It Should Just Work – Thermo-Stasis

Diana Aleksieva, Eric Choi, Thu Nguyen, Margaret Tran, and Victor Xia

December 13th, 2022

Capstone Design ECE 4440 / ECE4991

Signatures

Statement of work:

Diana Aleksieva:

My primary task was to design the power elements and to design the components that deal with the heat flow within the system. The first part involved finding a power supply that meets the requirements of our project, these included having a high enough voltage supply to run all the components and to allow a battery life of at least 2 hours. This resulted in a 12V lead acid rechargeable 20Ah battery to be chosen. In addition to having the power supply I helped create the protective circuit that included a rocker switch (power button) and a 10A fuse to protect the rest of the components. Since only the fans and Peltier modules needed 12V input I designed linear regulators that input 3.3V and 5V. The linear regulators were chosen based on the input and output voltage requirements as well as the current needed for each component that uses the 3.3 and 5 voltage power supplies. All this helped give power to all the components and to have the project be portable. The second part of my primary task involved me doing heatsink calculations, that resulted in thermal resistance values for an internal and external heatsink. These calculations were based on thermal calculations Eric did regarding how much energy needed to be moved from the system. To be able to reach these values fans were needed for both the internal and external heatsinks. The heatsinks and fans helped move the heat generated within the system and the Peltier devices can't be used at high current without heatsinks. Along with choosing the fans, I designed the circuit connection between the fans and the MSP. The internal fan is important for decreasing the thermal resistance of the internal heatsink and to circulate the air within the container, but when the lid is open having the fan on would push the cold/hot air out. To fix this problem I oversaw finding a lid button and creating the circuit for the connection between the button to MSP. Along with my primary responsibilities there were several other tasks that I was involved in. One task was soldering the PCB and doing the electrical testing, these were done alongside Eric. Another task included helping with the wood cutting, particularly cutting the handle and sanding some of the wood. Lastly, I helped with the assembly of the device alongside my group, this included cutting Styrofoam, helping screw the wood together, and crimping wire. I also helped put together the heatsinks/fans and Peltier devices together (twice) with Eric.

Eric Choi:

My primary task was to calculate and design the system that performs the heating and cooling for this device. This involved the use of thermodynamics equations to determine the required number of Peltier devices and how much power would be required to operate them for given thermal requirements. These calculations resulted in a minimum required thermal power of about 61.58W to meet the requirements in one hour, rather than two hours to introduce leniency, and included the target fluid (0.5 Liters of water), remaining air, and the maximum estimated heat loss. This thermal power specification was evaluated with various available Peltier modules to determine the design with four TEC1-12706 Peltier Modules, each operating with an input

current of 3 Amperes. These thermal calculations were necessary to be completed early in the project life cycle since it was the component that performed the primary goal of the product: heating and cooling and other subsystems, like the thermal dissipation and distribution system involving heatsinks and fans.

Another major task that I completed was the design of the circuit that performs the switching and power modulation for the Peltier modules. This design included the use of two BTN8962TA half bridges to create an H-bridge that can operate at high load currents. The circuit was designed with the intention of driving the Peltier devices, so the PWM signal needed to be filtered through an inductor to retain some efficiency. These inductor calculations were performed with a switching frequency that aligns with the specifications of the selected half bridges, minimizes the ripple current, and operates at the maximum load current. An 820 uH inductor was selected for filtering the Peltier power input since it produced the lowest ripple current of the available inductors while also having a saturation current above maximum expected current. A larger inductor would have been preferred, but higher inductance values with high saturation currents were not available within the budget of the project.

I was also the designer of the PCB boards that the project utilized. While each group member created their own respective schematics within KiCAD, I was responsible for importing and creating the correct footprints for each component and then placing each of them on the PCB design. I primarily focused on placing components in a way that corresponded to the schematic such that each system was organized together, but the systems that interact with external sources were close to the edge of the board for easy access. The first board that was sent out ended up working as intended, which helped streamline the remaining tasks. When the PCB board arrived, Diana and I manually soldered all components on the board, testing the systems along the way. This PCB board was important for the whole system since it connected the components in an organized manner, allowing the use of small surface mounted parts and making debugging easier.

I was actively involved with the assembly of the device and structuring of physical components. This involved the cutting and routing of wood that was used for the outer structure of the project. It also includes the cutting of Styrofoam that was used for insulation. I also cut, stripped, and crimped many of the wires that were used to make the connections between the main PCB and the external components. I was also a large contributor to the assembly of the final box, putting together the Peltier and Heatsink system with Diana (twice), screwing together the box alongside many group members, and applying the Silicone sealant for better insulation.

I was modestly involved in the development of the code as I developed the relevant code for the systems that I was responsible for, such as the code for driving the H-bridge, and later made improvements to existing systems. This included the configuration of all PWM pins (both the H bridge and the Fans) to the respective Timers. The most notable code improvement was fixing Clock configurations for the code such that the Clock frequencies were known and consistent such that the DCO Clock was running at 25 MHz. This improvement made many of the other systems more responsive, such as the temperature sensor readings occurring at a much faster rate and the Fuzzy control algorithm being called at a consistent 1 second interval rather

than roughly every 15-20 seconds. I also implemented the Watchdog Timer to operate with an expiration timer of about 2.68 seconds in case the code got stuck from unexpected behavior.

Lastly, I performed a lot of debugging for many of the systems and developed important fixes. During testing, the LCD screen was primarily powered through the MSP, but the LCD did not operate as expected when the circuit was tested. I quickly realized that the problem was occurring because the voltage at the contrast input of the LCD was not being set correctly. Even when adjusting the potentiometer, this was not fixed, but it displayed an important bit of behavior in which the 5V supply dropped when the potentiometer resistance was too low. This indicated that the current through the potentiometer was too high and led to the realization that the linear regulator had too low of a source impedance to act as the intended voltage divider. The potentiometer was then replaced with two resistors that set the contrast voltage to a known voltage that allows visibility. During the last few days before the Capstone demo session, one of the Peltier devices suddenly started to behave as an open circuit. It was too late to purchase a replacement Peltier device, so I decided to open the Peltier device. By connecting the device to a Virtual Bench with a 12V output, I attempted to find the row of semiconductors that caused the open circuit. By iteratively short-circuiting subsequent rows of semiconductors, I discovered the row with a break. Since the Peltier was otherwise already broken, I decided to solder a wire to create a permanent short circuit the bypassed the row with the broken semiconductor so we could salvage some performance, which successfully allowed the project to resume with minimal performance loss.

Thu Nguyen:

My main responsibilities were to select a microcontroller, implementing input/output abstractions, as well as some firmware code. In the initial state, I gathered the requirements for the components that interact with the microcontroller to select a suitable MCU. Initially, I chose a baseline MSP430 microcontroller, but it was proven to be insufficient for the project mainly due to the lack of GPIO pins. Hence, I created a pins mapping that listed all the inputs and outputs from relevant components, such as the buttons or temperature sensors, how many pins they need, and if there are any special requirements. From there, I was able to select a more suitable microcontroller. During the PCB design process, I communicated closely with Eric to ensure that the ports and pins of the components that interact with the microcontroller are accurate. I also communicated with Diana on the power supply requirements for the microcontroller and the LCD screen. I worked with Maggie closely to implement the LCD screen code, from defining the ports to implementing necessary commands for displaying the data we need. I also set up the timers for the LCD screen and the task scheduler. Input and output abstraction was a crucial part of our codebase since it provided more organization and better accessibility to the necessary inputs and outputs of the components. Hence, I defined the structs needed for inputs and outputs, which included the buttons' statuses, temperature sensors' readings for the inputs, and the output PWM rates for the fans and h-bridge. I worked with Victor on ensuring the tasks in the task scheduler are running at regular intervals. To ensure the microcontroller can run up to its maximum frequency of 25MHz, I wrote the C code to set up the DCO clock to run at the specified frequency, which was later utilized by Eric. For the

temperature sensors, I worked on the C code for the I2C communications and reached out to professors and graduate students, such as Prof. Delong, Prof. Barnes, and Chase Carson, when facing I2C difficulties. I also worked with Victor on changing the bit resolution from 9-bit to 12-bit resolution for the temperature sensors. During the testing process, I utilized the buttons on the microcontroller to display more inputs and outputs of the system on the LCD screen for debugging purposes. Additionally, I adjusted some macros for the fans PWM to better match the need of the device. I also ensured that the C code is well documented and up to standards.

Margaret Tran:

My main responsibilities were finding suitable pushbuttons, temperature sensors, and an LCD screen, selecting what materials would be used for the device, and creating the physical/mechanical CAD design for the box and the heatsink assembly for the Peltier modules. To get an initial design for the box and some preliminary design parameters, I performed calculations based on initial box dimensions and the wood casing and Styrofoam materials to calculate estimates of the amount of heat that would be lost through these materials, which gave us a starting point for the amount of heat loss our system would need to be able to counteract. I had a great learning curve to overcome in learning how to use Autodesk Fusion 360 to create a CAD model of the entire device, which went through several iterations of re-design. I also determined how to design and fabricate the heatsink assembly to prevent damage to the Peltier modules. This involved researching and finding suitable Belleville washers to handle the load and modeling it in CAD. With some advice from Professor Powell, I also determined how to physically incorporate the temperature sensor chips into the project by determining how to design the individual breakout boards on which the sensors are mounted, including use of the thermally conductive pads, thermal vias, and resistor networks to set a unique address for each sensor. I also designed the increment and decrement button protection circuit schematic as well as the temperature sensor circuit schematic with connections to the microcontroller. On the software side, I worked closely with Thu in implementing the LCD screen code. For fabrication of the device, I coordinated with the folks at the School of Architecture's Fabrication Lab and the Mechanical and Aerospace Engineering's Machine Center to assist us in our woodworking and metalworking. I also helped with machining the wood and assembling the box.

Victor Xia:

I was responsible for researching and implementing a suitable control algorithm, namely fuzzy logic control, for Peltier module temperature regulation. This involved defining the input membership functions, the specific ruleset for processing the membership functions, and picking a method for collapsing the fuzzy inputs into a concrete output. The fuzzy control algorithm I developed was also in charge of determining the operational mode of the device, and the threshold by which the operational mode changes. I was also in charge of developing an algorithm for controlling the RPM of the two heatsink fans. I worked with Thu on defining the system structure of the C codebase and implemented a task scheduler for running processes at regular intervals. I defined the structs needed for the fuzzy control, temperature sensors, fan

control, and button C files, and helped define many of the macros that would later be adjusted for efficient system parameter tuning. I also converted the 3D CAD of the project modeled by Maggie into a manufacture-ready format, and generated drawings with machining instructions and dimensions for the plywood components and heatsinks. I helped cut out most of the wood pieces that make up the outside of the box in collaboration with the rest of the team. I debugged the temperature sensor code to change the bit resolution from 9-bit to 12-bit resolution, and fixed integer overflow errors inside of the temperature conversion function and fuzzy output function. Finally, I oversaw debugging and tuning the fan and fuzzy logic control algorithms, to reduce steady state oscillations and ensure the system is configured optimally for the set temperature conditions.

Table of Contents

Table of Figures

Abstract

Thermo-Stasis is a portable, temperature-regulated compartment that is capable of heating and cooling objects. The user can choose to increase or decrease the temperature by interacting with buttons, as well as view the actual temperature in the chamber and the desired set temperature via a Liquid Crystal Display (LCD) screen. The MSP430 microcontroller communicates the user's input from the buttons to the Peltier-based heating and cooling element, based on the result of the fuzzy control algorithm. The device has three temperature sensors to keep track of the current temperature inside the compartment and the temperatures of the two heatsinks (internal and external). From there, the Peltier modules will generate electrical heating or cooling based on the user's input and the current temperature in the compartment.

Background

According to the CDC, an estimated 1 in 6 Americans become ill from foodborne diseases annually [1]. Foodborne illness can result from consuming food that has been improperly stored and/or chilled. Hence, the project is inspired by the need to better alleviate foodborne illnesses, especially on-the-go. Thermo-Stasis addresses the ongoing issues of improper food storage by providing a portable device in which the user can store items at a stable desired temperature for a prolonged period. An additional benefit of Thermo-Stasis is that it provides both cooling and heating options, giving the user the flexibility to use it for a variety of applications.

