HABIT-FORMING TOOTHBRUSH STAND

Design Review

Rahul Vasanth (rvasant2), Quinn Palanca (qpalanc2), John Kim (johnjk5)

TA: Shivang Charan

Spring 2022

Table of Contents

1 Introduction

1.1 Problem

There are few habits as impactful as good dental hygiene. Brushing your teeth in the morning and night can significantly improve health outcomes. Many struggle with forming and maintaining this habit. At times, a small nudge is needed to keep kids, teenagers, and adults of all ages aware and mindful about their brushing habits. Additionally, many tend to zone out while brushing their teeth because they are half asleep or have no idea how long they are brushing. Parents might have a difficult time getting children to brush in the morning and before sleep.

For homeless shelter staff, rehab facility staff, patients, or anyone else looking to develop and track this habit, they may want a non-intrusive, privacy-preserving method [1] to develop and maintain the practice of brushing their teeth in the morning [2]. Keeping track of this information but not storing it permanently through a mobile application is something that does not exist on the market.

1.2 Solution

Our solution works by adapting electric toothbrushes to meet user needs. Unlike specific toothbrush brands that come with mobile applications, our solution can be applied to all electric toothbrushes, preserves privacy, and reduces screen time. We will implement a habit-forming toothbrush stand with a microcontroller, sensors, and LCD display that will work as a central hub for storing our toothbrush. A band of sensors will be wrapped around the base of the toothbrush, providing data to the central hub. Lifting the toothbrush from the stand, turning it on, and starting the brushing process will display a timer that counts in seconds up to ten minutes. Dentists recommend brushing twice a day for at least two minutes at a time [3], so our device solves the problem of brushing too quickly or losing track of time and brushing for too long.

Additionally, the display will provide an auto-adjusting calendar for brushing, with 14 graphical values coming from brushing your teeth in the morning or night during the current 7-day period. This will augment the user's awareness of any new trends, and potentially help parents, their children, and other use cases outlined above. We specifically store just one week of data as the goal is habit formation, not permanent storage of potentially sensitive health information in the cloud. The timer will display red numbers until the user has brushed for at least two minutes, after which the timer will turn green, and the current day and time period marker will turn from red to green (Figure 1).

1.3 Visual Aid

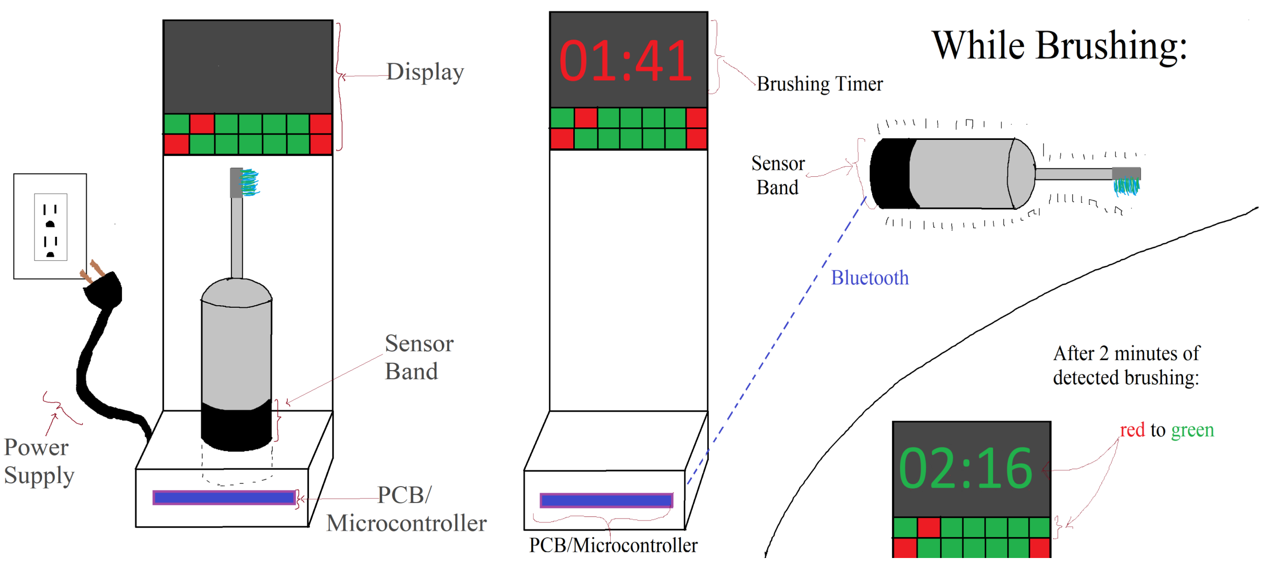

Figure 1: Device Physical Design

1.4 High-level Requirements

- 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).

- 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.

- The Habit-forming Toothbrush Stand can determine if the user is actively brushing their teeth with 95% accuracy.

