Foot Keyboard

Team 42 - ECE445 - Spring 2020 - Design Document

Neva Manalil, Nick Halteman, Aditi Panwar
TA: Johan Mufuta

17 April 2020

Table of Contents

1. Introduction

1.1 Problem and Solution Overview

There are nearly two million people in the United States currently living with limb loss and around 185,000 more amputations occur each year [1]. In order to help those facing limb loss, there are prosthetics amputees can purchase. However, even if someone has a prosthetic, it does not guarantee them the ability to type as many prostheses use pre-programmed gestures and are not very dexterous. Typing requires many different precise gestures which realistically cannot be programmed to the arm. The Universal Declaration of Human Rights states in Article 19 that everyone has the right, "to seek, receive and impart information and ideas through any media and regardless of frontiers" and Section 32 expands that to include the Internet as a human right [2]. Being able to connect to the internet with a computer is crucial to our society today, but current prostheses greatly hinders the ability for amputees to use computers. Alternative solutions involve text to speech software. But, there are limitations as it may be prone to inaccuracies and the software does not allow for function keys such as ctrl or alt [3]. We propose a device that will allow those with hand or arm loss to perform keyboard functions with their feet and have the autonomy to use a computer on their own.

Our solution would consist of a keyboard that is built to be operated by only a person's feet. Because people are typically much less dexterous with their toes, we will not be using a conventional array of buttons, each corresponding to a letter. Instead we have designed a direction based selection tool to take input by moving the user's entire foot. The top of our keyboard will have two large grip pads, one for each foot, that can be pushed in any direction. The left foot will serve as a group selector, with each direction selecting a group of keys. The right foot will then select a single key from that group of letters to input. We also have a button to toggle between uppercase and lowercase letters. Allowing for eight directions per foot plus the neutral center position and the case switch button, we can achieve up to 162 unique inputs. This enables us to easily map all 113 unique keystrokes a standard keyboard can perform. Additionally, the keyboard also has mouse functionality to ensure the computer can be completely controlled through only foot inputs. As memorizing the various inputs will take some time, we will also design a cheat sheet that can be placed on a desk (where a keyboard would normally be) so the user can quickly identify what foot positions they need for each key.

1.2 Visual Aid

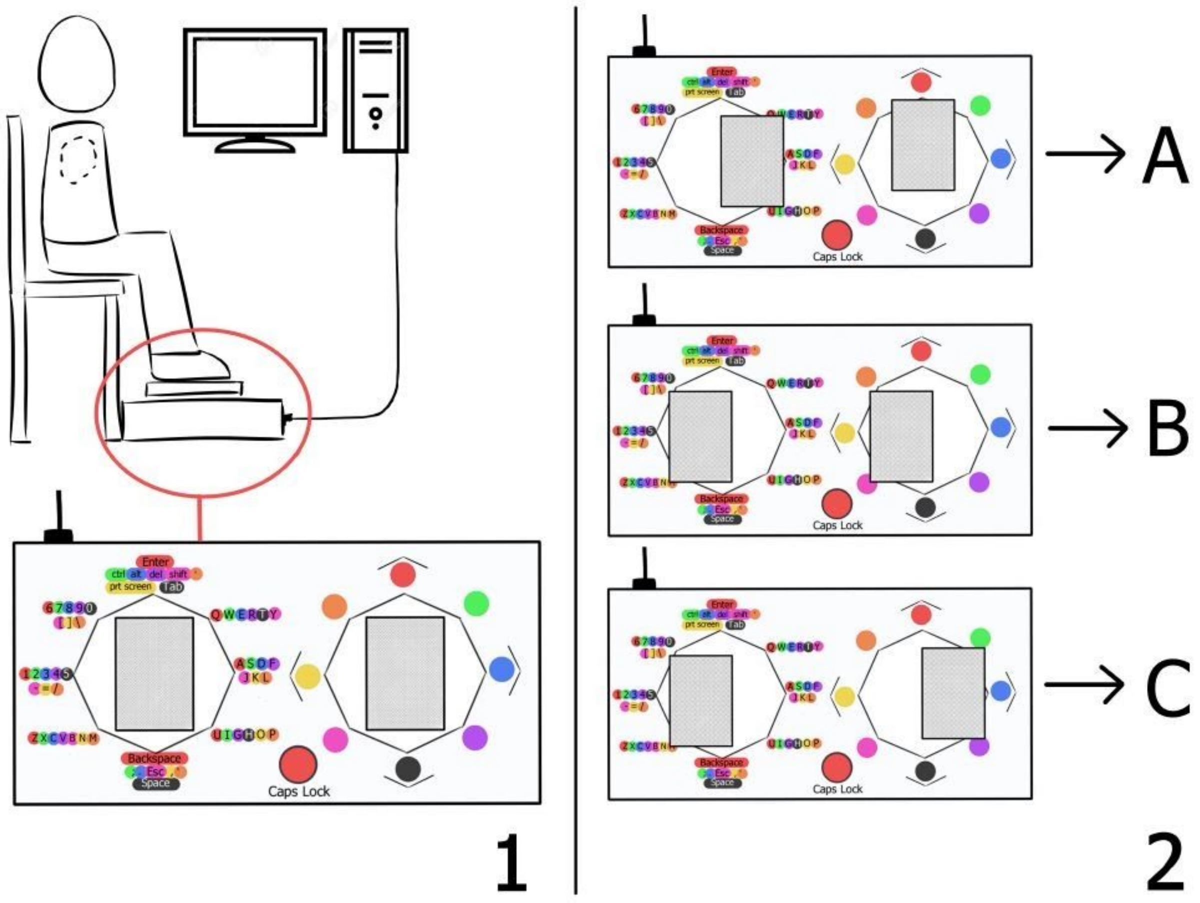

Fig 1 Visual Aid: 1. The foot keyboard connects to the computer through USB. The user must place their feet on the foot pads and slide the foot pad to the corresponding character and color groups to select a character or function key. To do keyboard selection the left foot must select a group first, otherwise the right foot may be used as a mouse input device. 2. Foot orientation to select characters A, B, and C. Only when both feet have made a selection will the character or function be executed.

