# Bluetooth Enabled Gloves for Controlling Music

Electrical & Computer Engineering

5/3/2023

## Controlling Phone with Traditional Gloves

- Inconvenient
- Time Consuming

**Problems with Existing Solution:**

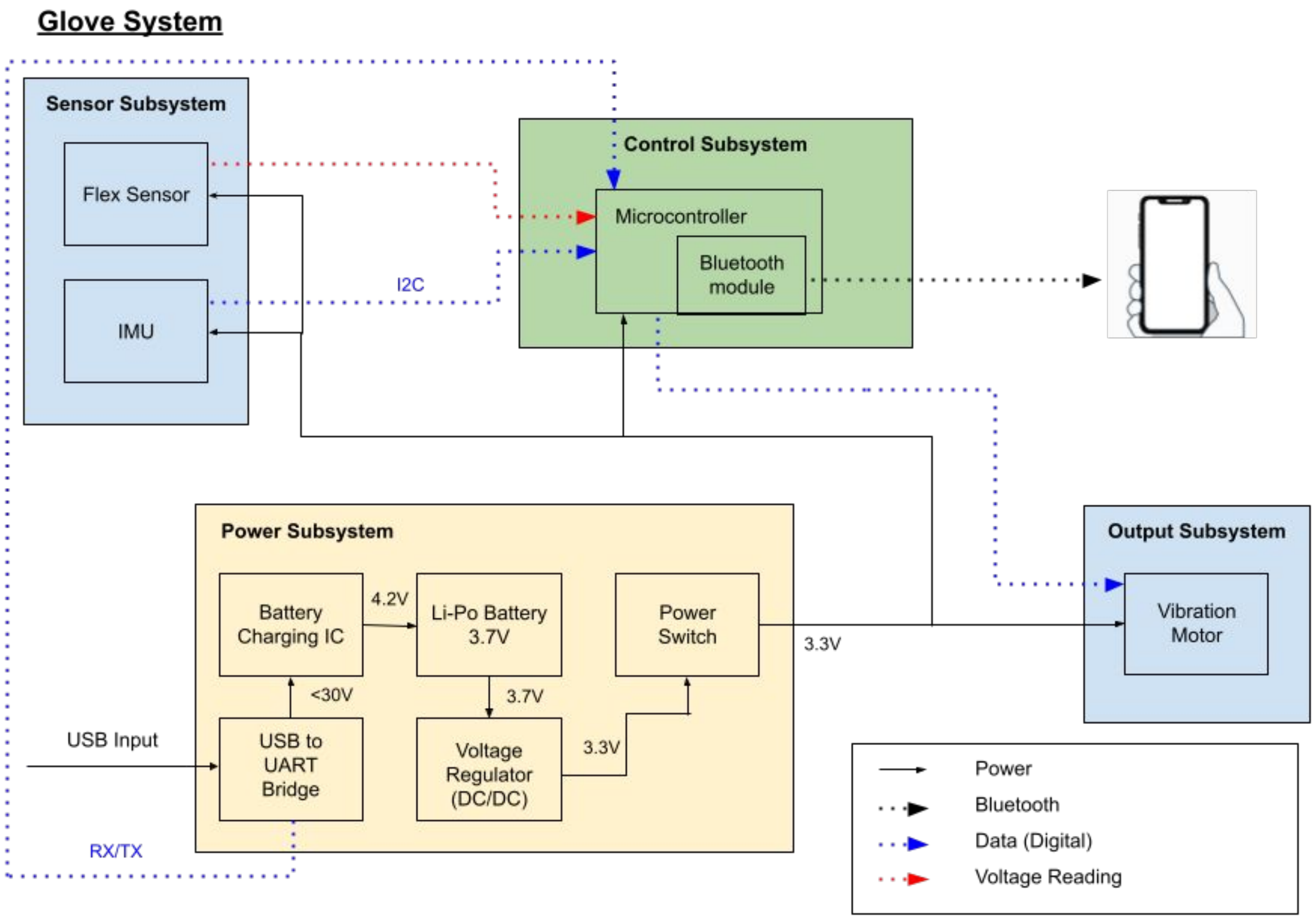
- Touch-sensitive fingertip gloves
  - Inconsistent
  - Difficult to access phone in pocket

## Bluetooth Enabled Gloves:

- Convenient
- Fast
- Consistent
- No need to hassle with phone

## Setup

- Switch on in calibration orientation to enter bluetooth pairing mode
- Connect to phone via bluetooth
- Play/Pause can be performed in any orientation
- Need to get into calibration orientation to send swipe commands
- Haptic feedback with each command