2 Design

2.1 Block Diagram

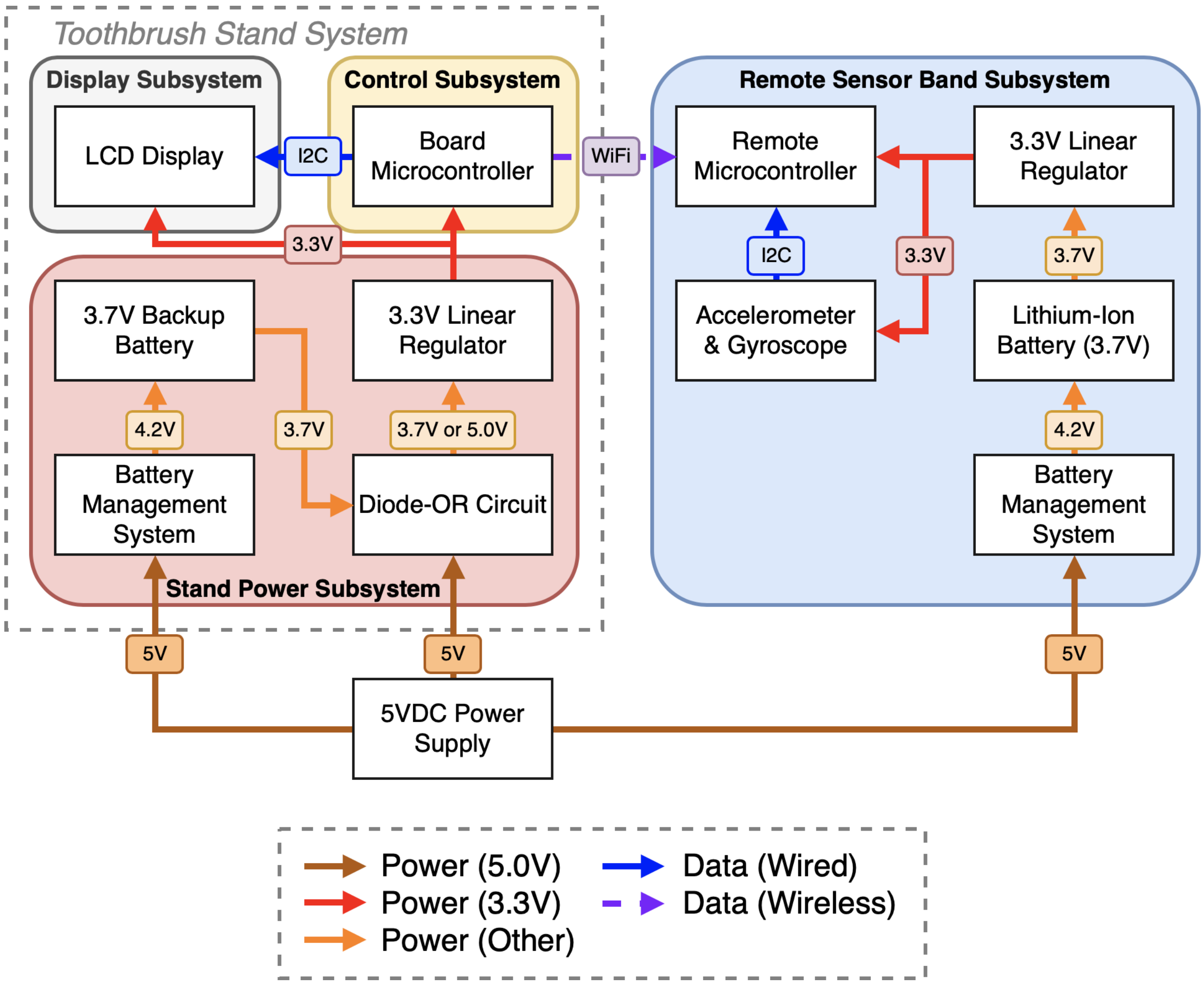

Figure 2: Block Diagram

2.2 Remote Sensor Band Subsystem

The remote sensor band subsystem will be tracking the user's motions during the toothbrushing process, utilizing sensor data by putting it through our algorithm to determine display instructions, and sending data to our board microcontroller in the toothbrush stand. By using a lithium ion polymer battery to power our sensor band we will be able to take the provided voltage of 3.7V and send it through a linear voltage regulator to bring the voltage down to 3.3V to properly power our ESP32 microcontroller and accelerometer sensor. This battery will be recharged when the toothbrush is placed on the stand through the use of an integrated circuit battery management system (BMS).

Additionally, the remote sensor band microcontroller will act as our access point such that the board microcontroller will request instructions based on the amount of time elapsed as well as the data from our dual accelerometer and gyroscope sensor. By doing so, the boards will be able to communicate and request data from each other through Espressif's ESP-NOW connectionless WiFi communication protocol [5].

2.2.1 Lithium-Ion Battery and 3.3V LDO Linear Regulator

Input: Rechargeable 3.7V Lithium-Ion Polymer Battery
Output: 3.3V±5% for LSM6DSLTR 6-axis Gyroscope + Accelerometer
3.3V±5% for ESP32-C3-Wroom-02 Microcontroller

Our remote sensor band will be powered using a single cell lithium-ion battery with an operating voltage of 3.7V. However, the voltage that will be supplied to our microcontroller and accelerometer sensor will be 3.3V that is stepped down from a linear regulator. We have provided ourselves with enough overhead to ensure functionality is maintained after battery drain has occurred. The battery will provide power to the components contained only in the sensor band when the toothbrush is not seated in the toothbrush stand.

To ensure that we can accommodate users who may exceed the dentist's recommended two minutes of brushing [3], we have calculated the minimum battery capacity that is needed for up to 10 minutes.

$$\text{Capacity [Ah]} = \text{Current [A]} * \text{Runtime [hours]}$$

$$\text{Capacity [Ah]} = .5A * .1666 \text{ hours} = 0.08Ah$$

While the battery capacity only needs to meet a minimum requirement of 0.08Ah while the user is actively brushing, by using a lithium-ion battery with a capacity of 1.5Ah we provide a reasonable overhead capacity if the user were to leave the toothbrush disconnected for up to three hours assuming maximum power consumption. The single cell battery is certainly larger than what is necessary, but it can supply the operating current necessary for the maximum power draw of our ESP32 microcontroller.

