# Plug and Play Stenography Keyboard

**Team number: 2**

**Team Members:**
**Haoqing Zhu** (haoqing3@illinois.edu)
**Soham Karanjikar** (sohammk2@illinois.edu)
**Rishi Krishnan** (rishik3@illinois.edu)

**TA: William Zhang**

**September 12, 2020**

**Course: ECE 445**

# 1. Introduction

## 1.1. Objective

There is currently no way for verbally disabled individuals to communicate with others in real time. Sign languages do provide a way out, but it is very impractical to expect everyone else to learn it in order to communicate. On the other hand, typing on traditional keyboards is too slow, they do not offer any solution to this problem.

Our solution is to build a plug and play stenography keyboard that has all the necessary hardware/software built within, so no extra installation is needed on the host side. This keyboard will offer typing speeds close to what humans speak at (180 - 250WPM).

## 1.2. Background

Traditional steno keyboards are very expensive as they are usually used only in court. They are also bundled with many court-reporting specific softwares which would be useless in everyday communication. Our project on the other hand builds the keyboard for a fraction of the cost while providing the same benefits.

Since our keyboard does not require much technical knowledge such as installing software or coding, it is very marketable to various demographics and the only necessary knowledge is learning how to use a stenography keyboard. Another benefit is the ability to change the dictionary in our keyboard so that users can make their own strokes for new words or new phrases, maybe even use a different language.

## 1.3. High-level Requirements List

i. The user is able to convert his/her personal dictionary (assumed in Plover JSON format) into our custom format (for efficiency), and load it onboard.
ii. The device is able to translate strokes taken from keyboard to keystrokes according to the corresponding dictionary entry.
iii. The user is able to dynamically add/delete/modify dictionary entries on the fly.

## 2. Design

### 2.1. Block Diagram

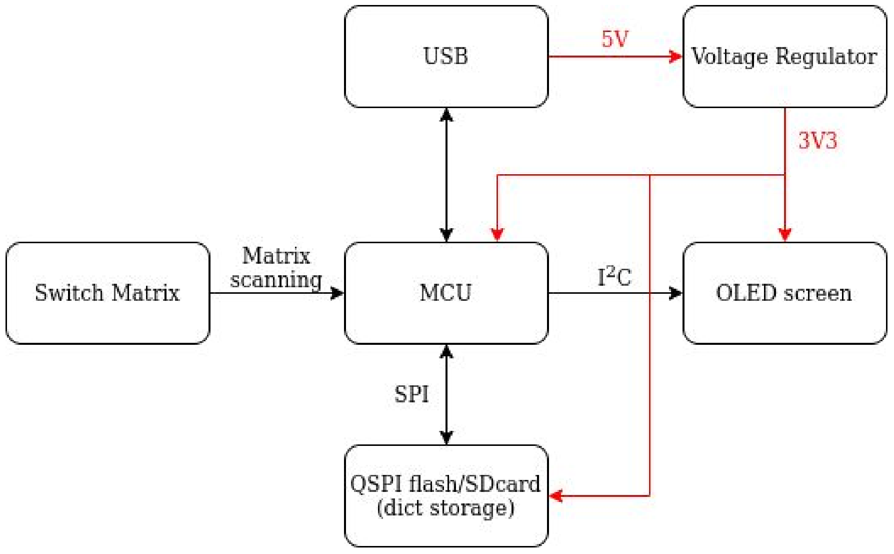

*Figure 1: Hardware Block Diagram*

The dictionary compiler would allow users to convert their personal dictionary into what we would be using. The SDcard or the SPI flash would store the dictionary onboard and the translation engine will translate the strokes from the keyboard into output according to the stored dictionary. The user interface through OLED and the keyboard itself would allow the user to modify the stored dictionary.

### 2.2. Functional Overview

2.2.1. Dictionary Compiler

The dictionary compiler is required to bootstrap the dictionary onboard. It will take a Plover JSON dictionary and convert it into a binary format that's easy for the MCU to read and manipulate.

2.2.2. Dictionary Storage

The SDcard or SPI flash and the associated driver code will take care of storing the binary dictionary. It will also provide fast random read/write access to the dictionary.

2.2.3. User Interface

The user interface code should communicate with the screen and display related information for the user about the keyboard status. It should also guide the user when editing the dictionary.

2.2.4. Translation Engine

The translation engine should read the dictionary stored onboard and control the output at a high level when the strokes come in from the keyboard. It should be able to translate at least 4 strokes per second.

2.2.5. Matrix scanning & HID handling

Matrix scanning and handling HID reports are essential parts for a keyboard, and they will be handled by the QMK firmware.

2.2.6. OLED Display

The OLED display should be large enough to show all the information related to the keyboard status and dictionary editing.

2.2.7. Switch Matrix

The switch matrix should have enough keys to comfortably stroke steno chords and be in ergonomic positions that the user can use without much effort

## 2.3. Risk Analysis

The biggest risk in this project is the dictionary editing support. A simple implementation should be fairly easy to do, given that a lot of the requirements are inplace. But a more complete implementation would require a more careful analysis of the Plover command set and narrow down the requirements. It may also require a redesign of the binary dictionary format to allow more efficient use of the available storage space.

## 3. Ethics and Safety

Many components such as the MCU and QSPI flash used in this project will be susceptible to electrostatic discharge (ESD). Precautions must be taken to prevent damaging these parts such as using anti-static gloves and ESD wristbands.

For ethics, we hold responsibility for our project, which is the first rule in the IEEE Code of Ethics [1]. In some sense, our keyboard can be thought of as a giant macropad capable of outputting text or keystrokes at high speeds. Using this keyboard as a tool to conduct abuse such as spam is unethical, according to IEEE Code of Ethics #8 [1]. In addition, use of the keyboards as button macros is a violation of IEEE Code of Ethics #4 [1].

## 4. Contingency Plans

When/If we need to move online, the challenge we would face would almost be the same as what we have now, which is the difficulty to get hardware for testing. This is something we can overcome as we can meet each other on campus even if not permitted to meet in the ECEB lab. In addition, at least one of us has the lab equipment needed (soldering iron, oscilloscope, etc.) So we are not too worried about transitioning to fully online as we are having the same struggles even with the current situation where classes are in-person.

## References

[1] “IEEE Code of Ethics,” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 17-Sep-2020]