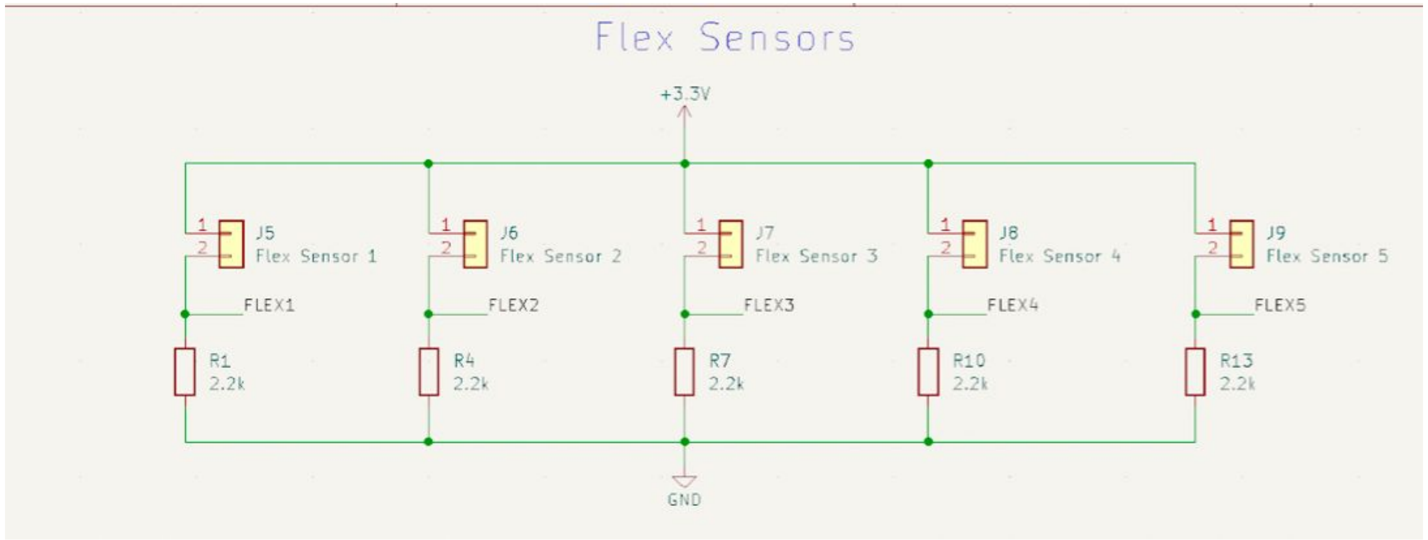



## High Level Requirements

- Correctly identify predetermined gestures from sensor readings
- Correctly convert predetermined gestures into correlated bluetooth commands
- Glove has a small form factor and battery lasts at least 3 hours

# Sensor Subsystem

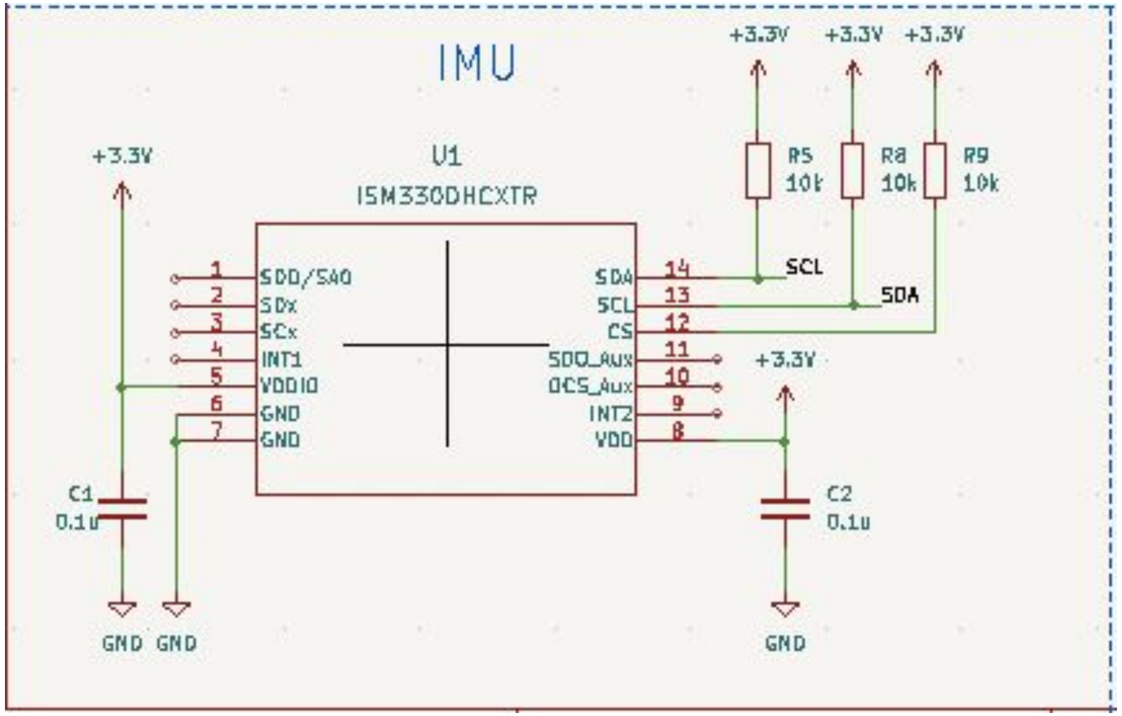
# Flex Sensors




- Mounted on gloves to move with user’s fingers
- Resistance profile changes with bend angle
- Create voltage divider circuit to measure the degree of bend

# IMU (Accelerometer and Gyroscope)

- Use to detect orientation and hand motion
- Gyroscope
  - Gives angular velocity
- Accelerometer
  - Gives direction and magnitude of acceleration
- Interface using I2C protocol



# Changes Based On Challenges


## Initial Problem

- IMU data was difficult to interpret

## Solution

- Added a calibration sequence in order to establish a zero position
    - Swipe commands are based on this zero position
- Even after calibration, gyroscope data was difficult to use
    - Slight movements produced large unexpected changes
    - Decided to only use accelerometer and take advantage of gravity

# Sensor Requirements and Verifications


*5 Trials of flexing and unflexing fingers. Peaks refer to flexed ADC value, and unflexed refers to 0 ADC value.*

## Requirement

We need a large binary change between unflexed and flexed positions.

## Verification

1. Connect the flex sensors to the microcontroller using a voltage divider circuit as specified in the circuit schematic
2. Ensure the flex sensors are attached to the fingers and are unflexed (fingers are straight)
3. Measure the raw ADC reading on the microcontroller using the serial monitor with fingers unflexed
4. Flex all fingers into a fist
5. Measure the raw ADC reading on the microcontroller using the serial monitor
6. Ensure the difference is greater than 150.
7. Repeat steps 2-4 for 5 trials

# Sensor Requirements and Verifications

| Trial | x (step 4) | y (step 3) | z (step 2) |
|---|---|---|---|
| 1 | 1.00 g | .98 g | 1.01 g |
| 2 | 1.00 g | .98 g | 1.01 g |
| 3 | 1.00 g | .98 g | 1.01 g |
| 4 | 1.00 g | .98 g | 1.01 g |
| 5 | 1.00 g | .98 g | 1.01 g |
| Mean Error | 0% | 0% | 0% |

*Mean Error of IMU Acceleration Data Across 5 trials for X,Y, and Z axis*

## Requirement

The accelerometer readings from the IMU sensor should be accurate to within ±5% when measured under the same motion.

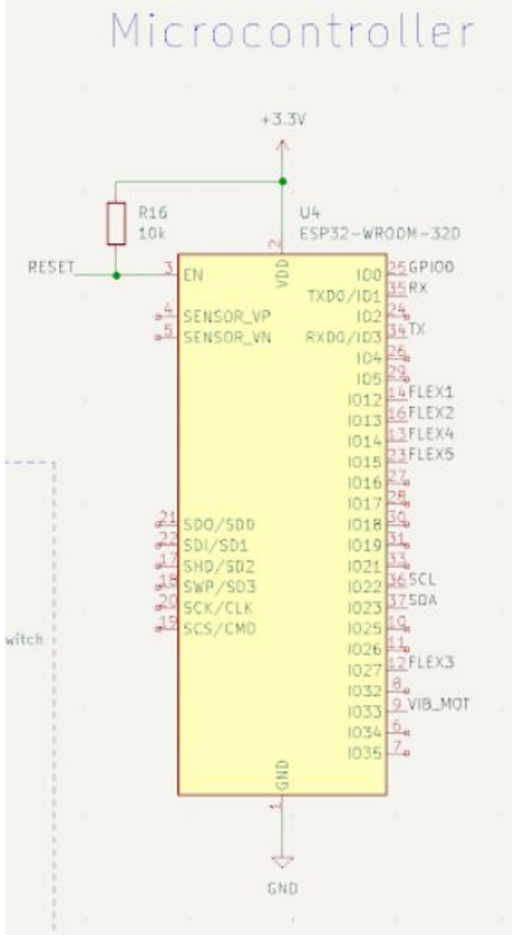
## Verification

1. Connect the IMU to the microcontroller and establish the $I^2C$ communication between the two devices
2. Measure z acceleration with IMU flat on a desk with the z-axis facing up (should be around 1g)
3. Measure y acceleration with IMU flat on a desk with the y-axis facing up (should be around 1g)
4. Measure x acceleration with IMU flat on a desk with the x-axis facing up (should be around 1g)
5. Repeat steps 2-4 for another 4 trials
6. Calculate mean error for steps 2-4 (should be less than 5%)

# Control Subsystem

## ESP32-WROOM-32E

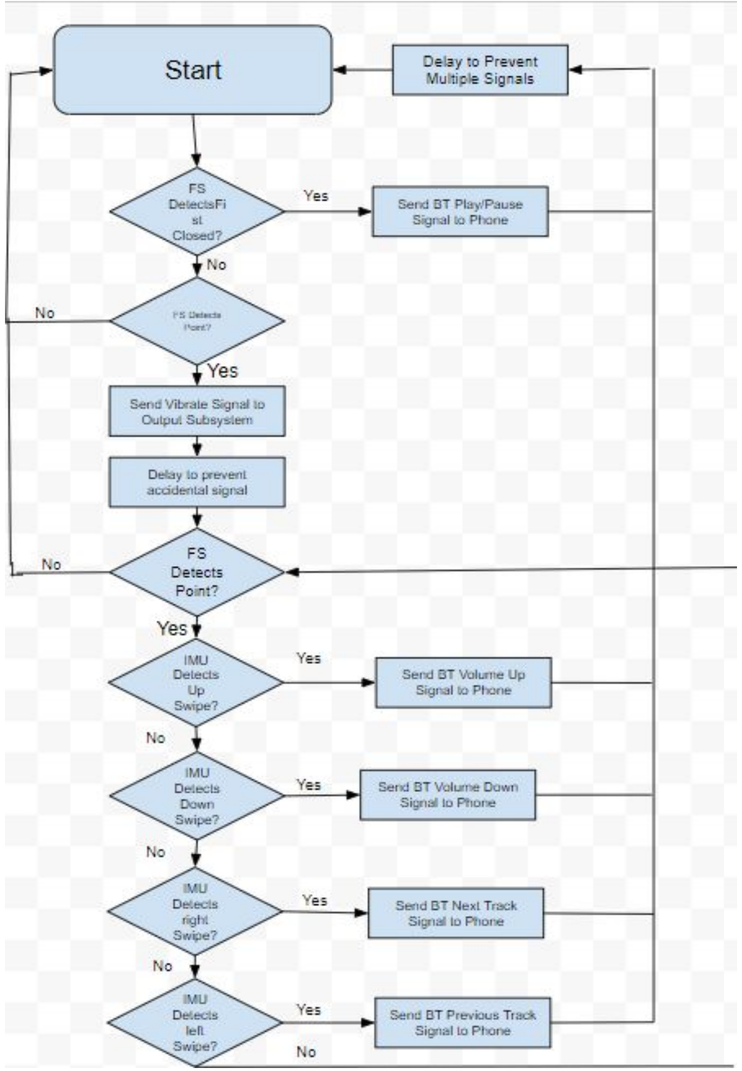
- ADC pins for flex sensor voltage readings
- Built-in bluetooth antenna module
- $I^2C$ serial communication protocol capability
- Flashable using USB to UART bridge
    - Use RS-232 signals to interface with Reset and GPIO0 buttons
- Reset button power cycles the microcontroller
- GPIO0 must be low to enter boot mode



# Initial Design

## Software Flowchart

- Calibrate in two finger point orientation
- Play/Pause
  - Flex all fingers
- Next/Previous Track
  - Right and left swipe from home orientation in two finger point orientation
- Up/Down Volume
  - Up and down swipe in two finger point orientation
  - Release bent fingers to stop volume command




Microcontroller Flow Chart

## Thresholding for accelerometer data

- Difficulty tuning acceleration threshold values for distinguishing commands.
    - Horizontal swipes when slightly angled diagonally were triggering vertical swipes
- Through trial and error, thresholds were tuned to find balance between commands

# Changes Based On Challenges

## Volume control

- Initial Plan:
  - Keep moving the glove in same direction to change volume.
    - Requires integrating acceleration to check if velocity is constant
    - Led to erratic data
- Final Iteration:
  - Use only acceleration thresholding and 2 finger point to trigger command and release 2 flexed fingers to terminate
    - Requires no math
    - Easy for user to do
    - Works consistently


Voltage at Output of Vibration Motor GPIO pin

2.292V / 470Ω = 4.8766 mA
4.8766 mA > 2 mA as specified in verifications

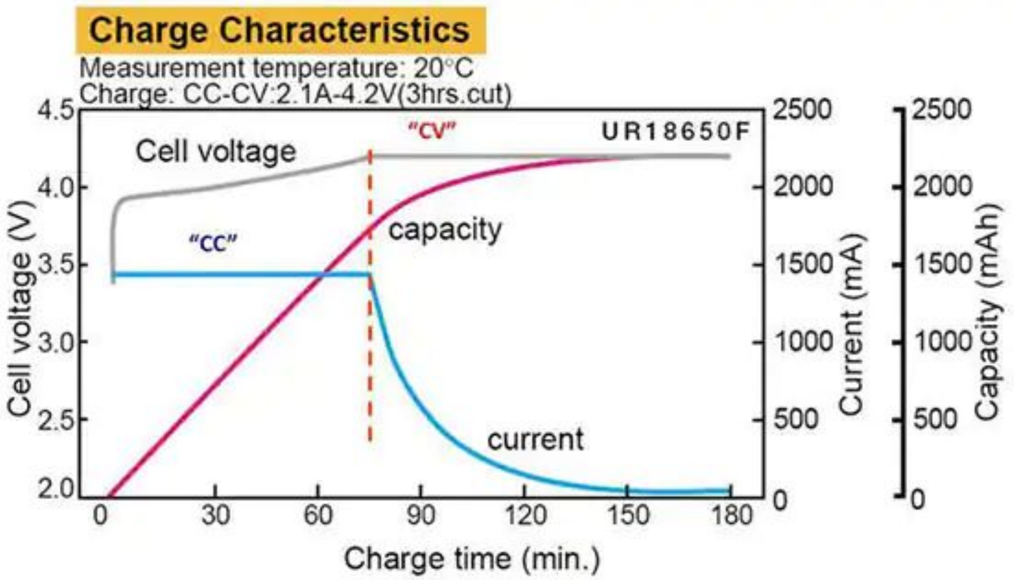
## Requirement

At least 2 mA of current must be provided from a GPIO pin in order to switch on the BJT transistor and drive the vibration motor.

## Verification

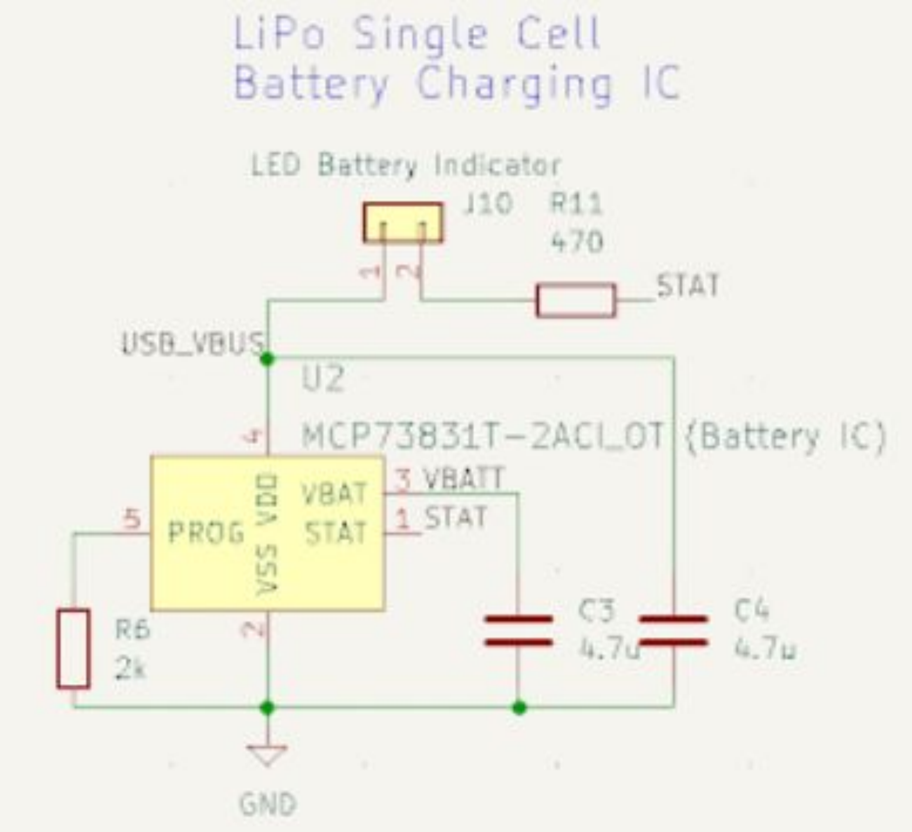
1. Connect a DC 3.3V power supply to the VDD pin of the microcontroller
2. Connect pin GPIO21 on the microcontroller to the base node of the BJT transistor in the vibration motor circuit
3. Create the Vibration motor circuit
4. Program the pin GPIO21 to be an output pin sending out a digital ‘1’.
5. Using a voltmeter, measure the voltage drop across the resistor R12
6. This measured voltage divided by 470 ohms should produce a value greater than 2 mA.

# Power Subsystem




## Charging Circuit

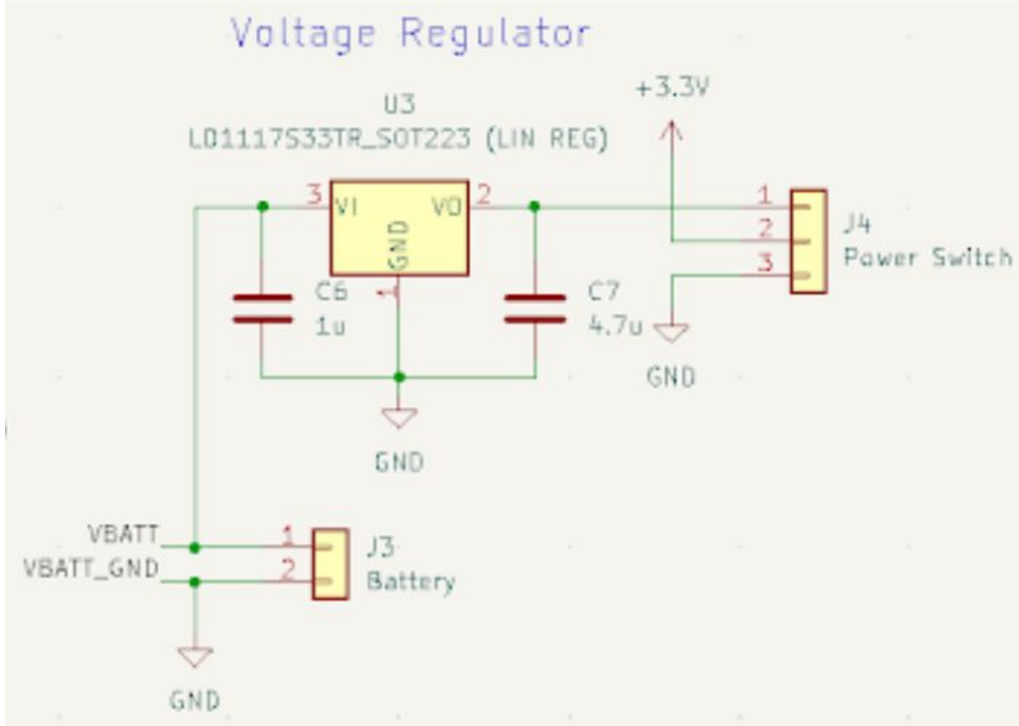
- Need to provide a constant current source to charge the battery to 4.2V
- Once charged, a constant voltage source is needed to keep voltage at 4.2V