	↑	↗	→	↘	↓	↙	←	↖	**Stomp**
↑	enter	ctrl/cmd	alt	del	tab	shift	prt screen	` (~)	home
↗	q (Q)	w (W)	e (E)	r (R)	t (T)	y (Y)			
→	a (A)	s (S)	d (D)	f (F)		j (J)	k (K)	l (L)	pg down
↘	u (U)	i (I)		g (G)	h (H)		o (O)	p (P)	
↓	backspace	; (:)	. (>)		space	esc	, (<)	‘ (“)	end
↙	z (Z)	x (X)	c (C)	v (V)		b (B)	n (N)	m (M)	
←	1 (!)	2 (@)	3 (#)	4 ($)	5 (%)	- (_)	= (+)	/ (?)	pg up
↖	6 (^)	7 (&)	8 (*)	9 (( )	0 ( ))	[({)	] (})	\ (|)	
Stomp	↑		→		↓		←		win
Neutral	Mouse Movement								Left/Right Click

Fig 2 Input Mapping: “Cheat sheet” that has the complete character mapping. The left column represents the left foot character group selection and the top row represents the right foot color selection. Characters in parentheses are the alternate characters when the “caps lock” button is enabled or when the previous character input was “shift.” If there is no character in parentheses, then the default character or function associated with that input will be selected. An input without anything mapped will not do anything. When there is no left foot input, the right pad controls the mouse. The right stomp button alone will function as a left click on a short press and a right click on a long press.

1.3 High-Level Requirements List

1. The keyboard must be able to replicate the 95 unique characters and 18 function keys found on a standard keyboard along with two mouse buttons and directional mouse movement using only the user’s feet.

2. The polling rate of the keyboard must be at least 36Hz.

3. The keyboard microcontroller must be able to send the correct keystrokes and mouse inputs to the computer.

2. Design

2.1 Block Diagram

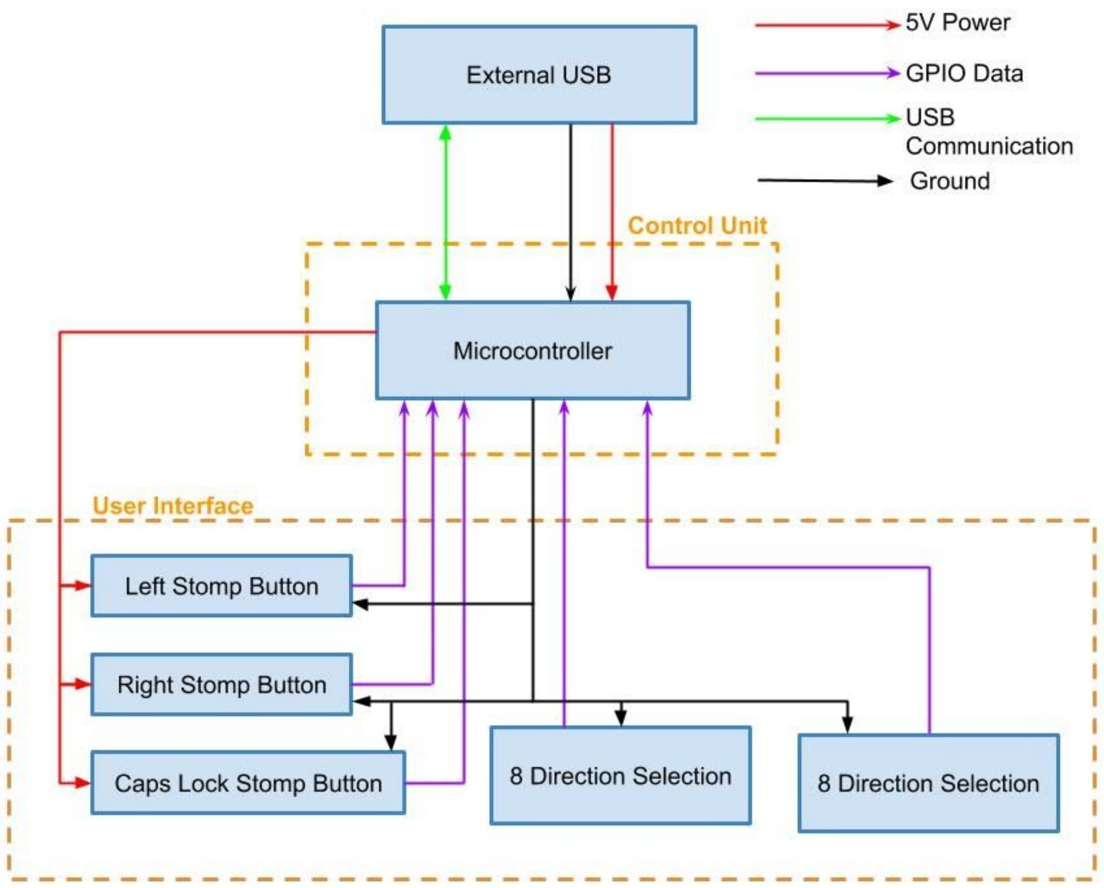

Fig 3 Block Diagram: The user interface gets the user input from the analog stick like device the user controls with their feet. The control unit will convert the user input to a keystroke or mouse movement and then execute the command on a computer.

Our design has two main subsystems, the user interface and the control unit. Through the user interface the user can provide their keyboard and mouse inputs by using the two 8 direction selection modules. Each of these consist of a set of switches which when active will determine the user's selection for a group of characters as well as a center stomp button for additional inputs. The external stomp button allows the user to alternate between capital and lowercase letters and alternate key inputs as seen in Figure 2. The control unit is responsible for converting the user input to the respective character or function as well as sending a signal to the computer via USB to execute the keypress or mouse input.

