

ILS — T2S

Project Report

Prabhkeerat Dhatt, Safwan Jamal, Cameron Faith

Group 7 | ENEL 300 | April 18

Table of Contents

[1.0 ACKNOWLEDGEMENTS 1](#_Toc6573146)

[2.0 INTRODUCTION 1](#_Toc6573147)

[2.1 OBJECTIVE 1](#_Toc6573148)

[3.0 TARGET USERS 2](#_Toc6573149)

[3.1 USER PERSONAS 2](#_Toc6573150)

[4.0 PLANNING 4](#_Toc6573151)

[4.1 PHYSICAL APPEARANCE 4](#_Toc6573152)

[4.2 TEXT-TO-PHONEME TRANSCRIPTION 5](#_Toc6573153)

[4.3 KEYBOARD 5](#_Toc6573154)

[4.4 FILTER DESIGN 6](#_Toc6573155)

[4.5 AMPLIFIER DESIGN 6](#_Toc6573156)

[5.0 DESIGN 7](#_Toc6573157)

[5.1 PHASE 1 7](#_Toc6573158)

[5.1.1 FILTER DESIGN 7](#_Toc6573159)

[5.1.2 AMPLIFIER DESIGN 8](#_Toc6573160)

[KEYBOARD CIRCUIT LAYOUT 10](#_Toc6573161)

[5.1.3 SOFTWARE DESIGN 11](#_Toc6573162)

[5.2 FEEDBACK 12](#_Toc6573163)

[5.3 PHASE 2 13](#_Toc6573164)

[5.3.1 NEW AMPLIFIER AND FILTER 13](#_Toc6573165)

[5.3.2 LCD DESIGN 15](#_Toc6573166)

[5.3.3 SOFTWARE UPDATES 16](#_Toc6573167)

[5.3.4 FINAL CASING 20](#_Toc6573168)

[6.0 CONCLUSION AND RETROSPECTIVE SUMMARY 22](#_Toc6573169)

[6.1.1 THINGS THAT WENT WELL 22](#_Toc6573170)

[6.1.2 THINGS THAT COULD HAVE BEEN IMPROVED 22](#_Toc6573171)

[7.0 REFERENCES 23](#_Toc6573172)

[8.0 APPENDIX A: IPA NOTATION AND TRANSCRIPTION RULES 24](#_Toc6573173)

[9.0 APPENDIX B: CIRCUIT COMPONENTS 25](#_Toc6573174)

[10.0 APPENDIX C: CIRCUIT SCHEMATICS FOR PRE-BUILT COMPONENTS 26](#_Toc6573175)

Figures

[Figure 1: Three different views of the 3D printed model illustrating our initial design. 4](#_Toc6573176)

[Figure 2: Original prototype dimensions. 5](#_Toc6573177)

[Figure 3: First band pass filter design. 8](#_Toc6573178)

[Figure 4: The planned design for our non-inverting voltage amplifier. 8](#_Toc6573179)

[Figure 5: The original circuit with design with an open circuit in place of Rf. 9](#_Toc6573180)

[Figure 6: The design of our operational amplifier with the appropriate resistor values. 10](#_Toc6573181)

[Figure 7: Keyboard circuit schematic. 11](#_Toc6573182)

[Figure 8: LM-386N pin layout. [5] 13](#_Toc6573183)

[Figure 9: Recommended LM-386 amplifier circuit. [5] 14](#_Toc6573184)

[Figure 10: Our LM-386 amplifier circuit without the 250uF capacitor in series. 14](#_Toc6573185)

[Figure 11: LCD pin layout. [6] 15](#_Toc6573186)

[Figure 12: LCD final pin layout. [6] 16](#_Toc6573187)

[Figure 13: Initialization of new pin numbers. 16](#_Toc6573188)

[Figure 14: Initialization of new variables. 17](#_Toc6573189)

[Figure 15: New code for processing "ENTER" input. 17](#_Toc6573190)

[Figure 16: New code for processing the "BACKSPACE" input. 18](#_Toc6573191)

[Figure 17: New code for printing characters to the LCD. 19](#_Toc6573192)

[Figure 18: A diagram illustrating the execution flow of the program that processes keyboard input. 20](#_Toc6573193)

[Figure 19: CAD model of the final casing. 21](#_Toc6573194)

[Figure 20: Dimensions of the casing. 21](#_Toc6573195)

[Figure 21: Schematic for the LM386N Low Voltage Audio Power Amplifier. 27](#_Toc6573196)

Tables:

[Table 1: IPA notation for consonants. 24](#_Toc6573197)

[Table 2: IPA notation for vowels and diphthongs. 24](#_Toc6573198)

[Table 3: Components list. 25](#_Toc6573199)

# ACKNOWLEDGEMENTS

We’d like to thank each of our professors, teaching assistants, and lab technicians for helping us reach our current ability and level of knowledge. We would not be able to do this project without your advice and guidance.

# INTRODUCTION

The ILS T2S is a *text-to-speech* (T2S) keyboard designed to facilitate communication for those who have temporarily or permanently lost the ability to produce comprehensible speech.

## OBJECTIVE

The goal of this project is to create a communication device that mimics natural human speech. This means that we will combine intonation, word stress, vowel length in a manner that sounds like that of a human voice unlike many of the text-to-speech technologies presently on the market.

# TARGET USERS

Our main target audience comprises of individuals who have temporarily or permanently lost the ability to produce speech. This includes those who are either physically—those who lack the physiological components to produce speech—or selectively—those who are unwilling to speak in social situations due to trauma or severe anxiety—mute. Our secondary audience includes individuals who are learning the English language. Due to the highly unphonetic nature of English text, many individuals find themselves needing to constantly lookup the pronunciations of new words that they come across. Our device can act as a “vocal dictionary” to provide them with the correct pronunciations.

