Wearable Communication Device for Deaf and Mute

By

Andrew Ko, hyunjun5@illinois.edu

Min Lee, Minhol2@illinois.edu

Yihan Ruan, yihanr2@illinois.edu

Final Report for ECE 445, Senior Design, [Fall 2021]

TA: Stasiu Chyczewski

December 2021

Project No. 19

Abstract

The wearable communication device is designed for deaf and mute people to communicate efficiently with their majority counterparts. The device has three main components: the power supply unit, control unit, and the keyboard unit which are combined to serve the users. Many people with hearing or vocal problems tend to have a difficult time exchanging words with others and the wearable communication device provides a simple solution by mimicking vocal communications with a stenographic keyboard and STT and TTS. The main competitor would be phones with apps that take advantage of Speech-to-Text and Text-to-Speech but most of them require an internet connection or do not allow real time communication because they only support screen keyboard.

Contents

1. Introduction

1.1 Purpose

Technology is evolving rapidly and provides benefits in almost every aspect of our lives. However, most solutions aim to make the majority, people without disabilities, feel comfortable where they can make more profits. Consequently, we thought it was worthwhile to design our project to focus on the people with disabilities to enhance the comfort in their everyday lives. Most of the time, random encounters and socializing proves difficult for deaf and mute people as sign languages are not one of the popular options in the verbally dominant world. In order to help with the cause and prevent discouragement, our wearable communication device will provide a new approach. The users will also be able to see what they type in, essentially on what is like one's regular online chat room. This provides deaf and mute people to have conversations with people other than just those around them who are proficient in sign language.

1.2 Functionality

There are many alternatives in which complete ideas can communicate with people without disabilities, but most of the time, they either require the internet or a phone to have certain apps downloaded at the time such as Google translate. However, this is an unlikely scenario for many deaf/mute people. Also, the typing on your phone can never get as fast as to allow for real time communication. With our wearable device, It is convenient and helps the mute/deaf communicate vocally real time. The device is attached with a steno keyboard that allows the user to type at the talking speed which is then converted immediately to speech through TTS and played through the speaker on the belt. The interactor is designed to respond via talking to the built-in mic with the display module that will put the speaker's words on screen.

1.3 Subsystem Overview

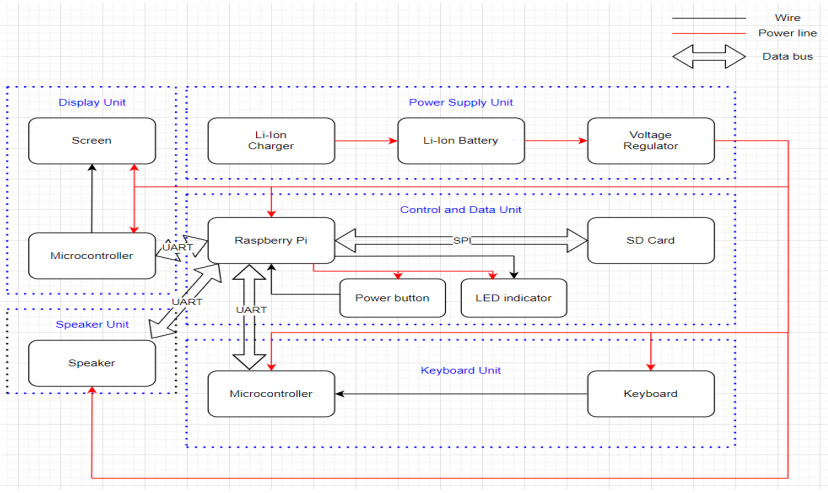

Fig 1.1 Block Diagram for the project

The wearable communication device can be subdivided into three main subsystems which include the power supply unit, control unit with speaker, raspberry pi and display, and finally the keyboard unit.

The Display unit is attached on top of the raspberry pi and connected via the pins on the board along with the speaker which is USB connected to one of ports on pi. All of these are under the same enclosure that together controls the device. The control unit lies on the side of the belt and is left hanging on a retractable wire so that the user can hand it over to the interactor and only left hanging when not in use.

The keyboard unit has its separate housing and is located in the middle of the belt. The keyboard is connected to the control unit via a USB C to B cable that allows data transfer and power at the same time.

The power supply unit is located on the other side of the belt, to the opposite of where the control unit resides and powers the raspberry pi in the control unit which then powers the rest of the external devices such as keyboard, speaker, microphone and the display.

2 Design

2.1 Power Supply

2.1.1 Design procedure

We wanted a power supply that could support our system consistently, and run for a few hours. The expected power consumption of each component is in the table below.

Table 2.1 Expected power consumption of components

Devices	**Voltage/Current requirement**
Raspberry Pi 4B	5V±5%, 1.25A (Max), 1.2A (Avg.)
Keyboard	5V±5%, (mcu max 27mA)
Display	5V±5%, 380mA
Mic	1.8-3.3V, 0.8mA
Speaker	5V±5%, 100mA

From the table, we can see that the total power consumption is about 1.7A at max. Therefore, we aimed to have a power supply that can output 5V at 3A.
Also, it would be convenient for the user to be able to charge the power supply through a simple wall charger. Thus, charging was the second feature of our power supply.

2.1.2 Design details

We used four 3.7V 18650 batteries with 2600 mAh capacity in parallel for a total of 10400 mAh capacity. This means the power supply can run for at least 6 hours at full charge. The most important part of our power supply is the consistent 5V at 2A output. To achieve this, we made a voltage regulator.