2.2 Physical Design

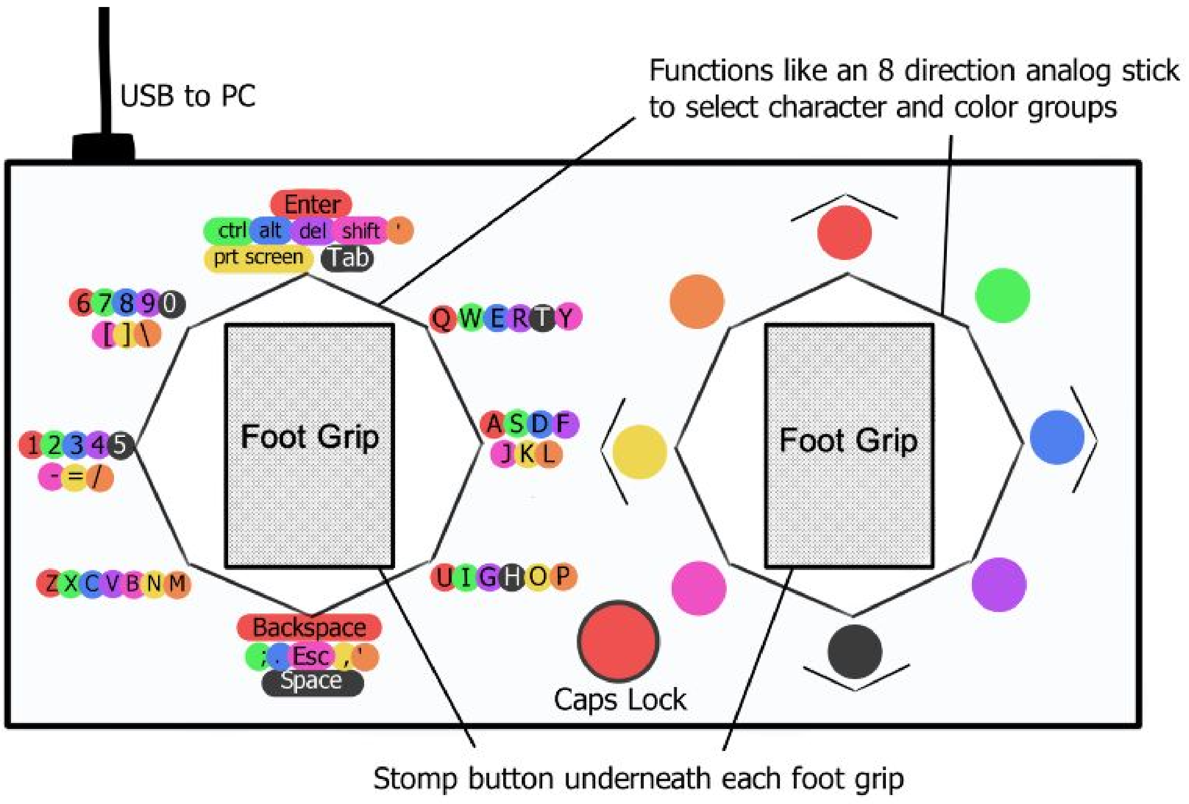

Fig 4 Physical Design: Keyboard keys are each set to a character and color group which are selected by moving the foot grips in the corresponding directions. Mouse control may be enabled by moving the right foot grip while the left foot grip is in the neutral position. Stomp buttons under the foot grips toggle between characters and navigation keys as well as allow for right and left mouse clicks. The caps lock stomp button enables alternate characters to be typed. For readability the full character mapping is not shown, see Figure 2 for the complete character mapping.

Each foot will control a large eight direction analog stick like device as shown in Figure 4. Each left foot position will control a group of characters (ex. A, S, D, F, J, K, L) and the right foot will select a specific character from that group. When the user slides their foot to any of the eight directional positions a switch will be activated to send a GPIO signal to the control group. In order to use the keyboard the left foot must make the selection first then the right foot. When the right foot is activated first, the keyboard enters mouse mode and the right foot pad will control mouse movement. In order for the user to toggle between capital and lowercase letters there is an external caps lock stomp button. The stomp button under the right foot will also serve as the left mouse click and right mouse click when held for three seconds. Additional stomp buttons are underneath each foot grip which can be toggled to access navigation keys. The user will also have a cheat sheet as shown in Figure 2 that can be placed on a desk to help identify which groups each character is in. Function keys will stack to allow multi-key shortcuts. The user will input all keys they want held in sequence starting with a function key and click the mouse to release them. As a special case, shift and then a letter will clear immediately.

2.3 User Interface Subsystem

The user interface consists of 16 switches, eight for each foot which are toggled when the user moves their foot to one of the eight directional positions. The switches will be placed radially around the moving foot pads and actuate when the pad has fully moved in a direction. There are also three stomp buttons, one under each foot grip when in the neutral position and one externally to allow for additional keystrokes such as the arrow keys and caps lock. Only if both left and right feet have made a selection will a signal be sent to the computer to do the selected keyboard action. Otherwise, the keyboard will function as a mouse input device.

The directional selection is composed of 16 Cherry MX keyboard switches. The Cherry MX switches have a bounce time of less than five milliseconds to minimize latency. The physical component is designed in such a way that the foot pad will not be able to move further than the minimum distance to activate the switch (0.16in) in order to not crush the switch (see Figures 5-7).

Three FS5700 pushbutton switches will be used as the stomp buttons. These buttons are the same ones typically found on guitar pedals and are thus rated to withstand the weight of a human leg. The two stomp buttons under each foot pad are momentary action switches which must be held to be active. The caps lock button is an alternate action switch which must be pressed once to activate and pressed a second time to deactivate it. The two buttons under the foot pads function to toggle between character/function key input and the navigation keys. The right stomp button also can take mouse clicks when the left foot is in the neutral position. The caps lock button, when toggled, will function the same as holding down the shift key on a standard keyboard.

Requirement	Verification
The foot pads return to center when the user removes their foot.	Slide the foot pads in each direction and ensure that it is able to return without needing the foot to push it back.
Pushing the foot pad fully in any direction will always engage the corresponding switch.	1. Use a benchtop multimeter to probe the output of each switch. 2. Slide the foot pad in each direction and determine if the output is pulled to ground.