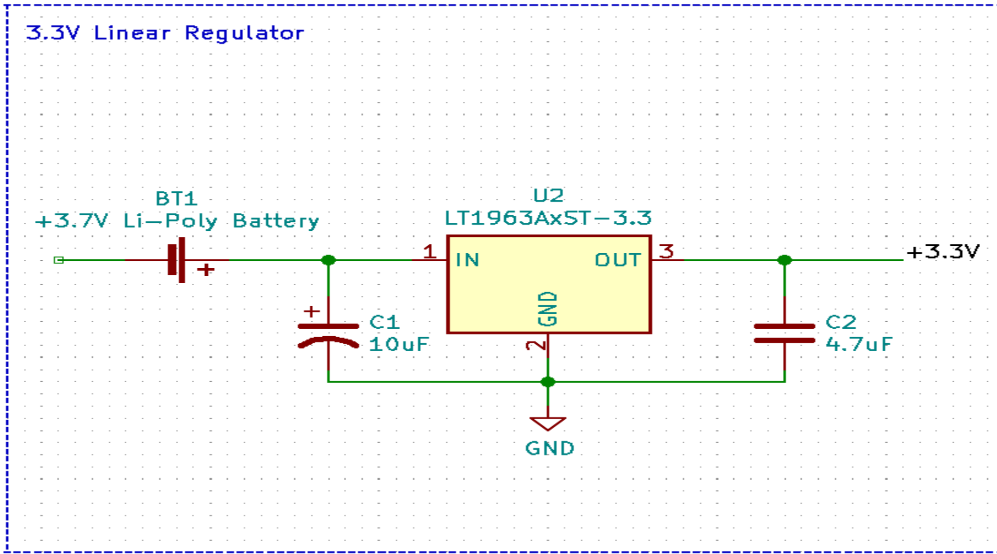

Figure 3: LT1963AxST +3.3V Linear Regulator Schematic

2.2.2 Battery Management System (MCP73831T-2ATI/OT)

Input: 5.0V±5% from SWI5-5B-N-P5 5VDC Power Supply
Output: 4.2V±5% regulated voltage for 3.7V Rechargeable Lithium-Ion Battery

Since our sensor band is powered by a lithium-ion polymer battery with the capability to be recharged we have elected to use an integrated circuit battery management system (BMS) that receives an input of 5.0V when seated on the toothbrush stand and outputs a regulated 4.2V to charge the battery. The specific model of BMS was chosen as it is designed to be most efficient charging a single cell battery such as the one used in our project.

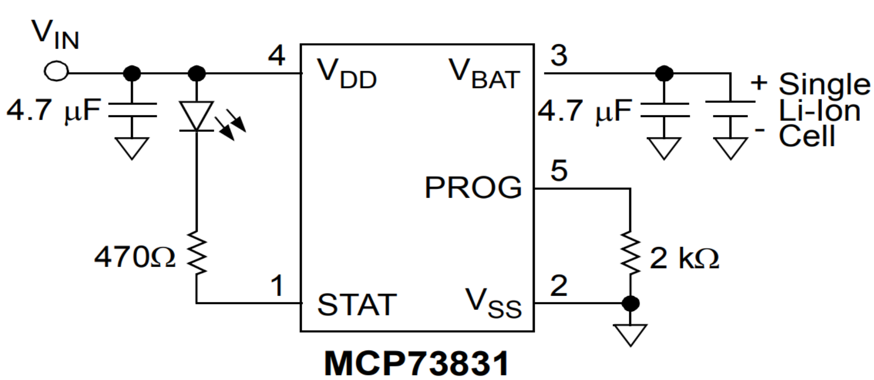

Figure 4: MCP7383 BMS Default Setup

2.2.3 Accelerometer (LSM6DSLTR Module 6-axis Gyroscope + Accelerometer)

Input: Physical movement and orientation of the sensor chip
Output: Digital acceleration and orientation data through I2C data line

The sensor band will contain both an accelerometer/gyroscope sensor that will capture data for our microcontroller. This will allow us to determine if the user has actually started the process of brushing their teeth, rather than a false positive. We will experiment with the overall rotational orientation, but knowing whether the toothbrush is parallel to the ground, or linearly accelerating back and forth will give us validation. The data will be computed on an ESP32 microcontroller attached to the sensor band which will send the instructions over ESP-NOW WiFi communication protocol to the ESP32 board microcontroller.

Our linear acceleration data will need to be detected when at ranges up to ±16g from our accelerometer; however, once unit testing has been completed we will be able to dial our sensitivity lower if needed. Our sensor supports ±2g/±4g/±8g to achieve the full-scale acceleration range and the available gyroscopic range is 125-2000 degrees/second.

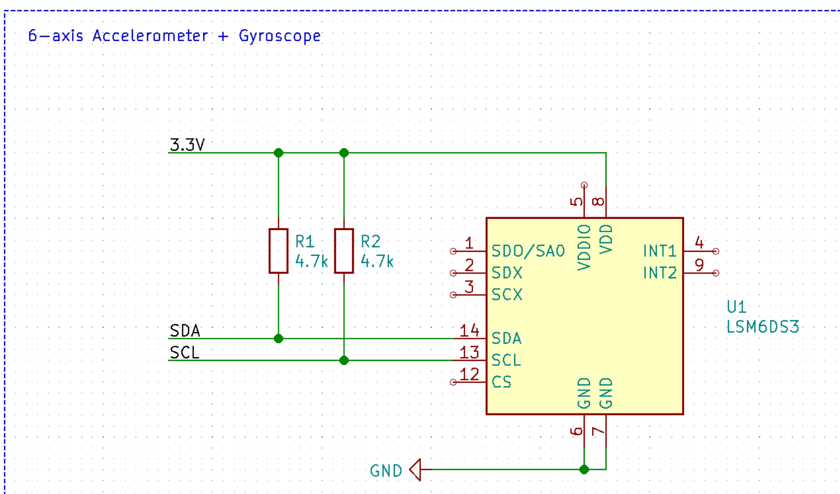

Figure 5: LSM6DSLTR 6-axis Gyroscope + Accelerometer Schematic

2.2.4 ESP32 Microcontroller (remote)

Input: 3.3V±5%, 6-axis acceleration and angular velocity data

Output: commands to transmit display instructions over ESP-NOW P2P WiFi Protocol, commands to read/write 6-axis data

The ESP32 microcontroller on the remote sensor band will constantly be polling for acceleration and orientation data from our 6-axis sensor. With built-in WiFi and Bluetooth Low Energy capabilities, we will be able to send our display data through Espressif's ESP-NOW protocol [5] to the board microcontroller that is housed in our toothbrush stand. By using this remote microcontroller we can perform any calculations remotely before sending the less time sensitive display data to our board microcontroller.

