UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Habit Forming Toothbrush Stand

Team 40

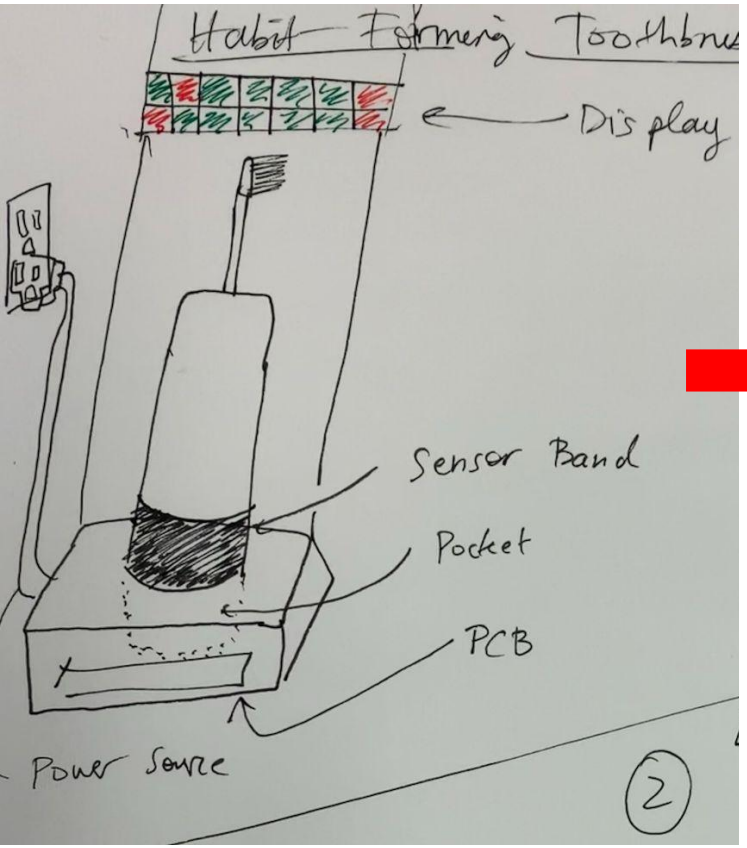
Quinn Palanca, Rahul Vasanth, and John Kim

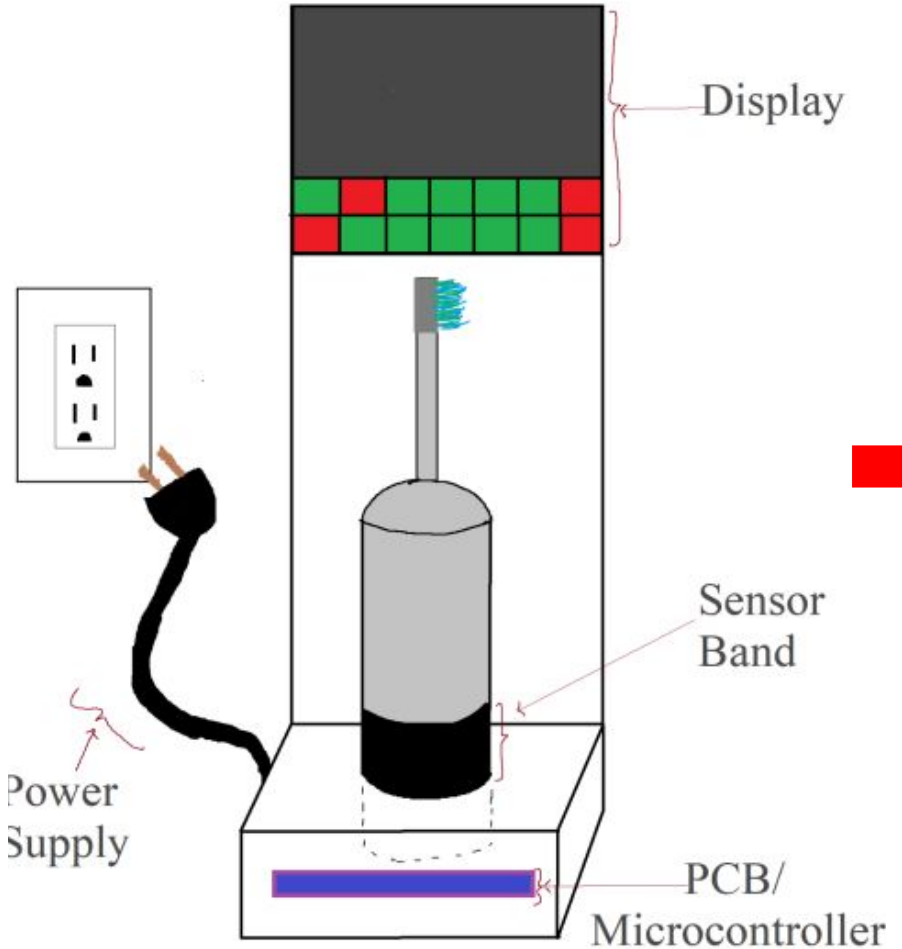
ECE 445 - Spring 2022

TA: Shivang Charan

Introduction

Initial Mockups





Solution

- ➔ **Habit Forming Toothbrush Stand that displays a brushing scorecard for the morning and evening over the course of the week.**
- ➔ **Can turn a basic or electric toothbrush into a “smart toothbrush”**
- ➔ **Can be utilized by a variety of target audiences**

High-Level Requirements

1. The Habit-forming Toothbrush Stand will start tracking time within 2 seconds after brushing activity has started, stop tracking within 1 second after brushing activity has ceased, and determine whether a user is underbrushing (less than 2 minutes), over-brushing (more than 4 minutes), or brushing sufficiently (2 to 4 minutes).
2. The display on the toothbrush stand shows, in the specified format, whether the user brushed their teeth in the morning and in the evening for the past 7 days.
3. The Habit-forming Toothbrush Stand can determine if the user is actively brushing their teeth with 95% accuracy.

