Event Attendance Tracker

By

Anand Sunderrajan (anands3)

Eric Layne (ealayne2)

Mason Edwards (masonae2)

Final Report for ECE 445, Senior Design, Fall 2020

TA: Dhruv Mathur

09 December 2020

Project No. 13

Abstract

We designed, built, and tested an attendance system to automatically record what booths a user has stopped at. This system would subsequently provide information about those booths, and update attendance statistics available to the booth presenters in real-time. The system operates using a “booth node” at each booth, in conjunction with either a smartphone application or a hardware device utilized by the event attendee. These two devices communicate over Bluetooth Low Energy (BLE) to determine which booth a user is present at, as well as for exchanging information about the booth and the user.

The system uses a custom algorithm for detecting attendance at booths based on the Received Signal Strength Indicator (RSSI) for the BLE connection with each nearby booth. This algorithm is successful, as it correctly identifies the booth a user is attending in the presence of multiple booths at similar distances for an event such as a career fair or the Consumer Electronics Show (CES).

Contents

1 Introduction

Every year, there are a multitude of large events globally wherein there are hundreds of presenters (i.e. exhibition booths) and thousands of attendees. Attendees attempt to remember information about each booth, and create a significant amount of waste from brochures and pamphlets given to them, that end up being thrown out shortly after the event. CES (Consumer Electronics Show) is a prime example of such an event; CES 2019 attracted 175,212 verified attendees compared to its 4,500 exhibitors [1].

To solve these issues, we created the Event Attendance Tracker. A system which provides an alternative solution for attendees to view nearby booths at an event and accurately track the booths they have visited, while simultaneously reducing the overall amount of waste generated from such events. Our system would further enable them access to information about the booths even after the event, which could include contact information to communicate with the presenters in a less hectic environment - all without needing to remember the name of every booth they stopped at. Through utilizing a self contained design, our system provides a low cost alternative to current market solutions which involve subscription based models [2]. Presenters are given a physical device to place at their booth which allows them to view the number of visitors to their booth, while event attendees are given a choice of either a physical or a software solution making our system highly accessible. Through utilizing BLE, we reduce the overall power consumption of our system, thereby making it a more efficient alternative to current heat mapping solutions present in the market [3].

Our system has several high level requirements which were met, ensuring the functionality of our device.

1. The booth node's battery needs to provide sufficient power for at least four hours of normal operation per charge.
2. The attendee device must log booths which are within three meters from the user's physical device or smartphone application, provided they meet the minimum time requirement for the booth. Additionally, they must be at least 6 meters away from any booth nodes present at other booths [4].
3. The transfer of data between the smartphone and booth node should be completed in under $250mS$ after initial connection. This is to ensure the typical 4 attendees [5] interacting with a booth presenter can be serviced by the booth node each second.

The Event Attendance Tracker consists of two physical components as well as a smartphone application. The first physical device is the booth node used by the presenters, seen in Figure 1, and is to be placed at each of the booths. The second is a flat card-like device, shown in Figure 2, which is used by the event attendees and could easily fit in their pockets. Both devices consist of a power button and a mini-B USB port for charging and data transfer.

The booth node and attendee device utilize the same hardware, as described in the left of Figure 3. This hardware contains three subsystems: the control unit, power supply, and I/O subsystem.

The control unit handles communication between the booth node and the attendee device over BLE. It also interacts with the I/O subsystem over a USB differential pair and an I^2C bus. This subsystem also takes an analog input from the power supply for measuring the battery voltage.

The I/O subsystem contains the USB port, power switch, and OLED display allowing the booth presenter to power, control, and monitor the device. This subsystem takes the battery voltage and USB port voltage supply as input, and outputs power to the battery charger and $3.3V$ regulator based on the switch position.

The power supply takes the $5V$ USB power input and uses it to charge the battery cell when it is available, while connecting the main power (ranging from $3.6V$ to $5.25V$) to the $3.3V$ regulator input, and using the enable pin of the regulator to only allow the device to operate at safe input voltages. This subsystem then outputs the regulated $3.3V$ to all other subsystems to power their respective devices.

The BLE communication between the booth device, and attendee device is used to both determine the attendance at a booth, and transfer relevant booth and user information between the modules.

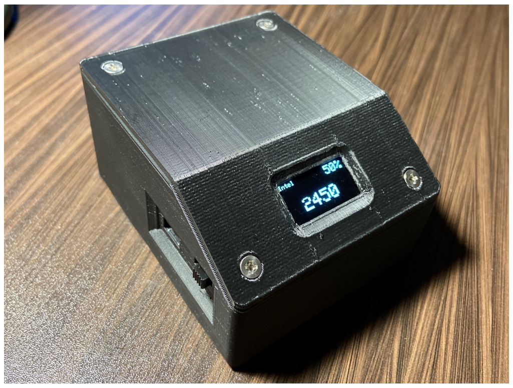


Figure 1: Assembled booth node

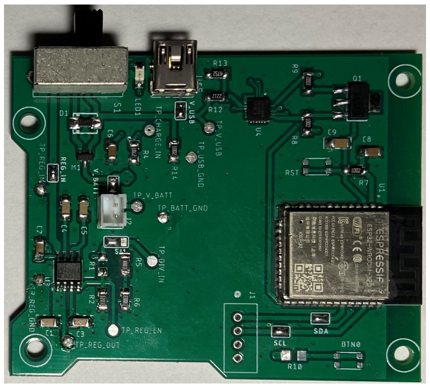


Figure 2: Attendee device

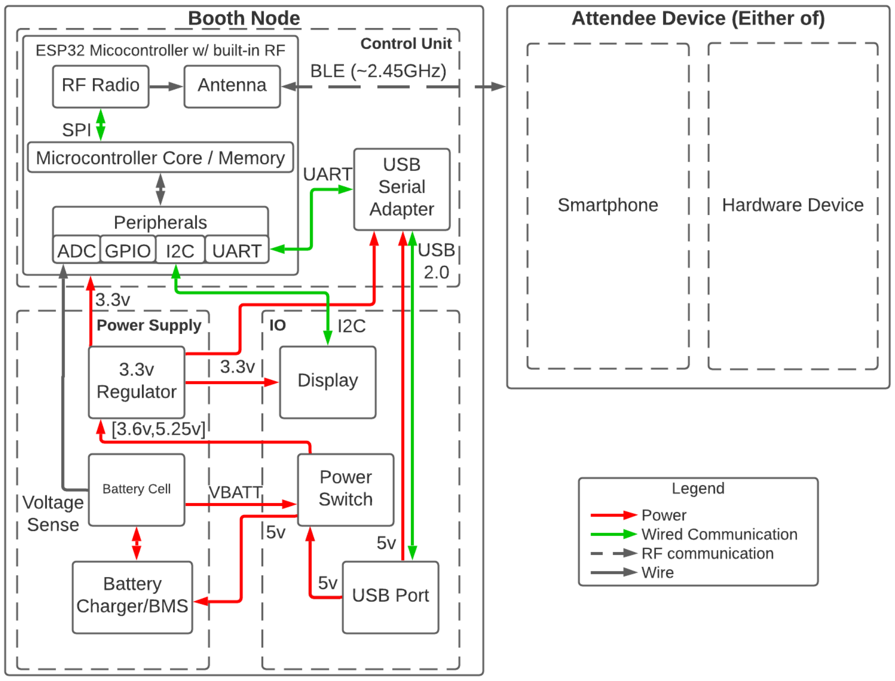


Figure 3: Event attendance tracker block Diagram

2 Design

The full schematic, PCB layout, code repositories/pseudo-code, and the full requirements and verification table can all be found in the Appendix in the aforementioned order.

2.1 Design Alternatives

During the design process of the Attendee Device, we encountered multiple roadblocks that led to modifications to our original design. Some of the more notable issues and their respective modifications were:

1. The changes made to the decision making equation due to the original equation choice being infeasible.
2. Changes to library usage within the smartphone application to make it compatible across different Android OS versions. Some examples of such include:
   - We were unable to use .replace and had to use .put to make it compatible with OS versions below 10.0. Despite having the same functionality for hash-maps (to keep track of nearby and attended booths and update their RSSI values), we would get unexpected errors/functionality upon using .replace and attempting to run the application on older smartphones.
   - We used an enhanced for loop to iterate through the hash-map and remove duplicate entries - thereby enabling us to constantly update the RSSI values - when building for Android 6.0. However, this method brought forth error messages and crashes on later OS versions, and as such we had to modify the iteration to be a normal for loop over the size of the hash-map.
   - While using the "list view" adapter type functioned as expected, it would add booths past the boundaries of the object and overflow, thereby causing booth information to show up on top of buttons, or even not show up entirely. As such, we had to move to a "scroll view" adapter which further required modifications of the entire UI layout for it to function correctly.
3. Building the application on Android 10.0 did not make it backwards compatible with Android 6.0, and alternatively building it on Android 6.0 did not make it forwards compatible. This was consequent to the variance in system permissions required to access the device Bluetooth hardware. Additionally, our system utilized BLE, which required specific hardware to be available on the smartphone [6]. As such we had to implement different checks based on the Android OS version to:
   - Ensure that a supported Android OS version was installed.
   - Ensure that the necessary hardware was present on the device.
4. Initially we utilized a verified MAC address and device name approach to filter out booth nodes from available Bluetooth devices. This was done using the base Bluetooth class. However, we had to modify our entire implementation to one that utilized BluetoothGATT class to enable communication between the booth node and smartphone [7]. This was necessitated consequent to the system requiring the booth node to count attendees that were marked as visited. Therefore, we subsequently moved from a MAC address filter to a UUID filter.
5. The use of a timer loop to continuously scan resulted in high resource consumption and the application constantly crashing. As such, we moved to a "Runnable" class/thread that would automatically execute

the scan functionality at a user configurable interval [8]. This consequently reduced our overall resource consumption to that shown in Figure 4.

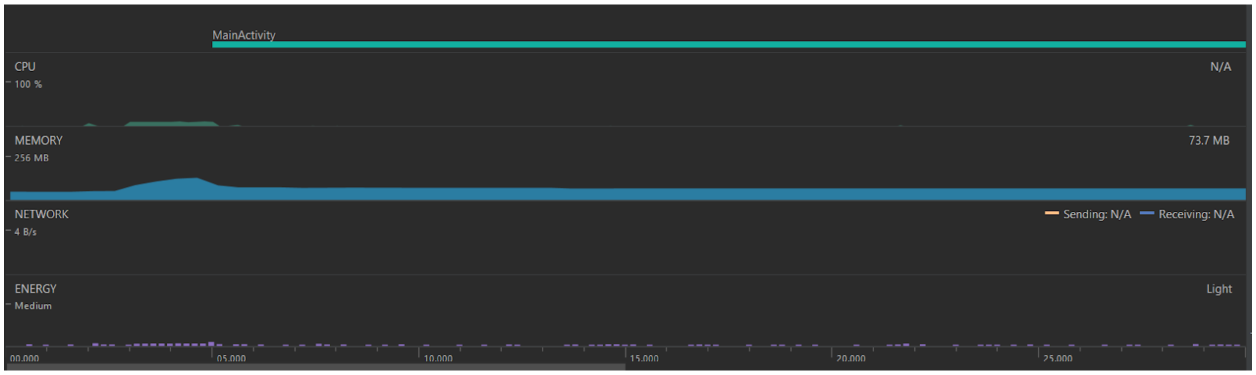


Figure 4: Smartphone application resource utilization

2.2 Power Supply

There were multiple design decisions made within the power supply unit; deciding the type and implementation of the 3.3V regulator and the choice of lithium polymer (Li-Po) battery cells. We chose a linear $3.3V$ low dropout (LDO) regulator over traditional linear regulators or switching regulators because of the low power requirements of our device, the reduced noise of linear regulators [9], and a lower voltage our device can operate at (increasing usable battery capacity). We chose to use Li-Po batteries because of the wide variety of manufactured shapes, sizes, and capacities available when compared to more standardized Lithium-ion or Nickel Metal Hydride (NiMH) cells.

Equation 1 shows describes the relationship between current consumption and battery capacity as a function of time.

$$C = I * t \tag{1}$$

where:

C = Capacity consumed [Ah]
I = Constant current draw [A]
t = Time [h]

Calculating the minimum required battery capacity using Equation 1, we found that $840mAh$ was the minimum given our requirement of 4 hours run-time and the measured maximum continuous current draw of $210mA$.

The regulator enable voltage was set based on Equation 2 which is replicated from the datasheet [10].

$$R_{up} = R_{down} * \frac{V_{en} - 1.22V}{1.22V} \tag{2}$$

where:

R_{up} = Resistance of high side resistor in voltage divider [Ω]
R_{down} = Resistance of low side resistor in voltage divider [Ω]
V_{en} = Desired enable voltage [V]

Using Equation 2, we found that our desired enable voltage should be $3.816V$ (to guarantee shutoff below input $3.6V + 0.1V$). For this, we would use $R_{up} = 100K\Omega$ (R5 in Figure 5) and $R_{down} = 47K\Omega$ (R6 in Figure 5).

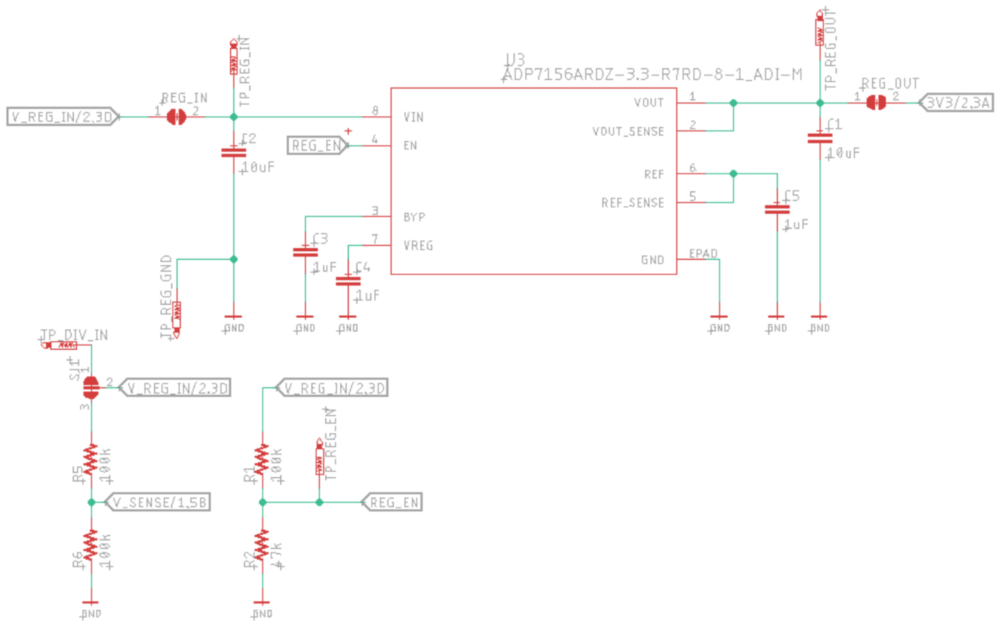


Figure 5: Regulator schematic snippet

Equation 3, found on the device datasheet [11], describes the relation between the programming resistor to the maximum charge current, which we set according to a maximum of $0.5C$, which resulted in a choice of $2.5K\Omega$ for a $400mA$ current limit.

$$I_{chg} = \frac{1000V}{R_{prog}} \tag{3}$$

where:

I_{chg} = Maximum charge current [A]
R_{prog} = Resistance of programming resistor [kΩ]

2.3 Control Unit

The design of the control unit was driven largely by the decision to use BLE for booth detection and communication. We considered using WiFi or a custom RF protocol for this, but BLE has the benefits of having lower transmit and quiescent power than WiFi [12]. We chose BLE over a custom RF solution due to complexity, and the architecture of BLE is beneficial for the simplicity of interacting with a wide variety of devices - from smartphones to simple Bluetooth modules. As a result of this decision, we chose to use an ESP32 microcontroller as it has inbuilt BLE support, and has a variety of peripherals (ADCs, hardware I^{2}C, and hardware UART) needed to meet other requirements [13].

Equation 4 was found by finding the linear regression of a collection of points defined by 10-bit ADC values correlating to input voltages, and scaled by a factor of two to find the actual battery voltage based on the ADC measurement of the control unit.

$$V_{sense} = 2.0 * \frac{U_{adc} + 173.5}{1240.4} \tag{4}$$

where:

V_{sense} = Measured battery voltage [V]
U_{adc} = 10-bit ADC value [full scale unsigned from 0.0V to 3.3V]

Seen below in Figure 6 is the reset circuitry for the ESP32 microcontroller. It is designed to allow the hardware to be reset by the button (designator KMR2), and contains the required pull-up resistor to enable the device paired with a bypass capacitor to avoid erroneous resetting. The figure also contains the auto-reset circuitry which takes the flow control lines of the serial communication and automatically resets the ESP32 and boots into the programming mode when specific control signals are sent (while ignoring other flow control signals) [14].

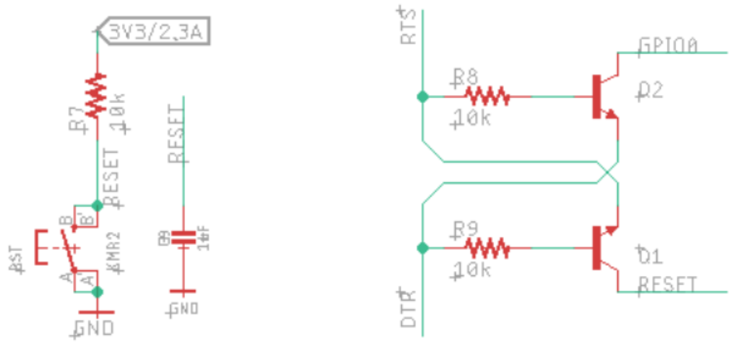


Figure 6: ESP32 reset/programming schematic snippet

2.4 I/O Subsystem

This subsystem includes an OLED display for presenting information to the booth presenters. The decision to use OLED over technologies like LCD was made based on the lower power consumption in our use case and wider viewing angles [15]. As OLED displays only consume power for the pixels which are lit, our use where most of the screen is off and only text is displayed means we have power savings as compared to powering the back-light of an LCD display. Further, the wider viewing angles allows booth presenters to easily read the display while sitting or standing at the booth.

The simulations shown in Figures 7 and 8 show the behavior of the switch circuit schematic represented in Figure 9. This circuit stops the PMOS from conducting, effectively disconnecting the battery from the regulator input when USB voltage is present. When the USB voltage is not present, the pull-down resistor allows the PMOS to conduct and the battery to provide power to the regulator.

2.5 Attendee Device

The design of the attendee devices were driven by the decision to provide a highly accessible system. Consequently, this module presented both a hardware and software solution to the event attendee. The design for

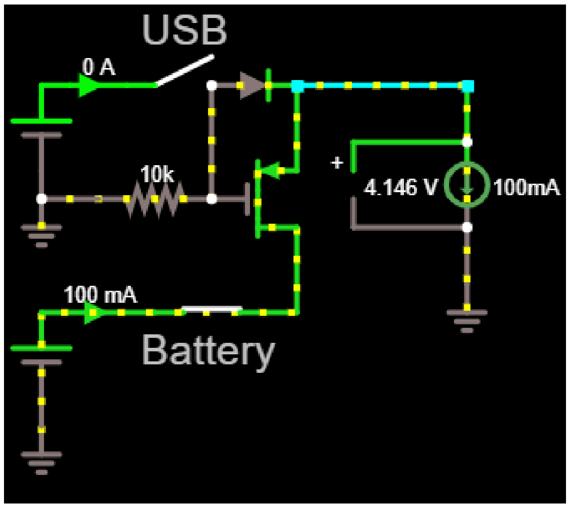


Figure 7: Simulation with only battery power available

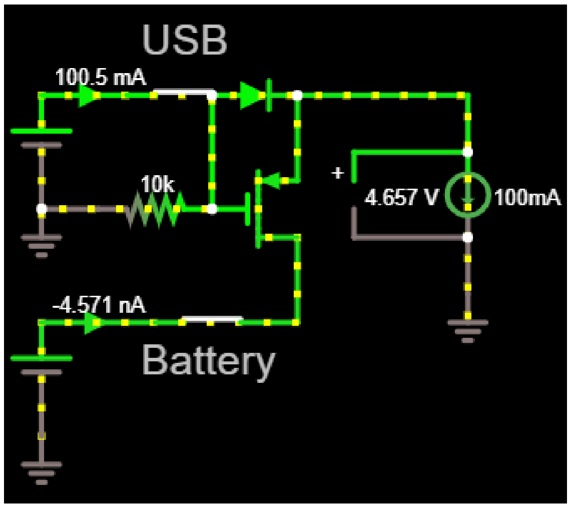


Figure 8: Simulation with USB power and battery power

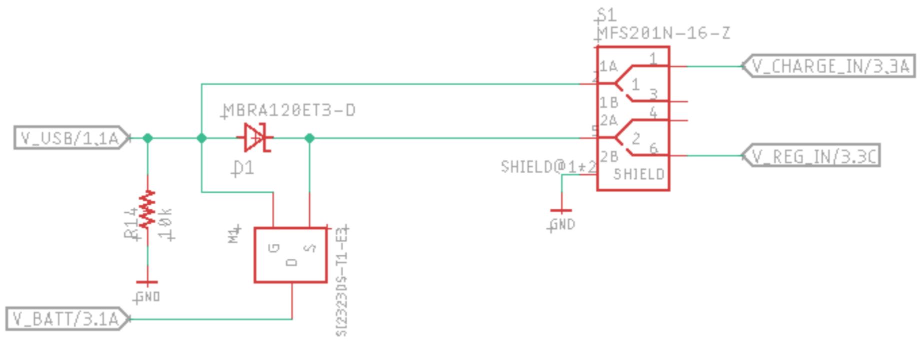


Figure 9: Power switch schematic snippet

the hardware solution utilized the same PCB as that of a booth node, albeit flashed with different firmware. For the software solution, we chose a smartphone application due to the convenience it would present for the user.

2.5.1 Physical Device

The firmware for the physical device was built using the C++ programming language with the Arduino framework available for the ESP32 microcontroller. This device did not have any major decisions made, as it was nearly entirely driven by the choices previously made for the booth node hardware as they share the same PCB and hardware management code. The only difference between the firmware is the attendee device runs the booth detection algorithm explained in section 2.5.4 rather than advertising as a booth.

2.5.2 Smartphone Application

The smartphone application was built using the Java programming language for Android OS. We chose to develop an application for Android OS due to restrictions posed on iOS application development. We were unable to gain access to a MacOS device which is necessary for developing an iOS application [16]. To develop the application for Android OS, we utilized Android Studio. We chose Android Studio for a few main reasons:

1. It is the official IDE for Android development.
2. It provides virtual device emulators to run applications on.
3. It contains an inbuilt resource monitor which provides graphs for CPU, memory, network, and power consumption.
4. It displays the application XML layout (User Interface) in real-time based on the XML code we write.
5. The ability to utilize custom made vector images for application icons.

Unfortunately, Android Studio does have some drawbacks. The most important drawbacks with regards to this project were:

1. The fragmentation of available libraries and functions across API/OS versions.
2. The multitude of device permissions required to build an application that utilizes Bluetooth.

2.5.3 Decision Making Equation

Our system utilizes nearby booths' RSSI values to determine which booth is closest to the attendee device. We initially planned to utilize Equation 5 [17] to assign an estimated distance value to each booth, and subsequently process the data to determine whether a booth had been attended or not.

$$d = 10^{\frac{A-rssi}{10n}} \tag{5}$$

where:

d	= Estimated distance [m]
A	= Transmission power of source device (booth) [dBm]
$rssi$	= RSSI of particular device [dBm]
n	= Unit-less path attenuation factor

However, upon attempting to implement this equation, we encountered two fundamental roadblocks:

1. Android OS does not allow us to access the "A" value. Furthermore, no external libraries with this functionality exist during our design process.
2. The path attenuation factor would be highly dependent on environmental factors (such as line of sight, interference, transmission power, etc.) and there was no feasible method to get a representative value which would work for all use cases.

As such, we transitioned to our own determination formula shown in Equation 6.

$$score = (1-\alpha) * score + \alpha * \frac{rssi - avg}{std\_dev} \tag{6}$$

where:

$score$	=	Score describing algorithm confidence in order of closest booths
α	=	Unit-less constant controlling time constant of exponential filter
$rssi$	=	RSSI of this particular booth whose score is being updated [dBm]
avg	=	Average RSSI of all booths found in the last scan [dBm]
std_dev	=	Sample standard deviation of RSSIs for all known booths [dBm]

The performance of our algorithm in a multiple booth node environment is shown in Figure 10. As evident in the graph, the algorithm performs as expected and marks device 3 as the closest over a one minute time period with a scan interval of 5s - leading to 12 update indices before it would be marked as attended. Despite other devices having higher RSSI values at update indices 6 and 9, our algorithm marks device 3 as the closest at that index as it takes into consideration the history of the nearest estimation, i.e. it considers how booth signals strengths compare to all others at each individual update, and keeps a sum of the current and past statistical Z-Scores for each to calculate the score for each booth. The ordering of booths by increasing scores is the algorithm's estimation of the order of booths from farthest to closest.

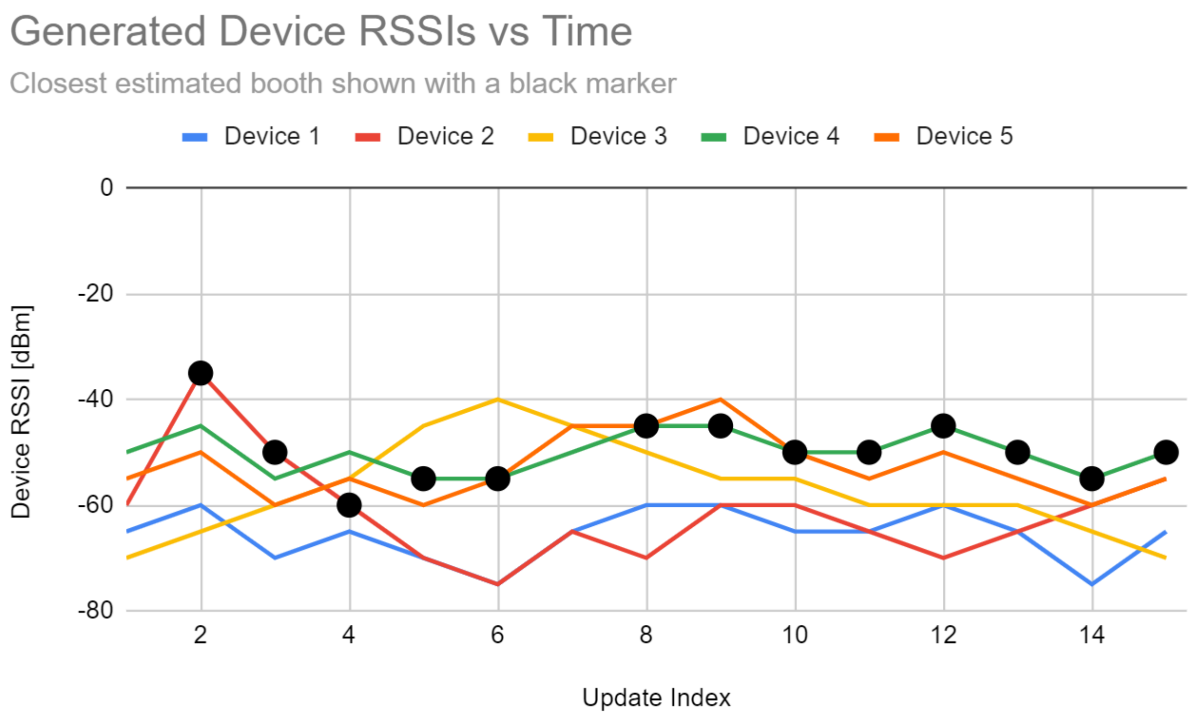


Figure 10: Decision making analysis graph

2.5.4 Algorithm Flowchart

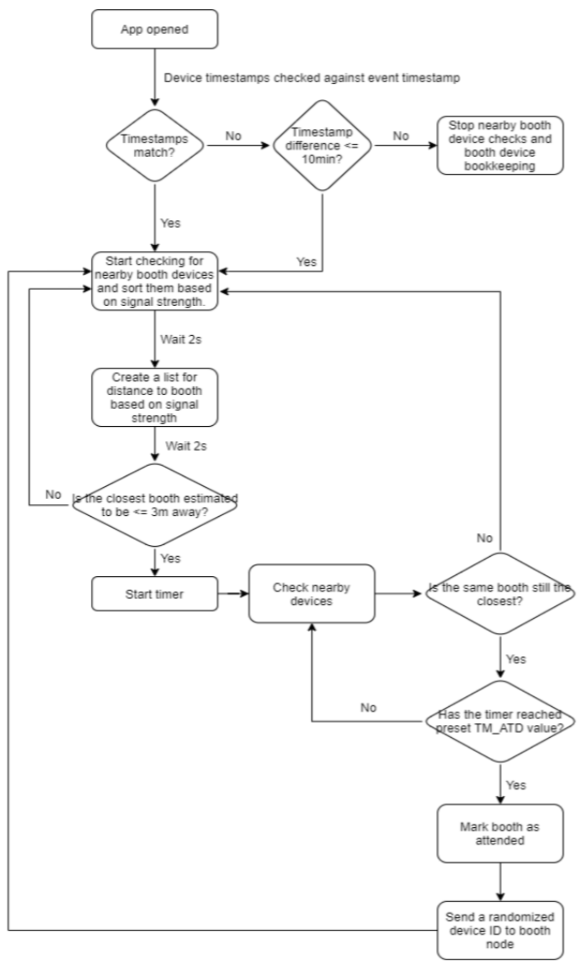

App opened
Device timestamps checked against event timestamp
Timestamps match?
No
Timestamp difference <= 10min?
No
Stop nearby booth device checks and booth device bookkeeping
Yes
Yes
Start checking for nearby booth devices and sort them based on signal strength.
Wait 2s
Create a list for distance to booth based on signal strength
Wait 2s
No
Is the closest booth estimated to be <= 3m away?
Yes
Start timer
Check nearby devices
No
Is the same booth still the closest?
Yes
No
Has the timer reached preset TM_ATD value?
Yes
Mark booth as attended
Send a randomized device ID to booth node

Figure 11: Old algorithm which utilized distance conversion

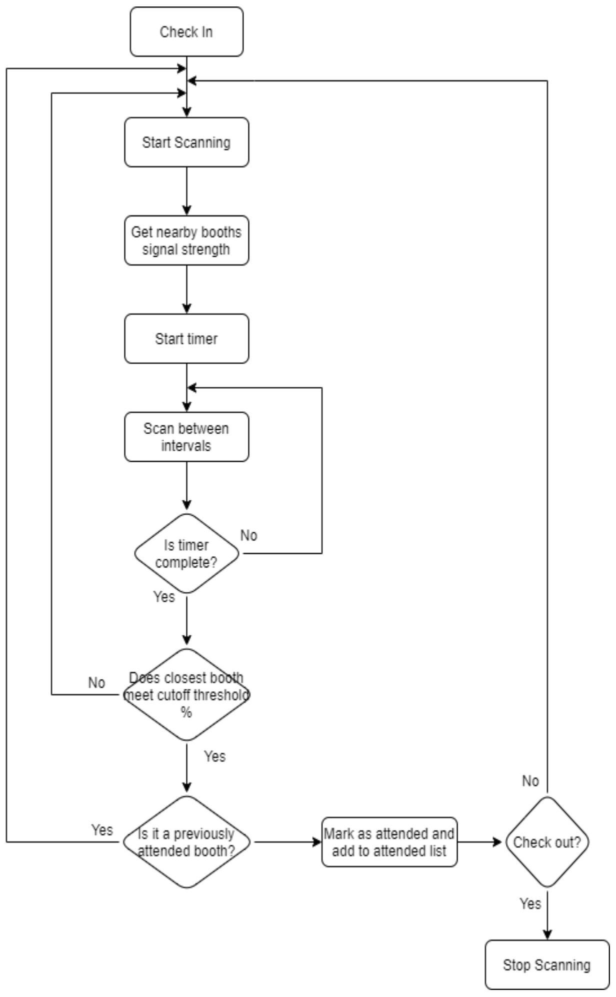

Check In
Start Scanning
Get nearby booths signal strength
Start timer
Scan between intervals
Is timer complete?
No
Yes
No
Does closest booth meet cutoff threshold %
Yes
Yes
Is it a previously attended booth?
Mark as attended and add to attended list
No
Check out?
Yes
Stop Scanning

Figure 12: New algorithm based on RSSI cutoff threshold over time

Figure 11 depicts the logical flowchart of the original algorithm we planned to use for developing our software, while Figure 12 shows the logical flowchart of the new algorithm we designed to combat the design issues we faced upon utilizing the original algorithm. Additionally, we transitioned from a "device and event timestamp comparison" process to a simplified "check-in/check-out" process to manage resource consumption with constant scanning due to the limitations posed by Android Studio with requiring additional permissions to access data from the device itself. The algorithm is controlled by three user configurable parameters - A time threshold for being marked as attended, a scan interval period, and a certainty threshold to describe how sure the algorithm must be to mark a booth as attended.

3 Design Verification

3.1 Power Supply

1. To verify the power supply can provide sufficient current, we provided $4.2V$ to the input of the regulator (simulating a fully charged Li-Po battery) and drew $750mA$ from the regulated $3.3V$ output while measuring the output voltage. This test showed the regulator voltage dropped to $3.295V$, well within the $\pm 0.1V$ tolerance.

2. Extending the previous load test, we verified it can provide this current at both the minimum operating voltage of $3.6V$ (Li-Po minimum operating voltage) and at the maximum of $5.25V$ (USB maximum operating voltage). The results showed that it maintained the voltage within the tolerance again; for the $3.6V$ input voltage the output voltage was $3.299V$, while for the $5.25V$ input the output was $3.289V$.

3. Our power supply also has to safely charge and maintain the Li-Po battery, ensuring it is charged within acceptable current levels and does not exceed the maximum voltage. Providing $5.0V$ to the subsystem, the current consumed by and voltage across the battery cell were monitored. Across a full charge cycle, the maximum current was $374mA$ and maximum voltage was recorded as $4.21V$; both measurements are well within the tolerance and should not cause issues with the battery cell over its lifetime.

3.2 Control Unit

1. We verified our device could be communicated with over USB for configuration and offloading data about attendance; this was done by connecting the device to multiple computers over USB and checking it shows up as a valid COM/tty device without driver errors. This verification was successful, as all three devices are able to communicate reliably over USB with two computers it was verified on.

2. The I^2C bus must be functional in order for the display to work so we can show information to the booth presenters. We verified this by connecting the I^2C bus to a test slave device and verified the control unit can communicate with the test device on the I^2C address needed for the display ($0x3D$), and another test address ($0x43$). The subsystem was able to detect and communicate with both addresses without issues.

3. To accurately estimate the battery capacity remaining, the ADC of the control unit needs to estimate the battery voltage within $\pm 0.035V$ of the actual voltage across the battery voltage range from $3.5V$ to $4.2V$. This was verified by measuring the voltage of the battery cell at the same time as recording the 10-bit ADC value corresponding to that voltage. Repeating this across the full battery voltage range, we then calculated the maximum deviation between the actual voltage and the estimated voltage based on the ADC value. This test passed, with a maximum deviation of $23mV$ across the battery input range.

4. The control unit must be detected as a BLE device when a scanning device is within $5m \pm 0.1m$ to ensure we can detect nearby booths. This was tested by advertising the control unit as a BLE device, and using a smartphone scanning application to detect the BLE device at various distances within the required range. This verification also passed, as the device was detected in each scan.

5. To service enough attendees at a booth to avoid starving a device of processing time, the control unit needs to transfer $4kB$ of data within one second over BLE (representative of sending $1kB$ of booth information to four attendees per second). This was tested by allowing the control unit to transmit $4kB$ of random data to a BLE serial terminal, and recording the time used (or a failing result, if not all data was sent within the time limit). Our device passed, as it was able to transfer this data without error within $115mS \pm 15mS$.

3.3 I/O Subsystem

1. In order to display information to the booth presenters, the I^{2}C bus connected to the display must all be functional. To verify, a test program was uploaded to toggle every pixel of the display individually and visually confirm the pixels were toggling as expected. This was the case, and our device was able to control each pixel individually without issue.

2. To ensure the safety of the Li-Po battery, the power switch circuit must not allow more than $\pm 0.1mA$ to flow into the BMS when the device is in "active" mode. This ensures that the battery current is not allowed to flow into the battery charger (creating a cycle which continually discharges the battery). This was successfully verified, as the leakage was measured to be $0.0mA$ (due to the $0.1mA$ accuracy of our DMM on the $0 - 10mA$ range).

3. To maximize the battery run-time of the hardware, when USB power is available, no more than $\pm 0.1mA$ is allowed to flow out of the battery voltage output. This was also successfully verified, and was measured to be $0.0mA$ (due to low current measurement accuracy limitations).

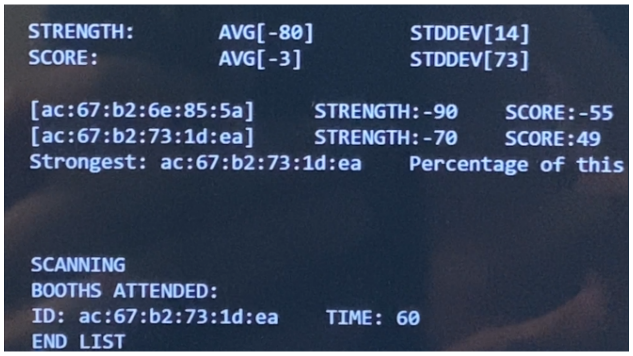
4. The USB port must provide sufficient current to charge the battery cell or to power the device (while in "active" mode), both without exceeding the maximum temperature of any component of the hardware. This test was completed by supplying USB power to the USB port and allowing the device to charge (verified by a DMM to be at the maximum current of $400mA$) for 15 minutes, and verifying all components stayed within their individual acceptable temperature ranges.

3.4 Event Attendee

The requirements for this block are mentioned in Appendix D. Our verification setup consisted of two booth nodes $5m \pm 0.1m$ apart. We subsequently placed the physical device and a smartphone with the application installed near each of the booths.

3.4.1 Physical Device Verification

To verify the requirements mentioned in Appendix D, we console logged the nearby booth MAC addresses and RSSI values and compared it to known values, as shown in Figure 13, verifying the first block requirement. Additionally, to test the functionality of our algorithm, we maintained the setup for 1 minute (which is a user configurable time period) and ensured that the closest booth - the one with the typically highest RSSI value and assigned score - was added to the attended booths list. The setup time is shown in Figure 13 as "Time", thereby meeting our second and third block requirements.

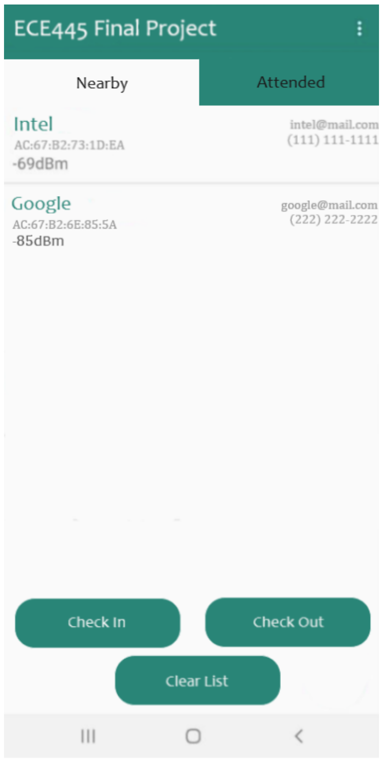


```
STRENGTH:      AVG[-80]       STDDEV[14]
SCORE:         AVG[-3]        STDDEV[73]

[ac:67:b2:6e:85:5a]     STRENGTH:-90    SCORE:-55
[ac:67:b2:73:1d:ea]     STRENGTH:-70    SCORE:49
Strongest: ac:67:b2:73:1d:ea    Percentage of this


SCANNING
BOOTHS ATTENDED:
ID: ac:67:b2:73:1d:ea    TIME: 60
END LIST
```

Figure 13: Attendee device algorithm detecting strongest booth based on RSSI value history

3.4.2 Smartphone Application Verification

To verify the requirements mentioned in Appendix D, we created tab views for nearby and attended booth lists as shown in Figure 14 and Figure 16. Figure 14 verifies our first block requirement. Subsequently we compared the displayed booth names and MAC addresses to the booth hardware to ensure correctness. We further verified that the closest booth has the highest RSSI value of all nearby booths. Lastly to test the functionality of our algorithm, we maintained the setup for the configurable one minute period and ensured that the closest booth was accurately marked as visited and added to the attended list, thereby verifying our second and third block requirements.



Intel intel@mail.com
AC:67:B2:73:1D:EA (111) 111-1111
-69dBm

Google google@mail.com
AC:67:B2:6E:85:5A (222) 222-2222
-85dBm

Figure 14: Nearby list on smartphone application

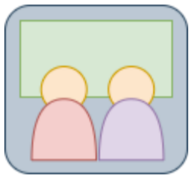
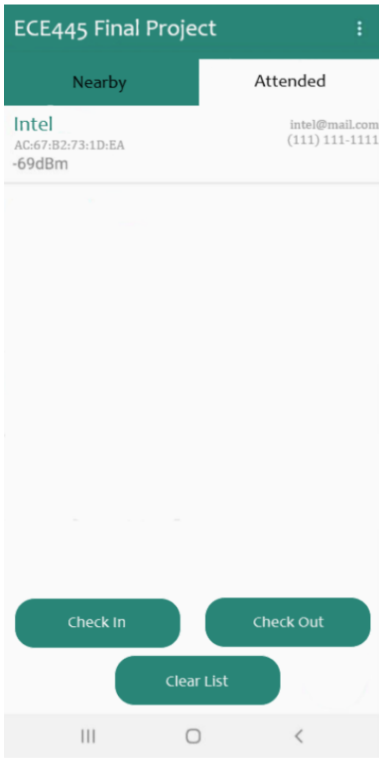


Figure 15: Custom application icon



Intel intel@mail.com
AC:67:B2:73:1D:EA (111) 111-1111
-69dBm

Figure 16: Attended list on smartphone application

4 Cost

4.1 Parts

The majority of our parts were inexpensive, passive components along with a few integrated circuits, a microcontroller, some connectors, as well as a display and switch for user interaction. In an industrial setting, we could have ordered these components in bulk and received better prices. The prices in the "Retail Cost" section of the table below were derived from the prices we paid for enough parts to build three devices. The prices in the "Bulk Purchase Cost" section were derived from the prices we would have paid if we ordered enough components for 50 devices.

Note that some of the line items can be used to manufacture multiple devices, these items are the custom PCBs, M3 screws, 3D printer filament, and OLED displays (these line items have the part name italicized).

Table 1: Parts Costs

Part	**Manufacturer**	**Retail Unit Cost [$]**	**Bulk Purchase Cost [$)]**
ADP7156ARDZ-3.3-R7	Analog Devices	6.76	5.82
B2B-PH-K-S(LF)(SN)	JST Sales America	0.17	0.10
CL31A106KQHNNWE	Samsung Electro-Mechanics	0.19	0.07
CL31B105KAHNFNE	Samsung Electro-Mechanics	0.14	0.06
B4B-XH-A_(LF)(SN)	JST Sales America	0.21	0.19
ESP32-WROOM-32D	Espressif Systems	4.50	4.20
5988230107F	Dialight	0.48	0.29
MCP73831T-2ACI/OT	Microchip Technology	0.56	0.47
SI2323DS-T1-E3	Vishay Siliconix	0.64	0.51
MFS201N-16-Z	Nidec Copal	1.15	1.04
RMCF1206FT100K	Stackpole	0.10	0.01
RMCF1206FT10K	Stackpole	0.10	0.01
RMCF1206FT2K7	Stackpole	0.10	0.03
RMCF1206FT470R	Stackpole	0.10	0.03
RMCF1206FT47K0	Stackpole	0.10	0.03
MBRA120ET3G	ON	0.40	0.30
KMR231NG ULC LFS	C & K	0.49	0.39
DZT2222A-13	Diodes Incorporated	0.33	0.11
2172034-1	TE Connectivity	0.93	0.77
CP2104-F03-GMR	Silicon Labs	1.35	1.30
PL803450	YDL	12.49	12.49
Custom PCB	PCBWay	5.00	5.00
U602602	UCTRONICS	6.99	6.99
3D PLA-1KG1.75-BLK	Hatchbox	22.99	22.99
M3 Screws	iexcell	10.99	10.99

Continued on next page

Table 1 – continued from previous page

Part	Manufacturer	Retail Unit Cost [$]	Bulk Purchase Cost [$]
	Total Parts Cost [per board]:	$77.26	
	Per Board Cost [minus shared line items]:	$31.29	

4.2 Labor

The majority of the cost of development for our system was the labor costs for the engineers; the hourly rate is based on an offer one of our team members had. This offer was in terms of an annual salary, so it was converted to an approximate hourly rate for a full time position.

Table 2: Labor Costs

Team Size	Completion Time (Hours per person)	Hourly Rate ($)
3	150	$45.00
	TOTAL LABOR COST:	**$20,250**

Table 3: Grand Total Cost

Section	Total ($)
Labor	$20,250
Parts [for 3 boards excluding shared line items]	$93.87
GRAND TOTAL COST:	**$20,343.87**

5 Schedule

Table 4: Project Schedule

Week	Anand Sunderrajan	Eric Layne	Mason Edwards
October 5	1. Create UI design sketches for application 2. Create custom icon sketches for application 3. Design Review	1. Finish PCB Schematic 2. Choose I/O, control unit, and power supply components 3. Design Review	1. Choose board components 2. Design Review
October 12	1. Design application wire-frame and building base application 2. Vectorize custom icon using Adobe Suite 3. Start BLE scanning application to display RSSI values for IPR	1. Finish PCB Layout 2. Finish generating PCB component BOM 3. Generate RSSI data	1. Assist finalizing PCB layout
October 19	1. Design software state diagram 2. Implement decision making algorithm ver. 1 3. Finish UI design for application 4. Start RF/BLE software	1. Start implementing control unit design 2. Start RF/BLE software	1. Order soldering supplies 2. Assisted writing hardware management software using Eric's equations
October 26	1. Finish BLE scanning application to display RSSI values for IPR 2. Work on RF/BLE software 3. Implement information to be shown on OLED display 4. Work on implementing the control unit design with Eric	1. Start assembling PCB 2. Work on RF/BLE software 3. Implement information to be shown on OLED display 4. Work on implementing the control unit with Anand	1. Start assembling PCB

Continued on next page

Table 4 – continued from previous page

Week	Anand Sunderrajan	Eric Layne	Mason Edwards
November 2	1. Finish control unit implementation. 2. Finish final UI designs on smartphone application 3. Debug application for compatibility across devices running Android 6.0+	1. Finish control unit and power supply implementation. Start testing and verification 2. Finish Display and IO testing 3. Finish RF/BLE software	1. Finish installing components on PCB that do not require hot air rework 2. Hand off boards and parts to Eric for hot air rework
November 9	1. Combine, test, and verify RF/BLE and smartphone application 2. Start testing and verification 3. Mock Demonstration	1. Combine, test, and verify RF/BLE and smartphone application 2. Mock Demonstration	1. Mock Demonstration
November 16	1. Finish implementing decision making algorithm ver 2. 2. Finish smartphone application 3. Full system testing 4. Demonstration	1. Finish implementing decision making algorithm ver 2. 2. Finish User Device Firmware 3. Full system testing 4. Demonstration	1. Demonstration
November 23	1. Prepare for mock presentation 2. Work on further additions and optimizations for smartphone application	1. Prepare for mock presentation	1. Prepare for mock presentation
November 30	1. Presentation and start work on final report	1. Presentation and start work on final report	1. Presentation and start work on final report
December 7	1. Finish final report	1. Finish final report	1. Finish final report

6 Conclusion

6.1 Accomplishments

The Event Attendance Tracker was able to accurately display nearby booths, determine the closest booth, and mark the closest booth as attended after user configurable parameters were met. By utilizing BLE, our system captured and analyzed relevant RSSI values to determine the closest booth. Through the use of continuous measurement intervals, the system was able to determine and track the closest booth in real time, while providing relevant contact information about that booth. It provided an extremely accessible alternative which is significantly lower cost than current market solutions.

6.2 Uncertainties

The usage of RSSI values to determine the closest booth presents uncertainty due to the inherent inadequacies of RSSI. This indicator is one plagued with high variability due to its susceptibility to environmental factors [18]. This was apparent in our initial testing where the RSSI value varied by up to $\pm 10dBm$ while the booth node and smartphone were at a constant distance, but with an object occluding line of sight. This compared to the change of $-10dBm$ from $150cm$ to $300cm$ of separation with clear line of sight makes the influence of environmental factors on distance estimation apparent.

6.3 Ethical considerations

Our device has a few potential safety concerns that must be addressed during the development process. IEEE's 7.8 Code of Ethics, Section I Policy 1 [19] states "To hold paramount the safety, health, and welfare of the public," which our device must uphold as it incorporates a lithium-polymer battery. This type of battery chemistry can be prone to explosions or fire when not kept in a safe voltage/current draw range or if exposed to high temperatures [20]. We must ensure that the battery control circuitry can maintain the operation of the device and keep the battery cell within safe operating ranges for both voltage and current draw. This, in addition to warnings about not exposing to extreme temperatures, will help to reduce the chance of the device posing a risk of personal injury or property damage. The best way to approach this challenge is to use conventional and reliable components and implement them to manufacturer specifications. This means we can leverage the development and testing the manufacturer went through in the design process to ensure the device operates as expected and will safely manage the battery.

This device will incorporate RF communication via Bluetooth Low Energy and any venture into RF transmission requires adhering to FCC guidelines.

> The FCC regulates radio frequency (RF) devices contained in electronic-electrical products that are capable of emitting radio frequency energy by radiation, conduction, or other means. These products have the potential to cause interference to radio services operating in the radio frequency range of 9kHz to 3000 GHz. [21]

Specifically, our device is what is designated as an “Intentional Radiator.” For this application we will be using an RF IC incorporated into an ESP32 SOM with an intentionally limited communication power. As such there will be little to no risk of introducing adverse amounts of RF interference, even with several of these devices operating in close proximity. By using an FCC certified device [13], the ESP32, as well as responsibly utilizing the RF communication (not constantly broadcast at max transmission power) through BLE, we can ensure the device does not interfere with the operation of other wireless devices nearby beyond what the standard for BLE allows.

Sections 1.3, 1.6, and 1.7 of the ACM code of conduct dictate that we be honest, be trustworthy, and to respect privacy [22]. Our system can be designed in a way such that it will not have to remotely store sensitive user data; however, we still have a duty to not hoard, mine, sell, or distribute any data that we are entrusted with which is temporarily stored locally in the application. The feature that our design uses to meet these responsibilities is that all BLE communication uses a randomly generated user ID which cannot be directly correlated to any specific user. This user ID can only be correlated when the user consents to have their information shared with their chosen booths they visited at a particular event. Ideally, we should act as a pure middleman between the attendee and the booth by not storing any of the data, and rather simply passing it along once an attendee consents to sharing their information. Furthermore, it is our responsibility to not abuse the trust that users place in the smartphone application. We must not abuse the processing power of the device we are given access to, nor attempt to extract any other data from their personal device. In the same light, we must ensure our application does not abuse its Bluetooth or location permissions required for functionality. To do this, we will ensure the application does not connect or communicate with nearby devices, and only does so with devices confirmed to be booth nodes.

Lastly, given the current global situation involving COVID-19, all members of the group would be following CDC recommended safety guidelines to prevent the spread of COVID-19, and receive testing as required, per the student guidelines provided by the University of Illinois [23]. We will conduct nearly all our work virtually unless in-person contact is absolutely necessary.

6.4 Future work

The hardware can be developed further to include an SD card or USB port to allow booth presenters to offload data collected from an event or to upload a configuration file for easily setting event specific parameters.The hardware attendee device is currently much larger than it needs to be, and with a second custom PCB could be made smaller to improve usability of the hardware device for those without a compatible smartphone.

In conjunction with these hardware modifications, we could further make improvements to the software portion of our system as well. The smartphone application can be developed for both iOS and wearable devices improving the accessibility for our system. Reaching out further, we can develop a database containing information about the booths for each event. This would subsequently facilitate the development of a login-based web interface (using React) permitting booth presenters and attendees to access relevant data and statistics. Integrating a server would allow in-house communication and interaction between booth presenters and attendees, without the need of external methods such as emails or phone calls. Further improving the decision making algorithm to one that is based on a machine learning model would allow for vast improvements in accuracy while maintaining the low cost and resource utilization of the device.

References

[1] “Attendance Audit Summary CES 2019,” pdf, Consumer Technology Association, Las Vegas, Nevada, 2019, Accessed December 8 2020. [Online]. Available: https://cdn.ces.tech/ces/media/pdfs/ces-2019-audit-summary.pdf

[2] “Event Attendance Tracking Software & Mobile App,” ScanTrakk, Nov 2017, Accessed December 8 2020. [Online]. Available: https://scantrakk.com/

[3] “Event Monitoring using Bluetooth low-energy beacons,” Beaconex LLC, Accessed December 8 2020. [Online]. Available: https://www.beaconex.io/services

[4] “Health Professions Career Fair,” Bradley University, 2019, Accessed December 8 2020. [Online]. Available: https://www.bradley.edu/offices/student/scc/employers/fairs/NursingPTFair.dot

[5] “How Many Recruiters Does It Take To Work A Career fair?” National and Association of Colleges and Employers, 2019, Accessed December 8 2020. [Online]. Available: naceweb.org/talent-acquisition/best-practices/how-many-recruiters-does-it-take-to-work-a-career-fair/

[6] Google, “Bluetooth Low Energy Overview,” Accessed December 8 2020. [Online]. Available: https://developer.android.com/guide/topics/connectivity/bluetooth-le

[7] Google, “BluetoothGATT Documentation Reference,” Accessed December 8 2020. [Online]. Available: https://developer.android.com/reference/android/bluetooth/BluetoothGatt

[8] Google, “Runnable Documentation Reference,” Accessed December 8 2020. [Online]. Available: https://developer.android.com/reference/java/lang/Runnable

[9] Renesas, “Linear vs. Switching regulators,” Accessed December 8 2020. [Online]. Available: https://www.renesas.com/us/en/products/power-power-management/linear-vs-switching-regulators

[10] *ADP7156*, datasheet, Analog Devices, Accessed December 8 2020. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ADP7156.pdf

[11] *MCP73831/2*, datasheet, Microchip, Accessed December 8 2020. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP73831-Family-Data-Sheet-DS20001984H.pdf

[12] de Carvalho Silva, Jonathan, Rodrigues, Joel, Alberti, Antonio, Šolić, Petar, and Aquino, Andre, in *LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities*, 2017, Accessed December 8 2020. [Online]. Available: https://www.researchgate.net/publication/318866065_LoRaWAN_-_A_Low_Power_WAN_Protocol_for_Internet_of_Things_a_Review_and_Opportunities

[13] *ESP32-WROOM-32D ESP32-WROOM-32U*, datasheet, Espressif Systems, Accessed December 8 2020. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf

[14] M. Lai, “ESP32 Boot Mode Selection,” Accessed December 8 2020. [Online]. Available: https://github.com/espressif/esptool/wiki/ESP32-Boot-Mode-Selection

[15] L. Bank, “How much current do OLED displays use?” 2019, Accessed December 8 2020. [Online]. Available: https://bitbanksoftware.blogspot.com/2019/06/how-much-current-do-oled-displays-use.html

[16] “Get Started with iOS App Development — AWS,” Amazon, Accessed December 8 2020. [Online]. Available: https://aws.amazon.com/mobile/mobile-application-development/native/ios/

[17] Fengjun Shang, Wen Su, Qian Wang, Hongxia Gao, and Qiang Fu, “A Location Estimation Algorithm Based on RSSI Vector Similarity Degree,” *SAGE Journals*, Aug 2012, Accessed December 8 2020. [Online]. Available: https://journals.sagepub.com/doi/full/10.1155/2014/371350

[18] Rong-Hou Wu, Yang-Han Lee, Hsien-Wei Tseng, Yih-Guang Jan, and Ming-Hsueh Chuang, “Study of characteristics of RSSI signal,” in *2008 IEEE International Conference on Industrial Technology*, 2008, Accessed December 8 2020. [Online]. Available: https://ieeexplore.ieee.org/document/4608603 pp. 1–3.

[19] “IEEE Code of Ethics,” Online, Institute of Electrical and Electronics Engineers, Accessed December 8 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[20] U. of Washington, “Environmental Health Safety: Lithium Battery Safety,” Seattle, Apr 2018, Accessed December 8 2020. [Online]. Available: https://www.ehs.washington.edu/system/files/resources/lithium-battery-safety.pdf

[21] “Equipment Authorization - RF Devices,” Federal Communications Commission, 2018, Accessed December 8 2020. [Online]. Available: https://www.fcc.gov/oet/ea/rfdevice

[22] “ACM Code of Ethics,” Online, The Association for Computing Machinery, Accessed December 8 2020. [Online]. Available: https://www.acm.org/code-of-ethics

[23] U. of Illinois, “University Guideline for COVID-19 for Students,” Online, Accessed December 8 2020. [Online]. Available: https://covid19.illinois.edu/guides/students/

Appendix A Final Schematic

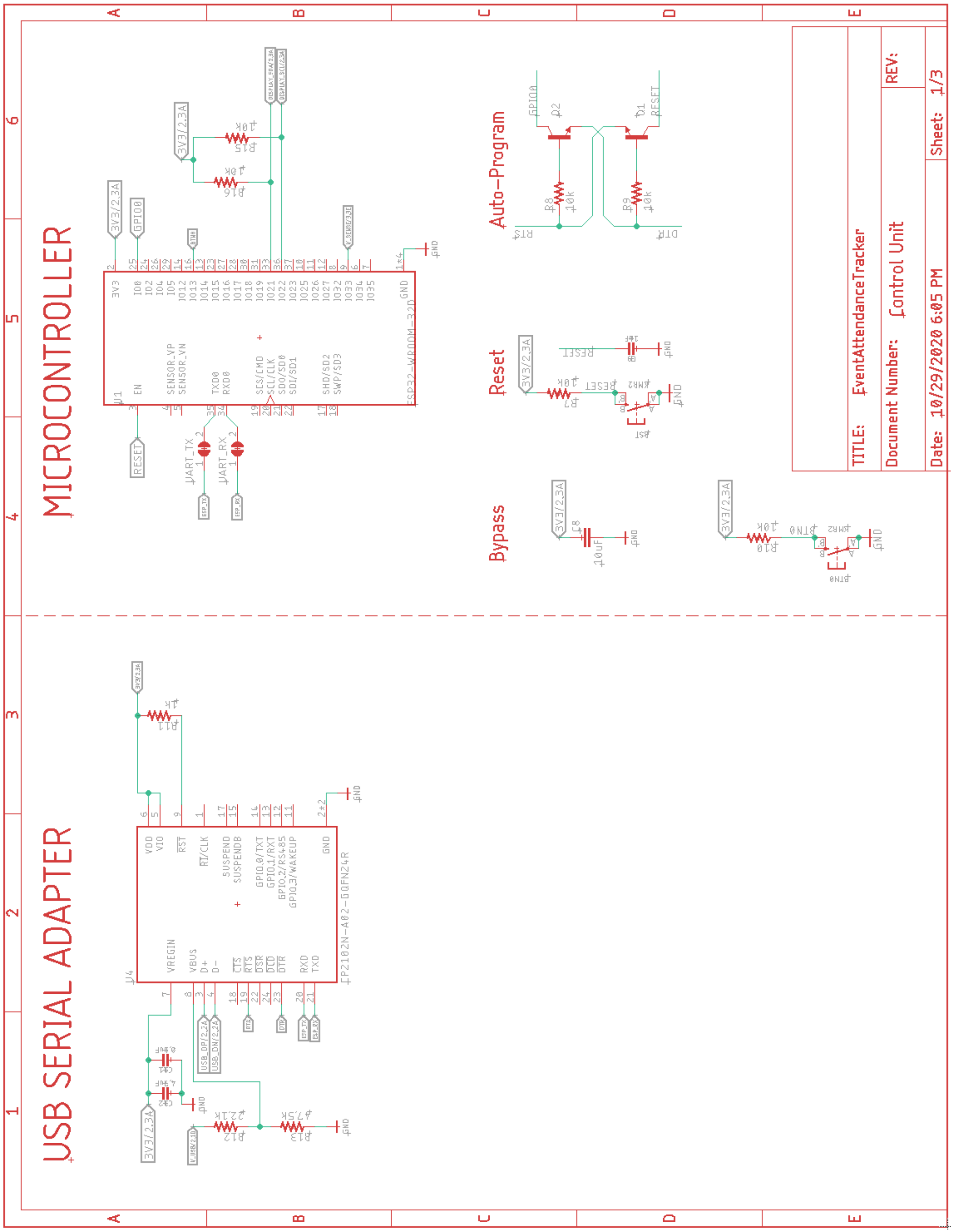


Figure 17: Control unit schematic

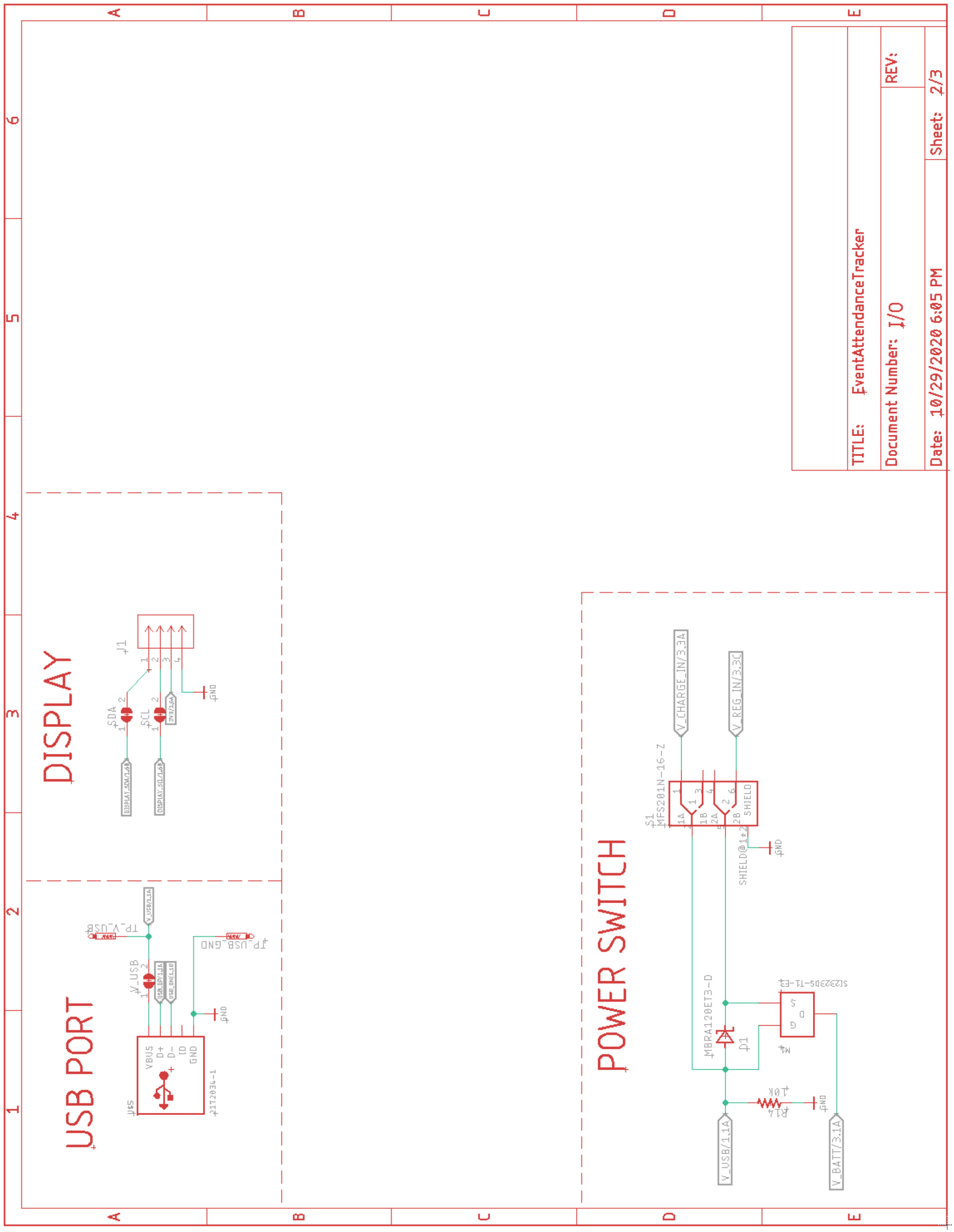


Figure 18: I/O subsystem schematic

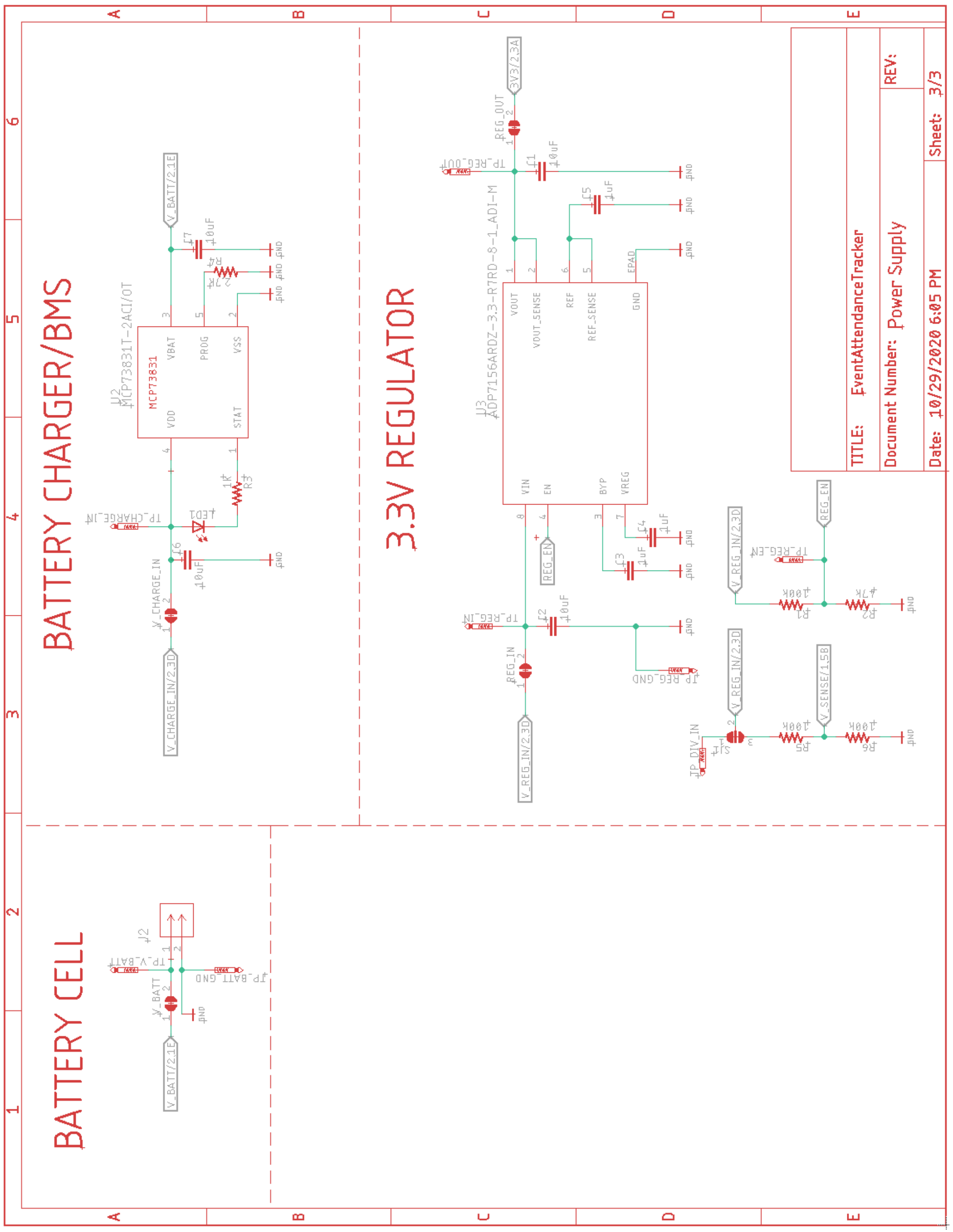


Figure 19: Power supply schematic

Appendix B Final PCB Layout

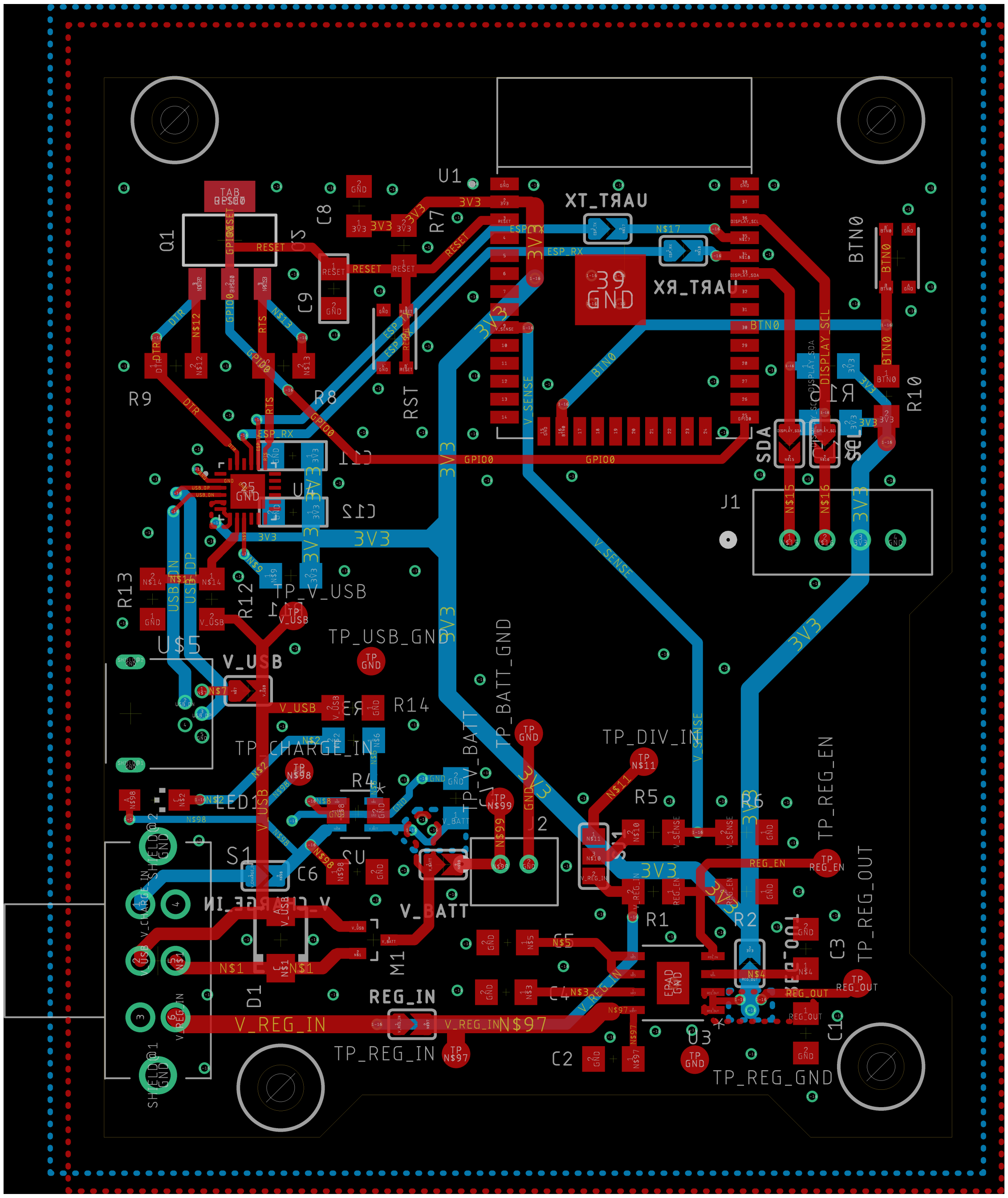


Figure 20: Booth hardware PCB layout

Appendix C Software Development

Code Repository Links

Full Code for Firmware found: https://gitlab.elayne.me/eric/ECE445_Project

Full Code for Smartphone application: https://github.com/AnandSunderrajan/FA20-ECE-445

Algorithm Psuedocode

Inputs: MIN_SCORE: minimum allowable score; MAX_SCORE: maximum allowable score; PENALTY: score penalty for not being detected at all in a scan; scores: Hash-map of all booths' MAC, their scores, last rssi value, and a boolean if they have been updated

```
foreach booth in scores do
    if booth.updated == false then
        booth.score = booth.score - PENALTY;
    else
end
update_RSSI_statistics();
foreach booth in scores do
    if booth.updated == true then
        booth.score = (1-$\alpha$) $* booth.score + \alpha * \frac{booth.rssi - avg\_rssi}{std\_dev\_rssi}$ else
    end
    foreach booth in scores do
        if booth.score  MIN_SCORE then
            delete booth;
        else
    end
    foreach booth in scores do
        booth.updated = false;
        booth.score = min(max(booth.score, MIN_SCORE), MAX_SCORE));
    end
    update_score_statistics();
```

Algorithm 1: Update Booth Scores

Inputs: MIN_SCORE: minimum allowable score; scores: Hash-map of all booths' MAC and their scores

Output: MAC address of booth with highest score

```
MAC = "";
max_found = MIN_SCORE;
while not at end of hash map; i++ do
    if scores[i]  max_found then
        max_found = scores[i].value;
        MAC = scores[i].MAC;
    else
end
return MAC
```

Algorithm 2: Get Strongest

Appendix D Requirement and Verification Table

Table 5: System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
Power Supply Requirements 1. Capable of outputting $3.3V \pm 0.1V$ at $750mA$ 2. Operate fulfilling requirement (a) across input voltage range $3.6V$ to $5.25V$ 3. Charges battery cell at no more than 0.5C without exceeding $4.2V + 0.1V$	Verification 1. Provide $4.20V$ from a bench power supply and draw $750mA$ from the regulator output; record and evaluate the output voltage using a DMM 2. Repeat $750mA$ load test for the input voltages $3.6V$ and $5.25V$ while recording the regulator output voltage with DMM. 3. Power the subsystem with $5.0V$ from a power supply; monitor the current drawn and voltage across the battery cell using a pair of DMMs	Status 1. Y: $3.295V$ output 2. Y: $3.6V$ input: $3.299V$ output; $5.25V$ input: $3.289V$ output 3. Y: Maximum voltage: 4.21V (after charge completion); Maximum current: 374mA (during constant current charge phase)
Control Unit Requirements 1. Must show up as a valid COM/tty device without errors when plugged into a USB port 2. I^{2}C bus must detect and communicate with any devices on the I^{2}C bus 3. Measure the voltage of the battery cell within $\pm 0.035V$ of actual across the battery range $3.5V$ to $4.2V$ 4. Must be detected as a BLE device when closer than $5m \pm 0.1m$. 5. Able to transfer 4kB of data within 1 second over BLE	Verification 1. Connect the device to multiple computers, and use a serial terminal to verify it is functioning as a valid serial device 2. Connect the (100kHz) I^{2}C bus to a test slave device, and verify the control unit can communicate with the device on each address needed 3. Measure the voltage of the battery cell using a DMM at the same time as recording the control unit's estimated voltage. Repeat this at various points throughout the battery range to find the maximum deviation from the actual voltage 4. Upload test firmware to advertise the control unit as a BLE device; use a BLE scanning application to detect this device, and repeat across various distances to ensure repeatability 5. Generate 4kB of random bytes, and upload a test firmware to transmit these bytes to a serial terminal over BLE. Record the time to transfer the data, and verify the bytes are unmodified	Status 1. Y 2. Y 3. Y: Max deviation $23mV$ 4. Y 5. Y: Transferred in $115mS \pm 15$mS

Continued on next page

Table 5 – continued from previous page

Requirement	Verification	Verification status (Y or N)
I/O Requirements 1. I^2C bus must be functional and able to toggle every pixel of the display 2. The power switch circuit must not allow more than $\pm 0.1mA$ to flow to the BMS when the device is in "active" mode 3. When USB power is available, no more than $\pm 0.1mA$ is allowed to flow through the battery voltage output 4. USB port must provide current to charge the battery cell or to power the device while in "active" mode, without exceeding the lower of 85°C or the maximum temperature of any component	Verification 1. Use a test program to drive the I^2C bus, attempting to toggle each pixel individually. Verify visually that each pixel toggles as expected. 2. Use a DMM to measure the leakage current between the switch circuit and BMS, switch the device to "active" mode. Record the current after reaching steady state 3. Place a DMM in series between the battery cell output and the power switch input, put the device in "active" mode. Record the current after reaching steady state 4. Connect a USB power supply to the USB port, and use a DMM to measure the current drawn from the USB port. Verify the current is as expected (for"active" or "charging" modes), confirm this current is sustainable for 15 minutes and no component exceeded their maximum temperature	Status Note: DMM current accuracy limited to nearest 0.1mA on 0-10mA range 1. Y 2. Y: 0.0mA leakage 3. Y: 0.0mA leakage 4. Y: Active: 137mA drawn from USB; Charge: 375mA drawn from USB
User Device Requirements 1. Detect all available BLE booth nodes within at least 6 meters using RSSI values, and sort them based on signal strength 2. Monitor how long a BLE booth node is within range of (according to a calibrated threshold) a BLE booth node and mark a booth as attended after the user configurable time threshold is met 3. Accurately display all nearby booths and attended booth nodes in separate lists	Verification 1. Surround the device with several booth nodes at varying distances and determine if the device correctly detects them all, has valid RSSI values for each, and sorts them in descending order based on signal strength 2. Keep a device nearby a booth node for a set amount of time, then move the device away. Check if the user device correctly flags the closest booth node as attended after the predetermined time period 3. Keep the user device near multiple booth nodes at varying distances and present both lists. Check if the each booth node is entered into the correct list	Status 1. Y 2. Y 3. Y