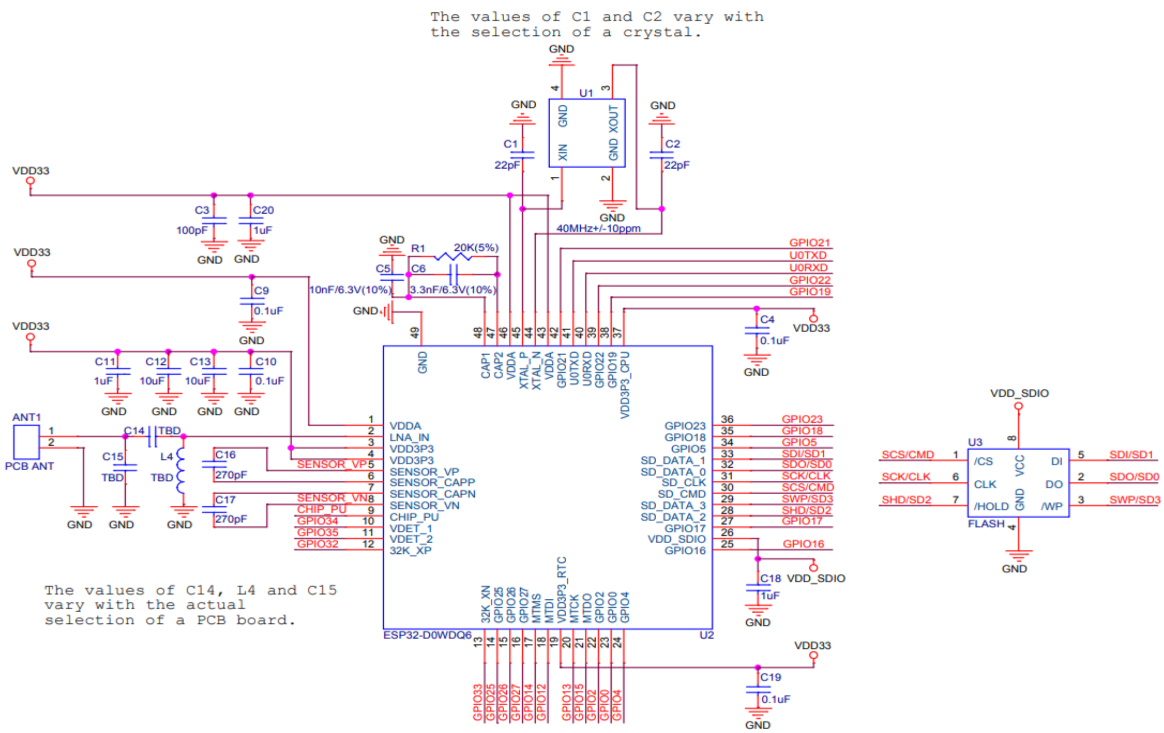

Figure 6: ESP32 Microcontroller Schematic by DigiKey

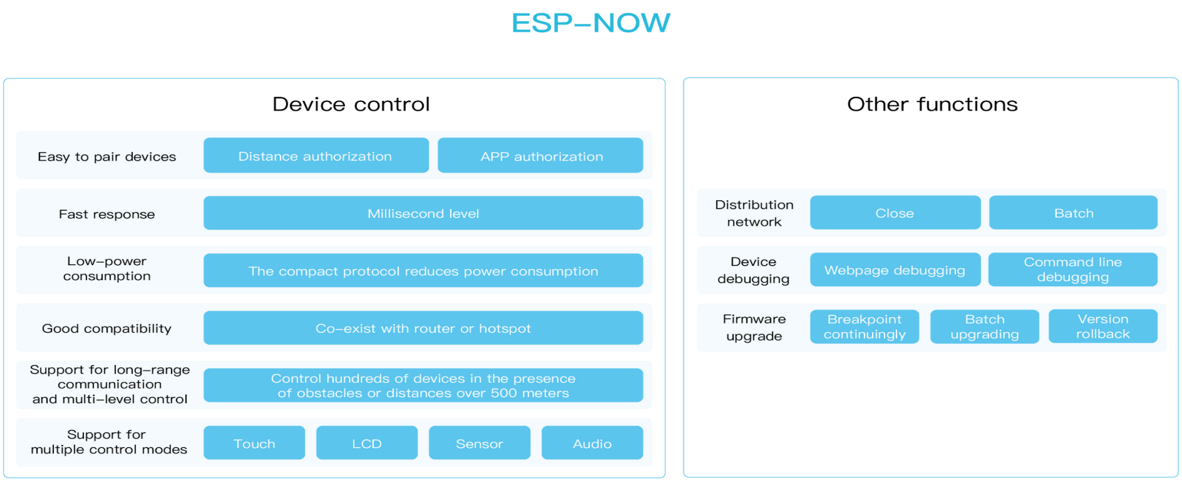

Figure 7: ESP-NOW Functionality List

Requirements	Verification
1. Li-Poly Battery provides 3.3V± 5% from a 4.2V source at an operating current of 500mA.	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. 1B. Connect a 1kΩ resistor and 1.9 kΩ resistor in between the output of the regulated voltage and ground; measure the output current using an oscilloscope and verify that it is at least 500mA.
2. 6-axis IMU can detect linear acceleration at ±16g and 0.488 mg/LSB (16 bit) sensitivity with an accuracy of 10% at an output data rate of 1.6kHz.	2A. Connect the IMU to the ESP32 MCU through I2C. 2B. Move the sensor at a fixed speed on the X, Y, and Z axes over five trials. 2C. Compare the data from each of the five trials and ensure that the readings are within 10% of each other.
3. 6-axis IMU can detect rotational movement with at most 360 dps with an accuracy of 10% at an output data rate of 1.6kHz.	3A. Connect the IMU to the ESP32 MCU through I2C. 3B. Rotate the sensor at a fixed speed across set angles pre-measured with a protractor. 3C. Compare the data from each of the five trials and ensure that the readings are within 10% of the measured angles.

4. The microcontroller can be recognized as a peer over ESP-NOW protocol and receive sensor data with a maximum latency of 50ms±25ms at baud rate 115200 bps.	4A. After running the code to set up the MCU, use a separate WiFi enabled device to test the peer to peer connection. 4B. Using request URLs for acceleration data, ensure that our MCU is properly receiving acceleration and gyroscope data at the rate of 115200 bps in the Arduino serial monitor. 4C. In unison with the board MCU, verify the latency of display requests through the output in the Arduino serial monitor.
5. Sensor Band Li-Poly battery is recharged when placed on the toothbrush stand.	5A. Disconnect the sensor band from the toothbrush stand and allow the battery to discharge over time. 5B. Once an hour has passed, measure the current across a 10Ω resistor with a multimeter until it reads between 0mA and 5mA. 5C. Using a digital supply, apply 5VDC to pin 4 of BMS. 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 1500mAh.
6. The Sensor Band can determine if the user is actively brushing their teeth with 95% accuracy.	6A. Generate Training Data Set with 50 instances of active brushing, and 50 instances of invalid brushing. (Attach sensor band to tooth 6B. Generate Testing Data Set with 40 data points. 6C. Tune weights and parameters of model on Training Data set until achieving 99% accuracy 6D. Obtain accurate results for 38/40 cases on Testing Set (95% accuracy)

Table 1: Remote Sensor Band Requirements and Verifications

2.3 Control Subsystem

The control subsystem is responsible for handling the wireless reception of the display information that is sent from our remote sensor band subsystem. Due to the time sensitive nature of our sensor data, the control subsystem will only be receiving the display information as it will reduce the power draw of our ESP32 microcontroller which can burst upwards of 500mA when operating on WiFi/Bluetooth [4]. By only sending the information necessary to update our LCD display, the board microcontroller will only need to be receiving signals twice – once when the brushing has begun and again when the brushing process has finished. Our decision to use WiFi was influenced by the ability to set up both the remote and board microcontrollers as access points so that Espressif's ESP-NOW WiFi protocol can be used to send data reliably with a lower power consumption.

This subsystem will be powered by the 3.3V regulated voltage output that is supplied by the Stand Power subsystem. Additionally, the subsystem will interface with the display subsystem by communicating over the I2C bus operating as the master device.

2.3.1 ESP32 Microcontroller (Board)

Input: 2.3V-3.6V, commands to transmit display instructions from remote ESP32 MCU
Output: commands to display subsystem via I2C bus

Using Espressif's ESP-NOW connectionless WiFi communication protocol, the ESP32 board microcontroller will be set up as the singular peer connecting to the remote microcontroller through a connection similar to WiFi. Since we are only keeping track of brushing during the morning and night, the microcontroller will spend the majority of the operating time in sleep mode. In figure 8 below we can see that the ESP32 microcontroller consumes around 126mA of current when WiFi activity is ongoing, but as time progresses and the device enters sleep mode the current consumption averages around 65 µA.

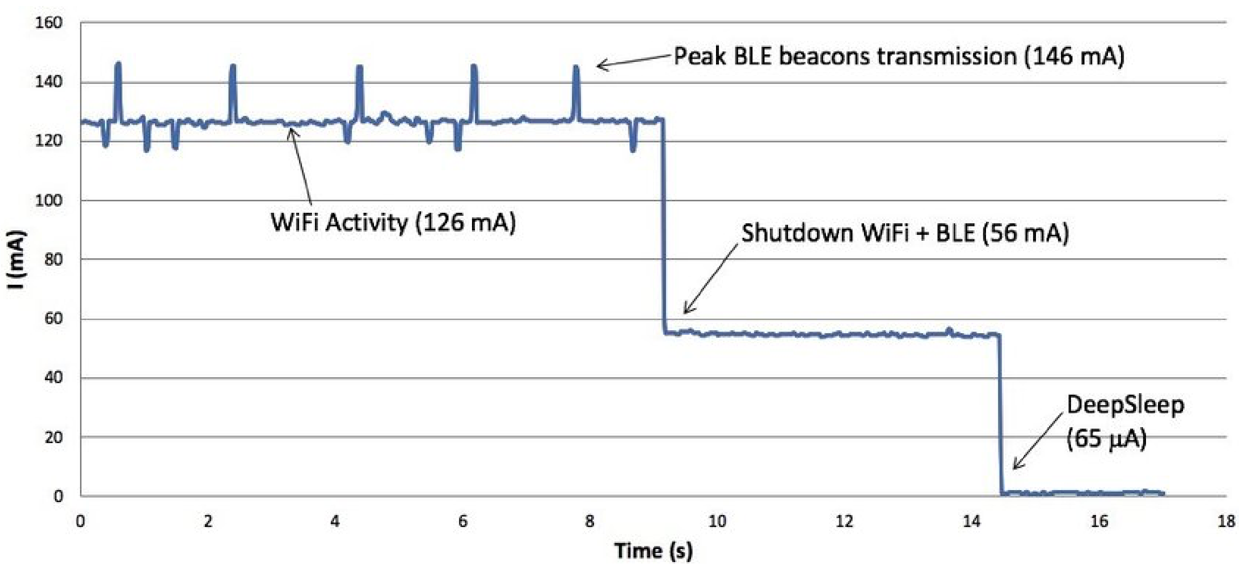

Figure 8: ESP32 Energy Consumption

We chose to use the ESP32 microcontroller to handle control operations for the toothbrush stand as the WiFi capability enables fast transmission of data collection, along with the support for the I2C protocol which our display is connected by.

Additionally, since we are going to need to keep track of real time so as to shift our calendar on the display, the microcontroller supports an RTC timer from two different internal sources: a 150kHz RC oscillator and a ~33kHz oscillator. We have elected to use the 33kHz oscillator as it provides better frequency stability than the 150kHz oscillator at the expense of a slightly higher deep sleep current consumption (by 5 µA).

Requirements	Verification
1. 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.
2. The subsystem can actively calculate time of day using an RTC and shift the calendar when a twenty-four hour period has passed.	2A. Using the standard C library in Arduino, use gettimeofday() function to output the local time in the serial monitor. 2B. Repeat step 2A after twenty-four hours have passed, validating the time matches the time of NTP. 2C. If the time displayed in the serial monitor is off by more than one minute, use the POSIX settimeofday() function to sync with SNTP.
3. The subsystem must detect and communicate with any device that is connected to the I2C addresses (7bit/10bit addressing modes).	3A. Connect an I2C capable device to GND, pins GPIO 22 and GPIO 21, and Vcc on the ESP32 I2C bus. 3B. Using Arduino IDE, run an I2C scanner sketch 3C. Verify the output in the serial monitor shows a hexadecimal address.