Objectives

➔ Practical

- Easy to Use
- Battery Powered

➔ Non-Intrusive

- Personal Data is not stored longer than a week

➔ Fast

- Accelerometer Data polled instantly
- Classifier predicts within 300 ms

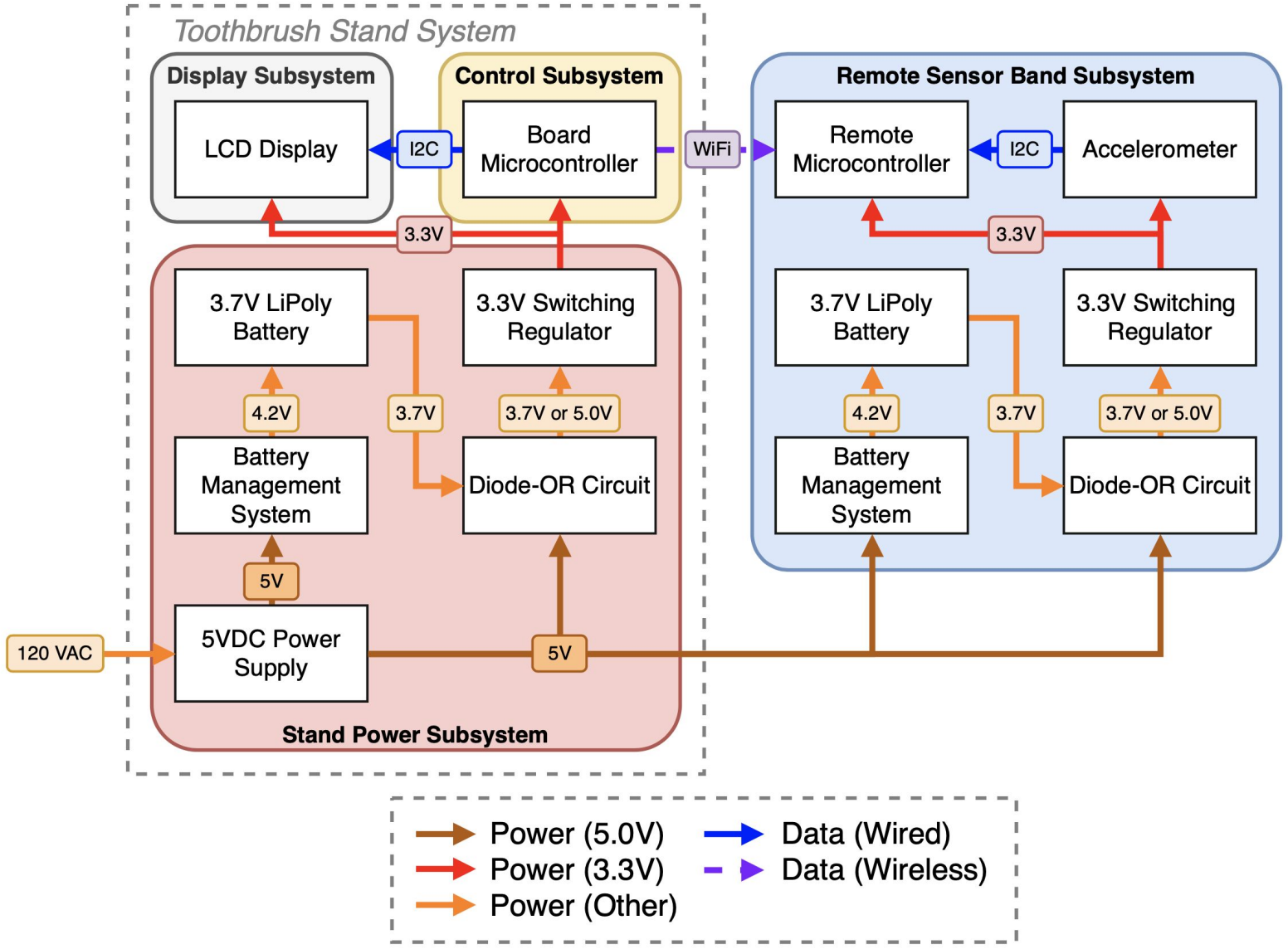
➔ Accurate

- 97% accuracy during prediction
- Further Training

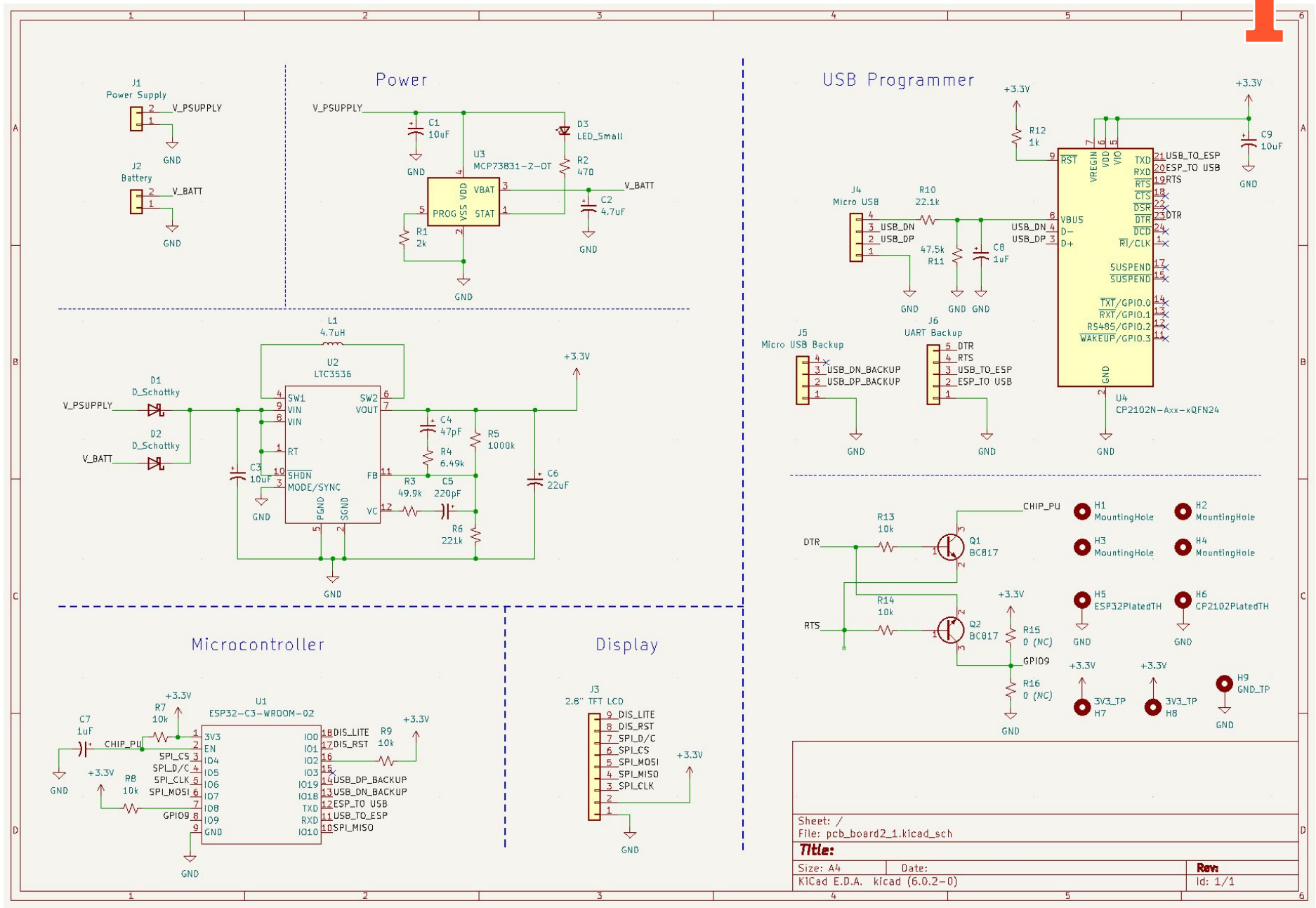
Design

Block Diagram

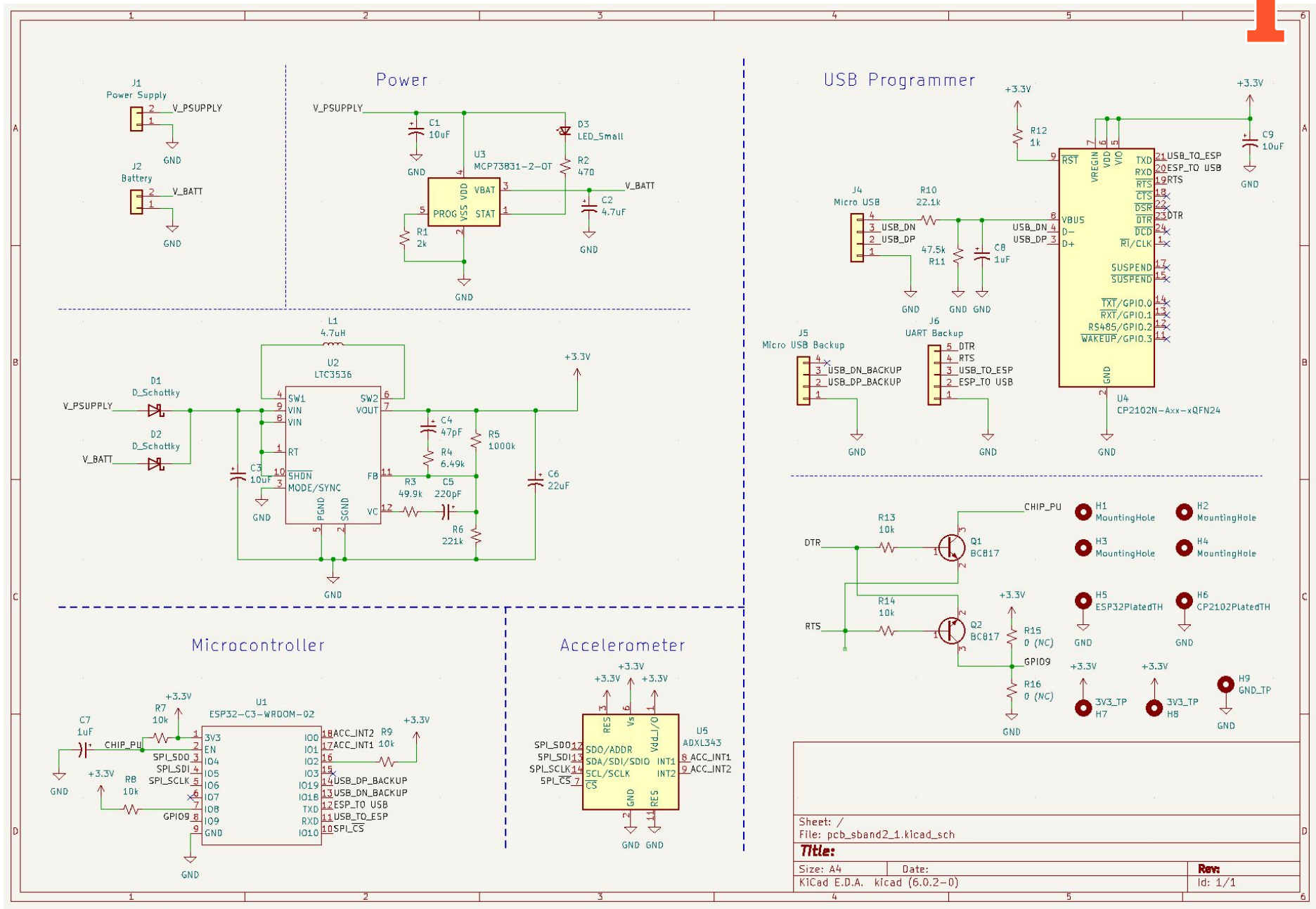
- ➔ Stand Power Subsystem
  - ◆ 3.7V Li-Poly Battery (2500 mAh)
  - ◆ 5V Power Supply
  - ◆ 3.3V Switching Regulator
- ➔ Display Subsystem
  - ◆ 2.8” TFT LCD Display
- ➔ Control Subsystem
  - ◆ ESP32-C3-WROOM-02
- ➔ Remote Sensor Band Subsystem
  - ◆ ESP32-C3-WROOM-02
  - ◆ ADXL343 3-axis Accelerometer
  - ◆ 3.7V Li-Poly Battery (850 mAh)
  - ◆ 3.3V Switching Regulator



Stand Schematic

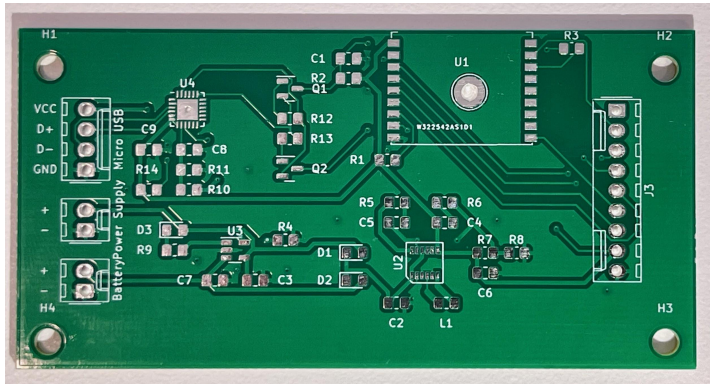


Sensor Band Schematic



PCB Development: Order 1/3

Stand



86x43 (mm)

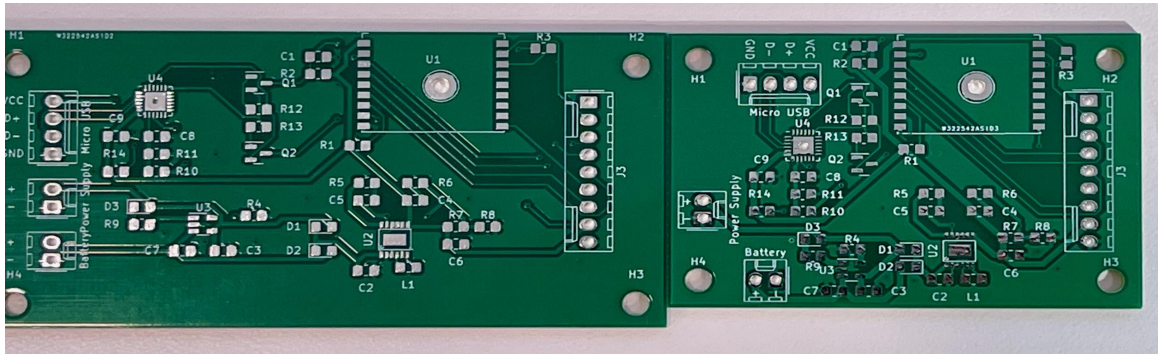
Sensor Band

(Did not make an order)

- ➔ Programming IC has unconnected pins
- ➔ Regulator has incorrect footprint
- ➔ Sensor band was not ordered

PCB Development: Order 2/3

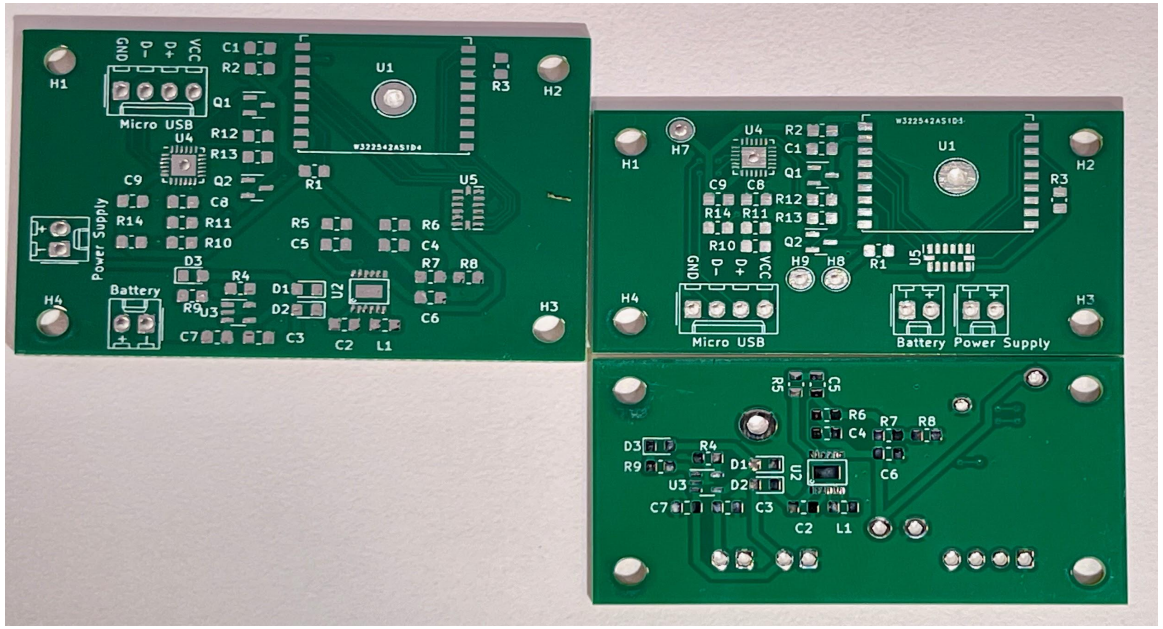
Stand



86x43 (mm)

60x40 (mm)

Sensor Band



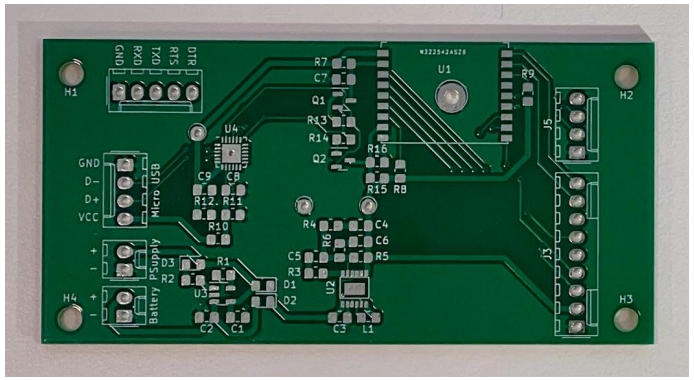
60x40 (mm)

56x30 (mm)
Double-Sided

➔ Programming IC has unconnected pins

PCB Development: Order 3/3

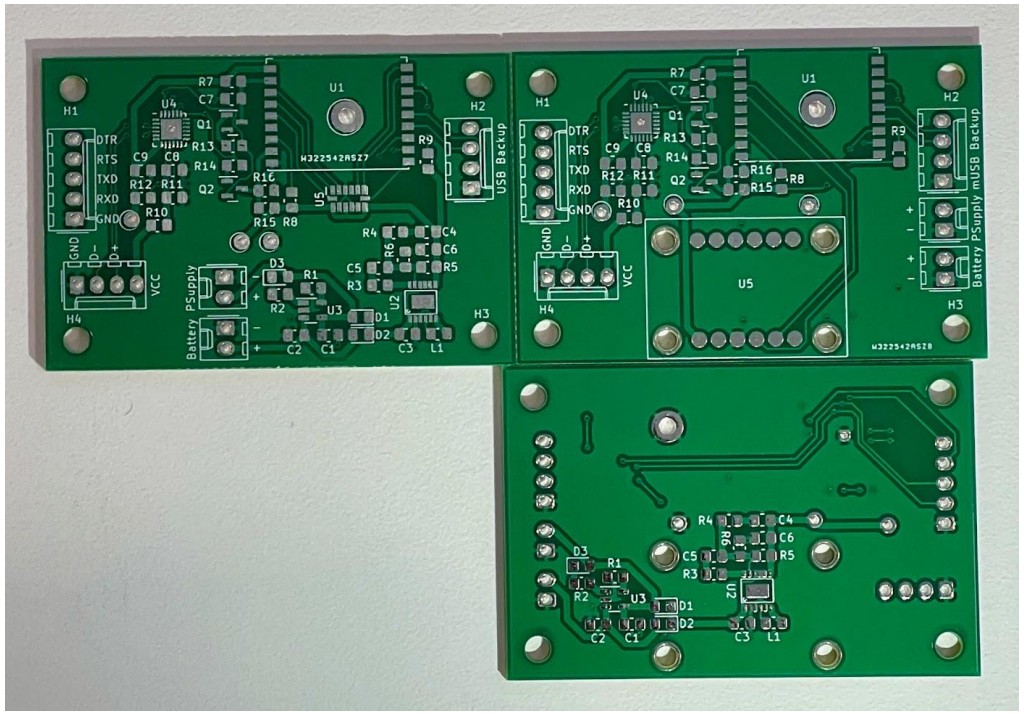
Stand



86x43 (mm)

➔ Complete PCBs!

Sensor Band



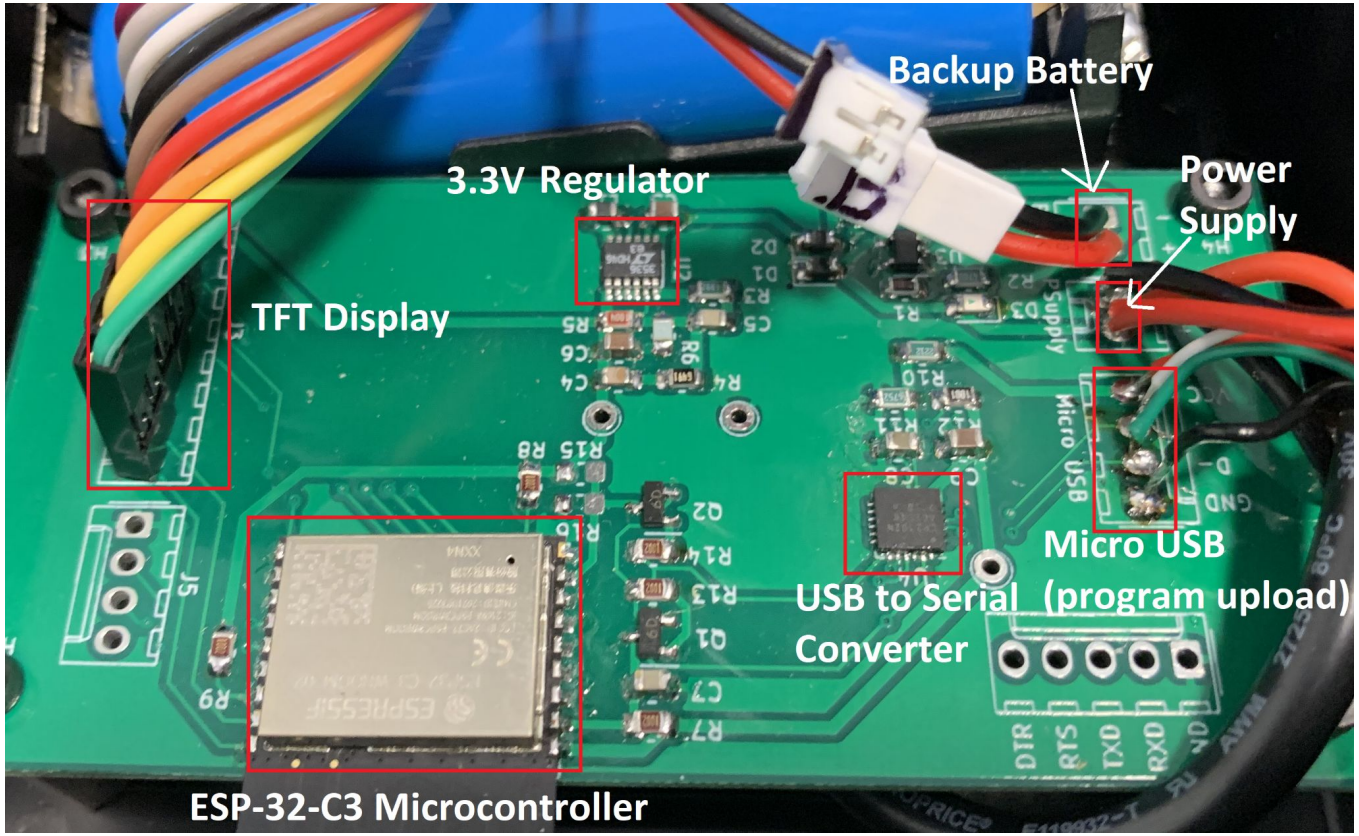
60x40 (mm)
ADXL343

60x40 (mm)
MPU6050
Double-Sided

Stand Power Subsystem

Stand Power Subsystem

- ➔ Rechargeable battery
- ➔ Battery management system
- ➔ Diode OR circuit
- ➔ Switching Regulator



Design: Stand Power Subsystem

RV Table

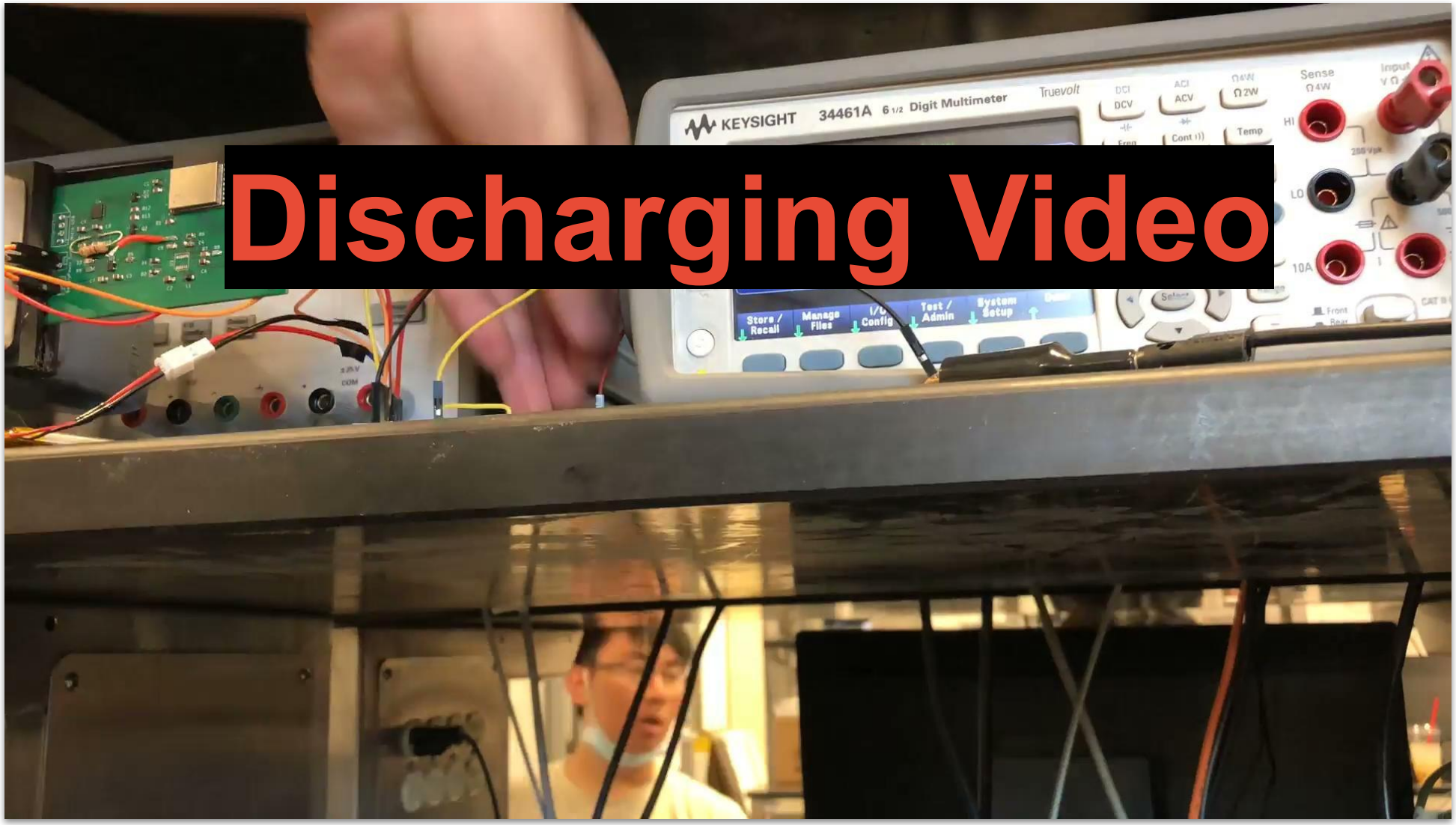
Requirements	Verification
1. Provide 3.3V±5% from a 3.7V-5.0V VDC source determined by the primary power connection.	1A. Using an oscilloscope, measure the output voltage of the linear regulator by probing the Vout with the positive terminal and probing GND pin on the ESP32 microcontroller.
3. Regulator can operate between 0mA to 750mA of current	3A. Place a 1kΩ potentiometer and a 4.4Ω resistor between the regulator output and ground. 3B. Sweep the potentiometer between 0Ω to 1kΩ and measure the output voltage using an oscilloscope, ensuring that the output voltage stays within 3.1V to 3.5V.
4. Voltage spikes that occur when disconnecting the 5V power supply must remain within 3.0V to 3.6V	4A. Ensure that both the power supply and the battery are connected 4B. Connect an oscilloscope to the regulator output and set it to trigger on the falling edge at 4.3V 4C. Disconnect the power supply and ensure that the voltage remains within 3.0V to 3.6V

RV Table

Requirements	Verification
5. BMS charges the Li-ion battery without allowing the battery to overcharge or undercharge.	5A. Allow the battery to discharge over time. 5B. Once an hour, measure the current across a 10Ω resistor with multimeter until it reads between 0mA and 5mA 5C. Apply 5VDC to the BMS pin 4. 5D. Using a multimeter, measure the current through the battery once every minute and use Riemann summation to calculate the total charge of the battery. Ensure that this value is less than or equal to 2600mAh.
6. Display shows record while device is unplugged for at least 12 hours.	6A. Apply 5VDC to the BMS pin 4 until the current through the battery is between 0mA and 1mA, as measured by a multimeter. 6B. Unplug the power supply from the wall outlet and let HFT Stand remain for 12 hours. 6C. Check that the device still has power by observing that the display is on.

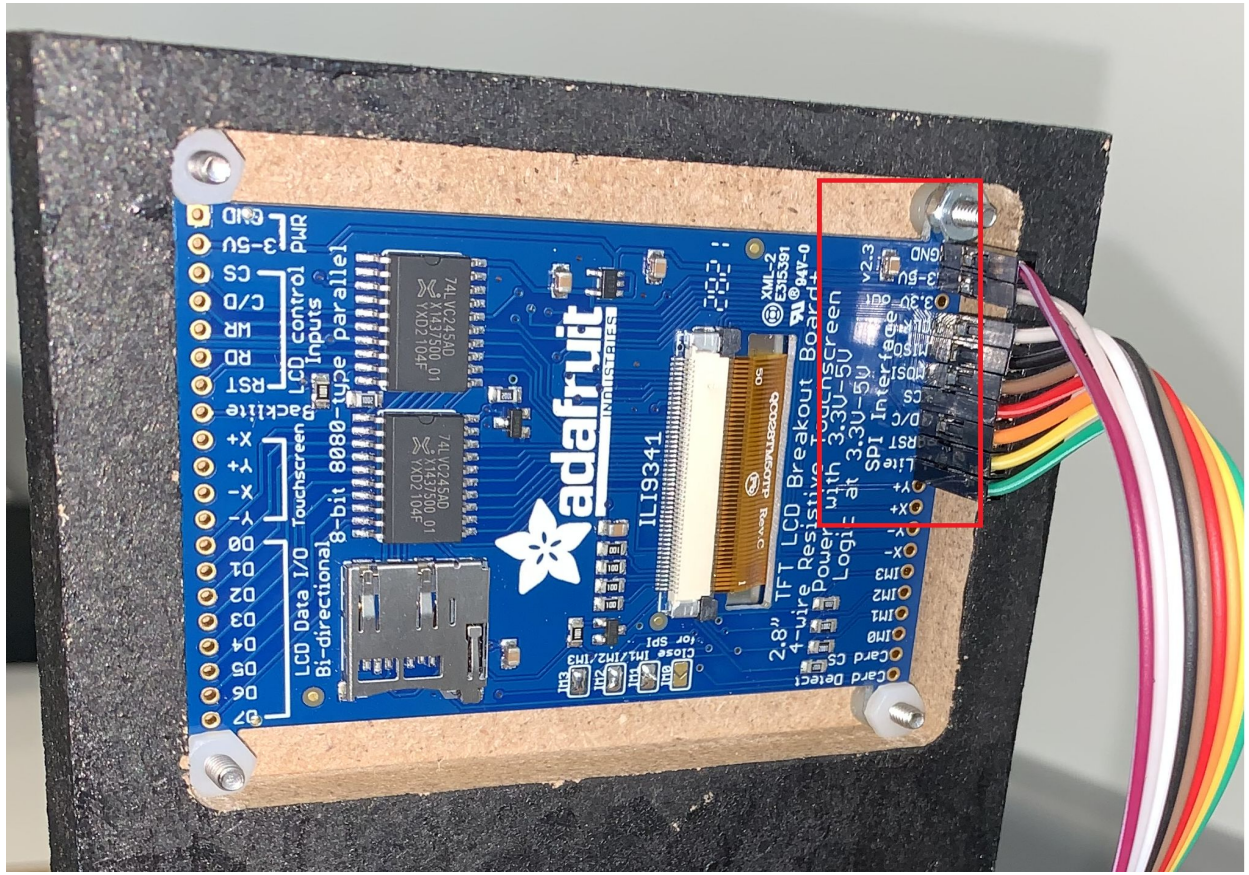
Design: Stand Power Subsystem

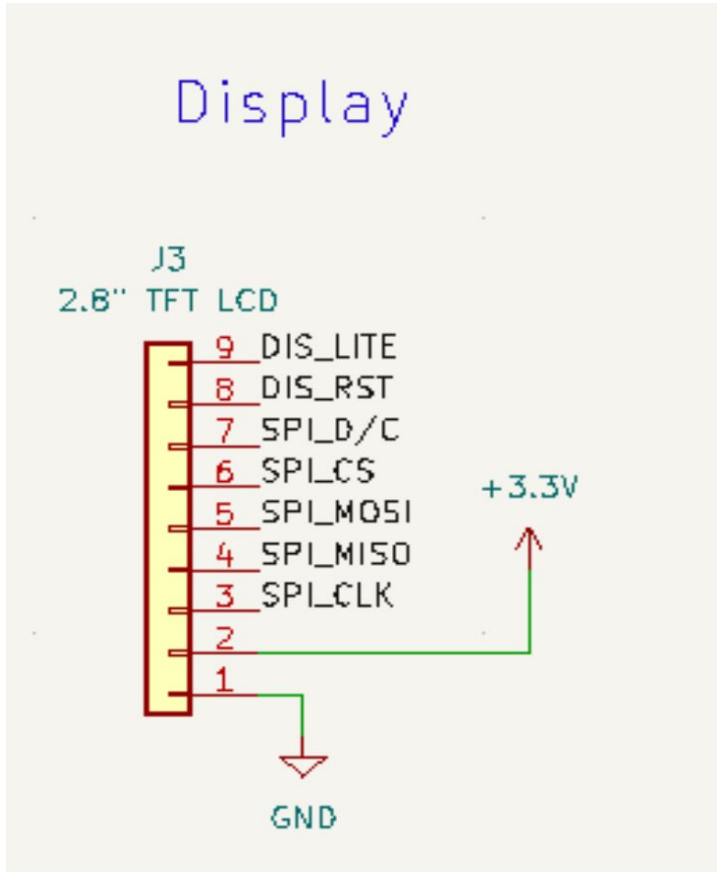
Design: Stand Power Subsystem



Display Subsystem

Adafruit ILI9341 2.8” Touchscreen Display





- ➔ Confirming a Display was a core bottleneck for our project.
- ➔ Wiring the RST pin was necessary
- ➔ Touchscreen functionality was not implemented.
- ➔ Utilized Adafruit GFX library.

Requirements and Verification Table

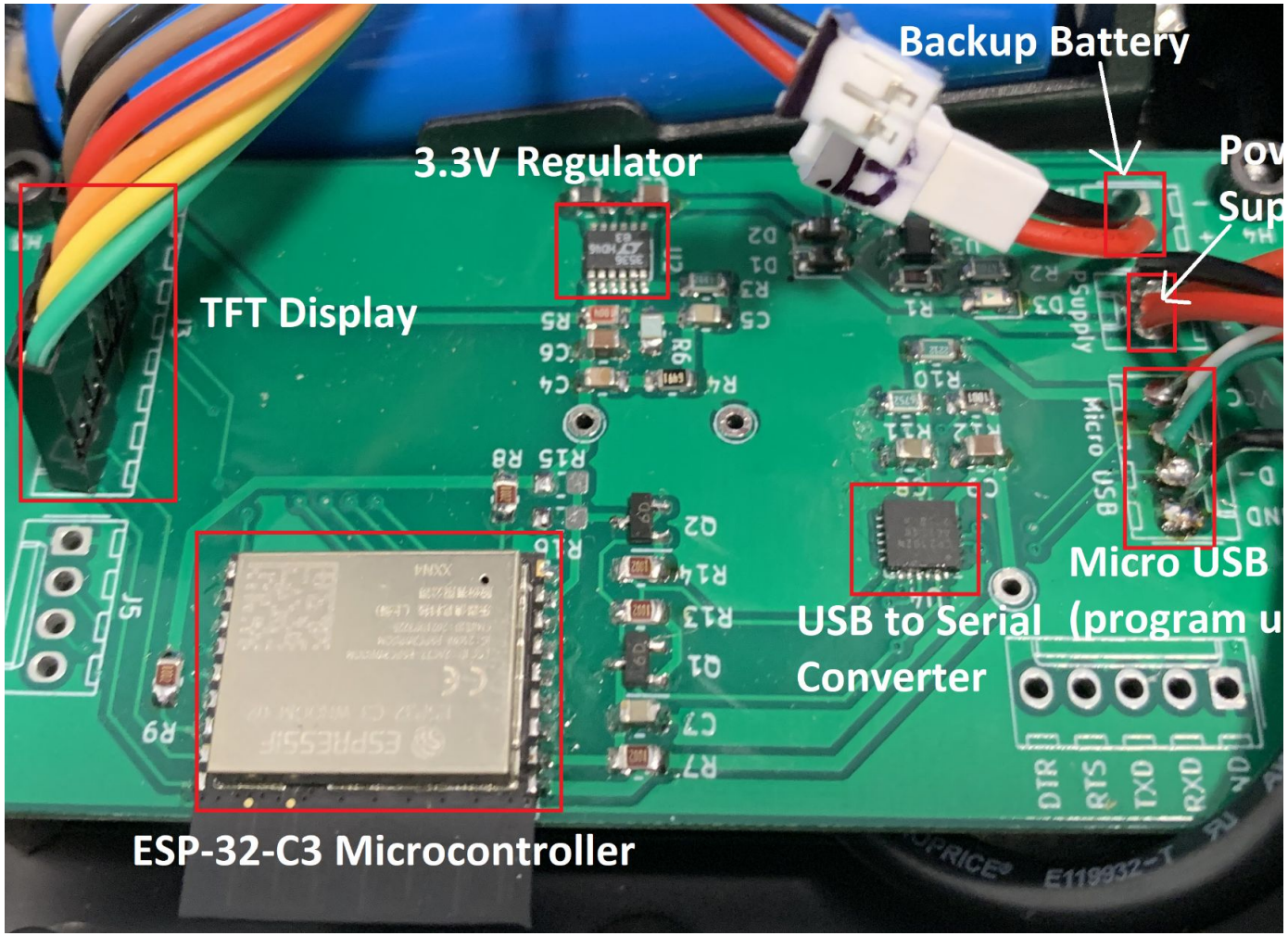
Requirement	Verification
1. Voltage received when updating the display stays within 3.3V±5% at 350mA.	1A. With a multimeter, measure voltage into Pin 2 (V_{DD}) from the Stand Power Subsystem and probing GND. 1B. Validate that the voltage measured display on the multimeter is within 5% of the operating voltage of 3.3V. 1C. Attach a10Ω and 75Ω resistor in parallel to the output of our board MCU and validate the operating current provided to display does not exceed 350mA.
2. Commands from Control Subsystem are processed within 250 ms.	2A. Connect the display over the I2C bus to GND, GPIO 22 and GPIO 21, and Vcc pins on the ESP32 board microcontroller. 2B. Send Start_Timer signal from ESP32. Microcontroller within Control Subsystem 2C. Verify through the serial monitor the time taken to receive the display instruction is under 250ms.
3. When the power cord is unplugged,the calendar and timer will still be displayed.	3A. Connect the display over the I2C bus to GND, GPIO 22 and GPIO 21, and Vcc pins on the ESP32 board microcontroller. 3B. Send Start_Timer signal from ESP32 Microcontroller within Control Subsystem 3C. Unplug the power supply cord from the wall outlet. 3D. Verify that the timer and calendar are still displayed with the power supply removed.
4. Records are shifted over to the left at the start of a new day, each record is distinct and not drawn with a large hard-coded red background.	4A. Run the Shift Test Program and verify outputs.
5. From 4am to 4pm the HFT Stand correctly marks the time period as morning and morning records are modified, and from 4pm the evening records are modified.	5A. Run the Shift Test Program and verify outputs.