## Battery IC

- Provides constant voltage source of 4.2V
- Provides variable constant current set with resistor R6
  - Set this to 500 mA
- Ensures safe fast charge current
  - < 0.5*C mA
  - C = 3000 mA
  - Max charge current = 1500 mA

# Initial Design

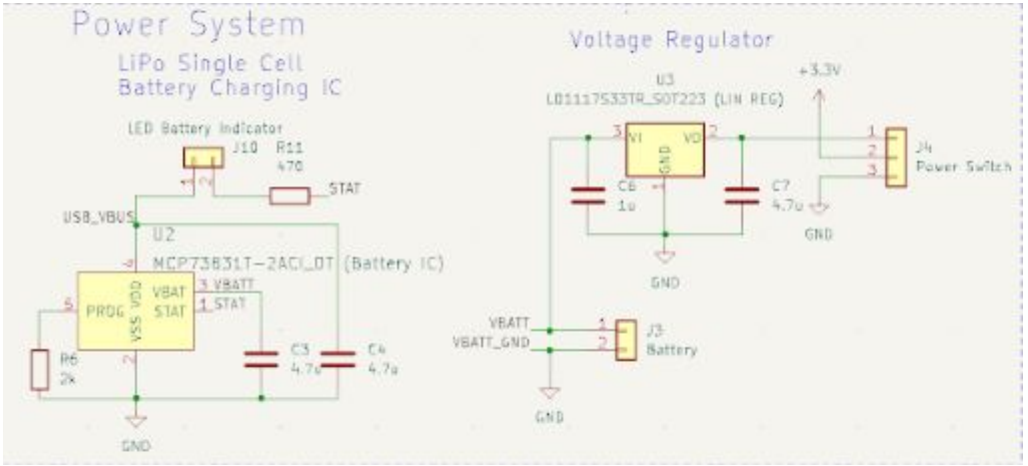



## Voltage Regulator

- Takes voltage input from battery ranging ~[3.7V - 4.2V]
- Steps it down to 3.3 V to properly power the three other subsystems

## Changes / Iterations

- Initial regulator had too low of a current limit
  - Max output rating was 800 mA
    - Too close to current draw of 796.5 mA
- Realized the minimum drop out voltage was 1V
  - 3.7V - 3.3V = 0.4V drop out
  - Need something with a lower dropout
- Found a new regulator that supplies 1A max current and very low dropout of 0.35V



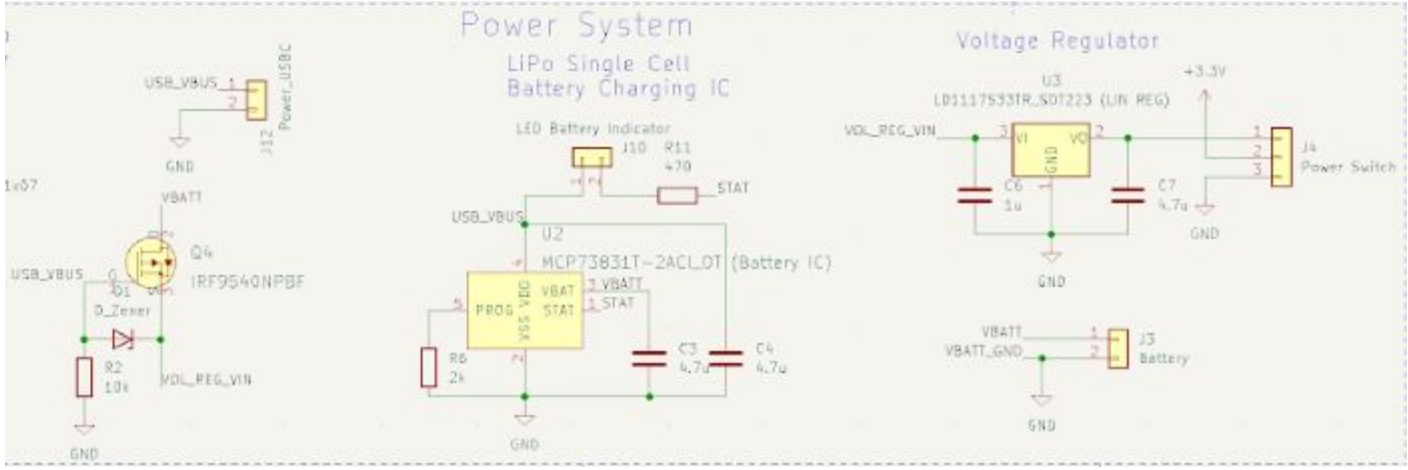

Without Switching Circuit

# Switching Circuit Problem

- Initially, if the USB was plugged in, the battery was being charged and trying to discharge at the same time

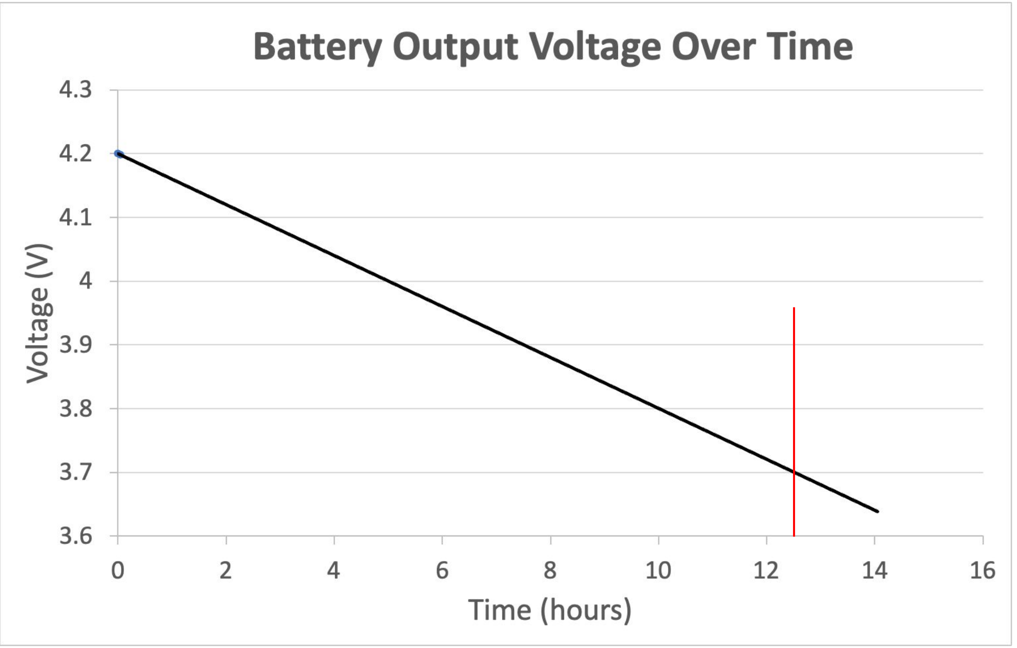
# Solution

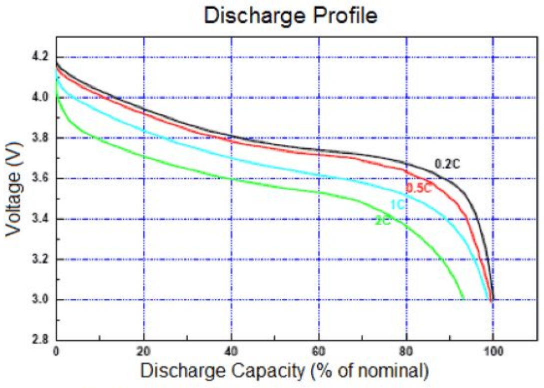
- Add P-MOS transistor to switch input of voltage regulator




With Switching Circuit

# Power Requirements and Verifications




Discharge: 3.0V cutoff at room temperature.

Initial Battery Voltage: 3.8226 V

After 3 minutes with 33% active use: 3.8206 V

4.2V - 3.7V = 0.5V
0.002V drop in 3 minutes
Thus, 0.5V drop will occur in approximately 12.5 hours

## Requirement

The battery must have a capacity of at least 2.39 Ah (in order to have a battery life of at least 3 hours).

## Verification

1. Build the entire circuit with the battery attached
2. Connect the terminals of the battery to a voltmeter and measure the output voltage and ensure it is above 3.7V.
3. Turn on the power switch.
4. Run the glove for 3 minutes, providing active gestures for 33% of the time.
5. Turn off the power switch.
6. Measure the change in the output voltage of the battery.
7. Extrapolate this data to see how long it will take for battery voltage to drain to 3.7V. Ensure this number is above 3 hours.

# Power Requirements and Verifications


*When running at full power, the current is maxed out at 893 mA. This fits within the component specifications*

22/6 ohms = 3.67 ohms resistor network
3.27V voltage drop

## Requirement

Total maximum current draw for all components is 796.5 mA. Thus, the output of our voltage regulator must supply a current of at least 800mA to the rest of the circuit.

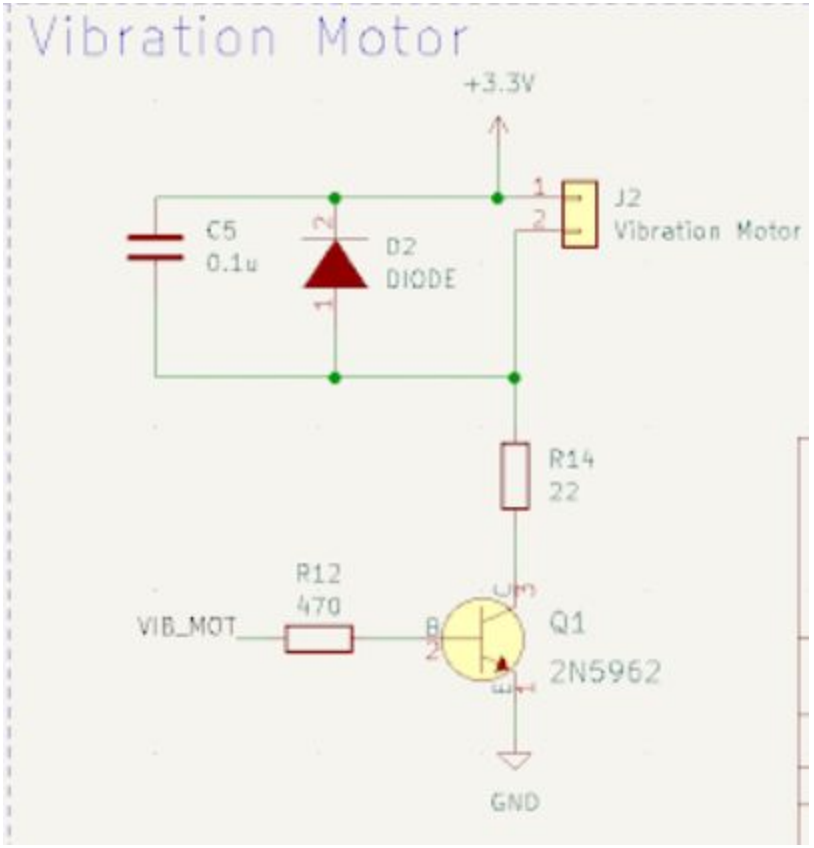
## Verification

1. Create the power subsystem circuit.
2. Hook up a resistor network (of approximately 4 ohms) to the output of the voltage regulator.
3. Probe the voltage across resistor network using a voltmeter.
4. Calculate the current flowing through the resistor network.
5. If we are able to draw more than 800 mA with no issues, this requirement is verified.

