Final Report

USB dictation device

Qingyu Li, Wennan Zhai, Shengyu Ge

TA: Chi Zhang

Spring 2020, 05/08/2020
Group 66

Contents

Abstract

Countless people are suffering from limb loss or hands disabilities, so they need something that can help them use computers without the keyboards. The original solution tends to create a prosthetic device that can be used to type with the user's feet, which makes advances in bionic arm technology. We tend to design a portable USB dictation device supporting common platforms. 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.

1. Introduction – USB Dictation Device

1.1 Updated Problem Statement

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. 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.

1.2 Updated Solution

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.

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 Updated High-Level Requirements

- 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.

1.4 Updated Visual Aid

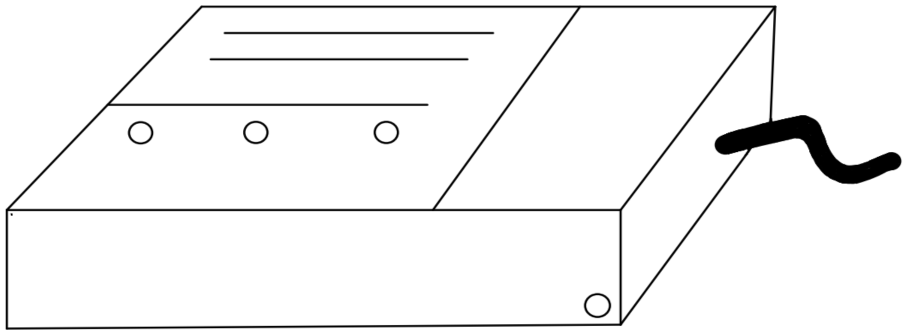


Figure 1.4.1 Product Overview

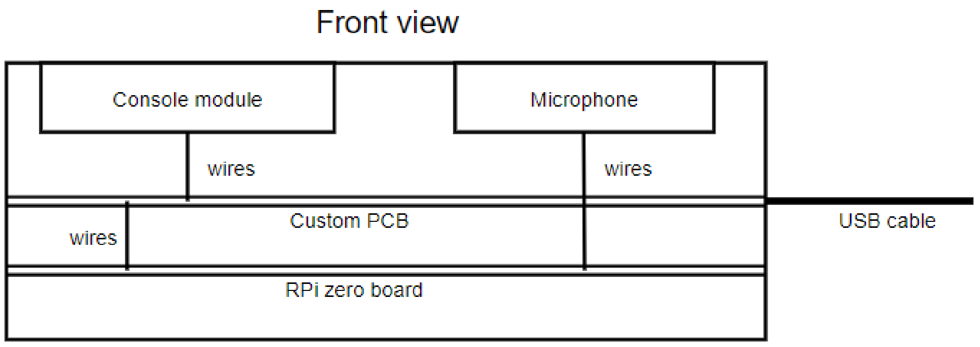


Figure 1.4.2 Product Front View

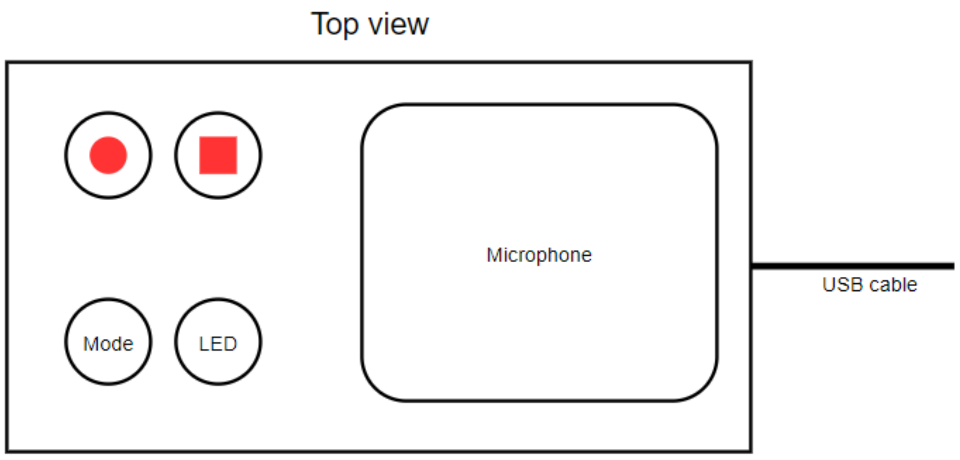


Figure 1.4.3 Product Top View

1.5 Updated Block Diagram

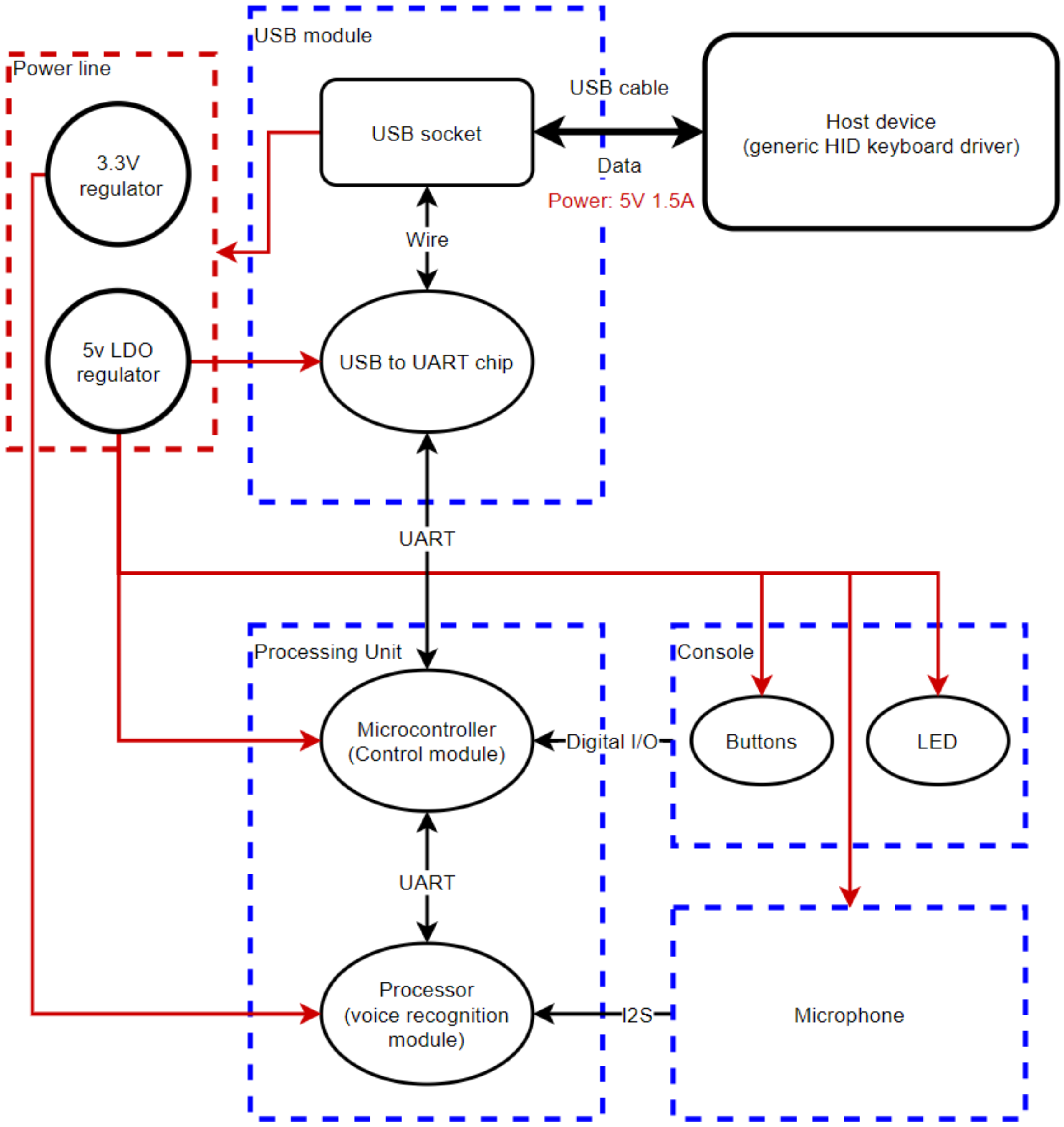


Figure 1.5.1 Updated Block Diagram

2. Second Project Implementation – USB Dictation Device

2.1 Circuit Schematics and Board Layout