In 2015, a similar project was created by Joseph Rautenbach in which Peltier modules were utilized to produce a temperature-controlled miniature refrigerator [2]. The main difference between Rautenbach's project and this project is that Thermo-Stasis can perform both heating and cooling on the object in the compartment, whereas Rautenbach's project can only perform cooling. Additionally, the idea of a Peltier module for refrigeration and heating has been proposed but has not yet been developed [3].

There are already active coolers and active heaters on the market, but systems that can easily do both are not common. Even the existing portable coolers are notably more expensive. Our project is unique in that it will merge the two systems to be able to perform both heating and cooling without the use of infrared heat, ice, coolant, or other similar means, while also being created on a budget.

This project draws on coursework from the Electrical and Computer Engineering (ECE) core classes. Firstly, the Electrical Engineering Fundamentals Series: ECE 2630, 2660, 3750. The electrical engineering skills developed in these classes were used for designing the power systems that are required for using the electronic components without causing safety issues or risking damage to the parts. Many of the components have varying requirements for supplying power, so the system was designed using the relevant systems and carefully chosen components so that it is able to draw enough power to work, but not so much that it is inefficient or damaging. Secondly, the Embedded Systems Curriculum: ECE 3501, 3502 - Embedded

Computing and Robotics. The microcontroller for this project takes in temperature data from the temperature sensors and then interfaces with the Peltier modules to adjust the temperature. The microcontroller also has hardware interfaces, like buttons and a screen, so that the temperature can be monitored and set by the user. Thirdly, the Computer Science (CS) Curriculum: CS 2150 - Program and Data Representation, CS 3240 - Advanced Software Development. The code development for this project used a lot of skills developed from these CS classes. The actual code used acquired knowledge of data structures and algorithms, object-oriented programming and abstraction, as well as general low-level programming, while the code development process was based around an Agile software development methodology.

Physical Constraints

Design Constraints

The project has two main constraints as required by the Capstone course. First, the project must utilize a custom Printed Circuit Board (PCB) designed by the team. Second, the project must include a microcontroller.

CPU Limitations:

The microcontroller selected for this project was the Texas Instruments MSP430F5529 [4] because of the large number of GPIO pins it has, as well as the necessary peripheral interfaces. This was chosen based on the need for at least 32 GPIO pins, inter-integrated controller (I2C), 4 Pulse Width Modulated (PWM) outputs, and at least 4 timers. Another microcontroller was chosen earlier in the semester, Texas Instruments MSP430G2553 [5], but it was found to not have enough GPIO pins, so it was replaced by the current microcontroller.

Software Availability:

KiCad [6] was used for circuit board layout as well as schematic layout and simulation. For embedded code, Code Composer Studio [7] was used for writing C code as the program is free and compatible with the MSP430 microcontroller. Autodesk Fusion 360 [8] was used to model the prototype, including the outer case design, heatsinks installation, and the rest of the required components. CADLAB [9] and GitHub [10] were used as version control for the KiCad files and C code files, respectively.

Manufacturing Limitations:

A major manufacturing limitation was the availability of parts due to global supply chain issues. There was also the limitation of finding parts that were in stock and available to arrive within the necessary timeframe. Additionally, there were some limitations on the PCB, as it required specialized equipment for its production and time to wait for it to be produced by the supplier. By far, the most significant manufacturing limitation was fabricating the device. Since the Peltier modules are compressed between two heatsinks with an aluminum bar extender in-between, there was extensive metalwork that needed to be done to cut an aluminum bar stock into the appropriate size as accurately and smoothly as possible, and to machine down fins of the

heatsinks so that holes could be drilled. Kemal Gokturk from the Mechanical and Aerospace Engineering Machine Lab thankfully assisted with the fabrication, however, the process was delayed by approximately two weeks due to his availability and the fact that he did not have a necessary part for one of his machines and needed to order it. Additionally, due to the nature of the Peltier modules and their vulnerability to overheating, we could not safely perform testing of this subsystem without first having the entire heatsink and fan assembly constructed. Thus, the metalworking fabrication was a major challenge faced, especially under the time constraint. Similarly, the construction of the wood casing needed to be completed at the Architecture School Fabrication Lab, so we were limited by their times and the availability of machinery there.

Cost Constraints

The main cost constraint for this project was the $500 budget for parts. We expected the power supply and cooling/heating elements to be the major expenses within the project. Specifically, we expected the battery, Peltier modules, heatsinks, and fans to use approximately half of the budget. We planned for the PCB, its components, and the microcontroller to use approximately 1/5 of the budget. The remaining portion of the budget was to be used for purchasing plywood for the box casing, expanded polystyrene (Styrofoam) for the box insulation, an LCD screen, and various miscellaneous items necessary for the construction of the device, such as thermal paste, ribbon cable, alligator clips, connectors, screws, nuts, and washers.

Tools Employed

The major software tools used for this project were Computer-Aided Design (CAD) tools and software for programming and debugging embedded code. These included KiCad [6] for the circuit board layout as well as schematic layout and simulation, Autodesk Fusion 360 [8] for creating the 3-D mechanical design of the device, Code Composer Studio [7] for programming and debugging the C code running on the microcontroller, and CADLAB [9] and GitHub [10] for version control and facilitation of concurrent development amongst team members. The main tools we needed to learn, as a team, were KiCad and CADLAB, as previous classes in the curriculum used Multisim and Ultiboard for schematic layout and circuit board layout. Additionally, some team members needed to learn Autodesk Fusion 360 and general 3-D CAD modeling principles.

Various tools for fabrication of the box were used. These include bandsaws, jigsaws, routers, a drill press, sheet sanders, a metal milling machine, power drills, and screwdrivers. Using these woodworking and metalworking tools was a great learning experience for several team members.

Societal Impact Constraints

Environmental Impact

The Peltier device that is used in this design is known to not produce any hazardous gas and substances. The Peltier module is comprised of semiconductor pellets fabricated from N-type and P-type Bismuth Telluride between two ceramic plates [11]. Bismuth Telluride is classified as hazardous waste, which means that it must be contained and disposed of according to federal Environmental Protection Agency (EPA) recommendations [12]. While disposal of Peltier modules has some environmental costs, the environmental impact of manufacturing them is relatively low on its own, because the elements are produced as byproducts of mining other metals. Bismuth Telluride is prepared by mixing bismuth and tellurium metals. Bismuth is a moderately-priced metal that is a byproduct of mining copper and lead. Additionally, bismuth can be produced through recycling. [13]. Similarly, tellurium is indirectly mined since it is recovered as a byproduct of copper and iron [14].

Sustainability

Having the power source as a rechargeable battery allows the device to have longer operational time if needed without switching out new batteries. However, disposal of rechargeable batteries has environmental costs. According to the EPA, rechargeable lithium-ion batteries have the potential to cause fires during transport or at landfills. Thus, care should be taken to recycle them properly to ameliorate this as well as other negative environmental impacts that are associated with the mining and manufacturing of new batteries such as air and water pollution and greenhouse gas emissions [15].

Health and Safety

Some safety concerns from this device may arise from moisture condensation from the cooling compartment, which poses risks to the electrical components and the users. Great care must be taken to ensure that electrical components, especially the Peltier module, are isolated from moisture via tight fitting heat sinks. Additionally, since this device is battery-powered, there are certain risks that come along with any battery-used device, such as damage to the battery due to surrounding temperatures as well as damage from the battery to the surroundings. When the battery is charging, the heat it releases can damage nearby cells, and lithium-ion failures can lead to combustion reaction [16]. Furthermore, in an environment with high water vaporization or moisture, charging the battery would pose more hazards as the by-products of water vapor and fluorine from the battery may produce hydrofluoric acid, an extremely hazardous chemical. To mitigate these safety concerns, it is crucial to carefully integrate protective measures for any sensitive components to ensure that their conditions are not affected by other factors from the systems. For example, the battery should be promptly removed from the charger once it is fully charged. Additionally, the charging environment should be kept dry and away from flammable objects. Prior to usage, the battery should be carefully inspected to avoid potential battery failures during service.

Ethical, Social, and Economic Concerns

This device does not take any personal data of the user, so there are no ethical issues surrounding data privacy. Since the device is made to have a simple, intuitive user-interface with buttons, it should not limit anyone from using it. Though economically, if this device were to go into production, the price would limit the amount of people who could use this device. Anyone with a low income might not be able to purchase this device since this could be considered a luxury item.

External Considerations

External Standards

Barr Group's *Embedded C Coding Standard* was used to minimize bugs in the firmware that will be used with the microcontroller and provide overall guiding principles for the development of embedded software [17].

Regarding IPC standards, the board for this device can be classified as Class 1. Some of the notable standards for this class of boards include IPC-2221 [18], which specifies a generic standard for circuit board design, and IPC-2615 [18], which provides standards for a board's dimensions and tolerances. Both standards were relevant to the project as they governed the design and fabrication of the circuit board used in the project.

The Inter-Integrated Circuit (I2C) Protocol is a protocol that allows communication between multiple peripheral, digital integrated circuits to communicate with one or more controller chips [19]. It only requires two signals to communicate, the serial clock (SCL) and the serial data (SDA). Pull-up resistors are needed for the SDA, SCL, and ALERT lines. This is to ensure one device would not be able to drive the line high while another tries to pull it low to prevent the potential for damage to the drivers or excessive power dissipation in the system.

Intellectual Property Issues

Our project is likely patentable because even though there exist some US patents with claims similar to ours, they have varying applications and structures. The first patent is US7174720B2, titled "Cooker utilizing a peltier device" [20]. The invention is an appliance that provides both heating and cooling of food by using a Peltier device. The first independent claim is as follows:

1. An appliance comprising:

a temperature regulation system comprising a convex portion and a Peltier device having an active side and a waste side

a cook vessel in contact with the active side, the cook vessel comprising a removable lid and having a concave portion disposed at the bottom portion of the cook vessel matched to the convex portion of the temperature regulation system; and

a waste side distribution system,

wherein the Peltier is located underneath the cook vessel and the active side provides temperature regulation to the cook vessel.

The following relevant dependent claims are similar to the claims of our project relating to its structure:

4. The appliance of claim 3, wherein the waste side distribution system comprises a heat sink.

5. The appliance of claim 4, wherein the waste side distribution system further comprises a fan.

7. The appliance of claim 6, further comprising a temperature sensor connected to the control unit.

This invention has similar claims as ours, in that its main functionality is to heat and cool food, and it also mentions the concern of proper storage of food for food safety as a motivation, which is also a major motivation for our project. However, unlike our device, which has a user interface and buttons to control the internal temperature of the device, the patented invention's user interface involves the user selecting time and/or temperature profiles to operate the device to achieve the desired results. Additionally, the device can heat food "until the food is sufficiently cooked to be edible," which suggests that it has a higher maximum operating temperature than our device, as ours is not intended to cook food; it is only intended to keep food at a warm enough temperature to inhibit bacteria growth.

The second patent is US20060237730A1, titled "Peltier cooler with integrated electronic device(s)" [21]. The invention proposes a configuration to cool electronic devices yielding a Peltier effect. The most relevant independent claim is as follows:

1. Electronic apparatus comprising a cooling member thermally coupled to a cooled member, said cooling member comprising a plurality of semiconductor elements configured to yield a Peltier effect when current is passed therethrough, the semiconductor elements having a non-rectilinear shape.

This invention is less similar to our project compared to the first patent, but it still has the same principle with a different application. The patent describes a specific configuration that has the purpose of cooling electronic devices. The main difference from our project and the previous patent is that this patented invention does not feature a user-interface to adjust the temperature of the device.

The third patent is US7082772B2, titled "Peltier temperature control system for electronic components" [22]. The patent proposes a cooling system for audio equipment that uses a temperature sensor and Peltier module in a feedback control loop to prevent overheating of the audio component. The independent claims detail a temperature control system for an electronic audio component comprising of various components, including a Peltier device, temperature sensor, feedback control circuit, fan and heatsinks. The configuration for this invention is similar to our configuration, but like the second patent, the application is different, as it is intended to

cool electronic components instead of food. Similarly, it does not feature a user-interface to allow the user to customize the set temperature, as our device does.

Detailed Technical Description of Project

Thermo-Stasis is a battery-powered box that serves to regulate the temperature inside the box to any temperature specified by the user.