The implementation of the LTC1700 as voltage regulator required some calculations. Looking at Appendix B figure 1, there are a N-channel and a P-channel MOSFET for the output current limit. The N-channel is turned on while the P-channel is off during normal operations, and P-channel turns on when N-channel is off. The P-channel is turned on until either inductor current is reversed or the next duty cycle begins.
The mosfets were selected based on their R_{DSON}. The following equations are used to calculate the required on-resistance of the mosfet.

First, the duty cycle of mosfet is calculated by the following equation

$$1 - V_{IN}/V_{OUT} = Duty\ Cycle \quad (2.1)$$

Using equation 2.1, we can get the duty cycle which is 16 ~ 26 % for 3.7 and 4.2 V input. We also needed to consider 4.2V input since the 3.7V battery cell has voltage of 4.2V at full charge. Then, we use the equation for $L_{MINBURST}$ which is needed for duty cycle below 36%.

$$L_{MINBURST} = \frac{V_{IN(MAX)}(DC)}{(f)(0.66)\left(\frac{I_{OMAX}}{1-DC}\right)} \quad (2.2)$$

Where: frequency = 530k Hz.

Then, $L_{MINBURST}$ = 0.807 uH. This is the minimum value for our inductor

For the Burst Mode operation, we need low ripple current of around .4*I_{0MAX} = .8A Though our minimum inductor size is .807 uH, we will be using 1.8uH to further reduce ripple current. With the inductor of 1.8uH, we use the equation below for actual ripple current.

$$\Delta I_L = V_{IN}\left(\frac{DC}{fL}\right) \quad (2.3)$$

Where: V_{IN} = 3.7V and 4.2V, L = 1.8uH

Then, the ripple current ΔI_L = 1A (3.7V), 0.7A (4.2V)

Now, to get the $R_{DS(ON)}$ we use the following equation.

$$R_{DS(ON)(MAX)} \cong \frac{\Delta V_{SENSE}}{\left(\frac{I_{O(MAX)}}{1-DC} + \frac{1}{2}\Delta I_L\right)(\rho_T)} \quad (2.4)$$

Where: ΔVSENSE = 63mV and ρ ≅ 1.2

For N-channel, $R_{DS(ON)(N-CHANNEL)} = \frac{63mV}{\frac{I_{O(MAX)}}{1-D} + 0.5(\Delta I_L)}$ = 3.2 mOhm

For P-Channel, $R_{DS(ON)(P-CHANNEL)}$ = (3.2)(.9) = 2.88mOhm

Where the .9 is from fig. 2.2, provided from the LTC1700 datasheet with duty cycle calculated above.

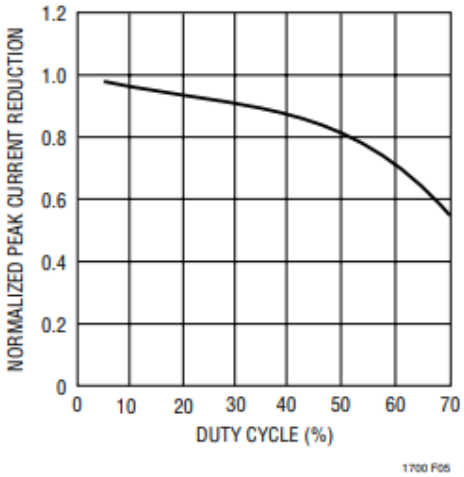

Fig 2.1 Max output current vs. Duty Cycle

The BG pin connected to the gate of the N-channel, has voltage swing between 0 to 5V, so out Vgs = 5V. The average current through the MOSFET is 1.3A.
Also, the TG pin connected to the gate of the P-channel has the same voltage swing, so our Vgs = 5V. The current through the P-channel is 2A.

To obtain the peak current of the inductor, we use the following equation 2.5.

$$I_{O(MAX)} = (I_{PK} - 0.5\Delta I)(1 - DC) \quad (2.5)$$

Where: I_{PK} = Peak Inductor Current | ΔI = Inductor Ripple Current | DC = Duty Cycle
Using the equation 2.5, we can get I_{PK} = 2.73A (4.2V), 3.2A (3.7V). Thus, we needed an inductor that does not saturate at 3.2A.

A power switch was added to disconnect the voltage regulator from the output, which will function as a power on/off switch.

Appendix B Figure 2 and 3 are the schematic of our charger and battery protection design. These were not implemented in the final demo prototype since we lacked time to debug the problems and redesign the circuit.

2.2 Stenographic Keyboard

2.2.1 Design procedure

Most important choice to make in designing the keyboard was the Microcontroller. The Microcontroller used in the stenographic keyboard is ATMega32u4 which has enough I/O pins to support the whole keyboard matrix and the microcontroller also runs at 16MHz from an external crystal oscillator with 22pF load capacitors. The values of the decoupling capacitors for the VCC pins

of the Microcontroller were determined based on the recommended values in the ATMega32u4 manual available online.

In programming the microcontroller to work like a keyboard, we used the QMK Firmware which is an open source software that allows developers to make their own custom keyboards. we modified an existing keyboard layout to match our needs for stenographic keyboard, the keymaps etc and finally flashed onto the microcontroller.

In order to complete the design of the stenographic keyboard, we needed to install “plover” which is an open source stenography dictionary/program that allows any keyboard to change its layout to be able to support stenographic style of typing with simultaneous keypresses.

2.2.2 Design details

After assigning four 0.1u capacitors and one 4.7u capacitor as decoupling capacitors for each of the Microcontroller’s VCC pins and adding a 16MHz crystal oscillator, next was to add a reset circuit for when we need to update the firmware to remap the keys as we update our user interface of the control module. (Fig. 4 of Appendix B) The AVR Microcontroller Hardware Design Considerations document recommends the reset circuit as Fig 2.3

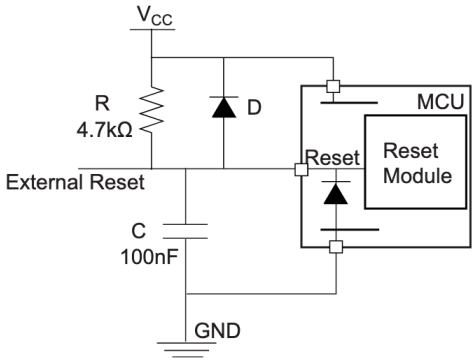

Fig 2.3 recommended reset circuit for the Microcontroller

Next was connecting a USB connector to the microcontroller and we started with connecting 22 ohms resistors as the ATMega32u4 manual requires them. Then It was necessary to ground the SBUS1 and SBUS2 pins which are only for transporting HDMI or thunderbolt signals that are not within the scope of this keyboard. To detect cable attachment and removal, channel configuration pins or CCS1 and CC2 pins need to be connected to Ground with 5.1k ohms resistors based on the USB standard 500mA. Last but not least, The usb connector needs an ESD protection system which left us with two options, either use rail to rail discrete diodes or use an IC such as PRTR5VoU2X which was the easier option.

Finally, the last part of the schematics (Fig. 5 of Appendix B) involve the row and column matrix of diodes which are where the keyboard switches will be soldered and detect keypresses so that the microcontroller can send information about which characters to be displayed on screen to

form sentences. The diodes are present to prevent ghost key presses by blocking the reverse flow of the current.

2.3 Software/Control

2.3.1 Design procedure

For our device, the software unit can be divided into three parts: STT module, TTS module and user interface. Those three modules will all be implemented using python code. Besides, In order to have a wearable device, we will use an SD card to store all python codes and corresponding libraries. We combined those three modules into one python file, which will be run in the terminal. STT and TTS modules will be called when we press certain keys in the user interface.

Table 2.2 Library used for three modules

Module	Library
STT	**Vosk**
TTS	**Pico**
User Interface	**Tkinter**

2.3.2 Design details

The method we use to test different TTS modules is to have one person wearing the headphone and the other person typing random words. We rate the sound quality on various modules based on how well the listener understands typed messages. Overall, we decided to choose Pico TTS for our project.

Table 2.3 TTS modules on raspberry operating system

TTS Module	Sound quality (out of 5)	Internet
Pyttsx3	**2.5**	**Offline**
Festival	**3.5**	**Offline**
eSpeak	**2.5**	**Offline**
Google TTS	**4.5**	**Internet required**
Pico	**4**	**Offline**

Since we decided to use existing software for STT and TTS, those two parts did not involve too many design choices. Creating the user interface, on the other hand, includes most of our design decisions for software parts. For example, instead of other keys, we decide to use function keys to start, switch and end the program, which will not be influenced by any typing during TTS. Moreover, instead of just showing the current sentence transferred from STT or TTS, we decided to display a chat history, which will include the latest four sentences. Besides, we also used distinct colors to differentiate messages from STT and TTS. We choose black for sentences transferred from STT and blue for sentences that will be used for TTS. Figure 2.5 shows the flowchart we designed for our user interface.

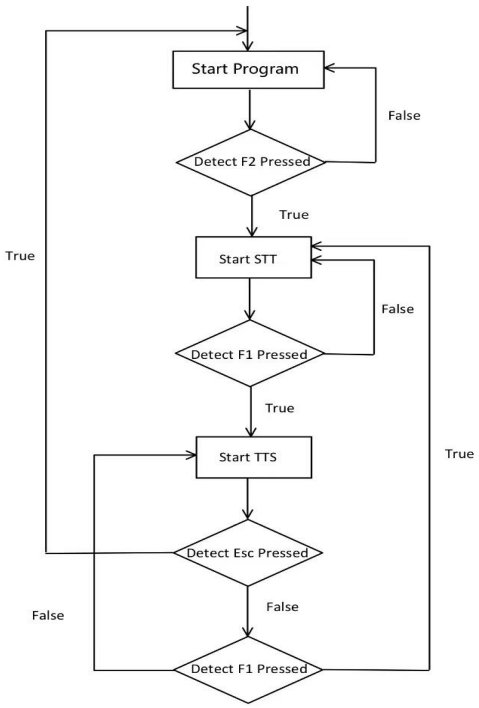

Fig 2.5 Flowchart for User Interface

2.4 Audio Devices

2.4.1 Design procedure

Our device needed a speaker that can output audio files from a TTS software. It was initially meant to be attached to the belt, thus requiring a volume enough for the interactor to hear. However, in the middle of the project, we decided to combine the display and the control module together. This means that if we want the speaker on the belt, we will have to run another long wire from the control unit. Therefore, we decided to keep the speaker on the control unit with a smaller volume.

We also needed a microphone that can take the audio input into the STT software. Initially, we planned on using a small microphone module with a 3.5mm jack input, but it was later found that the pi does not take microphone input via the 3.5mm jack it has. Due to lack of time, we decided to use a small third party USB powered microphone.

2.4.2 Design details

Our microphone was complete as is, but there was a problem with the speaker. We did not consider the need for an amplifier, and thought the speaker itself would have volume big enough for our purpose. However, the volume of the speaker was too small and we had to use an amplifier. Due to limitations on the time and schedule, we could not build the amplifier circuit out of our own PCB. The amplifier circuit was bought from Adafruit and used in conjunction with our choice of 8 Ohm speaker.

3 Design Requirements & Verifications

3.1Power Supply

Looking at Appendix A table 1, the power supply needed to provide Raspberry Pi with 5V with a switching current limit of 2A. The total battery capacity of 10400 mAh needed to run the device for approximately 6 hours of use at full load.

When tested using a constant current load circuit and measured using a voltmeter, the voltage regulator was providing correct 5V with maximum 2.32A. We were expecting a total of around 1.7A draw when connected to Pi, but the Pi and other modules combined were only drawing about 1.5A on average and 1.7A at peak. When tested at 1.8A current draw, the power bank lasted about 6.5 hours. Also, even though 5V was the requirement described in the Raspberry Pi website, we were seeing a low voltage warning when the load voltage was at 5.02V. Upon more research, 5.1V was the true requirement for the Pi. Thus we modified the voltage regulator to generate 5.12V.

3.2 Stenographic Keyboard

Based on the table 2 keyboard of Appendix A, the stenographic keyboard has two major requirements that need to be verified. They should have enough keys to contain the stenography layout and the function keys assigned to control the user interface to allow for switching between the speaker and playing the introduction message, etc. Also, the keyboard should be able to support simultaneous key presses to support plover which runs the translation to get output from stenographic inputs.

The verification for the requirements were done after the design was completed. The keyboard was connected to the control unit with the pi running plover stenography software and had it enabled. The result was meeting the expectations with controlling the device and being able to type sentences that users might want to say at fast speed, 150 words per minute to support real time communication.

3.3 Software

Two main requirements mentioned for the software parts are that all modules should be able to work without the internet, and modules should not influence others performances. For example, TTS should be turned off when the user chooses to use STT, or the STT module may take the output speech from TTS as inputs. We have met both requirements for our project. To illustrate, all libraries used for this project are offline libraries. For switching modules, we bind F1 key to the switch terminal function which allows us to jump between two while loops in STT and TTS functions using a globaling variable. Binding keys code in Appendix 3 shows all keys we used to achieve various functionalities.

3.4 Audio Devices

Our requirements for the speaker were the correct function of the speaker for 30 min duration, and range of 6ft. The first requirement was tested by constantly outputting MP3 files for 30 min using the TTS software. During the 30 min duration, the speaker consistently outputted the correct words and sentences. Since we changed the location of the speaker, our requirement for range was changed from 6 ft hearing range to about 2 ft range. Upon testing with our amplifier unit, it was found that the volume is slightly smaller than it needed to be, and we could hear the speaker from around 1.5 ft away.

The microphone was required to have a range of 3 ft, and generate a clean voice input for STT use. We first checked if the microphone is generating a clean signal for the STT, and the accuracy of STT was around 80% excluding some words that do not exist in our STT software dictionary. Also, as we moved the interactor farther away from the microphone, the microphone picked up correctly around 70% of the time when moved 3 ft away.

4. Costs & Schedule

4.1 Parts

Table 4.1 Parts Costs

Quantity	Part	Model Number	Total Cost ($)
1	Keyboard microcontroller	ATMEGA32U4	5.16
50	Keyboard Diodes	1N4148W-TP	0.68
3	FUSE	nSMD025-24V	0.15
1	Keyboard Crystal	LFXTAL082071	0.74
2	22 uF	CL21A226MQQNNNE	0.20
3	10 uF	CL21A106KOQNNNG	0.20
3	1uF	LMK105C6105MV-F	0.40
3	.1uF	0402YC104KAT2A	0.29
2	220pF	CL05C221JB5NNNC	0.20
1	330uF Polarized Capacitor	865080145010	0.20
1	68uF Polarized Capacitor	865080142006	0.34
1	Voltage Booster IC	LTC1700EMS#PBF	7.16
1	Power Supply PMOS	IRFTS9342TRPBF	0.58
1	Power Supply NMOS	SQJ416EP-T1_GE3	0.46
2	PCBs	PCBWAY	10
2	316k Ohm	CRCW0603316KFKEB	0.20
2	100k Ohm	CRCW0402100KJNEDC	0.20
2	22k Ohm	CRCW060322K0JNEAC	0.20
3	10k Ohm	RMCF0201FT10K0	0.30
1	1k Ohm	RE0603FRE071KL	0.10
1	330 Ohm	AC1206FR-07330RL	0.11
1	1.8uH Inductor	AIML-0603-1R8K-T	0.20
1	4-Cell Battery Holder	BK-18650-PC8	7.85
4	3.7V 2.6 Ah Li-Ion	UL1865-26-1P	19.96
1	MicroSD Card	Silicon Power 32GB 3D NAND	7.99
1	USB microphone	CGS-M1	13.99
1	5" LCD Display	MDT0500D2SH-HDMI	50
1	8 Ohm Speaker	ECE supply center	1.8
1	Raspberry Pi 4B	RaspberryPi	55
1	Keyboard Switches	Akko CS Switches	13.99
1	TVS Diodes 5.5V	PRTR5V0U2X	0.26
	Total		**198.91**

4.2 Labor

Table 4.2 Labor Costs

Name	Hourly Rate	Hour	Total/Person	Actual Cost ($) Total x 2.50
Andrew Ko	$30	160	$4800	$12000
Minho Lee	$30	160	$4800	$12000
Yihan Ruan	$30	160	$4800	$12000
Total Cost for the Team				**$36000**

4.3 Schedule

Table 4.3 schedule

Week	Minho	Andrew	Yihan
9/20/21	Schematic for power bank	Finish keyboard pcb design	Searching for STT and TTS modules
9/27/21	Finished first PCB and ordered parts	Get it reviewed and make changes accordingly	Searching for STT and TTS modules
10/4/21	Talked to machine shop about enclosure	Design enclosure for the display module. Consult on version 2 PCB	test different modules in my laptop
10/11/21	Soldered PCB and testing	Test and verify the PCB	test different modules in my laptop
10/18/21	Further testing and revision of PCB	Version 2 PCB design	testing modules on raspberry pi
10/25/21	Ordered second PCB and new parts	Finish and Order version 2 of PCBs	integrating STT and TTS modules
11/1/21	Testing audio devices	Work on integrating the keyboard and the Pi	integrating STT and TTS modules
11/8/21	Second PCB assembly and testing	Scripts and code to map the keys to contain stenographic layout and	searching for user interface library
11/15/21	Testing the PCB and audio devices. Solved some of the problems with the design.	Testing to ensure the functionality of keyboard to play words as mp3 and mic to to display words on screen	creating an user interface switching between modules
11/22/21	New part for PCB attached and tested and completed	Work on the integration and have the entire system running to check for areas of bugs	creating an user interface switching between modules
11/29/21	Speaker and Mic were tested and completed	debugging keyboard with stability issues and practice stenography	update user interface

5. Conclusion

5.1 Accomplishments

Overall, our project has met the majority of requirements we have stated in the design document. After booting the user interface, a brief description for the device will be played by the speaker when the user presses F3 key. With pressing F2 key, speech recorded by microphone will be transferred to blue text shown on the display. By pressing the F1 key, the user is able to switch to the TTS module and what the user typed is displayed as black text on the display. While the conversation is ongoing, the display will always display the latest four sentences. When the conversation is ended, the user can stop the program by pressing the escape key.

5.2 Uncertainties

There were some parts of the device that did not meet our expectations. We did an extensive amount of testing on the charging system of the power bank, but could not figure out the problem with the system. The poor first design of PCB for testing also made the process more difficult.

Another is the keyboard layout. We wanted a slightly diagonal design of the keyboard, which was supposedly more ergonomic. However, it was rather complicated to design the keyboard in the design we planned when using the row-col system which was required for our QMK firmware.

5.3 Ethical considerations

There can potentially be some safety hazards with our project as it contains batteries. The batteries should not be overcharged and handled with care as they might explode with rough use [2]. The battery may get damaged if the temperature is too high or too low. Therefore, we will add a thermistor and a protection IC to the charging loop to disconnect the battery from the charger when abnormality in temperature is detected. Also, the charger for the lithium battery will include an IC as suggested in ECE445 Battery Safety [3]. Physical abuse of the battery may also damage the battery, so the belt device might have its limits when the user plays sports or participates in vigorous physical activities [4].

Designed as an outdoor use device, the belt might be prone to some tough weather conditions, if there is a heavy pour of rain, for example. Since we are building the display module as an extendable unit, we should be careful in enclosures and circuit designs such that the users will not get electrocuted or get physically wounded by any of the device components.

Since we will be using open library sources for text to speech and speech to text in the translation process, we should make sure to cite the sources as to not infringe on one of IEEE code of Ethics - to seek, accept, and offer honest criticism of technical work, to

acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others.

5.4 Future work

One improvement we can make is to make the program easier to reboot. For the current model, we need an additional connection to the mouse to start the user interface and plover since we need to run the python file for user interface and plover on two terminals. What we can do for future work is to combine those two parts to an exe file and create a keyboard shortcut to access the program, which will make rebooting the program much easier.

Also, we can improve much on the ergonomics of our product. The casings can be made much lighter with lighter materials like plastic. The cables can be made more organizable. The belt can be made more comfortable for the user.

It would be convenient for the user to have the ability to charge the power bank without pulling out of the box, so we can complete the charging part of our power bank as well.

References

[1] J. Meyer, L. Dentel and F. Mecunier, "Speech Recognition in Natural Background Noise," *PLOS ONE*, November 19, 2013. [online], Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0079279 [Accessed Sep. 28, 2021]

[2] *Preventing fire and/or explosion injury from small and ...* (n.d.). Retrieved September 29, 2021, from https://www.osha.gov/sites/default/files/publications/shib011819.pdf

[3] *General Battery Safety*. Project :: ECE 445 - Senior Design Laboratory. Retrieved September 29, 2021, from https://courses.physics.illinois.edu/ece445/project.asp?id=6058

[4] *Battery regulation*. U.S. Consumer Product Safety Commission. (n.d.). Retrieved September 29, 2021, from https://www.cpsc.gov/Regulations-Laws--Standards/Voluntary-Standards/Topics/Batteries.

[5] *Raspberry pi typical power requirements*. Raspberry Pi Hardware. (n.d.). Retrieved September 30, 2021, from https://www.raspberrypi.org/documentation/computers/raspberry-pi.html#typical-power-requirements

[6] *Atmel ATmega16U4, atmega32u4 datasheet summary*. (n.d.). Retrieved December 8, 2021, from https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Summary.pdf.

[7]Project, O. S. (n.d.). Plover. Retrieved December 8, 2021, from https://www.openstenoproject.org/plover/.

[8]Qmk. (n.d.). *Releases · qmk/qmk_toolbox*. GitHub. Retrieved December 8, 2021, from https://github.com/qmk/qmk_toolbox/releases.

[9]*PCB designer guide*. PCB Designer Guide | Keyboard Designer Wiki @ ai03.me. (n.d.). Retrieved December 8, 2021, from https://wiki.ai03.com/books/pcb-design/chapter/pcb-designer-guide.

[10] *Tkinter*. TkInter - Python Wiki. (n.d.). Retrieved December 8, 2021, from https://wiki.python.org/moin/TkInter.

Appendix A Requirement and Verification Table

Table 1 Power Supply

Requirement	Verification	Verification status (Y or N)
1. The total capacity of connected battery cells store 2.6 A * # of battery cells (4). 2. Provided output voltage is 5V +/- 10%. 3. Output current output ranges from 0 to 2A.	1. Fully charge the battery and discharge at 2000mA for 6 hours. B. Measure voltage of the battery to check it remains above 3.0 V 2, 3. A. Connect output to the load resistance that will draw 2000 mA B. Measure using a voltmeter, check the output voltage is ~5V and current is 2A.	1. Y 2. Y 3. Y

Table 2 Keyboard

Requirements	Verifications	Verification status (Y or N)
1. This keyboard is required to be able to type any word based on the steno alphabet 2. The stenographic keyboard needs to be able to display words with stenography to support real time communication at 150 words per minute. 3. The keyboard should have function keys and can be used to boot the user interface when the device	1. A. The display can show every word user types within 2 seconds. B. When users press more than 1 key at the same time, the keyboard should be able display all of them before we integrate the keyboard with the steno api. According to the steno alphabet, this function should work up to 7 keys. 2. Users are able to type more than 30	1. Y 2. Y 3. Y

boots up initially.	words per minute when they are familiar with how to use a stenographic machine.	

Table 3 Software

Requirements	Verifications	Verification Status (Y or N)
1. Both modules are required to work without the internet, and they should be easily turned on and off. 2. Those two modules will not influence each other's performance.	1. A. Those two modules will start working after we press the power button with no more than 2 seconds delay. B. Both modules should be able to work continuously for more than one hour 2. One module is required to be turned off when the other module is currently used, and the TTS module should have a higher priority, because the TTS module displays the thoughts of our users.	1. Y 2. Y

Table 4 Control

Requirement	Verification	Verification Status (Y or N)
1. The Raspberry Pi is required to successfully connect and transfer signals between different units	1. A. The display unit can display what we have typed on the stenographic keyboard. B. The speaker can play MP3	1. Y 2. Y 3. Y

2. The Raspberry Pi can successfully perform speech-to-text module and text-to-speech module 3. The button is inputted correctly to the device without bouncing	files stored in Raspberry Pi within 2 seconds after we open the file. C. The Raspberry Pi can store sounds recorded by the microphone with no more than 2 seconds delay. 2. A. When we run the code stored in Raspberry Pi on a computer, texts will be played with no more than 2 seconds delay. B. By connecting Raspberry Pi and a computer, the computer can output sounds after we type a sentence in the terminal with no more than 2 seconds delay. 3. The device can successfully turn on or off with one press of the button	

Table 5 Display

Requirements	Verifications	Verification Status (Y or N)
1. The LCD Panel is required to clearly show texts typed by the keyboard 2. The LCD Panel is also required to clearly show texts transferred from speech. 3. The display can successfully be charged by the battery through the Raspberry Pi	1. Each word will be displayed on the display unit within 2 seconds after you type the word using the stenographic keyboard. 2. Sentences spoken by an user will be displayed word by word on the display unit within 2 seconds. 3. After the battery runs out, the display unit will reboot within 5 minutes after we recharge the battery.	1. Y 2. Y 3. Y

Table 6 Speaker

Requirements	Verifications	Verification Status (Y or N)
1. The speaker unit can play an MP3 file created and saved in the SD card 2. The physical dimension of the speaker should not be too large to fit on the belt but it should be able to produce audio loud and clear enough to be heard at talking distances.	1.For both STT and TTS modules, we don't need to store all the speech. We just need to convert the current sentence of speech or text, which should be no more than 30 seconds. A minute of the MP3 file populates the storage at about 1.5MB. Thus, the SD card should have enough storage. We will verify this by constantly typing and creating MP3 files for around 30 mins. 2. The SNR for this speaker is 60db, which is a normal voice level at distances ranging 1 to 4 meters. We will verify that the speaker is audible by having the interactor talk and listen at 6ft away from the user wearing the belt.	1. Y 2. N

Table 7 Microphone

Requirements	Verifications	Verification Status (Y or N)
1. The range of the microphone is long enough to pick up the interactor's sound. 2. The sound signal quality will be good enough to be used in the STT module.	1. A common distance during conversation is around 1 to 3 meters. Therefore, our microphone is required to record sounds within 3 meters. 2. After integrating the microphone with the STT module, there should be no output while nobody is talking.	1. Y 2. Y

Appendix B Block diagram & Schematics

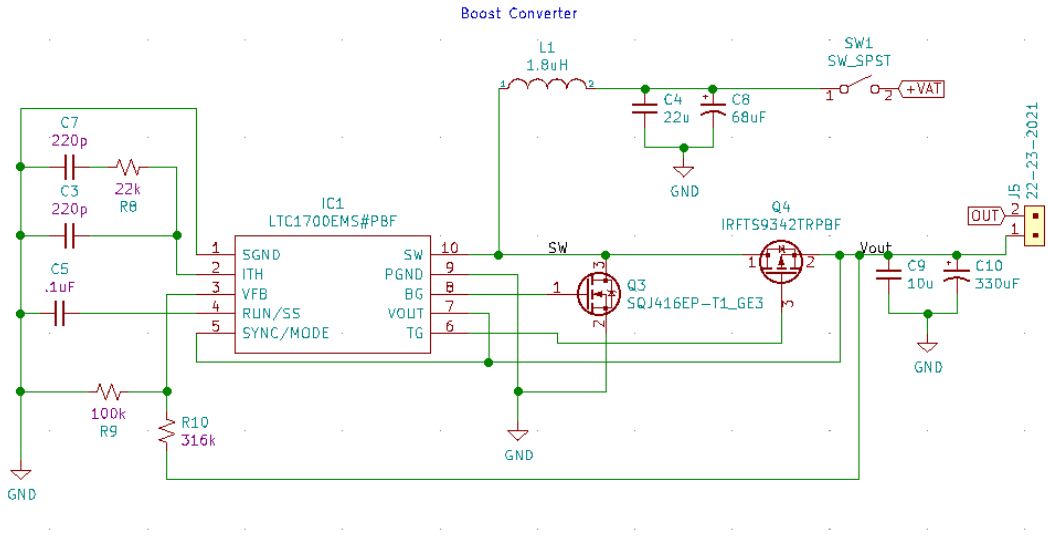

Fig. 1 Voltage Regulator Schematic

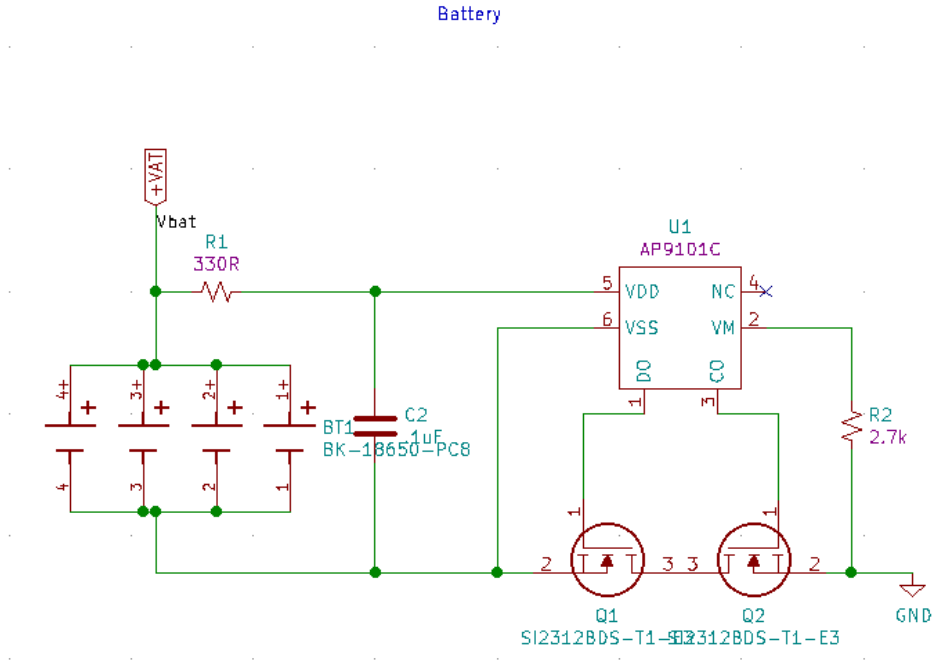

Fig. 2 Battery Protection Schematic

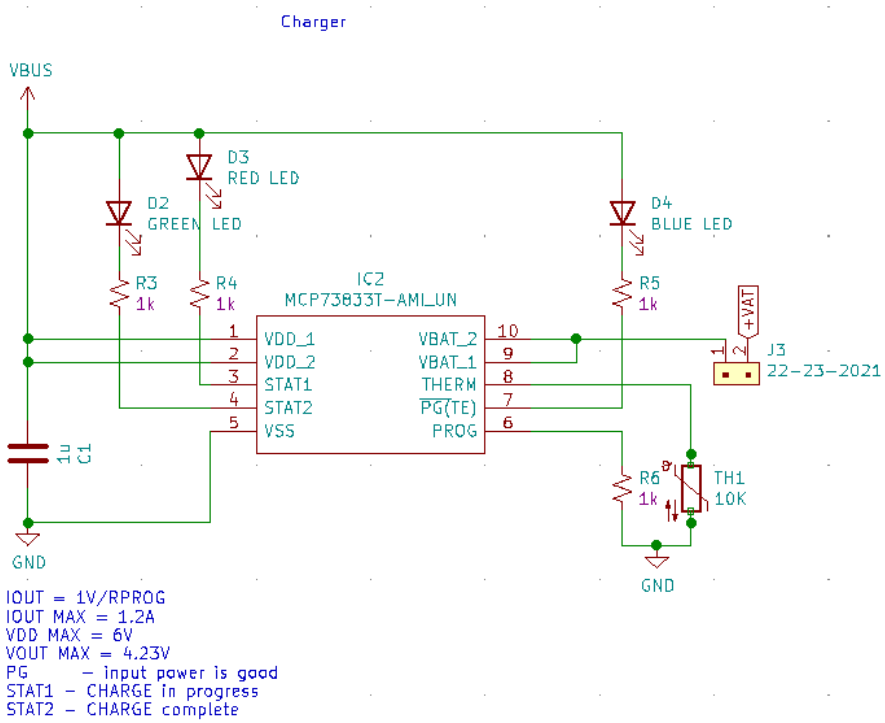

Fig. 3 Battery Charger Schematic

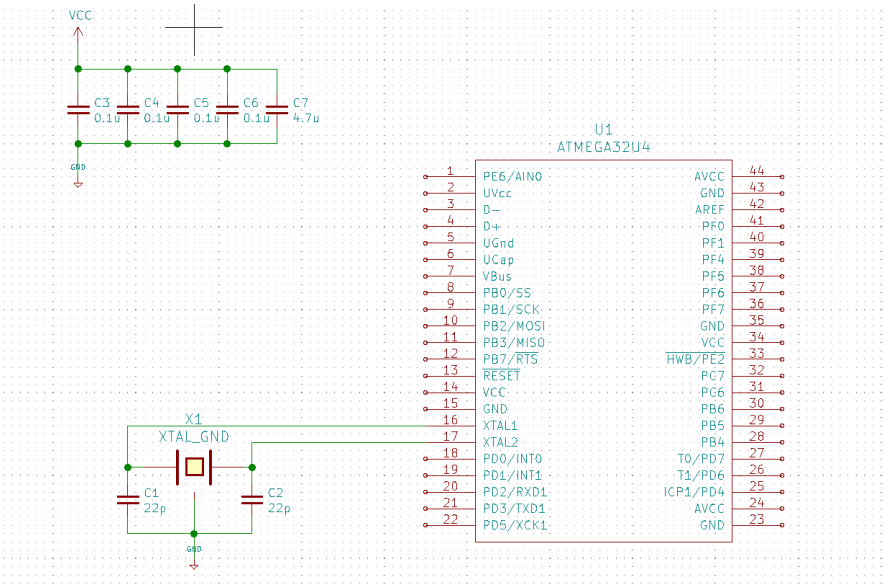

Fig. 4 ATMega32u4 and decoupling capacitors

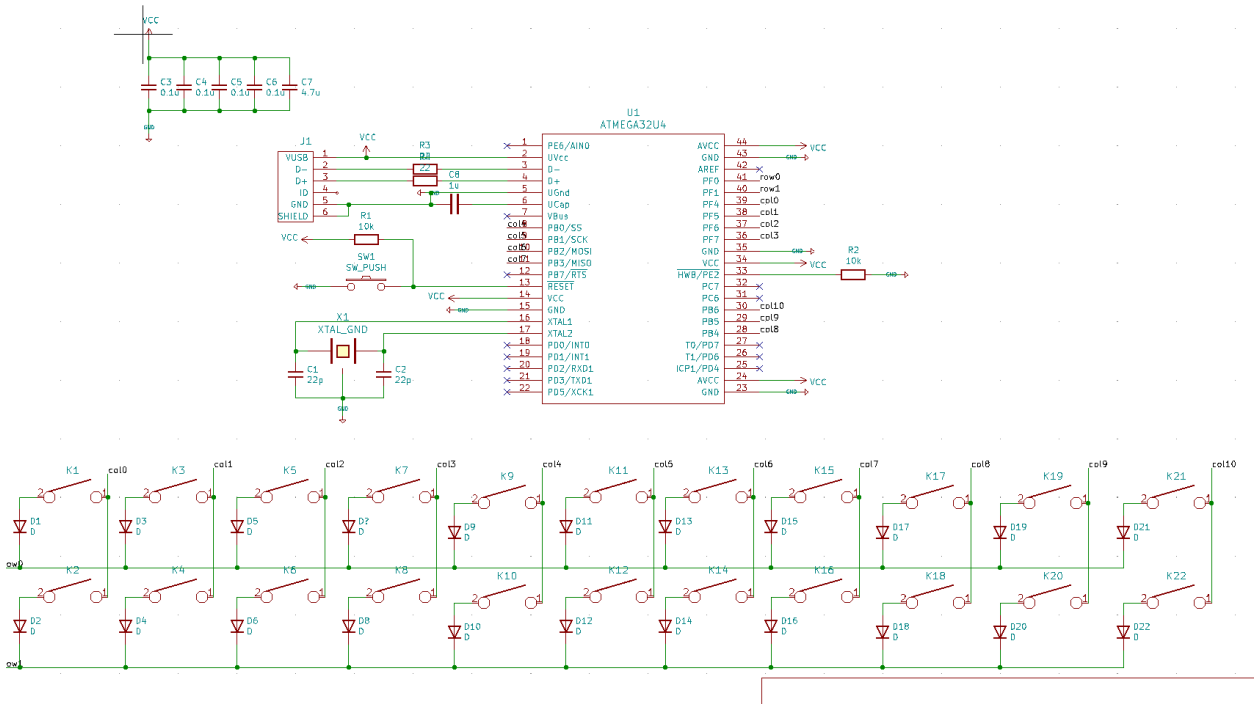

Fig. 5 complete schematic of keyboard

Appendix C Software Source Code

```
root.bind("<F3>", lambda event:intro_tts())

root.bind("<F2>", lambda event:startConv())

root.bind("<F1>", lambda event:swtf())

root.bind("<Escape>", lambda event:terminate())

root.bind("<Return>", lambda event:TTS_input())
```

Fig. 1 Binding Keys for the Program