Project Proposal: Plug and Play Modular Keyboard

1. **Introduction**

    - **Contributors:** Daniel Chen, Fangqi Han, Christian Held
    - **Objective:**
        i. While traveling or while at home one will need two different keyboards to work. Additionally, left handed typers can find the numpad on keyboards to be awkward to use. The typical solution to this is to either buy another small keyboard to travel or one can just start with a smaller keyboard.
        ii. Our solution to this problem is to make a keyboard with detachable modules that allow the user to conform to their working environment while also giving them maximum utility. The user can simply add or remove modules to the keyboard by plugging or unplugging their TRRS connectors. The adaptable firmware will also allow the user to still have the keypresses available even when in a condensed mode.
    - **Background:**
        i. A modern job-oriented citizen needs to have technology that lets them get their goals accomplished in a timely manner. In their office they have the space and comfort to use effectively a full size keyboard with all the amenities that allows them to work at maximum efficiency. Whilst traveling, they do not have such luxury. One must have multiple keyboards or one undersized keyboard while traveling. Independent reports that the average worker spends about 1700 hours a year in front of a keyboard and that time should be optimized.
        Source:https://www.independent.co.uk/news/uk/home-news/office-workers-screen-headaches-a8459896.html
    - **High-level requirements list:**
        i. The Main Keyboard is functional with at least 61 usable keys.
        ii. Auxiliary Modules are functional and able to plug into the main keyboard with at least 10 usable keys on each module.
        iii. Firmware is workable and partially customizable with at least 20 changeable keys.

2. **Design**

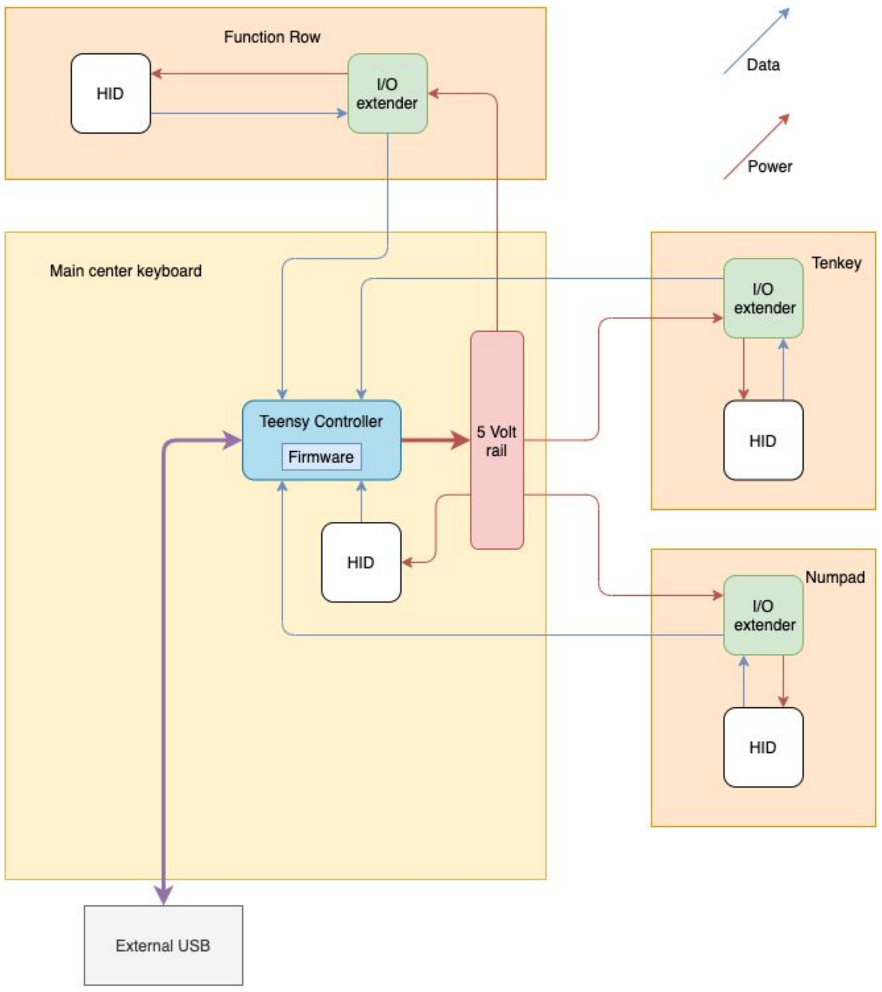
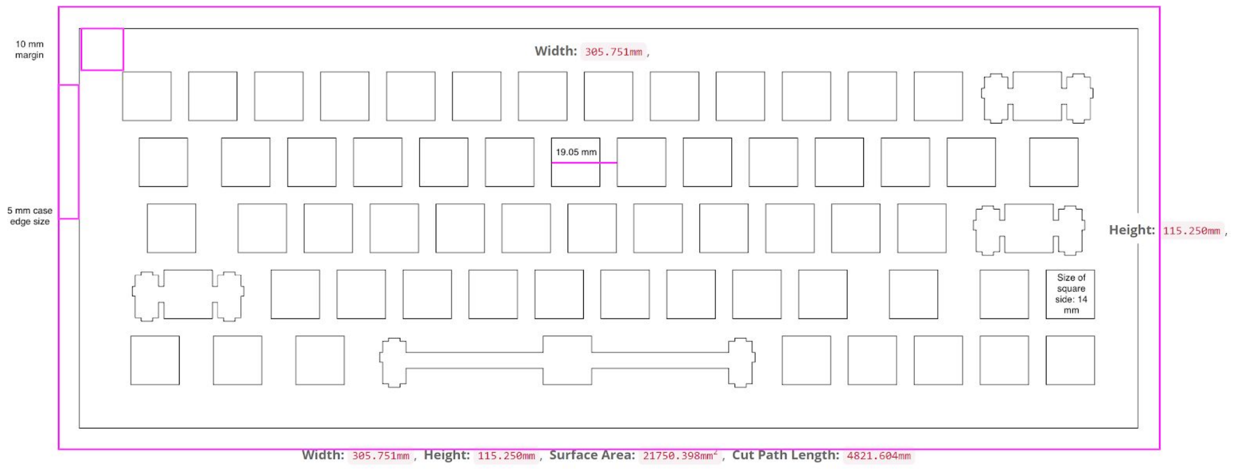