The typical user experience with the box starts with turning on the rocker switch to supply power from the battery to the other electrical components. The MSP430F5529 [4] loads up in default configuration, with the current temperature displayed ramping up from 0 to the current temperature over a course of 10 seconds and the default set temperature as the current temperature, truncated to the nearest degree. The box immediately starts to regulate the temperature of the box by driving PWM signals to the Peltier modules and the fans attached to the internal and external heatsinks. The set temperature of the box can then be changed by the user, with the push button on the left and right representing a 1°F decrease and increase in temperature respectively. The RPM of the fan attached to the external heatsink is dependent solely on the temperature of the external heatsink, with a higher PWM signal being driven as the temperature increases, regardless of outside ambient temperature conditions. The RPM of the fan attached to the internal heatsink is dependent on the temperature differential between the internal heatsink and the current temperature of the box. During periods of large temperature differential between the current temperature and the set temperature of the box, the internal heatsink temperature will often shoot past the set temperature. As the current temperature approaches the set temperature, the PWM being driven to the Peltier modules will approach a settle point, and the temperature of the internal heatsink and the current temperature of the chamber will achieve equilibrium. The set temperature can be changed at any time using the pushbuttons, and the system will respond accordingly to maintain the updated set temperature. Figure 1 shows a system block diagram of the entire system with all the major subsystems and major components. Figure 2 shows a flow chart of the system's behavior beginning with the user's input to the system through the increment and decrement buttons.

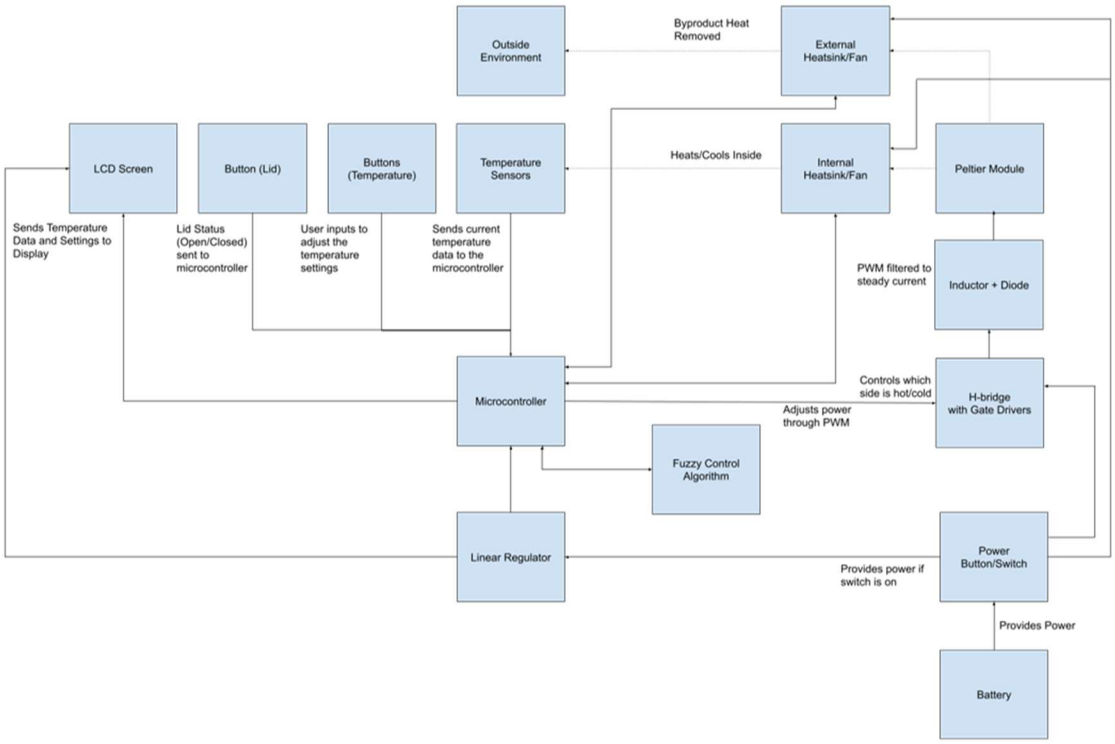


Figure 1: System Block Diagram

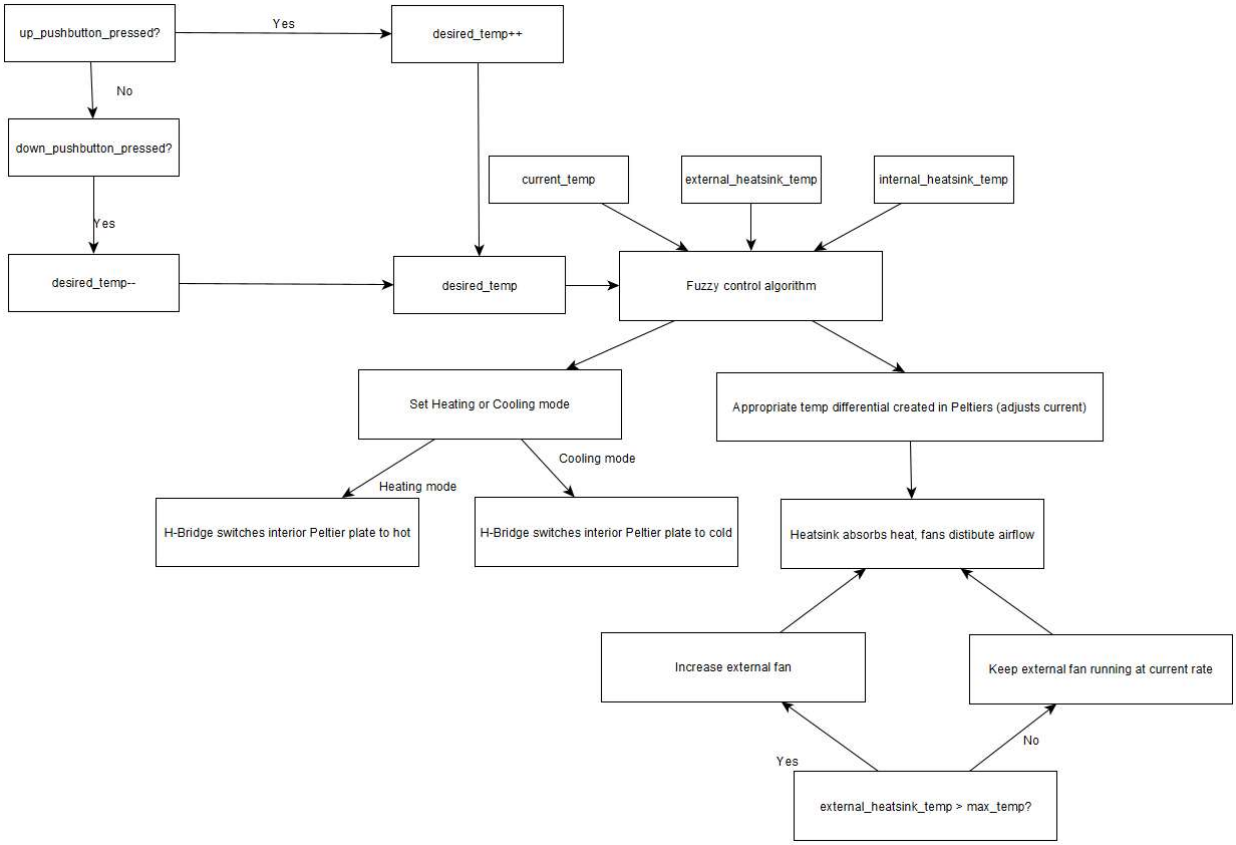


Figure 2: System Flow Chart

Hardware:

Printed Circuit Board (PCB):

The printed circuit board was designed to have one main board and four smaller circuit boards. The main board includes the microcontroller, connectors for external devices, and the circuit components for most systems. The general layout of the board itself was meant to have external connectors as close to the edges of the board to ameliorate the connections. The layout also meant to separate the systems, similarly to how it was set up in the KiCAD [6] schematic, this was such that all the related circuit components were in nearby proximity. Additionally, if subcircuits were duplicated, such as the three buttons, the placement on the PCB were also similarly placed. The PCB design was checked for Design for Manufacturability (DFM) standards using a free DFM service provided by Advanced Circuits [23] to ensure that no showstoppers were present. The PCB design files were then sent to Advanced Circuits and the manufactured PCB without any components or other modifications can be seen in Figure 3. The PCB design in KiCAD can be found in the figures in Appendix A and the PCB with all components soldered can be found in Appendix B. Due to time and budget constraints, all PCB components were soldered manually by two team members rather than outsourced to another company.

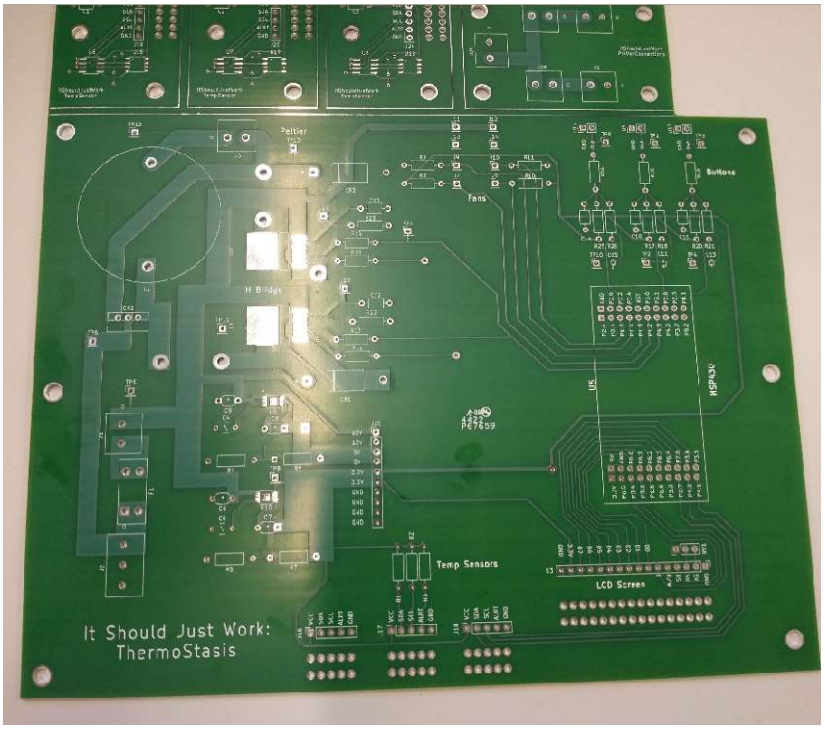


Figure 3: PCB directly after Manufacturing

The high-power external devices, including the 12V 20Ah battery, rocker switch, and Peltier modules, were connected using screw terminal connectors with wire sized at 12 American Wire Gauge (AWG). One of the smaller circuit boards was dedicated for the series and parallel configurations of the Peltier modules. These traces on the main board were also set to have a larger trace width, of at least 3.62 mm wide, to decrease the resistance and ensure that the high current did not cause the PCB to heat up too much. This trace width was calculated using the calculator included with KiCAD [6] with a trace thickness of 1 oz/ft^2, a temperature rise of 10 degrees Celsius, and a current of 6 Amperes. Most relevant trace widths were rounded up to 4 mm or larger for some leniency.

The lower power external devices, including the LCD screen, three buttons, three temperature sensors, and two fans, were connected using ribbon wire connected to 2.54mm header pins. Low power traces connected directly to a power source, like 12 volts, 5 volts, and 3.3 volts, had a trace width of 1 mm or larger. Remaining connections were narrower, typically around 0.5 mm. Low power connections to external connections used ribbon cables with manually crimped ends to be fit in 2.54 mm header connectors. These ribbon wires were then connected to the associated 2.54 mm header pins on the PCB. The remaining three smaller circuit boards were meant for the temperature sensor networks, which included the 0.1 uF bypass capacitors and the resistor networks [24] that set the address for the connected temperature sensors. These sensors required separate PCBs since the temperature sensors needed to be in physically different locations to retrieve the relevant temperatures. Each temperature sensor board also included thermal vias next to the “SDA,” “SCL,” and “ALRT” connections to the temperature sensors to help the temperatures adjust more responsively and accurately. Despite

each sensor requiring separate addresses to be used in the I2C protocol, the addresses on each PCB were all hard wired to ground, or “000,” such that the traces to the ground via could be cut later and the associated address bit could be driven high. One complete temperature sensor PCB with the address bits set to “010” can be found in Figure 4. This strategy for setting the address bits was recommended by Professor Powell.