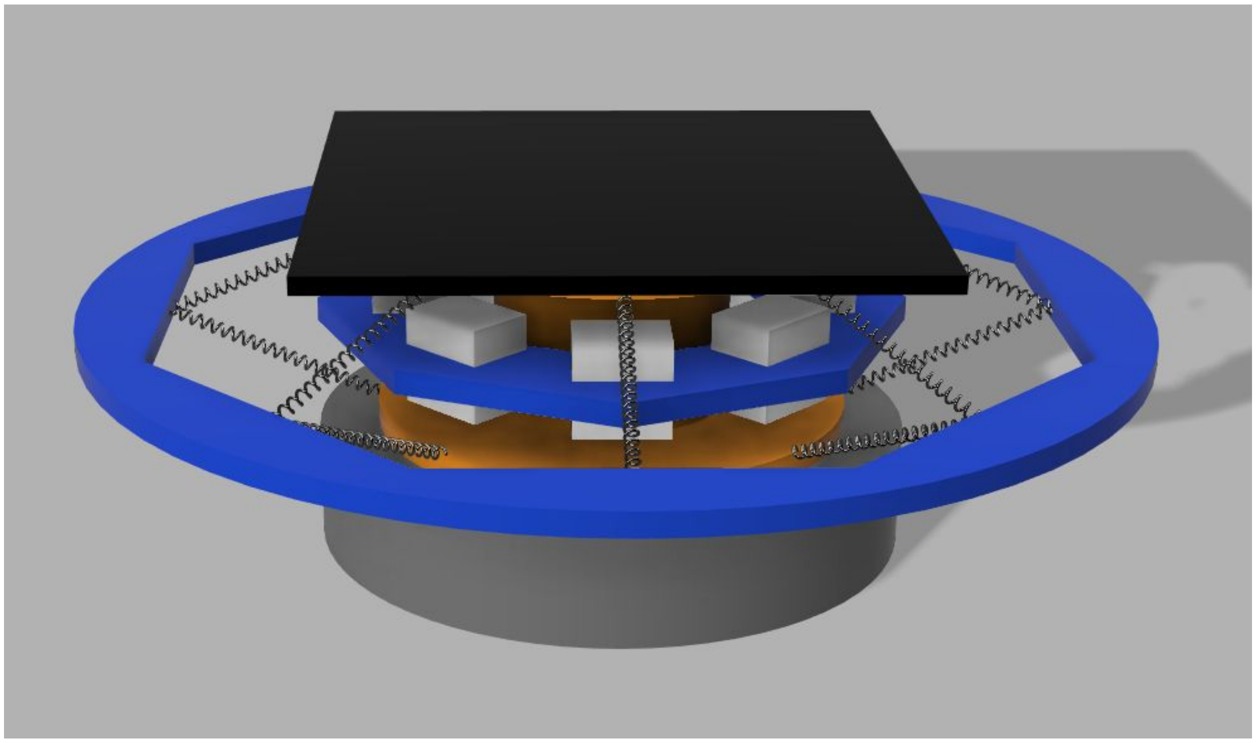

Fig 5 Foot Pedal Assembly: Below each foot grip (shown in black) a stem (shown in orange) that is free to move in any direction to actuate various switches. Springs are used to recenter the pedal after each actuation.

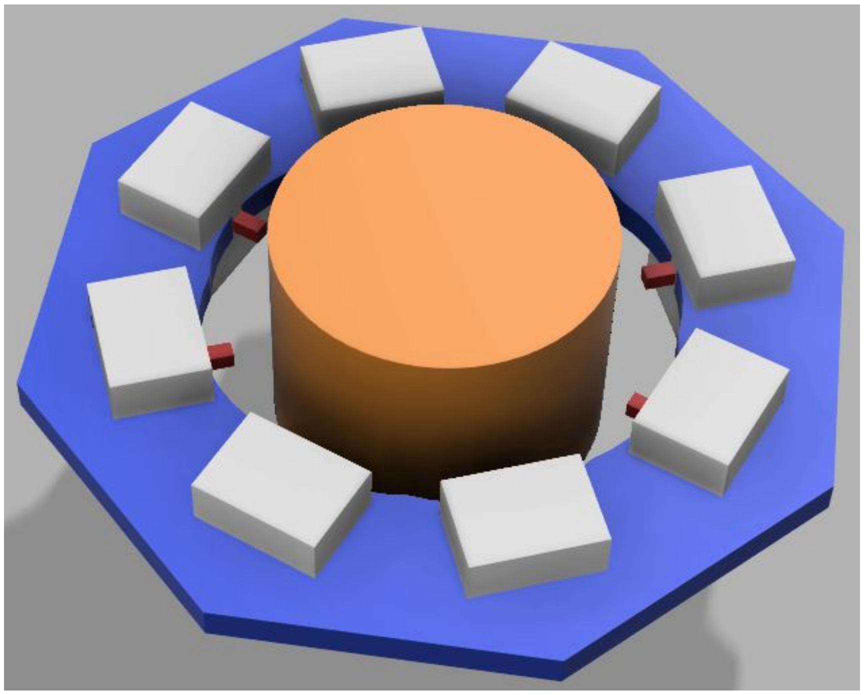

Fig 6 Direction Selection: Below each foot grip is a stem with eight switches mounted around it. The blue support ensures the orange stem cannot crush the switches and helps guide the stem in the correct direction..

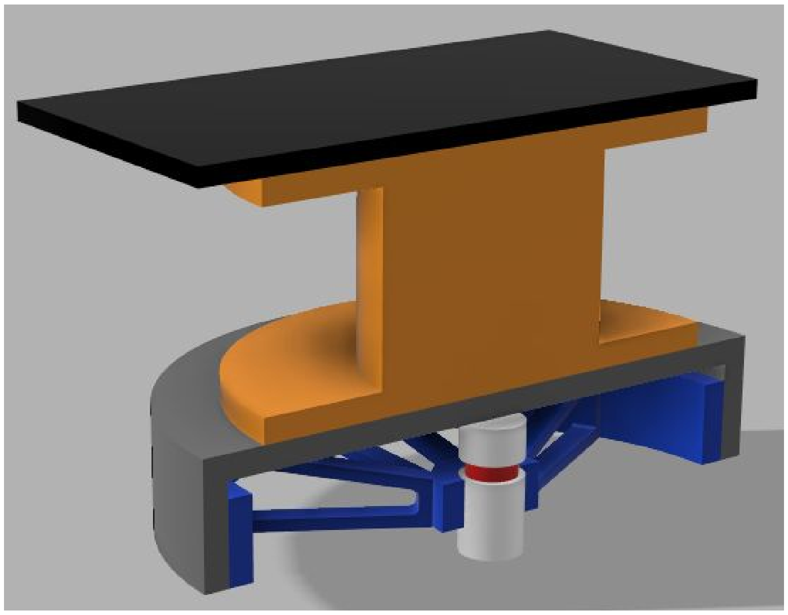

Fig 7 Stomp Switch: Below each foot pad is a stomp switch. When depressed, the blue support material contacts the gray cap preventing the switch from being crushed. The orange stem is free to slide over the gray cap.