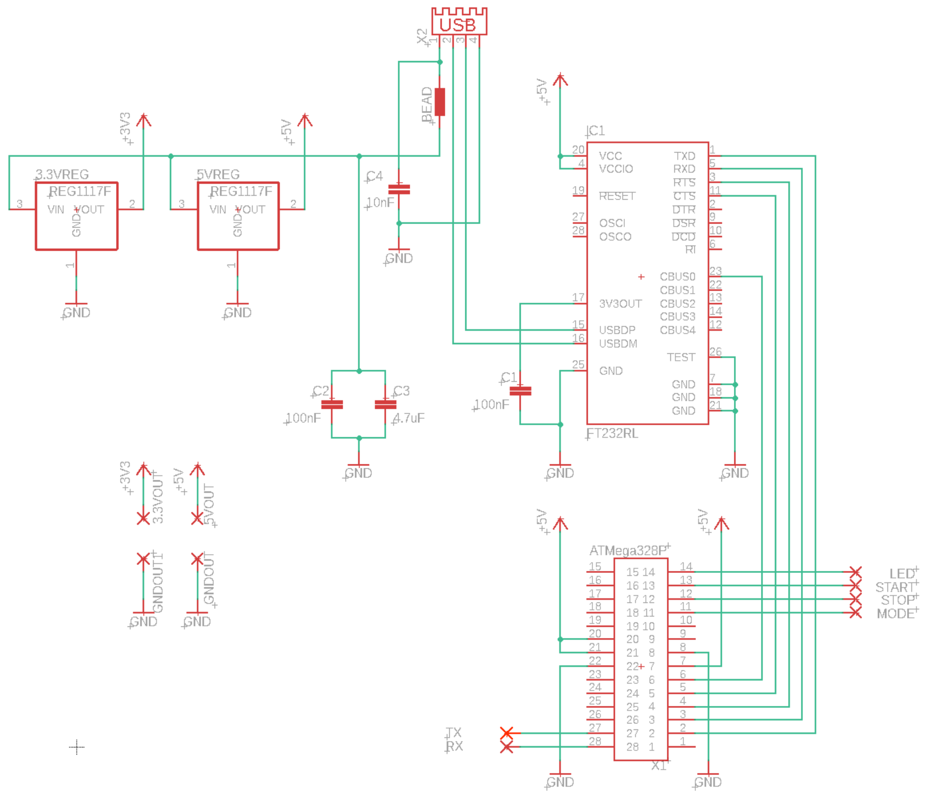


Figure 2.1.1 Circuit schematic

We designed the circuit for the main custom board and finished the prototype two-layer PCB layout for it. This board contains the control module and the USB module which are two key modules for the whole project.

The control module is the core of our project which manages all the data flow inside the device and the communication between the device and the host computer through USB. We used an ATmega328p here as our microcontroller because it is easy to work with and has relatively low power consumption. However, since the microcontroller needs to communicate with two other modules (USB module and Raspberry Pi) through UART protocol and ATMega328p only has one set of dedicated pins for hardware UART, an ATMega2560 which supports three hardware UART connections might be a better choice. We need to implement a simple software UART on two arbitrary pins on ATMega328p to compensate for this.

The USB module contains a serial-to-USB chip (FT232RL), a USB type-A female socket and two voltage regulators. We use the FT232RL chip to translate between the serial signal from the microcontroller and the USB signal from the host computer. The power pin on the USB socket is connected to two voltage regulators and they will draw current from the USB bus and provide power for all the other components in the device.

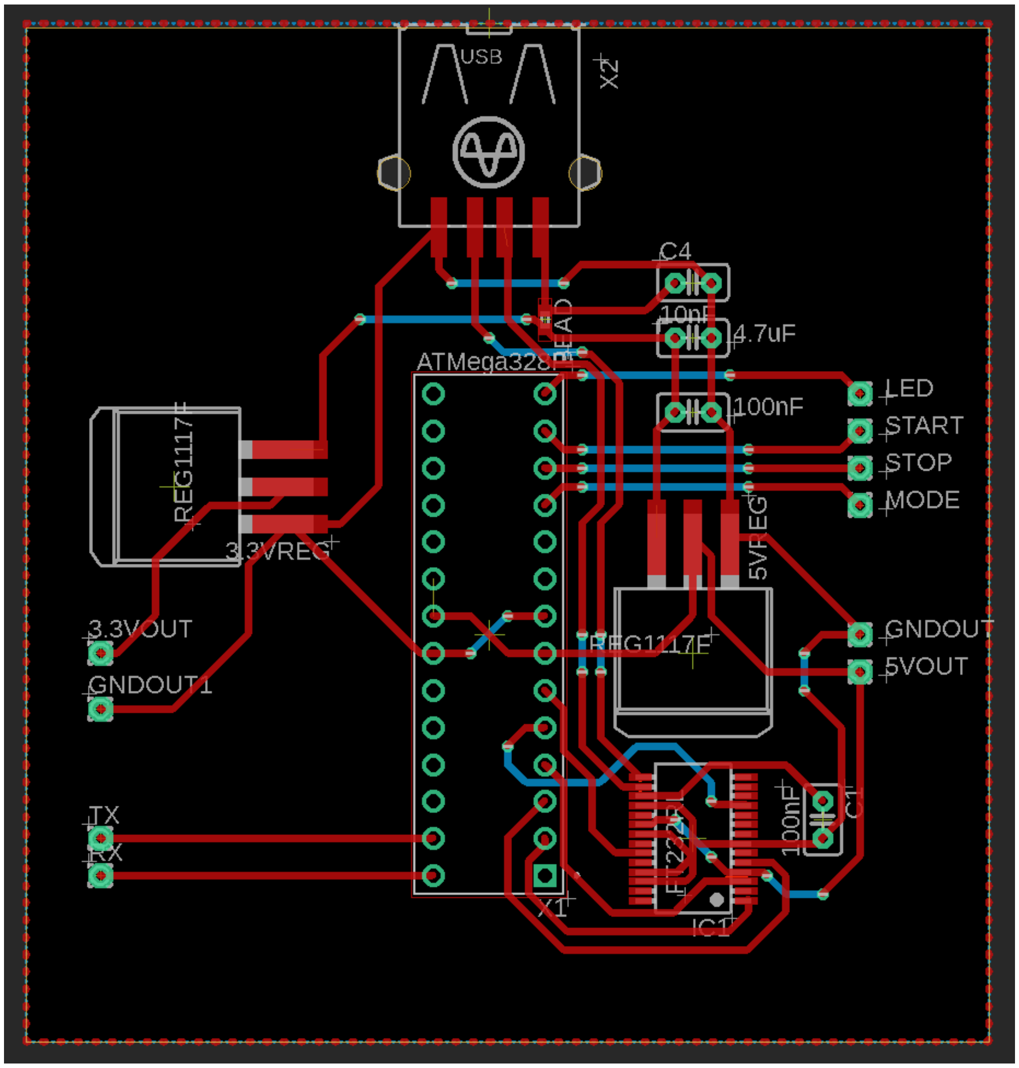
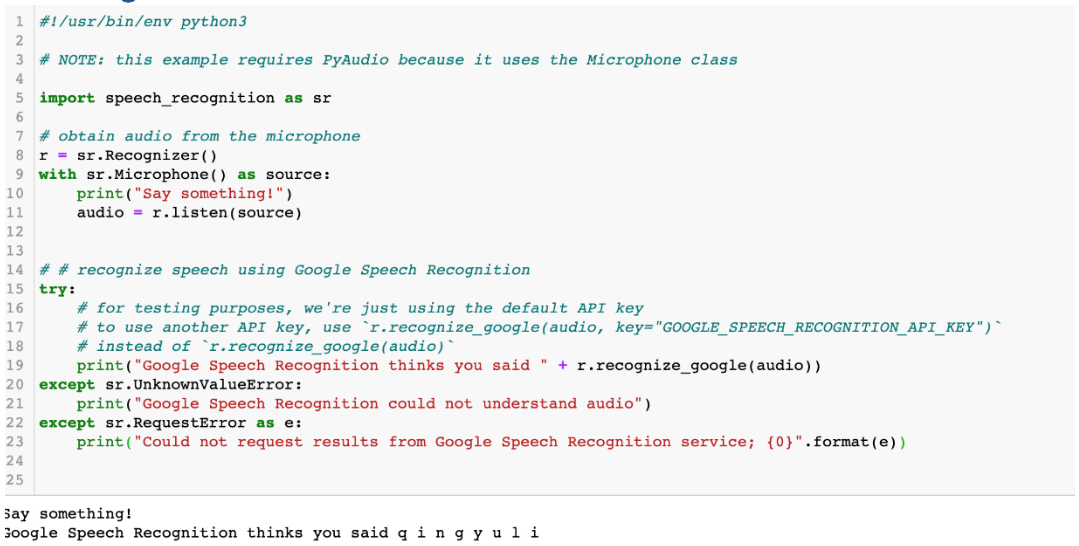


Figure 2.1.2 PCB layout for main board

There are several non-connected wire pads on the board since the components they are supposed to be connected to are not implemented on this board. The TX/RX pads and the 3.3VOUT/GNDOUT1 pads (on the left side of the board) are for the connection to the Raspberry PI Zero which is an independent board itself. The LED, START, STOP, MODE, 5VOUT/GNDOUT pads are supposed to be connected to the console module which will be another simpler PCB board consisting of an LED and three buttons.

