SECURE SMART LOCKER FOR DOORSTEP DELIVERY

By

Max Armbruster

Ernesto Marquez

Samarth Jain

Final Report for ECE 445, Senior Design, Fall 2020

TA: Yichi Zhang

9 December 2020

Project No. 8

Abstract

The Secure Smart Locker aims to reduce doorstep package theft by providing a secure receptacle for a courier to conveniently deposit parcels. The device connects to the owner's home Wi-Fi network, allowing the owner to set delivery codes within a mobile app, which are then used by delivery people to unlock the container using a keypad. This report follows the motivation, design procedure, system details, and verification behind our Smart Locker project. By successfully implementing almost every part of the design in our prototype, we were able to create a working framework for the Secure Smart Locker, which, if moved to a sturdier container, allows for the secure and convenient handling of doorstep deliveries. Additional future work is needed to implement and verify every feature in our design.

Contents

1. Introduction

Our project tackles a very interesting and critical problem that plagues our society today. In New York City alone, roughly 90,000 packages are stolen or disappear without an explanation every day [1]. The Secure Smart Locker is designed as a new security system for homeowners during the COVID-19 pandemic, when home deliveries have been on the rise [2]. This increased concern about package theft [1] is more of a real issue now than ever before. This project can connect to their own Local Area Network WIFI router and download the Mobile Application on their Android Device. While using the application the owner will be able to create a new order with a random numeric code that only the owner and the delivery driver will have access to. Whether it is from USPS, UPS, Amazon etc., they will be able to place the package in the secure Smart Locker and have video footage saved for the owner to be able to see.

The entire project consists of four different modules, Power Module, WIFI Module, IO Module and Control Module. Power Module has the external battery holder and power protection and conversion subsystems. WIFI Module has the filter, antenna, and LAN. IO Module comprises of the main hardware components which are lock, speaker, indicator lights, camera, keypad, and sensor. Finally, the Control Modules just contains the MCU and the Mobile Application. Upon coming up with this design the performance requirements that we assigned from our initial proposal, the high-level requirements were:

- Our locker should use a number pad to take a passcode input from the delivery driver, unlocking temporarily when the correct passcode is given. The passcode will be randomized and will last only as long as the delivery is "in progress". Once the delivery code is no longer needed, the box will not open with that code in the future.
- Our locker should use a speaker to give an audible indicator when the door is left open, and double as an alarm when the locker is unplugged from the wall or when the motion sensor detects malicious activity. The alarm should trigger within 3 seconds of the malicious activity taking place to give the owner adequate time to hear and respond to it.
- Our solution should include a mobile app that allows the owner to remotely unlock the container and receive delivery/pickup notifications. The notification should arrive no more than 10 minutes after the delivery/pickup in order to give the owner accurate information regarding their package's status. Remote unlocking should take no more than 5 seconds in order to keep this feature convenient.

The device will bring security and comfort when your delivered packages have a secure place to be stored. The following contents of the upcoming chapters will describe the design and the verification of the subsystems of the modules, the bill-of-materials of the project, and the conclusion of our findings while designing and finalizing the project.

2 Design

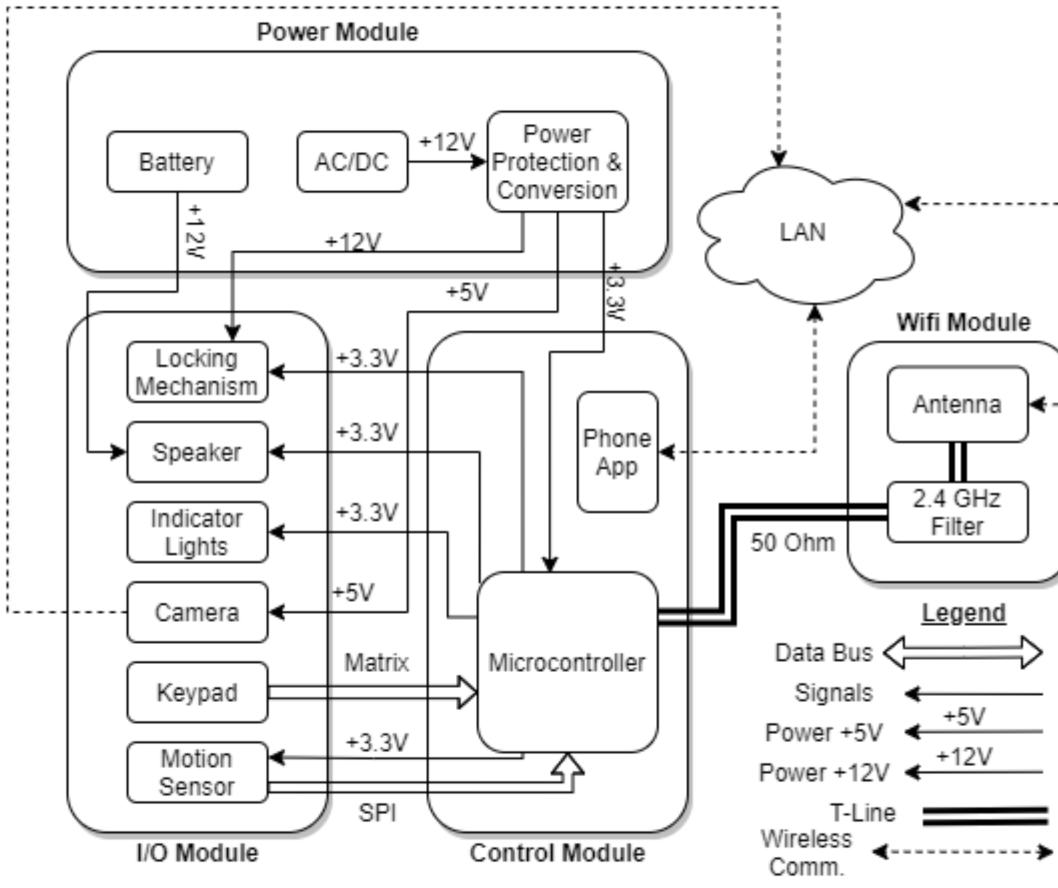


Figure 1: Block Diagram of Smart Locker System

Figure 1 shows the block diagram of our locker system. Our system consists of four primary modules: Power, Control, I/O, and Wi-Fi.

2.1 Power Module

The Power module is responsible for distributing safe and steady power to the whole system.

2.1.1 Battery

The Battery is used to power the alarm when the power supply is disconnected. Design of our Battery submodule came down to the component selection for the battery and the connection of the battery to the PCB. Our original intention was to use a rechargeable +12 V lithium-ion battery since we were familiar with this technology and knew that we could find high-capacity batteries of this kind at our desired voltage. Since we knew most power supplies used a 2.1 mm x 5.5 mm barrel connector, we wanted to see if we could find a battery with the same interface, as this would allow us to use the battery and power supply interchangeably. We found few li-ion batteries with this connector, but we did find several battery casings that did. To save money we decided to buy AA batteries and place these in a battery casing that connects eight of them in series to supply 12 VDC. We chose a casing that was relatively cheap and featured an On/Off switch that would allow us to disable the alarm without unplugging the battery every time. To ensure that AA batteries would provide us with the lifetime to make our alarm feature reliable, we calculated the capacity of the battery pack in Equation (1), using 2500 mAh as the typical charge in an AA battery. We then used the current draw of the speaker to calculate the lifetime of the battery pack in Equation (2). We determined that 2,000 hours of continuous use would be sufficient since the alarm should only be used in emergencies.

$$8\ Batteries \times \frac{2{,}500\ mAh}{Battery} = 20{,}000\ mAh \tag{1}$$

$$\frac{20{,}000\ mAh}{10\ mA} = 2{,}000\ hours \tag{2}$$

2.1.2 AC/DC

The AC/DC conversion for our Power module takes 120 VAC power from a typical home power outlet and converts it to 12 VDC for the PCB's power input. One-piece power supplies of this type typically use 2.1 mm x 5.5 mm barrel connectors, which we can easily interface with our PCB using a thru-hole barrel jack. Once we selected most of the components for the rest of our system, we were able to estimate the maximum current draw and determined that a 2 A supply would be sufficient. We also required that this supply would be relatively stable, as large fluctuations could trip our overvoltage protection. The supply we chose guaranteed a maximum +/- 300 mV deviation from 12 V.

2.1.3 Power Protection & Conversion

The Power Protection & Conversion submodule converts the 12 V power input to 3.3 V, while protecting our PCB from overvoltage, overcurrent, and reverse voltage inputs, all of which could damage our board. Figure 2 shows the general form of our power protection circuit. Our design combines a thyristor crowbar circuit with dual P-type MOSFETs for overvoltage and reverse voltage protection, respectively. A negative voltage presented at either the input or output of this circuit will cause the gate voltage of the FETs to exceed the source voltage, placing them in the cutoff range and blocking any current from flowing. Zener diode D1 is used to clamp V_{GS} to the operating voltage of the FETs that will achieve the smallest possible $R_{DS(ON)}$ (10 V in our case), which will also give us the smallest voltage drop across the FETs. Zener D2 blocks any current from flowing to the gate of the thyristor until the output voltage of the circuit exceeds the Zener breakdown voltage, which we choose as our desired maximum voltage output (13 V). When this value is reached the thyristor will trip and short the input of the circuit to GND, causing a very large inrush current that will blow fuse S1, isolating our board from the dangerous voltage. We chose the current rating of the fuse to be 2 A, allowing us to also use it as overcurrent protection even when presented with a safe voltage input. The input of the protection circuit is wired to the barrel jack that our 12 V power supply connects to, and the protected output runs to both our lock and our 3.3 V converter. The conversion is done using a RECOM voltage regulator, which is rated up to 1 A of current, more than the estimated current draw for our 3.3 V components.

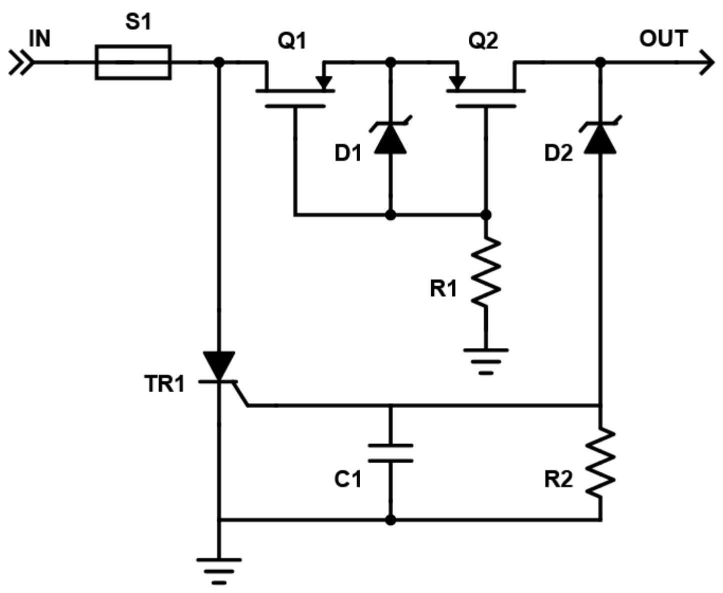


Figure 2: General form of our power protection circuit. The thyristor crowbar protects against overvoltage, while the P-type MOSFETs protect against reverse voltages. The fuse S1 is used for overcurrent protection, and as part of the crowbar circuit.

2.2 Control Module

The control module consists of the Microcontroller Unit (henceforth referred to as the MCU), and the Android mobile application. These two components are critical for the control of the entire project, with the MCU controlling the I/O from the mailbox, and the mobile app controlling the I/O from the phone's user interface. The app and the MCU communicate with each other over a WLAN TCP connection, with the MCU behaving like a TCP server and the app behaving as a TCP client. Both the app and MCU retain some common state, such as the active deliveries and the pseudo-randomly generated 6-digit codes for each delivery.

2.2.1 Microcontroller

The Texas Instruments CC3200 MCU sits in our PCB and has its GPIO pins exposed to the peripheral devices (refer to the I/O module) [3]. The MCU is responsible for communication with the app over TCP, taking the input from the numpad, camera and the accelerometer, and sending digital signals to the LEDs and speakers when necessary.

The MCU runs on a Finite State Machine, with 6 essential states of its program, as shown in the figure 3:

1. Start: The MCU starts in this state after powering up and remains in it until connected to the local Wi-Fi connection.
2. Client-Wait: The MCU is connected to the Wi-Fi and has its TCP connection set-up, but it waits for a TCP client to request a connection to it.
3. Reset-Input: The MCU is connected to a TCP client, or the mobile app. Now, it is in its stand-by mode, where it waits either for messages from the app, or from the I/O module.
4. Accepted-Code: Once a 6-digit code is accepted by the MCU, it needs to check if that 6-digit code is one of the pending delivery codes. If yes, then the control goes to the Accepted-Code

state, where the green LEDs are flashed, and the speaker is sounded to inform the delivery man that the box is open.

5. Wrong-Code: If that 6-digit code is wrong, then the app must be notified in case of a potential security threat. This state handles the code for a wrong code typed in.
6. Box-Unlocked: This state signifies that the mailbox is unlocked, and it will remain unlocked and the MCU will remain in this state until a '#' key is pressed.

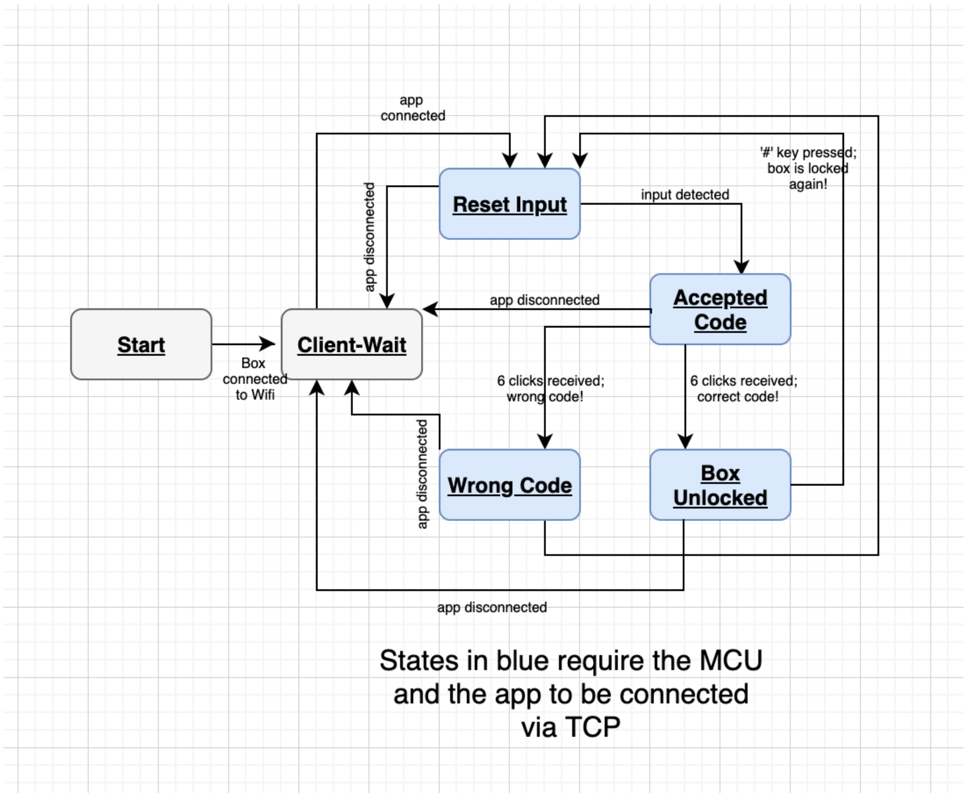
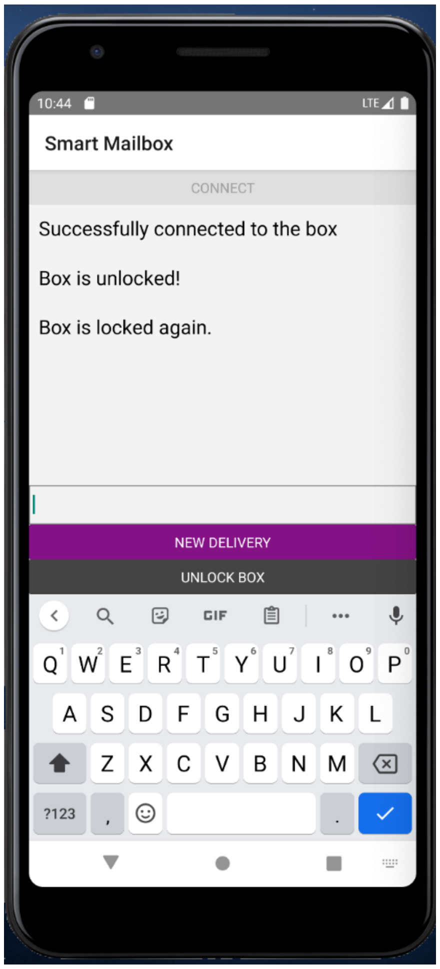


Figure 3: Finite State Machine of the MCU

2.2.2 Phone App

The phone app lets the consumer of our mailbox control the I/O from his end. The app behaves like a TCP client in its connection with the MCU and sends and receives messages over TCP with the MCU.



Successfully connected to the box

Box is unlocked!

Box is locked again.

Figure 4: A screenshot of the app

The app allows the user to connect to the MCU by hitting the “Connect” button, as the figure 4 shows. Once the two devices are connected, the communication can successfully take place. The user would create a new delivery by giving that delivery a name, and then a pseudo-random 6-digit code would be generated for that delivery. As soon as the delivery is created, the app stores that code as one of its pending delivery codes. It also sends a “ND” message to the MCU along with the 6-digit code to inform the MCU about the new pending delivery. It can handle up to **five** active deliveries.

On the MCU side, when a delivery is completed, the MCU sends a message such as “LOCKED” with the 6-digit code that was used to unlock the box. This code is then removed from the acceptable list of codes forever, by both the app and the MCU.

The app also allows the user to unlock the mailbox on his demand by pressing the “Unlock Box” button. This will send an “OPEN” message to the MCU and send the MCU into the “Box-Unlocked” state. A few examples of the TCP messages sent between the two control components are shown in the figure 5.

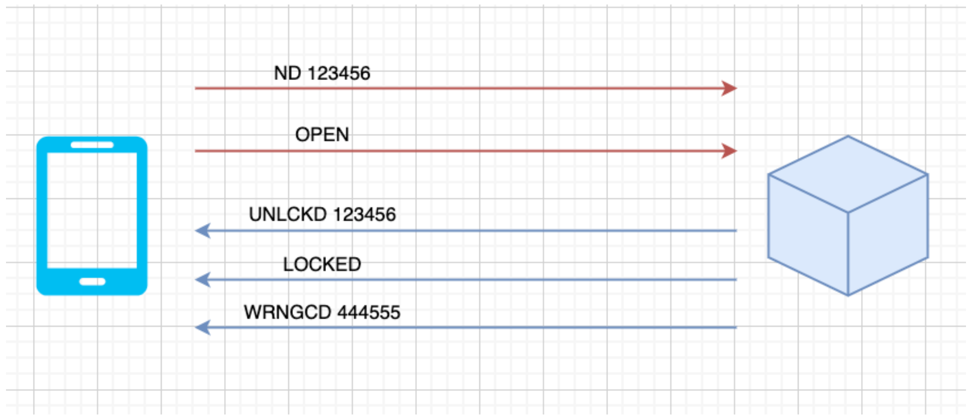


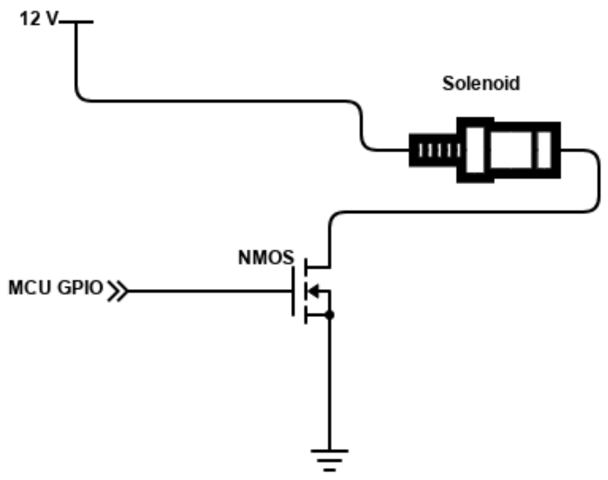
Figure 5: TCP communication between the devices

2.3 I/O Module

Our I/O module consists of inputs like our sensors and keypad, and outputs like our lights, speaker, and lock. These devices all interface with the control module.

2.3.1 Locking Mechanism

The locking mechanism is needed to securely lock down the box. A mounting lock is preferred, to mount the lock inside of the box. A pull-down type is preferred to save energy and have an automatically locked. To activate the lock, in the Power Conversation and Protection, it converts AC wall power to +12V power which powers the lock to be pulled. According to the Figure 6 signal when we want the lock to activate there is a NMOS, with a Vgs threshold of 1V. Perfect to use with GPIO signals which are about +3V. One side of the terminal for the lock is connected to +12V while the other side is connected to the drain terminal of the NMOS which is not connected to anything. If a signal is sent to the gate terminal from GPIO then the source would be connected to the drain which the source is connected to ground. Thus, making a complete circuit connection unlocking the lock. This subsystem is preferred for increased security, a complete and closed frame lock, to avoid any damage and tampering of the lock. Using an MCU and connecting to pin, to send Digital/Analog write to send signal to lock relay to send power from



wall to lock.

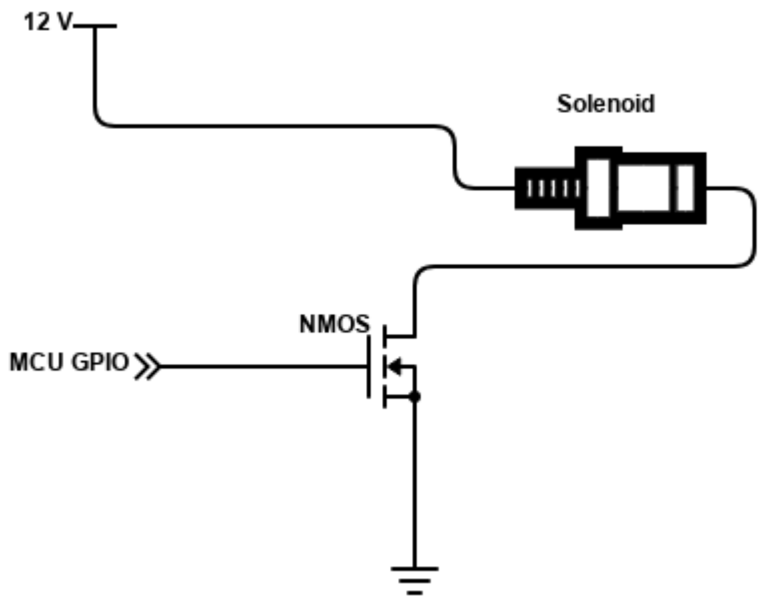


Figure 6: General form of our speaker circuit. The NMOS switches between indicator power from the PPC when the GPIO signal is sent.

2.3.2 Speaker

The Speaker submodule serves three purposes, acting as an indicator tone for locking and unlocking box, acting as an indicator tone for the Number Pad that is attached to the box, and as a very loud alarm when the wall ac adapter is disconnected. The way that it functions is that there is a general-purpose relay that can take any voltage from 3 to 24 VDC, as long the relay itself is turned on from 3VDC. Pin 1 is for the +12V from, Pin 10 is the GPIO Pin indictor, Pin 2 and 9 is to turn on the relay with 3VDC, and the Pin 5 & 6 is the output connected to the speaker. If the wall plug is connected, then the switch is connected to the GPIO to send signals for the first two purposes of the speaker. There are also two resistors to step down the voltage and not make the indicator beeping so high. If the wall plug is immediately disconnected, then there will be no power converted from Power Conversion and Protection Module that is, the AC wall power to +12VDC then to a DC-DC step down to +3VDC which is needed to turn on the relay, as shown in the figure 7. In this case then the relay will then connect to the +12VDC from an external battery to sound the loud alarm, third purpose.

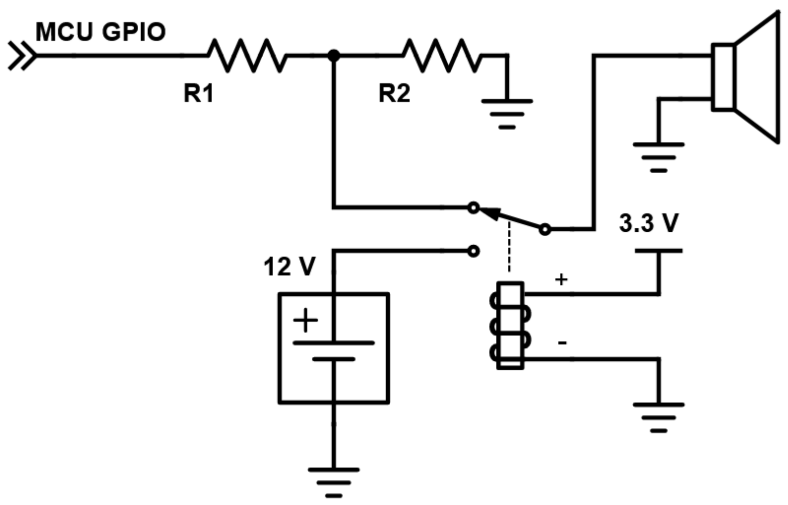


Figure 7: General form of our speaker circuit. The SPDT relay switches between indicator power from the MCU when the locker is plugged in and alarm power from the battery when the locker is unplugged.

2.3.3 Indicator Lights

The indicator lights are external LEDs mounted to the top of the locker, switched by the MCU to indicate the current system state. The yellow, blue, and green colors were chosen randomly by taking whatever was available to us in the lab. We added 100 Ω resistors in series with each LED to limit the current draw from our GPIO pins.

2.3.4 Camera

The Camera subsystem provides us with a live video feed of the locker's front side. We originally intended to use a device that would interface with the MCU, which would then send the video feed to the app over LAN. Since the MCU would only be passing the data along, we opted to use a camera with built-in Wi-Fi connectivity instead. This reduces the complexity of our system and increases our component options, as Wi-Fi cameras are quite popular. The camera we chose is compact and accepts a 3.3 V power input from our PCB.

2.3.5 Keypad

The keypad subsystem is very simple in principle. Connecting each pin from the Keypad to a GPIO pin of the MCU. It is preferred that the keypad had a variety of different characters. Numbers from 0-9, Letters from A-D, and Star Pound keys. This increases the complexity of the code to enter when trying to unlock the box. Having a thin keypad with small area to take less space and less bulky looking. To code the keypad, like most keypads, they are like a pull-down row and column to determine what key is pressed using a while loop in the code. Each pin with their thin and small resistive wires can take voltage from the GPIO and vice versa of +3V.

2.3.6 IMU Motion Sensor

The motion sensor with a QFN packaging designed on the PCB board. Component will be able to measure accelerometer and gyroscope in 3 dimensional positions. Provides the information that would be needed to read when the secure box is either tilted or forcefully moved to a different position. An additional layer of security.

2.4 Wi-fi Module

Our Wi-fi module allows our MCU to communicate with the phone app over LAN.

2.4.1 Antenna

The antenna in our Wi-fi module receives and transmits Wi-fi signals in the form of EM waves. We chose to use the recommended component in our MCU’s datasheet. Using a surface-mount antenna allows us to use a very short PCB trace as the only interconnect between the device and our MCU, minimizing the high-frequency parasitic effects that reduce the performance of the antenna.

2.4.2 Filter

The 2.4 GHz filter selects only the frequency our system uses for Wi-fi communication and attenuates any other bands that the antenna might pick up. Like the antenna, a surface-mount device is ideal for this application. The MCU’s datasheet recommends a good SMD part that we selected for use in our system.

3. Design Verification

While we did perform successful verifications for most of our submodule requirements, there were several submodules which we either could not implement in our final design or could not properly test due to time and lab access restrictions. Our time restraints resulted from several late PCB revisions intended to fix critical errors in our board, the final version of which we did not receive until the week of our demo, leaving us with one lab day to build and test most of our project.

3.1 Power Module

We were able to verify the stability of our 12 V power supply and battery pack but could not complete a full verification of the Power Protection & Conversion. While this submodule did provide us with stable current at the correct voltages, we did not have time after it was built to verify that it isolates the board from unsafe power inputs. We prioritized PCB assembly in the lab, and once it closed, we lost access to a variably power supply that we needed to test unsafe voltages.

3.2 Control Module

Unfortunately, we were not able to flash our code to the MCU on our PCB. While we were not able to pinpoint the precise reason why the MCU did not respond to our flashing attempts, there was likely a soldering issue which left certain MCU pins either unconnected or shorted to each other. The CC3200 uses a QFN package that required us to solder it using a heat gun reflow process. No member of our group was familiar with reflow soldering, and we were not able to get help with the procedure given such a short assembly window.

As a backup plan we used the CC3200 Launchpad [4] from Texas Instruments in conjunction with our PCB, both mounted to the underside of the locker’s lid. We were ultimately able to bypass the CC3200 MCU on the PCB by using the same MCU on the Launchpad. The rest of the connections were the same.

The app and the MCU successfully connected to the local Wi-fi and to each other using TCP over a WLAN connection.

3.3 I/O Module

There were successful and unsuccessful verifications of the subsystems in the IO Module. The lock and speaker were able to be verified for their appropriate purpose before placement on the PCB, as well as the indicator lights and the keypad. We swept the voltage power supply from the lab we were able to verify the lock with range 0-12 V which pulled the lock. We swept the voltage power supply from the lab for the speaker with range 0-12 V providing 105dB. Realizing that the 105dB at 12 VDC was significant, at 3VDC it was louder than anticipated. Designed and applied a voltage divider with 100 Ohm to reduce the signal about half, 1.5 VDC.

However, lock and speaker weren’t completely verified and tested on the PCB due to the multiple revisions/orders of a new PCB, it created delays. Needed to include an external way of connecting to the signal pins for both lock and speaker with an external MCU already build launchpad. Lock and speaker have their own relay/switch when certain situations arise as described in the design chapter. It was not test because of the limited time and needing to solder the through hole general relay for the speaker and the NMOS for the lock.

Furthermore, there were unsuccessful verification with the Camera, IMU, and the MCU itself. The purpose of the camera was to be completely wireless from the control module and be able to provide feed via WLAN. Unfortunately, as the project progress with long hours, the ribbon of the camera tore and was unable to be tested or demoed.

Finally, the IMU and the MCU reflow soldering is a challenging task. Needing a heat gun to reapply tin solder and adjust the solder led to unfortunately burning the chip, shortening pins, and creating unsuccessful connections of the pins from the MCU.

3.4 Wi-fi Module

Because we were not able to flash our code to the MCU on our PCB, we have no way of testing the antenna and filter on our board. Our Wi-Fi communication uses the antenna and filter on the Launchpad, as our code is running on the Launchpad MCU.

4. Costs

4.1 Parts

Table 1 Major Parts Costs

Part	Manufacture	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
IMU	TDK InvenSense	8.69	8.69	8.69
Number pad	Parallax Inc.	6.50	6.50	6.50
Speaker	PUI Audio Inc	4.08	4.08	4.08
Lock	SparkFun Electronic	9.95	9.95	9.95
Camera	SparkFun Electronic	43.75	43.75	43.75
Launch Pad	Texas Instruments	29.99	29.99	29.99
Serial Converter Module for UART	Izokee	6.99	6.99	6.99
Hex Standoffs	Small Parts	4.42	4.42	4.42
12V AA Battery Holder	CO RODE	8.12	8.12	8.12
12V Wall Power Adapter	TMEZON	7.99	7.99	7.99
M3 Screws	Prime-Line	2.19	2.19	2.19
AA Batteries	Amazon-Basics	6.49	6.49	6.49
Trash Bin	Walmart	20.00	20.00	20.00
Total		**159.16**	**159.16**	**159.16**

Table 2 PCB Hardware Components and Passive Components

Part	Manufacture	Actual Cost ($)
Resistors, Capacitors, Inductors, Fuses, MOSFET, Relay, Power Jack etc.	Variety PCB Components Manufactures	$52.77
Total		$52.77

4.2 Labor

$$Cost\ of\ Labor: 3\ Engineers * \frac{\$50}{hour} * 20\frac{hours}{week} * 10\ weeks = \$24{,}000$$

5. Conclusion

5.1 Accomplishments

Our group successfully implemented our Battery, AC/DC, Power Protection & Conversion, Locking Mechanism, Speaker, Indicator Lights, Keypad, and Phone App submodules. We were able to control each of these with our code, running on a CC3200 MCU and communicating with the phone app via local Wi-fi. In doing so we created the framework for a Smart Locker system which, when placed in a much sturdier container, allows for the secure and convenient handling of doorstep deliveries.

5.2 Challenges & Uncertainties

Unfortunately, due to unexpected PCB revisions, we only had one day in the lab to assemble our PCB and test many of our submodules. As a result, we were not able to successfully solder our MCU and IMU to our PCB, having to use the MCU on our CC3200 Launchpad board to run our code, which used a separate Wi-fi module to communicate.

If we had more time and soldering assistance, we are confident that we could fix our MCU problem and run our code on the PCB that we designed, without the need for the Launchpad. Even though we cannot verify this without testing our MCU and Wi-fi circuitry directly, we designed all of that by following the recommended setup in the CC3200 datasheet, and it was nearly identical to the circuitry found on the Launchpad.

On the other hand, given our collective inexperience working with IMUs, we are not totally confident that our motion sensor code would have worked straight away even if we were able to solder it. We would likely need more development time to learn about the device and debug our code before we could bring that submodule online and add even more security to our Smart Locker.

Additionally, due to the limited lab time and a broken ribbon cable on our camera, we were not able to test the Power Protection & Conversion or Camera submodules. While it is possible that the former is faulty, our informal tests showed that our board never lost power unexpectedly, with both our 12 V and 3.3 V nets showing the correct multimeter readings. The camera itself required no design on our end as it was a single component that we bought. Had we not broken it, we can safely assume that it would have worked within the specifications set and tested by the manufacturer.

5.3 Ethical Considerations

We will take the utmost care in complying with the ethics and safety standards that IEEE and ACM have laid out. To comply with the ethics and safety standards, this project will be of a high quality, and the processes and products will derive from high-quality professional work in accordance with the ACM "Professional Responsibilities" section 2.2. Our team and its individuals will take "personal and group responsibility for acquiring and maintaining professional competence" [5].

Confidentiality and privacy of data is a very critical detail to keep in mind for any project. Our project will comply with the confidentiality of the transferred data by making the connections between the app and the MCU secure. We used a TLS protocol over our TCP connection to ensure that no "man-in-themiddle" attacks could happen with our product. Any connection made between the app and the MCU will be over secure TLS, which would encrypt any data over the communication line. This adheres to the ACM standards on security ethics, as we "secure resources against accidental and intentional misuse, modification, and denial of service" for our project [5].

Power from the battery could potentially cause a safety hazard, if care is not taken. The project will comply with the safety standards concerning the power from the battery. The Lithium-Ion battery must never be charged or kept in extreme temperatures (below 320 F or above 1300 F), or it may be damaged or even explode [6]. Utmost care will be taken during this project to ensure that the batteries are properly inspected and in good condition to be used, thus adhering to the IEEE code of ethics stating "to hold paramount the safety, health, and welfare of the public" [7].

Finally, the project will ensure that the numeric code generated by the app will be local to the user's phone, and that the information will not be divulged anywhere else other than to the delivery service, by the user himself. Additionally, the numeric code will be randomized for every new delivery, and confidentially stored in the user's app storage. When the delivery is done, that numeric code will be discarded, to ensure that the same code cannot reopen the box again. This will adhere to the ACM standards on confidentiality, which state that "the nature or contents of that information should not be disclosed", according to the ACM guidelines in the section on respecting privacy, section 1.7 [5].

Overall, we believe that, upon sticking to the design choices according to this document, we pose no ethical issue or safety threat to the user.

5.4 Future Work

Our project achieved the basic high-level requirements that tackled the problem against package theft. But, to improve the project iteratively, we believe that a few additions would really take the project to the next level of convenience and robustness.

Our power supply. Currently, the power from the batteries is consumed non-stop, and there is no process to recharge them or let the user know of their status. We would like to add a feature where we could trickle-charge the secondary power supply from the primary power-supply, thus allowing the secondary power supply to last a lot longer than in the current scenario. The communication between the app and the MCU occurs on a WLAN TCP connection. Currently, the system only allows one TCP server and one TCP client. The next iteration of the project should ideally allow the system communication even remotely, such that a user of our system could check the status of his mailbox while away from the WLAN. For this, we would use a remote server which would connect to both the MCU and the app over Websockets, and provide the connection between them, thus removing the possibility of disconnections.

Our mailbox currently has no provision for temperature control. We believe that adding smart temperature control to our mailbox would be a very convenient feature, as it would allow groceries to be delivered into our box and kept fresh as well.

References

[1] M. Haag and W. Hu, "90,000 Packages Disappear Daily in N.Y.C. Is Help on the Way?," 3 December 2019. [Online]. Available: www.nytimes.com/2019/12/02/nyregion/online-shopping-package-theft.html.

[2] A. Berthene, "Online merchants gain an extra $107 billion in 2020 thanks to pandemic," 2020. [Online]. Available: www.digitalcommerce360.com/article/coronavirus-impact-online-retail/. [Accessed 17 September 2020].

[3] Texas Instruments, "CC3200 Datasheet," [Online]. Available: https://www.ti.com/lit/ds/symlink/cc3200.pdf. [Accessed 17 September 2020].

[4] "Texas Instruments, CC3200 Launchpad," [Online]. Available: www.ti.com/lit/ug/swru372c/swru372c.pdf. [Accessed 20 November 2020].

[5] "ACM Code of Ethics and Professional Conduct," 2020. [Online]. Available: www.acm.org/code-of-ethics. [Accessed 17 September 2020].

[6] "Safety and Health Information Bulletins | Preventing Fire and/or Explosion Injury from Small and Wearable Lithium Battery Powered Devices | Occupational Safety and Health Administration," [Online]. Available: www.osha.gov/dts/shib/shib011819.html. [Accessed 17 September 2020].

[7] "IEEE Code of Ethics," 2020. [Online]. Available: www.ieee.org/about/corporate/governance/p7-8.html. [Accessed 17 September 2020].

Appendix A Requirement and Verification Table

Component	Requirement	Verification	Verification status
Power Protection	1. Must accept a voltage input of 12V +/- 300mV 2. Must protect against overvoltage of 13V+ 3. Must protect against reverse voltage inputs of -1V or less. 4. Must protect against input currents of 5+ Amps	For each step below we will connect our input to a variable power supply and do the following, replacing the fuse each time it trips: 1. Sweep the voltage while maintaining a small current to ensure that the SCR and fuse do not trip due to these voltage inputs. 2. Continue (1) to ensure that the SCR and fuse trip before reaching 13V. 3. Sweep the voltage while our leads are connected in reverse, ensuring that the SCR and fuse trip before reaching -1V. 4. Set the voltage to 12V and slowly increase the current to ensure that the fuse trips before reaching 5A.	No, as our limited lab time prevented us from testing the completed PCB with a variably power supply, so we could not verify that the fuse in our protection circuit would blow for dangerous inputs.
Control	1. The MCU is functional 2. The Launchpad (LP) MCU connects to the local Wi-fi 3. The app connects to the local Wi-fi 4. The LP MCU connects to the app using TCP connection 5. The LP MCU and the app pass messages to each other on their TCP connection when key events happen	1. The MCU is able to boot load the code on its memory, and runs a simple "blinky" program. Unfortunately, the MCU was not able to load any code, because of soldering issues. 2. The blue LED on the mailbox turns on. It is programmed to turn on when the LP MCU is connected to the Wi-fi but not the TCP client. 3. If the mobile phone is connected to the same Wi-fi connection as the LP MCU, then the app is as well. 4. Hit "Connect" button on the app. The TCP connection is made, and the LP MCU acts like a TCP server, and the app	The control system worked (because of a fallback plan with the TI Launchpad)

		acts like a TCP client. Printing out the TCP message using UART interface confirms and verifies this. 5. The app sends an “ND” messages with the new delivery code, to the LP MCU. The LP MCU understands the message and stores the delivery code in its acceptable list of codes. Printing out the TCP messages using UART interface confirms and verifies this.	
I/O - Lock	1. Controlled via MCU signal gating the +12V power 2. Fail Secure (locked when powered off)	1. Toggle MCU control signal and observe correct lock behavior	Couldn’t send signal from MCU Jumped +3.3V to FET gate instead Works as intended
I/O – Speaker	1. Makes soft indicator sound when controlled by MCU 2. The speaker correctly produces a much louder sound when the box is disconnected from the wall outlet.	1. Code is typed and when code is correct makes small sound. 2. The power connection to the wall is taken out, loud sound is made.	Speaker very loud with +12V Too loud for “soft” indicator @ 3.3V Set up voltage divider to cut the indicator voltage in half Result was reasonable volume
I/O – Indicator Lights	The lights turn on and off according to state of the Finite State Machine of the MCU	The LEDs behave in the following manner: 1. Green LED: flashes green when the box is unlocked 2. Yellow LED: indicates that the MCU is in a reset state 3. Blue LED: indicates when the	The indicator lights functioned the way they were supposed to

		MCU is connected to the internet	
I/O – IMU – Motion Sensor	The accelerometer sends data to the MCU	Use simple code to test by retrieving data from the IMU to the MCU. When *unnatural* motion by triggering an alarm and sending data to the mobile app.	Unsuccessful. Soldering difficulty, and simple code no reading
I/O – Keypad	The numpad accepts the code typed in and sends the data to the MCU to process.	Run simple code to test a single character. Connect GPIO from MCU to keypad, connections create the correct 6-digit code in the MCU.	Running a simple code to test a single char, successful. Implemented all the rest of the characters.
I/O – Camera	1. Needs an antenna that can reach high frequency (2.4Ghz or 5Ghz) for internet connection 2. It will need to be powered from the Microcontroller with +3.3V	1. Connects to Local WIFI 2. Use the multimeter and the oscilloscope to measure whether the voltage and the current going to the camera are appropriate	We unfortunately broke the ribbon cable from the camera to the board
Wi-Fi	1. Antenna and Filter will be able to make a clean connection with WLAN network nearby	1. Compile and upload simple Server-Client code to view and connect with WIFI.	Unfortunately, no way to test since MCU not functional.