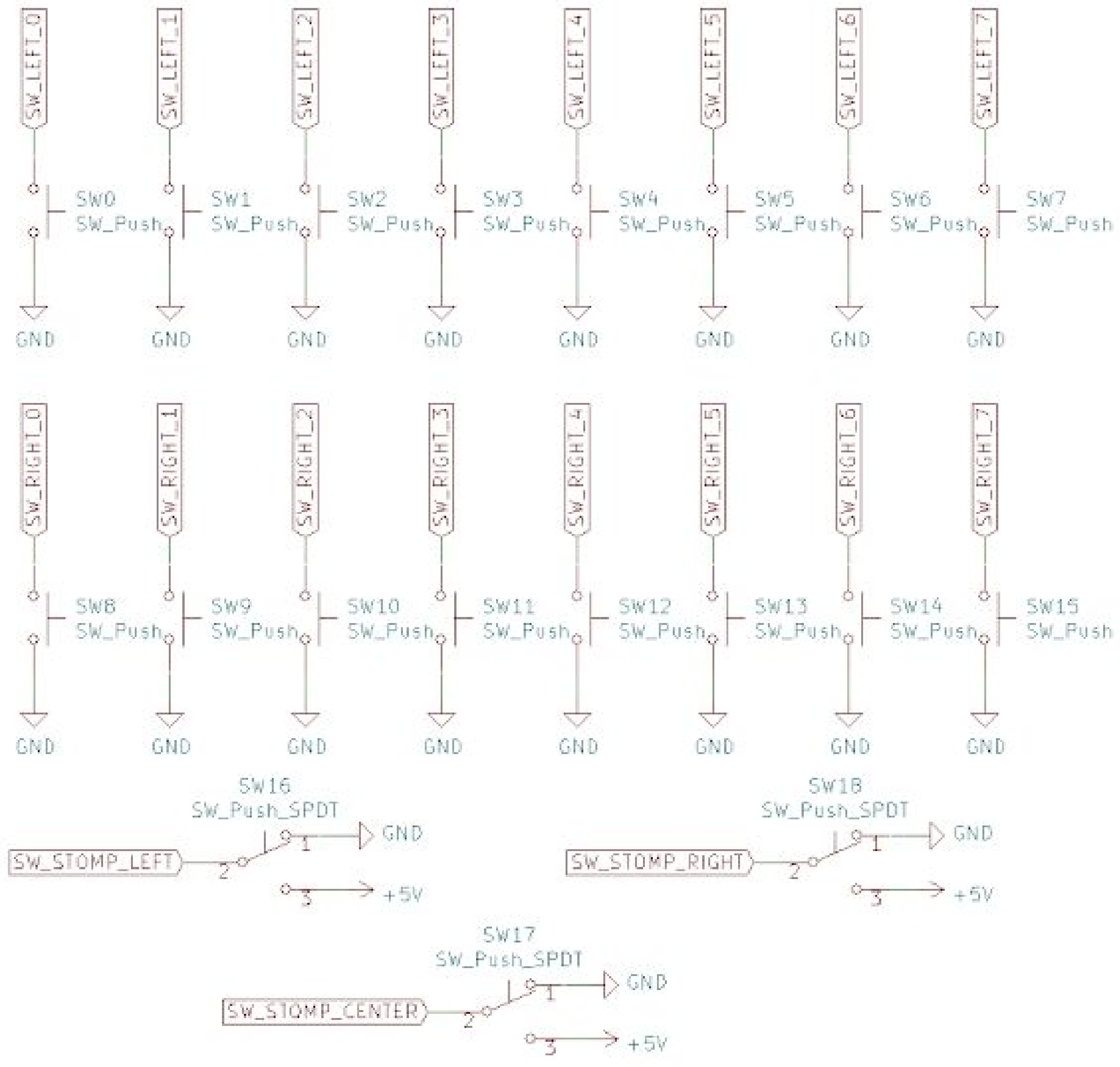

Fig 8 User Interface Schematic: Directional switches will be connected in a pullup configuration, while the SPDT stomp buttons will not.

2.4 Control Unit Subsystem

A microcontroller is required to communicate between the keyboard and the computer. The microcontroller will take and compile the inputs from the user interface to determine which character has been selected. Once selected, the microcontroller must send a signal to the computer via USB behaving as a standard USB keyboard/mouse would. The microcontroller would also receive power from the computer to power the device.

The microcontroller will be programmed to read the two inputs from the left foot grip and right foot grip in order to select the key which is pressed or to determine if the user is moving the mouse and then send the scancode for the pressed key to the computer or update the mouse position. As the switch input is mechanical, the switch data will be compared with cached switch data from the previous iteration to debounce the input. This adds a strict 5ms input delay, but it is necessary to prevent accidentally sending double or triple keystrokes.

To handle the USB communication, we will be using the Lightweight USB for AVR (LUFA) library. This library provides simpler interfaces for setting up and managing common USB connections such as the HID keyboard/mouse connection we will need to maintain with the computer. The library can also set up periodic polling updates using interrupts to keep the polling rate at exactly 100Hz.

For our development environment we will use CrossPack and Xcode for macOS, Ubuntu and AVR dev tools for Linux, and Atmel studio for Windows. To burn the firmware onto the microcontroller we will use Atmel's dfu-programmer.

Requirement	**Verification**
Correctly setup a USB HID connection	1. Plug the device into a windows computer. 2. The keyboard should show up as a Human Interface Device in the Device Manager.
Identify the correct keystroke when both foot controllers have selected groups.	1. Open a web based keyboard switch tester. 2. Confirm all key inputs register using the tool.

Requirement	Verification
Correctly debounce the input switches	1. Open a word processor on the host computer. 2. Test each key input 10 times 3. Each time a key combination is input, only one new character should appear on screen.
Send USB updates at 100Hz	1. Using a web based keyboard scan rate tool, measure the scan rate of the keyboard. 2. With a short enough keystroke, the tool should measure 100Hz.
Mouse input functions correctly	1. Connect the keyboard to a computer. 2. Verify the mouse moves in the correct direction in response to each input direction. 3. Verify left and right clicks function correctly.