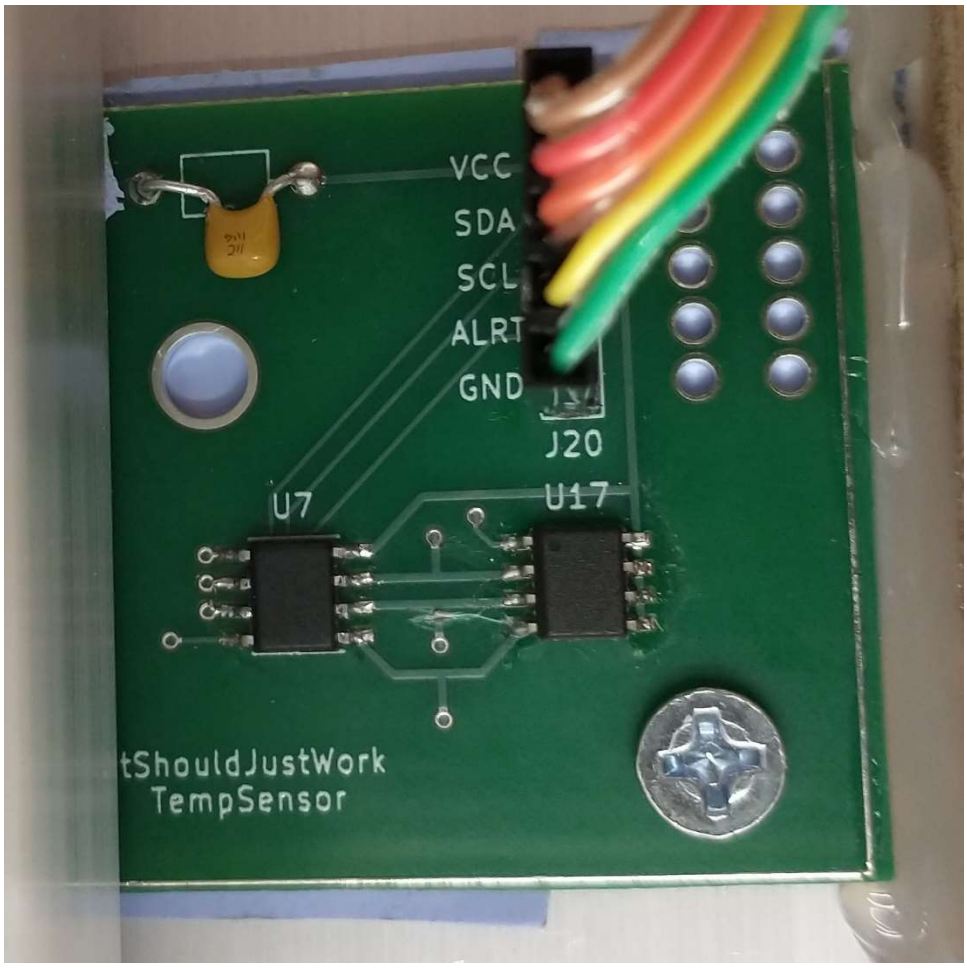


Figure 4: Temperature Sensor with address "010"

All PCB boards included large via holes intended for mounting the boards. These vias had a hole diameter of 4 mm and a via diameter of 5 mm to be used with 6-32 sized screws, which have a thread diameter of 3.505 mm. The temperature sensor boards had two screw holes, the Peltier configuration board had four screw holes, and the central PCB board had six screw holes.

Power Supply:

To power the whole system, a 12V lead acid rechargeable 20Ah battery [25] was chosen. A battery was picked over an ac power supply in order to enable the device to be portable, the battery was also 12 volts since this is what the fans [26] and Peltier devices [27] required. From initial thermal energy calculations there was a worst case total thermal energy requirement of

95.48 Wh. A battery with 20Ah would supply enough power to allow a battery life of at least 2 hours under worst case conditions. A rocker switch [28] and a 10A fuse were used to create a protection circuit between the battery output and the rest of the PCB. The power supply circuit can be seen in Figure 5. The different components within the system required different amounts of voltage and so linear regulators were implemented for the systems that needed less than 12Volts. The fans and Peltier modules ran from 12 Volts while the MSP, temperature sensors, buttons ran on 3.3Volts and the LCD screen runs on 3.3 Volts. The MIC5281YMME linear regulator was picked since the input voltage range allowed for a 12V input and the output voltage ranged from 1.27-5.5V allowing it to be used for both 3.3V and 5V power supply [29]. The resistor values found by the equation: Vout = Vref (R1/R2 +1), where Vref = 1.267, was used to set the output voltage to the desired amount. The current output of this linear regulator was a max of 25mA which allowed for enough current for the components after it, the LCD screen used 1.5mA and the MSP used 5mA. A linear regulator was chosen over switching regulators because the power generated by the linear regulators is low enough for the heat dissipation to not cause any problems. The circuit connection for the linear regulators can be seen in Figure 6.

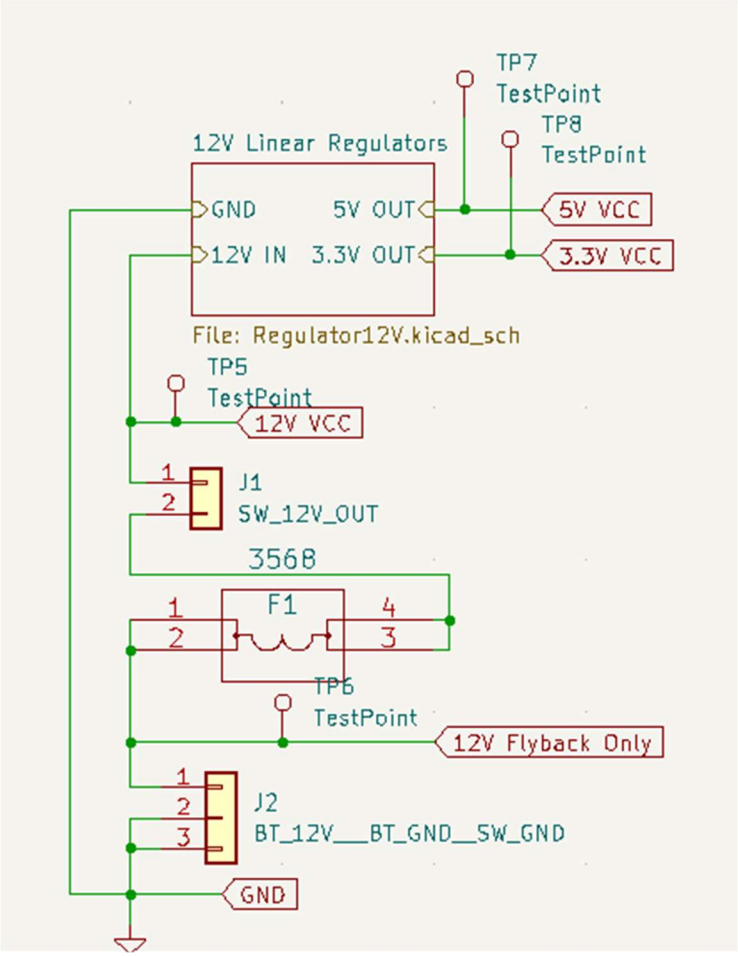


Figure 55: Power Supply Circuit

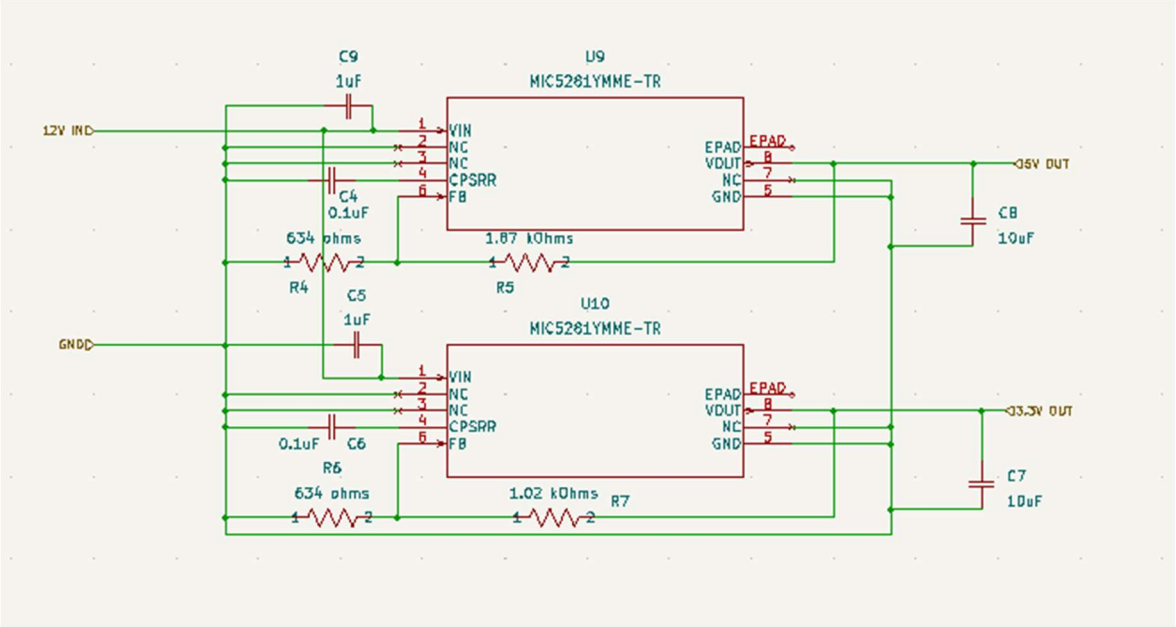


Figure 66: Linear Regulators

Peltier Modules:

The Peltier Modules [27] needed to be selected such that the thermal requirements could be met. Specifically, the Peltier Modules were designed around cooling 0.5 Liters of water from room temperature to refrigeration temperatures within two hours. Meeting this requirement in exactly two hours does not leave much room for error, so it was designed for a time limit of one hour. This was the equivalent of removing about 46.296 British Thermal Units (BTU) or 13.568 Watt Hours (Wh) of energy within the given time limit. The water is not the only thermal mass in the system, so the remaining air in the container along with heat loss was also included. The air from the 1080 square inch container increased the total energy removed from thermal mass to 33.90 Wh and the estimated heat loss with Styrofoam insulation was around 27.68 Watts. This resulted in a worst case total thermal power requirement of 61.58 Watts since the thermal energy needed to be removed in one hour. More detailed calculations can be found in Appendix C.

The thermal energy requirements were not the only design constraints used to select the Peltier components. The thermal energy requirement was used with the heat transfer rate from the cold side of the Peltier (Qc(W)) provided through the figures provided by the Peltier datasheets [27]. Simply choosing the smallest quantity of modules that can meet the bare minimum requirements was avoided, since the maximum cooling output occurs with a low coefficient of performance. The Peltier devices are not perfect and will convert some input electrical energy into heat, but the rate of cooling scales more linearly with current while the added heat scales more with the square of the current, thus causing diminishing returns at high currents. Providing the maximum current through each Peltier will add more heat to the system and would require a larger heatsink to maintain a hot side temperature that works. The heatsink calculations are elaborated more in the Heatsink/Fans section. Operating a larger quantity of Peltier devices at a lower electrical power would result in a more efficient cooling system. The

size of the container (and heatsinks) along with the overall budget for the project ended up also setting a limit on the number of Peltier devices used. These constraints resulted in the selection of four TEC1-12706 Peltier modules with specifications found in the performance curve graphs in Figure 7.

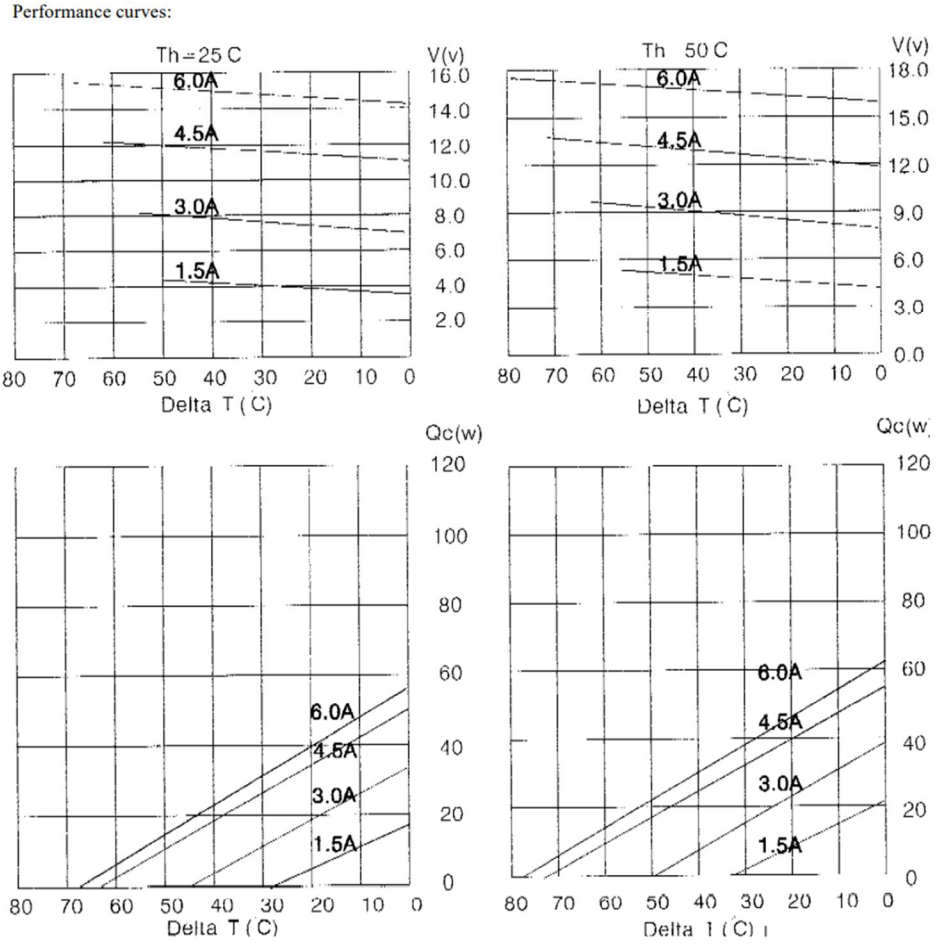


Figure 7: Peltier Datasheet Graphs

As seen by the performance specifications in Figure 8, the selected TEC1-12706 modules [27] are listed as having a resistance of 1.98 ohms, a maximum voltage of 14.4 volts, and a maximum current of 6.4 Amperes. It was not known at the time of design and component selection that the resistance of these devices is variable depending on the temperature differential such that the resistance increases with an increase in the temperature differential. As such, applying a constant voltage will not always produce the same current. A constant resistance around the specified 1.98 ohms was assumed for the design, so a constant 12-Volt power source would be able to draw a maximum of 6 Amperes.

Performance Specifications

Hot Side Temperature (°C)	25°C	50°C
Qmax (Watts)	50	57
Delta Tmax (°C)	66	75
Imax (Amps)	6.4	6.4
Vmax (Volts)	14.4	16.4
Module Resistance (Ohms)	1.98	2.30

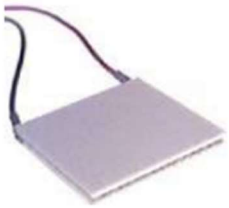


Figure 8: Peltier Performance Specifications (left) and Peltier Device (right)

With the calculated 61.58 Watts of thermal power being split equally between four Peltier modules [27], each module would require a cold side thermal power of 15.39 W. The performance curves from Figure 8 show that this requirement, alongside the desired temperature differential, can be met by providing at least 3 Amperes of current. With the assumed constant resistance, this meant that the Peltier module could be configured as two parallel branches of two Peltier devices in series, as seen in Figure 9. This configuration would draw 3 Amperes through each Peltier module and 6 Amperes in total for this system. This approach was somewhat flawed due to the previously mentioned resistance assumption. As the temperature differential increases, the resistance will increase which would then decrease the current and limit the performance.

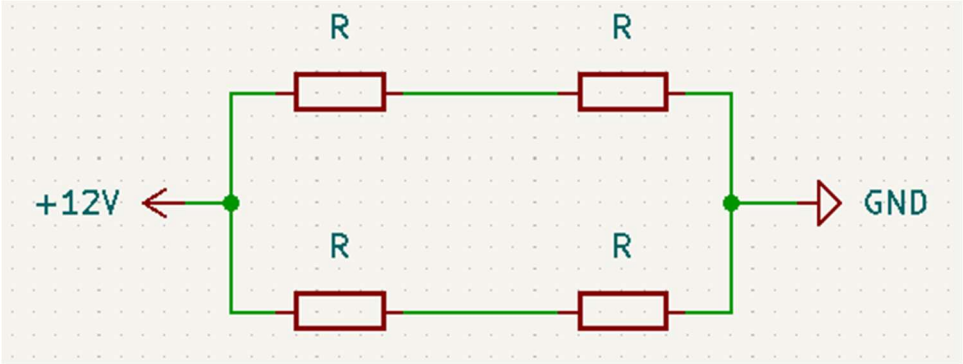


Figure 9: Peltier Configuration Layout

H-bridge:

The H-bridge was used as the switching mechanism to allow the same Peltier modules [27] to perform both heating and cooling. By configuring two BTN8962TAAUMA1 half bridges [30] with the Peltier devices as the load in between, the direction of the current through the load could be reversed by flipping the power source that each half bridge links to, either ground or the 12 volt power supply. The heating and cooling functionality is swapped when the direction of the current is flipped, but the Peltier devices also need to be modulated for a stable temperature. PWM control is an easy way to adjust power and would require one side to be held as a constant connection to ground while the other is switched between a connection to the power supply and a

connection to ground according to the PWM input. The efficiency of Peltier devices varies at different currents, so providing a more stable current will produce less undesired heat than if the same average current were to be provided by a direct PWM signal. To filter the PWM signal to appear like a stable current, an inductor was included in series with the Peltier load. The Schottky diodes connected from ground to the load elements are meant for drawing the inductor driven current from ground when the PWM signal is off. There are two diodes in this configuration since the circuit direction is meant to be switched. This inductor was calculated according to expected values. The maximum current draw was estimated at 6 Amperes, the switching frequency would be around 20 kHz, and a maximum tolerable ripple current was set around 30% from the desired current at a 75% duty cycle. Additionally, while calculating the integral for inductor energy, linear assumptions were used to simplify the calculations and actual simulated values appeared to display better performance. The final inductance value used for the inductor was 820 uH, since this was the largest available inductor that had a saturation current above the 6 Ampere current and fit within the budget [31]. More detailed calculations regarding the inductor can be found in Appendix D.

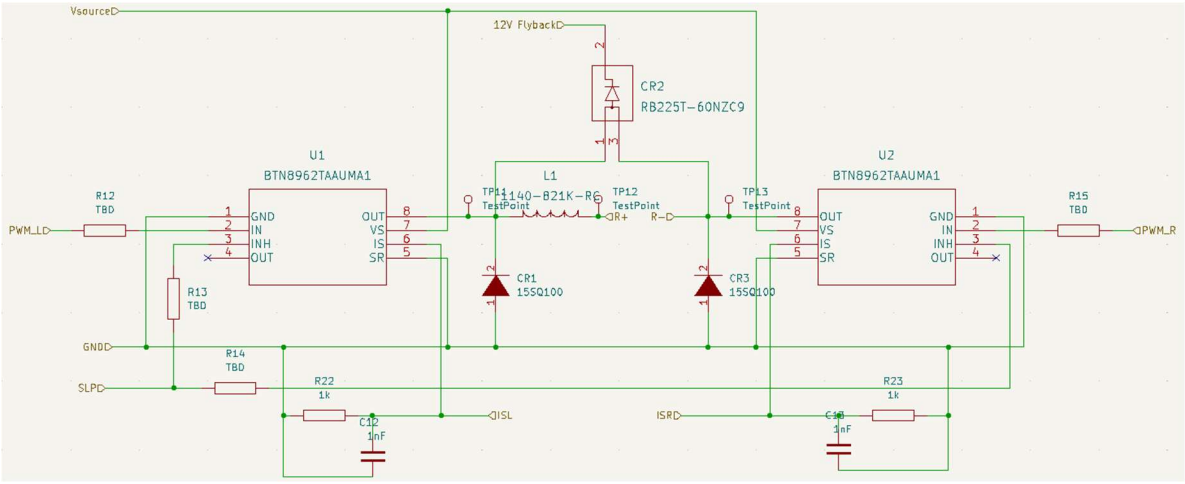


Figure 10: H-bridge Configuration with Circuit Protection

The H-bridge circuit seen in Figure 10 also includes circuit protection elements. The main concern was the inductor discharge protection, as a sudden broken circuit could force the inductor to output a high voltage that destroys other components, such as the transistors within the half bridges [30]. Even switching off the device before the inductor discharges could risk destroying components since it would then only have connections to ground. The flyback diode [32] was added to allow for a constant connection to the battery in case the inductor needs to discharge after the power switch [28] is flipped. Low power input protection elements, like resistors and bypass capacitors, were also added for microcontroller [4] interfacing with the half bridges.

Heatsink/Fans:

Along with the Peltier devices there needed to be heatsinks on either side to help move energy across the system. From calculations the worst case total thermal energy requirement was

61.583W. With four Peltier devices being used each device needed to remove 15 W from the container. With the amount of energy needed to be moved from the container and the energy created by the Peltier devices themselves the thermal resistance for the internal and external heatsink were calculated. Thermal resistance is the temperature rise per unit rate of heat dissipation. The thermal resistance for the internal heatsink was 0.33 C/W and the external heatsink was 0.22 C/W. More detailed calculations for thermal resistance for each heatsink can be found in Appendix E. Due to budget and availability constraints there was no heatsink that met these requirements without the need for a fan. Fans provide air flow and improve the thermal resistance of heatsinks. The heatsink that was selected was 512-6U heatsink [32] for both the inside and outside of the device. As see in Figure 11 the heatsink (6 inches) with no forced air has a thermal resistance of about 0.7, which is much higher than either calculation, but with an air flow of 50 cubic foot per minute (CFM) the thermal resistance can be decreased to 0.11 and with 100 CFM the thermal resistance is reduced to 0.07.

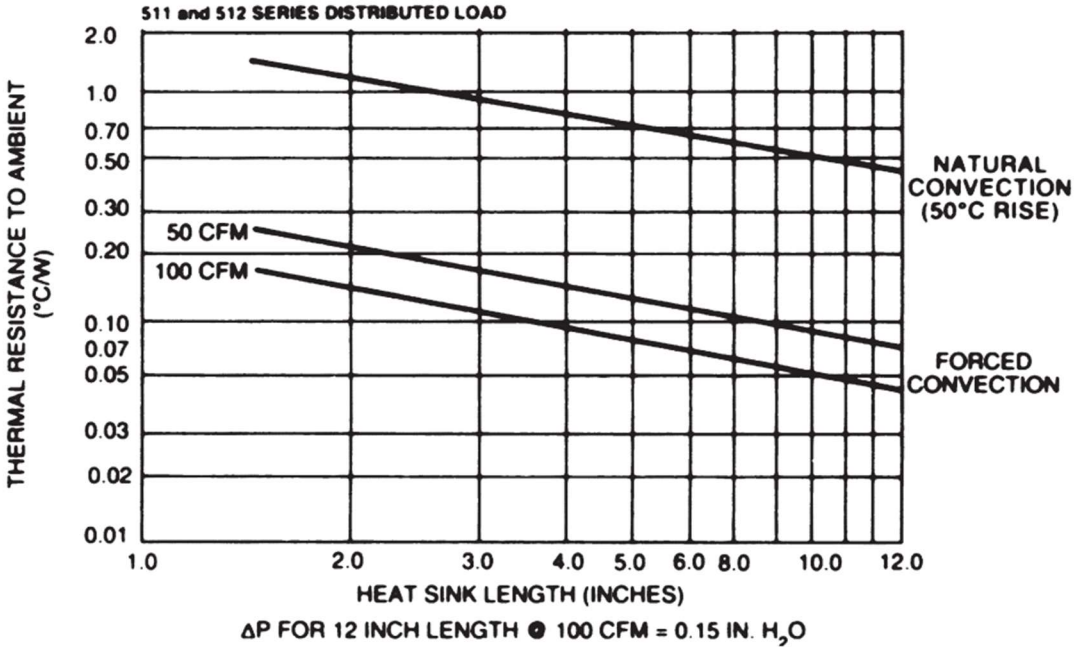


Figure 11: Heatsink Thermal Resistance Graph

The fan [26] was selected based on the size of the fan and the CFM it could provide. The final fans selected can provide 69.7 CFM, which should get the thermal resistance down to 0.1C/W, which is lower than the calculated values for both heatsinks. The fans had a PWM signal already, meaning that the device had its own internal MOSFET, thus reducing the circuitry needed to connect the fans to the MSP. The fans also had a tachometer signal output that sent the current PWM signal to the MSP. The signal output was connected electrically to the MSP but had not been implemented on the software side since there was no need for it with the current system in place. This signal could be used for additional testing.

Firmware:

The MSP430F5529 acts as the communicator between the user interface components, temperature sensors, hardware components, and the fuzzy control algorithm. All code was written in C using Code Composer Studio. A pin mapping for the microcontroller is described below.

Pin	Connect to	Pin	Connect to
3V3	Temp Sensor VCC	5V	
P6.5 (analog)	Current Sensors	GND	Temp Sensor GND
P3.4	LCD RS	P6.0	LCD D0
P3.3	LCD RW	P6.1	LCD D1
P1.6	LCD E	P6.2	LCD D2
P6.6 (analog)	Current Sensors	P6.3	LCD D3
P3.2 (SCLK)		P6.4	LCD D4
P2.7	Temp Sensor ALERT	P7.0	LCD D5
P4.2 (SCL)	Temp Sensor SCL	P3.6	LCD D6
P4.1 (SDA)	Temp Sensor SDA	P3.5	LCD D7
P2.5 (PWM out)	H-Bridge Hot	GND	
P2.4 (PWM out)	H-Bridge Cold	P2.0 (PWM out)	
P1.5 (PWM out)	Heatsink/Fans to internal	P2.2	Lid Button
P1.4 (PWM out)	Heatsink/Fans to external	P7.4	
P1.3	Heatsink/Fans from internal	RST	
P1.2	Heatsink/Fans from external	P3.0 (MOSI)	
P4.3	H-Bridge Sleep Pin	P3.1 (MISO)	
P4.0	Test Point	P2.6	Inc Button
P3.7	Test Point	P2.3	Dec Button
P8.2	Test Point	P8.1	
		J10 (3.3V)	Power (Battery)

Figure 1212: Pin Mapping for MSP

LCD Screen:

The 16x2 character LCD screen mainly displays the current temperature inside the compartment and the user's desired set temperature. The LCD controller is the HD44780U dot-matrix liquid crystal display controller [33], which reads and writes data to the screen via 8 data bits. This is a standard LCD controller that can operate in either 8-bit or 4-bit mode. Originally,

we had considered operating the LCD in 4-bit mode as this has the benefit of reducing the number of GPIO pins used for the LCD on the microcontroller. However, the major tradeoff for operating in 4-bit mode is latency. In 4-bit mode, 8-bit character ASCII values need to be divided into 4-bit nibbles and sent one at a time. Since we wanted to be able to display accurate, real-time temperature readings on the LCD, we decided that gaining extra pins on the microcontroller was not worth the tradeoff of generating latency for our application. Since the project only needs to write to the LCD screen, the Read/Write port is automatically set to low in software. By default, the LCD screen displays the current temperature and the set temperature in Fahrenheit. The LCD screen also has the ability to display different screens of information, such as the internal and external heatsink temperatures when both buttons are pressed simultaneously. We also utilized buttons on the MSP430 to be able to display more screens of information, such as the internal and external fan PWM, the PWM for the Peltier modules, and whether the device is operating in cooling or heating mode. These extra screen displays were implemented for debugging purposes. Figure 13 shows the pin connections for the LCD screen to the microcontroller.

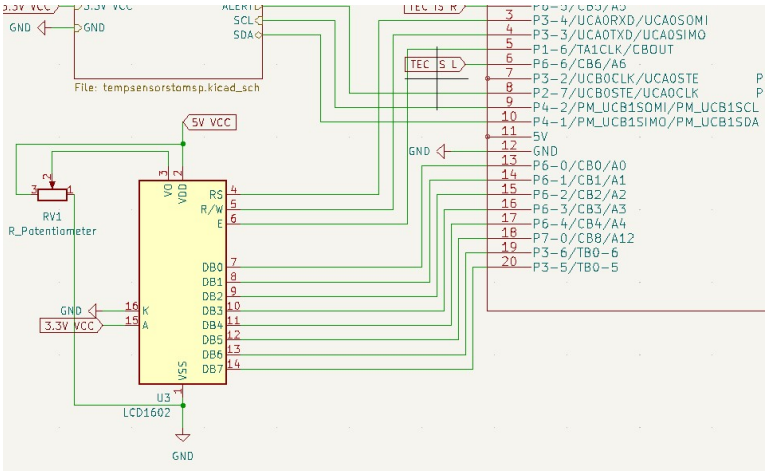


Figure 1313: LCD Screen to MSP430

Buttons:

There are three external pushbuttons – two buttons for incrementing and decrementing the set temperature [34] and one button for the lid [35] – that are utilized for the project, and two pushbuttons from the MSP430 microcontroller that are used for debugging purposes. The user will primarily interact with the increment and decrement buttons, which are mounted on the front of the outer case along with the LCD screen. The set temperature value displayed on the LCD screen will increase by 1 degree Fahrenheit if the increment button is pressed and will decrease by 1 degree Fahrenheit if the decrement button is pressed. Since the button's bounce time is 10ms, the period of the buttons' statuses reading task is set to 10ms, to ensure that the signal of the button stabilizes in order to accurately read the buttons' statuses.

The buttons have identical circuitry except that their output is connected to different pins on the microcontroller. They also each have their own protection circuitry to prevent potential damage to the microcontroller through the pins to which the buttons are connected. The circuit

has a pull-up resistor, which is necessary for the function of the pushbutton, a small capacitor and small resistor to serve as a reservoir for charge from one's finger and to limit the amount of current that goes through the switch when the button is pressed, and a TVS diode [36] to absorb abnormal voltages and suppress static electricity going into the microcontroller (Figure 14).

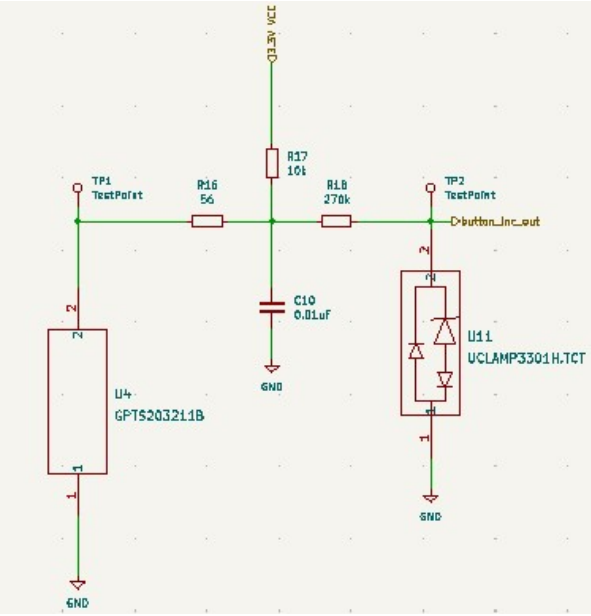


Figure 1414: Increment/Decrement Set Temperature Buttons

Another external button that is used is the lid button, which is mounted at the top of the outer case. This button also has protection circuitry (Figure 15). When the lid is opened, which leaves the lid button unpressed, the MSP430 microcontroller sets the PWM duty cycle for the internal fan to 0, effectively stopping the fan. Otherwise, the internal fan would run at the specified rate. This is to conserve energy when the box is opened.

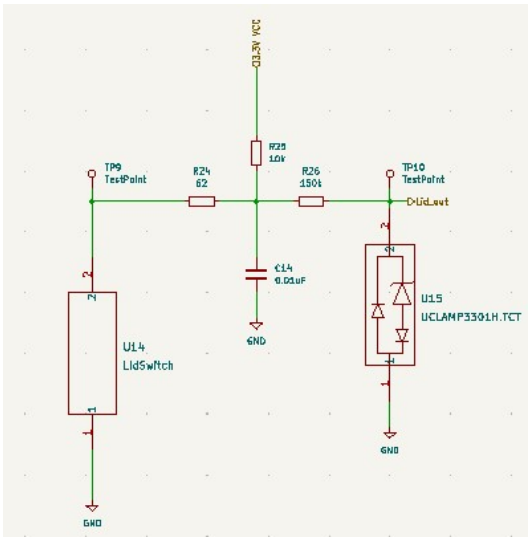


Figure 1515: Lid Button

For debugging purposes, a few features are implemented for different variations of buttons pressed. The internal and external temperatures are displayed when both the increment and decrement buttons are pressed. Additionally, the internal and external fan duty cycles are displayed on the LCD screen when both MSP430 buttons are pressed. Lastly, the h-bridge duty cycle and the current operation mode, heat or cool, are displayed when the rightmost MSP button is pressed. These features were added to aid with the debugging process.

Temperature Sensors:

Three temperature sensors [37] are used to collect temperature data for the compartment, the internal heatsink, and the external heatsink. The MSP430 microcontroller interfaces with the temperature sensors via inter-integrated communication (I2C). The I2C primarily communicates using the SCL and SDA lines, where the SCL is the slave clock, and SDA is the data line.

In order to increase accuracy in the temperature reading, the conversion resolution was set to 12-bit, the highest resolution, which can read temperature increments of 0.0625°C. The 12-bit resolution requires 200ms for conversion, hence the temperature reading task is done only every 200ms, which is elaborated more under the *Task Scheduler* section. To set the conversion resolution to 12-bit permanently, the MSP430 needs to send three consecutive bytes to the temperature sensor. First, it sets the sensor's pointer register to the nonvolatile configuration register, which is 0x11. From there, the MSP430 writes two bytes to the nonvolatile configuration register, where the upper byte is 0x60, corresponding to a conversion resolution of 12-bit and the rest of the default value stays the same in that byte. The lower byte is 0x00, the default value. The content of the nonvolatile configuration register will remain the same after power-up/reset, with the contents of the nonvolatile configuration register being copied to the configuration register on power-up/reset by default. As such, the nonvolatile configuration register was set only once, with subsequent deployment of the code no longer including the initialization of the bit resolution on the temperature sensors.

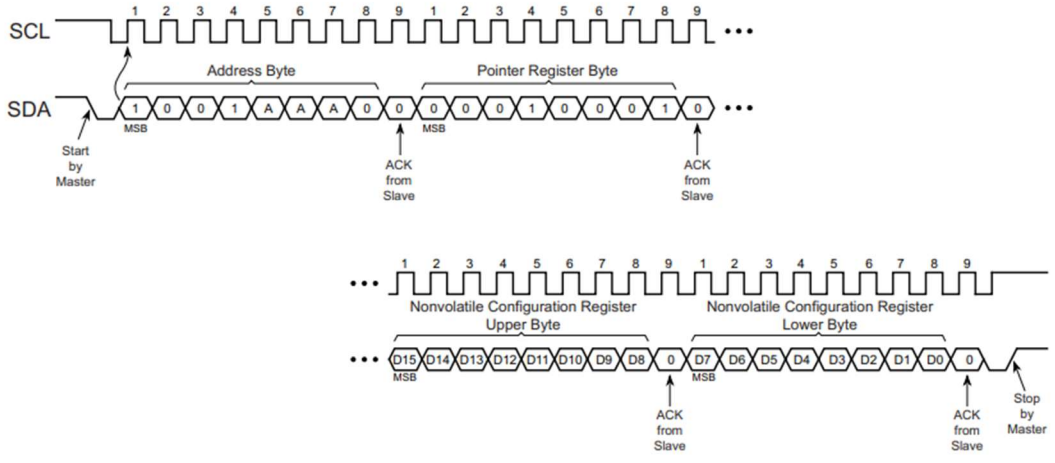


Figure 1616: Writing to the Nonvolatile Configuration Register

To read the data from each temperature sensor, the MSP430 first needs to set the value of the sensor's pointer register to the temperature register, which is 0x00. Then, the MSP430 can

read the data from the temperature sensors, which is stored in two bytes. The temperature data format is in two-complement and in Celsius, hence the data is converted to decimal after the read operation using the equations below, where *upper* is the upper byte read from the temperature register, and *lower* is the lower byte from the temperature register. If the temperature is negative, the first equation would be used. The temperature data is then converted to Fahrenheit. The temperature data is averaged out of the most recent ten readings to ensure the temperature data is not fluctuating too frequently.

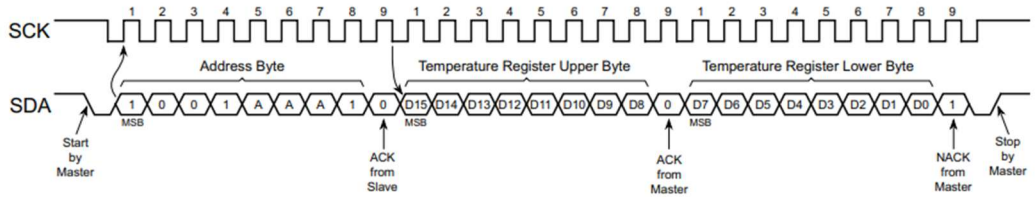


Figure 1717: Reading from Temperature Register

$$temperature = (upper \ll 4 + lower \gg 4) - 4096$$

$$temperature = upper \ll 4 + lower \gg 4$$

Figure 1818: Twos-Complement to Decimal for Temperature Data

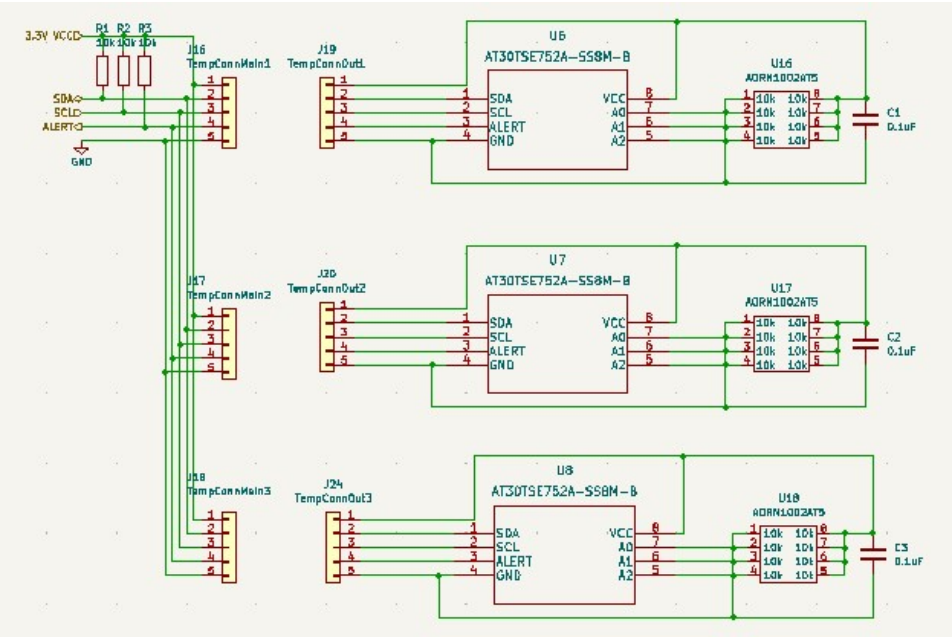


Figure 1919: Temperature Sensors

Fans PWM:

The fans operate off a linear control algorithm, factoring in inputs from the temperature sensors and the operational mode of the system. There are two fans present on the device: the internal heatsink fan, and the external heatsink fan. Each fan has a separate control schema for determining the RPM at which the fan should be run.

The internal heatsink fan is responsible for equalizing the temperature of the heatsink and the temperature of the chamber. As such, the fan PWM is set higher as the difference between the internal heatsink temperature and the current chamber temperature increases. The current control schema is shown in Figure 20. The fan will always be running at a minimum of 10% PWM and will ramp up linearly from 10% to 50% between 1 degree and 7 degrees of difference. From 7 degrees to 10 degrees of difference, there's another linear ramp up from 50% to 100% PWM. To prevent air from escaping too fast from the interior of the box upon opening of the box, a lid button status is also taken in as input, where an unpressed lid button represents an open box, and the PWM signal controlling the fan is set to 0.

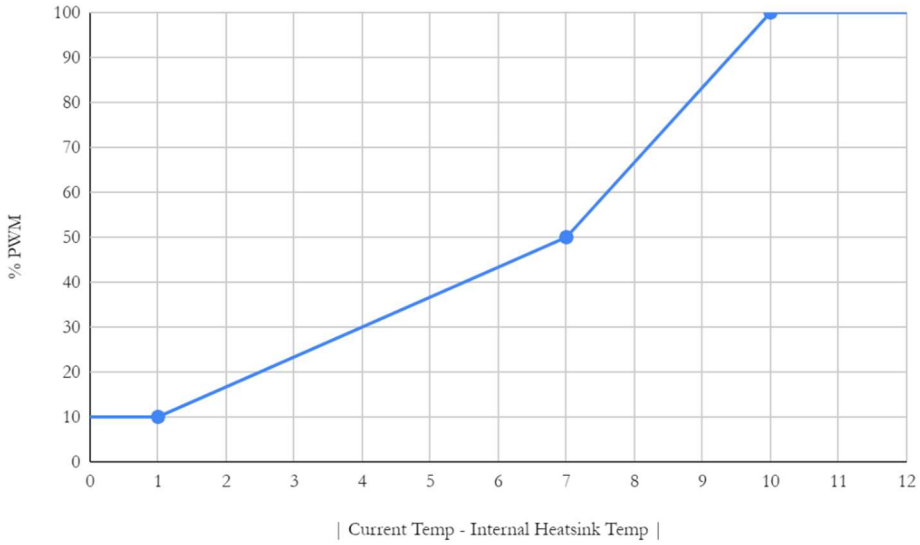


Figure 2020: Internal Heatsink Fan Control Schema

The external heatsink fan is responsible for keeping the heatsink as cool as possible while the system is under cooling mode. As such, the external heatsink fan's PWM signal is determined solely by the measured temperature of the external heatsink. Ideally, the external heatsink fan would take in as input the difference between the ambient temperature and the external heatsink fan, but there is no ambient temperature sensing capability present on the device. As shown in Figure 21, the fan operates at 50% PWM under low temperature conditions, ramps down to a base 10% PWM rate around room temperature and ramps up to 100% PWM between 80F and 90F external heatsink temperatures, with a consistent 100% PWM from 90F onwards.

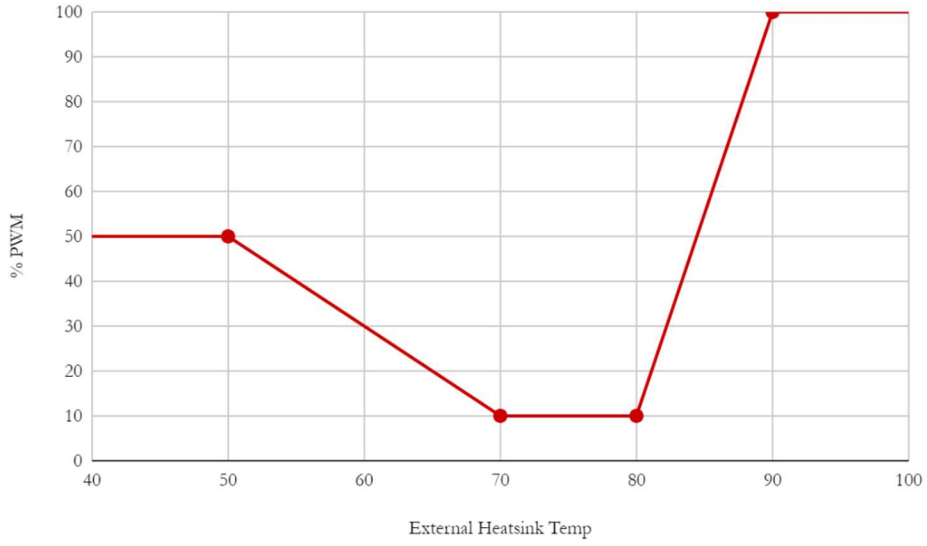


Figure 2121: External Heatsink Fan Control Schema

Fuzzy Control Algorithm:

The fuzzy control algorithm is responsible for determining an appropriate PWM response to be sent to the h-bridge. Other common control algorithms for temperature regulation devices include on-off controllers and PID controllers, with PID controllers being by far the most common controller type for temperature regulation. We chose to use a fuzzy logic controller over a PID controller because of the simplicity of developing a fuzzy logic model, as well as improved robustness of fuzzy logic compared to PID controllers over a wide range of operational conditions [37]. One characteristic of fuzzy logic control is that the operator can apply their experience in operating the system to tune the system parameters [39]. In comparison, PID controllers are heavily dependent on system parameter tuning, and require a separate tuning algorithm, like the Ziegler Nichols algorithm, for determining the optimal parameters for the controller. Their performance degrades upon leaving the operational range they were tuned for, often with overshoot behavior observed before settling at the set point temperature. Given the extremely wide range of temperatures the controller is meant to operate under, fuzzy control has been isolated as the most suitable controller for this project.

Due to using Peltier modules as the choice of cooling/heating for the device, the fuzzy function must first determine the operational mode of the device. When the set temperature exceeds the external heatsink temperature when in cooling mode, the system will flip to heating mode, and vice versa. This requirement was determined to avoid rapid oscillation of the system between heating and cooling modes. As a result, heating is performed by decreasing the PWM in cooling mode, and cooling is performed by decreasing the PWM in heating mode. The operational mode also determines the sign orientation of the inputs to the fuzzy logic controller.

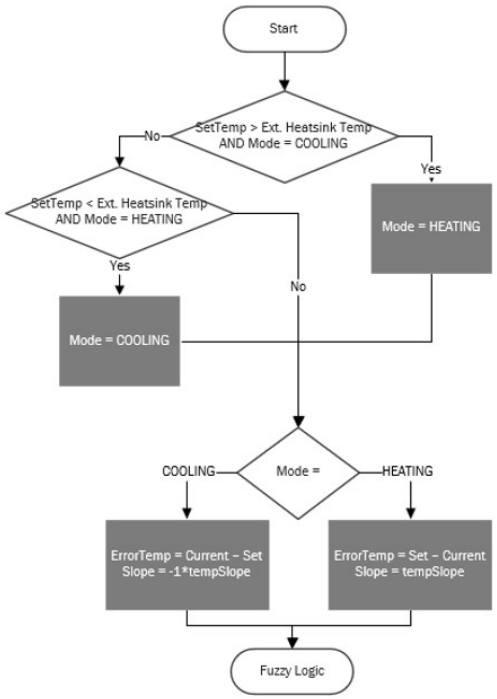


Figure 2222: Operational Mode Flow Chart

The first input to the fuzzy logic controller is the difference between the current chamber temperature and the set temperature. The membership function of the temperature differential input is constructed using triangular membership functions, as shown in Figure 23. There is overlap between the membership functions, resulting in a single input value resulting in a fractional degree of membership to multiple different linguistic terms. The linguistic terms for the membership function are assigned as follows: NB = negative big, NM = negative medium, NS = negative small, Z = zero, PS = positive small, PM = positive medium, and PB = positive big. The peak locations of each membership function are considered system parameters, and the values were tuned for optimal performance during testing.