Table 2: Control Subsystem Requirements and Verifications

2.4 Display Subsystem

The Display Subsystem receives feedback from the Control Subsystem that brushing has begun and when brushing has stopped. It also receives internal clock output from the ESP32 board microcontroller, which it utilizes to adjust the weekly record and change states. These three inputs can be categorized as “Start_Timer, Stop_Timer, and New_Day_Shift.” Additionally, pending the results of unit testing of our ESP32 built-in RTC timer, we will be using the 33kHz oscillator to reduce power consumption during sleep mode and to shift the calendar accordingly.

The display is powered by the same +3.3V regulated voltage from the Stand Power subsystem. In the default state, where the user is not brushing, the Display Subsystem will display the current record along the bottom of the display shown in figure 8, divided into seven subsections (for the past 6 days and the current day) which are further split up into morning and evening periods.

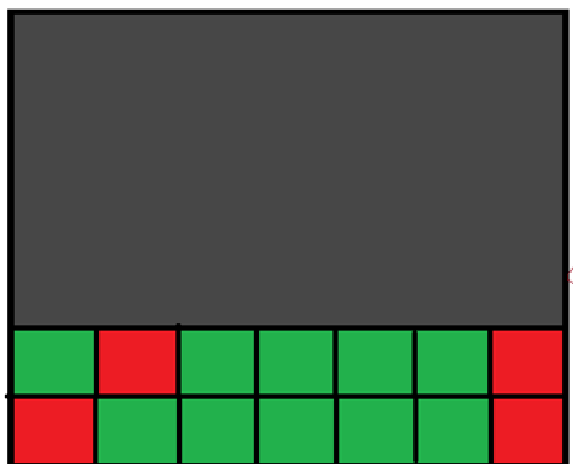

Figure 9: Default State of Display

Upon receiving a Start_Timer signal from the Control Subsystem, the Display displays a timer with large red numbers to indicate that the user has not yet brushed for the dentist recommended two minutes. After two minutes have passed, the Control Subsystem will send a signal, and the current time period will change from red to green on the Display, and the numbers on the timer will also change from red to green.

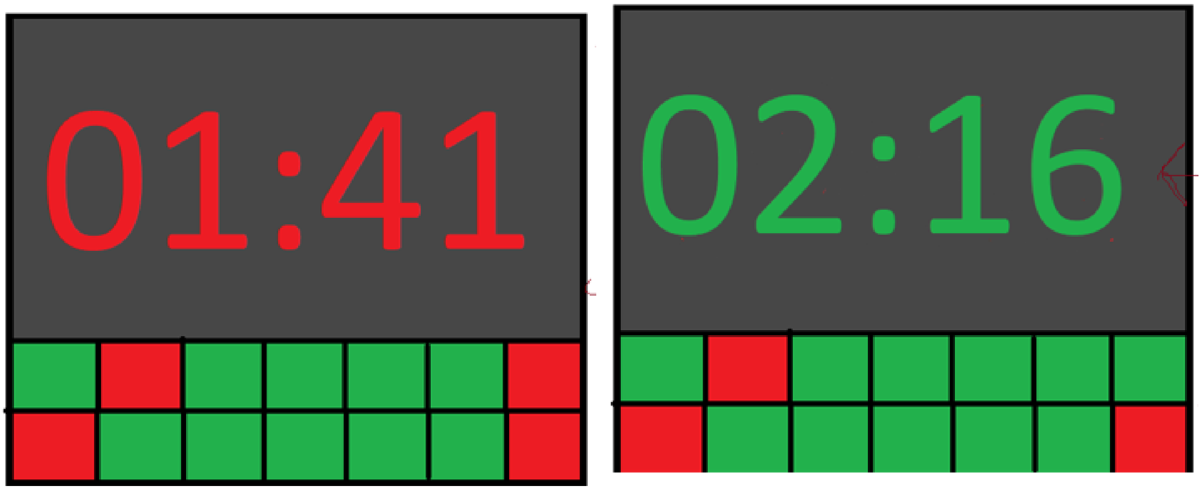

Figure 10: Transition from Underbrushing to Sufficient brushing

Upon brushing for longer than 4 minutes, the Control Subsystem will set the current record to red on the display. Finally, at 4am each day, the Control Subsystem will send a New_Day_Shift signal and shift the record on the display to the left by one day, with the new day at the right-most entry and both boxes marked red, and mark the current period as morning (upper 7 boxes). At 4pm each day, the Control Subsystem will mark the current period as evening (lower 7 boxes).

2.4.1 New Haven Display NHD-2.7-12864WDW3

Input: 3.3V±5%, display commands via I2C bus from ESP32 board MCU
Output: Process timer only when brushing is detected, one week calendar of morning and night that shifts after 24 hours

The current display we have planned for is the New Haven Display NHD-2.7-12864WDW3 from Mouser, a graphic OLED Display Module with 128 x 64 Pixel resolution over a 2.7” Diagonal Size. The display is powered with 3.3V, with a maximum voltage of 3.5V and minimum at 2.8V. Since the operating current of our ESP32 microcontroller is higher than that of the maximum “100% ON” condition of our display, we will need to use resistors to bring down the overall operating current such that we do not exceed 350mA.

Requirements	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 375mA.
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.

Table 3: Stand Display Subsystem Requirements and Verifications

2.5 Stand Power Subsystem

The Stand Power subsystem is connected to all subsystems of our overall design and has two primary functions: provide power indirectly to the Control and Display subsystem through the 3.3V linear regulator and assist the recharging of both the Lithium-Ion polymer sensor band battery and stand backup battery through the use of battery management systems.

The stand will receive its main source of power from a 5V power supply that converts the 100-120V, 60Hz AC voltage provided by the wall outlet to DC signal. This 5VDC input will be passed through to the BMS, which will be used to safely charge the backup battery. When the sensor band is docked to the toothbrush stand, the 5VDC input will be passed through to the sensor band BMS.

Additionally, the 3.3V LDO Linear Regulator inside our toothbrush stand will be receiving its source of voltage through a Diode OR system that will switch between the 5VDC power source and the 3.7V backup battery when the stand is unplugged from the wall outlet.

2.5.1 Lithium Ion Backup Battery and BMS

Input: Rechargeable 3.7V Lithium-Ion Polymer Battery
Output: 3.3V±5% for LSM6DSLTR 6-axis Gyroscope + Accelerometer
3.3V±5% for ESP32-C3-Wroom-02 Microcontroller

Similar to how the sensor band's battery is recharged, our backup battery is an additional lithium-ion polymer battery seated in the toothbrush stand at all times that is utilized to continue powering the display when the wall supply has been unplugged. The BMS will properly regulate the 5V supplied down to 4.2V such that the backup battery will be recharged when battery drain occurs. For additional reference, see sections 2.2.1 for documentation. If complete battery drain is to occur while 5V power supply is disconnected, the flash memory contained on our ESP32 board MCU will retain a copy of the calendar graphic that is restored once power is reconnected.

2.5.2 3.3V LDO Linear Regulator and Diode OR Circuit

Input: 3.7V±5% from Rechargeable Lithium-Ion Polymer battery
5.0V±5% from SWI5-5B-N-P5 Power Supply
Output: 3.3V±5% for ESP32-C3-Wroom-02 Microcontroller
3.3V±5% for New Haven Display NHD-2.7

The subsystem documentation for the 3.3V LDO Linear regulator requires an extra circuit component compared to the process described in section 2.2.1 of the sensor band subsystem. Due to the varying supply voltages of the 5V power supply and the 3.7V rechargeable backup battery, a Diode OR circuit is used to properly route the power when the power supply connected to the wall is disconnected. This ensures that we have a backup source of power that can be utilized without external assistance.

Requirements	Verification
1. Provide 3.3V±5% from a 3.7V-5.0V VDC source determined by the primary	1A. Using an oscilloscope, measure the output voltage of the linear regulator by

power connection.	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
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.

Table 4: Stand Power Subsystem Requirements and Verifications

2.6 Tolerance Analysis

An important part of our project is using acceleration data to determine when the user is brushing their teeth. We will be using the LSM6DSL inertial measurement unit from STMicroelectronics to capture the acceleration data on the sensor band. Two important considerations are how sensitive the sensor should be and how to recognize brushing activity with the acceleration data.

The accelerometer can be set to several different measurement ranges, which determines the largest acceleration that the sensor can measure. In order to read usable data, the range must be larger than the magnitude of acceleration that we expect to encounter. Although it may seem reasonable to select the largest measurement range, using a more narrow range increases the sensitivity of the sensor, so it is ideal to choose the narrowest range that can handle the largest expected acceleration.

We expect that a brushing motion will have a travel distance of less than 10cm, and the brushing frequency will be less than 8 times per second. Assuming a one-dimensional sinusoidal motion, we can model the displacement as:

$$d(t) = 5cm \times cos(2\pi \times 8s^{-1}\, t)$$

We can calculate the acceleration by taking the second derivative of the displacement:

$$a(t) = 5cm \times 4\pi^{2} \times 64s^{-2} \times cos(2\pi \times 8s^{-1}\, t)$$
$$a(t) = 126\, m/s^{2} \times cos(2\pi \times 8s^{-1}\, t)$$

We observe that the maximum expected acceleration is 126m/s^2, or 12.9*g* (where *g* is the acceleration due to gravity). The LSM6DSL has a measurement ranges of ±2*g*, ±4*g* ±8*g*, and ±16*g*, so we will use the ±16*g* setting for our project.

Concerning the detection of whether the user is brushing their teeth, we expect that the magnitude of acceleration will provide sufficient data for identifying brushing activity. However, we will need real-life tests to determine the threshold magnitude and to confirm whether the acceleration magnitude provides sufficient information on its own. We will also implement a buffer time of one or two seconds in order to prevent accidentally miscategorizing non-brushing activity as brushing activity.

In the event that the above method results in insufficient accuracy, we will model the acceleration data as an elliptical path and analyze the frequency, shape, and orientation. We will detect active brushing when these factors, along with how they change over time, reach conditions that are characteristic of active brushing. The exact conditions will have to be determined through real-life tests.

3 Cost and Schedule

Cost Analysis

Labor			Cost:	Count	Total Cost:
		Computer Engineer:	$14,000.00	2	$28,000.00
		Electrical Engineer:	$11,000.00	1	$11,000.00
Parts					
Type	**Manufacturer**	**Part Number**			
Display	Newhaven Display	NHD-2.7-12864WDW3	$39.22	1	$39.22
Microcontroller	Espressif	ESP32-C3-Wroom-02	$2.15	2	$4.30
Sensor	STMicroelectronics	LSM6DSLTR	$5.70	1	$5.70
5V Power Supply	CUI	SWI5-5B-N-P5	$20.72	1	$6.63
Li-Ion Batteries	Sparkfun	PRT-12895	$5.95	2	$11.90
BMS/Charger	Microchip	MCP73831T-2ATI/OT	$0.69	2	$1.38
3.3V Regulator	Analog Devices	LT1963AES8-3.3	$5.87	2	$11.74
PCB	PCBWay		$5.00	1	$5.00
Toothbrush	Oral-B	Pro Crossaction 1000	$49.99	1	$49.99
				Sum of Costs:	**$39,135.86**

Table 5: Parts + Labor Cost Analysis

Calculations:

According to data from the 2018-2019 graduating class, the average starting salary of a UIUC graduate is $96,992 for computer engineers and $79,714 for electrical engineers. Assuming an engineer works 2080 hours per year, a computer engineer earns $46.63 per hour and an electrical engineer earns $38.32 per hour. Our project is 12 weeks in length, and we expect to spend about 10 hours of time each week. We will also apply a 2.5 overhead multiplier in order to account for business costs, which encompasses salaries for other workers in the company. This would result in a total labor cost of:

$$\frac{\$96{,}992}{1\ year} \times \frac{1\ year}{2080\ hours} \times \frac{10\ hours}{1\ week} \times 12\ weeks \times 2.5\ =\ \$14,000\ (Computer\ Engineer)$$

$$\frac{\$79{,}714}{1\ year} \times \frac{1\ year}{2080\ hours} \times \frac{10\ hours}{1\ week} \times 12\ weeks \times 2.5\ =\ \$12,000\ (Electrical\ Engineer)$$

The overall cost of project design would be reduced if all components were ordered in bulk quantities; however, for the scope of the project the prices have been listed at their current purchase price.