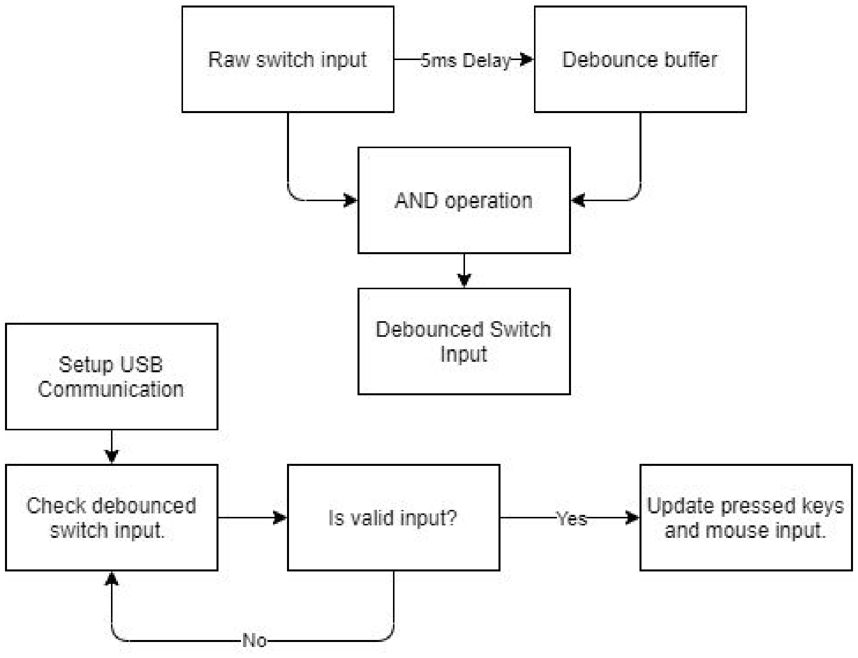

Fig 9 Control Unit Flowchart: Debouncing is implemented with a buffer. Sending and receiving data happens "asynchronously" with interrupts.

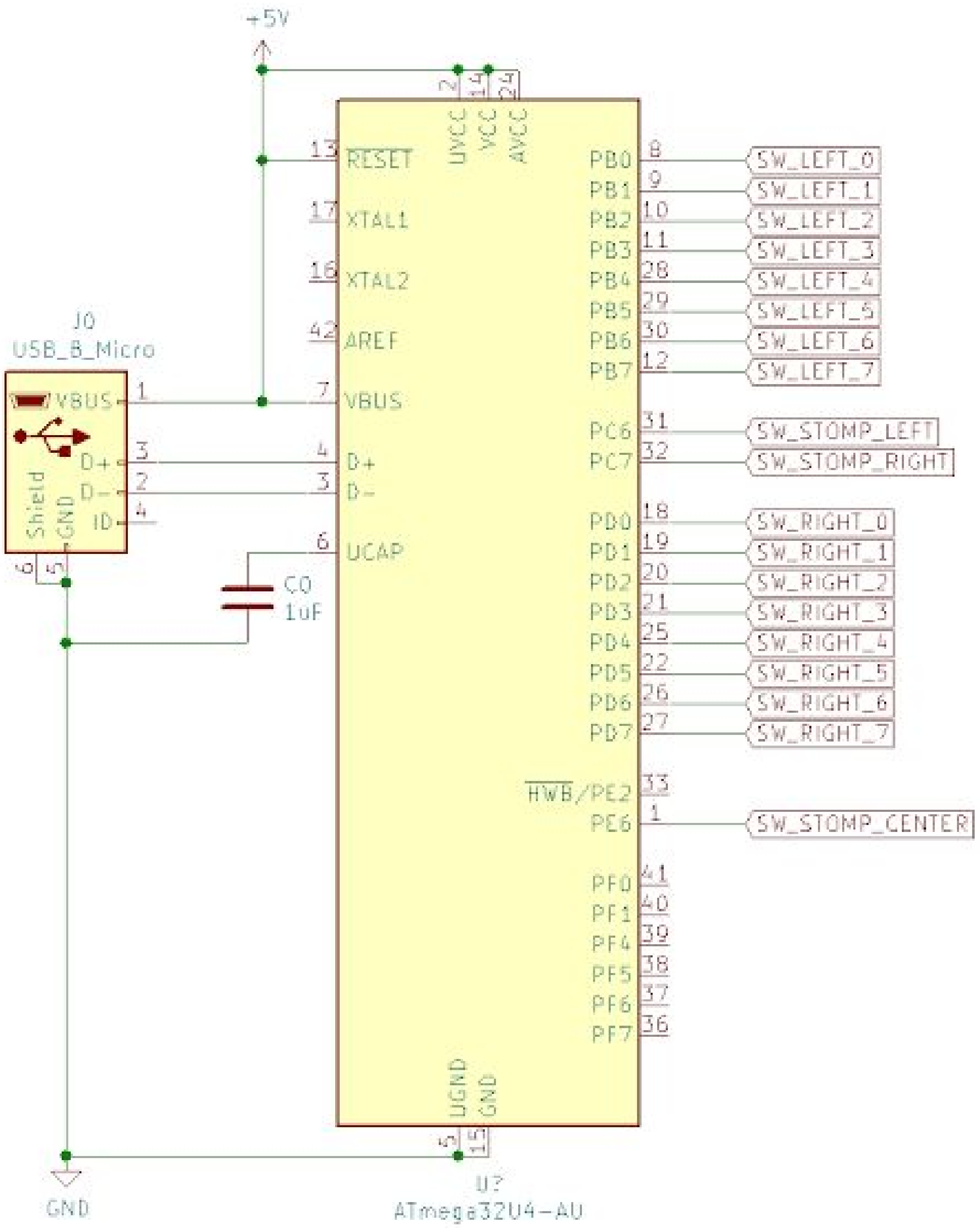

Fig 10 Control Unit Schematic: Power for the system will be supplied over USB.

