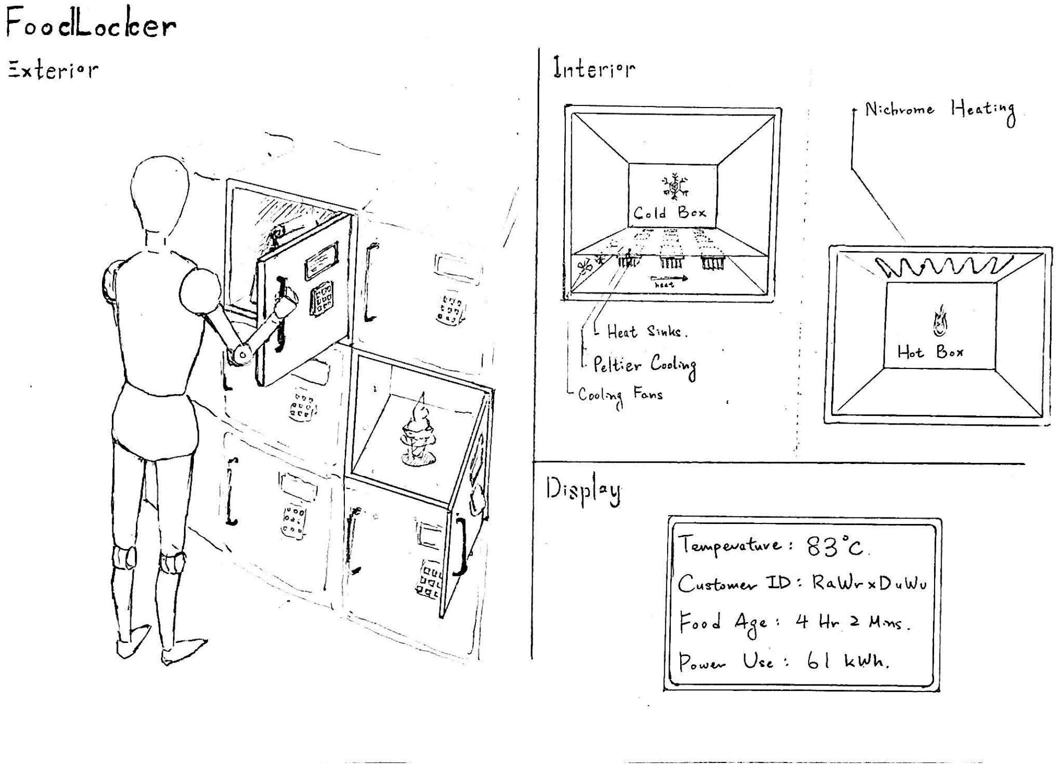
Food Locker

The Ate-am
Sharan Ramjee, Ninad Tummapudi,
Reid Domingo, Mao Huan Huang

Food Locker - Pitch and Sketch



User Story

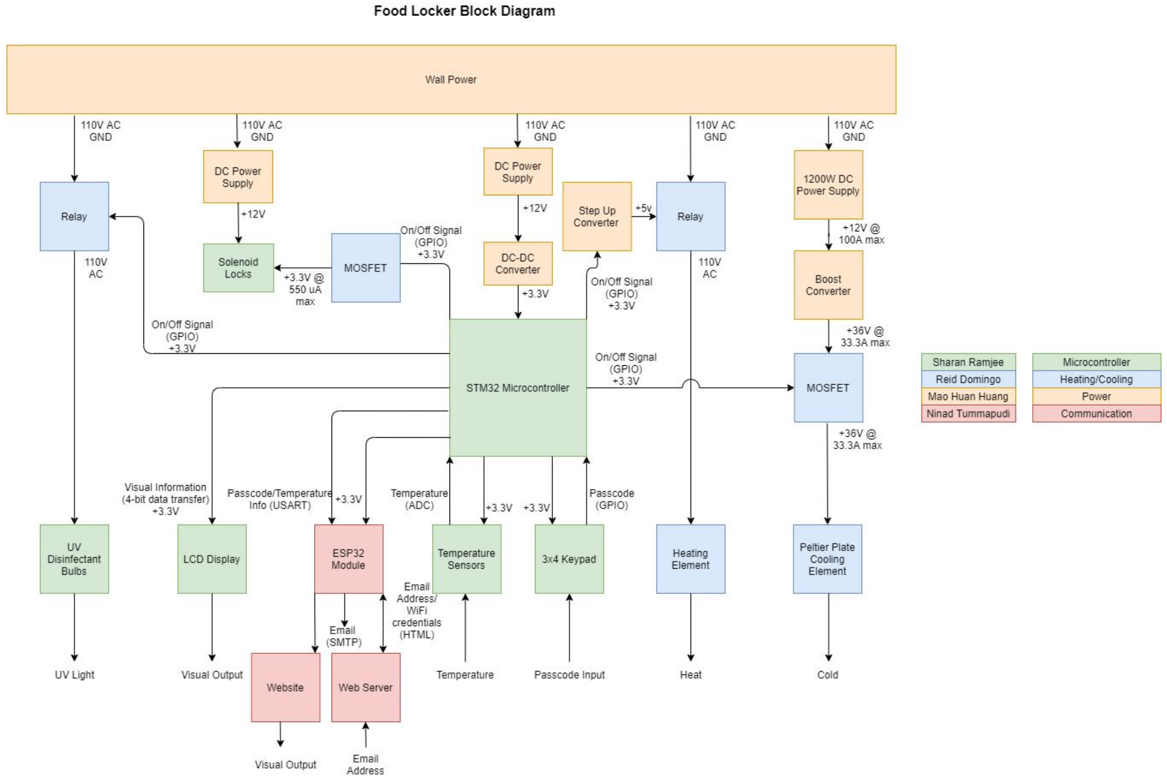
- Reid (delivery agent) delivers GrubHub meals around campus in order to fund his meteorological microcontroller projects. He is new to the job and has a hard time getting to certain destinations.
- Sharan (customer) is a hardworking 4.0 student taking 20 credits this semester and barely has time to eat while most establishments are open.
- Ninad (customer) ordered food online, but was in the shower when Reid arrived and couldn't open the door to receive his special delivery.
- Chris (customer) enjoys ordering food online, but is currently ill. For his safety and the safety of Reid, he needs to have his food delivered without any physical contact.

The food locker would solve all 4 of our problems.
Instead of finding his way around countless tight alleys and apartments, Reid can learn the locations of specific food lockers on campus to drop off the grub. As food can be left in these lockers for up to several hours, Sharan now has a wider selection of cuisines to choose from after getting out of his lab at 4 in the morning. Ninad no longer has to put off tasks and wait for his food, but has the freedom to do anything while Reid drops off his food. Chris can now safely access his food without having to make contact with Reid.

Team members and roles

- **Sharan Ramjee:** Responsible for the locking controller subsystem. Interfaced the STM32F0 microcontroller with the keypad, display, solenoid locks, temperature sensors, and UV bulbs. Implemented the control loop that controls the heating/cooling by turning on/off the elements based on the temperature readings inside the cabinets to maintain the selected temperature setting.
- **Ninad Tummapudi:** Responsible for the communication subsystem. Some important tasks include allowing the owner of the cabinet to connect the ESP32 device to the internet, allow the delivery agent to input the email ID of the customer, send an email with the passcode to the customer and host a website with the status of the food locker cabinet.
- **Reid Domingo:** In charge of the heating and cooling subsystem. Must use the signals from locking subsystem and power lines from power subsystem to use an electric heating device to heat the warming locker and peltier plates to cool the cold locker. Three separate temperature settings must be maintained depending on which is selected
- **Mao Huan Huang:** Taken on the power subsystem, in charge of boosting and attenuating a 12V input to 36V and 5V respectively. The process of which includes designing and simulating the buck and boost converters in LTspice, designing the boost converter PCB in EAGLE, and building the buck converter on breadboard.

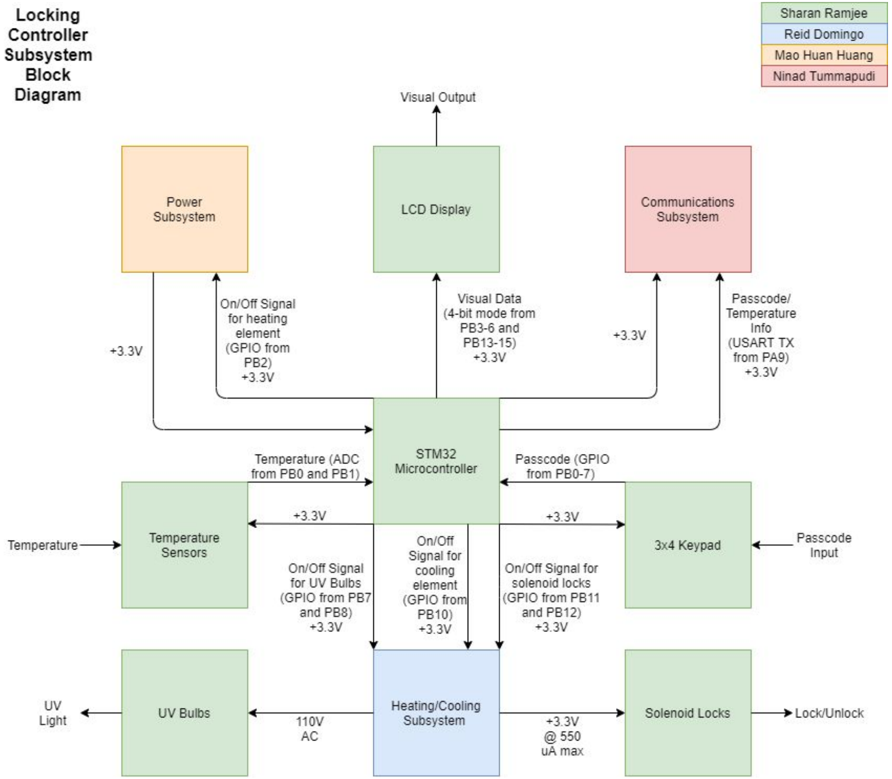
System Block Diagram



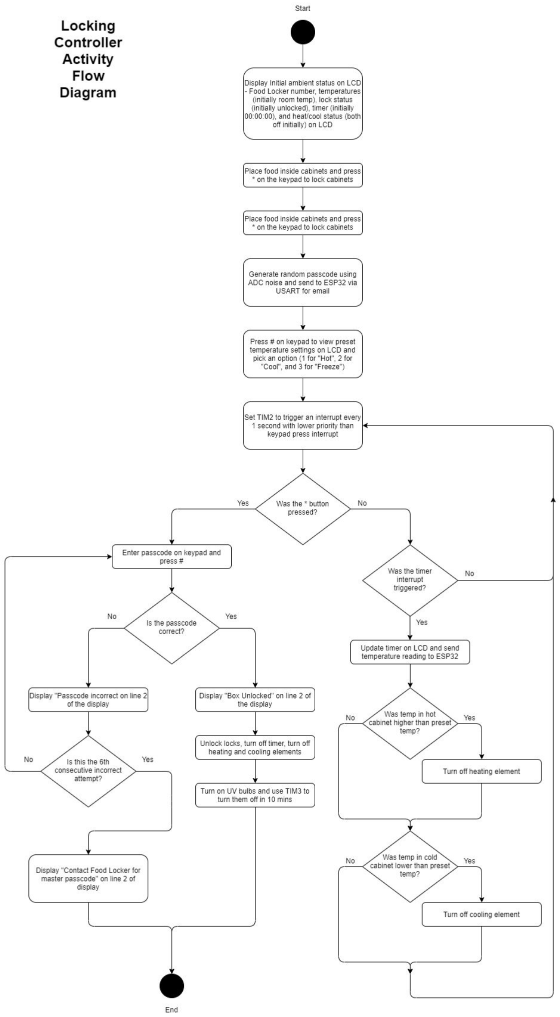
Requirements

1. The cabinet has 3 temperature settings, “hot” for foods that are meant to stay hot (65 ℃), “cool” for foods that are meant to stay cool (3 ℃), and “freeze” for foods that are meant to be frozen (-3 ℃). [Satisfied]
2. The cabinet automatically adjusts the internal temperature according to the preset settings. [Satisfied]
3. The boost converter should raise the voltage and lower the amperage of the 1200W power supply to levels where wires and leads will not heat up.
4. The user’s email will be entered on a webpage after the locker has been locked by the delivery person at the time of delivery to send an email. [Satisfied]
5. The cabinet is accessible using a randomly generated key sent to the customer via email. [Satisfied]
6. There should be temperature sensors within the cabinets that will output the temperature to be displayed via a display on the locker. [Satisfied]
7. The amount of time since the food was delivered will be displayed via an LCD display on the door of the cabinet. [Satisfied]
8. The solenoid locks will lock the cabinets once the correct passcode is entered. [Satisfied]
9. The UV bulbs will turn on for 10 minutes when the locker is unlocked after usage. [Satisfied]
10. The customer should be able to check the status of his order by accessing the Food Locker webpage which displays the temperature of the cabinet, the time the food has been in the locker for, the order number and the preset temperature option chosen by the delivery driver. [Satisfied]

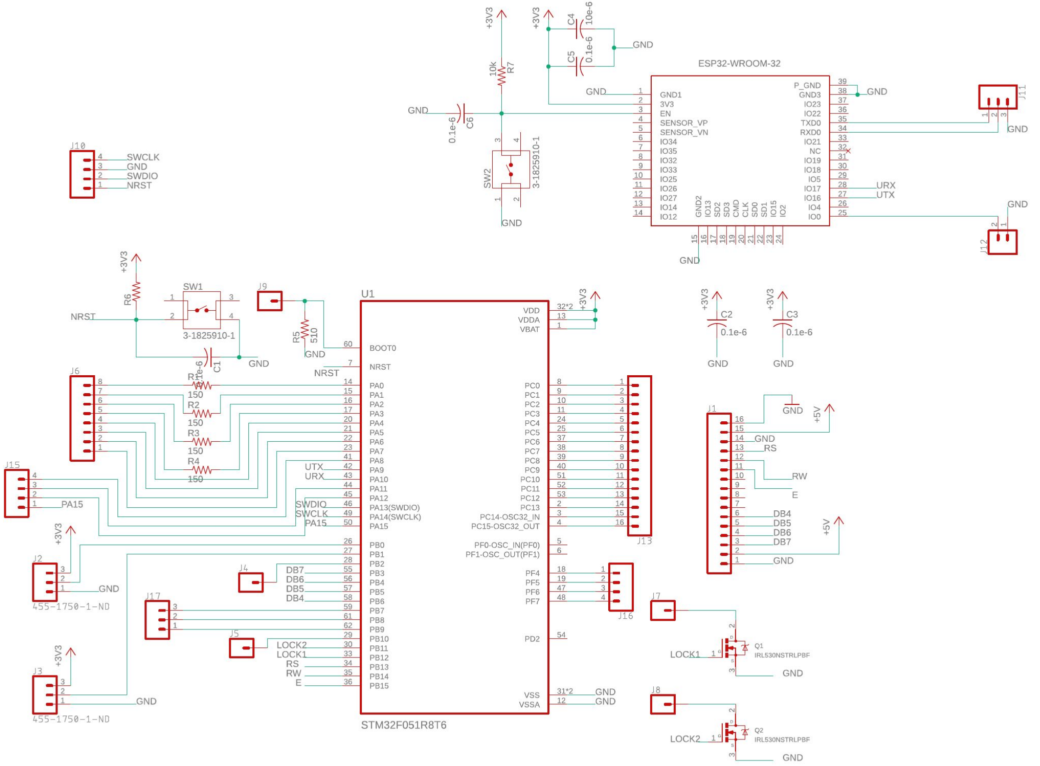
Locking Controller Subsystem - Sharan



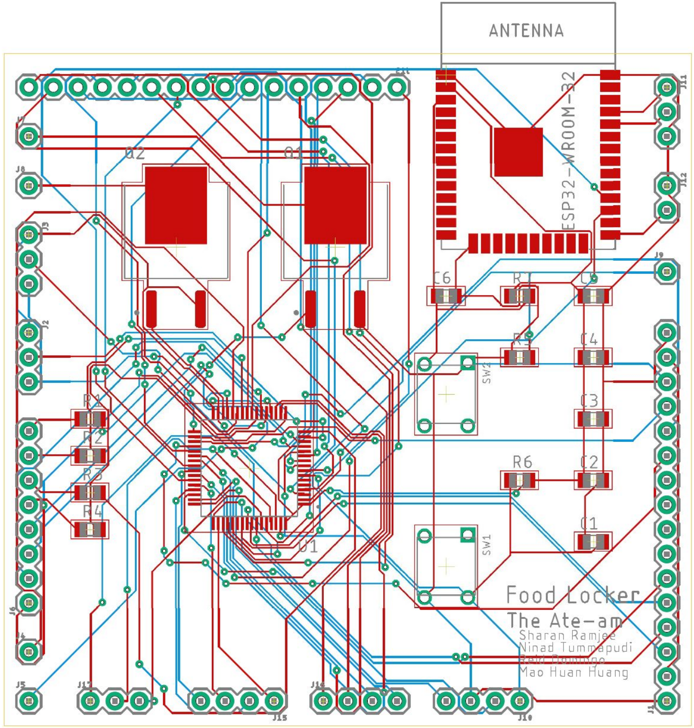
Locking Controller Activity Flow Diagram



PCB Schematic



PCB Layout



ANTENNA
ESP32-WROOM-32
U2
U1
C6
C5
C4
C3
C2
C1
R1
R2
R3
R4
R6
SW1
SW2
Food Locker
The Ate-am
Sharan Ramjee
Ninad Tummapudi
Mao Huan Huang

Locking Controller Theory of Operation

- The locking controller subsystem is responsible for all of the microcontroller interfacing and controls the interactions among the subsystems in the Food Locker.
- The 20x4 display shows ambient information about the Food Locker: Locker number, temperature, locker status, timer, and heating/cooling element statuses.
- The keypad is used to lock/unlock the cabinets. Once food is placed in the cabinets and * is pressed on the keypad, a random passcode is generated and sent to the communication subsystem via USART.
- In the locked state, the solenoid lock is locked, the timer ticks, and the heating/cooling are controlled via the control loop that turns on/off based on the temperature readings inside the cabinets to maintain the selected temperature setting (all of which are reflected appropriately on the ambient display).
- In the background, the microcontroller periodically transmits food locker status data to the communication subsystem every 1 second via USART.
- The customer is given 5 attempts to unlock the cabinets. Entering the correct passcode will unlock the cabinets, return the Food Locker to its initial state, and turn on the UV bulbs for 10 minutes to disinfect the cabinets.
- After all unlock attempts are used, the Food Locker can only be unlocked using the master passcode, which the customer must obtain by contacting the Food Locker service providers.

Locking Controller Specifications

1. The locked door must not open when pulled with a horizontal weight of 10 lb.
2. The passcode generated must be random as measured by the Pseudorandom Sequence standard (As learned in ECE 404) i.e. New passcode generated is –
   a. Independent of the previously generated passcode.
   b. Cannot be predicted from the previously generated passcode.
3. Display the temperature inside the cabinets, the timer (Time since food was delivered), whether the cabinet is locked or unlocked, and whether or not the heater and cooler are turned on or off on the LCD and make sure that the text is not cluttered to the point where a reasonable person cannot read the LCD.
4. The LCD screen receives enough power to be able to produce a brightness of at least 50 lumens to be able to be readable at night.
5. The UV Bulbs will show reduced bacteria growth on a petri dish after 2 days of incubation.

Locking Controller Justifications

- **STM32F051R8TC**:
  - Sufficient computation power to meet specifications.
- **HD44780 20x4 LCD display** (4-bit data transfer mode):
  - Adequate space for displaying all necessary ambient information.
  - Contrast control capabilities allow the display to be bright enough to be legible in the dark.
- **12 Button 3x4 Keypad** (GPIO):
  - Adequate for entering passcode information (numerical data).
  - Two keys (* and #) sufficient for switching between modes/backspace/enter.
- **Solenoid Locks** (GPIO):
  - Requires less current to drive compared to alternatives on the market.
  - Does not heat up and dissipate power when turned on for a long time compared to alternatives on the market.
  - Strong enough to meet the cabinet security specification.
- **UV Bulbs** (GPIO):
  - Does not contain ozone (unbearable smell) compared to alternatives on the market.
- **TMP36 Temperature Sensors** (ADC):
  - Can handle extremely hot/cold temperatures (well beyond range of specifications).
  - Extremely cheap and small compared to alternatives on the market.

Locking Controller Implementation Status

The Locking Controller subsystem has been completely implemented with the following proofs of integration with the other subsystems:

- Communication subsystem (Ninad)
  - Generated passcode and temperature data (periodically) sent via USART.
  - Temperature setting for the cabinets received via USART.
- Heating/Cooling subsystem (Reid):
  - LED turns on to show that the GPIO signal can be sent to turn on UV bulbs.
  - Heating/cooling display status changes to reflect on/off status of heating/cooling elements as controlled by the control loop.
- Power subsystem (Mao Huan):
  - +12V DC power received from the power supply.
  - +3.3V DC power received via USB cable from laptop.

Communication Subsystem - Ninad

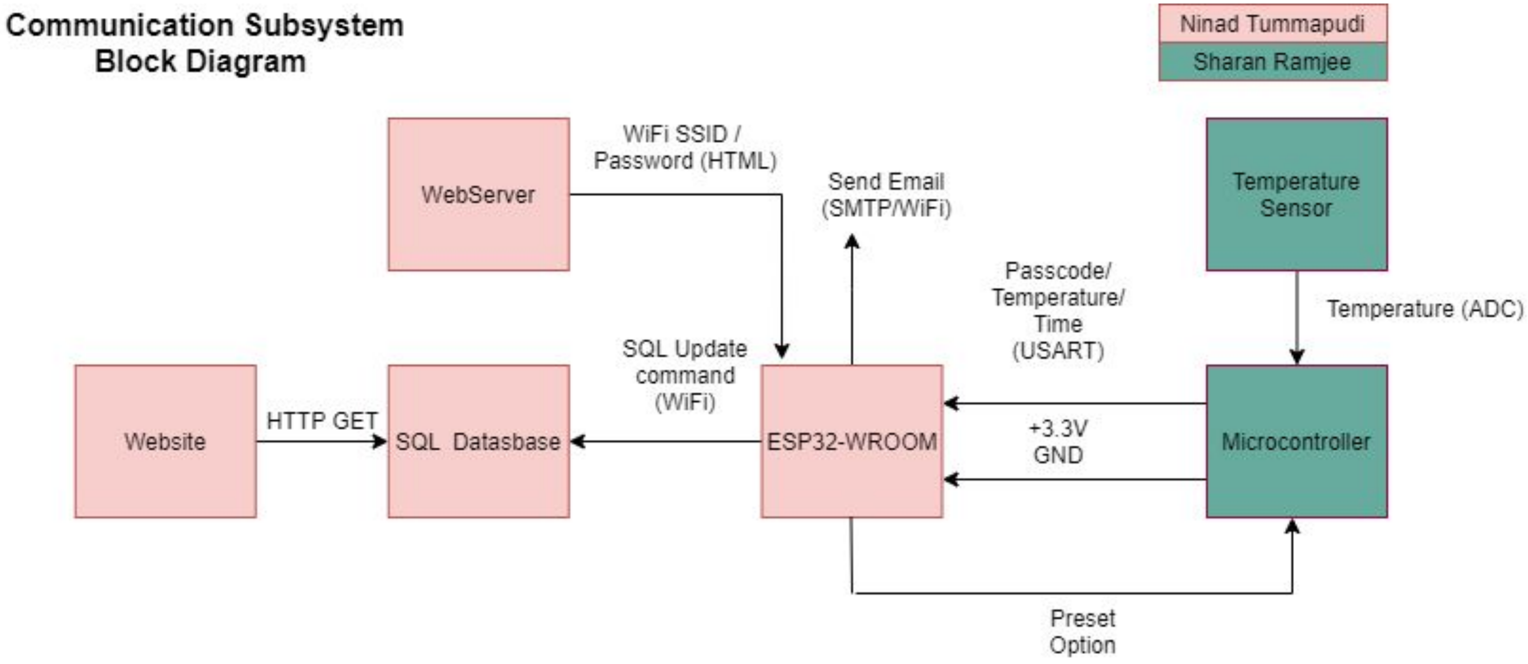


Figure 1

Software flow diagram

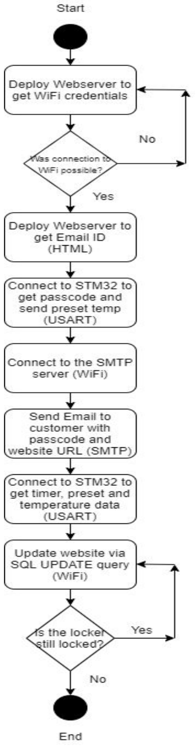


Figure 2

PCB Schematic

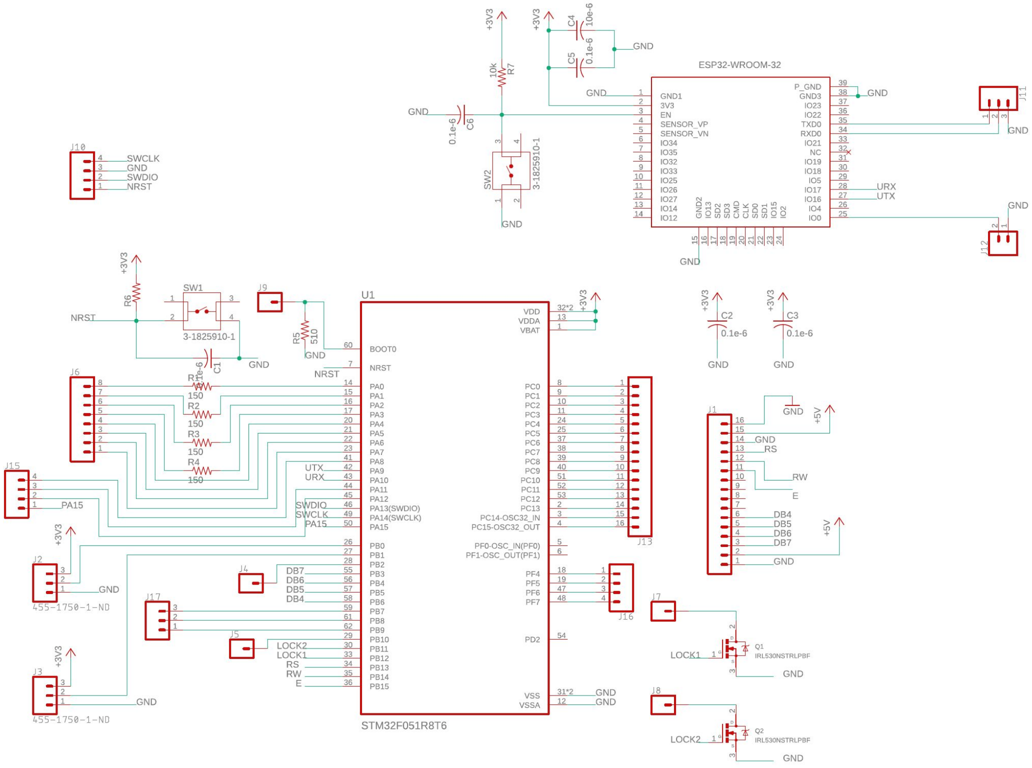


Figure 3

Subsystem Interactions

The communication subsystem interacts solely with the Locking subsystem in the following ways:

- Receives +3.3V from the STM32F0 microcontroller
- Sends a number value (0: Freeze, 1: Cold, 3: Hot) once to the STM32F0 microcontroller to indicate the preset temperature chosen by the delivery agent (USART)
- Receives the passcode generated by the STM32F0 microcontroller one time (USART)
- Receives the temperature of the cabinet from the STM32F0 microcontroller every 10 seconds (USART)
- Receives a flag once from the STM32F0 in order to start the timer on the website (USART)

Specification #1

The user should be able to connect to the ESP32’s **Access Point** if the user is within **20 feet** from the cabinet and access a web server by entering a url to enter the local WiFi credentials. The WiFi router can be upto a distance of **300 m** and with a frequency of **2.4 gHz**.

Justification:

- The ESP32 requires an internet connection in order to be able to connect to the SMTP server to send out emails to the customer
- An internet connection is also required to access the SQL database of the website in order to update the temperature value with the current value of the cabinet to allow the customer to see the status of their order

Specification #1 Status

The ESP32 requires a first time setup in order to be able to connect to the WiFi. This setup can be redone at any time. In order to setup WiFi, the owner of a specific food locker cabinet will require to input a username and password (Figure 5) in order to reach the setup page (Figure 6) in order to avoid tampering. The webpage will display the status of the WiFi connection.

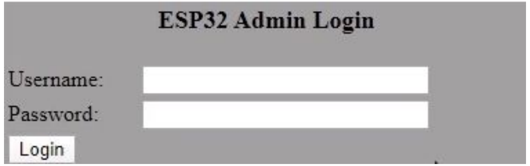


Figure 5

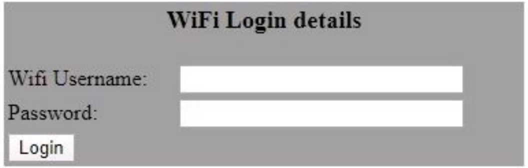


Figure 6

Specification #2

The ESP32 should be able to deploy an Email Input Form via **HTML** for the delivery agent to enter the user's email address, the order number and the preset temperature option (Freeze/Cold/Hot) if they are within **20 feet** of the cabinet.

Justification:

- Once the delivery agent places the food in a locker, they will require a way to alert the customer that the food has arrived and what passcode to use to open the locker. In order to decrease the workload for the delivery agent, an Email Input Form can be accessed and once the email and the preset temperature is filled in, the customer is sent an email with instruction on how to pick up their food from a Food Locker.

Specification #2 Status

The delivery agent can access the Email Input Form (Figure 7) by connecting his WiFi enabled device to the ESP32 and entering the static IP address of the ESP32. Once the email is sent to the customer, the delivery agent receives a response from the ESP32 (Figure 8).

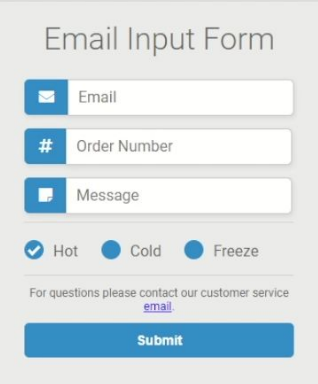


Figure 7

Figure 8

Specification #3

The ESP32 should be able to receive the passcode, temperature and timer information from the STM32F0 microcontroller and send the preset temperature option to the STM32F0 via the **USART** protocol.

Justification:

- This subsystem interaction is required in order to display the most current temperature of the cabinet and the time since the food was placed in the cabinet on the website in order for the customer to be able to view the status of his order.

Specification #3 Status

The ESP32 and the STM32F0 communicate using the USART protocol. This is achieved by connecting the RT pin and the TX pin of the ESP32 to the TX pin and the RT pin of the STM32F0 respectively (Figure 9).

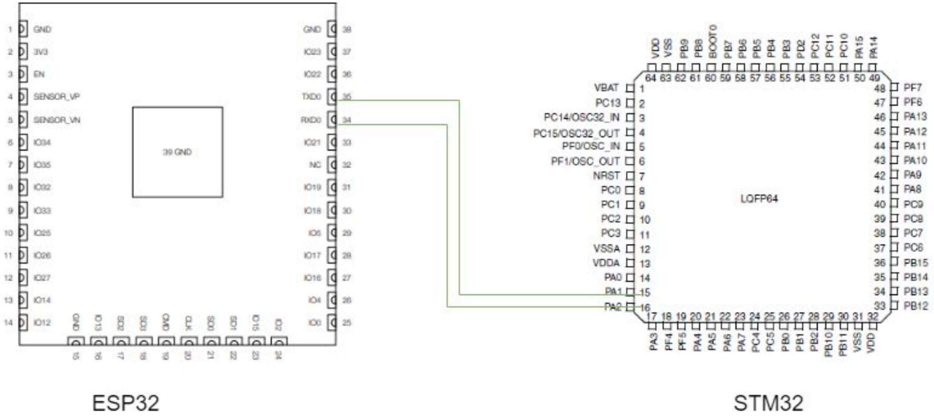


Figure 9

Specification #4

The ESP32 should be able to connect to the **SMTP** server and send email through that server from the Food Locker email ID to the customer email ID.

Justification:

- The customer will need to know how to access his food from the Food Locker cabinet and this information is provided to them via an email sent from the Food Locker email ID.
- The SMTP server allows emails to be sent to various email servers including those that require SSL/TLS authentication.

Specification #4 Status

Once the delivery agent enters the email ID of the customer on the Email Input Form, an email (Figure 10) is created with the one time use passcode, the order number and the message the delivery agent has written to the customer and a link to the website showing the status of their food in the Food Locker.

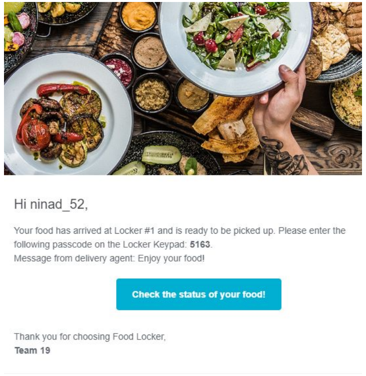


Figure 10

Specification #5

The customer should be able to access a webpage via url (hosted on **Google App Engine**) from any device that is connected to the **internet**. The website should display the time their order has been in the Food Locker cabinet for and the current temperature of the Food Locker cabinet.

Justification:

- This specification is more of a convenience feature for the customer to be able to view the status of his food via a url sent to them in an email.
- There is also an added benefit of tracking data over time for the Food Locker team (more on Specification #6).

Specification #5 Status

The customer will be sent a url to the webpage (Figure 11) where he will be able to see the status of the Food Locker cabinet in which his order is placed. No sensitive information such as the passcode or customer's email ID is displayed on the webpage for security purposes. The webpage of the url: https://food-locker-sd.uc.r.appspot.com/. The website is hosted on Google App Engine with an SQL database at the backend to update the values on the website.

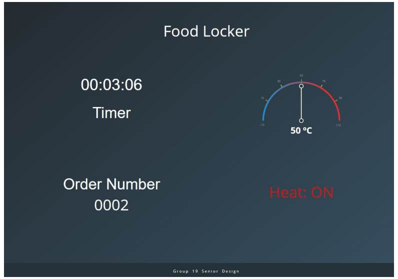


Figure 11

Specification #6

The ESP32 will instruct the STM32F0 via **USART** to turn off the heating/cooling elements for a period of time when the cabinet is at safe temperature (-3°C for Freeze, 3°C for Cold and 65°C for Heat).

Justification:

- By collecting data for the Food Locker cabinet over a period of one week (prior to the implementation of this feature), we will be able to determine for how long the cabinet can stay at the required temperatures without needing heating/cooling.

Specification #6 Status: Unable to test

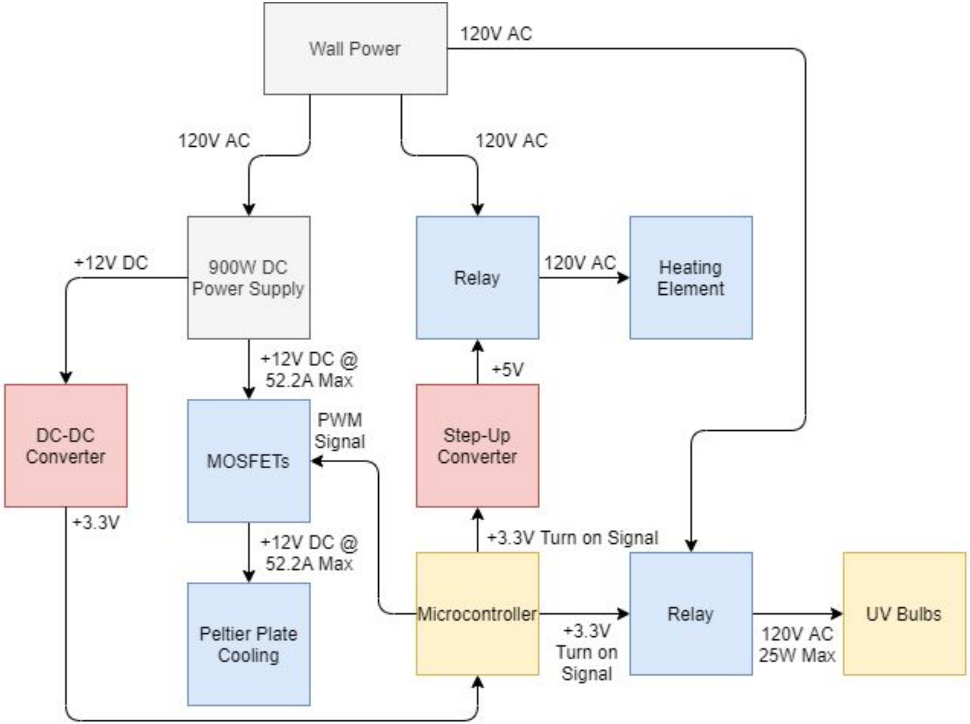
Due to not having access to the cabinet due to the quarantine, it is impossible to determine how long the cabinet can stay at a specific temperature. However, the ESP32 plots the temperature of the cabinet over time for each use of the cabinet. By analyzing these plots, in theory we would be able to determine how long the heating/cooling elements can be turned off for once a certain temperature is reached. The amount of time they can be turned off for will ultimately depend on how good the insulation of the cabinet is.

Example: If the heat mode is chosen and the heating element is able to heat the cabinet to 67°C, we can turn off the heating element for 10 seconds before the temperature falls below the 65°C threshold.

Theory of Operations (Summary)

- The communication subsystem’s first operation is to allow the user to connect the ESP32 to the WiFi and allow the WiFi information to be changed at any time through a webserver hosted by the ESP32.
- The subsystem will deploy a web server requesting the email ID of the customer and the preset temperature option that the delivery agent can access by connecting their device to the ESP32 which acts as an Access Point.
- The subsystem sends the preset temperature option to the locking subsystem via USART.
- The subsystem receives the passcode from the locking subsystem via USART, generates an email with the passcode and sends it to the customer’s email ID through the SMTP server over WiFi.
- The subsystem receives temperature of the cabinet updates from the locking subsystem via USART and updated the values on the website via SQL UPDATE queries over WiFi to the backend database.

Heating/Cooling - Reid

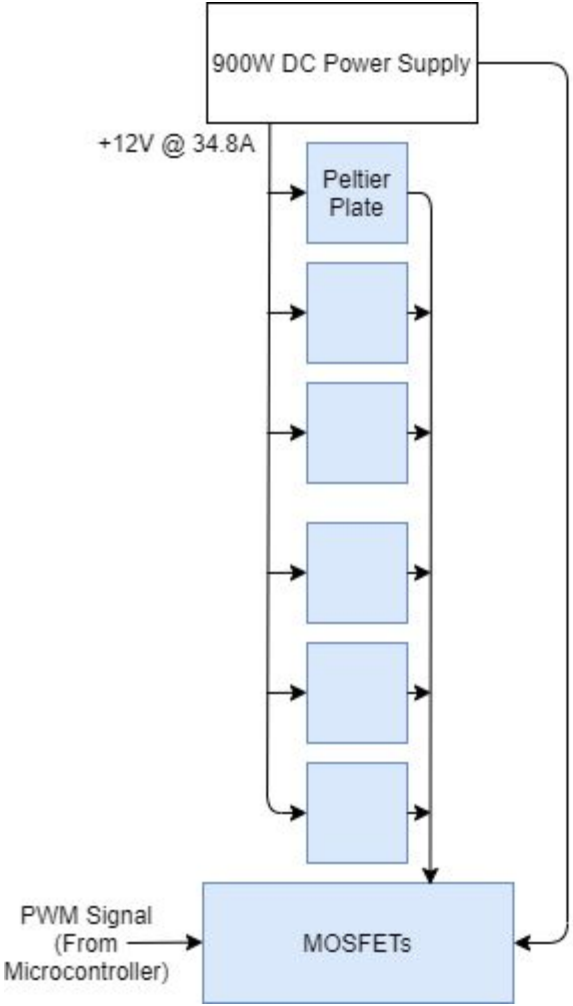


Subsystem Interactions

- Receive +3.3 (Stepped up to +5V from Power Subsystem) from control loop to turn on and off 120V relay controlling the heating element
- Receive PWM signal from control loop to turn on and off MOSFETs controlling the peltier plates
- Receive +36V @ 33.3A max from Power Subsystem for peltier plates
- Receive +5V from Power Subsystem to step up microcontroller signal to control logic of relay

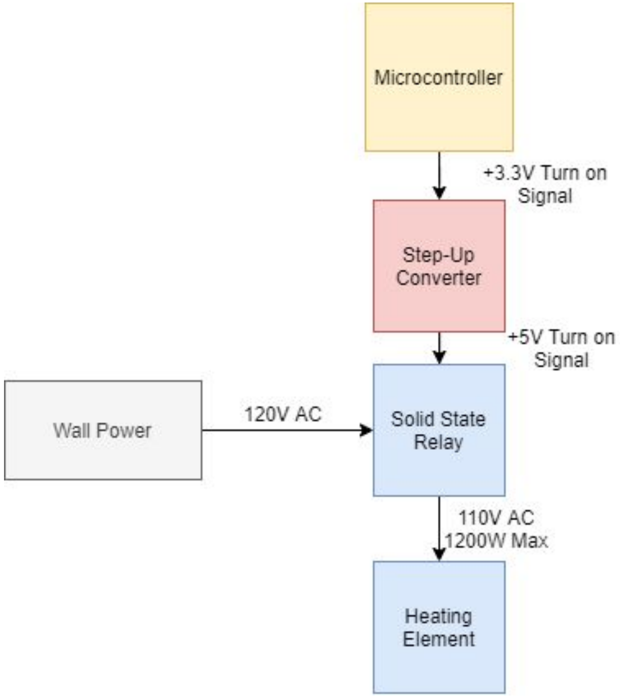
Peltier Plate Configuration

- Original design used 18 plates using 36V @ 34.8A total
- Revised design has each plate using 12V @ 5.8A each
- Using 6 plates total in parallel, this comes to
  12V @ 34.8A total



Heating Element

- Control loop used to regulate temperature inside
- Heating element is turned on and off using a solid state relay
- The relay control logic requires 5V, so the GPIO signal must be stepped up to 5V before being used



Specification #1

The heating compartment must warm and hold a temperature of 65°C. This temperature was chosen because the FDA recommends a minimum of 60°C when keeping food warm to prevent bacteria growth.

This specification was partially met. The locker is able to heat up to this temperature, but it is not able to be regulated because temperature sensor and control loop are not within this subsystem.

Specification #2

The cooling compartment must cool down and hold a temperature of -3°C. This preset was chosen to ensure frozen foods stay frozen.

A single peltier plate cools down to a temperature of roughly -5°C when powered with 12°C @ 5.8A. Unfortunately, because this was built on a breadboard instead of a PCB, using all 18 plates to draw over 100A was not an option. Using 6 plates, 3 on top and 3 on bottom of the locker, the inside was able to cool to roughly 5°C.

Specification #3

The cooling compartment must cool down and hold a temperature of 3°C. This preset was chosen to ensure things like cold drinks stay cold, but not frozen.

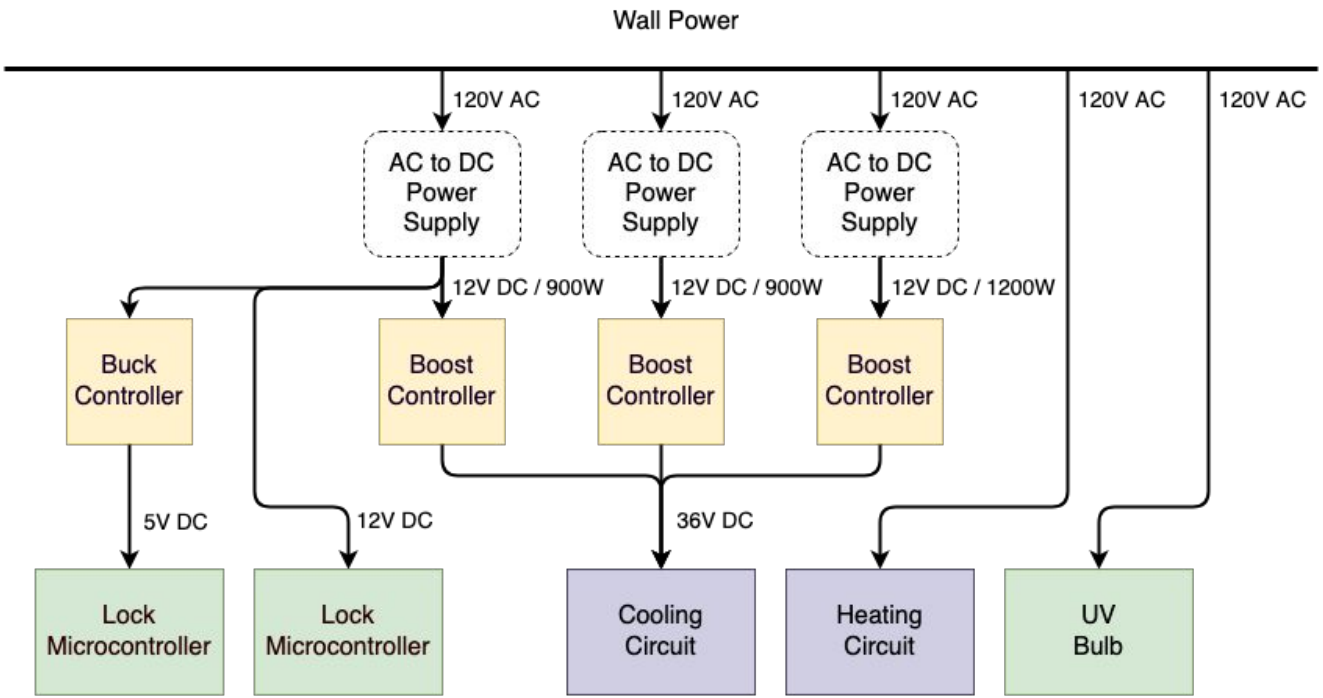
This specification was nearly met despite the usage of only ⅓ of the peltier plates. With the addition of the boost converter and PCB, this specification could have been met.

Theory of Operation (Summarized)

- This is the endpoint of all other subsystems. Controlling the heating and cooling elements to maintain the correct temperatures takes coordination and communication with all other subsystems
- The subsystem will use the power lines from the power subsystem and the control lines from the locking subsystem to meet the given temperature requirements
- The heating locker will use an 120V AC electric heating element that will be turned on and off with a 5V logic solid state relay
- The cooling locker will use peltier plates, heatsinks, and fans to maintain the cool temperature settings
- The peltier plates were originally to be controlled with a single MOSFET and boost converter, they are now controlled with 6 MOSFETs

Power Subsystem - Chris

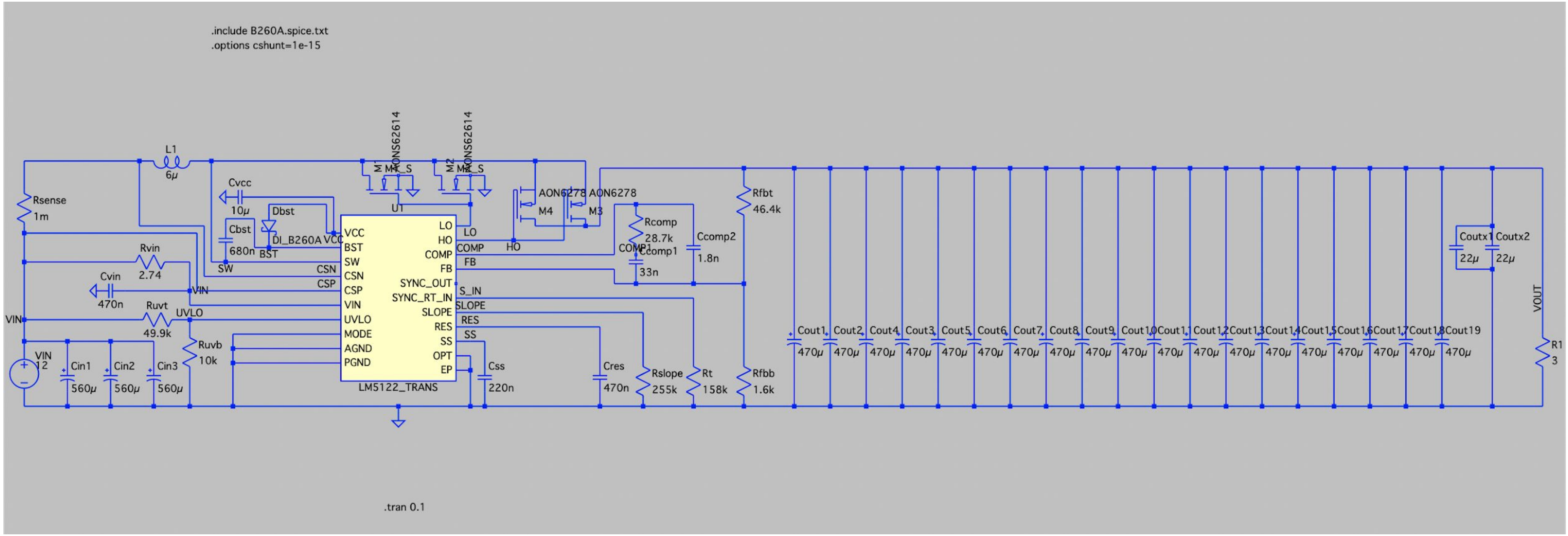
Mao Huan Huang
Sharan Ramjee
Reid Domingo



Subsystem Interactions & Specifications

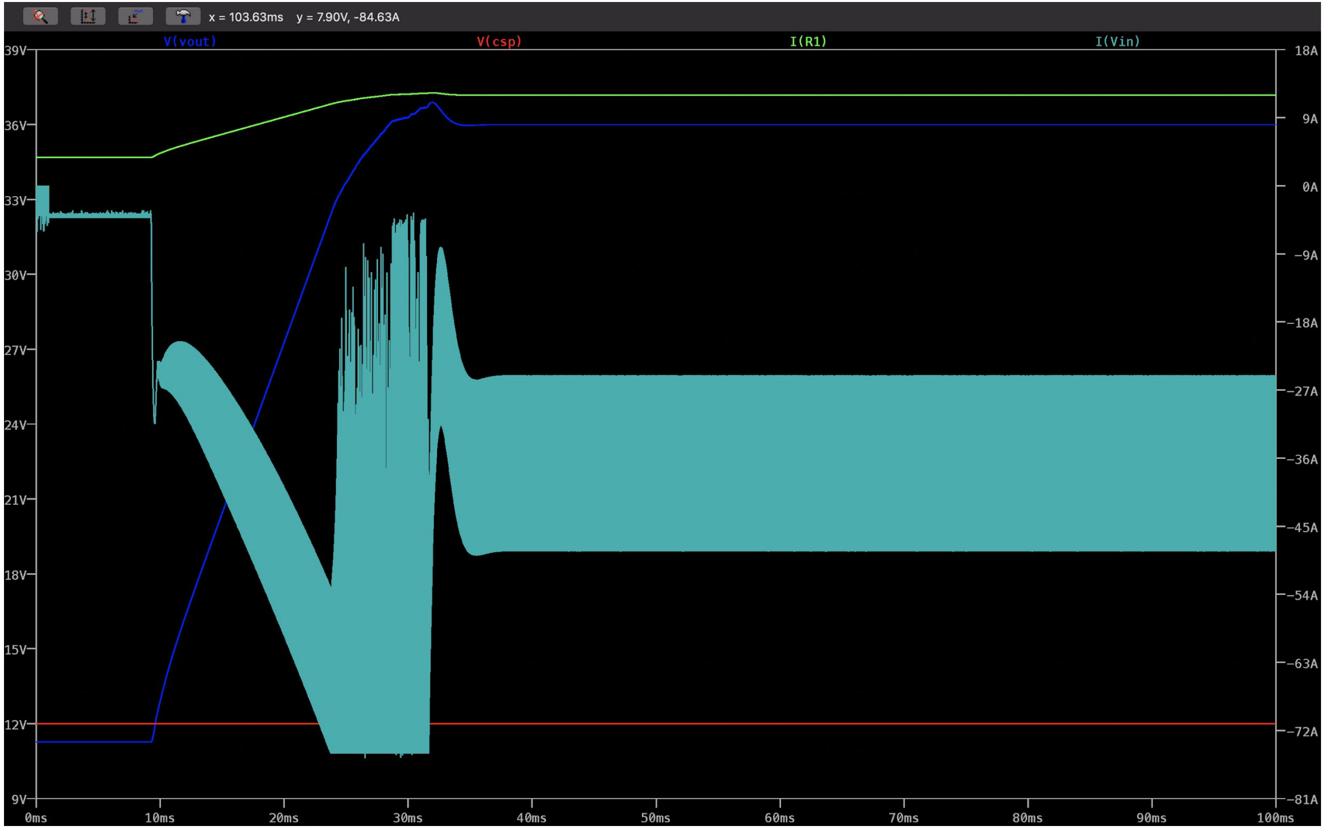
- Supply 36V DC via boost converter to the cooling circuit, 3 boost converters are connected in parallel to drive enough current to sufficiently cool peltiers.
- Supply 5V DC via buck converter, linearly regulated to 3.3V DC to locking microcontroller.
- Wall power supplies 120V AC to heating circuit and UV bulb
- 12V DC from DC power supply supplies solenoid lock

Boost Converter

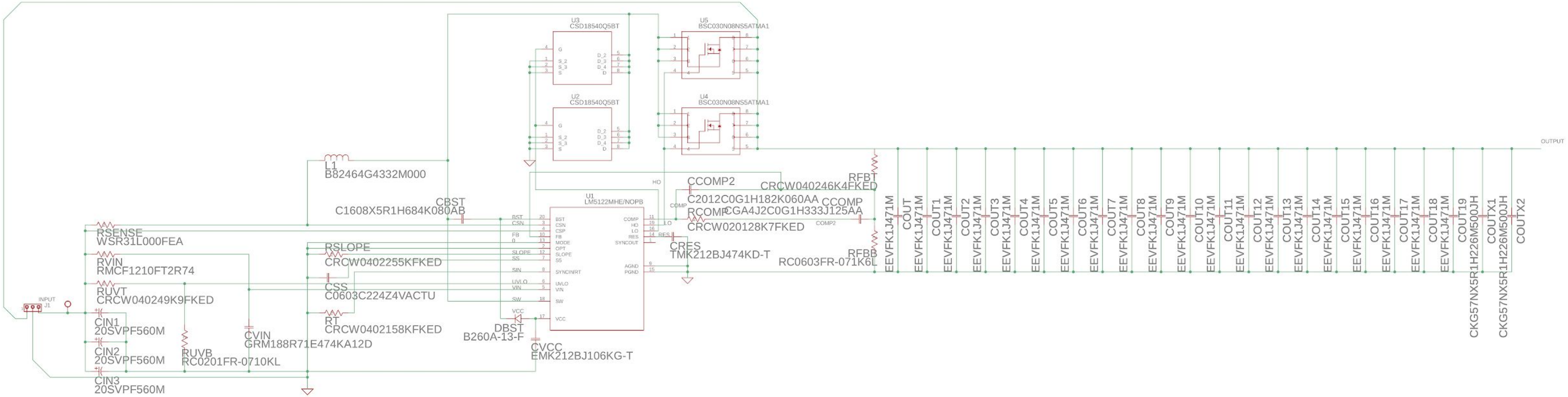


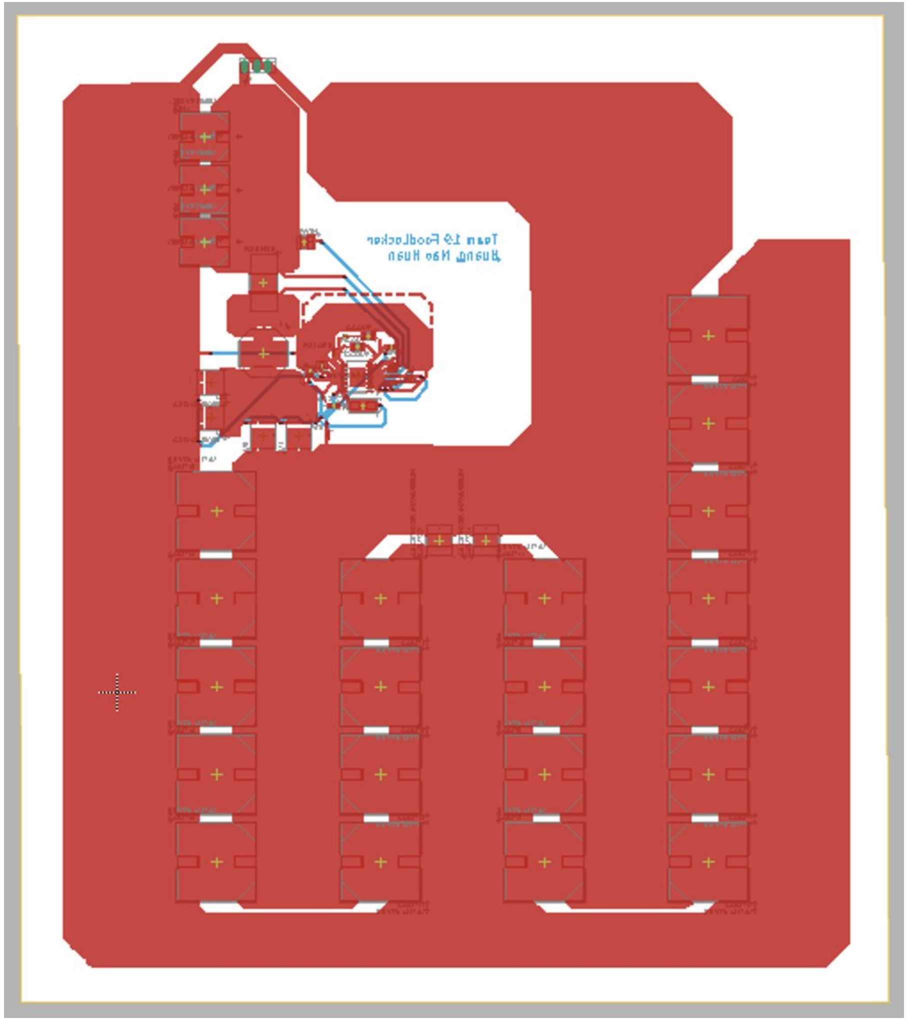
Results

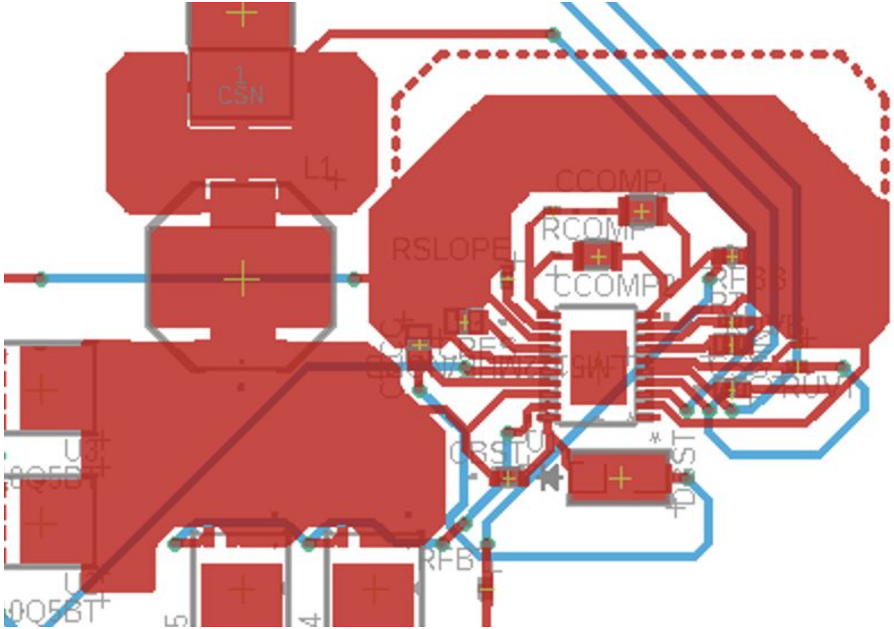
Boost converter simulation produces 36V output at 12A



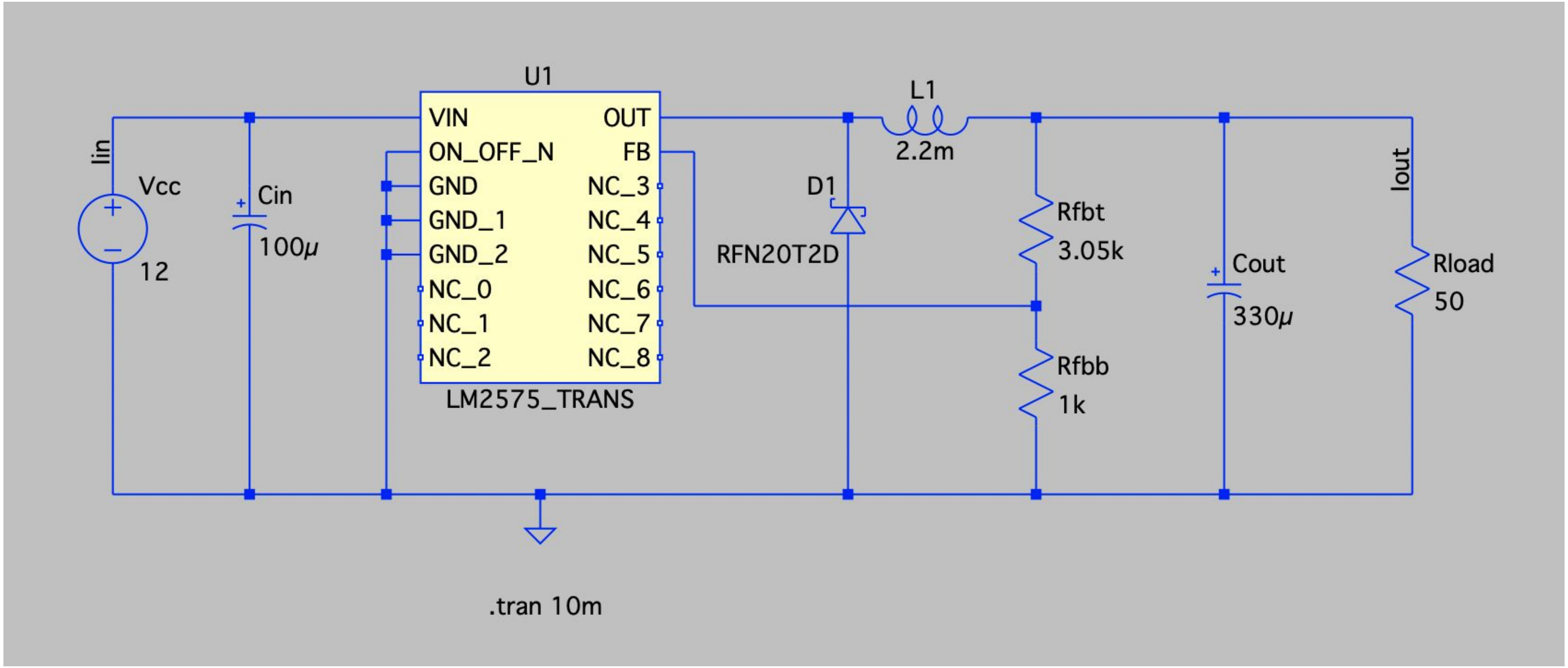
PCB Schematic & Board Design





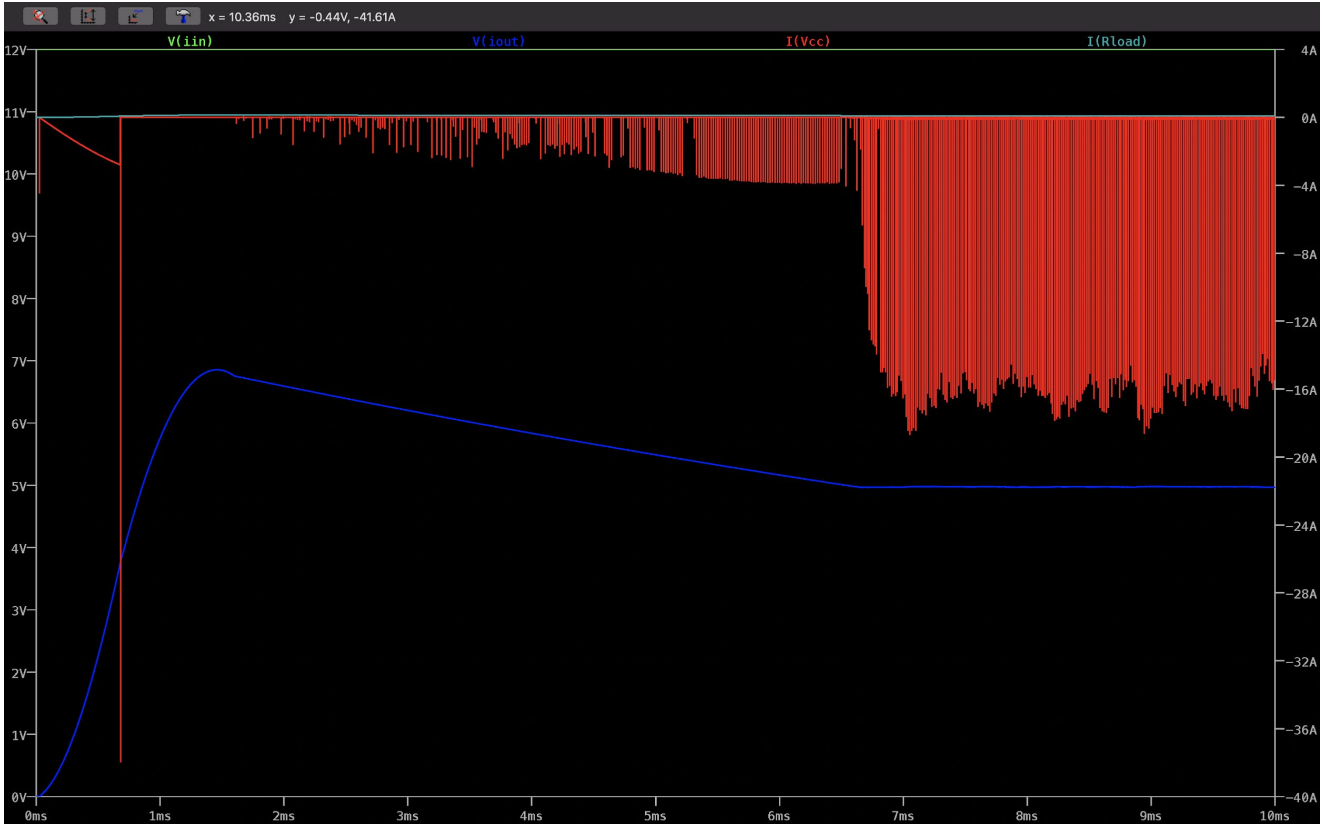

CSN
L1
CCOMP
RCOMP
RSLOPE
CRST
RFB

Buck Converter



Results

Buck converter simulation produces 5V output at 100mA



x = 5.60ms y = 79.65mA
I(Rload)
140mA
130mA
120mA
110mA
100mA
90mA
80mA
70mA
60mA
50mA
40mA
30mA
20mA
10mA
0mA
0ms
1ms
2ms
3ms
4ms
5ms
6ms
7ms
8ms
9ms
10ms

Theory of Operation

- The converters use boost and buck topologies to achieve the desired voltage outputs. We avoid linear regulators for high current applications due to its inefficiency. We could strive for better efficiencies with better, custom wound inductors, however for this proof of concept we have opted for mass produced options for ease of purchase and design.
- The buck outputs 5V for it will be easier to attenuate 5V to 3.3V than amplifying 3.3V to 5V.

Recommendations

- Design improvements:
  - Develop an app (Android/iOS) for better integration with the Food Locker.
  - Allow users to unlock the locker using the app/webpage when they are in proximity of the Food Locker (by adding Bluetooth or location functionality).
- Unmet requirements:
  - Accurate calculations for how long the heating or cooling elements can be turned off for before the temperature of the cabinet reaches an undesirable level. In order for real testing to be done, the cabinet, the temperature sensors, the ESP32 and the STM32 need to be completely integrated.
  - Boost converter requirement not met.