USB Dictation Device

Design Document

Team: 66 TA: Chi Zhang

Qingyu Li, Wennan Zhai, Shengyu Ge

Original project: Prosthetic Hand for Typing, Team #1 Spring2020

Contents

1. Introduction

1.1 Objective

Many people with disabilities have trouble typing on their own. Although many mobile phones support dictation, it is complicated to setup on a PC or other platforms. The prothesis which utilize bionic arm technology is expensive. People with disabilities might not be able to afford it and moreover it is hard for them to get used to using prothesis. Our solution to this problem is a portable USB dictation device supporting common platforms. From the PC's perspective, the device functions as a common keyboard. To the user, instead of typing, it recognizes sounds of English words (or sounds of characters in a different working mode), split them into characters and sends the corresponding scan codes to the connected host device.

1.2 Background

Countless people are suffering from limb loss or hands disabilities. As the National Center for Health Statistics indicated, there are 50,000 new amputations every year in USA [1]. Those people really need something that can help them use computers without the keyboards.

Similar dictation devices can be found on the market but many of them require additional driver installation, take disk space and ram resources to run additional programs or even require some hardware acceleration devices. Another important problem with the existing products is the complexity. These devices often come with lots of functions unnecessary for our use case and create a steep learning curve for the users with disability. One example is the "PHILIPS SPEECHMIKE PREMIUM PUSH BUTTON LFH3500" which can be easily found on google.[2]

Figure 1.2.1 Philips product

The product in the picture above comes with many appealing functions apart from speech recognition such as customizable keys, track ball navigation systems and barcode scanner. However, these functions will not always be friendly to people with disabilities and will

not be the focus for our product. This product also has hardware requirements and takes up nearly 1 Gigabyte disk space for program and driver installation.
Our product differs from the existing products on the market in a way that we aim to solve the problem in a simpler way especially for individuals with disabilities. Our device has simple user interface, does not require hardware or driver installation and is hot plug enabled and ready to use in an instant. When connected to a PC host, the device appears as a general keyboard and all the speech recognition is running on the device itself. Another important point is the speech recognition can go wrong. A lot of Companies and researchers are doing analysis and improving the software of the speech recognition by making it be more accurate. However, when voice recognition does have a difficult time recognizing certain words, it is possible to offer a backup way for inputting the key code to the PC since our firmware is based on a real keyboard firmware. We want to utilize this technology to help those people with disabilities who really need this kind of device to help them type something on their computers.

1.3 High-level Requirements List

- Be able to collect the user's voice through microphone, recognize the characters/words spoken by the user and convert them to the correct keycodes approximately above 95% (+/-5%) accuracy.
- OS can successfully detect the device, install generic drivers, and receive keycodes sent from the device.
- Be able to switch between different states which perform different functionalities such as pausing, recognizing or different recognition modes with a minimum of 5V(+/-5%) 0.84A(+/-5%) (4.2W) and a maximum of 5V(+/-5%) 1.5A(+/-5%) (7.5W) power from a single USB 3.0 connection.