2.5 Tolerance Analysis

To ensure the device can handle even the fastest typists, our device should be able to handle 216 words per minute, the current world record as set by Stella Pujunas in 1946 [4]. At an average of 5 characters per word, this comes out to about 55.6ms in between each character or 27.8ms between key updates (assuming one key down and one key up update per character). This means our device must, at a minimum, poll at 36Hz. Given the only notable delay in our code will be the 5ms delay for debouncing, we can easily process a key update 5 times faster than the fastest human can trigger them (200Hz). Setting the polling rate at 100Hz (10ms between updates) allows for plenty of time to process the switch inputs and send data over USB while over doubling the rate a human would ever need.

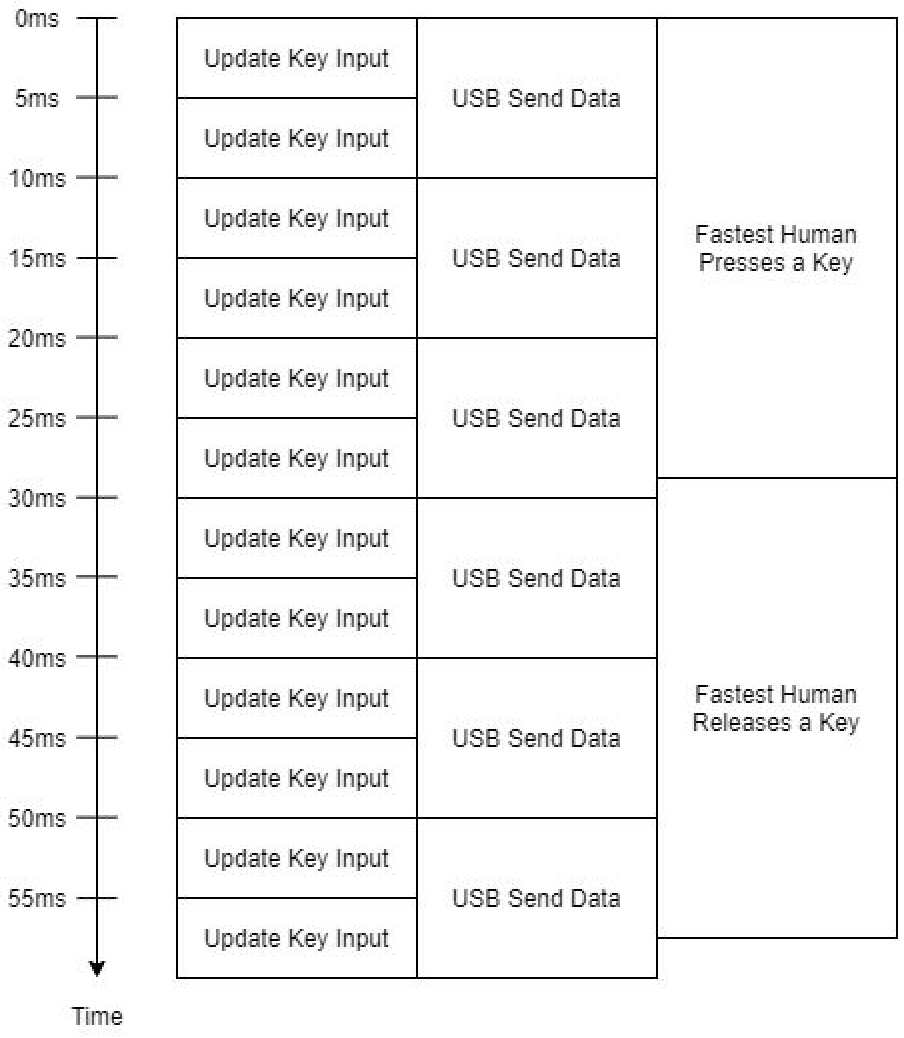

Fig 11 Timing Diagram: The key inputs are processed and sent to the computer much faster than humans have ever been recorded typing.

3. Project Differences

3.1 Overview

At the start of Spring 2020, Project Team 1 designed a prosthetic hand to type on a standard keyboard. The design consisted of a hand prosthetic and a set of four buttons to be pressed by the user's toes. Each of the toe buttons corresponded to the prosthetic's four fingers on the keys so when a button was pressed the corresponding finger would press down on one of the keys. The fingers did not have the capability of shifting up or down a row, so the prosthetic was fixed to pressing only the home row keys (i.e. A, S, D, F). The prosthetic was designed for a user who did not have the use of only one hand, so it is expected that the user typed normally with their other hand to press the remaining keys. The prosthetic and foot controller communicate over Bluetooth, and due to the prosthetic typing on a standard keyboard there is no need for direct communication between the device and the computer. Both the prosthetic and foot controller are powered using rechargeable batteries. The main reason a prosthetic was incorporated into the design was for the user to have the visual feedback of typing with two hands and enable the user to use a familiar keyboard layout.

In our design, we chose to remove the prosthetic component entirely and create a way for the user to do keyboard functions entirely with their feet, turning a two part solution into a one part solution. The original design could not solve the issue of needing to type all the characters associated with the replaced hand, so it would only marginally improve the user's ability to type from just using only one hand. While it does require the user to familiarize themselves with a new keyboard layout, with our design the user can control the computer entirely with their feet. Since it only requires feet input, the device is more accessible to those who do not have the use of either of their hands and/or arms. We also chose to have the entire foot control the input selection which is a much more natural foot motion than pressing buttons with individual toes.

Instead of Bluetooth we chose to use a direct USB connection to communicate with the computer. This provides two benefits as we will experience less latency than Bluetooth as well as we do not require the use of an external battery so the user would never have to deal with the device running out of power during normal operation. The tradeoffs of this design are that USB is more complex to set up in terms of the wires being in the way and plugging in a USB device with one or no hands is more challenging. However, this is mostly a one time issue as after the device is plugged in no additional setup is necessary.

3.2 Analysis

A tradeoff between the original design and the one we propose is the latency of the user selecting a character or function to the character being displayed on the screen. In the original design, after the user presses a button with their toe, an 8-bit signal is sent to the typing prosthetic over Bluetooth and then the prosthetic must move to press the key. This raises the opportunity for latency at two instances, when the signal is being sent over Bluetooth and for the prosthetic to react to the signal that was received. The Bluetooth module in the original design communicates with the microcontroller using USART protocol which has a data transfer rate of 960 bytes per second [5]. The solution we are proposing uses USB 2.0 to communicate with the computer which is reported to have a 480 Mbps data transfer rate [6]. Having a much faster data transfer rate ensures that we are less likely to experience significant latency. Bluetooth can also face interference from other wireless signals such as Wifi or other Bluetooth devices due to all operating at a 2.4 GHz frequency further increasing potential sources of latency [7].

Additionally, the original design is a battery powered device which according to the design documentation, requires three to five hours to reach full charge and has a three hour battery life [5]. This is a long time the user has to wait between uses especially considering the user is only expected to type at least 25 words per minute which is much slower than the average typing speed of 41 words per minute [4]. Their slower expected typing speed makes it so the user cannot do as much during those three hours.

4. Cost and Schedule

4.1 Cost Analysis

4.1.1 Parts

Item	Part # or Manufacturer	Count	Cost
ATMEGA32u4 Microcontroller	ATMEGA32U4-AU	1	$4.00
FS5700 Series Pushbutton Switch (“Stomp button”)	FS5700SPMT2B2M2QE	3	$15.45
Cherry MX Switch	MX1A-C1NW	16	$16.96
Micro USB Female Port	Molex	1	$0.78
USB-A to Micro USB Cable	Anker	1	$7.99
PCB	PCBWay	1	N/A
1uF Capacitor	CL10A105KP8NNNC	1	$0.10
Total Cost	**$45.28**		

4.1.2 Labor

From the ECE Illinois website, the average starting salary for a student graduating with a degree in computer engineering is $84,250 [8]. If working 52 weeks a year for 40 hours a week that salary is equivalent to earning $40.50/hour. We estimate our work period to be seven weeks with an estimated work week of 15 hours per week. For three people this would lead to a total cost of:

3 people x $40.50/hour x 15 hours/week x 7 weeks x 2.5 = **$31,893.75**.

4.1.3 Total Cost

$45.28 (Parts) + $31,893.75 (Labor) = **$31,939.03**

4.2 Schedule

Week	Neva	Aditi	Nick
1	Complete the design document		
2	Design document revisions and prepare for the design review		
	Order parts and talk to the machine shop regarding the physical design	Research on how to read the right keys with our design	Research USB communication
3	Design the PCB	Research on Keyboard's firmware	Design the PCB
4	Order the PCB	Write code to read and map inputs from the switches	Write code to handle USB communication with the computer
	Unit test all components		
5	Test the user interface	Test the control unit	Testing on an online Keyboard switch tester
6	Assembling parts in the physical design		Soldering the PCB
7	Prepare for demo and final report		

5. Ethics and Safety

There are a few safety hazards that must be taken into consideration with our product. As the device will be out of sight of the user, it must not have any sharp edges or otherwise dangerous surfaces that the user could unknowingly injure themselves on. The user should be able to apply significant force to any part of the frame without it harming them. As an additional precaution, the frame should not be made of a material that could break or shatter under significant weight. Because we have moving parts, we must ensure the foot pads and stomp buttons cannot pinch the user's feet at any point in their range of motion. Thus for the safety of the users our project must comply with IEEE Code of Ethics #9 [9]:

"*to avoid injuring others, their property, reputation, or employment by false or malicious action;*"

Since our product caters to the people who are unable to use the keyboard with their hands, we must be realistic in stating claims about the features and success of the product, in accordance with IEEE Code of Ethics #3 [9]:

"to be honest and realistic in stating claims or estimates based on available data".

We will make sure we vigorously test our product to get a better accuracy of the success of our product.

Our product does require firmware to interface with the computer and thus we comply by the ACM Code of Conduct #2.8 [10]:

"computing professionals should not access another's computer system, software, or data without a reasonable belief that such an action would be authorized or a compelling belief that it is consistent with the public good."

We will ensure our firmware does not gather any data from the computer and only operates as we advertise, to be an alternate keyboard input device.

For the success of this product we will consider all the constructive criticism and suggestion on improving the performance which adheres to the IEEE Code of Ethics #7 [9]:

"to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others".

6. Citations and References

[1] Amputee Coalition, “Limb Loss Statistics”, *The Amputee Coalition*, 2015. [Online] Available: https://www.amputee-coalition.org/resources/limb-loss-statistics/. [Accessed: 03-Apr-2020]

[2] United Nations, “Universal Declaration of Human Rights”, *United Nations*, 2020. [Online] Available: https://www.un.org/en/universal-declaration-human-rights/index.html. [Accessed: 03-Apr-2020]

[3] K. Kuligowska, P. Kisielewicz, A. Włodarz, “Speech Synthesis Systems: Disadvantages and Limitations”, *International Journal of Engineering and Technology,* 2018. [Online] Available: https://www.researchgate.net/publication/325554736_Speech_synthesis_systems_Disadvantages_and_limitations. [Accessed: 03-Apr-2020]

[4] Ratatype, “Average Typing Speed Infographic”, *Ratatype*, 2020. [Online] Available: https://www.ratatype.com/learn/average-typing-speed/ [Accessed: 17-Apr-2020]

[5] R. Krishnan, R. Hatalkar, V. Marcinkevicius, “ECE 445 Design Document”, *ECE445*. [Online] Available: https://courses.engr.illinois.edu/ece445/getfile.asp?id=16825 [Accessed: 17-Apr-2020]

[6] SONY, “What are the USB data transfer rates and specifications?”, *SONY*, 2019. [Online]. Available: https://www.sony.com/electronics/support/articles/00024571 [Accessed: 16-Apr-2020]

[7] ITeadStudio, “HC-05 -Bluetooth to Serial Port Module”, *ITeadStudio*, 2010. [Online] Available: https://www.electronicaestudio.com/docs/istd016A.pdf [Accessed: 17-Apr-2020]

[8] ECE Illinois, “Salary Averages”, *ECE Illinois*, 2014. [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages.asp. [Accessed: 27-Feb-2020].

[9] IEEE, "IEEE Code of Ethics", 2020. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html/. [Accessed: 02-Apr-2020].

[10] ACM, “ACM Code of Ethics and Professional Conduct”, 2020. [Online]. Available: https://www.acm.org/code-of-ethics [Accessed: 16-Apr-2020].