Figure 1. Block Diagram

An external USB powers the main microcontroller that is the operational key piece. With just the base keyboard connected solely the keyboard can still function. The other modules will take power and give data through a wired connector. When connected the microcontroller will recognize their data through an I/O extender from the respective module. The firmware on the microcontroller will be able to process any number of key presses, but the main limitation will be on the data feed back to the computer.

- **Physical Design:**


10 mm margin
Width: 305.751mm ,
19.05 mm
5 mm case edge size
Height: 115.250mm ,
Size of square side: 14 mm
Width: 305.751mm , Height: 115.250mm , Surface Area: 21750.398mm² , Cut Path Length: 4821.604mm

i. The sizing and construction of the key layout and housing is complicated and would require significant explanation to go through the whole system. However, there are resources like http://www.keyboard-layout-editor.com/#/ and http://builder.swillkb.com/ that help in this design endeavor. The bulk of the work is easily procured through open source and free software.

- **Functional Overview:**

i. **Key Switches (HID)** - The human interface devices for our keyboard are mechanical key switches which when pressed shorts a connection and completes a circuit. When these switches are combined into a matrix and hooked up to a microcontroller, firmware for the microcontroller can match these completed circuits to which keys were pressed and put letters from our fingers on to the screen. As the main component that interacts with the user, this is one of the most important components and provides most of the usability and with its specific configuration will do much of the organizing for the microcontroller.

ii. **Microcontroller** - The microcontroller receives power from the computer and disperses it through the connected systems. This power is sent through the matrix that contains all the keyboard switches, as well as to the external components through the I/O extenders. The HID and I/O extenders will send back data they receive, which is organized and sent to the computer as typing inputs. The power and the thinking of the keyboard is performed by the microcontroller.

iii. **I/O Extender** - These components on the exterior modules are the hub for power and data transfer on the modules. They receive power from the main 60% keyboard and use it to allow the HIDs to send data back to the

I/O extender. It interprets the data and returns it back to the central keyboard via I2C connection.

iv. **Connectors** - The wiring between the main hub keyboard and the modules will be a TRRS cable. This is commonly known as a 3.5 mm jack. The TRRS version has four separate wires associated with it. The four things transferred will be five volts, ground, SDA, and SCL (the latter two are part of the I2C protocol and are the only lines that return from the modules to the main hub).

v. **Firmware** - Firmware is software coded in the microcontroller that will detect the attachedness of modules, scan the keyboard matrix for pressed and released keys, interpret keys based on preset layouts or user determined values, and send out corresponding signals through an USB connection. The firmware also provides programmability, allowing users to modify their keyboard layout to produce user-determined characters or potentially commands. Examples include changing the fn layers, having programmable keys, and other features that would help with user productivity and customization.

- **Block Requirements:**

Block Diagram Item	**Requirements**
Key Switches	Be able to translate a press into a short on the circuitry within 20 ms.
Microcontroller	Have a polling speed that allows a complete scan of the keyboard in around 100 ms. Each key should be able to be read (assuming the module is attached). Be able to run the firmware.
I/O Expander	Have a polling speed that can interpret key presses and send an I2C signal to the microcontroller within 10ms .
Connectors	Capable of sending four separate signals across a distance of 1 - 6 inches.
Housing	Able to hold all 61 - 10 keys depending upon which part of the keyboard is being housed. Have a

	reasonable amount of strength that the average use will not permanently deform the housing.
Firmware	The combined keypad matrix should have at least 101 entries, corresponding to 101 keys of a traditional keyboard. Keypad matrix for the default keyboard without extensions should cover around 60% of the keys, while each attached auxiliary module would extend the matrix by their size. The firmware needs to periodically scan pressed keys, map their coordinates to find their values, and send out signals accordingly. The time for a complete scan of the keyboard should take around 100 ms. The firmware also needs to periodically poll the status of attached auxiliary modules and adjust the scanning range accordingly. The keypad matrix should be customizable through user input, allowing the user to change corresponding functions of programmable keys.

- **Risk Analysis:**
    i. The Teensy must be able to process all the information being given to it at one time. If the the controller has issues reading the I2C or general bookkeeping issues the keyboard will not function well or at all.

3. **Ethics and Safety**

- According to 1.2 of ACM code of conduct, we should design every part that could come in contact with the user to be safe to touch and interact with. Such contact points should not abrade the user in any way. With using ACM 2.8, any firmware or drivers we utilize will be our own construction or open source material. Our design for this keyboard is intended to be touched and used directly by humans. This means that we should adhere to ACM 3.3 by creating designs that are ergonomic and help the user have a more efficient experience, a crux of our background to this project.