2. Design

2.1 Block Diagram

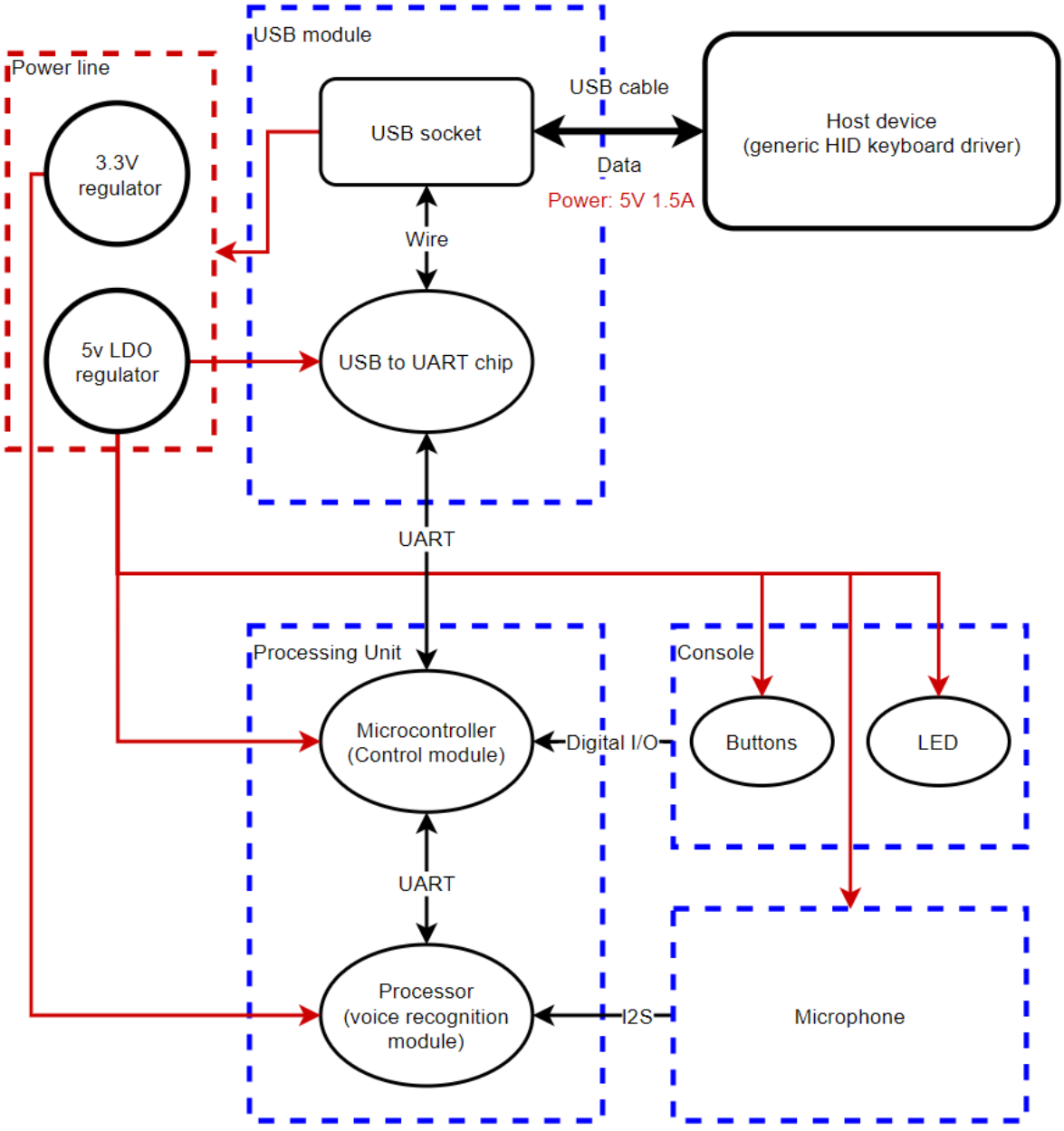

Figure 2.1.1 Block Diagram

2.2 Physical Design

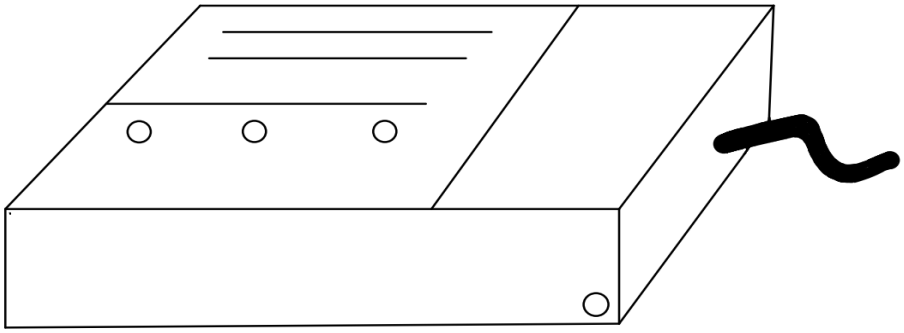

Figure 2.2.1 Product Overview

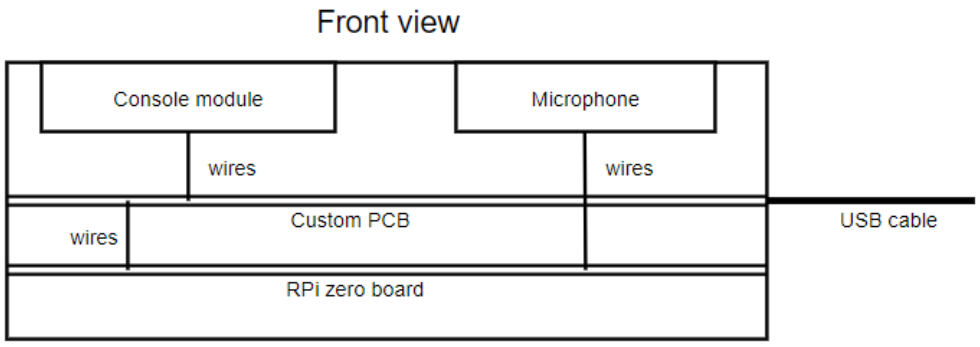

Figure 2.2.2 Product Front View

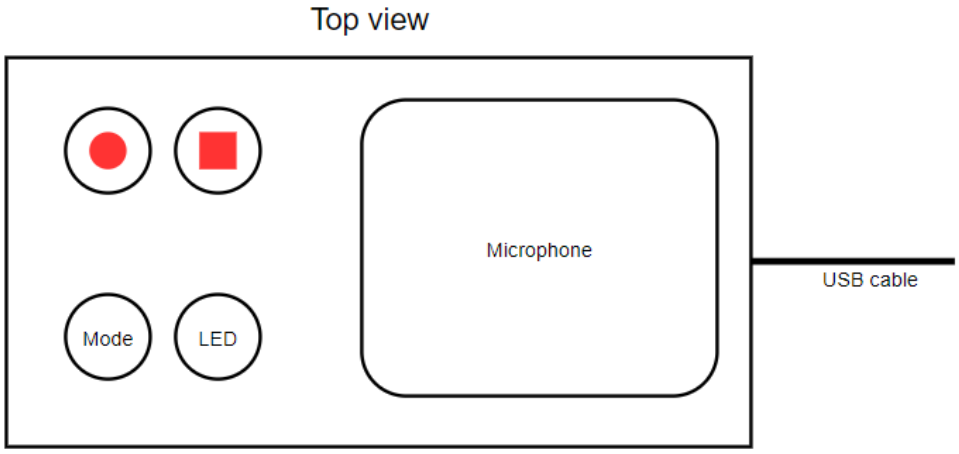

Figure 2.2.3 Product Top View

2.3 Subsystem Descriptions and RV table

2.3.1 Processing Unit

The processing unit is the core of this device. It consists of two components, a microcontroller for the control program (or firmware) and a high-speed processor for the voice recognition module.

The microcontroller runs the control program for the whole device (detailed description in 2.3.2 Control module). The processor runs a voice recognition program (detailed description in 2.3.3 Voice recognition module) for processing the microphone input and sending results to the microcontroller. The processor is on a Raspberry Pi zero board(RPi board) for easier interfacing and the RRi board is used exclusively for this computing purpose.

This subsystem is separated from the two software modules and is responsible for the hardware communication and input/output connections.

Requirements	Verifications
1. The processing unit should be able to receive digital signals from the console module. 2. The processing unit should be able to receive digital signals from the microphone. 3. The processor and the microcontroller should be able to communicate though UART protocol.	1. Verified in 2.3.2 Control module. 2. Verified in 2.3.4 Microphone. 3. a. Connect TX/RX pins on the microcontroller to RX/TX GPIO pins on the RPi and make sure both device is properly powered. 3.b. After the python libraries (e.g. WiringPi) are installed on Pi, open Pi's UART at 9600 baud rate by writing "wiringpi.serialOpen('/dev/ttyAMA0', 9600)". 3.c. Connect an LED to the microcontroller and upload a program to the microcontroller which lights up the LED by setting high to the corresponding digital pin when digitalRead(RX) is high. 3.c. Connect the microcontroller to Send message to the microcontroller by "wiringpi.serialPuts(serial, 'hello')." and see if the LED lights up.

2.3.2 Control module (software)

The control module (or firmware) is a software program running on the microcontroller in the processing unit. The functionality of this program includes keeping/transitioning system states, interacting with the console module (button pressing, led lighting etc.), receiving the processed speech recognition results from the voice recognition module and sending out scan codes to the host device through USB.
When communicating with the host device, this device appears as a generic HID keyboard device and can use the built-in HID drivers for the OS without installing a specific driver. To achieve this goal, the implementation of the control program (the part for mimicking a general keyboard) will be based on an open source keyboard firmware development project, QMK Firmware.[3]

Requirements	Verifications
1. The control module should be able to send data to the host computer. This requirement is also part of the high-level requirement for the whole device.	1.a. Modify the finished control program to keep sending the scan code for letter A through USB. 1.b. Connect the device to a host computer through the USB socket with male-to-male USB 3.1 cable. 1.c. Open a text editor and see if a string of “A” is typed.
2. The control module should be able to transition into states specified by the console’s input.	2.a. Upload the program with debug information to the microcontroller with Arduino IDE app. 2.b. Properly connect the console parts to the microcontroller. 2.c. Press the start button on the console to see if the LED which indicates the system is recording and conducting voice recognition is lit up. 2.d. Press the “Mode” button to see if the LED for each function mode is lit up. 2.e. Press the stop button to see if the LED indicating idle status is lit up.

2.3.3 Voice Recognition Module (software)

The voice recognition module is a software program running on the RPi processor. Upon working state, the voice recognition module will keep analyzing the input from the microphone. Once it recognizes a word or a character, it splits the words into characters and then transmits the corresponding scan codes to the control module and the control module will send them to the host computer through USB protocol.
The voice recognition module should be able to output corresponding characters or words given the bitstream of bit data.

Requirements	Verifications
1. The voice recognition module should be able to receive data from the microphone.	1. After the I2S microphone is verified, this requirement is already satisfied.
2. The voice recognition module should be able to generate correct outputs when fed with random inputs.	2.a. Install and set up the open source library on our own computers. 2.b. Connect a microphone device and start the program and read our test inputs loud to see if the printed outputs match our test inputs.
3. The voice recognition module should be able to receive signals from the microcontroller to start or stop the program.	3.a. Connect the digital pins of the microcontroller with those of the microprocessor according to UART protocol. 3.b. Use digitalWrite to send signal and digitalRead on the Pi to check the result.

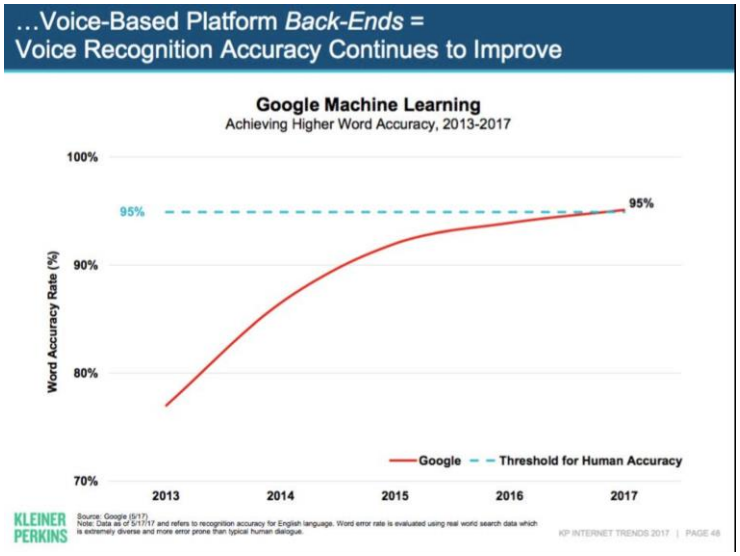

Figure 2.3.3 Recognition Accuracy

2.3.4 Microphone

The microphone is connected directly to the processing unit and is used to pick up the user's voice.
It should be able to convert the sound to the digital signal and send it to the processing unit or directly to the voice recognition module.

Requirements	Verifications
1. The microphone should be able to transmit the recorded data to the microprocessor via I2S protocol.	1.a. Connect the I2S Microphone's five ports to the corresponding ports on the Raspberry Pi Zero with the mic port facing up. 1.b. Setting up the Pi's configuration files to enable the I2S support according to the online guide [4]. 1.c. If all is working correctly, we should see the VU meter at the bottom of the terminal with SSH into the Pi.

- Mic 3V - Pi 3.3v
- Mic Gnd - Pi Gnd
- Mic SEL - Pi Gnd (this is used for channel selection. Connect to 3.3 or GND)
- Mic BCLK - BCM 18 (pin 12)
- Mic LRCL - BCM 19 (pin 35)
- Mic DOUT - BCM 20 (pin 38)

Figure 2.3.4-a Port Configuration

2.3.5 Console

The console is used to control the system. It has some buttons on it to control the different states of the system like start, pause or stop. There is a buzzer to indicate when the diction starts or stops. It has some basic function keys such as “delete”, “space” to better control the typing. It may have a full keyboard layout to correct any char that couldn’t be recognized through voice for several times.
It should be able to generate digital or analog signal in correspondence to the buttons or keys pressed. It should also be able to notify the user about the state of the system.

Requirements	Verifications
1. The buttons on the console should work accordingly.	1. Connect the button to the digital pin of the microcontroller and press the button, see if the digitalRead function returns a HIGH.
2. The LED on the console should light up normally.	2. Connect the LED to the digital pin of the microcontroller, the ground and a resistor, then use digitalWrite function to output a HIGH signal and observe whether or not it lights up.

2.3.6 USB Module (Power and Communication)

The whole system is powered by USB connection to the PC, and the USB protocol is also used to transmit the key code to the PC.
It should be able to draw enough current with 5V from the computer and transfer data between the system and the PC.

Requirements	Verifications
1. The module should be able to connect to the PC and be recognized as a general-purpose device.	1. Use male-to-male USB 3.0 compatible cable to connect the device with the PC and check whether there is a pop-up window on the bottom right corner.
2. The module should be able to draw enough current from the PC, which sums up to a maximum of 7.5W with a 5V voltage.	2.a. Leave the system connected to the PC and put the system into test mode (the power pin from the USB 3.0 port is disconnected). 2.b. Use the multimeter to probe the power pin of the USB 3.0 port on our system and the test pin (the current now flows through the multimeter). 2.c. Let the system runs for a while and check whether the current measured exceed 1.5A limit.

2.4 Data Flow

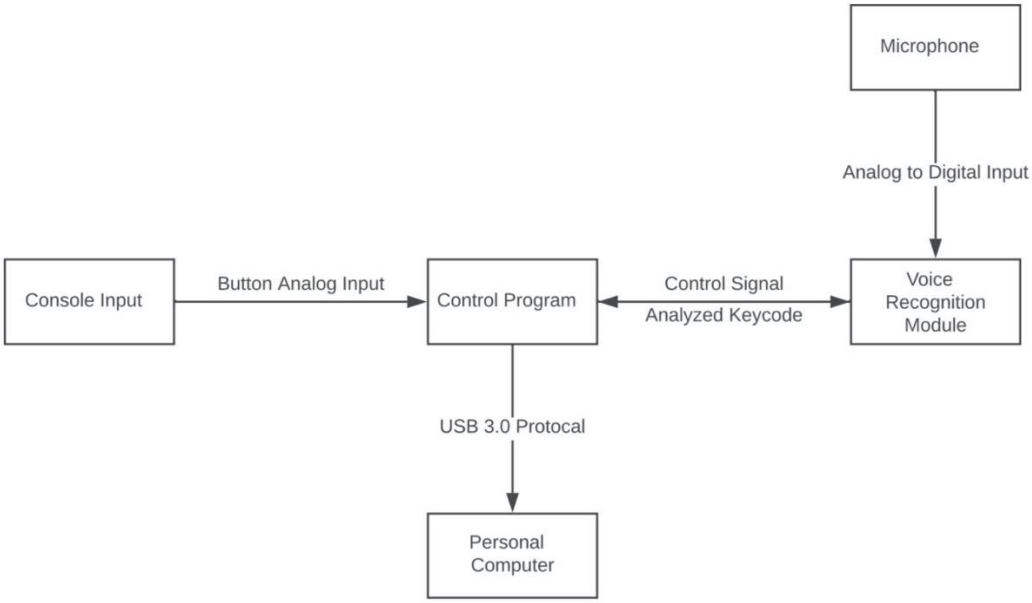

Figure 2.4.1 Data Communication

2.5 Tolerance Analysis

1. The processing unit serves as a power line to the whole system, so being able to draw enough current and deliver them to the other parts of the system is crucial:
Pi Zero requires a minimum of 120mA (0.7W) with only USB and WIFI turned on;
Console requires approximately a maximum of 100mA (0.5W);
Microphone requires a maximum of 600mA (3W).
In total:

$$0.7 + 0.5 + 3 = 4.2W \quad \text{(Eq 2.5.1)}$$

$$7.5 - 4.2 = 3.3W \quad \text{(Eq 2.5.2)}$$

So, we have pretty good extension capability in terms of power consumption.

2. Another part that is of high risk is the voice recognition module. For now, our choice is the Raspberry Pi Zero. It will have to run some open source code to output a word or a character given a duration of voice input:

The data rate of the microphone has a maximum of 64kHz with 24 bits:

$$64k/s * 3B * 100 = 19.2MB/s \quad \text{(Eq 2.5.3)}$$

So, the module must at least be capable of processing 19.2MB data per second given the approximation that the intermediate data is 100 times larger than the input data.

3. Project Differences

3.1 Overview

The original solution tends to create a prosthetic device that can be used to type with the user's feet. It makes advances in bionic arm technology, which is very expensive to implement, and it is relatively hard for people with certain disabilities to get used to. Additionally, there are a lot of people having disabilities on both of their hands and feet, so they are still not able to use the device proposed by the original solution. Our solution makes an advance in the speech recognition technology and the functionalities will be integrated into a portable USB device that is connected to the computer. It can recognize the sounds of English characters and sends the corresponding characters' keycodes to the connected host device. One tradeoff is that the output keycodes might not be 100% accurate.

3.2 Analysis

Our solution solves the original problem in a fundamentally different way from the original solution in many aspects. First, it serves different customers. The original prosthetic hand mostly helps the people without one or both hands while our solution is beneficial to many other customers such as blind people. Second, the physical appearance and working process are vastly different from the original. Our solution is a simple plug-in USB device that powered by the USB ports. The users only need to plug the device into their PC, and it will be recognized as a regular keyboard. Once the device is connected and the driver is installed, the user can start "typing" with ease. Third, the design is fundamentally different from the original. In the original solution, inputs are collected through the four buttons controlled by users' feet. The outputs are displayed by motors moving five fingers on the prosthetic hand. In our solution, inputs are collected via microphone and processed by the microprocessor, outputs are sent to other devices through USB ports.