Schedule

Week	Rahul Vasanth	Quinn Palanca	John Kim
2/28/22	Work on Machine Shop design; prepare design review docs.	Assist with PCB design; programming the microcontroller WiFi comm.	Work on PCB Design; order remaining components
3/7/22	Finalize machine shop order; Gather accelerometer/gyrometer sensor data	Programming the microcontrollers WiFi comm. and validate data transmission	PCB design order (round 1), gather accelerometer/gyrometer sensor data
3/14/22	**SPRING BREAK**		
3/21/22	Development and testing of toothbrush stand base display subsystem	Programming accelerometer data analysis algorithm and complete diagnostics	2nd revision for PCB audit/design; power subsystem unit testing and debugging
3/28/22	Assembly of toothbrush stand subsystems (power, control, and display)	Programming accelerometer data analysis algorithm and complete diagnostics	PCB assembly, mounting, and soldering
4/4/22	Finalize accelerometer data model and algorithm	Finalize accelerometer data model and algorithm	Sensor Band battery and backup battery charging validation

4/11/22	Full system testing and debugging	Full system testing and debugging	Full system testing and debugging
4/18/22	Mock Demo	Mock Demo	Mock Demo
4/25/22	Demonstration; prepare Presentation materials	Demonstration; start Final Paper	Demonstration; start Final Paper
5/2/22	Presentation and Final Paper	Presentation and Final Paper	Presentation and Final Paper

Table 6: Project Schedule and Task Allocation

4 Safety & Ethics

Privacy is an issue of great prominence and plays a central role in our design. According to sections 1.3, 1.6, and 1.7 of the ACM Code of Ethics and Professional Conduct, focusing on Privacy, “Only the minimum amount of personal information necessary should be collected in a system” [7]. Our system is optimized for user privacy, as the bare minimum data is stored for at most one week in a manner that is clear to the user and in a manner that cannot be exported or stored. The Habit-forming Toothbrush Stand preserves user privacy, a fundamental human right in a manner with the knowledge of the people affected, for legitimate ends. Sensitive user information will not be stored on-device in order to minimize data collection and prevent unauthorized access to personal data. Additionally, having a method to “reset” the toothbrush stand and the display is worth exploration for individuals in shared environments.

Section I-1 of the IEEE Code of Ethics emphasizes that striving to comply with ethical design and protecting the privacy of others is paramount [8] . Additionally, section II of the IEEE Code [8] states that it is essential to “to treat all persons fairly and with respect, to avoid harassment or discrimination, and to avoid injuring others.” Since our device uses a display we must ensure that there are no effects that could lead to photosensitive epileptic seizures for vulnerable users. Our design does not include specific animations or effects, we will only be using the display to show the calendar and timer. We have explored the possibility of different colors or unique identifiers in our numbers so that colorblind users can utilize our device with minimal disruption.

The sensor band could potentially be a choking hazard for children, so our device must be utilized by children above four. Additionally, the battery that powers our sensor band is not rechargeable so battery disposal must be done responsibly. To ensure we properly use and dispose of batteries safely, we will abide by the safety standards of the Division of Research Safety [9] at the University. Electrical hazards and improper battery or power management could pose safety risks which we will account for through our project design. In accordance with Section I-5 of the IEEE Code of Ethics [8] we will accept and implement feedback from teaching, assistants, Professors, and Machine Shop staff.

Additionally, our system comes in contact with water, therefore we need to take precautions about water damaging our circuits, or worst case potentially coming into contact with the power outlet. Bathroom sinks and outlets are already designed in a manner that would not require additional measures, however, addressing the risks of water entering the toothbrush stand or the sensor band is important. Machine shop staff recommended the following: sealant over the circuits/sensors, a waterproof plastic covering, or having everything sealed which would prevent water from damaging our system.

References

[1] C. Weller, "Silicon Valley parents are raising their kids tech-free — and it should be a red flag," *businessinsider.com*, Feb. 18 2018. [Online]. Available: https://www.businessinsider.com/silicon-valley-parents-raising-their-kids-tech-free-red-flag-2018-2?op=1 [Accessed: Feb. 10, 2022].

[2] M. Richtel, "A Silicon Valley School That Doesn't Compute," *businessinsider.com*, Oct. 22 2011. [Online]. Available: https://www.nytimes.com/2011/10/23/technology/at-waldorf-school-in-silicon-valley-technology-can-wait.html [Accessed: Feb. 10, 2022].

[3] T. J. Salinas, "When and how often should you brush your teeth?," *Mayoclinic.org*, Oct. 7 2021. [Online]. Available: https://www.mayoclinic.org/healthy-lifestyle/adult-health/expert-answers/brushing-your-teeth/faq-20058193 [Accessed: Feb. 10, 2022].

[4] ESP32S3 Series, Espressif, Jan. 2022. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf

[5] R. Santos and S. Santos, "ESP-NOW Two-Way Connection Between ESP32 Boards," *randomnerdtutorials.com*, Jan. 2020. [Online]. Available: https://randomnerdtutorials.com/esp-now-two-way-communication-esp32/ [Accessed: Feb. 10, 2022].

[6] L. Bank, "How much current do OLED displays use?," *bitbanksoftware.blogspot.com*, June 2019. [Online]. Available: https://bitbanksoftware.blogspot.com/2019/06/how-much-current-do-oled-displays-use.html [Accessed: Feb. 10, 2022].

[7] Association for Computing Machinery, "ACM Code of Ethics and Professional Conduct," *Association for Computing Machinery*, 2018. [Online]. Available: https://www.acm.org/code-of-ethics [Accessed: Feb. 10, 2022].

[8] IEEE Board of Directors, "IEEE Code of Ethics," *Institute of Electrical and Electronics Engineers*, 2020. [Online]. Available: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf [Accessed: Feb. 10, 2022].

[9] J. Richards, “Battery Safety,” *Division of Research Safety*, 2020. [Online]. Available: https://drs.illinois.edu/Page/SafetyLibrary/BatterySafety [Accessed: Feb. 10, 2022].