This board is essential to the success of our project. It handles the power supply for all the other components and provides the hardware base for our firmware and software implementations. All the high-level requirements for our project rely on this board directly or indirectly.

2.2 Voice Recognition Module



```
#!/usr/bin/env python3

# NOTE: this example requires PyAudio because it uses the Microphone class

import speech_recognition as sr

# obtain audio from the microphone
r = sr.Recognizer()
with sr.Microphone() as source:
    print("Say something!")
    audio = r.listen(source)


# # recognize speech using Google Speech Recognition
try:
    # for testing purposes, we're just using the default API key
    # to use another API key, use `r.recognize_google(audio, key="GOOGLE_SPEECH_RECOGNITION_API_KEY")`
    # instead of `r.recognize_google(audio)`
    print("Google Speech Recognition thinks you said " + r.recognize_google(audio))
except sr.UnknownValueError:
    print("Google Speech Recognition could not understand audio")
except sr.RequestError as e:
    print("Could not request results from Google Speech Recognition service; {0}".format(e))


```

```
Say something!
Google Speech Recognition thinks you said q i n g y u l i
```

Figure 2.2.1 Google Voice Recognition Testing

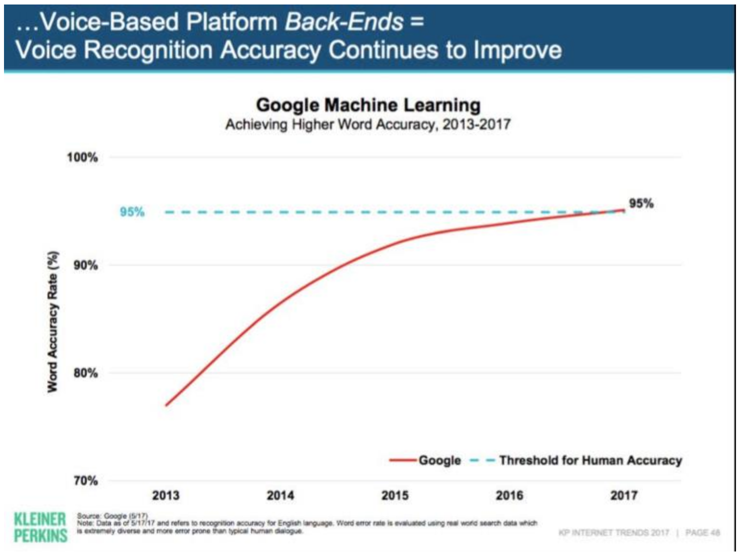


Figure 2.2.2 Google Voice Recognition Accuracy

We have tried using Google’s voice recognition module as our candidate for final implementation using Python. The set-up process is pretty straightforward and easy to implement. After it, it will keep listening to output chars or words according to the user’s voice inputs.

2.3 Control module

We have an interconnecting program to trigger this voice recognition module with respect to user's actions on the console such as pressing the start button or halting it by pressing the stop button on the console. The digital signals from the console are transferred to the Pi via the ATMega328P.

```
int state = 0;
int main(){
  while(1){
    bool start, stop;
    start = digitalRead(START_PORT);
    stop = digitalRead(STOP_PORT);
    // CS starts
    if(start)  state = 1;
    else if(stop) state = 0;
    /* more button responses or interactions
    ……
    */
    // CS ends
  }
}
```

