USB Dictation Device

Design Document

Team: 66 TA: Chi Zhang

Qingyu Li, Wennan Zhai, Shengyu Ge

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 really 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 characters (or sounds of words in a different working mode) and sends the corresponding characters' keycodes 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 keywords.

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 96%(+/-1%) 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

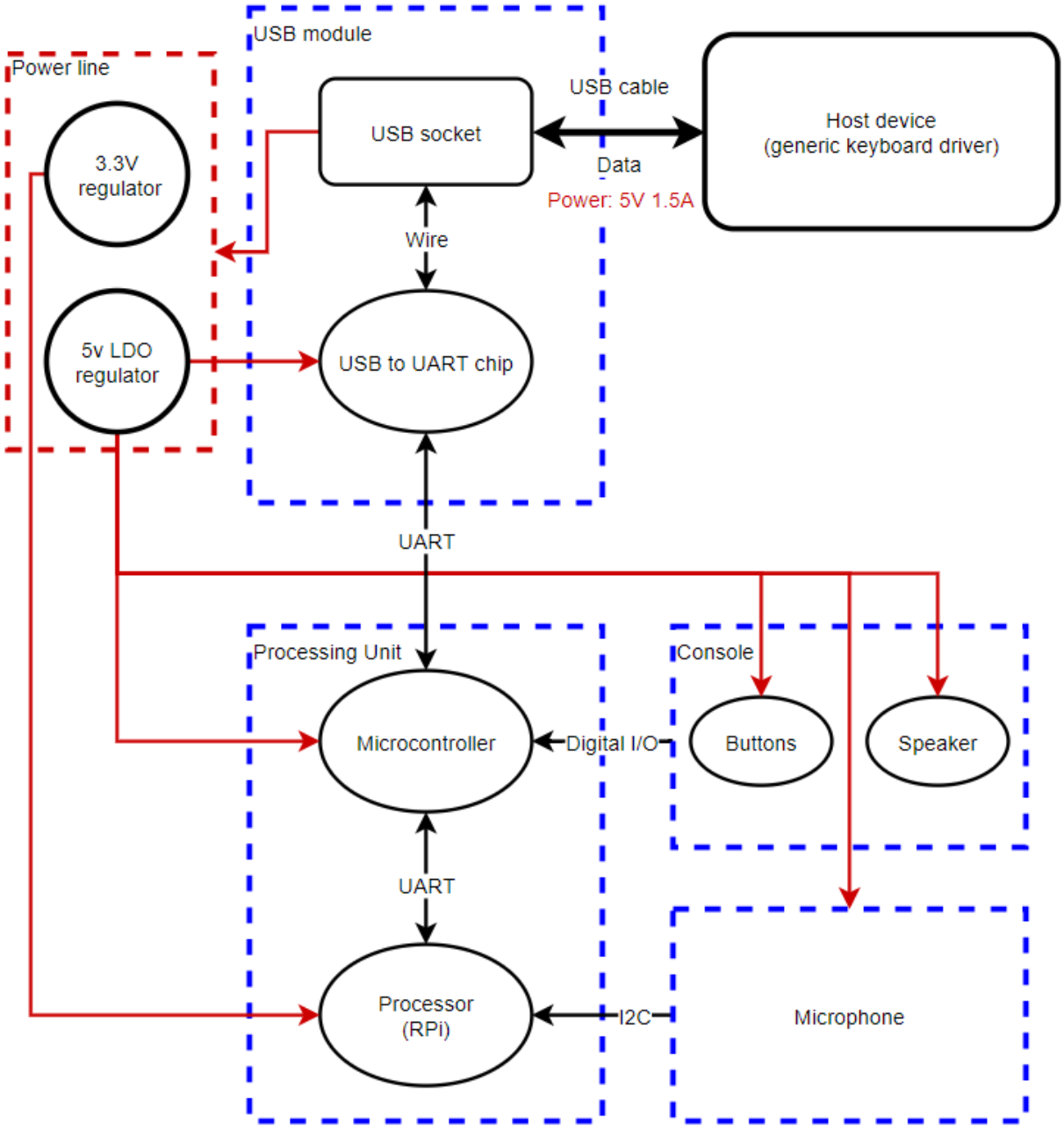

Figure 2.1.1 Block Diagram

2.2 Physical Design

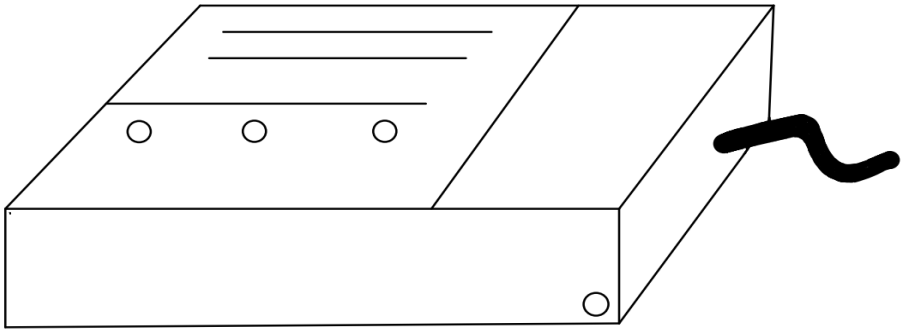

Figure 2.2.1 Product overview

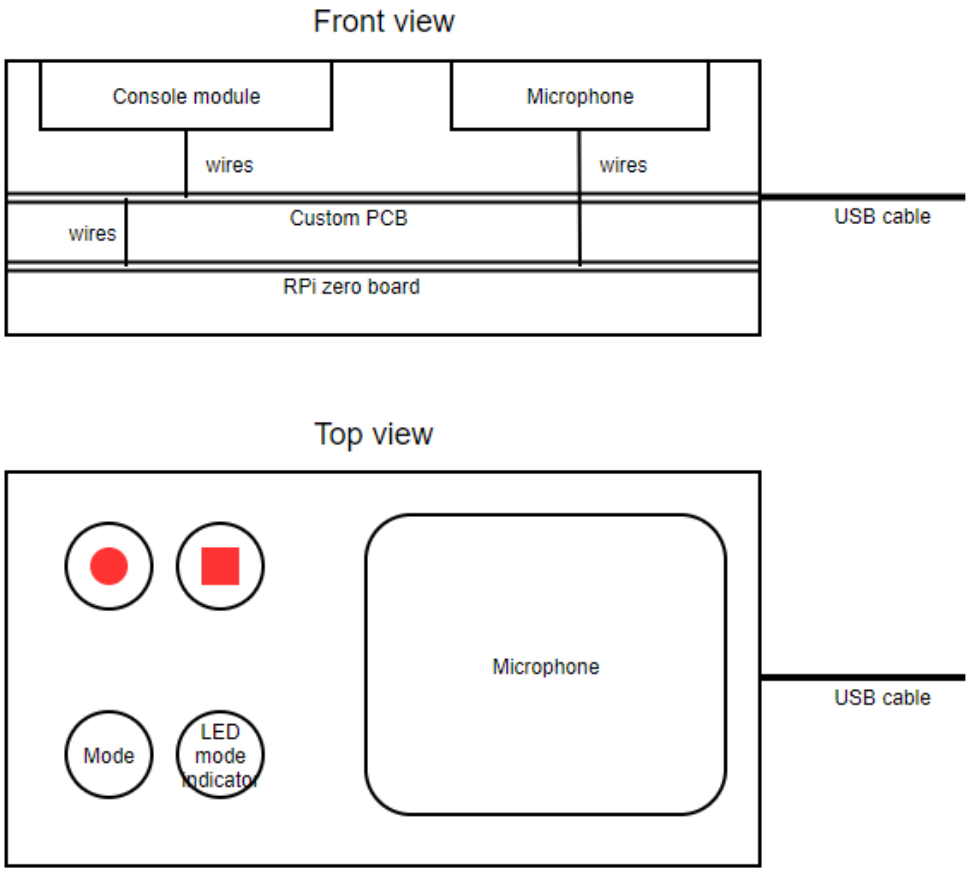

Figure 2.2.2 Product Front/Top view

2.3 Subsystem Descriptions

2.3.1 & 2.3.2 Processing Unit and Voice Recognition Module

The processing unit consists of two components, a microcontroller and a high-speed voice recognition module. The microcontroller responds to the console for user's interactions such as starting input and pausing input. Upon working state, the voice recognition module will keep analyzing the input from the microphone, and once it recognizes any char that corresponds to any key on a keyboard it then transmits the key code through USB protocol to the computer.

It should be able to receive the complete voice data, transmit it to the voice recognition module and get back the processed result from it as the core functionality. It should also be able to read input from the buttons for switching states and various interactions between the user and the system.

The voice recognition module should be able to output corresponding characters or words given the bitstream of bit data.

2.3.3 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.

2.3.4 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.

2.3.5 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.

2.4 Subsystem Requirements and Verifications

2.3.1 Processing Unit with Voice Recognition Module

Requirements	Verifications

1. The processing unit should be able to receive analog signal and digital signal from the other components. 2. The processing unit should be able to generate analog signal and digital signal to the other components.	1&2. (Connections to other components and functionalities are verified in other subsystems' verifications).
3. The processing unit should be able to transition into states specified by the console's input.	3.a. Upload the program with debug information to the microcontroller with Arduino IDE app. 3.b. Properly connect the console to the processing unit. 3.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. 3.d. Press the switch button to see if the LED for each function mode is lit up. 3.e. Press the stop button to see if the LED indicating idle status is lit up.

2.3.2 Voice Recognition Module

Requirements	Verifications

1. The voice recognition module should be able to receive data from the microphone. 2. The voice recognition module should be able to generate correct outputs when fed with random inputs. 3. The voice recognition module should be able to receive signals from the microcontroller to start or stop the program.	1. After the I2S microphone is verified, this requirement is already satisfied. 2.a. Install and set up the open source library on our own computers. 2.b. Start the program and read our test inputs loud to see if the printed outputs match our test inputs. 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.

2.3.3 Microphone

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 [3]. 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 2.3.3-a Port Configuration

2.3.4 Console

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.
3. The speaker on the console should sound accordingly.	3. Connect the speaker to the digital pin of the microcontroller, then use digitalWrite function to output a HIGH signal and check if the speaker is sounding.

2.3.5 USB Module (Power and Communication)

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.

2.5 Data Flow

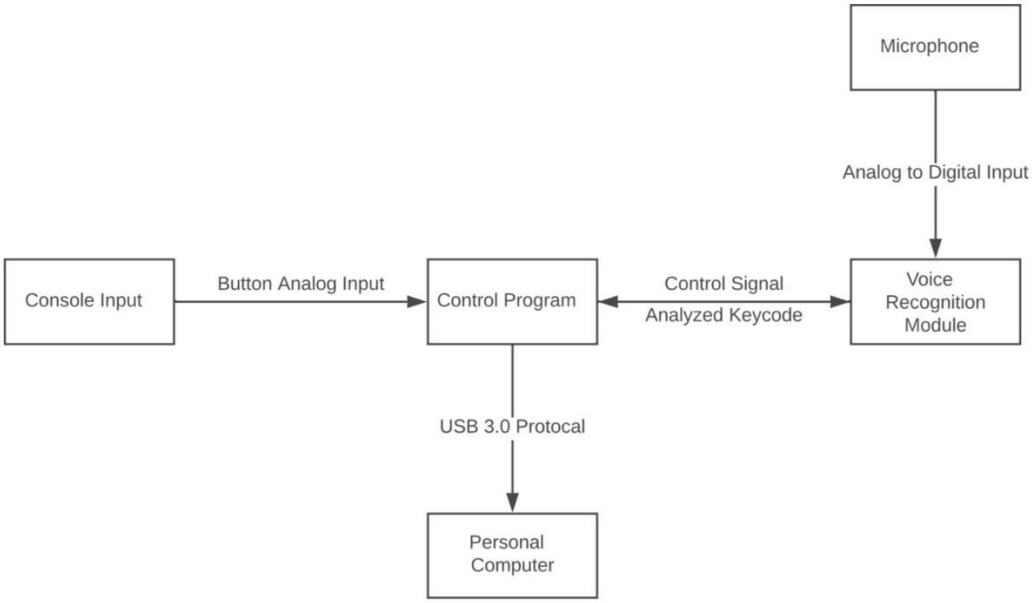

Figure 2.5.1 Data Communication

2.6 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:

$$0.7 + 0.5 + 3 = 4.2W \dots\dots\dots (1)$$
$$7.5 - 4.2 = 3.3W \dots\dots\dots (2)$$

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:

$$64k/s * 3B * 100 = 19.2MB/s \dots\dots\dots (3)$$

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.

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:

$$3 * 20\$/hr * 10hr/week * 10week * 2.5 = \$15{,}000$$

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	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 5	Integrate the microphone to the whole system	Integrate the microphone to the whole system	Test the accuracy the dataset with the trained model

Week 6	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 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" [2].
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;" [2]
    - 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;" [2]
    - 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] Ieee.org, "IEEE IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 29- Feb- 2020].

[3] 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