors were soldered on to the board, we began testing the sound actuation system. First, we simply connected the function generator of the Virtual Bench to a test pin connected to the input of the power amplifier, and supplied a simple 440 Hz sinusoid at a very low amplitude. After slowly increasing the amplitude of this signal, we began to hear a sound emanating from the speaker that corresponded with the 440 Hz signal (or A note). This was checked using a Tuner application for iOS called GuitarTuna [25]. However, we noticed a fair amount of distortion and popping from the speaker. We addressed this issue by going back to and carefully combing through the datasheet of the power amplifier until the problem was determined to be caused by an improper selection of resistors and capacitors.

Once this was fixed, the sound quality was markedly improved. However, we ran into another issue where after a certain time, the power amplifier would shut off and the speaker would no longer produce any sound. Again, to fix this we returned to the data sheet and examined it until the issue was caught. As it happens, this power amplifier possesses shutdown functionality - when a voltage is applied to pin 1 of the chip, it turns itself off and will not start up again until the supply voltage is refreshed. To temporarily fix this, we placed a jumper wire on the shutdown pin and connected it to a ground pin on the board. This solved our problem instantly, and as a result, we removed the temporary solution and soldered an extra wire to the bottom of the board, connecting pin 1 of the power amplifier to a ground pin. This provided a much more stable connection and removed unnecessary wires from the board.

Once the power amplifier was functioning properly with just a single synthetic note, we moved on to testing it with the output of a DAC. To do this, we used the same MSP430 launchpad and DAC header as before and jumped the output to the test pin connected to the input of the power amplifier. We tested it with just one note/sensor first and found positive results - the speaker was producing a note equivalent to the frequency of a wave generated from the output of the sensor. As shown in figure 25 below, while this test did work, it was very messy with lots of external wires required to ensure that everything was properly connected.

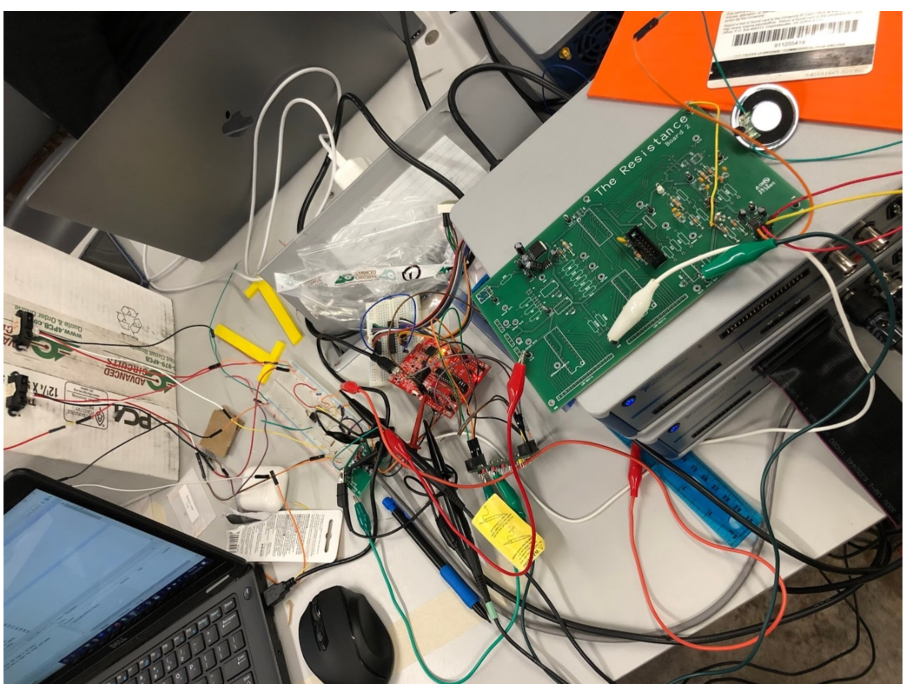

Figure 25: Breadboard Testing of Whole System

After we knew that our sound actuation system and ADC/DAC code worked, we soldered on the physical DAC chip along with the requisite filters, and continued testing. Now, we got rid of the DAC header board and connected the MSP430 launchpad to the DAC chip on the board. The same steps were performed as before and when the sensors detected a user's hand, a note was played from the speaker corresponding to the sensor's output.

This test confirmed that our system worked with one note. However, we still needed to ensure that it could play two notes simultaneously. To accomplish this, as mentioned in the technical description, we used a summing amplifier. We first tested our design of this summer by building it on a breadboard and then feeding two different frequency sine waves in and observing the output. Once this was tested and confirmed, we soldered the components onto the board. From here, we connected both pitch sensors to the MSP and observed two separate notes being played when the two sensors output different signals. This agreed with what we expected.

At this point, all components of our project had been tested both separately, and while interconnected. The final step was to solder on all of the remaining parts and test the whole board when powered with a 9V battery. Once the board was fully populated, we connected a battery and heard notes when we moved our hands in front of the sensors, thus confirming proper functionality.

Final Results

The final product produced was a digital, polyphonic Theremin which plays notes depending on the location of the user's hands. The device consists of two IR proximity sensors which determine the pitches of two distinct notes and a third IR proximity sensor which determines the volume of the output. It also contains a switch which can be used to toggle between playing one or two notes. The device is packaged and presented in a visually appealing manner, as seen in figures 26, 27, 28, and 29 below.

The only constraint specified in the initial proposal that was not met was the inclusion of the two LED matrices to inform the user of the current notes being played. This was because the MAX7219 LED driver chips [26] ordered need 3.5V input to register a logic high, and the MSP430 is only able to produce an output of up to 3.3V. We did not realize this until it was too late to order new LED drivers, as the initial chips ordered did not ship until after Thanksgiving break. However, we should have been more thorough in the examination of the LED driver datasheet and caught this problem before it occurred. An attempt was made to get the LED displays to work using a transistor setup, but it was unsuccessful. A picture of this transistor setup can be seen in figure 30 in the appendix.