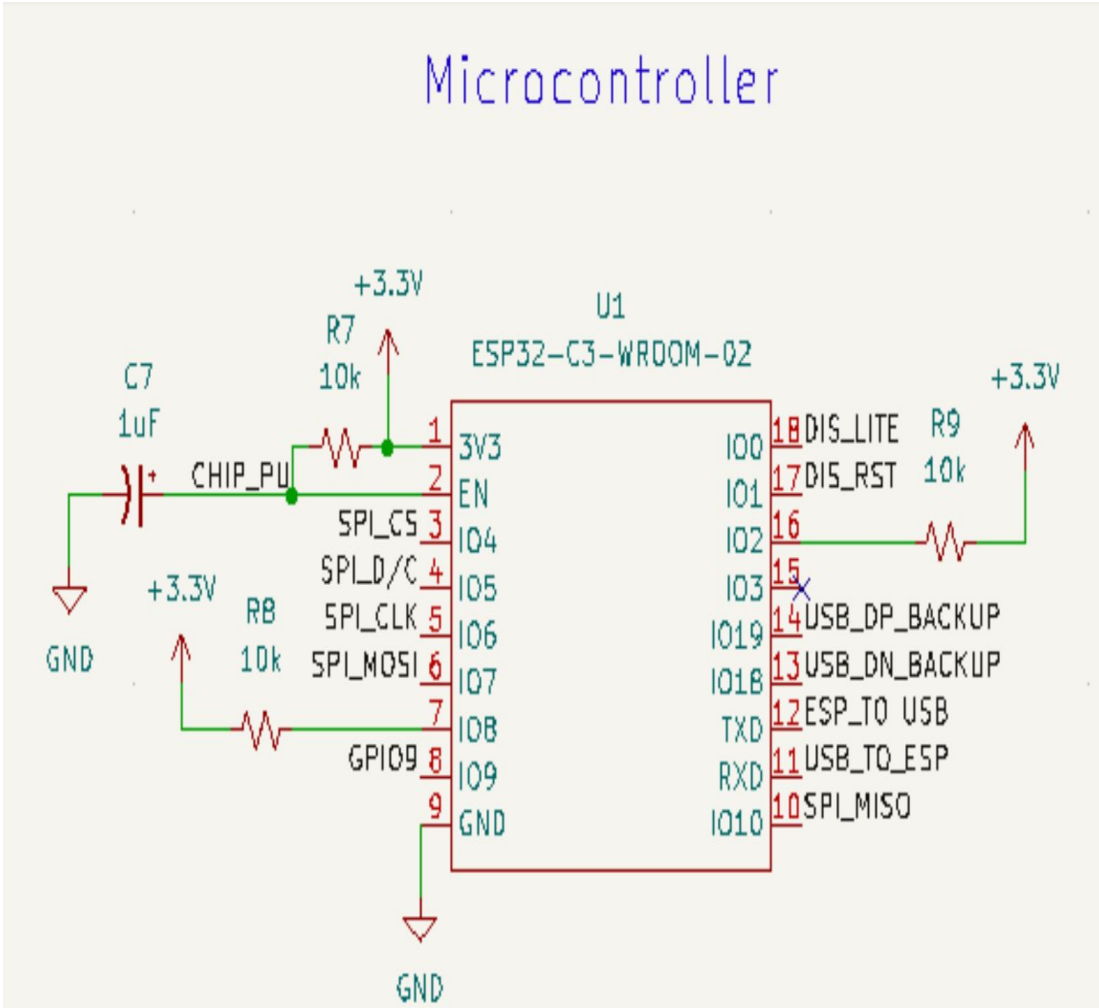
Design: Display Subsystem

- ➔ Verification of 3rd requirement: backup battery powers display and maintains record.
- ➔ Flash Memory through EEPROM Library posed additional challenges.

Control Subsystem

Design: Control Subsystem





Control Subsystem Requirements and Verification Table

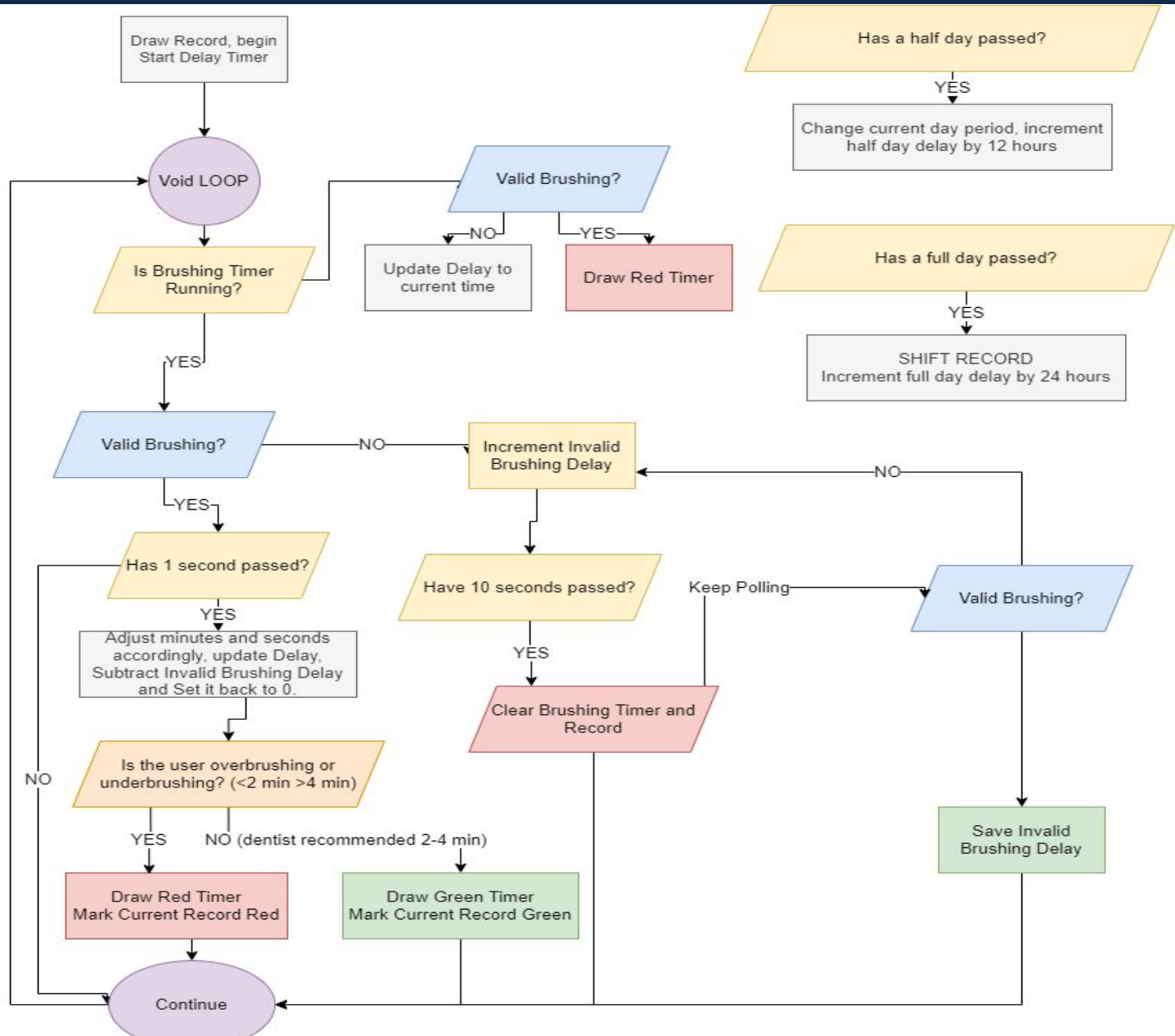
Requirement	Verification
The board microcontroller can interact with the ESP32 remote peer over ESP-NOW WiFi protocol and receive display instructions at a baud rate of 115200 bps.	1A. Include the libraries Wifi.h and HTTPclient.h in the software code 1B. Set the SSID and password to the network credentials of the remote ESP32 server (“ESP32-access-point” and “ece445” respectively). 1C. Send an HTTP request to the default URL (192.168.4.1). 1D. Verify through the ESP32 client serial monitor contains an HTTP response with a value higher than > 0.
The subsystem can actively calculate time of day, manage the shift from morning to evening, and shift the calendar when a twenty-four hour period has passed.	Successful run of shift program which simulates morning and evening cycles and change of day over a week.
The subsystem must detect and communicate with any device that is connected to the I2C addresses (7bit/10bit addressing modes).	The ESP32-3C Microcontroller can run display test programs including the shift program which can be seen on the Adafruit ILI9341 Display.

Half Day and Full Day Functionality through Shift Test

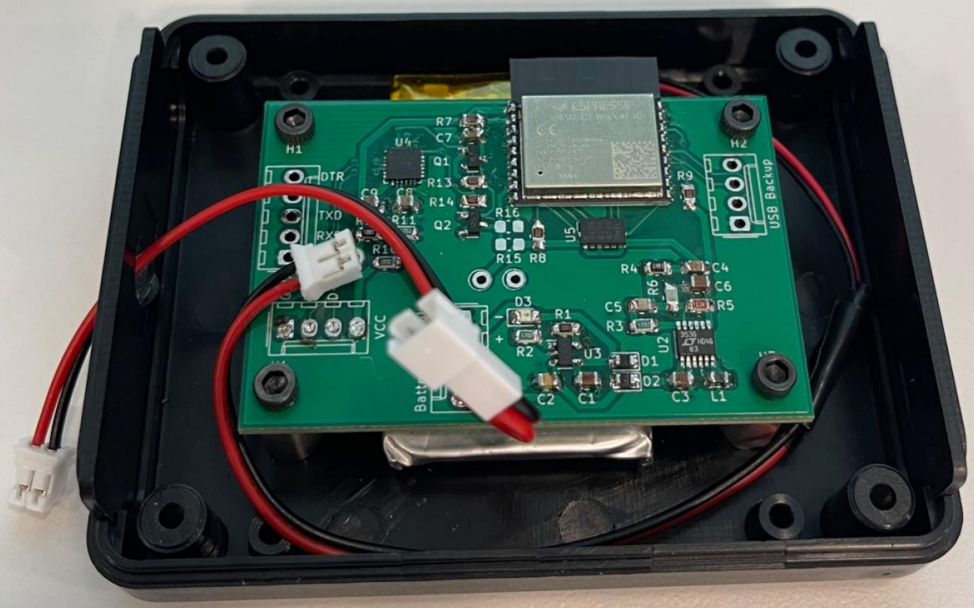
- ➔ This test demonstrates functionality of internal clock and control subsystem:
  - ◆ Morning and Evening Adjustment
  - ◆ End of Day Shift Helper Function
- ➔ We also verified our second high-level requirement.

Control Subsystem Workflow

- setup and void loop
- The millis() function
  - Limit: 50 days
- Modular functions
- ESP32-C3 and ESPNOW protocol
- polling for valid_brushing
  - ML Classification had to be moved onto Sensor Band



Remote Sensor Band Subsystem

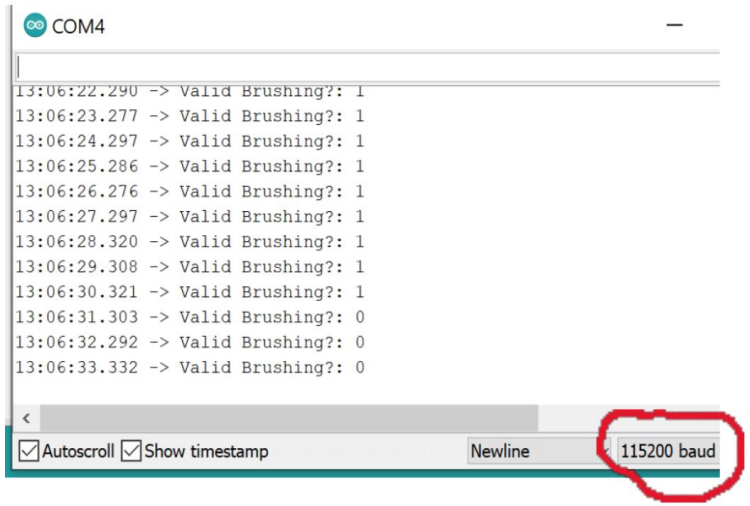


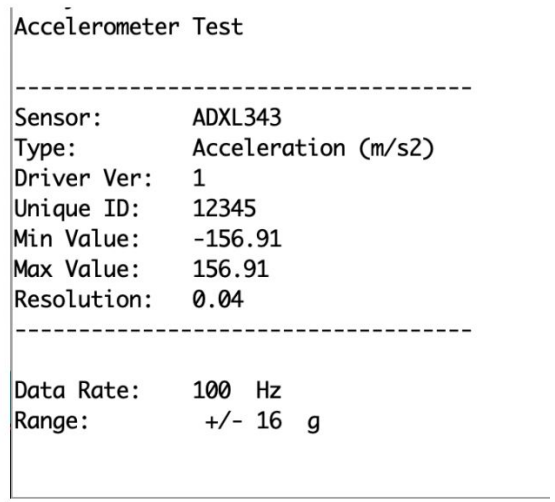
Design: Remote Sensor Band Subsystem

Requirements and Verifications

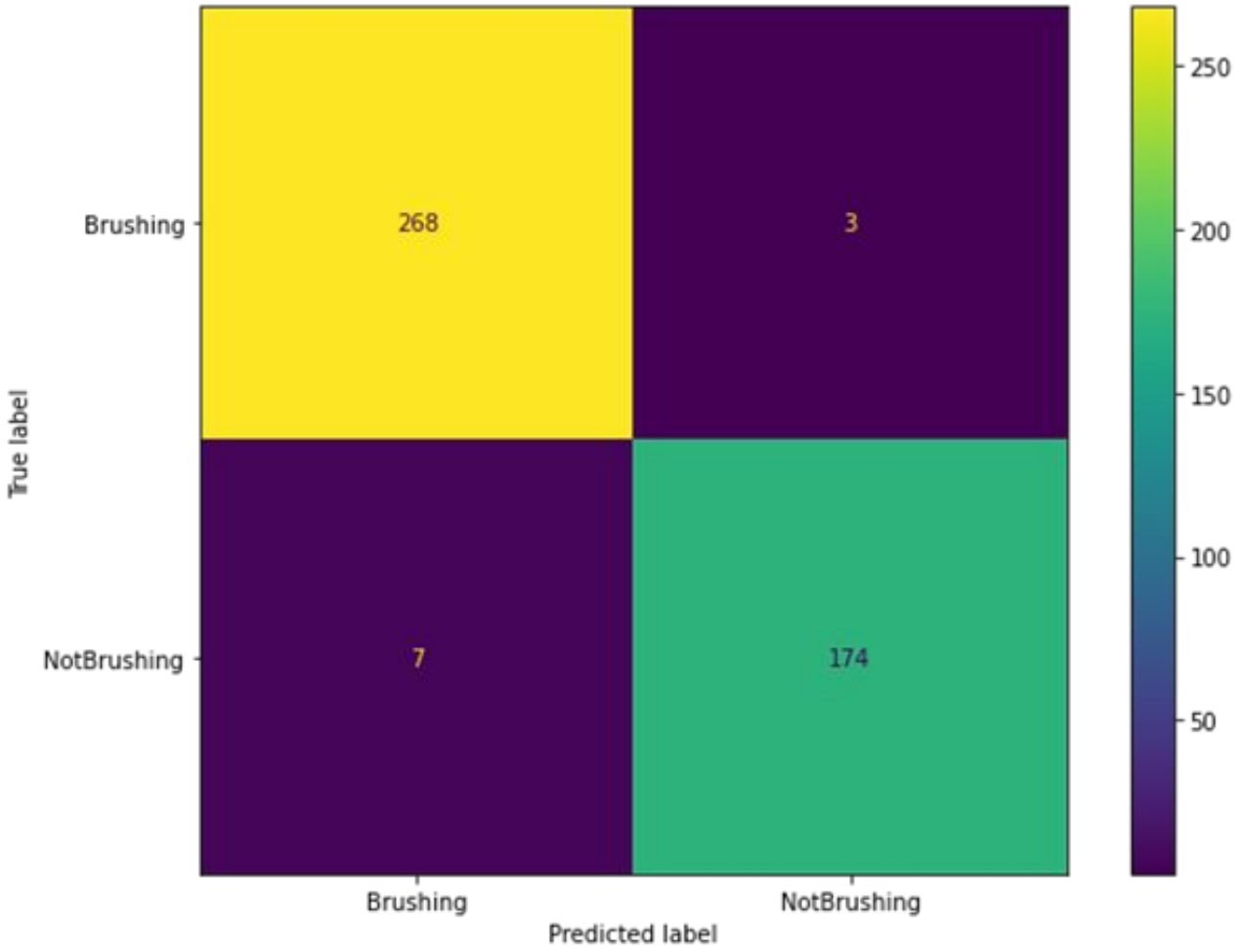
Requirement	Verification
Li-Poly Battery provides 3.3V± 5% from a 3.0 to 4.2V source at a current up to 500mA.	Oscilloscope showed output of 3.3V at regulator from the Li-Poly battery at all operating voltages.
Accelerometer can detect linear acceleration at ±16g and 0.4 mg/LSB (16 bit) sensitivity with an output data rate of 100 Hz.	Accelerometer sensitivity and data rate set in the software during the testing phase of design
Microcontroller is recognized as a peer over ESP-NOW protocol and delivers sensor data within a maximum latency of 50ms±25ms at baud rate 115200 bps.	ESP-NOW testing sketch shows test messages are delivered with an average latency of 63ms.
Sensor Band Li-Poly battery is recharged when placed on the toothbrush stand.	BMS LED turns on when connected and a full charge cycle takes about 3 hours.
The Sensor Band can determine if the user is actively brushing their teeth with 95% accuracy.	Prediction matrices show 97% prediction accuracy on generated test sets

- ➔ Determining Battery Life
  - ◆ Active
    - 50mA average draw
    - 850 mAh / 50mA = **17 hours**



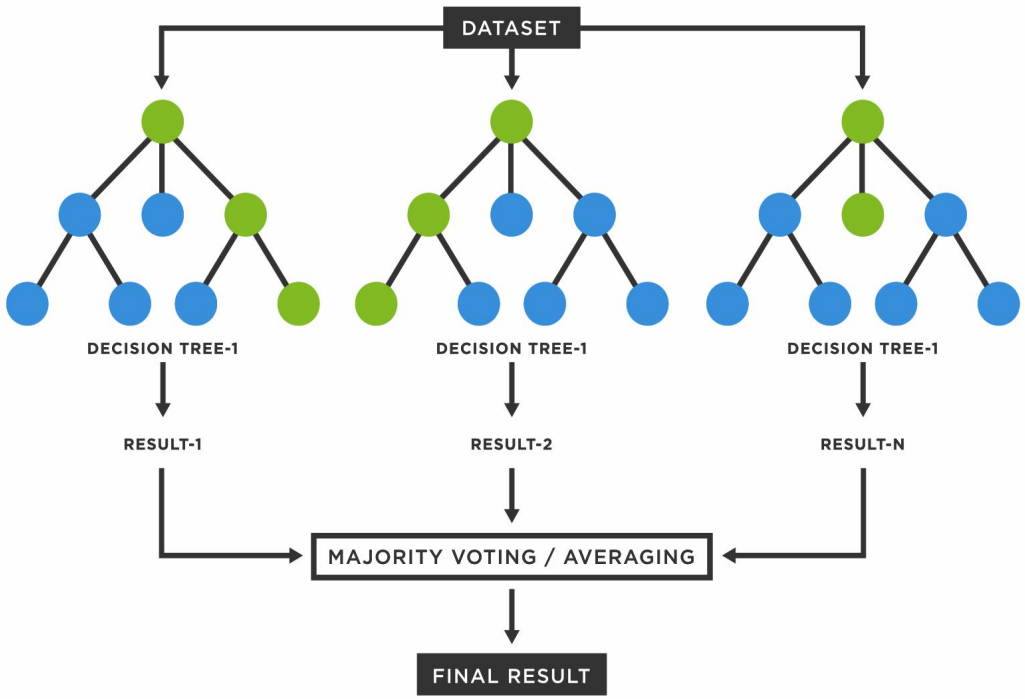


Brushing Detection Algorithm



Design: Brushing Detection Algorithm

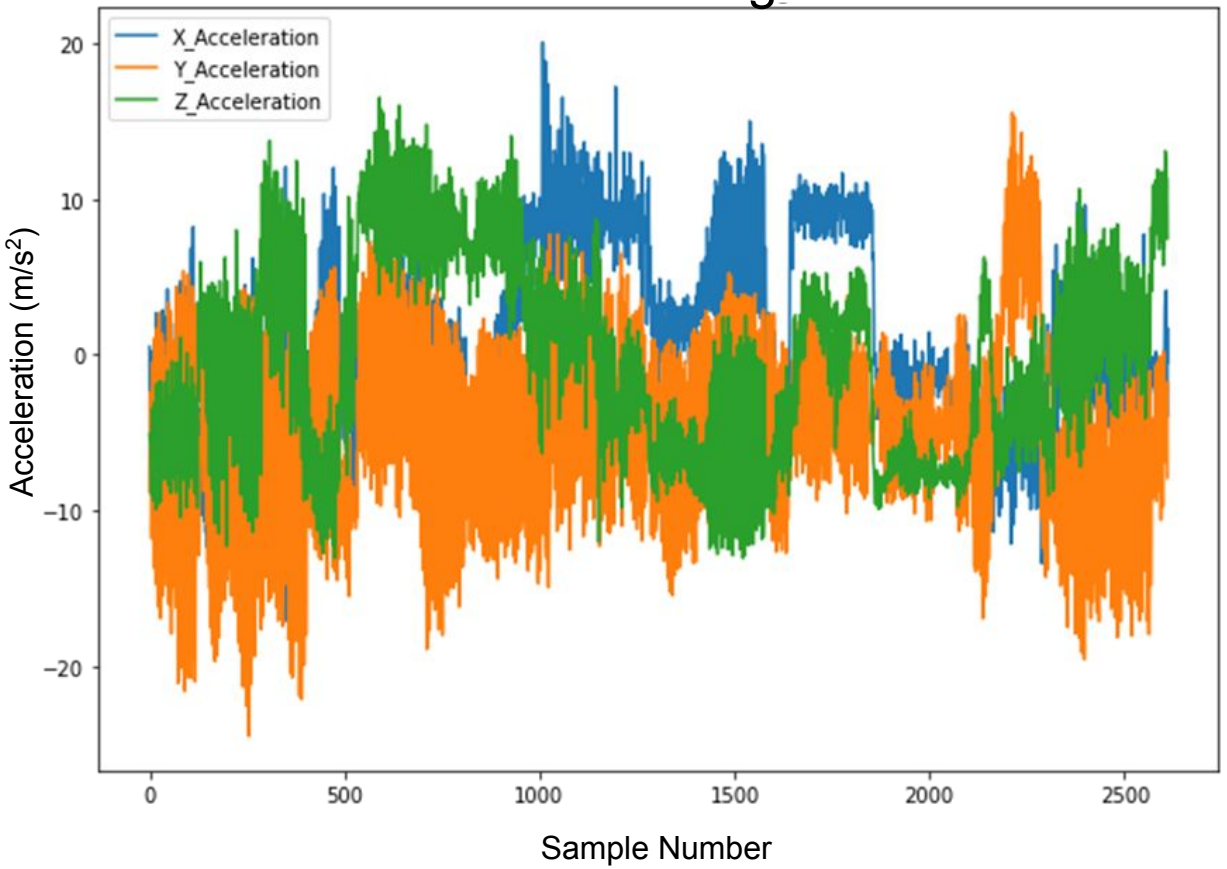
- Utilizes the random forest classification
  - Decision trees are uncorrelated
  - Majority voting
- Based on windows of acceleration data
  - Instantaneous isn't enough!
  - Window shifts to fill new data samples
- Achieves 97% prediction accuracy on generated testing sets
  - Classification Accuracy
    - 442 / 452 = **0.97787 or 97.78%**
  - Error Rate
    - 10 / 452 = **0.02212 or 2.212%**



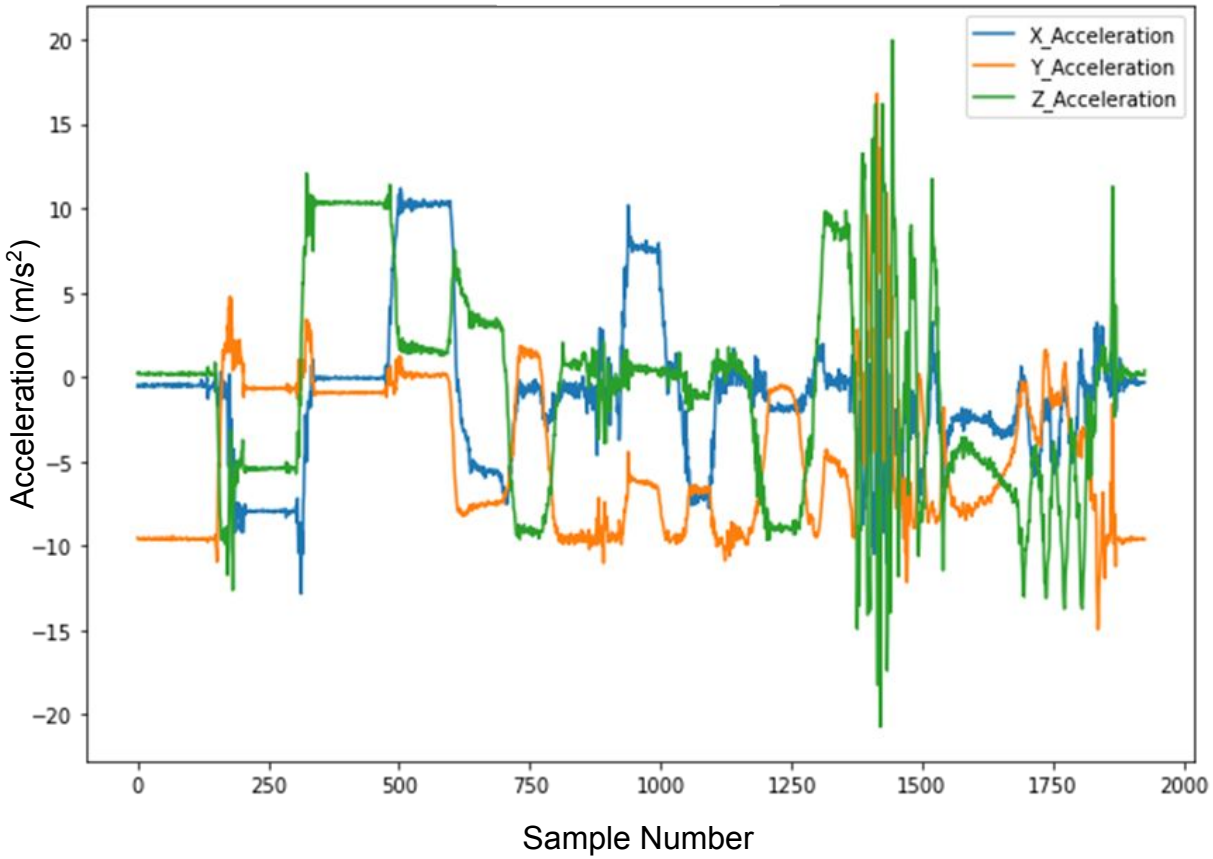
Random Forest High Level Algorithm

Design: Brushing Detection Algorithm

Active Brushing Data



Non Brushing Data



Time Series Data for Non-Brushing and Active Brushing Processes

Wrapping Up

Wrapping Up

Successes and Challenges

- ➔ Functional Prototype
- ➔ Classifier Accuracy above 95%
- ➔ Data is preserved

- ➔ Sensor Band Size
- ➔ PCB Design and uploading to Microcontroller
- ➔ Charging Mechanism

Future Revisions

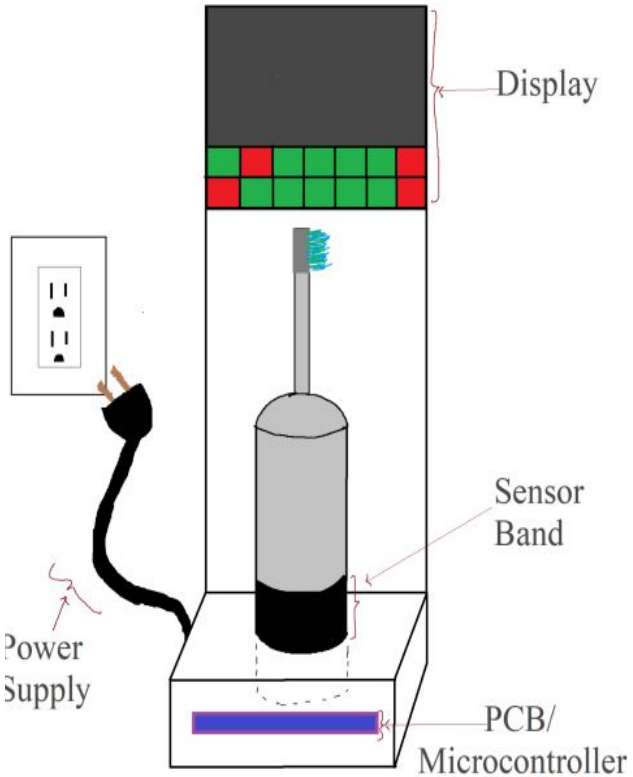
- ➔ Downscale the size of Sensor Band
  - ◆ Different designs
- ➔ Incorporate electric toothbrushes for our brushing detection algorithm.
- ➔ Multiple profiles for different users.
- ➔ Explore how to adapt product for more audiences, including colorblind users.

Ethics and Safety

- Battery Safety
  - Li-Poly
- Photosensitive Animations
  - Flickering
- Privacy
  - Transparent Data
  - Minimal Storage

Summary and Conclusion

- ➔ Project was successful
- ➔ Minimal changes to initial design
- ➔ Acknowledgements



Questions

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN