MULTIPURPOSE TEMPERATURE CONTROLLED CHAMBER (FOR CONSUMER APPLICATIONS)

By

Isaac Brorson

Mitchell Stermer

Stefan Sokolowski

Final Report for ECE 445, Senior Design, Spring 2024

TA: Selva Subramanium

May 1 2024

Project No. 42

Abstract

This study explores the design and development of a novel multifunctional device capable of acting as both a heating element and a cooling chamber. The research focuses on integrating heating and cooling functionalities within a compact and efficient appliance suitable for consumer use. Through experimentation and optimization of thermal management systems, the aim is to create a versatile device that maximizes energy efficiency, minimizes space requirements, and enhances convenience for users. The findings of this study offer insights into the feasibility and potential benefits of combining oven and refrigerator functionalities in a single appliance, addressing diverse cooking and food storage needs in a compact and innovative solution.

Contents

1. Introduction

Oftentimes, people store food in various locations such as the kitchen, refrigerator, or freezer with a vague idea of how the temperature of that item will change over time. In some cases, they'll come back hours later only to find that their food is not properly thawed or frozen. A few specific examples of this could include: putting a beverage in the freezer to quickly cooler it, only to forget about it and later find it exploded and frozen; planning to cook a steak, but forgetting to move it from the freezer to the refrigerator the previous day; or setting food out overnight in order to prepare it for the next day only to find that it didn't thaw as expected.

Our solution is a programmable temperature-controlled chamber which allows a user to set their own temperature curve of a food item they plan on consuming soon. By setting a temperature curve, the user will be able to precisely control what temperature changes the food undergoes within a certain interval of time. The design will consist of a chamber, where we power a band heater to increase temperature, and thermoelectric coolers with heatsinks to dissipate energy inside the chamber. In sum, this device offers precise control over food temperature. Someone would use this device by placing their food item in the device's insulative chamber and closing the lid. Next, the user interface would present a variety of options: standard heating or cooling presets for common food items, or the ability to set a detailed temperature curve. The user may then leave while the chamber performs its function and return at the desired time at which they set it to be ready. Temperature controlled chambers on the market are exorbitantly expensive and large for a household kitchen, and this design is better suited for consumer applications.

2. Design

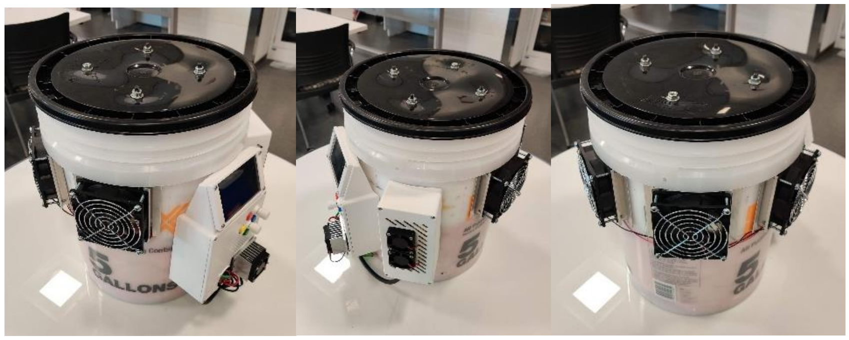


Figure 1: Pictures of Finished Product

2.1 Block Diagram and High-Level Requirements

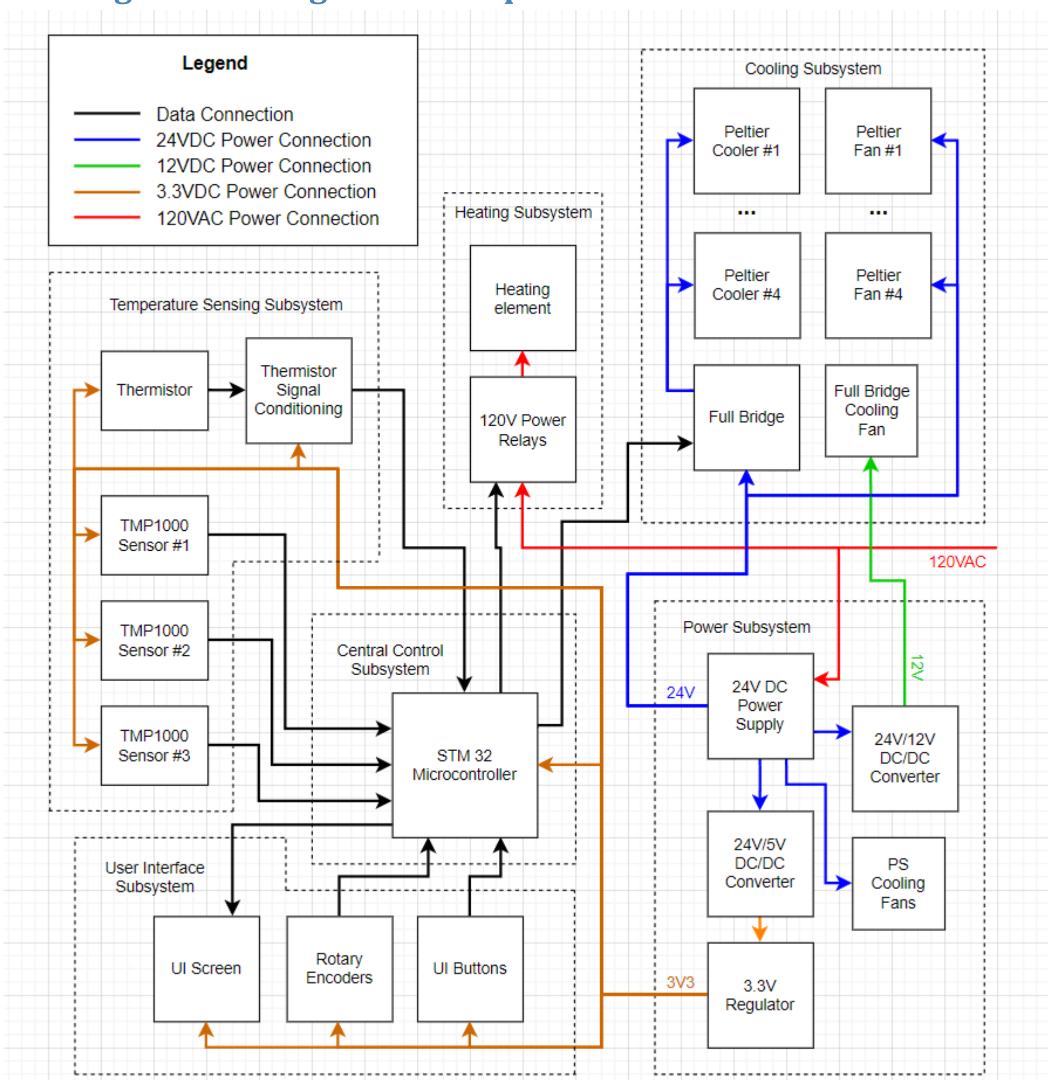


Figure 2: Block Diagram

The power subsystem receives 120 V alternating current (AC) from a nearby outlet and outputs 24 V direct current (DC) to both the heating and cooling subsystem. The power subsystem also utilizes a 24-5 V DC buck converter, a 24-12 V buck converter, and a 3.3 V linear regulator to output to the central control subsystem as the STM32-401RE microcontroller uses 3.3 V logic.

The core of the control system comprises the microcontroller, which serves as its central component. In addition to interfacing with the power subsystem, it receives inputs from temperature sensors and user interfaces. User interactions dictate the information displayed on the Liquid Crystal Display (LCD), while temperature data governs the activation of either the heating or cooling subsystems.

The user interface consists of external buttons, rotary encoders, and the LCD connected through the 3.3 V power. Buttons and encoders are used to traverse the menu system, shown on the display, and set the parameters for the heating curve. The LCD can also graphically display the heating curve and real-time temperature measurements from the temperature sensing subsystem.

The temperature sensing subsystem consists of three external Inter-Integrated Circuit (I2C) temperature sensor breakout boards used to read the temperature at various locations just outside the outer chamber. In addition, we employ a Negative Temperature Coefficient (NTC) thermistor integrated within a probe to accurately measure the temperature of food items within the inner chamber. This data is read by the control subsystem where it converts these voltages to temperatures.

Our heating system consists of an AC heating band, whose power is switched using an on-board relay which is then controlled by the STM's GPIO pins.

The cooling system employs two bootstrapped half bridge converters which allow reverse polarity switching to the output and running the Peltier modules in both cooling and heating modes with the same drivers. Two bootstrapped gate drivers are driven by the STM32's timer pins which output a PWM signal of specified duty cycle.

2.2 Design Changes

Electrical

Electrically, the biggest change was seen in our cooling system half bridges. Because of issues in driving high power loads with our initial design, we had to add external inductors and capacitors to smooth our output voltage while only utilizing one half bridge instead of two, thus removing our ability to reverse the polarity of the voltage output. For more information, please read the Cooling Subsystem Verification Section for more information on our struggles and final solutions.

Mechanical

Several aspects of this design changed throughout the process of building the device. We originally imagined that we'd have two nesting cylindrical containers with fiberglass insulation evenly distributed throughout the gap separating them. The inner chamber would be made of aluminum for its thermal conductivity, and we would use metal standoffs to fix the inner chamber in place relative to the outer chamber. We would design an insulative lid that would be attached using a hinge.

Several of these plans saw changes. First, we couldn't afford an aluminum inner chamber and went with a stainless-steel Bain-Marie. This wasn't our preferred solution because stainless steel has a much lower thermal conductivity than aluminum, but it was our best option given our budget. Secondly, with the realization that we would need something to keep the fiberglass insulation away from the user, we decided to go with an acrylic ring instead of metal standoffs to hold the inner chamber in place. The inner chamber would drop into the opening of this ring, and the ring would be a close fit to the inside of the outer chamber. Thirdly, we realized that due to the complex geometry of the Peltier cooler adapters and heat sinks surrounding the inner chamber, we wouldn't be able to get fiberglass insulation to fill out the entire gap between the inner and outer chambers. Instead, we would need to cut Styrofoam to precisely fill the gap around the upper part of the chamber. Fiberglass would still be necessary around the lower part of the chamber because the heating band (which is wrapped around the lower part of the chamber) would get hot enough to melt Styrofoam. Finally, we ditched the idea of adding a hinge to the lid because it would create unnecessary mechanical complexity. The 5-gallon bucket that we chose for the outer chamber came with a press-fit lid, and we realized it would be much simpler and just as effective to use this lid. Initially we didn't think we'd need any waterproofing around the screws attaching the Peltier cooler adapters to the inner chamber, but after seeing that they didn't sit flush to the surface, (the machinists didn't want to drill mounting holes at an angle) we decided to add rubber washers to keep the inner chamber waterproof. We didn't end up drilling a hole in the side of the inner chamber for the thermistor temperature probe as planned because it was much simpler to just route the wire under the lid. Lastly, we had to modify our electrical boxes (Eboxes) to accommodate active cooling. We originally thought that some airflow slots would be enough to cool the power supply, but after testing the power supply under load, we found that active cooling was needed, so we added two cooling fans to the 3D design of the power supply Ebox cover. As for the main board Ebox, we printed its cover with space for a cooling fan, but then it turned out we would need a heat sink extending out from the main board to cool the H bridge MOSFETs. Unfortunately, we needed to cut a hole in the main board Ebox cover for this heat sink because there wasn't enough time to redesign and reprint it. (See Figure B.2.1 in Appendix B.2 for a look at the cutout we made)

One design change that occured relatively late in the design process was switching from thermal grease to thermal pads on the cold side of the Peltier coolers. Initially, we planned to use thermal grease on both sides of the Peltier coolers. However, a significant number of Peltier coolers were breaking during the installation process, and we weren't sure why. Eventually we realized that they were fracturing because of mechanical strain put on them by curvature in the surface of the aluminum Peltier adapters the machine shop made for us. Their surface was raw aluminum extrude, which apparently wasn't flat enough for the ceramic of the Peltier coolers. Thus, when we clamped them down with the heatsinks, the corners were under too much stress and would fracture. We found that we could solve this problem using thermal pads on that side of the Peltier coolers. The thermal pads have neough thickness to account for the curvature in the aluminum adapters, and successfully prevented the Peltier coolers from breaking. (See Figure B.4.1 in Appendix B.4 for a cross section of the Peltier cooler and adapter design.)

Rather than employing multiple thermistors as originally planned, a singular thermistor was situated within a probe to monitor food temperature, while the chamber temperature was assessed through the averaging of data collected from three integrated temperature sensor PCBs (Printed Circuit Boards). Additionally, we originally planned on implementing an LED (light emitting diode) as a helpful indicator of the device status while in operation but ended up focusing our attention on the core functionality of the project.

Software

The key principles of the software design remained steady throughout the project. The original idea was to construct a graph of the desired heating curve that the user sets through the interface. After developing the interface and verifying the temperature sensing subsystem, and the internal clock (which is discussed in Section 3.6.2), we realized it would be easy and convenient to include two other graphs, for the actual food and chamber temperatures since we were already reading them and printing the values on the display. Unlike the desired heating curve, which is plotted all at once, these additional graphs would be incrementally plotted as time elapsed.

The only real technical change pertained to setting the duty cycle of the Peltier system when cooling. At first, we were going to use a Proportional-Integral-Derivative (PID) system to precisely determine the optimal duty cycle of the voltage fed into this module. However, after many cooling tests were run, we determined that cooling took a considerably long amount of time. Thus, we deviated from the original plan and chose the most efficient duty cycle for cooling, which we found was about 60%. More detail on this figure can be found in Section 2.3.4.

2.3 Submodule Design

Throughout this section, various subsystems will be described, for questions regarding schematics and PCB layout sections, please refer to Appendix C to get all the pertinent information.

2.3.1 Power Subsystem Design

Mechanical

The 24VDC power supply we chose for our project lacked any cover or enclosure, and had exposed high voltage AC, necessitating an Ebox for it. We took measurements of the power supply and began designing an enclosure for it using Fusion 360. We made the base of the Ebox fit the curvature of the outer chamber so there wouldn't be any gap. We added a cutout for an IEC plug in the bottom of the Ebox, and a cutout for the DC output wires in the upper part of the Ebox facing the chamber. We added slots for airflow and cutouts for two cooling fans in the Ebox cover. Finally, we had to add a circular cutout for the 120VAC power cord to extend from the IEC plug inside the Ebox out to the main board Ebox. This hole was split into two parts, and would clamp down on the cable when the Ebox cover was screwed in. Once we were finished with the design, we printed it using a 3D printer and mounted it to the outer chamber. (Appendix B.1, Figure B.1.1)

Electrical

The power subsystem consisted of an off the shelf 24V AC-DC power supply (EMH350PS24) capable of supplying up to 300W. This is mounted externally and the 24V is then supplied to our PCB, where the rest of the power subsystem steps down the voltage to various levels that are needed to properly operate the rest of the systems. The power subsystem on the PCB then consist of two buck converters, one which steps the 24V down to 12V to provide the proper voltage for our gate drivers while the other buck steps down to 5V to provide the proper voltage for the relay driving circuit and our 3.3V LDO, which takes 5V and linearly steps it down to 3.3V for all of our logic level systems on the board. This architecture was selected to limit switching noise on our 3.3V rail while providing the proper voltages for all our systems.

2.3.2 Central Control Subsystem Design

Mechanical

The main board Ebox was more complicated than the power supply Ebox because there's a lot more to consider for the main board. Two 120VAC power cords need to connect to terminal blocks on the main board. Since these are large cables and are also carrying mains voltage, we needed to create a system to hold them in place. Otherwise, if they were free to move around, a connection could pull out of a terminal block and electrify some part of the device that the user could touch. We designed a channel for the 120VAC cables to run though, holding them in place in a position that makes it simple to make connections to the terminal blocks. One cable (the cable supplying power) would enter the main board ebox from the outside, and the other cable (the power cord for the heating band) would pass through the wall of the chamber to the heating band. We created a slot facing the inner chamber for various wires including the heating band power cord to get routed through the wall of the chamber. We added an angled mounting surface for the user interface display screen, as well as cover to attach and protect the LCD. We added holes in the front of the Ebox cover for the user interface buttons and rotary encoders. We added airflow slots and space for a cooling fan into our 3D design, but after printing it, we realized we'd need to make modifications to this part of the cover. Instead of reprinting the Ebox cover, we cut a hole in it to fit the heatsink and fan that were needed for the power MOSFETs. (Appendix B.2, Figure B.2.1)

Electrical

Electrically, the central control system is simple, comprising an SMT32F401RE and the required reset architecture and decoupling capacitors to ensure proper operation. The Central control system also contains a debug LED, proper ST-Link programming headers, and an optional ESP32 coprocessor which can communicate to the STM over UART and serve to offload come compute or give the capability to operate over the internet at some point in the future. We speak about these extra capabilities in the Future Work section (5.4).

Software

We chose to use an STM32401RE microcontroller as it has just enough pins to fit all our peripheral requirements. It is programmed in C as opposed to C++ using STMCubeIDE, as our design does not require object-oriented programming. The following pins were configured: SPI protocol for the LCD; I2C protocol for the temperature breakout boards; an Analog to Digital Converter (ADC) pin to read the voltage across a thermistor, and (GPIO) connections to the heater and Peltier system. Timer pins were configured to the Peltier pins to control the duty cycle of the Pulse Width Modulation (PWM).

2.3.3 Heating Subsystem Design

Mechanical

The heating subsystem was composed of a band heater which we were able to get for free from the machine shop. The band heater almost perfectly fit around the circumference of the inner chamber, but we needed to cut off some excess material from it to allow it to fit snugly in our design. We used special high-temperature polyimide tape to secure the band heater to the outside of the inner chamber. We needed to use a saw to cut off a temperature control knob from the band heater to allow it to fit into the outer chamber. We positioned the band heater towards the lower part of the chamber so that the natural convection current formed by the band heater would serve to even out temperature differences across the chamber. (Appendix B.3, Figure B.3.1)

Electrical

The heating subsystem is electrically very simple. We did not need to implement any fast-switching architecture because our heater band was AC. So simply all we had to do was implement two relays to completely isolate the AC power from the band heater and have them controlled by two GPIO on the STM32.

Software

Discussion of the Proportional-Integral-Derivative (PID) system is in Section 2.3.7. Since the band heater runs off a relay, we can effectively construct a PWM signal low duty cycle. In the final design, when heating, we used a 20% duty cycle with a period of ten seconds to power the heater. We found that at higher duty cycles, the chamber would overshoot the target temperature and would need to cool to compensate.

2.3.4 Cooling Subsystem Design

Mechanical

The cooling subsystem is based around Peltier coolers, which are semiconductor devices that create a temperature difference across themselves when a voltage is applied. Since Peltier coolers are flat with ceramic faces and our inner chamber is cylindrical, we needed thermally conductive adapters to connect the Peltier coolers to the chamber. We had the machine shop machine four of these for us out of aluminum. They added six mounting holes in each, four to attach the adapter to the chamber, and two to attach the Peltier cooler and heatsink to the adapter. See Figure B.4.1 in Appendix B.4 for a cross section of the Peltier cooler and adapter design. The heatsinks clamp down on the Peltier coolers, ensuring good thermal contact. Cooling fans are screwed onto the heat sinks.

Electrical

The cooling system was initially designed to have two bootstrapped half bridges to run the Peltier modules in both forward and reverse, we initially used IRF3710 MOSFETs and the ADP3110AKRZ-RL gate driver but later in the design, we found that these MOSFETs did not have the proper current requirements for our design, so we had to switch to a higher power MOSFET from the ECE supply center. We also found that bootstrapped converters do not play well with output capacitance, so we had to place inductors on the output and in order to have a large enough capacitance, we had to use electrolytic polarized capacitors on the output, meaning the use of both half bridges two run the Peltier modules in forward and reverse could no longer be implemented. So we had to switch to a single half bridge running the Peltier modules in one direction and an external LC network to smooth the output.

Software

As stated, information on the PID control is in Section 2.3.7. Upon determining that cooling is needed, we set the Capture/Compare Register (CCR) of the timer connected to the Peltier PWM output pin to a constant 60% duty cycle. This is because after various trials were run, we determined 60% to be the optimal duty cycle while still minimizing the risk of the MOSFETs overheating.

2.3.5 User Interface Subsystem and Software Design

Mechanical

The main board Ebox includes a cover for mounting the user interface screen, as well as holes for the user interface buttons and rotary encoders to pass through. See Figure B.2.1 in Appendix B.2 for a look at the user interface mechanical design.

Electrical

The user interface subsystem was simple from an electrical perspective, as it only contained several buttons and encoders, with which I had to implement a debouncing circuit to allow the STM32 time to recover from repeated button presses by the user. The 14-pin connector for the screen just had to be wired to the corresponding STM pins with the proper functions and then placed close to the chip for easy routing.

Software

We utilize a 4.0’’ ST7796 LCD with via Serial Peripheral Interface (SPI) protocol. Relevant connections to the microcontroller include SPI configured pins, such as the Master In Slave Out (MISO), Master Out Slave In (MOSI), clock (SCK), as well as a few GPIO: Chip Select (CS), Data Command (DC), and Reset (RST). We utilized device drivers for a 4.0’’ ILI9488 display and modified them accordingly.

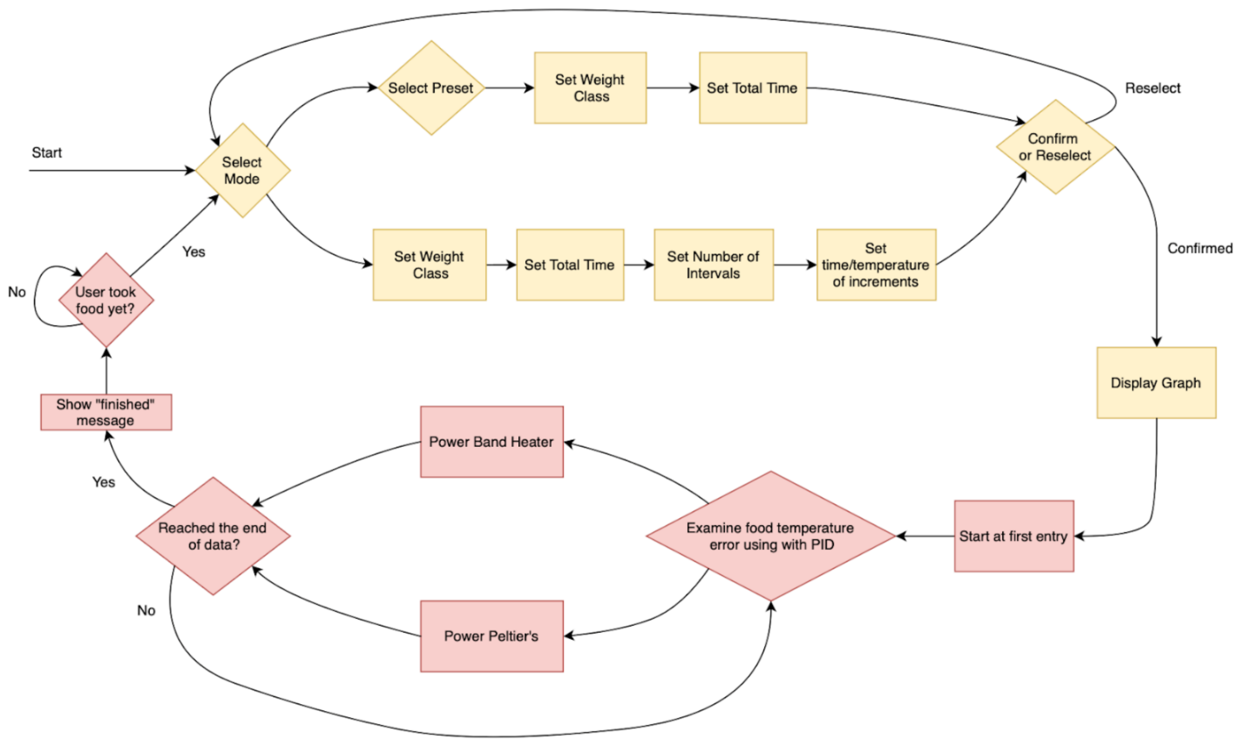


Figure 3: State Diagram of Software Design

Yellow blocks correspond to the setup phase in software, where the user enters parameters of the heating curve he or she wishes to set, whereas red blocks pertain to the active operation phase, where power to all modules is supplied and the device attempts to follow the relevant heating curve. We will now describe the most important blocks and their design decisions.

Minor Setup Submenus

Figure 4: Various Setup Phase Submenus

Most of the menus in Figure 3 are self-explanatory and simple. The main menu is not pictured in Figure 3, as it only has a few options. In this menu, the user has the option to set the text color, to the temperature scale (°F) or (°C), or to create and run a heating curve. If the user wishes to set a heating curve, the user is presented with two options: the first being a "Preset" mode which allows the user to select from a menu of pre-programmed heating curves so that he does not have to set the data curve manually. The other option is "Entry" mode, allowing the user to select how many data points they wish to construct a heating curve from–and then set those time vs temperature data points. The user then sets parameters such as the total time, weight class, and number of intervals they wish to control, all of which are done by using the buttons and rotary encoders.

Data Entry Submenus

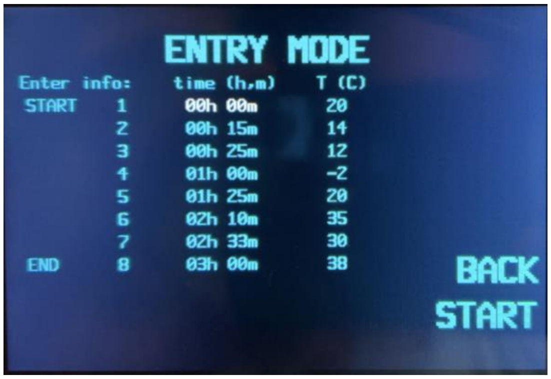


Figure 5: Data Entry Screen Example

Figure 5 shows an example of a user setting a heating curve of eight intervals the user will set. The column of times is stored as an array, and the temperature column to the right is stored as the corresponding temperatures array. Upon selecting "START" the data is then graphed and displayed on the LCD in a clear and user-friendly manner. Operation begins, where the device begins to follow the heating curve, where at any point the user may return to this screen and modify the data. Construction of the graph will be discussed in the verification section.

2.3.6 Temperature Sensing Subsystem and Temperature Control Design

Mechanical

As stated previously, we utilized two methods to measure temperatures, the first of which being an NTC thermistor embedded inside a metal probe to measure the food temperature inside the chamber. Three TMP1075 breakout boards around the outer wall of the inner chamber, spaced 120° from each other to obtain temperature readings from all portions of the chamber. These sensors are placed inside of the Styrofoam insulation and have thermal pads connecting them to the wall of the inner chamber, meaning they should share the same temperature as the adjacent chamber wall.

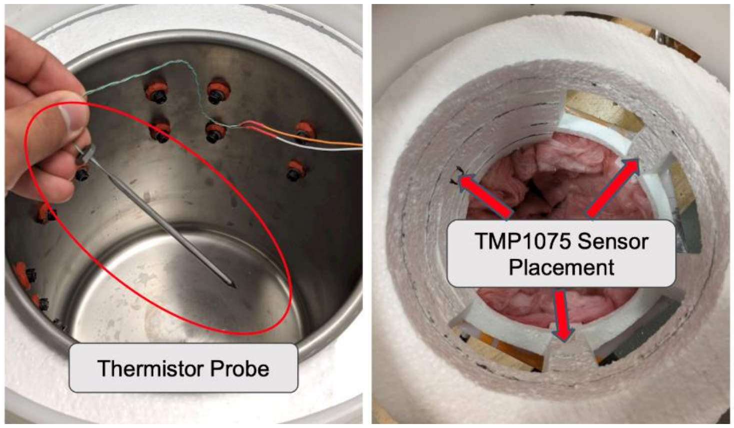


Figure 6: Mounting of Thermistor and Temperature Sensor Boards

Electrical

We had two electrical systems meant for measuring temperature. The first one being analog thermistors, which were chosen because of their superior measurement resolution at our temperature range as well as the ease of hardware implementation, as all we had to do was design a resistor divider and then pass that signal through an op amp buffer.

The second system that was developed was our external TMP1075 breakout board. The schematics and designs for this board can be found in Appendix C. We effectively designed an external board which had an RJ12 connector and a TMP1075 I2C temperature sensing IC on it. The board then allowed us to daisy chain connectors together on the I2C bus and place sensors around the enclosure while providing an accurate temperature reading that was not influenced by tuning variables or external analog noise like the thermistor.

In general, our design philosophy behind the temperature sensing subsystem was providing redundancy to have multiple sensors and technologies so inaccuracies could be reduced.

Software

Temperature Reading

Using Figure C.1.3 located in Appendix C.1, we read the analog voltage and work backwards to calculate the resistance of the thermistor. We then can use the relation between temperature and resistance of a thermistor:

$$R_1 = R_0 e^{\beta(T_1^{-1} - T_0^{-1})}$$

Equation 1: Resistance of Thermistor

$$T_1 = \frac{BT_0}{T_0 \ln\left(\frac{R_1}{R_0}\right) + B} - 273$$

Equation 2: Temperature of Thermistor

Where T_0 is 298 K, R_0 is the resistance (kΩ) of the thermistor at T_0, B =3500 K is the resistance temperature dependance, T_1 is the temperature of the thermistor (K), and R_1 is the resistance (kΩ) of the thermistor at T_1. Equation 1 can be rearranged to form Equation 2.

Internal Clock

Operation phase consists of a loop where in each iteration, food and chamber temperature are read, and the microcontroller polls for user input. In each iteration, we read from a timer register configured at 1 MHz. The timer counts upwards from 0 and uses 16 bits, implying a maximum value of 2^{16} -1 = 65535 microseconds. The elapsed time is found by comparing the value of the register with the value read during the last iteration. Furthermore, to account for overflow, we take use the relation

$$t_{elapsed} = \left(t_1 - t_{last}\right) \bmod (65\,535)$$

Equation 3: Elapsed Time (μs) in Each Iteration

Where $t_{elapsed}$ is the elapsed time in each iteration, t_1 is the value read from the register in the current iteration, and t_{last} is the value read from the last iteration.

This implies that the code executed within each iteration takes less than 65535 microseconds, or about 65 milliseconds, which it does.

PID Loop

The following equation set is used to calculate the PID value.

$$\begin{gathered}
\Delta T_{error} = T_{target}(t) - T_{food}(t) \\
P = 0.01 k_p \Delta T_{error} \\
I = 0.01 I_{last} + k_i \Delta T_{error} \\
D = 0.01 k_d (\Delta T_{error} - \Delta T_{prev}) \\
PID_{total} = P + I + D
\end{gathered}$$

Equation 4: PID value calculation

Where T_{target} (t) is the target temperature interpolated from the data the user set at that time, T_{food} (t) is the actual food temperature, ΔT_{error} is the error between food and target temperature at any given time (intervals measured in minutes), ΔT_{prev} is the previous temperature error (from one second ago), P is the proportional error value, I is the integration error value, D is the derivative error value, k_p is the proportional error scaling constant, k_i is the integral scaling constant, k_d is the derivative scaling constant, PID_{total} is the value which determines to power the heaters or the coolers. The constants k_p , k_i , and k_d were chosen such that a typical PID_{total} value lies within the interval [-255, 255]. Essentially, if PID_{total} is positive, the heater is powered, and if it is negative, the coolers are powered.

3. Design Verification

Our design subsystems were verified in many ways, and throughout the verification process, we ran into multiple issues, meaning we had to make some design changes in our final version of the PCB and mechanical design.

3.1 Power System Verification

The power system was first assembled by itself and then verified using a programmable load. Initially during the verification, when plugging in our buck converters, the converters drew a large amount of power and shorted themselves from power to ground. After looking at the data sheet one more time, it was revealed that our initial design had connected the enable pin to 24V even though the enable pin was only rated for 5.5V. As a result, a resistor divider was added in later revisions and a bodge had to be put in the first revision of the board to allow the proper voltage on the enable pin. Please refer to Figure C.2.2 in Appendix C.2 for a picture of the bodge and an image describing the design change we made.

Once the enable pin was tied to the proper voltage, the buck converters worked without any major problems. They were then verified to be within our initial specifications of wattage independently using a programmable load. Below in Table 1, one can see the power outputs read on the programmable load. A few of these values are roughly 10 percent off our final value and this can be attributed to voltage sag on the lines going to the load. I also measured the voltage at the source and if that is factored in, the correct power was reached. Please refer to Appendix C.4 for the pictures of the load readings and the voltage measurement at the source.

Table 1: Power Verification Table

Power Regulator	Power Specified	Power Verified (at load)
24V AC-DC Power Supply	100W	101.6W
12V Buck	5W	5.67W
5V Buck	10W	8.176W
3.3V LDO	1W	0.89W

3.2 Central Control Subsystem Verification

The Central Control Subsystem was verified by simply flashing a blink sketch to the STM32F401RE, this confirmed that the device could be flashed and programmed through ST-LINK and that our GPIO initialization operated correctly.

3.3 Heating Subsystem Verification

Since the heating subsystem consisted of just two relays, they were verified using a simple GPIO toggle sketch on the STM32 and then their output was measured using a multimeter to verify that the relays shorted during the specified GPIO state.

3.4 Cooling Subsystem Verification

The cooling system is where the most issues were experienced. We initially began the verification process by assembling the half bridge components and giving a 50% PWM duty cycle at 10Khz a try on the output. That output on the oscilloscope can be seen in Appendix C.4. Once that wave was verified, we added capacitance to the output and this time when plugging in the same system. The current draw grew to the limit on the power supply and then the gate driver physically popped and broke. I had to look at my schematic again and determined the bootstrap setup could not handle output capacitance. As seen in the below schematic, the bootstrap capacitor (C23) is charged when the switching node (SWN1) is connected to ground. The charge on this capacitor is then used to drive the gate of the high side MOSFET (Q3). So, this means that capacitance on the output, or switching node, would mean that the voltage on the capacitor C23 would not be 12V but rather something smaller or even negative given a high enough output PWM. Thus, the high side switch can never be turned on.

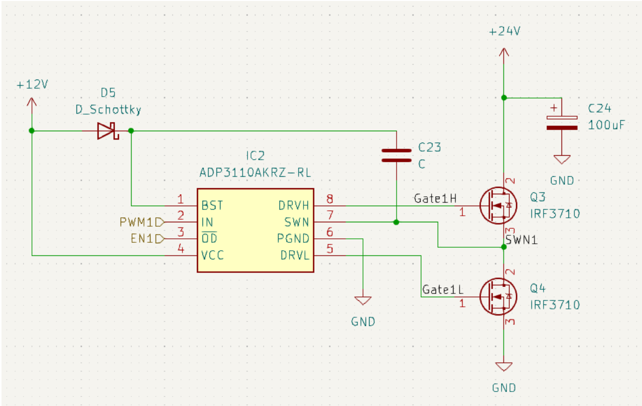


Figure 7: Boot-Strapped Half Bridge Schematic

So, to fix this issue, an inductor had to be placed on the output of the board to account for the switching node to stay at a constant current but change its potential, thus allowing the bootstrap cap to charge. Once this was done, the system was stable at no load. But when put under a large load, the system began to run into heating issues. We also needed to create an external high power LC network to account for this new topology and had to buy custom components to handle the 24V and up to 7A current draw on our output. Once this new LC network was put on, high-power loads began to experience very high heating, and a heatsink and fan was placed on the output MOSFETs. For more visual information on how they were installed, please refer to Appendix C.2. Even with this heatsink and fan combo, we were still overheating, so we were forced to operate the coolers at a lower PWM, meaning that we lost out on quick cooling and had to operate in a slightly less efficient regime, although we were still able to suffice our high-level requirements and cooling requirements.

At the end of the day, we settled on using the bootstrapped network with a large inductor and capacitor on the output and a beefy heatsink being run at a lower PWM. This means that we were operating in a slightly less efficient voltage range, and this is the one part of the project we would like to fix in the future to truly make the device work as we initially envisioned.

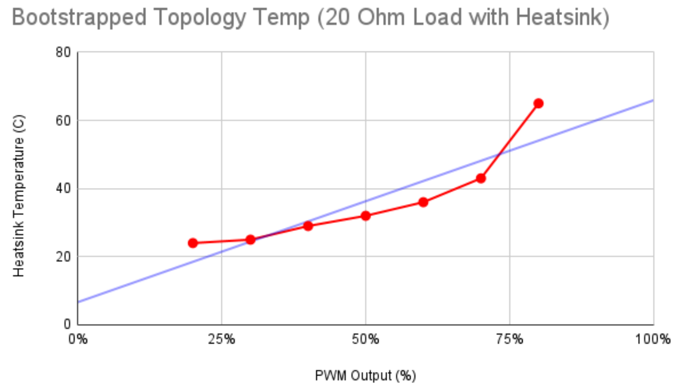


Figure 8: Graph Depicting the Heating of Bootstrap MOSFET's during High Power Operation

3.5 User Interface Subsystem Verification

The user interface subsystem works hand in hand with the control subsystem, so upon the verification of said subsystem, most of the requirements for this one will already be met. Verification of the storage of the data the user entered is done by plotting the time vs temperature graph. This graph also serves as a neat visual. Below is an example of a heating curve data followed by the plotted heating curve.

Table 2: Example Heating Curve Data

Index	0	1	2	3	4
VALID_TIMES (minutes)	0	10	13	37	40
VALID_TEMPERATURES (℃)	28	40	40	15	15

In Table 2, the times array (of size n = 5) is denoted as VALID_TIMES which each value is in units of minutes. The corresponding temperature array is denoted as VALID_TEMEPRATURES in units of ℃. To construct the graph, we need to first determine the vertical and horizontal scales. The vertical scale (temperatures) is determined by whether the Celsius or Fahrenheit temperatures are used. The horizontal scale (times) is determined by the total time of the heating curve. Thus, the number of pixels in the x-direction representing one minute, as well as the number of pixels in the y-direction representing one degree of temperature in either Fahrenheit or Celsius must be known to plot each data point. Additionally, the LCD display is 480 $\times$ 320 pixels, where higher y-coordinates pixels are located lower on the screen. Specific equations about data transformation to the LCD graph are in Appendix D.

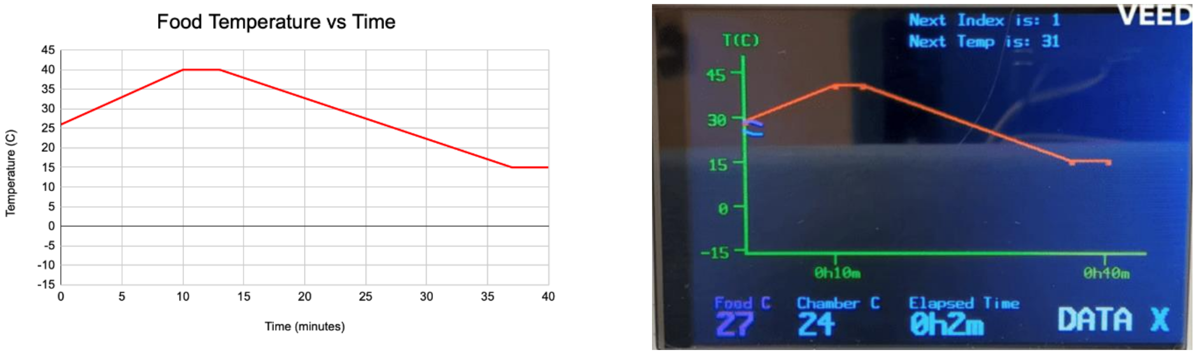


Figure 9: Comparison between Graph and LCD Drawn Graph from Table 2 data

Both plots are equivalent implying that the data is stored correctly and that the equations used to construct the graph which were presented in the previous section for drawing are correct.

3.6 Temperature Sensing Subsystem and Temperature Control Verification

3.6.1 Temperature Sensor Verification

The thermistor and three PCB integrated temperature sensors are verified by comparing their values with an external thermometer of sorts. Many tests were performed within the temperature ranges 0-40 °C (the minimum requirements for our design) where the reading on each sensor was first read from the microcontroller and then compared with a known value taken by measuring the temperature of that point in the chamber using an infrared temperature gun. The same verification method was repeated for measuring the temperature of the thermistor inside the probe lying within the chamber.

3.6.2 Temperature Control Verification

The internal clock is easily verified by comparing the value printed on the screen with an external clock. Since we already verified the temperature reading and the graph construction, to verify the PID loop, we examine the displayed food temperature on screen, read the current time elapsed to determine where we should be on the curve, and observe whether the heater or the TECs needed to be powered. Using a multimeter, we could directly verify the voltage feeding into the Peltier module when cooling was needed. Likewise, a multimeter was used to verify the relay switching for when the band heater needed to be powered.

3.7 Verification of all High-Level Performance Requirements

This section highlights that we achieved all performance requirements stated and how we verified them.

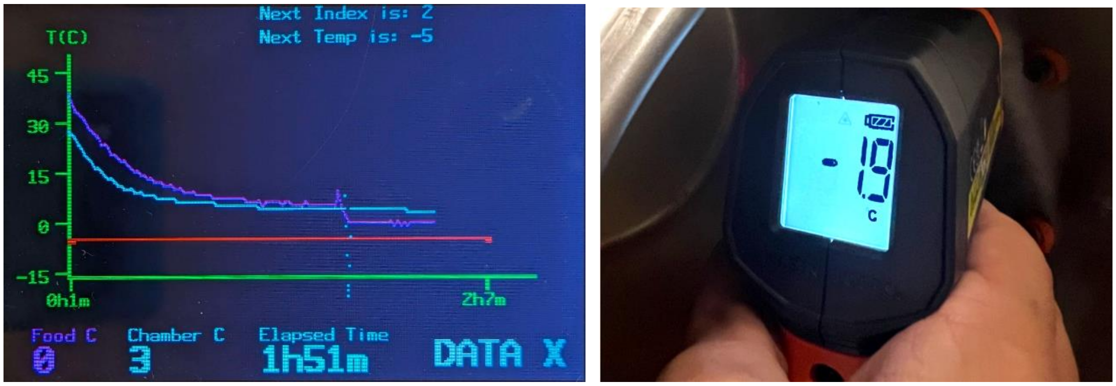


Figure 10: Cooling Chamber down to 0 ℃ from 40 ℃

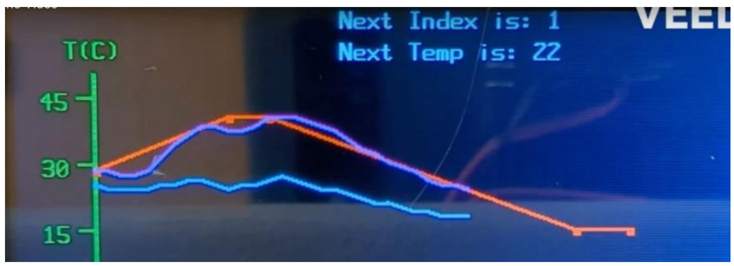


Figure 11: Desired Heating Curve (red) vs Actual Food Temperature (purple)

Figure 11 is also plotted from the data from Table 2. Figure 10 shows the satisfaction of our first performance requirement, drawing a heating curve that the food temperature will follow. Together, Figures 10 and 11 show that our second high level requirement of reaching temperatures as low as 0 ℃ and as high as 40 ℃ was achieved. We can now examine the error between the desired and actual temperatures using Table D.2. 1 located in Appendix D.2.

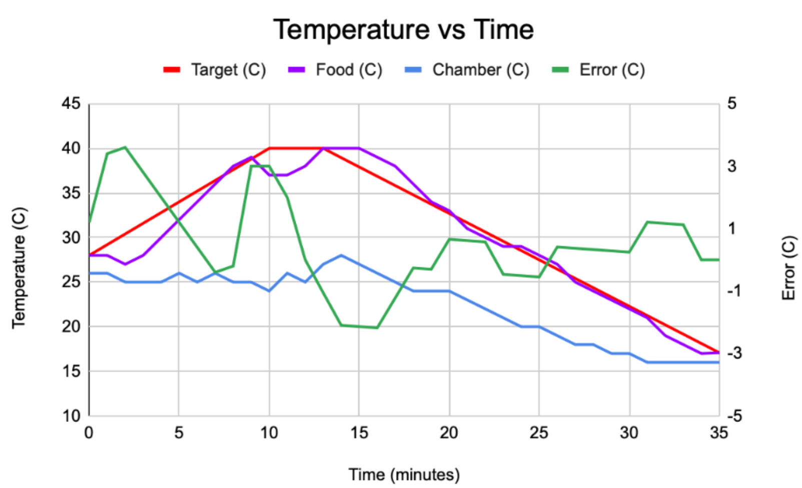


Figure 12: Graph of Error Between Target Temperature and Food Temperature

The error (green) is the difference of the target temperature (red) and actual food temperature (purple) curves in units of °C and is plotted on the right vertical axis. Figure 12 shows that over the course of 35 minutes, despite either heating or cooling, the error in temperature never drops below -5 ℃ and never exceeds 5 ℃.

4. Costs and Schedule

4.1 Parts

Parts were purchased from multiple sources and ended up exceeding our allotted $150 budget.

4.1.1 Electrical Cost Analysis

PCBs were ordered through the ECE 445 PCB order process. The ECEB Supply Center supplied most of the passive components and a few small integrated circuits. Finally, we ordered the rest of the parts from Mouser and Amazon. Many of our passive components and user interface parts were already accounted for. We assumed a build volume of five boards, because that is the minimum order quantity from our PCB manufacturer and from experience, having this many backups is useful during testing and possible mistakes. Additional parts were found online and at the ECEB Service Center. Finally, we found a 4.0'' 480x320 Touch Screen which costs $20.73 after shipping. So, in total, after $25.04 for the PCBs, $35.49 for the Mouser Parts, and $20.73 for the screen, all the electrical components incurred a total cost of $81.23 to produce 5 boards.

The preceding analysis concerns the expenditure associated with the necessary items. However additional purchases were necessitated due to the failure of certain parts during integration such as PCBs and TECs. For more information on specific components that were purchased and exact quotes for our parts, please refer to Appendix C.3.

4.1.2 Mechanical Cost Analysis

Many of the mechanical parts were purchased in person, but a few came from various university sources and our project funding. After adding everything together, our mechanical portion of the project added up to $200.99. For more information on specifically what was bought, please refer to Appendix B.5.

4.2 Labor

We could reasonably expect salaries of around $50/hr. Over the course of about 2 months, we'd likely spend an average of 3 hours per day working. So, the total labor cost will be $50/hr x 2.5 hr/day x 60 days x 3 people = $22,500.

4.3 Total Cost

Appendix C depicts cost estimates for the electrical and mechanical portions of the project. With the electrical cost at $81.23, the mechanical cost at $200.99, and the labor cost at $22,500, the total cost of the project will be $22,782.22.

4.4 Schedule

The revised final schedule describing each group member's weekly responsibilities is shown in Appendix E.

5. Conclusion

5.1 Accomplishments

In sum, all subsystems were verified. The PCB was fully functional and communicated accordingly with the program running on the microcontroller, working together to heat and cool to follow arbitrary heating curves as planned. We were able to reach 0 ℃ and 40 ℃. Section 3.7 briefly confirms these achievements and shows the accuracy at which the food temperature can follow a target temperature curve, which concludes verifying the three performance requirements.

5.2 Uncertainties

The main uncertainties with our current project lie within the power electronics for our cooling system. As explained in the Cooling System Verification section, we are currently running our cooling system at a power below the ideal voltage output because of overheating issues. There are still occurrences of our system failing and while we were able to get it to a stable state, sometimes the MOSFET driver dies during extended operation. It is these uncertainties in design that made us think of switching to some isolated gate drivers and possibly dedicated buck converters with constant current capabilities to improve these uncertain elements.

5.3 Ethical considerations

The drastic temperature changes this device exhibits requires heating elements located toward the bottom of the container. Without proper precaution, this could pose fire hazards, or at the very least, extreme heats unsafe for human proximity. The integral component of the device itself is the temperature sensing, so the internal temperature will never exceed the maximum that we set. Layers of fiberglass will be used around and underneath the container and between the container and the lid, which will act as heat insulation. Also, heating wires at high voltages poses significant safety risks, especially if short circuits or other failures occur. Our original plan was to use resistive coils to act as the active heating, but after further consideration, we decided to substitute this heating method with a band heater. The band heater wraps around the food/drink to be heated, reducing the potential electrical fire hazard or even direct burn risk for this subsystem. Also, we will incorporate as many enclosures as possible to isolate subsystems when possible. The electric box will be located outside the outer chamber, ensuring it will not be affected and cause any electric hazards by the extreme temperature changes inside. As stated in the Institute of Electrical and Electronics Engineers (IEEE) Code of Ethics Section I, it is our obligation to "disclose promptly factors that might endanger the public or the environment." The highest safety priority is that the user operates the device with absolutely no risk to themselves or others and we will take all precautions to ensure this. Keeping this in mind, our user interface system will be efficient yet intuitive to minimize all difficulties inexperienced users could have with the device.

This device offers flexibility and adaptability to different cooking and food storage requirements, particularly in areas with limited access to conventional kitchen appliances. Economically, this innovation could lead to cost savings for consumers by consolidating two essential kitchen functions into a single device, potentially reducing the need for separate appliances and associated expenses. From an environmental standpoint, the efficiency gains achieved by minimizing energy consumption and space requirements could contribute to sustainability efforts, reducing overall resource usage and carbon emissions. Societally, the accessibility of a versatile cooking and refrigeration solution could enhance food preparation capabilities, dietary diversity, and culinary creativity for individuals and communities, fostering improved nutrition and quality of life.

5.4 Future work

Given more time, we feel that we could redesign the mechanical, electrical, and software systems mainly pertaining to the Peltier cooling capabilities, as our device was able to heat much faster than it cooled. This would improve the cooling rate, allowing the device to follow more drastic cooling curves.

From an electrical perspective, we want to further improve the quality of our high-power systems while improving the quality of life of dealing with the electronics. So, in the future, we would like to move the user interface buttons and encoders to another board while moving the screw terminal connectors and programming pins to the front of the board to more easily access and wire the device

On the cooling system side, we would want to move to isolated gate drivers to improve the quality and reliability of our system as well as transitioning to a dedicated buck converter to drive the output with constant current and operate the Peltier modules at their maximum efficiency. Hopefully when these changes are made, we will have a much more reliable device that operates at its maximum capabilities.

On the mechanical side, there are several changes we'd make if we were to build this project again. First of all, we'd use an aluminum inner chamber, because the low thermal conductivity of the stainless steel inner chamber we used increased the temperature gradient across the chamber, and made it more difficult to precisely control the chamber temperature. The temperature we would measure on the thermistor temperature probe would be significantly different depending on where we placed it in the chamber. The chamber and our temperature sensors were slow to respond to heating and cooling due to this low thermal conductivity. Using an aluminum chamber with higher thermal conductivity would make it easier to get our PID loop to be accurate. Secondly, we'd like to design another revision of the main board Ebox to accommodate the heatsink for the power electronics. Thirdly, we would probably use a higher number of Peltier coolers because it seems like four was the bare minimum we needed to reach below freezing, and the idea of this project was optimistically to reach several degrees below freezing. Finally, we would add an RGB LED to the main board Ebox that would act as an indicator to the user as to what mode of operation the device is currently in. We were thinking it could shine red during a heating phase, blue during a cooling phase, and green when holding temperature, ready for the user to extract their food.

Specifically, during the transition from heating to cooling, occasional instances of temperature overshooting occurred, leading to elevated temperatures. Subsequently, the cooling process required several minutes to rectify this deviation. By revising the circuitry, there is potential to enhance the efficiency of the TECs by operating them at an elevated duty cycle, thus improving overall system performance. Also, in software, we could further examine the feedback loop used to determine if the heater or the TECs need to be powered. Specifically, we could run further tests on the amplification constants of the proportional, integral, and derivative correction factors and hopefully minimize overshoot from the heater.

References

[1] Conrad Electronic SE. “Specification of Thermoelectric Module.” (), [Online]. Available: https://asset.re-in.de/add/160267/c1/-/en/000189115DS02/DA_TRU-Components-TEC1-12706-Peltier-Element-15V-6.4A-65W-L-x-B-x-H-40-x-40-x-3.8mm.pdf (visited on 02/08/2024).

[2] Conrad Electronics SE. “Specification of Thermoelectric Module TEC1-12706” [Online] Available: https://asset.re-in.de/add/160267/c1/-/en/000189115DS02/DA_TRU-Components-TEC1-12706-Peltier-Element-15V-6.4A-65W-L-x-B-x-H-40-x-40-x-3.8mm.pdf (visited on 02/08/2024).

[3] IEEE. ”IEEE Code of Ethics.” (2024), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2024).

[4] Meerstetter Engineering. “Peltier Elements.” (2024), [Online]. Available: https://www.meerstetter.ch/customer-center/compendium/70-peltier-elements#D_Heatpumped%20vs%20Current (visited on 02/08/2024).

[5] Meerstetter Engineering. “Peltier Element Efficiency.” (2024), [Online]. Available: https://www.meerstetter.ch/customer-center/compendium/71-peltier-element-efficiency (visited on 02/08/2024).

[6] The Engineering ToolBox “Thermal conductivity of fiberglass insulation - temperature and k-values.” [Online] Available: https://www.engineeringtoolbox.com/fiberglas-insulation-k-values-d_1172.html (visited on 02/08/2024).

Appendix A Requirement and Verification Table

All our requirements were verified in Section 3. This is shown with Y in the Verification Status column.

Table A.1: Power Supply Subsystem

Requirements	Verification	Verification status (Y or N)
24V Output at minimum 100W	• Wire power supply to wall outlet • Set up test setup with power supply connected to programmable load with high gauge wire • Once everything is set up, plug device into wall and verify correct power output on load	Y
5V Output at minimum 10W	• Once 24V power verified, assemble 5V buck on the board • Use the 5V test points on the board to connect programmable load and program load to correct power output • Switch the buck on and verify correct power draw is reached with minimal (less than 250mV) voltage sag	Y
12V Output at minimum 5W	• Once 24V power verified, assemble 12V buck on the board • Use the 12V test points on the board to connect programmable load and program load to correct power output • Switch the buck on and verify correct power draw is reached with minimal (less than 500mV) voltage sag	Y
3.3V Output at minimum 1W	• Assemble LDO component on the board • Use the 3.3V test point to connect programmable load	Y

	• Ensure proper power can be drawn from LDO and minimal voltage sag (less than 150mV) is achieved on the output of the LDO	

Table A.2: Central Control Subsystem

Requirements	**Verification**	Verification status (Y or N)
Control PWM and polarity of output to Peltier modules	• We will assemble the half bridge component of the PCB • Then, we will apply a 50% PWM signal to each half bridge while applying a 0% PWM signal to the other side • We then read an oscilloscope measurement on the output and ensure that we are getting the proper outputs with the proper polarities	Y
Control user interface and user input from on board touch screen and buttons/encoders	• Assemble the user interface components on the board • Connect screen, buttons, and encoder • Write and flash echo sketch, where user input is piped into feedback LEDs on the board and to the serial monitor	Y
Be able to switch power going to heating modules	• Assemble the heater subsystem and relays • Use slow PWM to switch the system on and off to ensure that relay control is working properly • Verify by hooking up multimeter to output and making sure it works	Y
Can read temperature from various points on the product and process values to control temperature outputs	• Assembled temperature sensing subsystem on board • Connect external thermistors and TMP sensor sub-boards	Y

	• Use a similar echo sketch to pipe sensor data to the serial monitor and ensure proper readings by using and external temperature sensor to confirm readings	
Capability to set heating curves and execute them over extended periods of time.	• First, control of power going to the heating and cooling subsystems needs to be programmed • Once that is done, program the ability to use the RTC on the STM to follow setting heating and cooling points • Verify temperature stability by programming heating curves and using external thermocouples and temperature sensors to ensure the system is following the proper temperature profiles	Y

Table A.3: Sensor Array

Requirements	**Verification**	Verification status (Y or N)
Accurately relay temperature information (within 5 degrees C) back to STM microcontroller	• Assemble the Sensor Array part of the PCB after verifying both heating and cooling components • Connect sensors and breakout boards • Turn on heating components, measure temperature using both sensor technologies on the board, then measure the same location using an external thermocouple • Compare the temperatures from all three values and attempt to tune the temperature readings from both sensors to fall into the same value as the external thermocouple	Y
Multiple temperature sensing options to verify measurements and ensure proper operation	• Design a PCB with various temperature options • Verify each option independently using the same method as described above	Y

	• Use both sensing options in order to get redundant measurements in case of sensor malfunction or misuse	

Table A.4: Cooling Subsystem

Requirements	**Verification**	Verification status (Y or N)
Cool the product to freezing temperatures (0°C)	• Construct the mechanical components of the build • Attach necessary cooling components and their fans/heatsinks • Operate all the fans using the assembled and tested full bridge setup at the maximum allowable duty cycle of our bootstrapped gate drivers • Ensure we can reach freezing or below at these maximal conditions	Y
Have the ability to switch polarity to help in cases of both heating and cooling	• Assemble full bridge assembly • Operate one side of the half bridge at 0% duty cycle and the other at the max duty cycle • Operate fans in one mode and turn them off in the other • Use external temperature sensors to verify that we are getting the correct temperature coefficient as expected	Y

Table A.5: Heating Subsystem

Requirements	**Verification**	Verification status (Y or N)
Should be able to heat chamber to above 40 C	• First assemble the relay section of the board • Plug in the heating coils into the relay, making sure we are connected to the NC contacts • Drive the relays using the STM32 and ensure our coil heats up • Use an external temperature sensor the ensure that chamber is reaching desired temperatures after giving chamber ample time to heat up	Y
Allow control of slow PWM using relays	• Connect heaters to the assembled relay section of the board • Use slow PWM (PWM period around 5s) to pulse the heater and attempt to control heating to hover around desired temperature • Will need to design feedback system using temperature sensors to ensure this can properly work	Y

Table A.6: User Interface Subsystem

Requirements	Verification	Verification status (Y or N)
The device should allow the user to set a preset heating curve from a menu to execute at a predetermined time. Additionally, the device will also allow the option for the user to	• First ensure that STM32 is working and can be properly flashed by verifying using a simple blink sketch • Once that is done, assemble the user interface portion, verifying each component as it builds by using echo sketches flashed to the STM32	Y

customize their own temperatures vs times curve.	• Program the ability to execute heating curves and test using manual heating curves as explained in the RV table for the Central Control System • Program the screen that depicts the set up of a heating or cooling curve, allow a user to input the curve they want to heat • Input and test a curve constantly monitoring using an external temperature system outside of the products internal sensors	
The device should display the current temperature of the food and chamber on the LCD.	• Verify the display can properly work • Print test text to the display with a simple test sketch • Complete verification of temperature sensors • Pass temperature sensor values in real time to the screen to allow user to read them	Y
Device should have the option to set time to ensure proper timing of device	• Verify working of RTC system as described in Central Control System • Verify screen, buttons, and encoders are properly working • Program the ability for the user to set the time registers on the device through the GUI on the screen	Y

Appendix B Mechanical References

Appendix B.1 Power Supply Ebox Mechanical Design

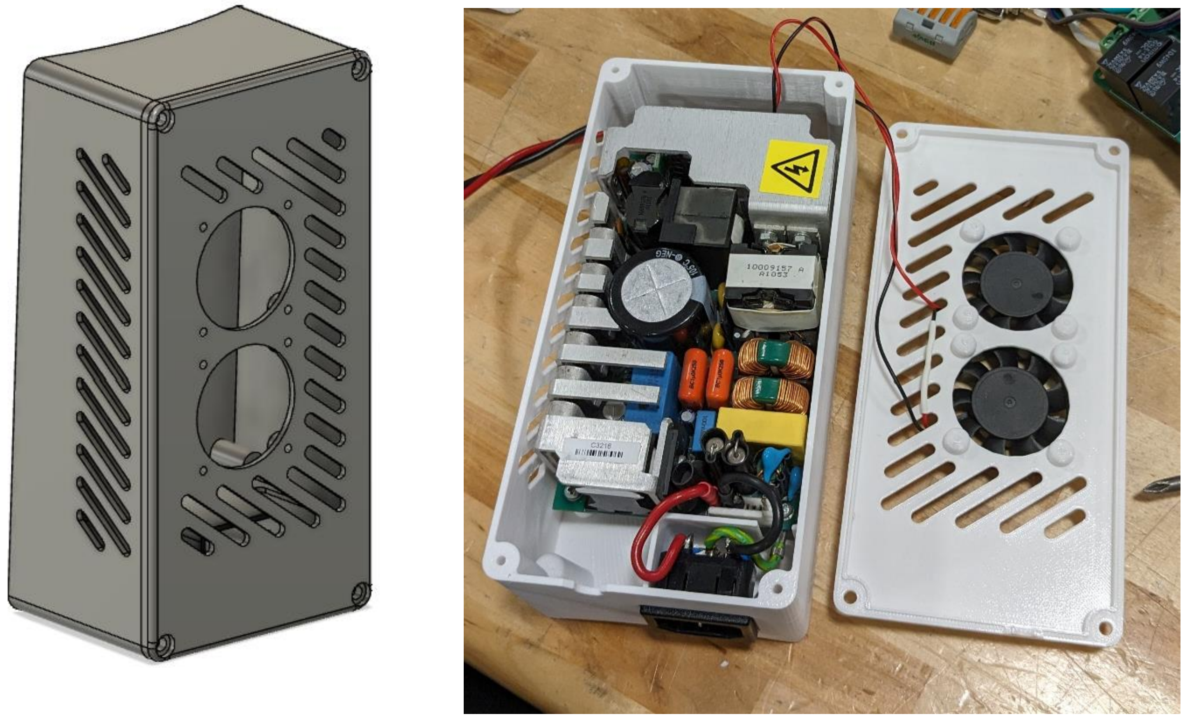


Figure B.1.1 Power Supply Ebox 3D Design and Finished Print

Appendix B.2 Main Board Ebox Mechanical Design

Figure B.2.1: Main Board Ebox 3D Design and Finished Print

Appendix B.3 Heating Subsystem Mechanical Design

Figure B.3.1: Heating Band Placement

Appendix B.4 Cooling Subsystem Mechanical Design

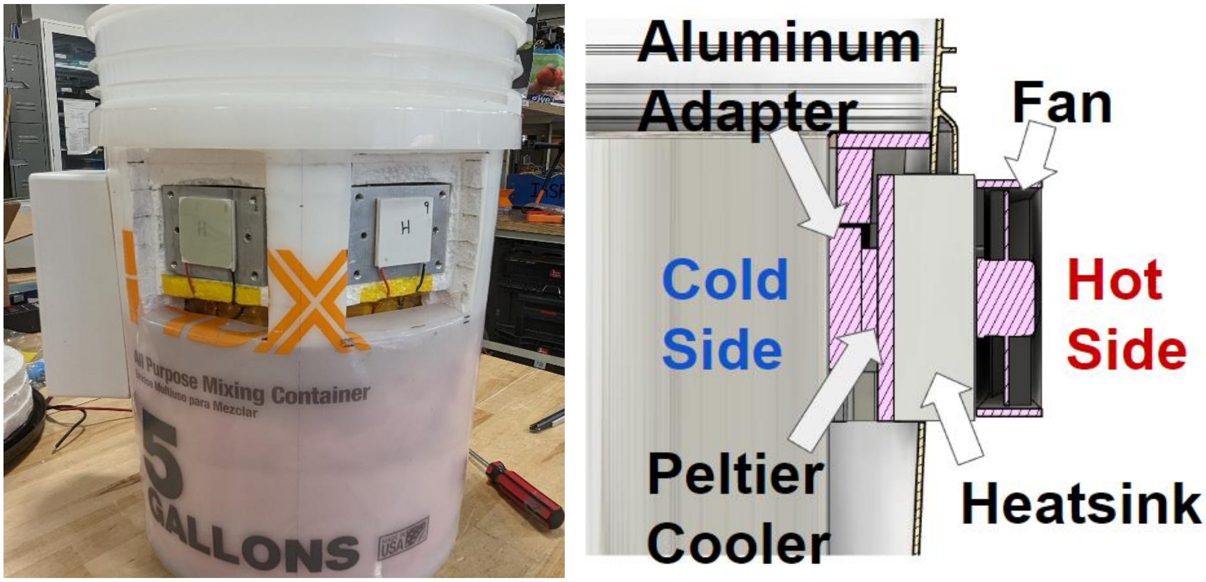


Figure B.4.1: Peltier Cooler and Adapter Mechanical Design

Appendix B.5 Mechanical Cost Analysis References

Item	Qty.	Cost per unit	Total Cost	Total Cost after Tax & Shipping	Vendor
5 Gallon Bucket	1	$6.98	$6.98	$7.61	Home Depot
5 Gallon Bucket Lid	1	$2.98	$2.98	$3.25	Home Depot
8.25 Qt SS Bain Marie, 8" dia, 9.75" ht.	1	$40.65	$40.65	$59.27	Katom Restaurant Supply
Lid for 8.25 Qt Bain Marie	1	$13.30	$13.30	$19.39	Katom Restaurant Supply
Fiberglass insulation	1	$9.97	$9.97	$10.87	Home Depot
Peltier Cooler + HS + Fan module	4	$23.54	$94.16	$100.60	Amazon
			Total Sum:	$200.99	

Table B.5.1: Total Cost of mechanical components. The total cost is $200.99.

Appendix C Electrical References

Appendix C.1 PCB Schematics

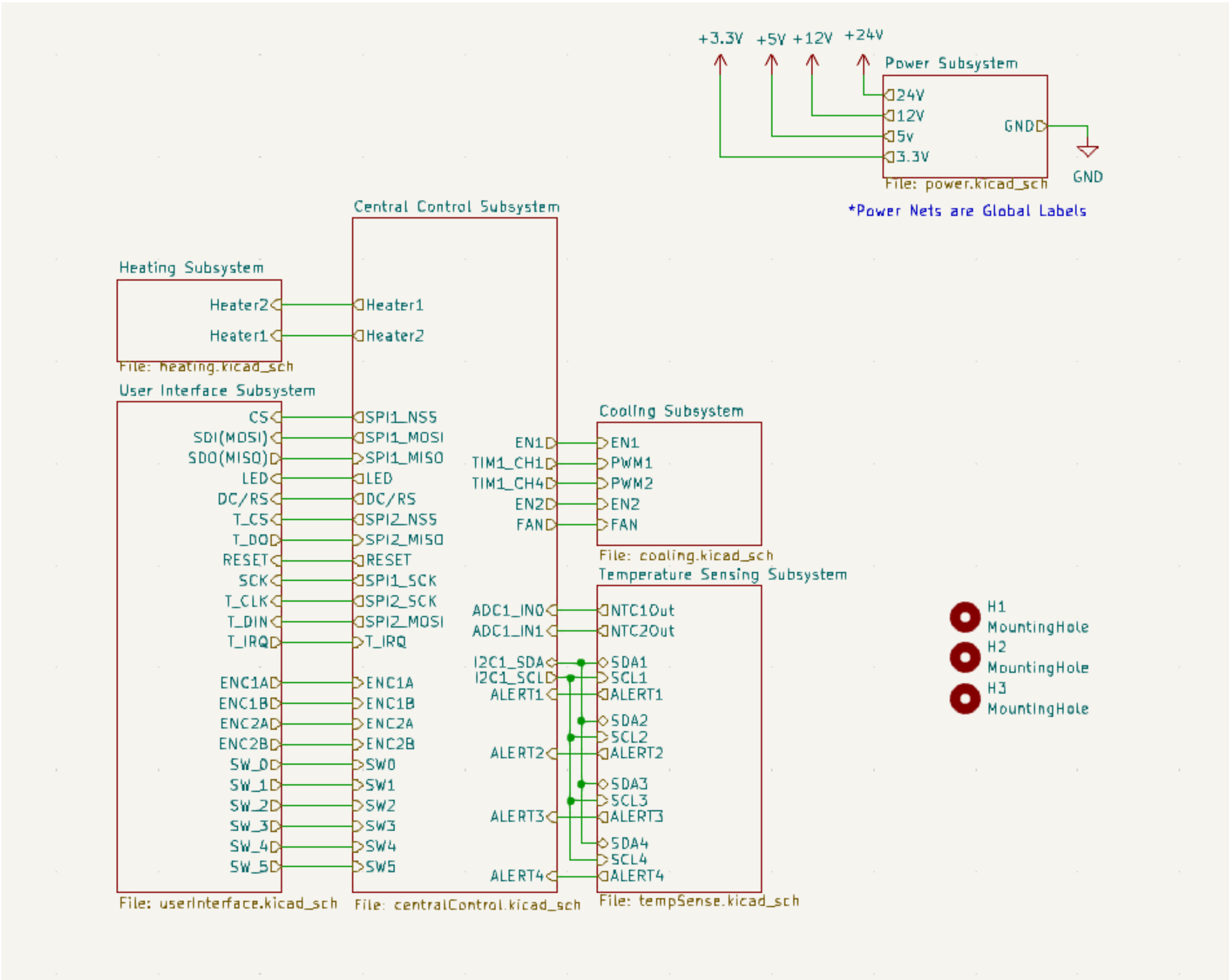


Fig C.1.1: High Level Schematic

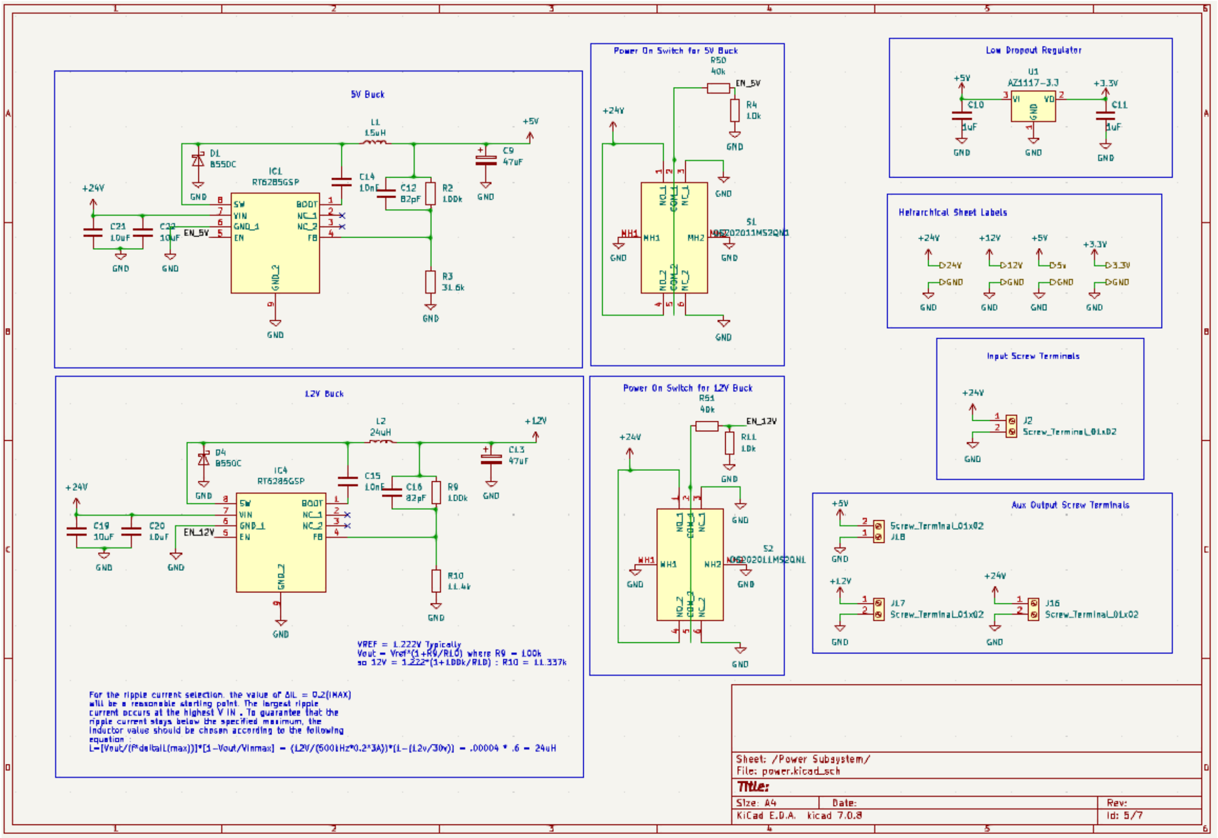


Fig C.1.2: Power Subsystem Schematic

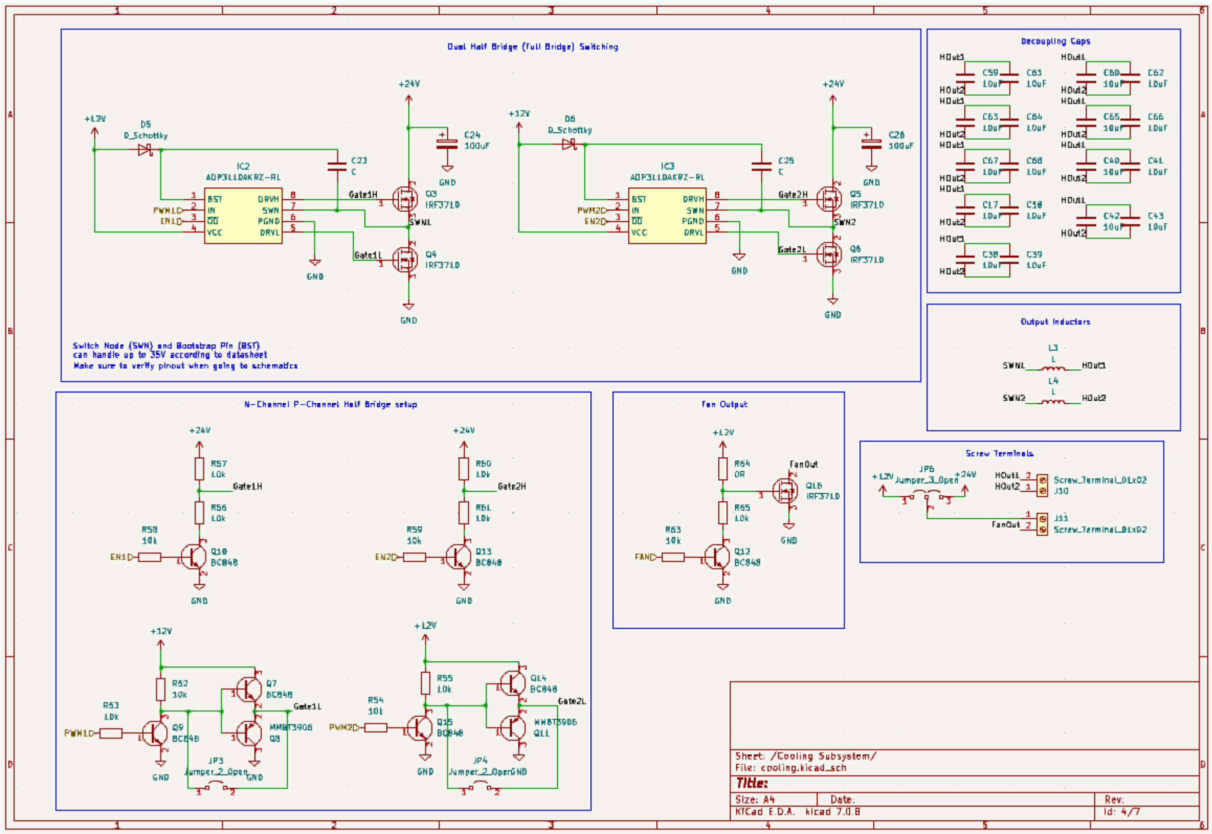


Fig C.1.3: Cooling Subsystem Schematic

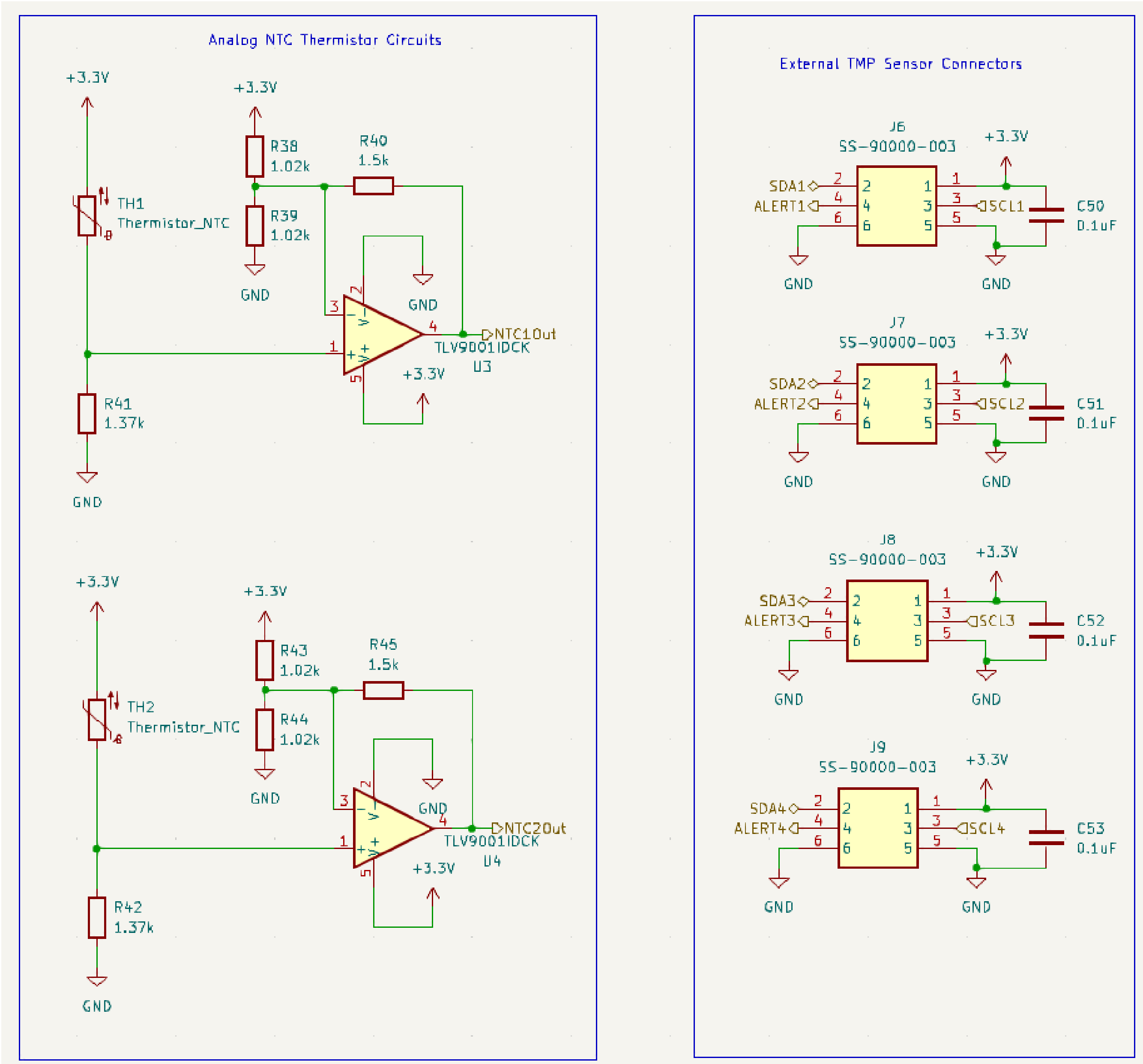


Fig C.1.4: Temperature Sensing Schematic

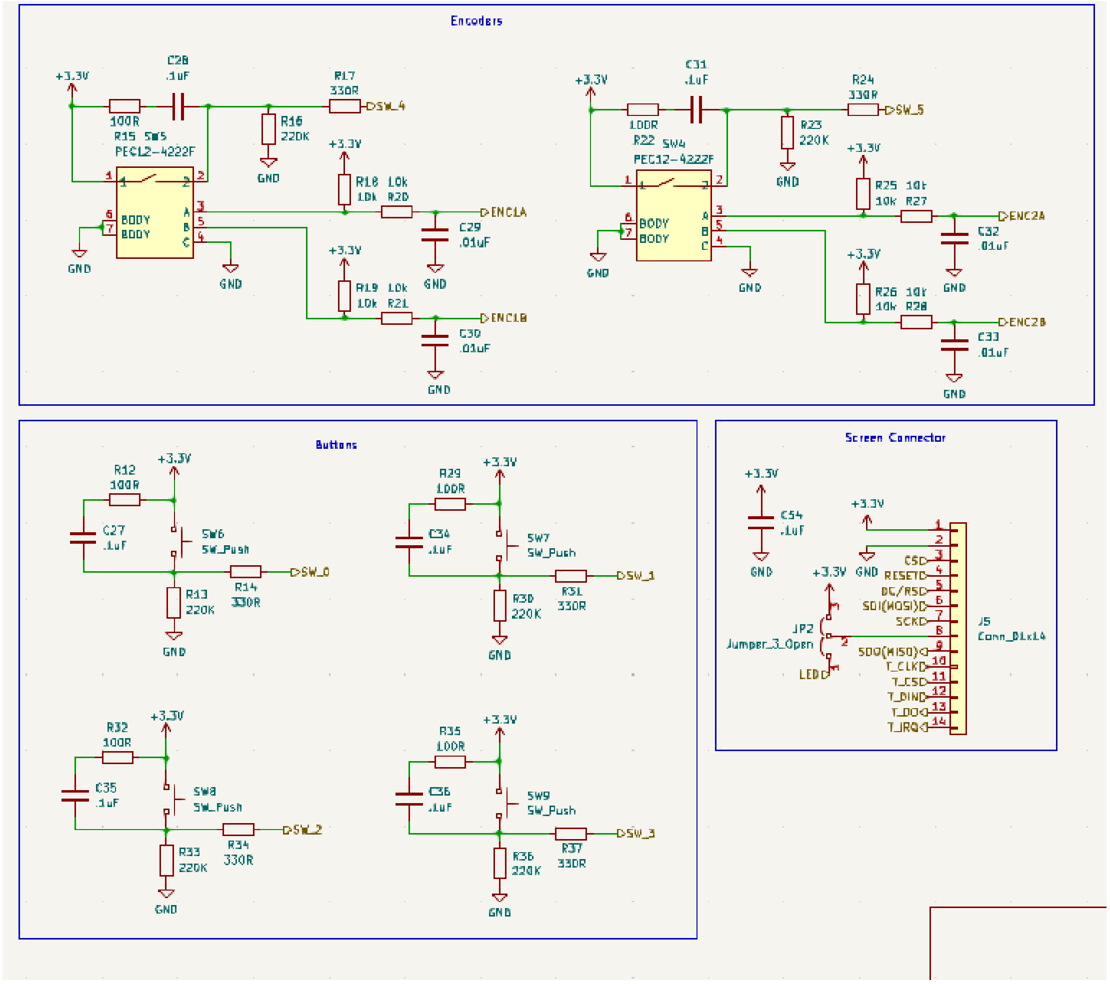


Fig C.1.5: User Interface Subsystem Schematic

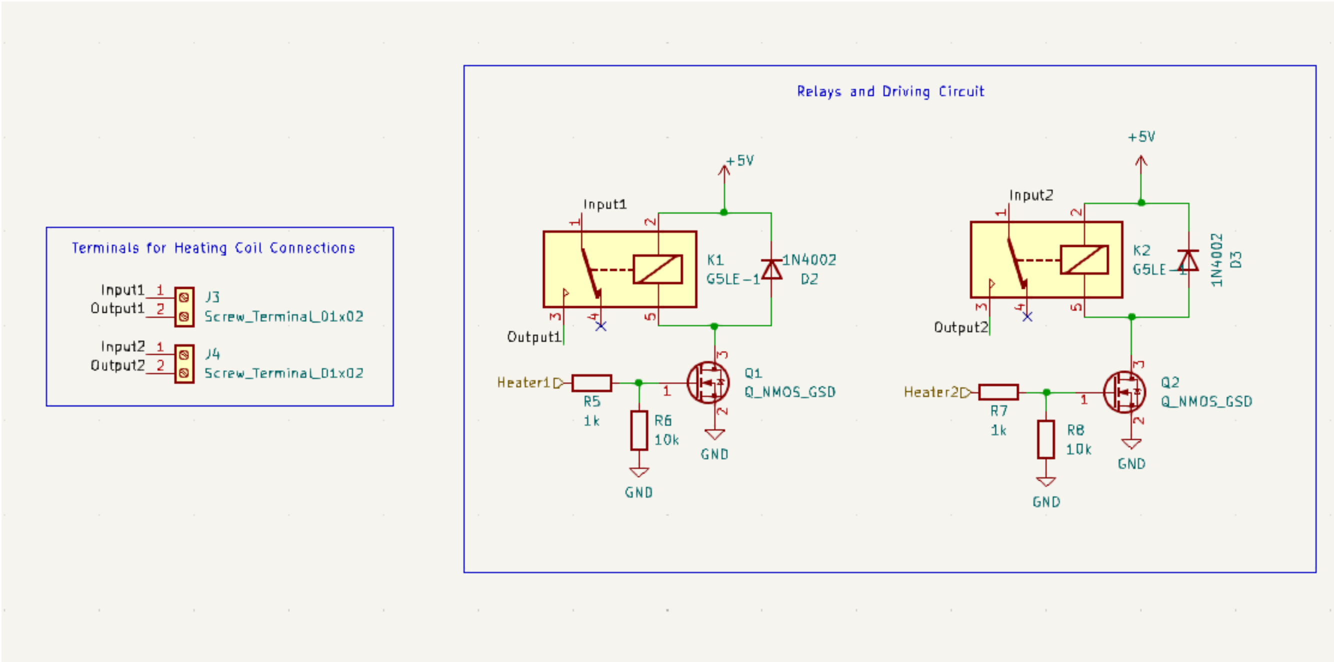


Fig C.1.6: Heating Subsystem Schematic

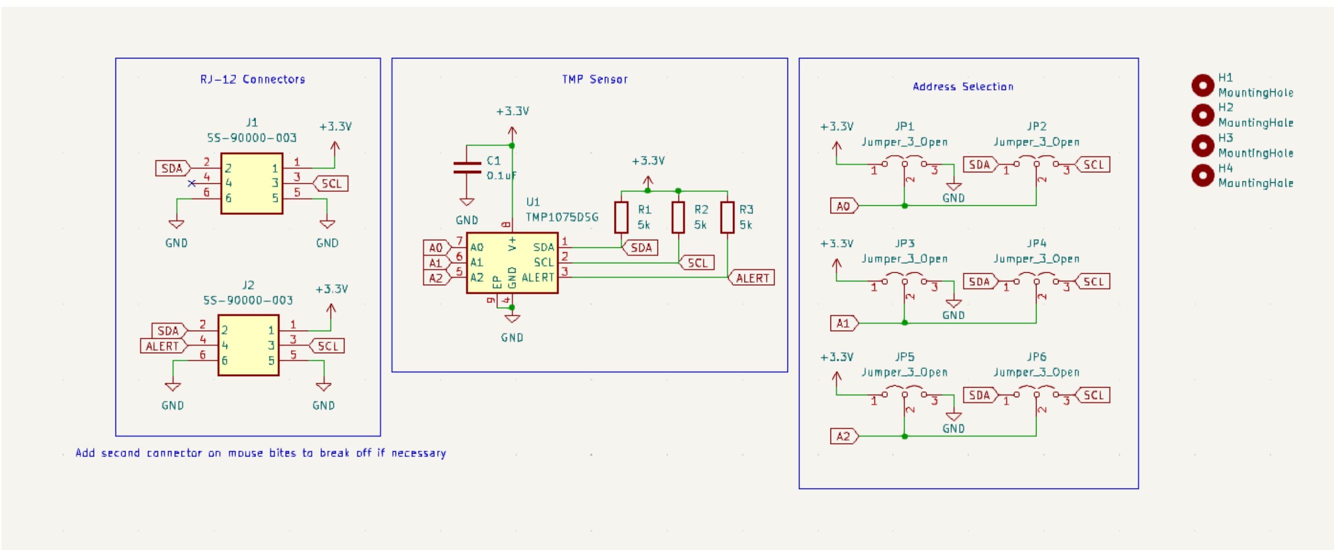


Fig C.1.7: TMP Breakout Board Schematic

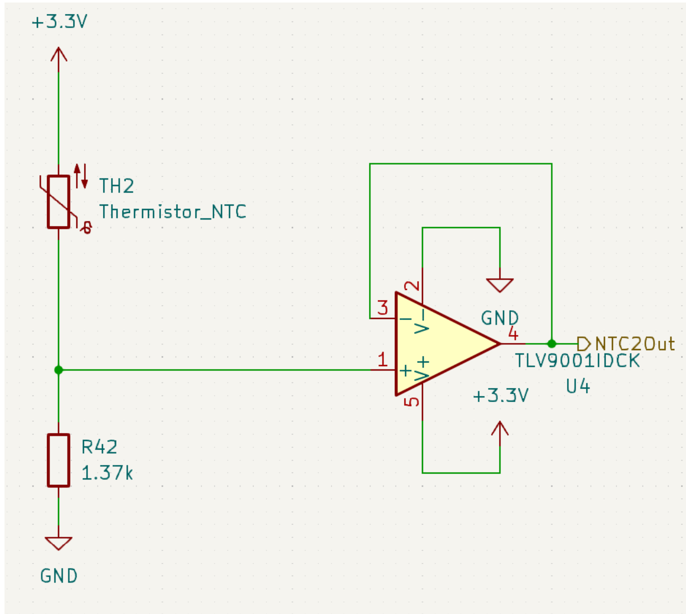


Fig C.1.8: Post-Bodge Thermistor Tuning Schematic

Appendix C.2 PCB Pictures/Screenshots

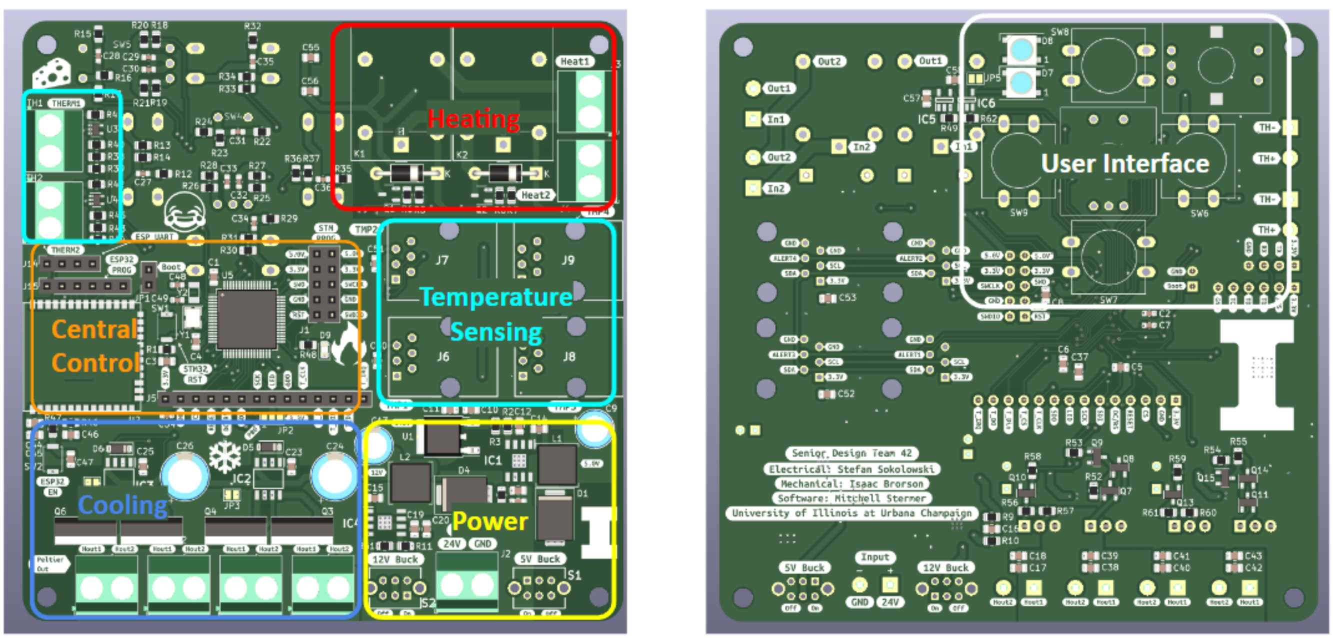


Fig C.2.1: Final PCB Render with Labelled Subsystems

Fig C.2.2: Enable Pin Bodge Wiring

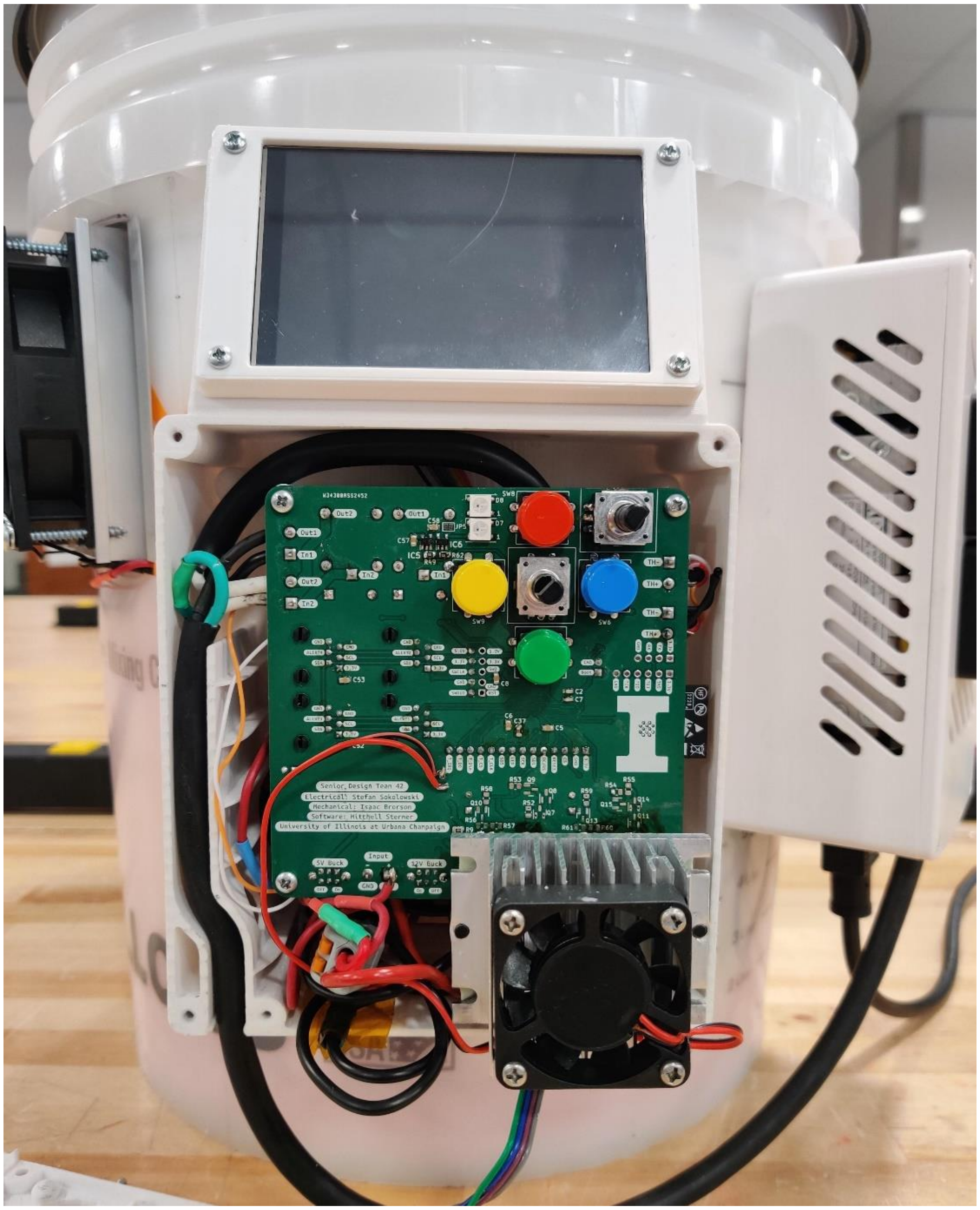


Figure C.2.3: Assembled PCB with Heatsink in Enclosure

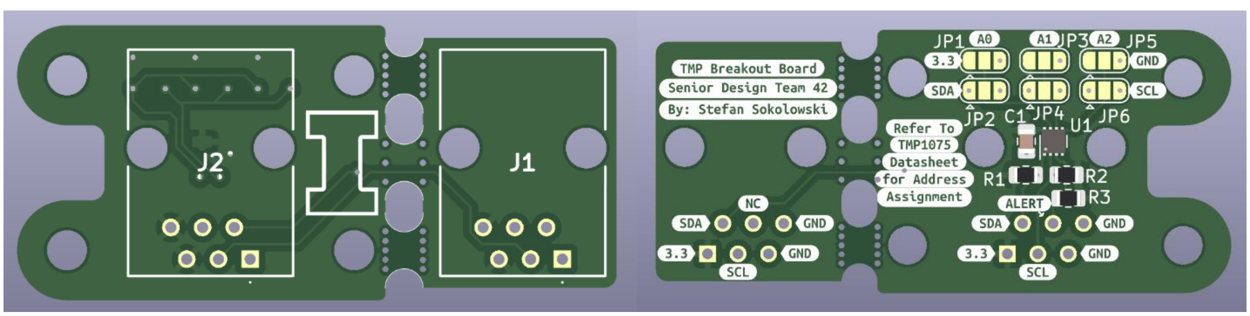


Figure C.2.4: PCB Render of TMP1075 Breakout Board

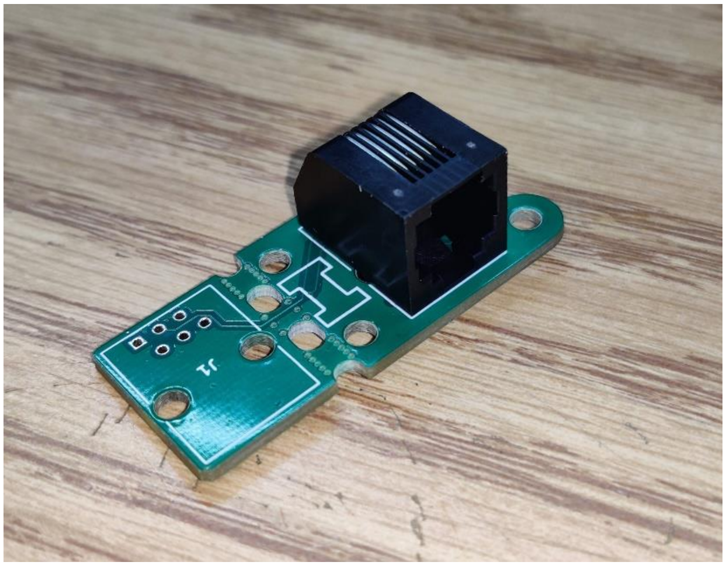


Figure C.2.5: Final Assembled TMP1075 Breakout Board

Appendix C.3 Electrical Cost Analysis References

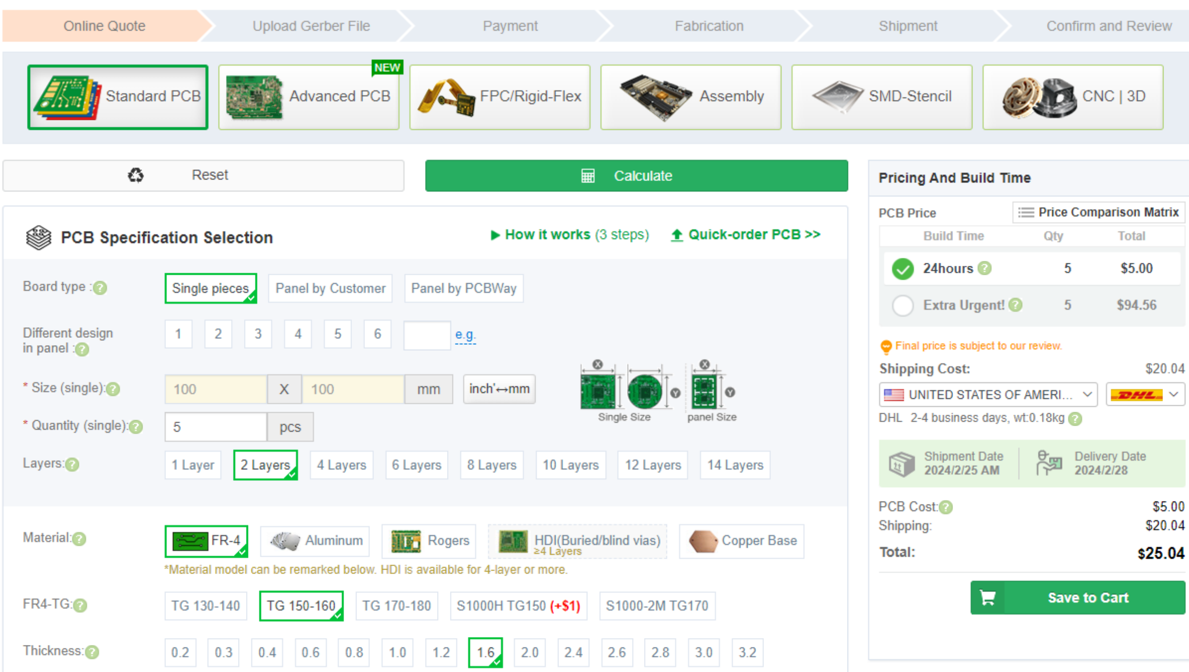


Figure C.3.1: Total PCB cost including shipping

Table C.3.1 : Components Used from the Service Shop

Part	Designators	Service Shop Part #	Quantity per Board	# of Boards	Total Quantity

1k 0805 Resistor	R1, R5, R7, R38, R39, R43, R44	RMCF0805JT1K00	7	5	35
10k 0805 Resistors	R6, R8, R18, R20, R19, R21, R25, R27, R26, R28	RMCF0805JG10K0	10	5	50
Screw Terminals	J2, J3, J4, J10, J11, J12, J13, TH1, TH2	1715721	9	5	45
10uF Capacitors	C17,C18,C38,C39,C40,C41,C42,C43, C19, C20, C21, C22	GRM21BR61H106ME43L	12	5	60
1.5k 0805 Resistors	R40, R45	RMCF0805FT1K50	2	5	10
330 Ohm 0805 Resistors	R17, R24, R14, R31, R34, R37	RK73H2ATTD3300F	6	5	30
0.01uF 0805 Capacitors	C29, C30, C32, C33, C14, C15	C0603C103K4RAC7867	6	5	30
0.1uF 0805 Capacitors	C28, C31, C27, C34, C35, C36, C1, C4, C5, C6, C8, C3	CL21F104ZAANNNC	12	5	60
3.3V LDO	U1	AZ1117CD-3.3TRG1	1	5	5
100k 0805 Resistors	R2, R4, R9, R11	RMCF0805JT100K	4	5	20
1uF 0805 Capacitors	C10, C11	CL21B105KBFNNNG	2	5	10
STM32F401	IC1	STM32F401RBT6	1	5	5
10pF Capacitor	C2, C7	0603N100J500CT	2	5	10

Table C.3.2: Components Bought from Mouser

	Mouser #	Mfr. #	Manufacturer	Order Qty.	Price (USD)	Ext.: (USD)	
1	755-RB550ASA30FHT2RB	RB550ASA-30FHT2RB	ROHM Semiconductor	20	$0.319	$6.38	
2	835-RT6285GSP	RT6285GSP	Richtek	10	$0.761	$7.61	
3	71-CRCW0805220KFKEAC	CRCW0805220KFKEAC	Vishay	100	$0.014	$1.40	
4	603-AC0805FR-07100RL	AC0805FR-07100RL	YAGEO	100	$0.011	$1.10	
5	652-PEC12R4222FS0024	PEC12R-4222F-S0024	Bourns	4	$1.61	$6.44	

6	530-SS-90000-003	SS-90000-003	Bel	20	$0.383	$7.66	
7	595-TLV9001IDCKR	TLV9001IDCKR	Texas Instruments	10	$0.348	$3.48	
8	637-1N4002	1N4002	Diotec Semiconductor	20	$0.071	$1.42	
						Merchandise:	$35.49

Appendix C.4 Electrical Verification

23.73V*4.28A=101.6W

12.3V*.46A=5.67W

3.17V*.28A=0.89W

4.15V*1.97A = 8.176W

Figure C.4.1: Power Subsystem Verification on Programmable Load

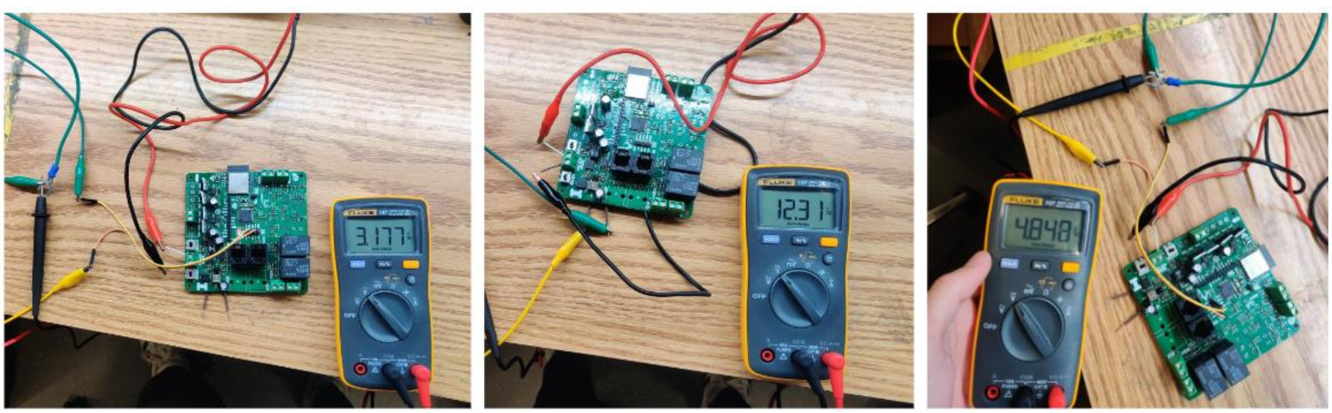


Figure C.4.2: Voltage at the Source During Load Testing

Figure C.4.3: PWM Output from Half Bridge During Testing

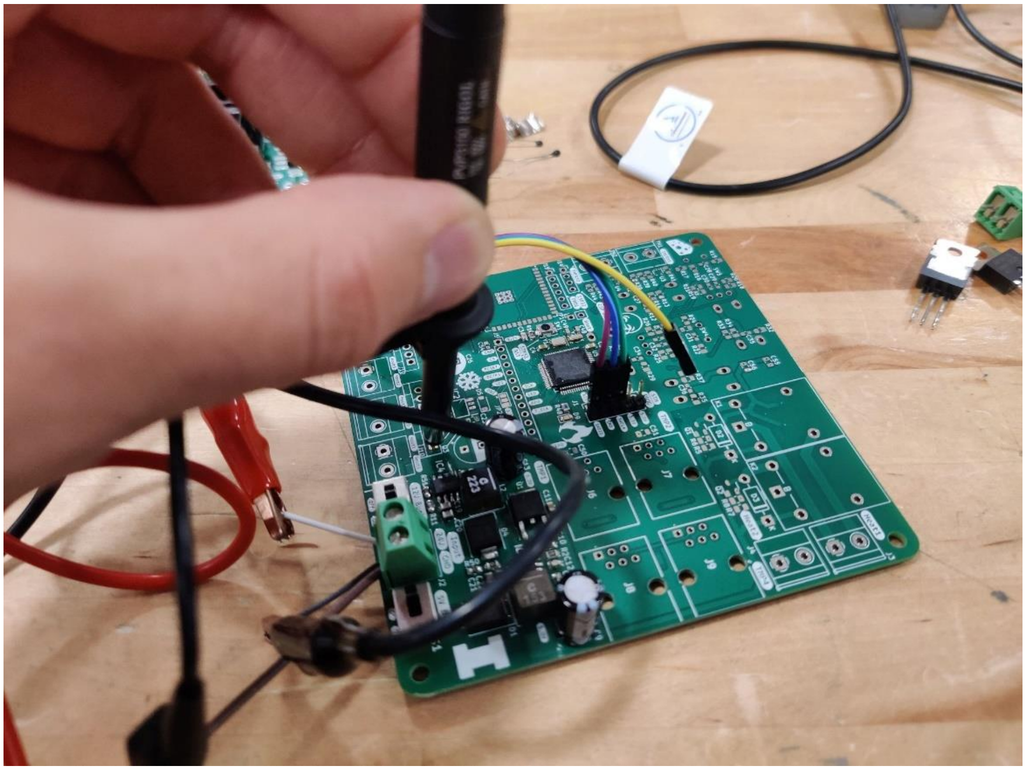
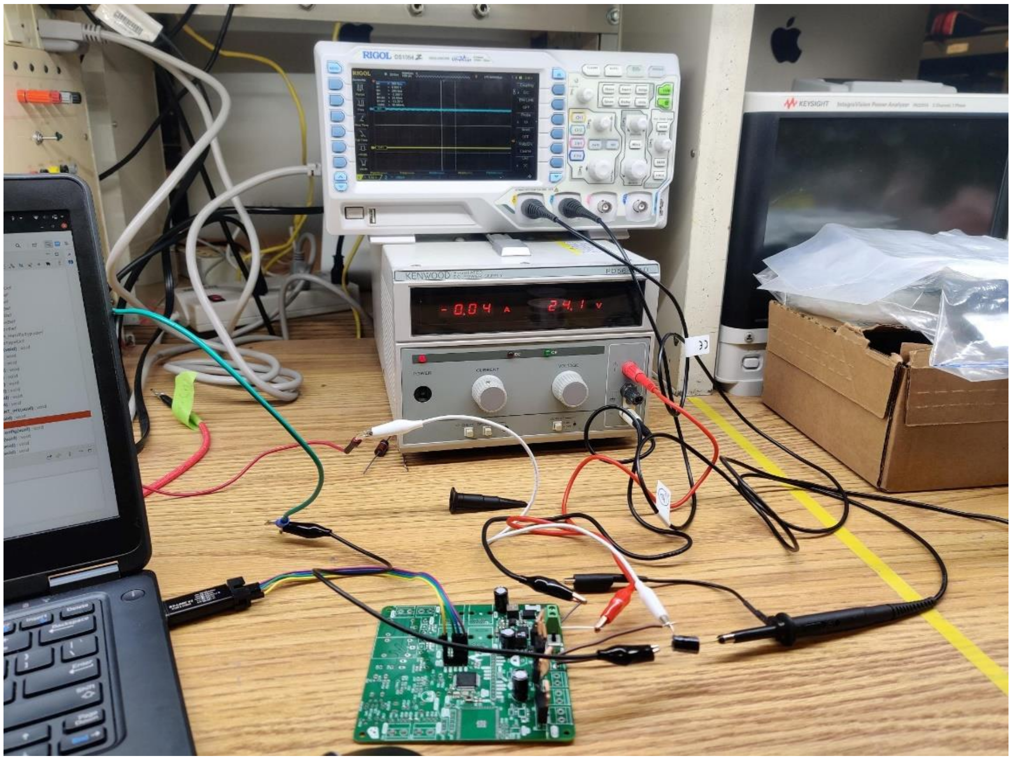


Figure C.4.4: Test Setup for Half Bridge Testing

Figure C.4.5: Output of Voltage Levels



Appendix D Software References

Appendix D.1 Equations used to Construct Graph of Heating Curve

All these equations are well confirmed and are used to draw the red target temperature curve representing the user entered data on the LCD display.

$$ppd_C = \frac{319 - v_{\text{offset}} - y_{\text{start}}}{45 - (-15)} = \frac{y_{\text{end}} - y_{\text{start}}}{60}$$

Equation D.1.1: Vertical Distance in Pixels Representing One Degree Celsius

Where 319 is the maximum pixel height corresponding to the bottom row of pixels on the LCD, v_{offset} is the vertical offset from the bottom row, y_{start} is maximum pixel height on the graph, y_{end} is simply the difference between 319 and v_{offset} , and 45 and -15 are the maximum and minimum temperatures displayed on the graph in °C.

$$ppd_F = \frac{319 - v_{\text{offset}} - y_{\text{start}}}{120 - 0} = \frac{y_{\text{end}} - y_{\text{start}}}{120}$$

Equation D.1.2: Vertical Distance in Pixels Representing One Degree Fahrenheit

Where 319 is the maximum pixel height corresponding to the bottom row of pixels on the LCD, v_{offset} is the vertical offset from the bottom row, y_{start} is maximum pixel height on the graph, y_{end} is simply the difference between 319 and v_{offset} , and 120 and 0 are the maximum and minimum temperatures displayed on the graph in °F.

$$ppm = \frac{x_{\text{end}} - h_{\text{offset}}}{\text{VALID\_TIMES[n-1]}}$$

Equation D.1.3: Horizontal Distance in Pixels Representing One Minute

Where x_{end} is the maximum x-pixel, or the right bound of the graph, h_{offset} is the minimum x-pixel, or left bound of the graph, and VALID_TIMES[n-1] is the last time entry in the time data array (i.e. the total time of the heating curve).

For each entry i such that $0 \le i \le n$, where n is the amount of data point entries:

$$\text{For all i such that } 0 < i < n,$$

$$i_x = \text{VALID\_TIMES[i]} * ppm + h_{\text{offset}}$$
$$i_y = 319 - v_{\text{offset}} + \text{VALID\_TEMPERATURES[i]} * ppd = y_{\text{end}} + \text{VALID\_TEMPERATURES[i]} * ppd$$

Equation D.1.4: Pixel Coordinate Transformation of each Data Point

Where i_x is the x coordinate, offset from x=0 the leftmost column of the screen, and i_y is the y coordinate, offset vertically from y=319, the lowest row of the screen.

Appendix D.2 Heating Curve Error Data

Table D.2.1: Food Temperature Error

Time (m)	Target T (C)	Food T (C)	Chamber T (C)	Error ΔT (C)
0	28	28	26	1.2
1	29.2	28	26	3.4
2	30.4	27	25	3.6
3	31.6	28	25	2.8
4	32.8	30	25	2
5	34	32	26	1.2
6	35.2	34	25	0.4
7	36.4	36	26	-0.4
8	37.6	38	25	-0.2
9	38.8	39	25	3
10	40	37	24	3
11	40	37	26	2
12	40	38	25	0
13	40	40	27	-1.057
14	38.943	40	28	-2.098
15	37.902	40	27	-2.139
16	36.861	39	26	-2.18
17	35.82	38	25	-1.221
18	34.779	36	24	-0.262

19	33.738	34	24	-0.303
20	32.697	33	24	0.656
21	31.656	31	23	0.615
22	30.615	30	22	0.574
23	29.574	29	21	-0.467
24	28.533	29	20	-0.508
25	27.492	28	20	-0.549
26	26.451	27	19	0.41
27	25.41	25	18	0.369
28	24.369	24	18	0.328
29	23.328	23	17	0.287
30	22.287	22	17	0.246
31	21.246	21	16	1.205
32	20.205	19	16	1.164
33	19.164	18	16	18.123
34	18.123			
35	17.082			
36	16.041			
37	15			
38	15			
39	15			
40	15			

Appendix E Revised Schedule

Table E.1: Project Schedule

Week of 2/19	• Design Review with Instructor and TAs (Everyone) • Review order with TA and submit order request (Stefan) • Finish designing PCB (Stefan) • Begin learning STMCUBEIDE (Mitchell) • Begin 3D CAD for mechanical design (Isaac) • Visit machine shop to submit design (Isaac) • Order parts (Isaac and Stefan)
Week of 2/26	Software: (Mitchell) • Begin designing UI and LCD with rotary encoder and buttons Electrical: (Stefan) • Finish Schematics • Get BOM for electrical components that will be on board • Begin to order components Mechanical: (Isaac) • Complete mechanical CAD • Make sure all mechanical parts are ordered • Submit design to the machine shop for the parts we need machined
Week of 3/4	Software: (Mitchell) • Continue learning STMCUBEIDE • Continue designing UI and LCD with rotary encoder and buttons Electrical: (Stefan) • Design review of boards • Order boards Mechanical: (Isaac) Begin mechanical construction: • First make sure Peltier coolers can be mounted to the inner chamber • Figure out how to attach the heating band to the inner chamber

	• Decide on a mechanism for attaching the lid
Week of 3/11	Software: (Mitchell) • Continue developing UI and LCD system • Begin developing Peltier element logic and start integrating software with PCBs Electrical: (Stefan) • Waiting for boards • Help out with software and GPIO bringup for STM32 Mechanical: (Isaac) • Finish all drill holes / cutouts in the clamber walls that are needed for Peltier coolers, wires, standoffs, etc… • Put the chamber together • Build the lid
Week of 3/18	• Integrate PCB, chamber, and LCD together (Everyone) • Boards should arrive, begin assembly of boards and spot testing of subsystems (Stefan and Mitchell) • Attach Ebox to the side of the device (Isaac) • Connect all wires that need to run through the walls of the chamber (Isaac) • Mount LCD & user interface (Stefan and Mitchell)
Week of 3/25	• Same tasks as last week and additionally: • Begin running thorough tests to ensure system runs properly (Everyone) • Verify modification of subsystems and mechanical components (Everyone) • If time, make changes to the board and order a new one (Stefan)
Week of 4/1	• Same as last week and additionally: • Run through practice demos for next week (Everyone)
Week of 4/8	• Run through practice demos for next week

Week of 4/15	• Mock demo with TA
Week of 4/22	• Resolder MOSFET system for Cooling and Assemble Backup PCBs (Stefan) • Finalize the Mounting of the Peltier Coolers (Isaac) • Continue Running Tests for PID Loop, specifically cooling (Mitchell)
Week of 4/29	• Final Presentation