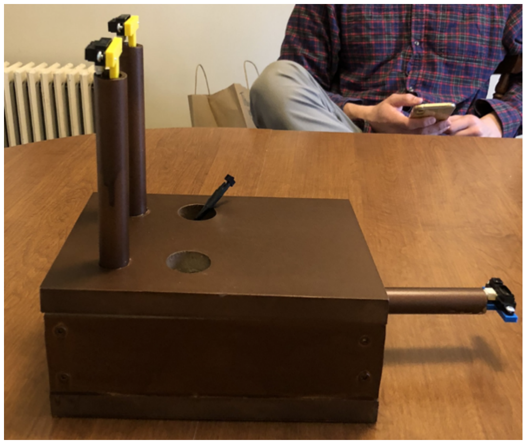

Figure 26: Digital Theremin Final Product – Side View

Figure 27: Digital Theremin Final Product – Angled View

Figure 28: Digital Theremin Final Product – Open Box Side View

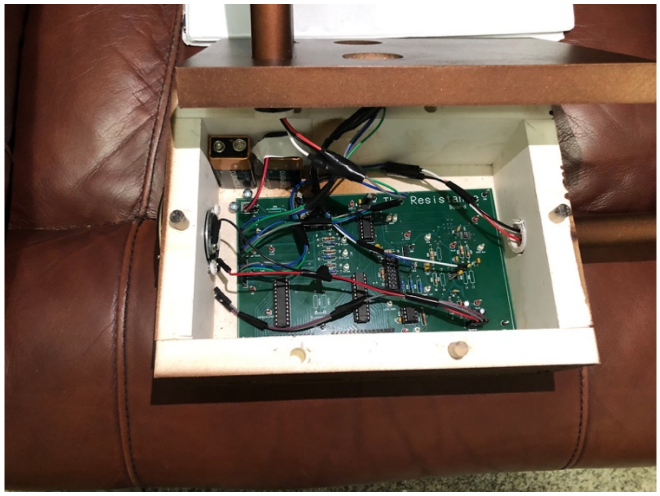

Figure 29: Digital Theremin Final Product – Open Box Aerial View

Costs

Overall, this was a fairly inexpensive project. Table 2 in the appendix below gives a detailed cost breakdown of all of the electrical components that were required to build our Theremin. The total cost for the components used in this design was $63.86. It is worth noting that we did not spend any money on resistors or capacitors, as those were taken from our lab kits for no cost. In addition, all materials used for the enclosure were obtained from a construction site's dumpster at no cost, so no money was spent in that area either. As for PCB manufacturing, we are unaware of the exact cost of this procedure as all boards for the class were submitted and printed in bulk orders handled by the school. As a result, that cost was not included in the breakdown.

If we were to manufacture our product in 10000-unit quantities, the costs would decrease dramatically. In terms of the electrical components needed, if the same DAC and through hole resistors and capacitors were used, then the price per unit would be almost halved, totaling at $35.68 per unit. This already is a huge reduction in price. However, if a new dual DAC was selected over the old quad DAC, in addition to using chip resistors (i.e. 0508 packages), then the price savings would be even greater, with the total dropping even further to $24.13 per unit. And on top of the flat-out savings per component, using smaller resistors and capacitors would substantially decrease the surface area needed to construct the board, which would provide even further savings on both PCB manufacturing costs, and the materials necessary for the enclosure. With this in mind, it is reasonable to assume that using automated equipment to produce orders in bulk would massively save money on the cost of producing a unit.

Future Work

In the process of manufacturing the digital polyphonic Theremin, numerous possible extensions were thought of that would add to/change the product's functionality. One very simple extension would be to change the notes produced by sensor 2. All it would take to do such would be to recalculate the necessary TA1CCR0 values and update them in the OutputControl.h header file. This would allow the user to produce notes over a greater frequency range and depending on the notes selected, could produce some interesting output.

Another possible extension of our project would be to change the output signal's form. The output signal that we use is in the form of a pure sinusoid; however, it could be interesting to experiment with sound produced by a sawtooth wave, a square wave, etc. To do such, the user would have to derive their own equation for the new waveform and would have to update the lookup table values accordingly. This could be interesting to examine the effects that these waves have on the sound produced by the speaker.

One final possible extension of our project would be to include the use of a second IR sensor to determine a second volume. This would allow two users to play the Theremin simultaneously. If the notes produced by sensor 2 were shifted an octave up or down, the inclusion of this feature could allow for some interesting-sounding duets.

In hindsight, there are definitely some pitfalls that we would recommend a future group look out for. One example would be to make sure that the boards and parts that you need are sent in on time, the earlier the better. It is much easier to fix any problems that may arise if you have enough time to highlight the issue and then order new parts to fix it. Another piece of advice would be to really examine the voltage and current requirements of each component that is used in the device. If we had kept a running list of all of this information for each component and really took the time to dig through the datasheets, we would have caught the issue with our MSP not being able to provide a high enough voltage to the LED Matrix drivers. As a result, our final product would have been even better and we would have fully satisfied all of the requirements that we laid out in the beginning of the semester.

References

[1] Grimes, G. (2019). How a Theremin Works. [online] HowStuffWorks. Available at: https://electronics.howstuffworks.com/gadgets/audio-music/theremin1.htm [Accessed 24 Sep. 2019].

[2] Demiris, Y. & Wu, Y. (2011). Towards anthropomorphic robot Thereminist, International Conference on Robotics and Biomimetics, Tianjin, China, December 2010.

[3] Brito, P., Gomez, D. and McCabe, P. (2016). Final Project Theremin. [online] MIT Analog Electronics Lab. Available at: https://web.mit.edu/6.101/www/s2016/projects/djgomez_Project_Final_Report_s.pdf [Accessed 16 Sep. 2019].

[4] McKenzie, S. and Rablau, A. (2019). Infrared Theremin. [online] Cornell Engineering. Available at: https://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2013/sjm298_ar568/sjm298_ar568/sjm298_ar568/index.html [Accessed 16 Sep. 2019].

[5] "MSP430G2553IN20." Digi-Key, 296-28429-5-ND, Texas Instruments, 2019, https://www.digikey.com/product-detail/en/texas-instruments/MSP430G2553IN20/296-28429-5-ND/2638885.

[6] "NI Circuit Design Suite: NI Multisim, NI Multisim MCU Module, NI Ultiboard." 14.1, National Instruments Corporation, 2007.

[7] "Code Composer Studio (CCS) Integrated Development Environment (IDE)." 7.4, Texas Instruments, 2017, www.ti.com/tool/CCSTUDIO.

[8] "SolidWorks." Dassault Systemes, 2019, www.solidworks.com/.

[9] "Introduction to IPC Inspection Guidelines: IPC-A-600 vs. IPC 6012." Bay Area Circuits, 27 July 2019, https://bayareacircuits.com/introduction-to-ipc-inspection-guidelines/.

[10] "GitHub." Microsoft Corporation, 2019, https://github.com/.

[11] "The MakerBot Replicator+." MakerBot Industries, LLC, 2016, https://www.makerbot.com/3d-printers/replicator/.

[12] Asha.org. (2019). Loud Noise Dangers. [online] Available at: https://www.asha.org/public/hearing/Loud-Noise-Dangers/ [Accessed 26 Sep. 2019].

[13] "Decibel X: dB Sound Level Meter." SkyPaw Co. Ltd, 2019, https://apps.apple.com/us/app/decibel-x-db-sound-level-meter/id448155923.

[14] Greenlee, Jonathan (2018). United States Patent No. US10395630B1. Retrieved from http://patft.uspto.gov/netacgi/nph-

Parser?Sect1=PTO1&Sect2=HITOFF&p=1&u=/netahtml/PTO/srchnum.html&r=1&f=G&l=50&d=PALL&s1=10395630.PN.

[15] Greenlee, Jonathan (2018). United States Patent No. US10199022B1. Retrieved from https://patents.google.com/patent/US10199022B1/en?q=theremin&q=sensor&oq=theremin+sensor

[16] Freeland (2000). United States Patent No. US6066790. Retrieved from http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=6066790.PN.&OS=PN/6066790&RS=PN/6066790\

[17] "BA3259HFP-TR Voltage Regulator." Digi-Key, BA3259HFPCT-ND, Rohm Semiconductors, 2019, https://www.digikey.com/product-detail/en/rohm-semiconductor/BA3259HFP-TR/BA3259HFPCT-ND/3768982

[18] "L6932D1.2TR Voltage Regulator." Digi-Key, 497-5724-1-ND, STMicroelectronics, 2019, https://www.digikey.com/product-detail/en/stmicroelectronics/L6932D1.2TR/497-5724-1-ND/1299970

[19] "GP2Y0A41SK0F IR Sensor." Digi-Key, 1855-1015-ND, SHARP/Socle Technology, 2019, www.digikey.com/product-detail/en/sharp-socle-technology/GP2Y0A41SK0F/1855-1015-ND/3884447?utm_adgroup=Sensors%20&%20Transducers.

[20] "LTC2604CGN#PBF DAC." Digi-Key, LTC2604CGN#PBF-ND, Linear Technology/Analog Devices, 2019, https://www.digikey.com/product-detail/en/linear-technology-analog-devices/LTC2604CGN-PBF/LTC2604CGN-PBF-ND/962870.

[21] "MCP6L92T-E/SN Operational Amplifier." Digi-Key, MCP6L92T-E/SNCT-ND, Microchip Technology, 2019, https://www.digikey.com/product-detail/en/microchip-technology/MCP6L92T-E-SN/MCP6L92T-E-SNCT-ND/2060318.

[22] "TL072ACP Operational Amplifier." Digi-Key, 296-7190-5-ND, Texas Instruments, 2019, https://www.digikey.com/product-detail/en/texas-instruments/TL072ACP/296-7190-5-ND/378396.

[23] "SP-3605-1Y 1.5W 8Ω Speaker." Digi-Key, 433-1265-ND, Soberton Inc., 2019, https://www.digikey.com/product-detail/en/soberton-inc/SP-3605-1Y/433-1265-ND/9924426.

[24] "LM4991MAX/NOPB Power Amplifier." Digi-Key, 296-35283-1-ND, Texas Instruments, 2019, https://www.digikey.com/product-detail/en/texas-instruments/LM4991MAX-NOPB/296-35283-1-ND/3738991.

[25] "GuitarTuna: Guitar, Bass Tuner." 5.12.0, Yousician Ltd., 2019, https://apps.apple.com/us/app/guitartuna-guitar-bass-tuner/id527588389.

[26] "MAX7219 IC LED Driver." Digi-Key, MAX7219ENG+-ND, Maxim Integrated, 2019, www.digikey.com/product-detail/en/maxim-integrated/MAX7219ENG/MAX7219ENG-ND/774127

Appendix

Current Project		Products	1
Part	Qty	Price/Unit	Price
MSP430g2553	1	2.7	2.7
LTC2604 DAC	1	18.42	18.42
LM4991 Power Amp	1	1.35	1.35
IR Sensors	3	11.18	33.54
MCP6L92T-E/SN OpAmp	1	0.78	0.78
TL072ACP OpAmp	1	0.98	0.98
BA3259HFP-TR	1	1.53	1.53
SP-3605-1Y	1	1.76	1.76
L6932D1.2TR	1	2.8	2.8
Resistors	20		0
Capacitors	17		0
		Tot	63.86

Using same DAC + ThroughHole Resistors/Capacitors

Future Designs		Products	10000
Part	Qty	Price/Unit	Price
MSP430g2553	10000	1.32	13200
LTC2604 DAC	10000	12.19	121900
LM4991 Power Amp	10000	0.57	5700
IR Sensors	30000	5.49	164700
MCP6L92T-E/SN OpAmp	10000	0.65	6500
TL072ACP OpAmp	10000	0.42	4200
BA3259HFP-TR	10000	0.75	7500
SP-3605-1Y	10000	0.975	9750
L6932D1.2TR	10000	1.28	12800
Resistors	200000		1710
Capacitors	170000		8840
		Tot	356800
		Tot Price/Per	35.68

*Worth noting that there are only ~400 of these DACs available^^^

Using New DAC + Chip Resistors/Capacitors

Future Designs		Products	10000
Part	Qty	Price/Unit	Price
MSP430g2553	10000	1.32	13200
PCM1774RGPR	10000	1.46	14600
LM4991 Power Amp	10000	0.57	5700
IR Sensors	30000	5.49	164700
MCP6L92T-E/SN OpAmp	10000	0.65	6500
TL072ACP OpAmp	10000	0.42	4200
BA3259HFP-TR	10000	0.75	7500
SP-3605-1Y	10000	0.975	9750
L6932D1.2TR	10000	1.28	12800
Resistors	200000		1870
Capacitors	170000		476
		Tot	241296
		Tot Price/Per	24.1296

				Through Hole		Chip Resistors	
				Units	10000	Units	10000
Resistors Used	Qty	Resistors U	Qty	Price/Unit	Price	Price/Unit	Price
330Ω	1	330Ω	10000	0.00855	85.5	0.00935	93.5
820Ω	1	820Ω	10000	0.00855	85.5	0.00935	93.5
1kΩ	2	1kΩ	20000	0.00855	171	0.00935	187
2.7kΩ	1	2.7kΩ	10000	0.00855	85.5	0.00935	93.5
4.7kΩ	2	4.7kΩ	20000	0.00855	171	0.00935	187
6.8kΩ	1	6.8kΩ	10000	0.00855	85.5	0.00935	93.5
10kΩ	5	10kΩ	50000	0.00855	427.5	0.00935	467.5
22kΩ	1	22kΩ	10000	0.00855	85.5	0.00935	93.5
26kΩ	1	26kΩ	10000	0.00855	85.5	0.00935	93.5
39kΩ	3	39kΩ	30000	0.00855	256.5	0.00935	280.5
47kΩ	1	47kΩ	10000	0.00855	85.5	0.00935	93.5
100kΩ	1	100kΩ	10000	0.00855	85.5	0.00935	93.5
	20		200000				
*If we were to incorporate a power switch, there would be 1 extra 10kΩ				Tot	1710	Tot	1870
				Tot Price/Per	0.171	Tot Price/Pe	0.187

				Through Hole		Chip Caps	
Caps Used	Qty	Caps Used	Qty	Price/Unit	Price	Price/Unit	Price
0.01µF	2	0.01µF	20000	0.052	1040	0.0028	56
0.047µF	2	0.047µF	20000	0.052	1040	0.0028	56
0.1uF	2	0.1uF	20000	0.052	1040	0.0028	56
0.33µF	2	0.33µF	20000	0.052	1040	0.0028	56
1µF	4	1µF	40000	0.052	2080	0.0028	112
2.2nF	1	2.2nF	10000	0.052	520	0.0028	28
4.7µF	2	4.7µF	20000	0.052	1040	0.0028	56
10µF	1	10µF	10000	0.052	520	0.0028	28
47µF	1	47µF	10000	0.052	520	0.0028	28
				Tot	8840		476
				Tot Price/Per	0.884		0.0476

Table 2: Cost Breakdown

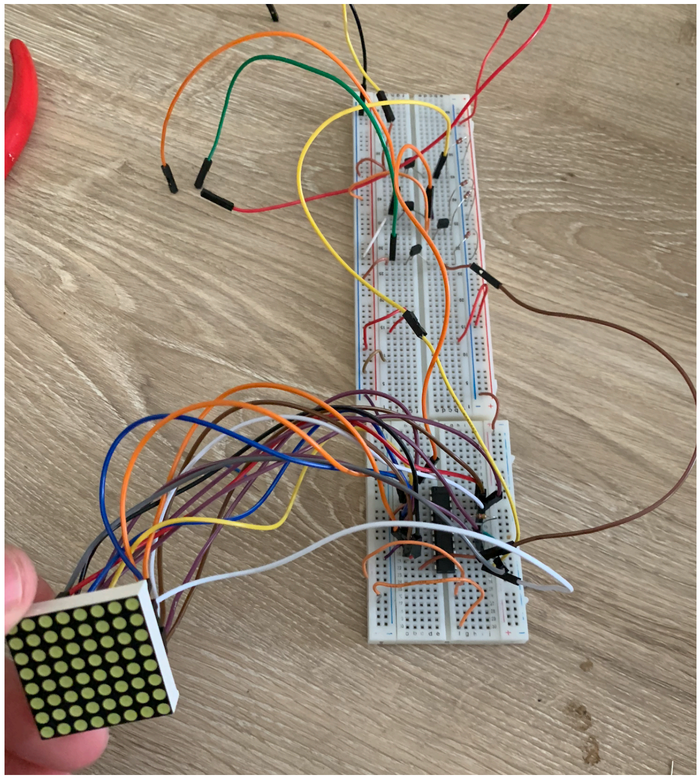

Figure 30: Attempt to use MOSFETs as Voltage Level Translator