## USER PERSONAS

|  |  |
| --- | --- |
| Sam | |
| Background | Sam is a fifteen-year-old boy. As a child, he began stuttering whilst speaking. His peers at school teased him excessively about his disorder which causes him to develop severe anxiety towards speaking. His parents eventually made the decision to homeschool him and have him see a therapist. He has made significant progress with his therapist and while he speaks normally at home with his parents, he is still unable to speak openly in most social situations. |
| Needs | A method to communicate with those around him. |
| Goals | Easy and fast communication; affordable technology that doesn’t require too much effort and mimics natural human speech patterns. |
| Perceptions | Sam has tried other text-to-speech technologies in the past, but he did not like how they sounded so robotic and unnatural which brought unwanted attention. He does not believe technology can help him |
| Motivation | Sam is extremely lonely. His refusal to speak outside of his home has left him with very few friends. On the inside, Sam is just like other teenagers. He wants to talk to people and maintain a fulfilling social life. He also secretly hopes to have a loving girlfriend who accepts him. |

|  |  |
| --- | --- |
| Cassandra | |
| Background | Cassandra is a woman in her mid-20s that teaches English to adult students in Turkey. She was born in Canada and is fluent in both English and French. In high school, she took a German class and developed a love for languages which lead her to pursue a career in teaching English as a second language. She is deeply passionate about her work and she is dedicated to helping ensure her students are provided with the skills they need to be proficient speakers of the English language. |
| Needs | A tool to teacher her students about the pronunciation of different English words. |
| Goals | Affordable technology that is intuitive for Turkish speakers who are already familiar with typing the Latin alphabet. |
| Perceptions | Cassandra has researched the methods of instruction used by ESL teachers from around the world and she is open to trying new technologies. |
| Motivation | Cassandra wants to give her students the tools they need to be independent learners. She also understands that learning a new language takes a lot of effort and she wants to make the experience enjoyable for her students. |

|  |  |
| --- | --- |
| Fatima | |
| Background | Fatima is a woman in her late 50s. She had the misfortune of being involved in a car accident that caused her chest cavity to collapse onto her lungs. To deliver oxygen to her lungs, surgeons had to carry out a tracheostomy where they opened a hole in her windpipe. While Fatima is recovering from her accident, the medical apparatus prevents her from speaking. |
| Needs | A way to communicate with and let her needs be known to friends, family, and nurses in the hospital. |
| Goals | Minimal wrist movement. |
| Perceptions | Nurses provided Fatima with a white board to write out her thoughts, but Fatima finds it slow and painful to move her wrists. She is eager to try out any piece of technology that could help alleviate her troubles. |
| Motivation | While Fatima’s family comes to visit her every, she feels extremely lonely because she is unable to communicate with them. She also wants a way to communicate vocally so she can get the attention of nurses with less effort. |

# PLANNING

## PHYSICAL APPEARANCE

Our initial plan for the device was a compact keyboard attached to a box containing an LCD screen and a speaker. We wanted to keep the keyboard small to minimize the amount of wrist movement that would need to be carried out (ideally, the user would not have to move their wrists at all whilst typing). A 3D printed model of the initial design is shown in *Figure 1* along with its dimensions in *Figure 2*.



Figure : Three different views of the 3D printed model illustrating our initial design.

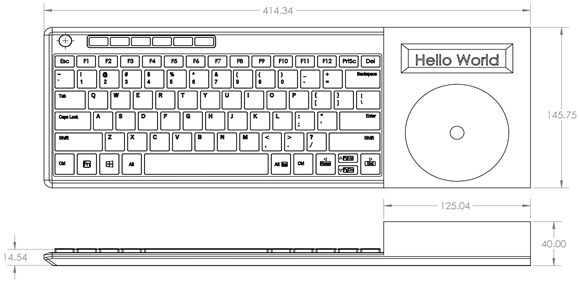


Figure : Original prototype dimensions.

## TEXT-TO-PHONEME TRANSCRIPTION

Our project will use a software algorithm to convert English text to a series of phonemes using notation from the International Phonetic Alphabet (IPA)—a table of symbols is provided in *Appendix A*. This programming will be done on the PIC16F1778 microcontroller that was provided to us for the project by our instructors.

The program will take English text and process it using a series of commonly employed spelling conventions. For example, the sequence of letters <*tion>* will be interpreted as \*ʃɪn*\ and the letters <*o*> and <*u*> will be default-in*terpreted* as \*oʊ\* and \ *ə*\ respectively, meaning that a word like <production> would be interpreted as \*proʊdəkʃɪn*\.

In the example above, angular brackets, <>, are used to denote English text whereas right-leaning slashes, \\, are used to denote phonetic transcriptions. This prevents misinterpretation between the two forms (e.g. <*bat*> does not correspond to /*bat*/). “Default-Interpretation” refers to the default value assigned to a letter if the program does not detect an alternative pronunciation using the surrounding letters (e.g. the letter <*i*> in <*bit*> will be interpreted as /*ɪ/*, but the letter <*i*> in <*kite*> will be interpreted as \a*ɪ\*, because the letter <e> at the end of <kite> indicates a particular pronunciation that differs from that of the default. In essence, the default-interpretation is used when no other rules apply.

## KEYBOARD

The PIC microcontroller has a port to accept serial input, but not for USB. To connect a keyboard, we’ll have to source a PS/2 keyboard since most modern keyboards use USB rather than serial.

## FILTER DESIGN

Through some preliminary research, we determined that the human voice ranges in frequency from around 100 Hz all the way to 3.5 kHz [1]. Since modern telephones range in frequency from 400 Hz to 3 kHz, we ultimately chose these values for our filter design as well [2]. Whilst calculating the necessary component values, we decided to set our capacitor values to those that we have available to us in class and then we solved for the resistor values seeing as resistors are easier to source at different values.

## AMPLIFIER DESIGN

For the initial stages of our project, we decided to use the same non-inverting amplifier that we designed in our *ENEL 343* class. If the amplifier proves to be insufficient for our purposes, we will adjust the resistor values accordingly to fit the needs of our project.

# DESIGN

The design section highlights the technical aspects of our project and how our device works.

## PHASE 1

The primary objective of phase 1 was to get the device working. For our group, this meant the speaker needed to output sound from the Arduino. We started by attempting to get the device to work with pre-recorded messages Next, we programmed the device to read user input from a laptop in the form of a string of characters. Once we got that working, we connected the Arduino directly to a serial keyboard and did some software modifications until we achieved the desired response.

### FILTER DESIGN

There wasn’t very much hardware design for phase 1 as all we needed was an amplifier, filter, keyboard and to connect them all together. To get the device to a testable state, we first designed the filter/amplifier so that we could start testing the audio immediately. While we already had a filter designed in a previous *ENEL 343* lab we opted to design our own so we could tailor the design specifically to the needs of our project.

For our high-pass filter, we chose a capacitor value of 0.1 microfarads and we ended up with a resistor value of 25000 ohms.





For our lowpass, filter, we chose a capacitor value of 0.01 microfarads and we ended up with a resistor value of 33333 ohms which we rounded down to 33000 ohms, a resistor value that was available to us in the lab. This had the effect of increasing the corner frequency to 3030 Hz which is an acceptable value for our purposes.







Our overall bandpass filter schematic is shown in the figure below.

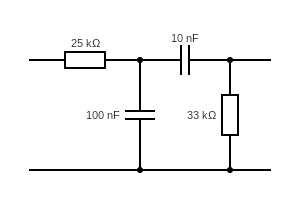


Figure : First band pass filter design.

### AMPLIFIER DESIGN

In order to amplify the volume of our speaker, we decided to use the non-inverting amplifier that we designed in our fourth hands-on exercise in *ENEL 343*. A schematic of the theoretical design is shown below in *Figure 4*.

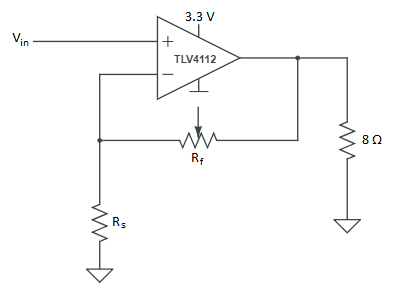


Figure : The planned design for our non-inverting voltage amplifier.

The design employed a TLV4112 operational amplifier with the following constraints:

* The load (i.e. the speaker) has an internal resistance of approximately 8 ohms.
* The max current that can be supplied by the TLV4112 is 300 milliamperes.
* To dynamically change the gain during circuit operation, a 10-kiloohm variable resistor will be used.
* The maximum voltage that the TLV4112 can supply is 3 V.

The voltage gain for a non-inverting operational amplifier is defined as:



As can be seen from the equation above, maximum voltage gain is observed when Rf is at it’s maximum; that it, when Rf is equal to 10 kiloohms.



When Rf is set to 10 kiloohms, the resistor can be approximated as an open circuit as shown in *Figure 5*.

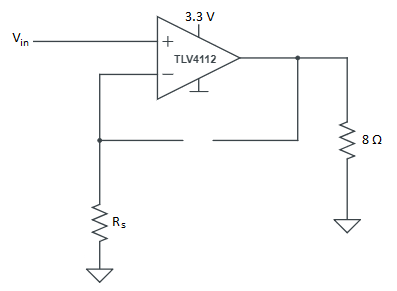


Figure : The original circuit with design with an open circuit in place of Rf.

At that point, the only current moving through the output resistor would be the current coming from the TLV4112. As stated in the constraints above, the maximum current that the TLV4112 can supply is 300 mA. Using this value, the maximum voltage that will be observed across the output resistor would be:



We will ultimately transfer our system from an Arduino Uno to the PIC microcontroller, so for the maximum gain calculations, we’ll use the maximum output voltage of the PIC, which is 825 microvolts. The maximum gain of the amplifier can then be calculated as:



Using this value, we can calculate the value the optimal value for Rs:





This value can then be rounded down to 4.7 kiloohms, which is the closest resistor value that we have available to us in the lab. This will have the effect of raising the gain to 3.1 which isn’t inappropriately distant in relation to the original calculation. The final amplifier schematic is shown below in *Figure 6*.

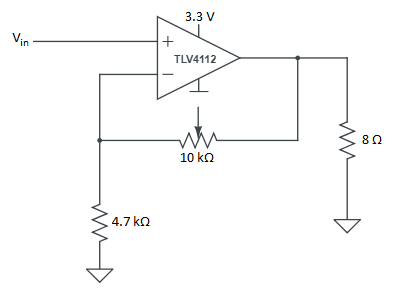


Figure : The design of our operational amplifier with the appropriate resistor values.

### KEYBOARD CIRCUIT LAYOUT

The keyboard design was quite simple as all we had to do was figure out the PS/2 conventions and find a way to hook up the keyboard to the Arduino. Firstly, we ordered a simple Female adaptor to connect the keyboard to and selected a DELL RT7D20 as our keyboard. Next, we selected the Arduino pins to use for the keyboard clock and data and simply connected it all together. A schematic can be found in the figure below.

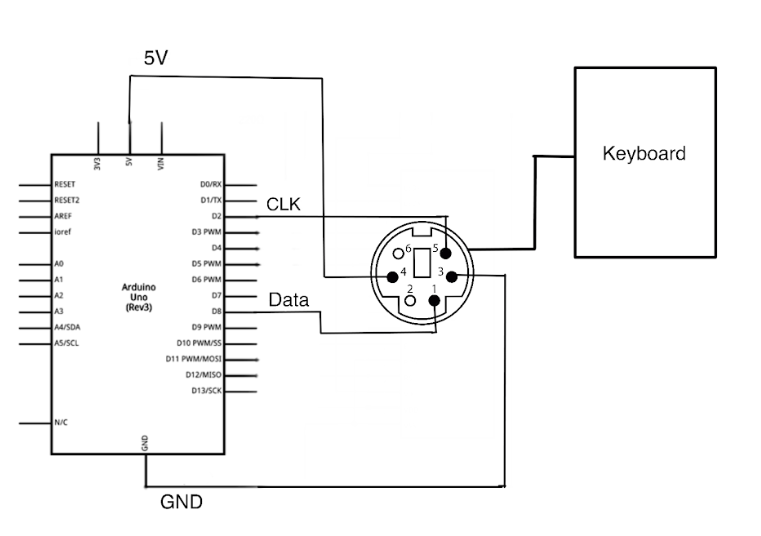


Figure : Keyboard circuit schematic.

### SOFTWARE DESIGN

With a working hardware design, we could easily develop the code to get our project working. At a high level, our code needs to first receive the inputs from the keyboard and then store that input in a buffer. Next, it will convert the buffer stream from characters into phonemes, one of the 42 sounds utilized in a typical Canadian accent. Those phonemes will then be converted into a series of mp3 files that will be outputted to the speaker.

We were able to source a text-to-speech library that is almost fully compatible with the Arduino. The library was found on GitHub and created by jscrane [3]. This library would give us a good framework to work off and create our project. This project worked very similarly to how we wanted to produce our device. The software has functions that take strings and use 271 English rules to convert into a series of phenomes and then into a series of binary arrays to play.

This software library gave our project a good foundation to work on. Once we had this working on our computer, we began to work on getting the PS/2 keyboard input working. The PS/2 convention of communication is quite simple. The Arduino will always produce a clock output and the data pin will only send signals once a key is pressed. All data is transmitted one byte at a time and each byte is sent in a frame consisting of 11-12 bits [4].  These bits are:

* 1 start bit.  This is always 0.
* 8 data bits, least significant bit first.
* 1 parity bit (odd parity).
* 1 stop bit.  This is always 1.
* 1 acknowledge bit (host-to-device communication only)

All we had to do was convert the bits received into ASCII characters and a couple of special characters.

Once the keyboard was fully functional, we developed code within the loop to process individual inputs and store them in a buffer array for output. Once the enter key is pressed, the code sends the buffer array to the TTS function to be processed for output.

## FEEDBACK

After presenting our phase 1 design, we received feedback from fellow students in the ILS program. We used this feedback to guide our work as we began phase 2 of our project.

One problem that we faced at the time was the minimal memory of the PIC microcontroller: our software used a lot more RAM than the PIC was able to supply. We got several responses telling us that if our program works sufficiently well on the Arduino, then we should either: (i) use the Arduino as our main processor; or (ii) use the Arduino in conjunction with the PIC to process our audio. We discussed the options with instructors and teaching assistants in the lab and we ultimately made the decision to pursue the former option and use the Arduino in place of the PIC microcontroller.

Another major problem that we had was that our audio was quite noisy, and our signal contained a lot of unnecessary frequencies outside of our desired range. The solution that other students and instructors suggested was to use a higher order filter that would have a sharper cutoff. Fellow students also suggested that since we didn’t have a buffer between the high-pass and lowpass components of our filter, the output voltage was being divided over the two resistors in the circuit. We made the decision to redesign our filter as a second-order filter with buffers in between the sub-filters.

A slightly less visible, but no less significant problem was that of the volume of our device. We had originally opted to employ the same amplifier that we designed during our fourth hands-on session in *ENEL 343*, but upon testing, it was quickly discovered that the Arduino Uno had a considerably different output voltage range from that of the PIC. Since we made the decision to use the Arduino Uno in place of the PIC microcontroller, we would also have to redesign our filter and amplifier to fit in with the new specifications. Going into phase 2 of our design cycle, we decided to test out a new amplifier that was created specifically for use on the Arduino Uno.

## PHASE 2

Phase 2 was an attempt to improve our design after testing. We wanted to improve the audio and add an LCD display to show keyboard input.

### NEW AMPLIFIER AND FILTER

The old amplifier and filter design needed to be improved. The design was very complicated and didn’t produce loud enough audio. To find the best design, our team opted to conduct some research on the best options for our device. The op-amp we finally decided to use was the LM-386N. Seen in the figure below.

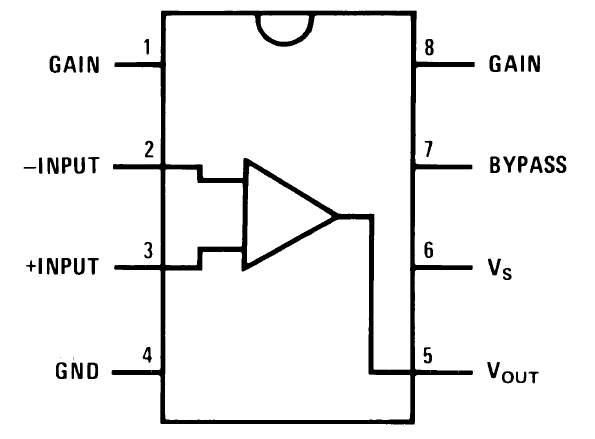


Figure : LM-386N pin layout. [5]

We chose this amplifier as it would allow us to adjust the gain of the amplifier from 0 to 200 dB very easily while still being able to produce enough current to drive the speaker. For our initial design, we opted to use the recommended schematic found on the specification sheet [5], shown below.

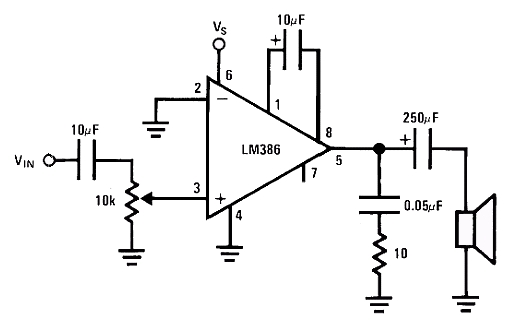


Figure : Recommended LM-386 amplifier circuit. [5]

This design worked well, however, we found the capacitor in series with the speaker created too much noise when the Arduino wasn’t producing audio. We decided to remove the capacitor and use the final schematic below.

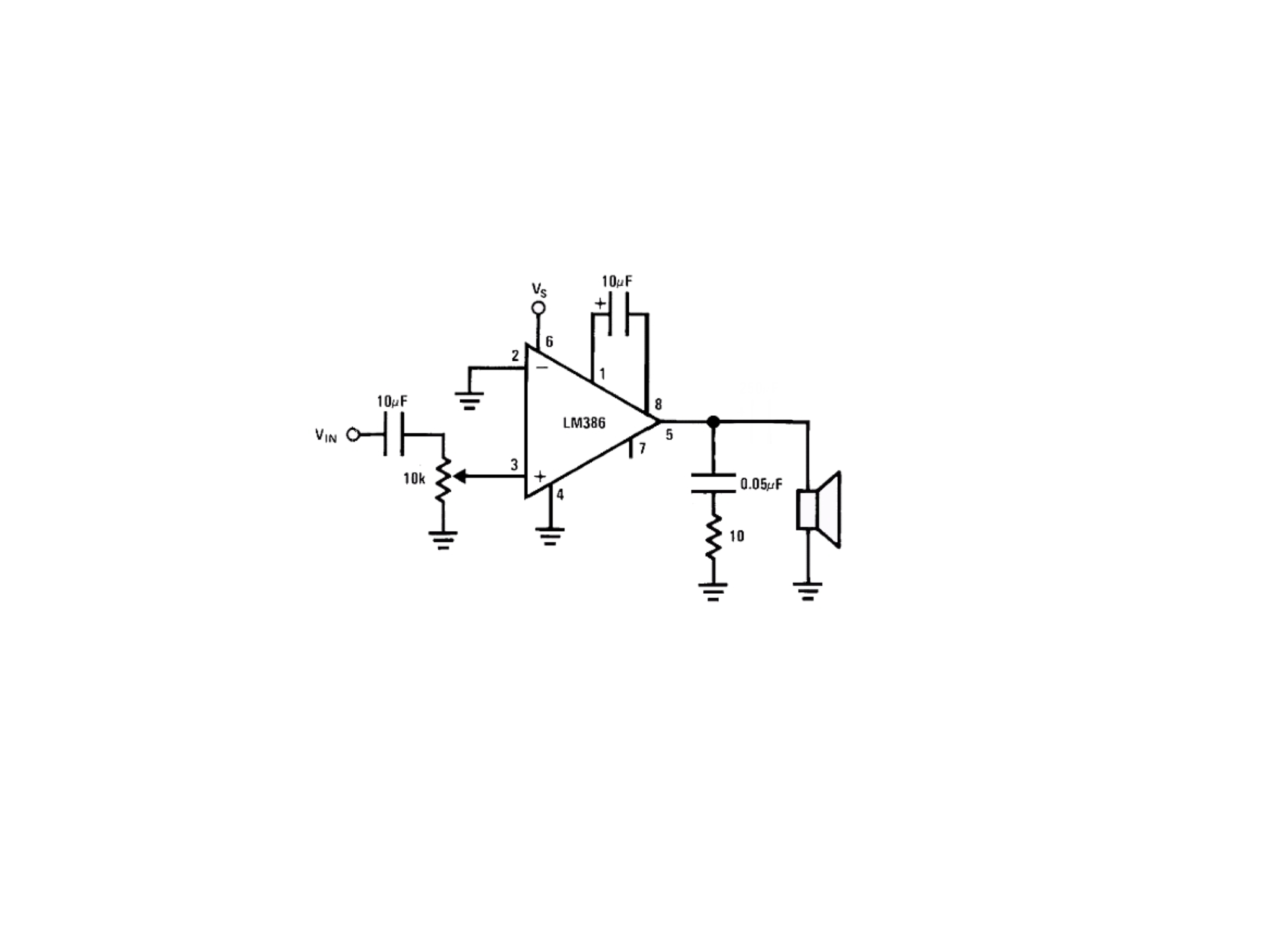


Figure : Our LM-386 amplifier circuit without the 250uF capacitor in series.

### LCD DESIGN

For the user to see the keystrokes, it was critical to have an LCD or visual display to see the text being inputted. Our group decided to use a parallel LCD design as we had one on hand and could immediately start to use it. A schematic of this can be seen below.

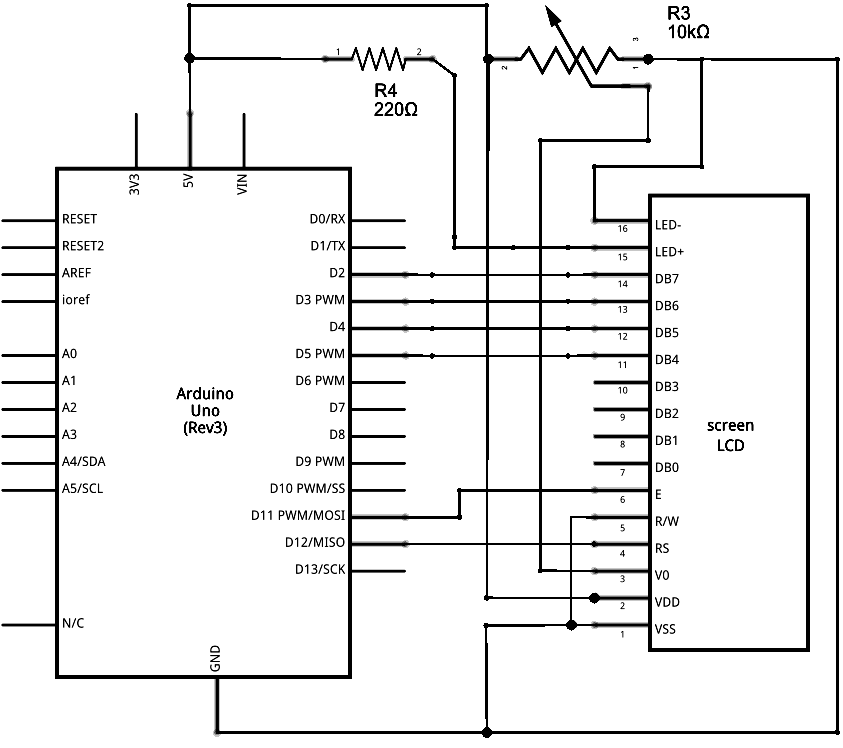


Figure : LCD pin layout. [6]

We eventually moved pins D2-D5 to pins D4-D7 to keep the original pin layout we had before the LCD. We also removed the 220 Ohm resistor to get a brighter LCD display. This can be seen in the figure below.

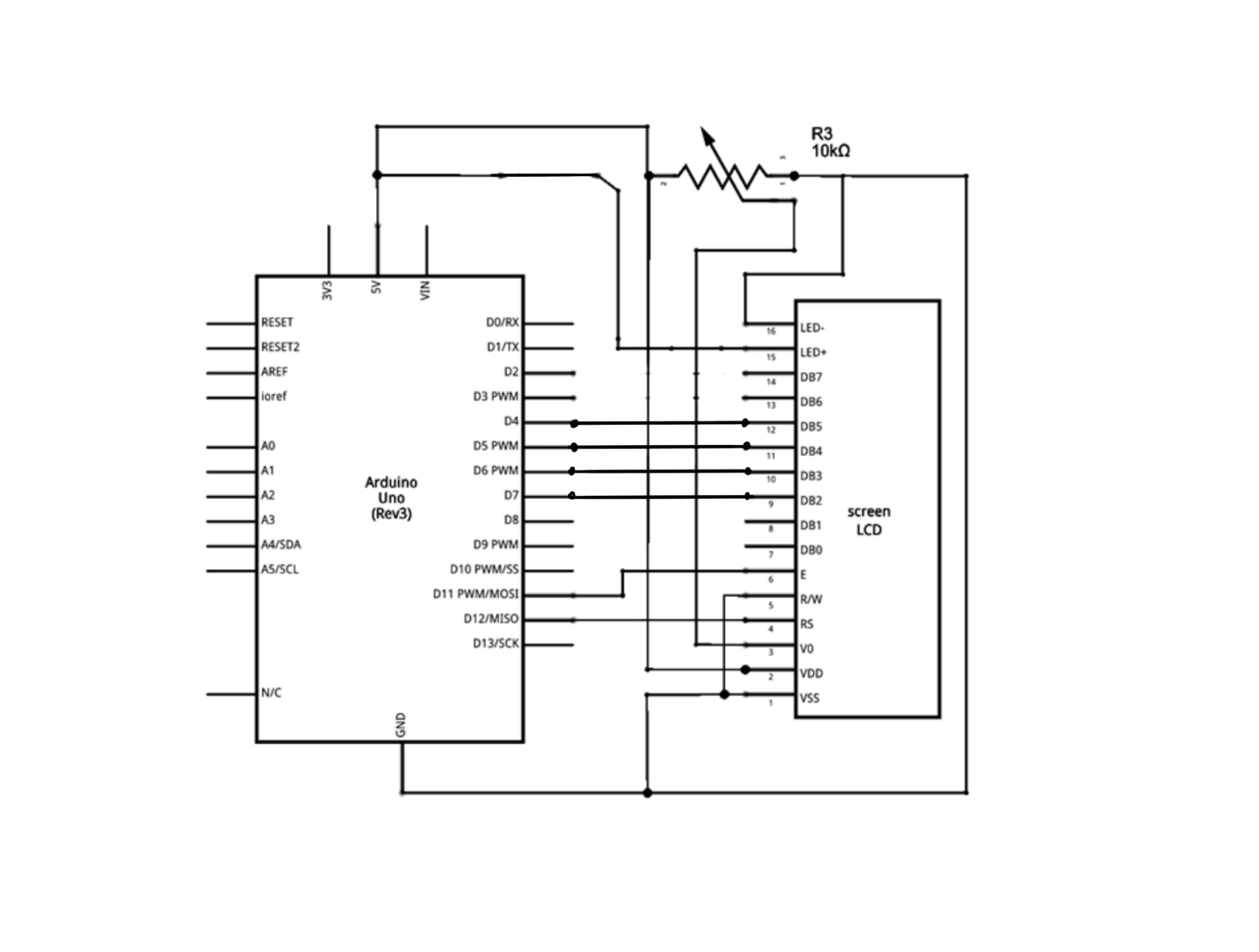


Figure : LCD final pin layout. [6]

### SOFTWARE UPDATES

There were no software adjustments needed for the new amplifier circuitry as we didn’t adjust the pin layouts. However, we needed to incorporate the new code to update the LCD whenever the keyboard was pressed. This meant we needed to print any of the alphanumeric characters as well as remove characters when we pressed delete or “ENTER.” Firstly, we needed to initialize both the new LCD and the new variables we needed. This can be seen in the figures below.

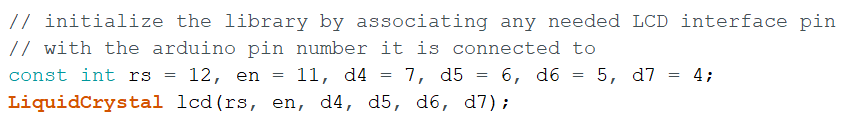


Figure : Initialization of new pin numbers.

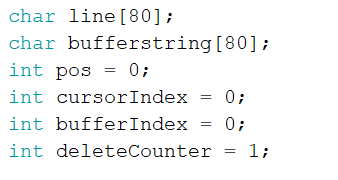


Figure : Initialization of new variables.

cursorIndex and deleteCounter were both needed to keep track of what’s on the screen at any moment. The new code to deal with the “ENTER” input can be seen in the figure below.

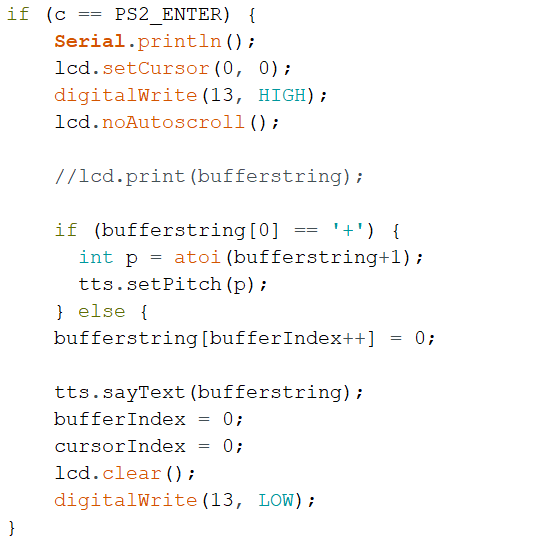


Figure : New code for processing "ENTER" input.

In summary, this code now also clears the screen once the audio has been outputted and resets the LCD into its “startup mode” by setting the cursor index to zero and turning off “autoscroll.” Autoscroll allows the LCD to print more characters by shifting all characters to the left once a new character is printed. The “DELETE” section can be found below.

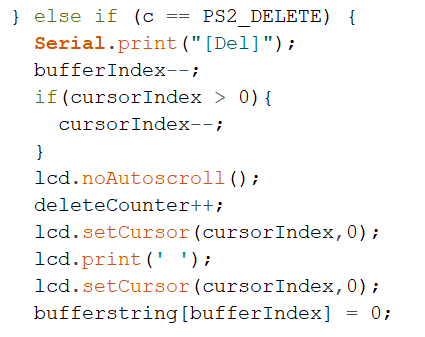


Figure : New code for processing the "BACKSPACE" input.

In summary, the updates to this code replace previous characters with spaces to erase them from the display and allowing the user to see the effect of the delete. Regular character output can be found in the figure below.

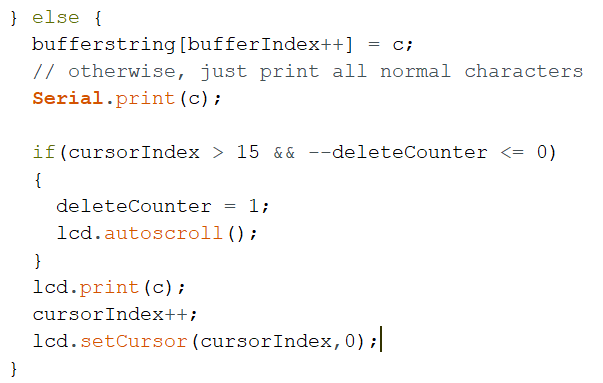


Figure : New code for printing characters to the LCD.

In summary, the updates simply print the characters to the screen, but once there have been 15 characters printed to the screen it also turns on auto scroll.

A final block diagram of how the software works can be found below in *Figure 15.*

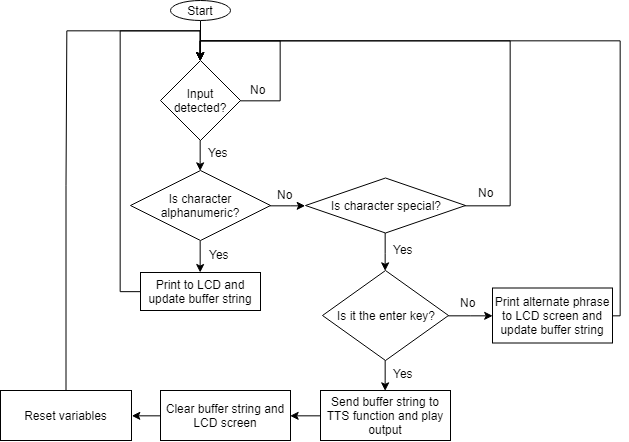


Figure : A diagram illustrating the execution flow of the program that processes keyboard input.

In the diagram above, “special characters” refers to the TAB key and the four arrow keys. These keys are set to output certain pre-programmed phrases.

### FINAL CASING

To make our device presentable, we opted to design a casing for our final product. The casing needed to be small enough to not look bulky while still being large enough to hold all our components. We needed to have to display fully visible as well as have holes for the speaker, keyboard connection and power connections. Our final CAD model can be seen in the figures below.

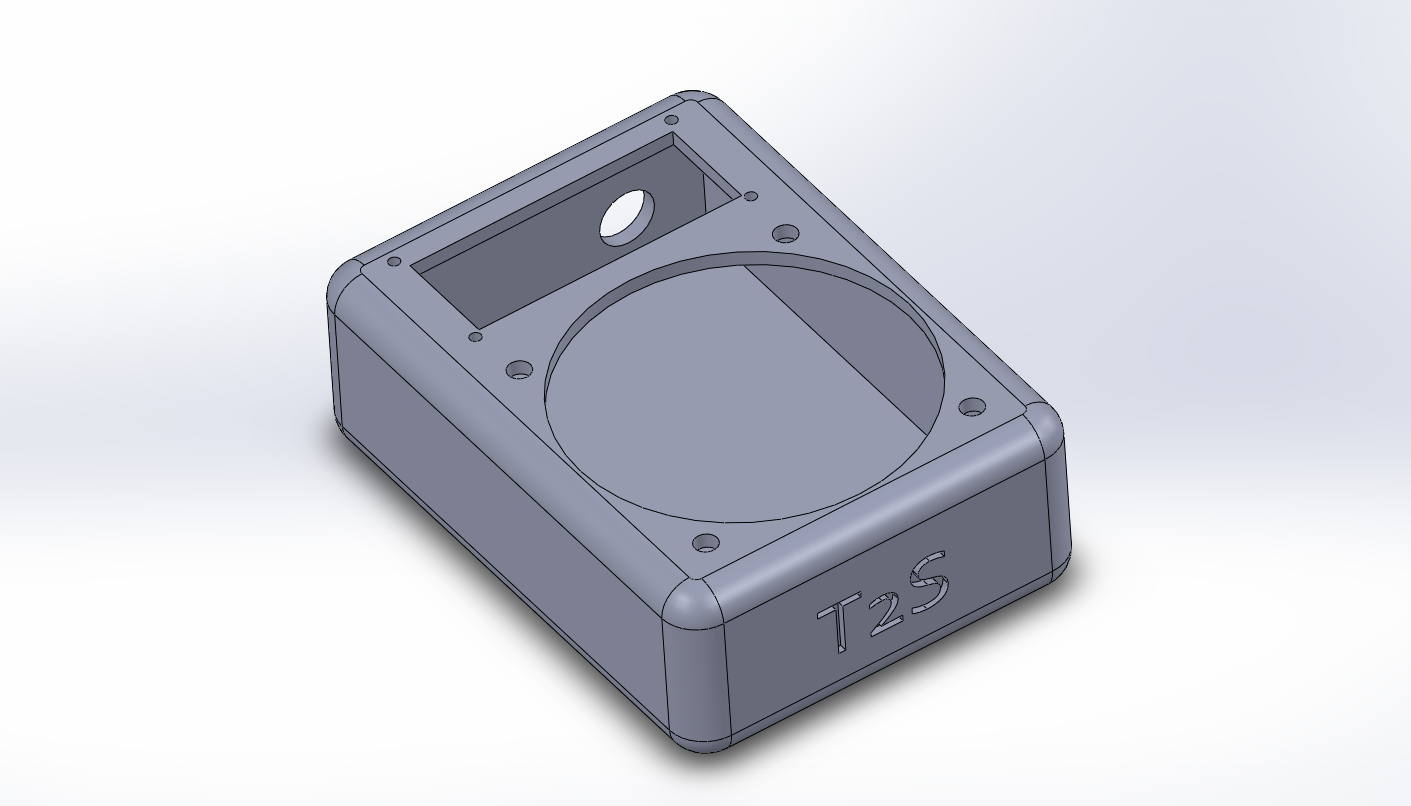


Figure : CAD model of the final casing.

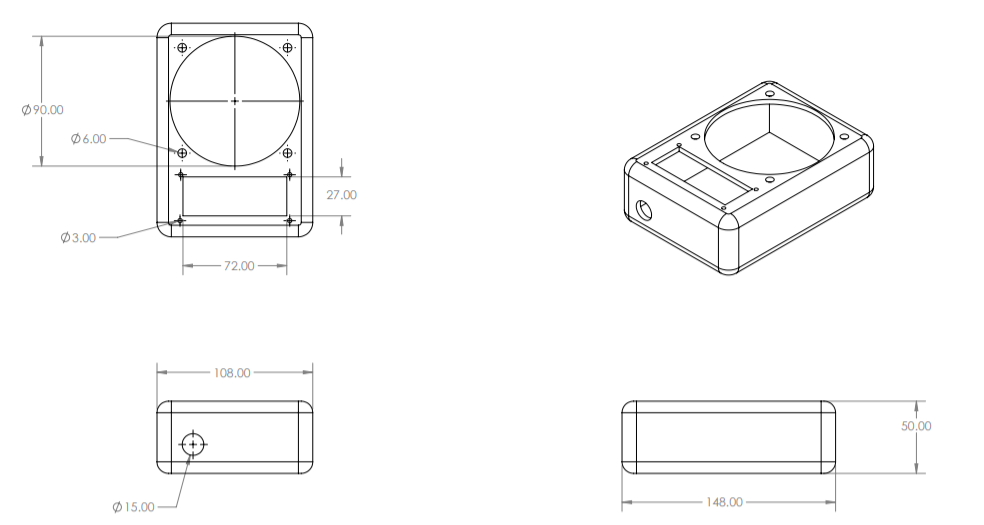


Figure : Dimensions of the casing.

# CONCLUSION AND RETROSPECTIVE SUMMARY

Overall, we achieved a vast majority of what we set out to do. In its current state, our project is an achievement that represents three weeks’ worth of work that we are proud to put our names behind. While our audio quality is not quite comprehensible to most users of the product, we were able to design a working amplifier, process keyboard input, print output to an LCD screen, and create an aesthetic casing to hold our circuitry.

### THINGS THAT WENT WELL

Our team worked together in a very effective manner and we were able to delegate tasks efficiently through one-on-one discussions, digital messaging, and our daily stand-ups. We were able to finish the major hardware and software components of our project almost a week before we were set to present our project which left us plenty of time to adjust and work on planning our produ5ct pitch.

### THINGS THAT COULD HAVE BEEN IMPROVED

While we *were* able to delegate tasks effectively, a common theme that emerged was that we’d have one person working on hardware, one person working on software, and one person working on documentation/presentation. On the surface, this doesn’t seem like a bad arrangement since we were able to work effectively to achieve a finished product; nevertheless, by separating ourselves into our own little niches, we missed out on a lot of learning experiences. If we had split up the task areas equally between the three of us, we would have been more well-rounded.

# REFERENCES

|  |  |
| --- | --- |
| [1] | "Human Voice Frequency Range," [Online]. Available: http://www.seaindia.in/blog/human-voice-frequency-range/. |
| [2] | "How Stuff Works," [Online]. Available: https://electronics.howstuffworks.com/telephone6.htm. |
| [3] | jscrane, "TTS," [Online]. Available: https://github.com/jscrane/TTS. |
| [4] | "PS2 Protocol," [Online]. Available: http://www.burtonsys.com/ps2\_chapweske.htm. |
| [5] | "TI LM-386," [Online]. Available: http://www.ti.com/lit/ds/symlink/lm386.pdf. |
| [6] | "Fritzing," [Online]. Available: http://fritzing.org/home/. |

# APPENDIX A: IPA NOTATION AND TRANSCRIPTION RULES

Table : IPA notation for consonants.

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Reference** | **Symbol** | **Reference** |
| \b\ | ***b***at | \s\ | ***s***ack |
| \p\ | ***p***ine | \z\ | ***z***oo |
| \k\ | ***k***ite | \θ\ | ***th***ing |
| \d\ | ***d***affodil | \ð\ | ***th***at |
| \t\ | ***t***as***t***e | \r\ | ***r***ight |
| \f\ | ***f***air | \w\ | ***w***eb |
| \g\ | ***g***as | \l\ | ***l***ie |
| \v\ | ***v***ibrate | \j\ | ***y***ellow |
| \h\ | ***h***at | \n\ | ***n***ight |
| \ʃ\ | ***sh***ine | \tʃ\ | ***ch***ief |
| \ʒ\ | plea***s***ure | \dʒ\ | ***j***ail |
| \ŋ\ | bli***ng*** | \m\ | ***m***ine |

Table : IPA notation for vowels and diphthongs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Reference** | **Symbol** | **Reference** |
| \i\ | f***ea***t | \ʊ\ | b***oo***k |
| \ɪ\ | b***i***t | \oʊ\ | b***oa***t |
| \eɪ\ | l***a***te | \ɒ\ | l***o***t |
| \ɛ\ | l***e***t | \a\ | ***a***rmour |
| \æ\ | f***a***t | \ə\ | b***u***t |
| \o\ | ***o***rder | \ɚ\ | bett***er*** |

# APPENDIX B: CIRCUIT COMPONENTS

Table : Components list.

|  |  |  |  |
| --- | --- | --- | --- |
| Component Description | Quantity | Price ($ CAD) | Method of Acquirement |
| E1115 PS/2 Keyboard to ASCII Converter | 1 | 17.95 | Ordered on Adafruit |
| 27KΩ ¼ W Resistor | 4 | 0.40 | Ordered on DigiKey |
| ALUM 100uF 16V Cap | 2 | 0.88 | Ordered on DigiKey |
| ALUM 10uF 16V Cap | 4 | 1.92 | Ordered on DigiKey |
| 0.325 W MONO 8DIP IC Op-AMP | 4 | 5.46 | Ordered on DigiKey |
| MIKROE-55- Display Board, Character LCD 2x16, Blue Backlight | 1 | 8.82 | Ordered on Newark |
| 10KΩ ¼ W Resistor | 4 | N/A | ILS Classroom |
| 10Ω ¼ W Resistor | 4 | N/A | ILS Classroom |
| Polar 250uF 50V Cap | 1 | N/A | ILS Classroom |
| Polar 10uF 50V Cap | 1 | N/A | ILS Classroom |
| Polar 0.05uF Cap | 1 | N/A | ILS Classroom |
| 1 W 8DIP IC LM386 Op-Amp | 1 | N/A | ILS Classroom |
| 220Ω ¼ W Resistor | 1 | N/A | ILS Classroom |
| 25kΩ ¼ W Resistor | 1 | N/A | ILS Classroom |
| 33kΩ ¼ W Resistor | 1 | N/A | ILS Classroom |
| 8Ω ¼ W Resistor | 1 | N/A | ILS Classroom |
| Polar 100uF 50V Cap | 1 | N/A | ILS Classroom |
| Polar 10uF 50V Cap | 1 | N/A | ILS Classroom |

# APPENDIX C: CIRCUIT SCHEMATICS FOR PRE-BUILT COMPONENTS

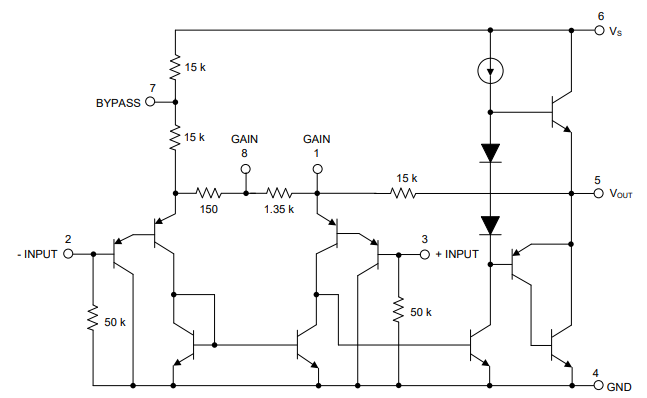


Figure : Schematic for the LM386N Low Voltage Audio Power Amplifier.