Appendix 2.3.1 Part of Code for Interconnecting Program

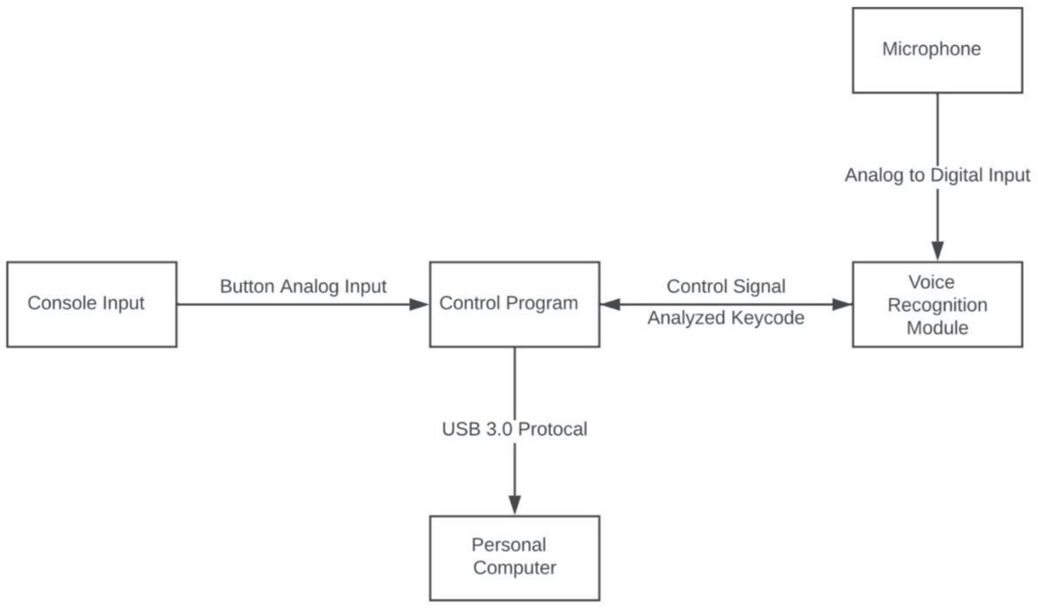


Figure 2.3.2 Data Communication

Making sure our data path is unblocked is critical for the whole project since having smooth inputting experience is our primary goal for this USB dictation device. To achieve it, we have to test the necessary channels for all of our components and make sure that we could send and receive data required for communicating, processing and outputting purposes. Some of the

protocols and channels are similar to what we implemented in the first project and we have already tried communicating between those specific devices, so we can just reuse those works to continue this project to ensure our data path works well.
Achieving the above data path is the decisive and core part for our first high level requirement since it makes sure everything else we do is meaningful.

3. Second Project Conclusions – USB Dictation Device

3.1 Implementation Summary

We were able to design and implement the circuit and PCB for the main board, part of the control software and part of the voice recognition module. And we have started working on the keyboard-like firmware which will be based on the QMK firmware. These components almost cover all the requirements for our whole project.

The circuit and PCB are designed and implemented by Wennan Zhai. These are the hardware backbone of the project and are necessary for the success of any other components.

The control module software is implemented by Qingyu Li and Wennan Zhai. The control software has two parts, one is responsible for the data flow and state transition inside our device, the other is responsible for communication with the host. We have implemented a prototype version of the first part which is crucial to our whole project because it connects all modules together and makes them function as a whole. The implementation to this part of the software will change a lot over time as we will optimize the data flow and improve performance and reliability.
The implementation of the keyboard-like firmware is pretty difficult since we need to read the code base of QMK and parse out the relevant information for our project. We chose QMK because it is open source and can save us a lot of time coping with the windows device drivers. We have started testing with the original firmware they wrote and eliminating the non-relevant parts for our project scope.

The voice recognition module is implemented by Shengyu Ge and Qingyu Li. This module is the purpose and the core functionality of our project. Although we do not implement the speech recognition algorithm ourselves, it is still a difficult task to make the algorithm work correctly in a timely manner. We also need to filter and make adjustments to the processed results to suit our different working modes. We will continue to improve the efficiency and correctness of this module along the line.

3.2 Unknowns, Uncertainties or Testing Needed

- System bootup time is a challenging task to accomplish:
  To achieve better user experience, we need to speed up the bootup time for the Pi we used to a maximum of 10 seconds, which is not possible unless we have our custom minimal OS configuration installed without any additional kernel components. So, this will be our most difficult goal to achieve for our plug-to-use product.
- Voice recognition module of our design may or may not need Internet connection:
  Depending on the situation, we may use a network-based voice recognition module, which requires us to determine whether we use the on-board WIFI connection for Pi exclusively or the Internet connection with USB 3.1 port. The performance including latency, processing speed and transfer rate varies with different methods and we have not been able to test them and compare them to determine our final implementation.

- USB module needs further and comprehensive testing:
  Providing enough power for the whole system via a single USB 3.1 connection is crucial for our project. So, in order to ensure the stability of our design, we have to use professional tools from ECE 445 Lab to conduct some comprehensive and systematic tests about the power delivery capability and data transfer speed.

3.3 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.

3.4 Project Improvements

- Replacing ATMega328P and Pi Zero with a single high-performance microcontroller:
  Given more time, we could refine our project by using a microcontroller to achieve speed boost and better integrity. Since Pi needs to boot up the whole system, but we only want it to run a specialized program, which is both a waste of time and computation resource. With a high-performance microcontroller, we could have more energy-efficient system with higher integration by having only one computation module instead of two new functional microcontroller we design and the voice recognition module we compile, our device will achieve significantly better performance.

- Solving the security issues:

As mentioned in the safety section, we currently do not have a perfect solution to protect the users' private information from hacking by other people with some purposes. However, within a year, I think we can have a more secure system by encrypting users' voice input and translated key code with some cryptographic hash functions, like SHA-2, SHA-256, SHA-512. With the pair of users' IDs and the value after the cryptographic hash functions, even if the data gets stolen, it is still safe since people who hack those data are not likely to crack the password or get the information they want.

- Adding hot plug functionality to our device:
  It will be very convenient for users if they can add or replace this device with more flexibility. Therefore, we can add the hot plug functionality to our USB dictation device, which makes this device be used just like the mouse, keyboard or portable hard drive, so users will also be able to interchange this device on different computers. This improvement will certainly benefit those people who need to work on more than one computer.

4. Progress Made on the First Project

4.1 PCB Design

We have finished our PCB design to have all the functionalities except the power module (in our design we separated the powering elements from the rest of the PCB design). Main components on this PCB are the ESP32 chip, the ATMega328P microcontroller and an RTC module. We have select signals from the ATMega328P to choose from tens of motors and control signals to control the H bridge for instructing the direction of movement. There are also communication channels between the ESP32 chip and the microcontroller to exchange data. More on the PCB are the pins for incoming power delivery, I2C connections for the display and speaker and additional components such as LEDs and buttons for interactive purpose (necessary capacitors and resistors also on board).

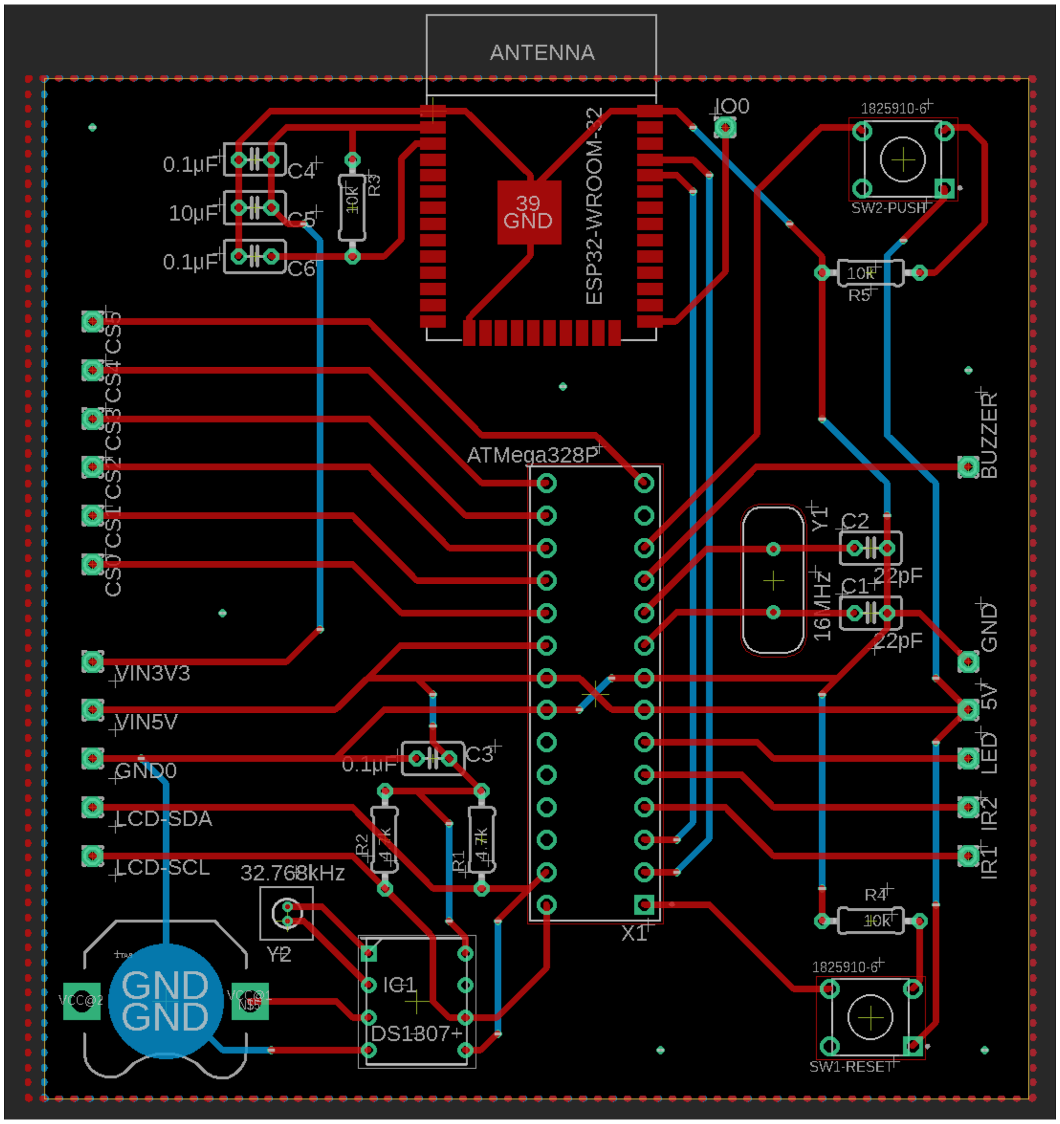
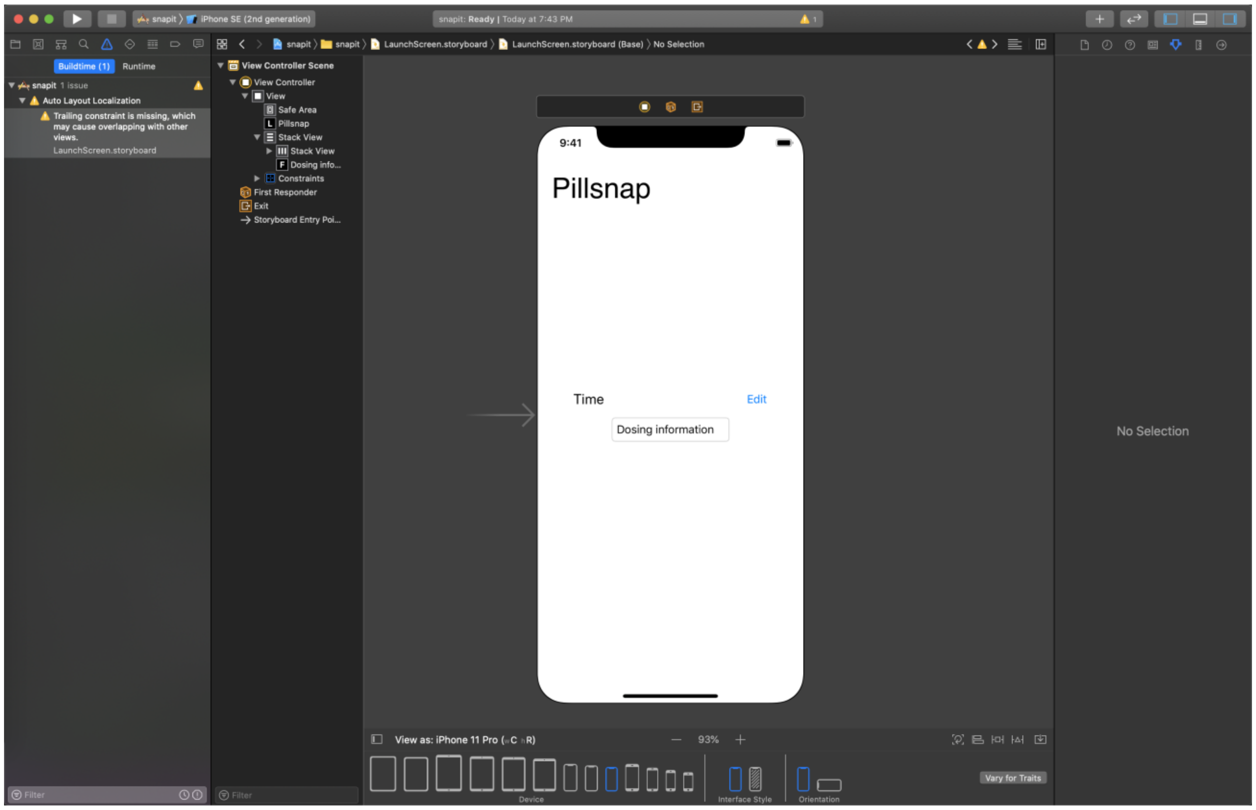


Figure 4.1.1 Processing PCB Design for Automatic Pill Dispenser

4.2 Software Design

We have also tried designing and implementing the software design for our project but only managed to come up with some ideas regarding the interface of the software. Below is our implementation of our App’s home screen. There is a dedicated area where user could add different remainders for taking pills by setting the time and compartment number of the needed pills. Moreover, we are planning to add multiple-user feature for our app to enable many to share the same automatic pill dispenser.



Trailing constraint is missing, which may cause overlapping with other views.
LaunchScreen.storyboard

Figure 4.2.1 Overall Layout for Pillsnap App

5. Reference

[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].