# Output Subsystem

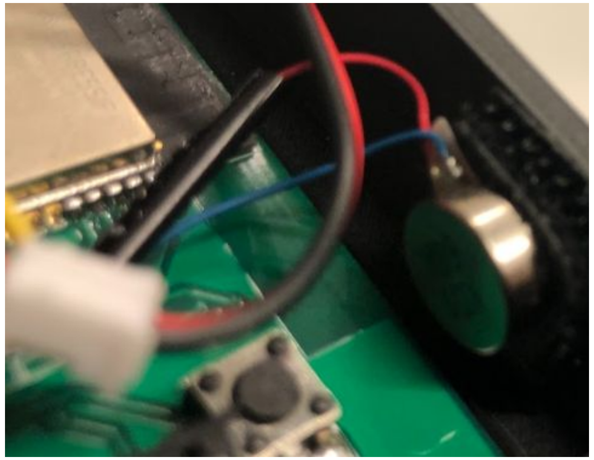
## Vibration Motor

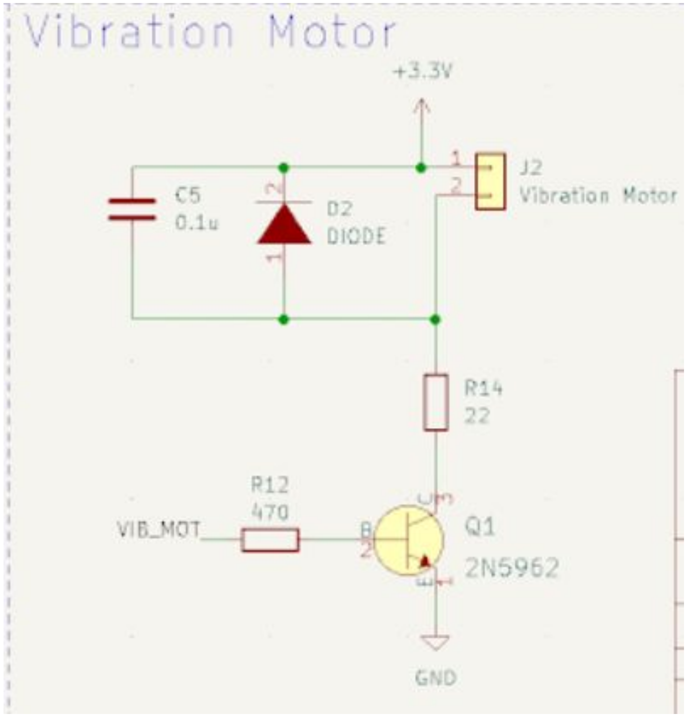
- Microcontroller cannot provide enough current to drive the motor
  - Need to include a BJT transistor to amplify Base current
- Microcontroller sends high or low signals to turn the vibration motor on or off



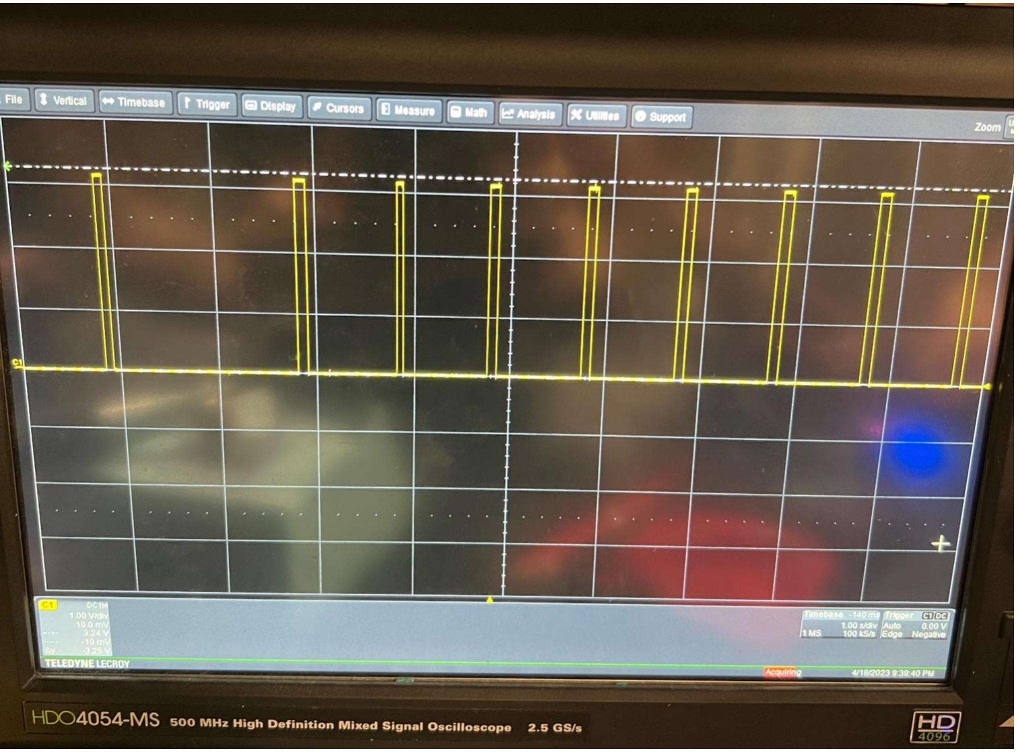
## Vibration Motor

- No major problems
- Wires connected to motor are small and fragile
  - Electrical tape was used to protect wire
- Needed to tune resistor values to provide desired current flow through the motor





# Output Requirements and Verifications




*Vibration Motor Circuit Output*

## Requirement

Vibration motor vibrates when a digital high is provided from the microcontroller and the vibration motor turns off when a digital low is provided from the microcontroller.

## Verification

1. Connect a DC 3.3V power supply to the VDD pin of the microcontroller.
2. Connect pin GPIO21 on the microcontroller to the base node of the BJT transistor in the vibration motor circuit (specified in the figure 13). Create the Vibration motor circuit from figure 13.
3. Connect an oscilloscope to pin GPIO21 of the microcontroller and to ground.
4. Program the microcontroller to send a digital high to pin GPIO21. Ensure that the voltage reading is 3.3V ± 0.1V. Inspect the vibration motor to check if it is vibrating.
5. Program the microcontroller to send a digital low to pin GPIO21. Ensure that the voltage reading is 0V ± 0.1V. Inspect the vibration motor and ensure it is not vibrating.

# Physical Design

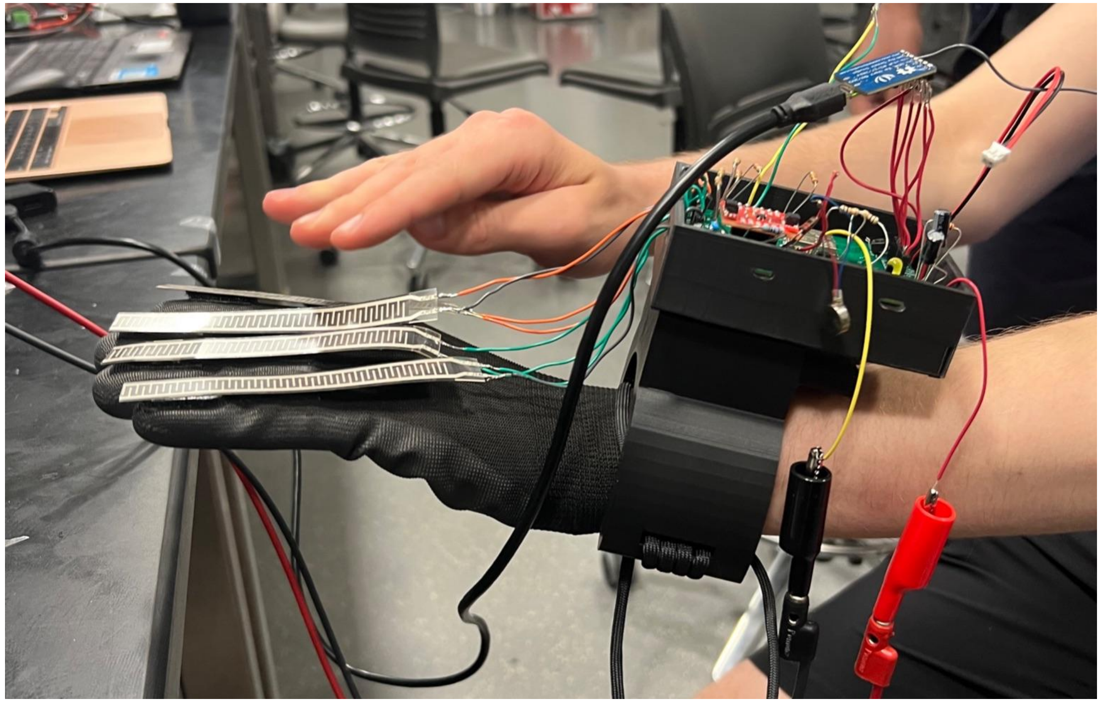
# Initial Design

## Glove

- Mount Flex Sensors directly on fingers using adhesive tape and velcro strips

## PCB Enclosure

- Mount enclosure behind hand on wrist
- Make separate wrist mount and enclosure and mount using paracord
- Feed flex sensor wires into pcb enclosure through 5 holes



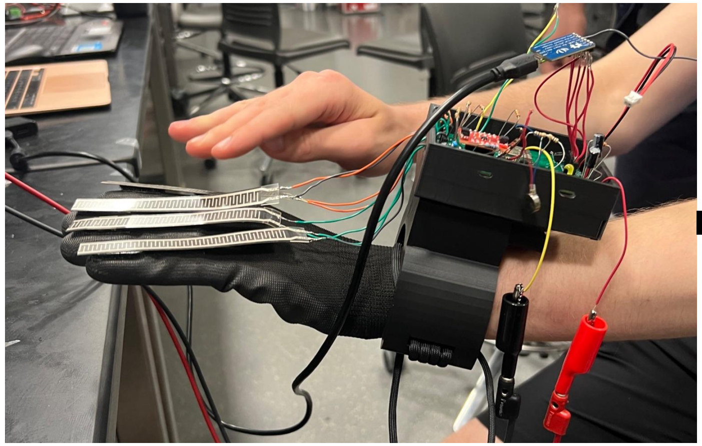
# Final Design

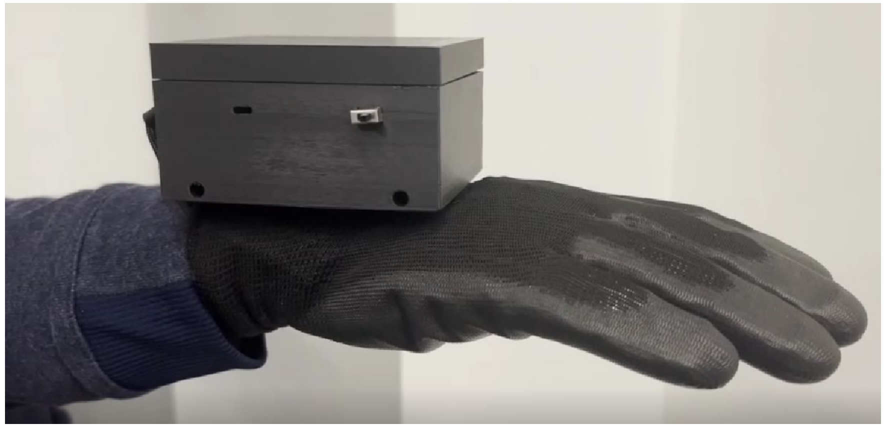
## Glove

- Flex sensor must be mounted properly to ensure accurate readings
- Sandwich flex sensor between two gloves
    - Flex sensors are properly enclosed and discrete
- Glove is extremely easy to put on and take off

## PCB Enclosure

- Designed and 3D printed the enclosure
- Mount enclosure behind hand on top of hand to reduce footprint of design
- Mount enclosure using adhesive instead of paracord for sleeker look





# Success

## Met or Exceeded All Requirements

# Safety and Ethical Considerations

## Safety

- No exposed wires or electronics
- No skin-electronics contact
- Haptic feedback and delays are in place to prevent accidental signals that could disorient the user

## Ethics

- Glove does not connect to the internet or collect personal data
- Bluetooth connection must be accepted on both ends

# Conclusion

## What we learned

- End to end design of a project from initial idea phase to a final product design
    - Learned testing, root-cause analysis, and verification of product using data
    - Teamwork

## Further Work

- Add battery life indicator
- Make PCB and encasing smaller

# Questions?

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN