ECE 445: Final Report

Collective Child Tracking System

Team 39
Mark Hafter
Omar Joya
Ellery Tomaszkiewicz

ECE 445 Senior Design, Spring 2017
TA: Jose Sanchez Vicarte

Abstract

Parents, school teachers, and supervisors are constantly concerned about the safety and well-being of children in a public environment. With an increasingly interconnected world, there is similarly increasing opportunity to leverage today's technology to support child safety. A comprehensive tracking system would reduce stress on adult supervisors by facilitating the location of multiple children at the same time, in a diverse range of environment. The documented, tested, and verified Collective Child Tracking System (CCTS) outlined in this document fulfills this objective. We explore the design considerations of the CCTS and then verify that each subcomponent functions properly and integrates with the rest of the system. Finally, we reflect on the successes and uncertainties of the project, as well as future work that could be done on the design. Altogether, the final report is a compact summarization of the work accomplished for the CCTS and the primary lessons learned from this procedure.

Table of Contents

1 Introduction

1.1 Background

According to the Federal Bureau of Investigations, there were 465,676 reports of missing children in 2016 alone [1]. Therefore, it is no surprise that parents and other adult supervisors consistently worry about the safety of the children in their care. Teachers and counselors especially worry about ensuring the safety of groups of children when on an excursion or field trip, fearing that a child could easily slip away. With this target market in mind, the problem of supervising many children at once, and without the high-demand dependence on the Internet, remains highly relevant and presents a large area of opportunity.

Among all the cases of missing children, the vast majority of them are fortunately resolved within the first few hours [2]. This indicates a massive opportunity for the prevention of wandering and straying children in large and public group settings. It is also important to note that on average, children receive their first smart phones at around the age of 10 [3]. As a result, there is no direct method of tracking a child′s whereabouts until they enter middle school age. There are countless child-tracking devices that are currently on the market and seek to fix this issue. However, they all suffer from similar problems: relatively short battery life, complete reliance on GPS technology, or short range of tracking [4].

1.2 Our Solution

The Collective Child Tracking System (CCTS) that we propose addresses the supervision of multiple children through wearable devices in a unique configuration. The adult supervisor would be equipped with a radio receiver that would consistently read signals periodically from wearables worn by the children. Once a wearable exits the approximate 100 meter range of the transmitter, the wearable will send a GSM signal to the supervisor′s phone, carrying an alert message with a last-known location. The innovation of the system lies with how the location is determined: GSM technology allows the wearable to detect nearby cell phone towers and their relative signal strengths. Google′s geolocation API implements this raw data to determine an approximate location that is shown to the supervisor through their smart phone. Altogether, the CCTS provides a quick and seamless system that pinpoints the location of a child as they wander too far away from the central group

Our proposed CCTS design accomplishes three main facets:

1. The system determines whether a specific child device is within a maximum 150m range of the adult device at intervals of five seconds.
2. The child devices, upon loss of contact with the adult device, transmit via SMS the current cellular telemetry data from the GSM modem for geolocation.
3. The child devices functions for about 11.5 hours, well beyond the length of a typical school day.

2 System Design

Because of the multi-layer nature of the CCTS design, its components can be divided primarily between hardware and software. The majority of this section explains how the two devices in the design will function together. The electrical and hardware components of the CCTS are illustrated in Figure 1. In addition to the two devices in the system, components that function externally are included in the diagram. These components are critical in ensuring the full functionality and sustainability of the CCTS design.

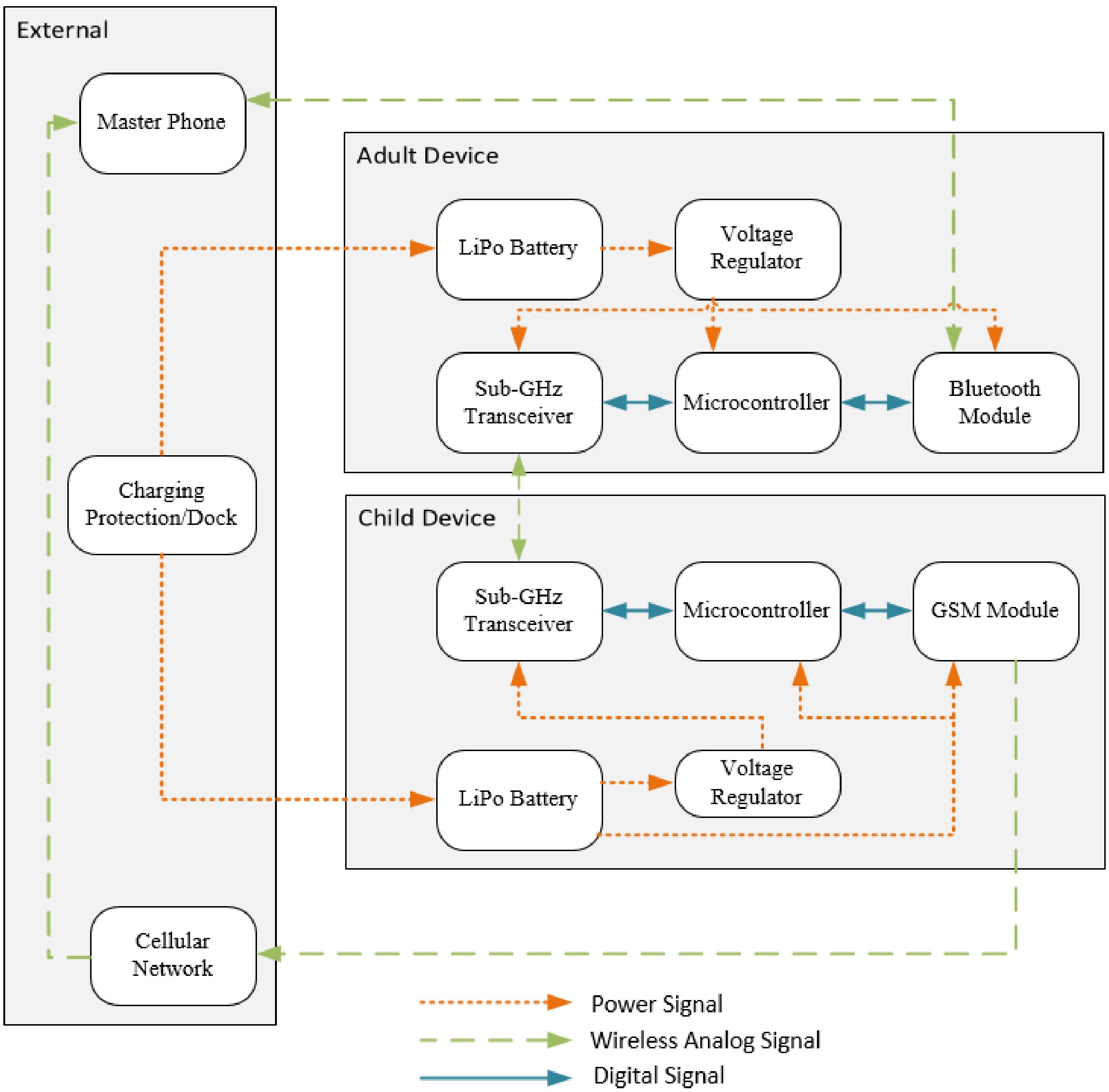


Figure 1: Block diagram overview of CCTS

2.1 Microcontroller

Location: Adult Device, Child Device
Unit: ATmega823P Microcontroller

The microcontroller serves as the primary computation and processing unit for the variety of signals coming from and into the adult device and child device. For example, the unit will synchronize the clocks of the transceiver modules so they may appropriately communicate with each other. Moreover, the logic of the microcontroller will determine the next action or state in the CCTS system, as described in Section 2.1.1.

The configuration of the microcontroller can be found in Figure 2. For the adult device, we required simultaneous SPI interfacing with the Bluetooth and transceiver modules. The ATmega328p supported software SPI that enabled us to implement this. Moreover, the 16MHz clock capability supports the ability to communicate with thirty child devices in just 5 seconds. For the child device, we achieved lower power consumption at an operating voltage 4.0 volts, along with modified library support that allowed for custom modification of the GSM module. This capability was crucial in surveying the location of surrounding cellular towers.

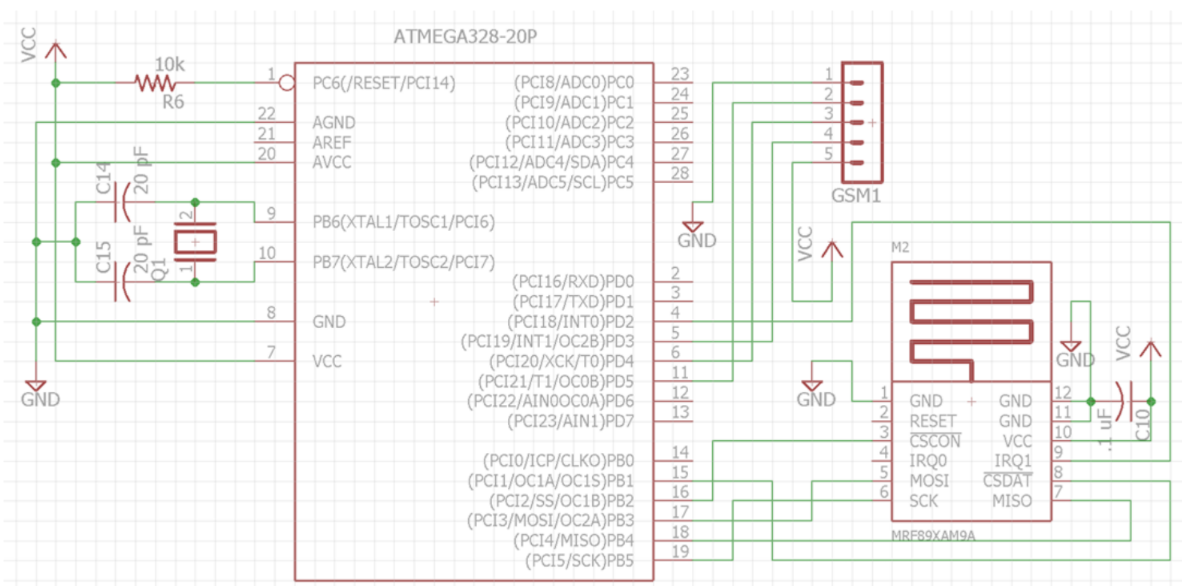
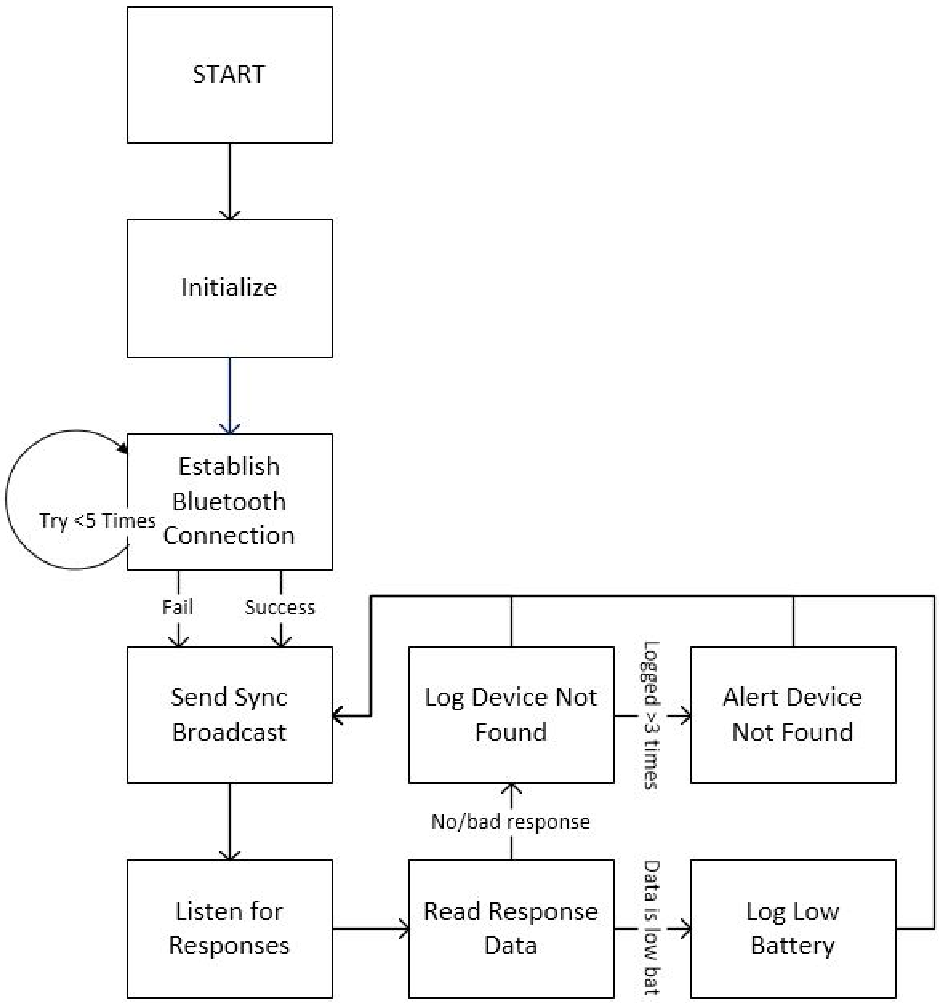


Figure 2: Microcontroller schematic and configuration

2.1.1 Adult Device Logic

The adult device will initialize and attempt to establish a Bluetooth connection with the adult smart phone. If it is successful, it will proceed. If not, it will try again a number of times, and will still continue to operate even without a Bluetooth connection (the system can still verify that the child devices are in range, the smart phone connection is only for operation of the application. As a backup, the SMS messages can be received by the phone without the application). If a device fails to respond more than seven times, the adult device will mark it as lost and alert the user to this condition. It will continue to attempt to reestablish communication with all devices.

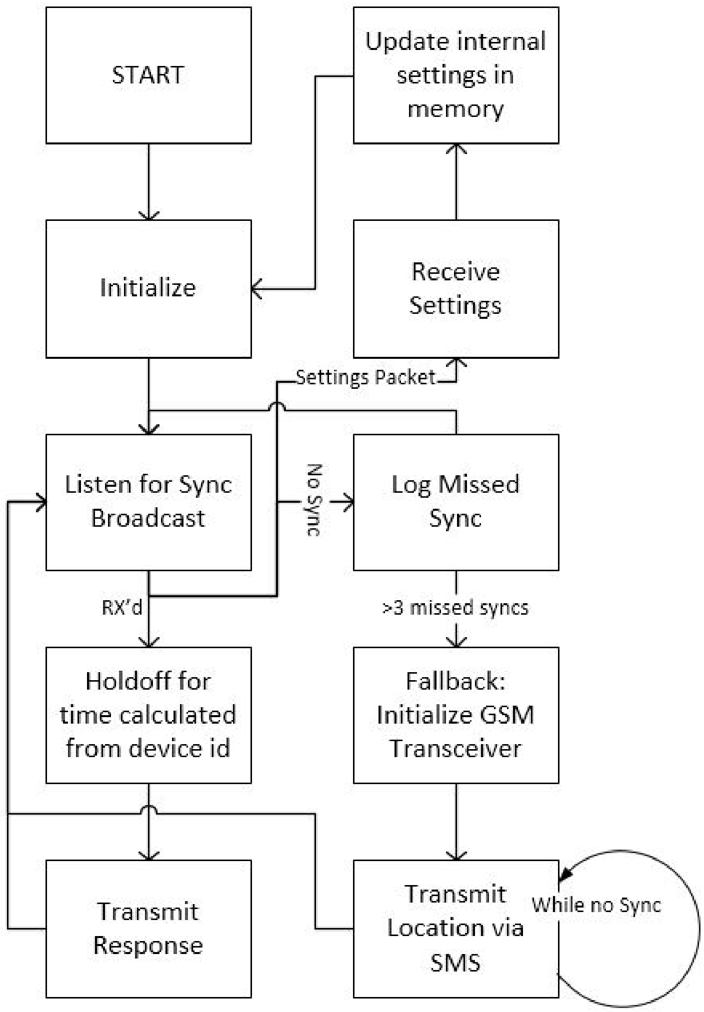
Figure 3: Block Diagram of Adult Device Algorithm



2.1.2 Child Device Logic

The child device initializes and begins to listen for a sync broadcast from the adult device. This broadcast is both a trigger to respond as well as a method to synchronize timing between child devices. The child device will then wait for a set amount of time from the reception of the broadcast. This time is calculated based on the device ID of the child device. The child device then transmits a response to the adult device indicating that it is in range and has heard the broadcast. In the even that a sync broadcast is not received more than three times, the child device will enter fallback mode, activate its GSM modem, and begin to relay its location via SMS to the adult smartphone. In a future version, there will also be a mechanism for changing settings such as device ID and group ID (to be used later to set frequency). This will be via packet transmitted by the adult device.

Figure 4: Block Diagram of Child Device Algorithm



2.2 Sub-GHz Radio Transceiver

Location: Adult Device, Child Device
Unit: Microchip MRF89XAM9A Sub-GHz Radio Transceiver

The 920 MHz radio transceiver module was our choice for communication between the adult device and the child device. Its desirable range and SPI interface provided an ideal solution for low-power applications. The circuit configuration is demonstrated in Figure 5 with the pertinent connections to the microcontroller labeled. On the adult device, the transceiver is connected to different pins on the microcontroller. This is to accomodate the bluetooth transceiver which blocks until it can control the SPI pins at nondeterministic times, causing interoperability issues with another device on the standard SPI pins. Instead, we initialize a software SPI to allow the microcontroller to communicate with the transceiver on the adult device.

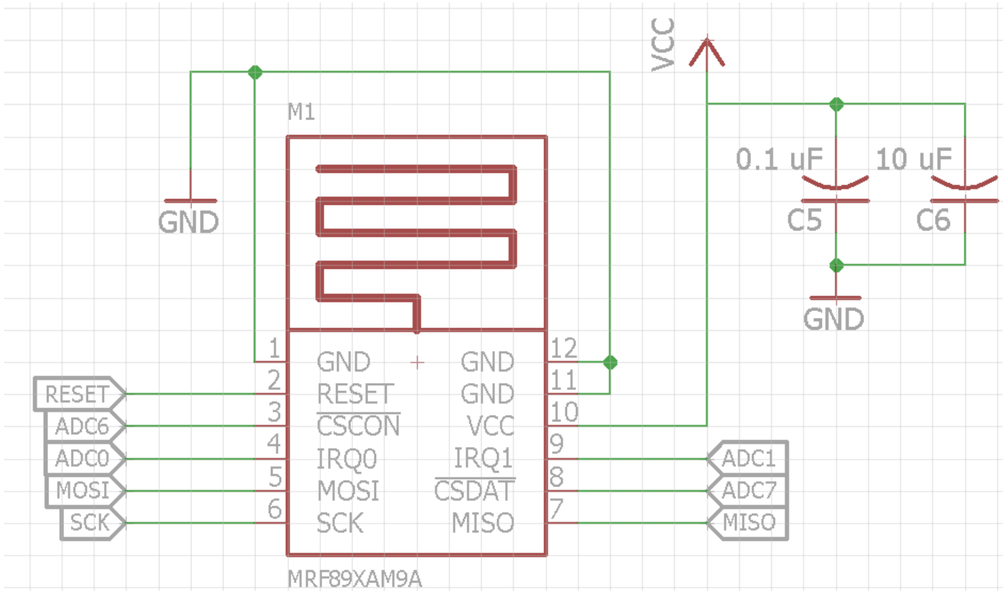


Figure 5: Sub-GHz schematic and configuration

The radio transceiver for both the child and adult devices was designed to be powered at the regular 4.2 volts provided by the battery, though we use a voltage regulator to bring the input voltage down to 2.8 volts. The transceiver sends one packet per time slot (which occurs every 5 seconds) containing no information other than its device ID. In the future, this could include other data such as battery health. The child's antenna will periodically transmit a "handshake" with the adult device until it is out of range. Once a connection has been lost or is too weak, the microcontroller receives this new state from the transceiver and employs the appropriate response for the rest of the modules in the system.

To better understand the link budget and range between two Sub-GHz transceivers, a simple application of the Friis transmission equation will give us an estimate of the device's range. From the transceiver's datasheet, we obtain the following information:

- Reception sensitivity: -105 dBm or 3.16 x 10^{-11} mW (FSK modulation)
- Output power: 10 dBm or 10mW
- Antenna Gain: -0.9545 dBi or 0.8027 omnidirectional
- Operating Frequency: 915 MHz

The Friis transmission equation has the generic form of

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi R)^2} \tag{1}$$

If we algebraically rearrange the equation and solve for our desired R range, we obtain a new form of the equation.

$$R = \sqrt{\frac{P_t G_t G_r \lambda^2}{P_r (4\pi)^2}} \tag{2}$$

Suppose we design for a conservative sensitivity of -60 dBm instead of the specified minimum sensitivity of -105 dBm. We calculate the resulting range to be as follows:

$$R = \sqrt{\frac{(10)(0.8027)(0.8027)(\frac{3x10^8}{915x10^6})^2}{(10^{-6})(4\pi)^2}} = 66.23m \tag{3}$$

2.3 GSM Module

Location: Child Device
Unit: SIMCom SIM800H GSM Module

In the event of loss of contact with the adult device RF transmitter, the microcontroller triggers a fallback condition during which it will activate the GSM modem. The GSM modem then, upon receiving start-up commands, automatically establishes a connection with the global GSM network as an SU (Subscriber Unit). The GSM module communicates with the microcontroller using a serial connection running at 115200bps using "AT+" codes, standard codes for use with cellular devices, and well-defined in the data sheet [5]. We modified a 3rd party library with the necessary modifications to optimize for our uses. The modem gathers data from cellular towers in range and relay the following data to the microcontroller for each neighboring cell tower:

- Cell ID
- Cell Location Area Code
- Mobile Country Code
- Mobile Network Code

Then, the microcontroller reformats this data and transmits it via SMS to the adult telephone, which creates a JSON request to the Google geolocation API.

Upon reuniting the child device with the adult device, and receiving three consecutive messages from the adult device, the microcontroller will shut down the GSM module.

2.4 Bluetooth Transceiver

Location: Adult Device
Unit: Bluefruit LE nRF8001 Module

This particular Bluetooth module serves as a front-end client, so any microcontroller may use SPI interfacing to control it. Moreover, the nRF8001 provides seamless integration with smart devices, which is our intent with this unit. With the bluetooth module, the master phone displays information based upon the status of the child devices that are nearby. For example, if a child device loses contact with the adult sub-GHz transceiver, the phone will read the data sent by the adult device that indicates that a specific child device is now lost. The phone will remove this alert once the child device is back in range, all of which is communicated via Bluetooth. Moreover, each child device has a unique identifier programmed into the microcontroller, which will also be sent to the phone via Bluetooth. The partial circuit implementation of this chip is shown in Figure 6, where most of the components have already been implemented for our ease of use.

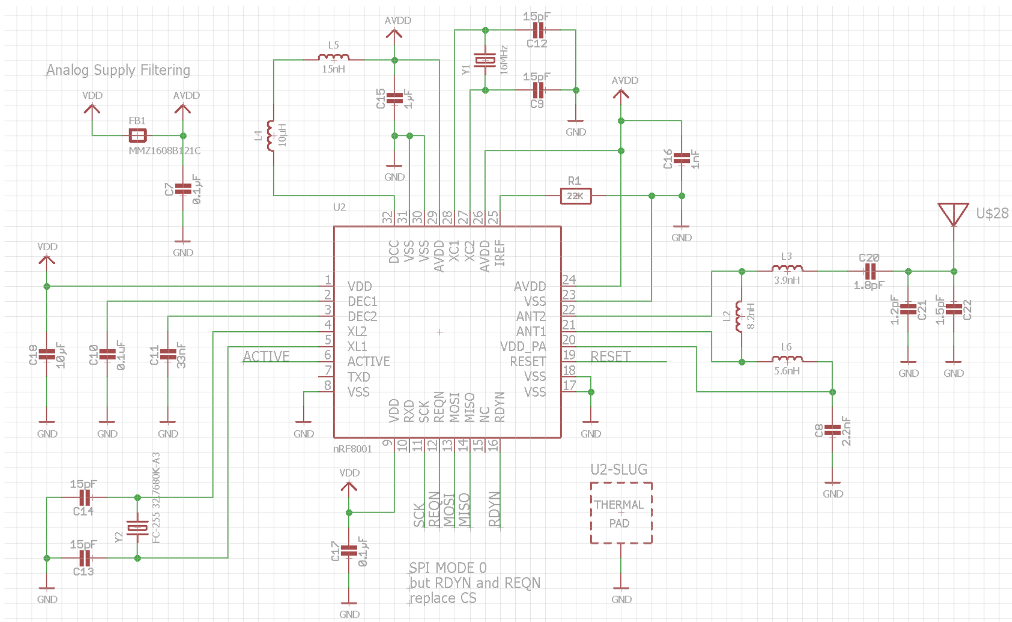


Figure 6: Bluetooth module schematic and configuration (partial)

The Bluetooth transceiver is powered by a 2.8 volt source, just like most of the other components in our design. The module allows us to concentrate on only the SPI-specific pins for programming and implementation of the chip. The transceiver can communicate with the microcontroller via SPI [6], and presents itself as a UART interface to the adult phone.

2.5 Lithium Polymer and Battery and Charger

In order to power each device, a 500 mAh lithium polymer mini rechargeable battery is used. It has a 3.7 V to 4.2 V output voltage and is only 40mm x 25mm x 4.9mm in size. This is important because it needs to fit into a device that can be put on a child's wrist. Another very important feature is the time the battery will last. In our high-level requirements it needs to last at least 5 hours. We were able to accomplish this, as shown in the equation 4 below.

$$Time_B = \frac{Battery\ Capacity}{Avg\ Current\ Draw} = \frac{500x10^{-}3Ah}{0.04Amps} = 12.5\ hours \tag{4}$$

This equation uses current draw without peaks. At peak current (approximately 0.06 Amps), the battery time will be equal to 8.33 hours. Since the device will not always be at peak current, the combination of both current draws will cause the battery time to be about 11 hours. This fulfills the requirement of having the device be on for at least 5 hours.

Charging the battery is accomplished by the 5V input Micro-B USB connector to battery, MCP73831 LIPO Charger). The circuit layout is shown below in Figure 7.

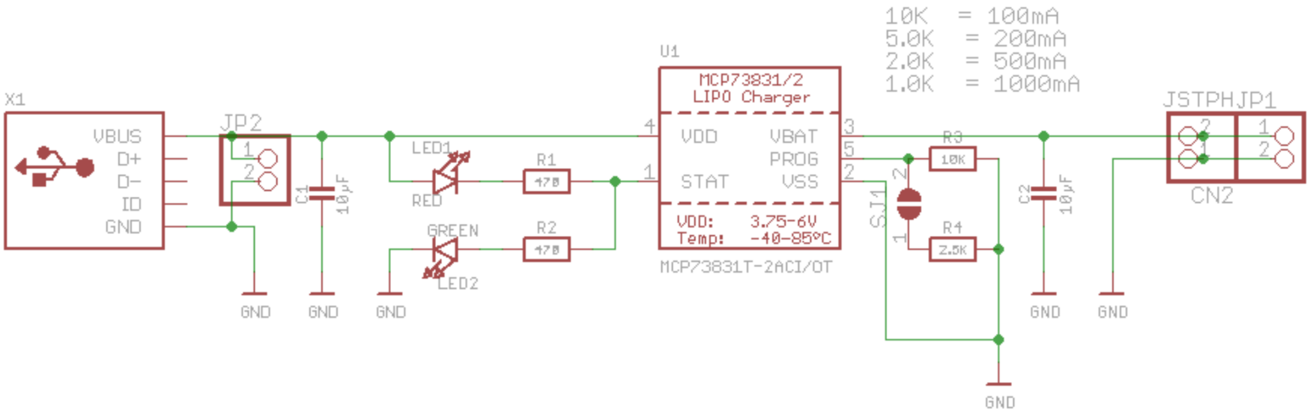


Figure 7: Battery Charger Configuration [7]

We soldered a jumper on the pcb to have a 500mA charge instead of 100mA. When the battery is charging, a red LED glows and when the charge is complete, the green LED glows. This is very important to know that the battery is at full capacity and will be able to last all 11 hours. The charging of the battery is performed in three stages. First a preconditioning charge, then a constant-current fast charge, and finally a constant voltage trickle charge to keep the battery topped-up. This is important to keep the current low when the battery is almost at full charge.

2.6 Voltage Regulator

The Sub-GHz transceiver and the Bluetooth LE chip both need an input voltage below 3.3V. In order to accomplish this, we use an adjustable voltage regulator (LM317). The circuit diagram is shown below in Figure 8.

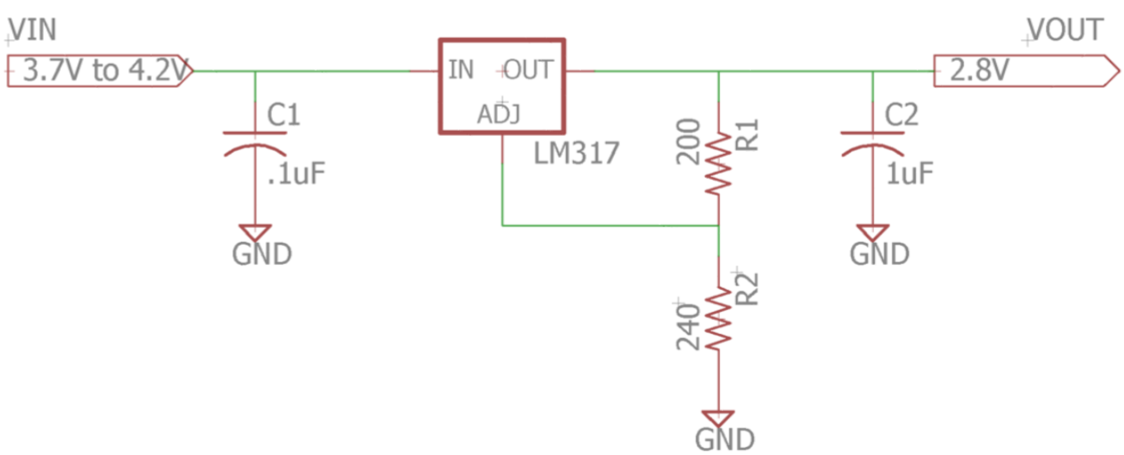


Figure 8: Voltage Regulator Schematic and Configuration

Shown in the figure above, the input voltage is 3.7V to 4.2V from the battery and then goes through the voltage regulator which outputs a voltage of 2.8V. This was accomplished by using the equation 5 below to determine what two resistors to use.

$$V_{out} = 1.25(\frac{R_2}{R_1} + 1) \tag{5}$$

From this equation, we determined we can use a 200 Ω and a 240 Ω resistor to have a 2.8 V output. This circuit is used in both the adult and child devices.

2.7 Android Phone Interface and Geolocation

The Google Maps Geolocation API is a tool provided by Google to pinpoint an approximate location of a device based on its surrounding cell towers and WiFi nodes [8]. For the scope of our project, we implemented only cell tower information. A request for the child's device approximate location is triggered by an SMS message that contains raw cell tower information, including area codes and unique cell IDs. Via JSON request, the raw information can be input into a customized Android application that displays the coordinates on a map.

The overall interface of the Android application were minimal and friendly for the purposes of our project. As shown in Figure 11, the application is primarily a main screen that refreshes every five seconds with a new status update from each device, communicated via Bluetooth.

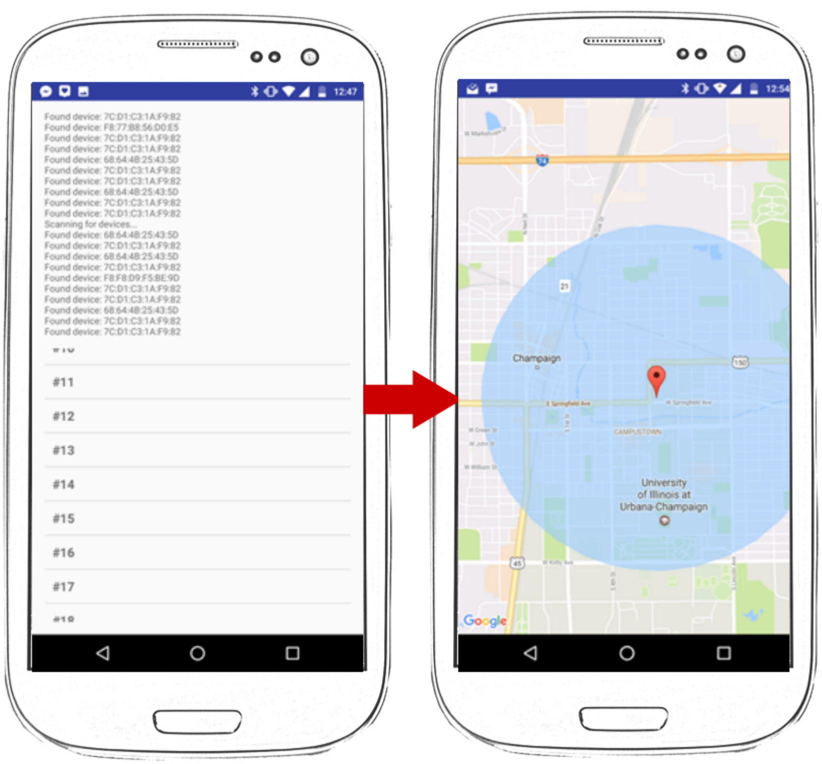


Figure 9: Application prototype with response system

All functions were programmed and implemented with the standard Android Studio and Android SDK, targeted for Android API 23 (Marshmallow).

3 Verification

Guided by our requirements and verification plan (Appendix A), we were able to quantify and verify the function of individual components in the CCTS. The tools we used in assessing these requirements consisted primarily of laboratory tools such as the oscilloscope and multimeter, in addition to heavy use of firmware/software serial monitor.

3.1 Microcontroller

As the core of the project, the microcontoller needed to successfully support several different devices operating simultaneously and interpret data in a way that would be meaningful to all peripherals. Test results for verification steps as described in Appendix A are provided in Appendix C. These tests demonstrate the appropriate clock drift as as well as the ability of the microcontroller to process packets in a very short time. Specifically, the microcontroller was able to process packets in just over 2200 μs on average (figure 12), which is far below our requirement of 0.01129 s. The TDMA (Time Division Multiple Access) multiplexing scheme was a successful choice in allowing multiple devices to communicate in a single period. In figure 13 we see that the interrupt timer maintains a very consistent delay between receiving the packet and transmitting a reply. In this case, we take note that the average delay is similar to figure 12, with an additional $(ID-1)\cdot 0.15$ s delay, meeting the clock drift requirement. We demonstrate in figure 14 receiving a signal (simulated) from more than one child device. The microcontroller is successfully able to retrieve the device ID from the packets received, and as demonstrated in the lab, will consistently receive thirty child devices as specified. Though we did not implement a battery monitor for the child devices, this would have been trivial: connect the battery to an analog input of the microcontroller and include that value in the packet sent to the adult device.

3.2 Sub-GHz Radio Transceiver

The transceiver is essential for detecting whether or not a child device was in range, and was required to support addressability, as well as packetization of information. In our tests, we were able to consistently receive packets (as indicated by the phone app) at up to 150m in ideal conditions (line of sight with the devices positioned at least $\frac{1}{2}\lambda \approx 15$cm from the center of the body and not obstructed by the body) and up to 70m in normal indoor conditions (walls and people obstructing, device near the body). Using the RadioHead libary [9] we were able to easily implement packets and addressing for the transceivers. Using UDP-like unreliable transmission, we detected when a packet was not received properly without creating additional congestion in the form of ACKs and retransmissions.

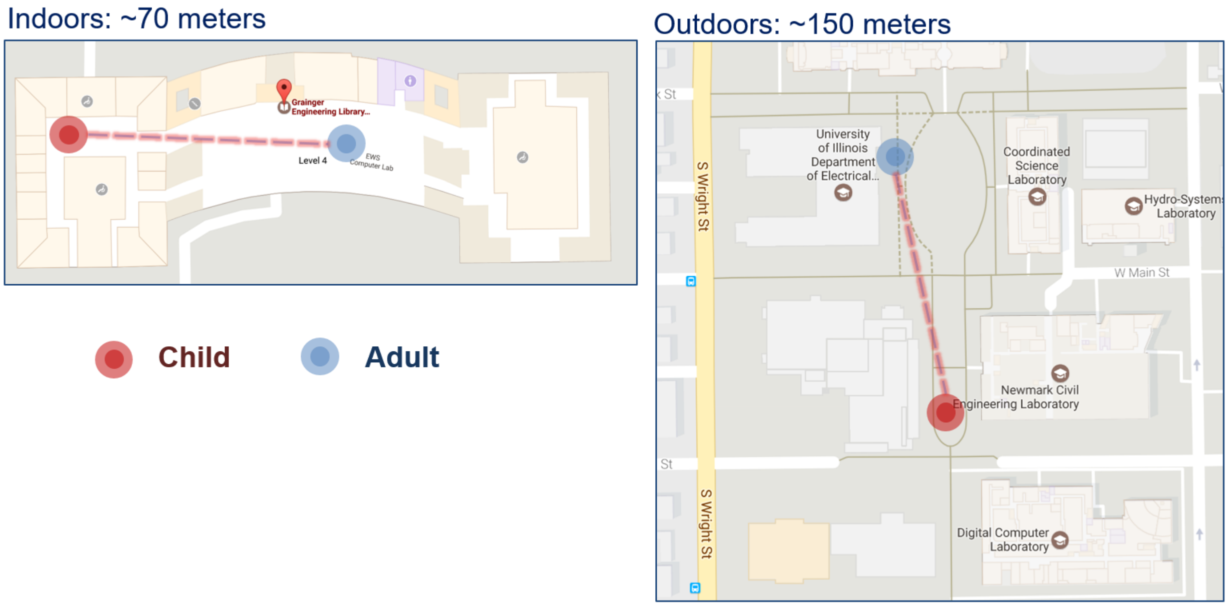


Figure 10: Child and adult communication range testing

3.3 GSM Module

The GSM module is the core of the child device's fallback mode. It was required to quickly connect to the GSM cellular network and obtain cell tower data from the serving cell as well as any neighboring cell tower. By modifying an existing library [10] we were able to add the correct commands to obtain the required data from the SIM800 chip we used. Our code allows the GSM module to boot up in under 15 seconds, and if in range of a GSM tower will connect to it and establish service in that time frame. Our software makes periodic queries to the GSM module, which periodically scans for non-serving GSM towers in range. We transmit this data via SMS to the parent cell phone in a comma-delimited format. As shown in figure 11, the android app parses this data and sends it to the Google API for conversion into a geolocation. Using the ¡longitude, latitude, radius¿ vector returned, we can display the child's location on a map.

3.4 Bluetooth Transceiver

The adult device and the adult cell phone communicate using the bluetooth transceiver. The adult device sends a packet every 5 seconds containing an array of 8-bit integers: 0 indicates a device that has never been seen, 1 indicates that the device has just been observed, 2-9 indicate 1-8 packets missed. Due to the asynchronous updating of these values, 2 may also indicate that the device has just been heard from. While we initially required the bluetooth module to pair with the adult phone, we discovered during testing that the phone could connect to the bluetooth module without pairing. In the future, we would want to ensure for security purposes that the devices are required to be paired, and encryption is used. We were able to observe the duty cycle of the bluetooth transmission by monitoring the chip select pins of the bluetooth transceiver as well as the power draw. We observed a spike during advertisement, connection, and transmission, but we concluded using this technique that the bluetooth module would not draw significant power from the battery.

3.5 Power System

The power system consists of the voltage regulator, the charging of the battery, and power management. In our original design we were going to charge the lithium polymer battery through a charging IC, MAX1551, which would have been powered from wall-wart, 5V output. The IC would monitor the battery voltage and it would have decreased the output current to zero amps gradually as the battery voltage approached rated voltage. Though, when our pcb design did not work correctly, we needed to take a different approach. We found the MCP73831 LIPO Battery Charger which accomplished all the requirements that we wanted. Since it charges the battery in three steps, the last step decreased the rated current by more than 50 percent during the constant voltage trickle charge. Even though we did not design the charging circuit, the charger we found was able to fulfill our requirements. All the devices, battery, voltage regulators, and the battery charger were able to maintain thermal stability below 45 degrees C.

3.6 Android Application

We provided user-friendly status updates from the child devices to the adult supervisor. This was accomplished by having standard pairing protocols for native connectivity to Bluetooth module through Android interface. This can be shown below in Figure 11 which shows the android monitor with received cellular data and converted JSON request.

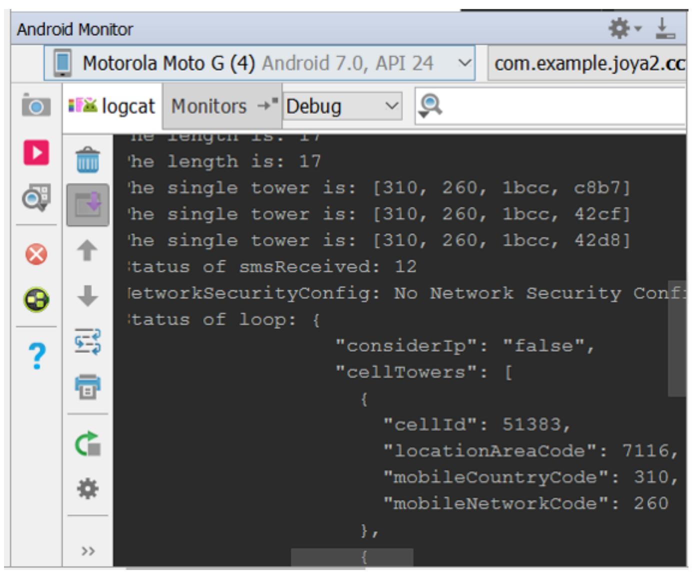


Figure 11: Android monitor with received cellular data and converted JSON request

This successfully shows a pairing of the smart phone with the Bluetooth module and verifies data transfer using serial monitor for microcontroller and app for smart phone. We were also able to run on Android Marshmallow (API 23). We were able to verify this with Android USB debug options to verify the correct system parameters. Lastly, we were successful in accessing Google API and displaying coordinates on a Google map. This was verified by having an alert message on the interface and by isolating the Google map in the application.

4 Cost and Schedule

4.1 Components and Labor

For the prototype version of our product, we list the costs that we have incurred in its production. These prices are *non-bulk*, as these items were purchased for two child devices and one adult device in mind. Moreover, for the convenience of fast prototyping, we have depended on integrated modules and breakout boards, driving up costs for each component. The costs calculated below in Table 1 are for the pcb design of the child device and adult device individually.

Table 1: Cost of Components

Item	Description	Child Quant.	Cost/ Child Unit	Adult Quant.	Cost/ Adult Unit
1	PCB	1	$5.00	1	$5.00
2	SIM800L Module and Antennas	1	$11.99	-	-
3	SIM Card With GSM Service	1	$9.00	-	-
4	Est. Monthly Cell Service Charge	1	$9.00	-	-
5	Microchip MRF89XAM9A Transceiver	1	$7.95	1	$7.95
6	Atmel ATmega328P-AU Microcontroller	1	$3.24	1	$3.24
7	Bluefruit LE nRF8001 Breakout	-	-	1	$5.99
8	Lithium-Polymer Battery	1	$8.95	1	$8.95
9	Battery Charger	1	$9.65	1	$9.65
10	Voltage Regulator	1	$2.20	1	$2.20
11	Quartz Crystal Oscillator	1	$0.95	1	$0.95
12	Various RLC Components	-	$1.5	-	$1.50
			$69.43		**$45.43**

The calculation of labor was made with the assumption that the project spans for 14 weeks and that for every week, an average of 15 hours was spent on the project per teammate.A padding factor of 2.5 was added to the total cost to account for variability.

Table 2: Cost of labor

Name	Rate	Hours	Total Cost
Ellery Tomaszkiewicz	$28.00	216	$15,120.00
Mark Hafter	$28.00	216	$15,120.00
Omar Joya	$28.00	216	$15,120.00
			$45,360.00

5 Conclusion

5.1 Accomplishments

The project was successful in that we were able to implement the core functionality that we proposed in the beginning of the semester. We first specified that the child tracking system needed to track a child device within 50 meters of an adult. Our final design was able to exceed this expectation by tracking up to thirty child devices simultaneously within a 150 meter line-of-sight radius. We also set the goal of being able to connect to the cellular network to collect meaningful telemetry data. After hours of library modification, and debugging on both software and firmware levels, the CCTS was able to collect and project the results in a user-friendly phone application.

The battery life for both the child device and the adult device surpassed our goal of 5 hours. Because of the standby current draw of 40 mA and our decision to use a high capacity battery, we accomplished a battery life of about 12 hours. Altogether, we were able to accomplish these milestones through reiterations of our original design of the system and evaluating what components we needed to focus on most in our limited amount of time. Finally, the team accomplished a better understanding of the design process and intricacies involved with the field of electrical and computer engineering. Through trials of prototyping, debugging, and building, the team gained experience in multi-disciplinary and technical teamwork

5.2 Uncertainties

Despite the successes in our project, there are several uncertainties that would need to be discussed in the future to fit the market for which the CCTS is intended. Throughout our design process, we placed the form factor of the child device outside the scope of the project. We did not address how we can ensure that the child is always wearing the tracking device, which is a critical vulnerability in the system. Several ideas have been introduced, such as a locking system or movement sensing to determine that the device continues to stay with the child.

Another uncertainty involves the accuracy of the location capabilities of the CCTS. We found that in densely populated cities, the CCTS could locate a child within a specific block. However, when tested in a smaller town (Champaign) with fewer cellular towers in the area, the child's location had a radius of about 200 meters. Moreover, we were not able to test the CCTS inside buildings that lack cellular service, leaving another critical vulnerability in the system.

5.3 Ethics and Safety

Because of our use of rechargeable lithium batteries, there are possible safety risks. The thermal stability of materials in the battery at high temperatures and the occurrence of internal short circuits may cause "thermal runaway" [11]. This is a concern because our device is a wearable and any type of high increase in temperature could cause harm, such as burns, to the user. Lithium batteries are safe and heat related failures are rare, but is an important concern to be aware of when designing our device. In accordance with IEEE Code of Ethics, #1 and #9: "To accept responsibility in making decisions... " and "Avoid injuring others..." [12] we will confirm all materials are safe to use, test on humans only when 100 percent certainty of safety, and ensure the well-being and health for the users of our device, the general public, and the environment.

A privacy risk, specifically for the children, to be aware of is that the CCTS can be used to capture the location of the children, in near-real time. The location data needs to be safely unreachable from wanted eyes; Hackers, kidnappers, and other people that could cause harm to the children cannot be able to find this data. We will need to abide by IEEE Code of Ethics, #2: "To avoid real or perceived conflicts of interest..." [12]. The parents of the children will need to be well informed of all concerns related to the CCTS. Moreover, teachers by law must assume the role of "replacement parent," (*in loco parentis*) meaning they have the right of control over the children. The balance between teacher and parent authority regarding child tracking can be guided by the IEEE Code of Ethics, #5, "To improve the understanding of technology; its appropriate application, and potential consequences" [13].

A final primary concern discussed in this section is the assurance of reliable supervision. Because the childś device runs on a finite-energy battery, there may be some concerns about whether the battery could lose power or malfunction during its use without the adult noticing. To mitigate this fear, we are designing our CCTS to ensure that when the child deviceś battery is low, an alert will be sent to the adult smart phone via Bluetooth so that the problem may be remedied. Another concern for protection is the child's ability to remove the device, preventing any proper tracking. Although not in the primary scope of our project, a future iteration of our device would consider this seriously.

5.4 Future Work

The CCTS as presented in this report is only the beginning of a comprehensive and robust means of child safety. The final requirement that we weren't able to accomplish in our project is making the child devices wearable. Given more time, we would have used the knowledge gained from prototyping to make a final design for the PCB components of the CCTS. This would have given us the freedom to consider the physical housing of the child and adult devices. Moreover, we would have liked to test the relationship between transceiver input power and the distance in which the adult can detect a child device. Many of our components were "hard-coded", such as the number of child devices that are tracked and the power fed into the transceiver antennas. In the future, the CCTS could introduce more options for functionality that can be configured with the smart phone application. Lastly, in addition to surveying surrounding cellular towers for geolocation data, we can increase its accuracy by adding wi-fi capabilities to the child device. However, this would come at the greater expense of cost and power consumption, which can be explored int he future.

6 Works Cited

References

[1] N. C. for Missing and E. Children. (2017) Donate: Key facts. [Online]. Available: https://www.missingkids.com/KeyFacts

[2] K. Bilich. (2016) Child abduction facts. [Online]. Available: http://www.parents.com/kids/safety/stranger-safety/child-abduction-facts/

[3] B. Chen. (2016, jul) What's the right age for a child to get a smartphone? [Online]. Available: http://www.nytimes.com/2016/07/21/technology/personaltech/whats-the-right-age-to-give-a-child-a-smartphone.html

[4] P. Lamkin. (2017, feb) The best kids trackers: Using wearables for child safety. [Online]. Available: https://www.wareable.com/internet-of-things/the-best-kids-trackers/

[5] *SIM800h Data Sheet*, 1st ed., SIMCom, https://cdn-shop.adafruit.com/datasheets/sim800h_hardware_design_v1.00.pdf, aug 2013.

[6] *nRF8001 Single-chip BluetoothÂ® Low Energy Solution Product Specification*, 1st ed., Nordic Semiconductor, https://www.nordicsemi.com/eng/content/download/2981/38488/file/nRF8001_PS_v1.3.pdf, mar 2015.

[7] *ADAFRUIT MICROLIPO AND MINILIPO BATTERY CHARGER*, adafruit, https://www.adafruit.com/product/1904, may 2017.

[8] *The Google Maps Geolocation API*, Google, https://developers.google.com/maps/documentation/geolocation/intro, 2017.

[9] *RadioHead Packet Radio library for embedded microprocessors*, AirSpayce Pty Ltd., http://www.airspayce.com/mikem/arduino/RadioHead/index.html, 2017.

[10] *Adafruit FONA Library*, Adafruit, https://github.com/adafruit/Adafruit_FONA, 2017.

[11] L. Florence. (2012) Safety issues for lithium-ion batteries. [Online]. Available: http://www.ul.com/global/documents/newscience/whitepapers/firesafety/FS_Safety%20Issues%20for%20Lithium-Ion%20Batteries_10-12.pdf

[12] (2017) Ieee code of ethics. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html

[13] S. N. Roberts. (2016) Tracking your children with gps: Do you have the right? [Online]. Available: http://corporate.findlaw.com/law-library/tracking-your-children-with-gps-do-you-have-the-right.html

A Requirements and Verification

Table 3: Master list of requirements and verification

Component	Requirements	Verification
Microcontroller (20 pts)	*Must use SPI interfacing and digital logic to facilitate communication of status conditions from child to adult device in 5 s cycles* 1. Maintain clock drift below 0.002258 s per s to prevent collisions between child device transmissions. (5 seconds per cycle, 30 max devices + 1 adult device, 0.15 s allocated to a transmission and 0.01129 buffer on each side of data to avoid collisions, drift over 5 seconds) (5 pts) 2. Identify specific out-of-range child devices from 8-bit ID byte (2 pts) 3. Monitor battery life and in-range conditions with less than 5 s response time (3 pts) 4. Child device: process broadcast packet in below 0.01129 s to avoid miscalculation of transmission timing window. (Signal propagation time is negligible at less than 1 μs). Alternatively, utilize collision-avoidance algorithm similar to 802.11 and verify best-effort reception of all packets within the 5 s cycle. (10 pts)	1. Run a diagnostic program that will print out system clock values every 3 seconds. Using an external source (watch), verify internal clock drift 2. Send a response from two child devices to the adult device and using a serial monitor, verify reception and identification of both packets uniquely. 3. Using a serial monitor, program microcontroller to log begin and end times of battery life finding subroutine, and echo them to serial. Compare times. 4. Simultaneously run adult and child devices with a serial monitor and synchronized external clock, program microcontroller to log transmission and reception times on serial monitor and compare.
Radio Transceiver (8 pts)	*Must be able to maintain "handshake" communication between adult and child devices via analog signal transmissions in packets* 1. Send and receive data in packet format and successfully detect a packet error rate (PER) of about 20% (5 pts) 2. Ignore any packet not addressed to specific device (3 pts)	1. Using serial monitor, program microchip to echo data being transmitted via the transceiver. Check for unique address and monitor accuracy of transmitted packets. 2. Flood channel with packets addressed to a device other than the device in testing, monitor device in testing to verify that it only receiving packets directed towards it.

Lithium-Ion Battery (0 pts)	*Must be able to store enough energy to keep device powered* 1. Maintains thermal stability below 45°C 2. Battery must last at least 5 hrs.	1. Test max voltage and current through battery until fully charged then fully discharged making sure the thermal stability < 45°C. 2. Discharge a full charged battery at full load and make sure it lasts 5 hours.
GSM Module (7 pts)	*Must be able to connect to GSM network in <15s and send location of neighboring towers or location via GPRS* 1. Ready to send within 15 seconds from application of power (2 pts) 2. Retrieve and send neighboring tower data per Google API specifications– be able to send a text message of raw data (4 pts) 3. Successfully connect raw data to Google API for a meaningful response available on the parent application (1 pt)	1. Apply power to device and use diagnostic routine running on Arduino to monitor status. Use diagnostic commands to verify network connection and status. 2. Use diagnostic routine to call "AT+ENGR" command, successfully print an itemized list of returned data on the diagnostic console.
Bluetooth Transceiver (6 pts)	*Must coordinate child device status with smart phone interface with minimal latency (less than 1s).* 1. Deliver results of transmission cycle to adult smart phone every 5 seconds (3 pts) 2. Duty cycle of below 20% to conserve battery (2 pts) 3. Successfully pair Bluetooth module with parent phone (1 pt)	1. Open BlueFruit UART debugging application and monitor update times from Bluetooth device. Also monitor via serial Bluetooth transmit cycles 2. Verify Bluetooth enters sleep mode using microcontroller and serial monitor, and that least 80% of the time it is in sleep mode and not transmitting
Voltage Regulators (0 pts)	*Must be able to output 3.3V to microcontroller or the sub-GHz transceiver and 3.7V to GSM module* 1. Must maintain output voltage within ±5% from a 3.7-4.2V source 2. Maintains thermal stability below 45°C	1. Measure the output voltage making sure it stays within 5% of 3.7V. 2. Draw 100mA at 3.7V and 4.2V input voltages (the two extremes) for awhile making sure it maintains thermal stability below 45°C.

Charging and Power Management (1 pt)	*Must be able to charge the battery to 3.7V-4.2V with a continuous charge* 1. After fully charged battery, IC must lower to 50% of rated current (1 pt) 2. Charging at maximum current and voltage can be sustained below 45°C.	1. Connect IC to battery and fully charge. Test to make sure then current begins to drop towards 0A. 2. Test rated voltage and current through IC and observe the thermal stability of the IC. It should not go above 45 °C.
Software Interface (Android) (5 pts)	*Must provide user-friendly status updates on child devices to the adult supervisor* 1. Native connectivity to Bluetooth module through Android interface. Must have standard pairing protocols (1 pt) 2. Must run on Android Marshmallow (API 23) (1 pt) 3. Successfully access Google API and display coordinates on a Google map via Wi-Fi or network connections (2 pt)	1. Successfully pair smart phone with Bluetooth module and verify data transfer using serial monitor for microcontroller and app for smart phone. 2. Using smart phone Android OS settings, verify system parameters with Android USB debug options. 3. Receive coordinates from the Google API as a simple alert message on the interface. Isolate application to show just the Google map.
Miscellaneous (3 pts)	1. System functions with up to 30 child devices (3 pt)	1. Using an arduino or other suitable control mechanism, simulate the presence of other child devices by sending packets representing those devices' responses to the adult device. 2. Observe functional behavior and no unexpected outages

B Google Geolocation API

The Google Geolocation API drives the conversion between raw cellular data into meaningful coordinates for the user interface. The following parameters are retrieved from the GSM network and are used as inputs for the API via a JSON request.

- *Home Mobile Country Code (MCC):* 310, 311, 312, 313 for the United States [ref]
- *Home Mobile Network Code (MNC):* Varies by cellular network
- *Radio Type:* Our GSM module uses 'gsm'
- *Carrier:* Examples include T Mobile USA and AT&T
- *Consider IP:* This provides the IP geolocation of the requesting device in case cell tower signals are not available
- *Cell Towers:* Provides an array of cell tower objects

A "cell tower object" in its proper JSON format is shown in the code snippet shown below. Evidently, the GSM module will be able to attain this critical information and send it to the master phone to determine a best estimate for the device's location.

```
{
  "cellTowers": [
    {
      "cellId": 42,
      "locationAreaCode": 415,
      "mobileCountryCode": 310,
      "mobileNetworkCode": 410,
      "age": 0,
      "signalStrength": −60,
      "timingAdvance": 15
    }
  ]
}
```

Once the master phone requests an approximate location with the above JSON request, it will receive a message back with estimated coordinates. The formatting of the API's reply is shown below:

```
{
  "location": {
    "lat": 51.0,
    "lng": −0.1
  },
  "accuracy": 1200.4
}
```

C Microcontroller and Timing Verification

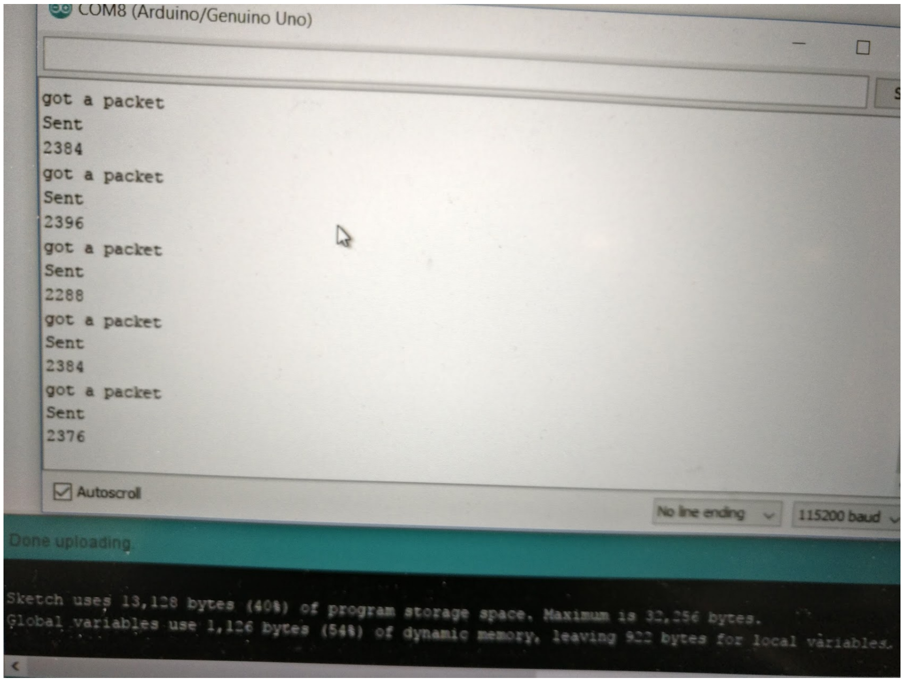


Figure 12: Timing of microcontroller packet processing time: Device ID = 1

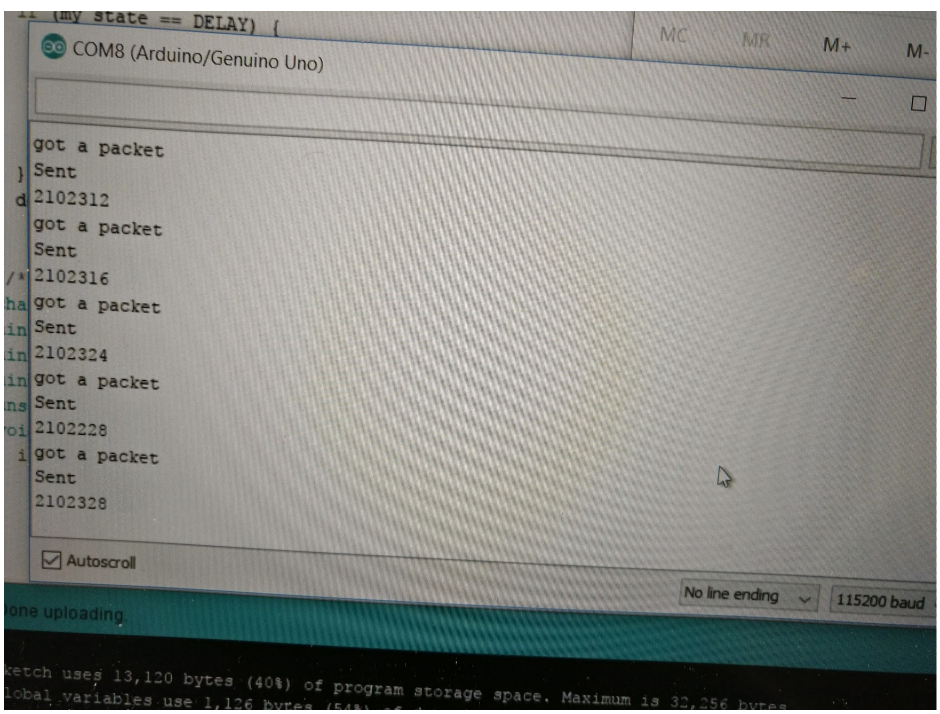


Figure 13: Timing of microcontroller packet processing time: Device ID = 15

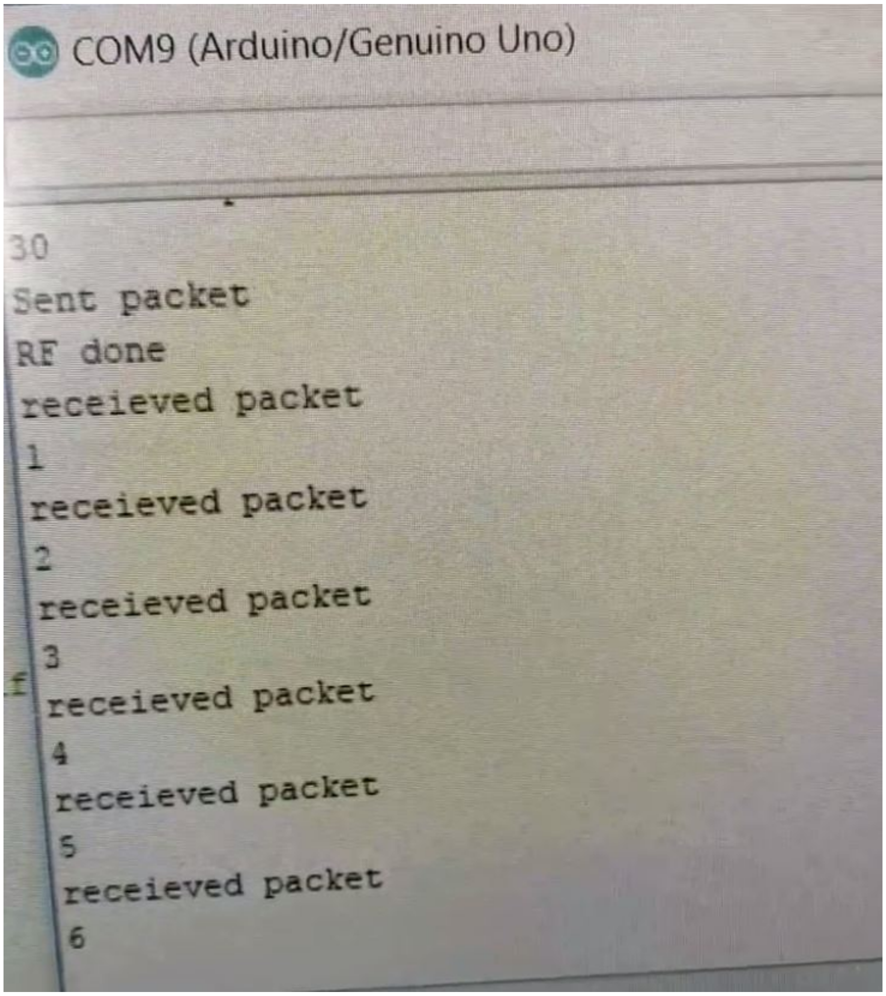


Figure 14: Unique Identification of Child Devices by Microcontroller