Figure 3.2.1 Original Team's Physical Design

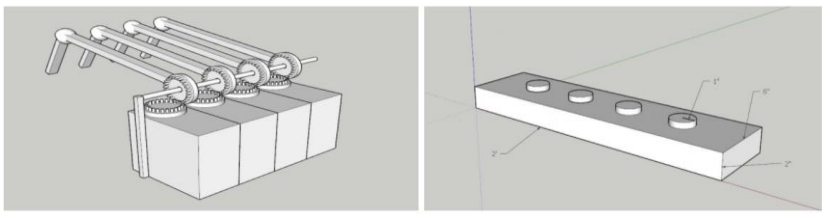

Figure 3.2.2 Original Team's 3D Model

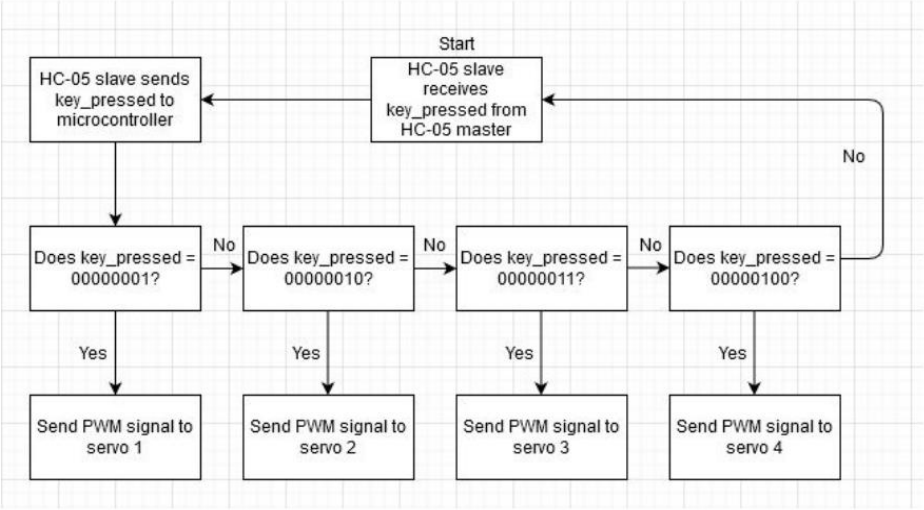

Figure 3.2.3 Original Team's Flowchart for Prosthetic Hand

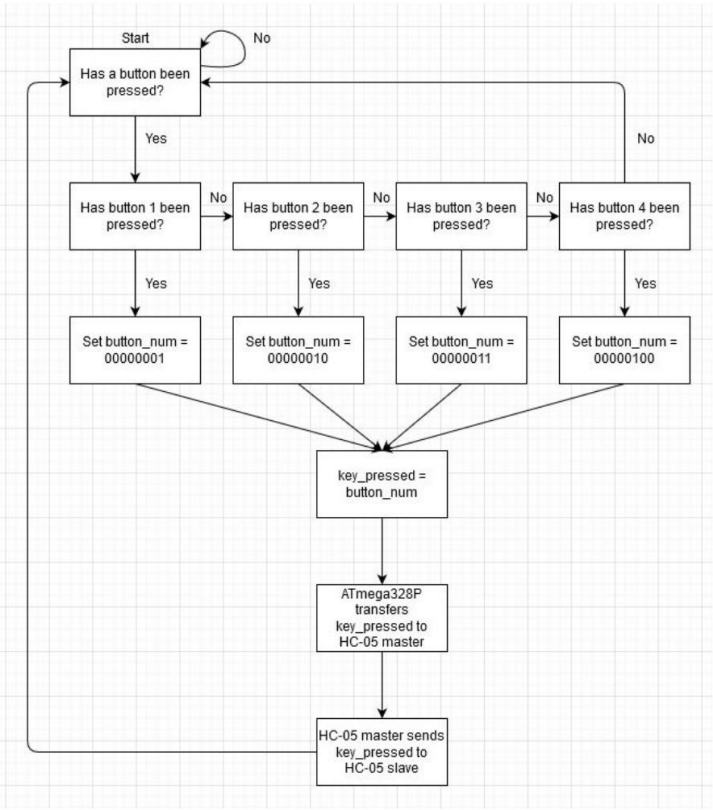

Figure 3.2.4 Original Team's Flowchart for Foot Button System

4. Cost and schedule

4.1 Cost

Labor:

Our development costs are roughly estimated to be $20 per hour, 10 hours of work per week for 3 people and 10 weeks in total that we can contribute to this project:

$$3 * 20\$/hr * 10hr/week * 10week * 2.5 = \$15{,}000 \quad \text{(Eq 4.1.1)}$$

Material:

Raspberry Pi Zero: $25.99
ATMega 328: $2.08
I2S MEMS Microphone: $6.95
PCB: ~$30
USB 3.0 Port: ~$4.99
Others: ~$10
Total: ~$80.01

4.2 Schedule

	Qingyu Li	Wennan Zhai	Shengyu Ge
Week 1	PCB design	PCB design	PCB design
Week 2	Coding main program and training model	Generating correct control signals to any specific pin with the breadboard	Coding main program and training model
Week 3	Unit test program output	Unit test circuit output	Unit test circuit output
Week 4	Unit test program output	Unit test circuit output	Unit test circuit output
Week 5	Create a driver to ensure the stable USB connection	Check the functionality of the console	Create a driver to ensure the stable USB connection

Week 6	Integrate the microphone to the whole system	Integrate the microphone to the whole system	Test the accuracy of the dataset with the trained model
Week 7	Any unfinished task from above and final integration	Any unfinished task from above and final integration	Any unfinished task from above and final integration
Week 8	Prepare mock-up (if needed) and prototype refining	Prepare mock-up (if needed) and prototype refining	Prepare mock-up (if needed) and prototype refining

5. Ethics and Safety

There might be some potential safety problems with our projects. The data of the input voice and the output characters might be hacked by people with some evil purposes. We do not currently have a perfect solution to this, so under most of the circumstances, users are not recommended to say their private information like password of the bank account, personal address, etc. to this device.

We thoroughly went over the 10 ethics mentioned on the IEEE Code of Ethics and we firmly believe that we will obey the rules of these ethics.

1. We hope that we can make people's life much easier and incorporate their lives with advanced technology, especially those people with disabilities who have trouble typing on their own. This fulfills the IEEE Code of Ethics, #5: "to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems" [5].
2. "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment;" [5]
    - Our project will not affect the safety of the public. It uses electricity as its main power supply, so it will not have a negative effect on the environment.
3. "to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;" [5]
    - Our project will not have conflict of interest and even if the conflict exists, we will inform the affected parties.

References

[1] Statistics on Hand and Arm Loss. [Online]. Available: http://www.aboutonehandtyping.com/statistics.html. [Accessed: 02-Apr-2020].

[2] Philips Speechmike Premium Push Button LFH3500. [Online]. https://www.transcriptiongear.com/philips-speechmike-premium-push-button-lfh3500.html?gclid=CjwKCAjwYT1BRAFEiwAd2WRtpBldfs78JOw7BVDzrPexnPlGlEG9G0qU9CEQXettN5zyMzEM6PTyBoCXekQAvD_BwE

[3] QMK Firmware. [Online]. https://qmk.fm/

[4] Raspberry Pi Wiring & Test, 2020. [Online]. Available: https://learn.adafruit.com/adafruit-i2s-mems-microphone-breakout/raspberry-pi-wiring-and-test#raspberry-pi-i2s-configuration

[5] Ieee.org, "IEEE IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 29- Feb- 2020].