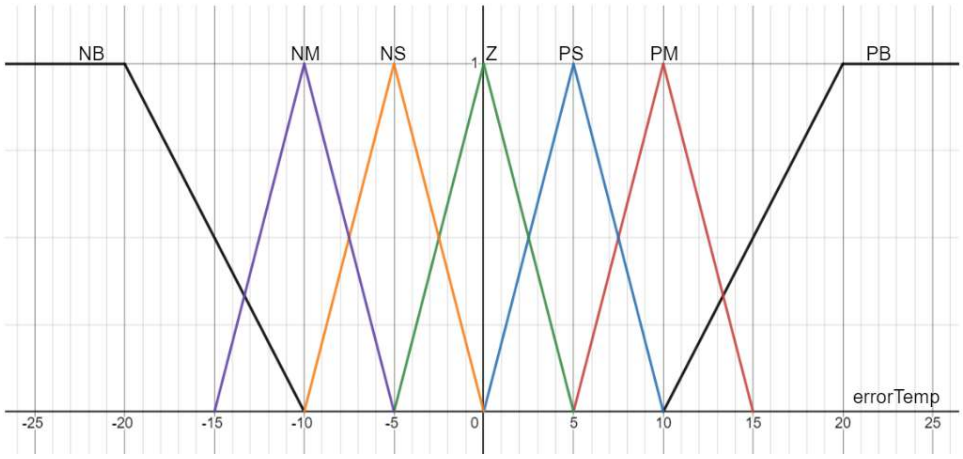


Figure 2323: Temperature Differential Membership Function

The second input to the fuzzy logic controller is the rate of change in the current chamber temperature. The rate of change is calculated by taking the difference between the current chamber temperature and the average of the past ten chamber readings. The membership function of the change in temperature input was similarly constructed using triangular membership functions, as shown in Figure 24, and the assignment of the input to its membership function works off the same concept as the temperature differential membership function. The linguistic terms for the change in temperature membership function operate off a different set of peak location values.

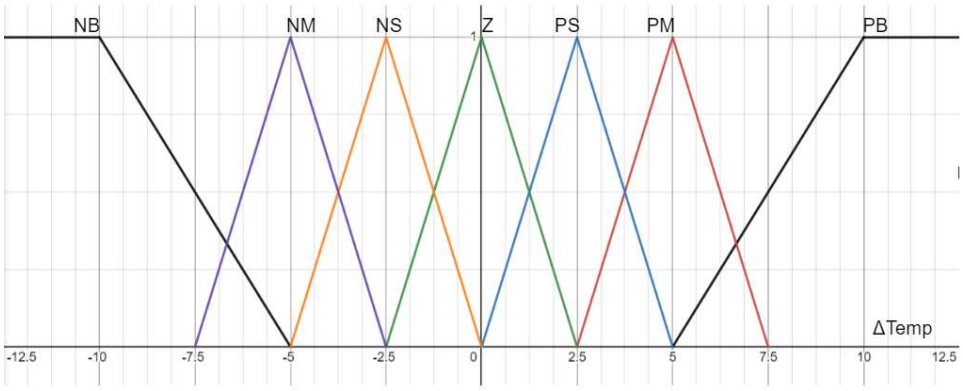


Figure 2424: Change in Temperature Membership Function

Having converted the input values of the temperature differential and the change in temperature into a degree of membership to various linguistic terms, a ruleset is constructed to transform the linguistic membership values into a concrete PWM output. The table below defines the relationship between the temperature differential linguistic terms and the change in temperature linguistic terms. The output linguistic terms for the ruleset similarly have separate values from the input linguistic terms. The actual output from the fuzzy logic controller is a value by which the existing PWM output value should be changed by. These values were also tuned during testing to determine the sensitivity of the system to changes in system temperature conditions.

		ΔTemp						
		NB	NM	NS	Z	PS	PM	PB
ErrorTemp	NB	NB	NB	NB	NB	NM	NS	Z
	NM	NB	NB	NB	NM	NS	Z	PS
	NS	NB	NB	NM	NS	Z	PS	PM
	Z	NB	NM	NS	Z	PS	PM	PB
	PS	NM	NS	Z	PS	PM	PB	PB
	PM	NS	Z	PS	PM	PB	PB	PB
	PB	Z	PS	PM	PB	PB	PB	PB

Figure 2525: Output PWM Ruleset

The actual output is determined by taking a weighted average of the different output linguistic terms. The weight of the output linguistic term is determined by taking the maximum weight value between the two input linguistic terms, given that they're non-zero. For example, if the temperature differential input is assigned a weight of 0.7 on NM and 0.3 on NS and the change in temperature input is assigned a weight of 1 on PS, then the output will be a weighted average between an output value of 1 on NS and 1 on Z, resulting in the final output value being the middle value between the values corresponding to NS and Z output linguistic terms.

The final output from the fuzzy logic controller is bounded by the minimum and maximum values that can be assigned to the h-bridge PWM, which is 0 and 624 respectively. If the fuzzy logic controller determines a negative value — that is, if the output function has a degree of membership to the PB, PM, or PS output linguistic terms — and the current PWM signal being driven to the h-bridge is already 0, then the PWM signal will remain 0. Same situation applies if the fuzzy logic controller determines a positive value, and the current PWM signal being driven to the h-bridge is already the maximum value of 624.

Peltier PWM:

The MSP430F5529 [4] interfaces with the Peltier modules [27] by providing logic level inputs to the half bridges [30]. The inhibit pins of both half bridges are driven by a single GPIO output such that, when the inhibit pin is driven high, the H-bridge is operating as intended, but when driven low, both sides of the H-bridge are linked to the power source. The logic inputs of the half bridges depend on the desired functionality set by the Fuzzy Control algorithm. When heating functionality is set, one half bridge is set to a constant connection to ground by setting the associated P2.4 GPIO pin to a GPIO output of 0. The other half bridge is set to operate using a PWM signal asserted by the P2.5 GPIO pin adjusted for its Peripheral Module Output function linked to the Timer A2 Capture Compare Register 2 (CCR2). This CCR2 value is set using a determined value from the previously mentioned Fuzzy Control algorithm. The cooling functionality effectively reverses this logic such that the half bridge that previously connected to ground is operating using a PWM signal and the other half bridge is now driving to ground. The P2.4 GPIO pin that drives the PWM signal for cooling is similarly linked to Timer A2 for its Peripheral Module Output function, but it is linked to Capture Compare Register 1 (CCR1) instead of CCR2 and this value is also determined by the control algorithm.

The Timer A2 on the MSP430F5529 [4] needed to be configured for a PWM frequency at 20 kHz based on the switching frequency limitations of the half bridges [30]. This was done by setting the Timer clock source to the Master Clock (MCLK). The MCLK is configured to the Digitally Controlled Oscillator (DCO) clock, configured to run at a frequency around 25 MHz. The Timer A2 clock source divider was set to 2 and then Capture Control Register 0 (CCR0) was set to 624 to set the overall PWM frequency to 20 kHz.

Task Scheduler:

A task scheduler is implemented to handle the execution of all the I/O devices. The MSP430 microcontroller needs to collect input and send output signals to four primary interfaces: the buttons, the heatsink fans, the h-bridge circuit, and the temperature sensors. The

task scheduler for the system is configured on Timer1_A0, with the timer CCR0 value set such that the timer interrupt handler triggers every 10ms. The task scheduler itself is an interrupt service routine that tracks the number of interrupt triggers that have occurred and calls the tasks according to the frequency defined for the task. Using the 10ms period generated by the timer, we can set the tasks to run at a known frequency.

The first task running on the task scheduler is the fuzzy control algorithm, at a frequency of 1Hz, or once every second. This task is responsible for calling the fuzzy logic function for computing the PWM output to the h-bridge powering the Peltier devices, then updating the CCR value on the timers connected to the h-bridge pins to modulate the power being supplied to the Peltier devices. This task is also responsible for setting the operational mode of the device to either heating or cooling. The frequency of 2Hz was chosen to ensure that the computation-heavy fuzzy function was not being called too often, and that the PWM output is changed at a slow rate to account for the thermal inertia associated with the heatsinks and the thermal diffusivity of the air inside the box,

The second task running on the task scheduler reads all three temperature sensors, at a frequency of 5Hz, or once every 200ms. The period of 200ms was chosen to accommodate the conversion period necessary for the 12-bit resolution of the temperature sensors. The temperature values from this task are used as inputs in the rest of the system.

The third task handles the fan control algorithm, at a frequency of 1Hz, or once every second. The exact frequency at which the internal and external fan run are not particularly important. The fan algorithm runs based on the current temperature conditions of the device, and thermal inertia means that the current conditions of the device change at a gradual rate. This task is also responsible for writing the PWM outputs from the fan control algorithm to the CCR values of the timers controlling the fans.

The fourth and final task being handled by the task scheduler is the button handler, at a frequency of 100Hz, or once every 10ms. This task reads from five different buttons: the increment temperature button, the decrement temperature button, the lid button, and two of the buttons on the MSP430F5529. The task is also responsible for changing the set temperature of the device based on the pressing of the increment and decrement buttons. The task period of 10ms was chosen as a suitable debounce period for ensuring that the signal of the buttons stabilizes between reads upon press or release.

Outer Case/Mechanical:

The mechanical design for the case is comprised of several major components: wood casing with cutouts for peripheral devices such as the LCD and buttons, expanded polystyrene (Styrofoam) insulation, and the heatsink assembly. The overall dimensions of the box are: 14.25” x 12.5” x 15.75”. This includes the battery compartment at the bottom. The dimensions of the insulated interior chamber where the objects to be cooled/heated are placed are 11” x 8.5” x 7.5”.

The wood casing is 0.5” thick plywood. It has cutouts for the LCD screen, increment and decrement buttons, and the heatsink (Figure 26). In the back of the device, there is an opening

with bumpers forming a slot for the battery to sit horizontally (Figure 17). There is a middle wood panel separating this battery compartment from the insulated chamber of the device. The box also features a wood lid affixed to a Styrofoam panel that fits snug inside the Styrofoam lining the walls of the box.

Figure 2626: Front and Heatsink Sideview of Device

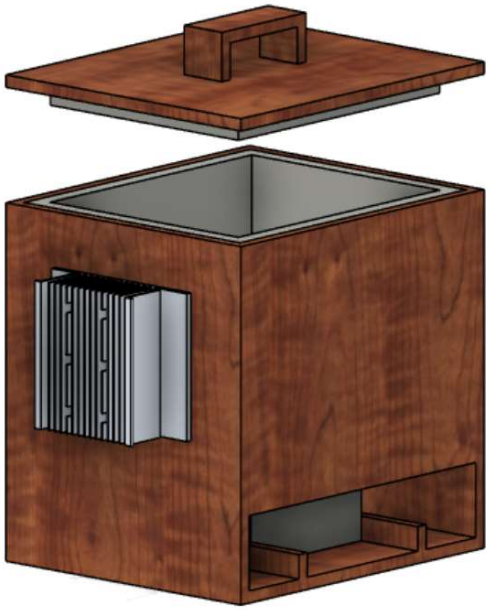


Figure 2727: Back and Heatsink Sideview of Device

One panel of Styrofoam insulation is nominally 0.75” thick, but its exact thickness was approximately 0.70”. While we had originally planned to use a single layer of Styrofoam for insulation in the chamber, upon testing, we found that temperatures were not changing fast enough, which we attributed to insufficient insulation. Thus, we added an additional layer of Styrofoam to three walls of the box to increase its insulation for a total of 1.5” thick insulation on every wall except the heatsink wall. We found that this did indeed improve the results, and the only tradeoff was slightly decreased volume in the chamber.

Designing the heatsink assembly was a relatively challenging part of the mechanical design. The heatsinks are necessary to prevent the Peltier modules from overheating. There were several considerations that needed to be taken into account when determining how to design this assembly and incorporate it into the box. Firstly, the Peltier modules, which are affixed to the external heatsinks, are only 0.15” thick, but they need to be able to have good thermal contact with both the external and internal heatsinks. Thus, a square bar of stock aluminum with length and width of 1.5” (approximately the same length and width of a Peltier module) needed to be purchased and machined down to four 0.55” long chunks to be placed between the Peltier module and the internal heatsink (Figure 28). Furthermore, compression and expansion of the heatsink assembly during cooling and heating needed to be considered to prevent the Peltier modules from cracking while still ensuring that they are able to maintain good thermal contact with the heatsinks. Thus, Belleville washers were utilized to account for this compression and expansion. The four Peltier modules are arranged in a square, anchored by a screw above and below each one, for a total of six screws. Each screw also needs to be thermally protected from the heatsink, so nylon shoulder washers were used so that the screws do not directly contact the heatsinks. The screw assembly is as follows: screw, Belleville spring washer, flat washer, and nylon shoulder washer. Additionally, a temperature sensor breakout board was screwed onto the base of each heatsink with a thermally conductive pad in-between to allow for thermal contact between the thermal vias on the board and the heatsink. Finally, a fan was affixed to each heatsink, which simply requires a screw on each side of the heatsink. The fan is necessary to dissipate the heat on the heatsinks to prevent them from overheating as well as to circulate the air in the chamber of the box. The fans are not modeled in the CAD design, but they are implemented on the prototype.

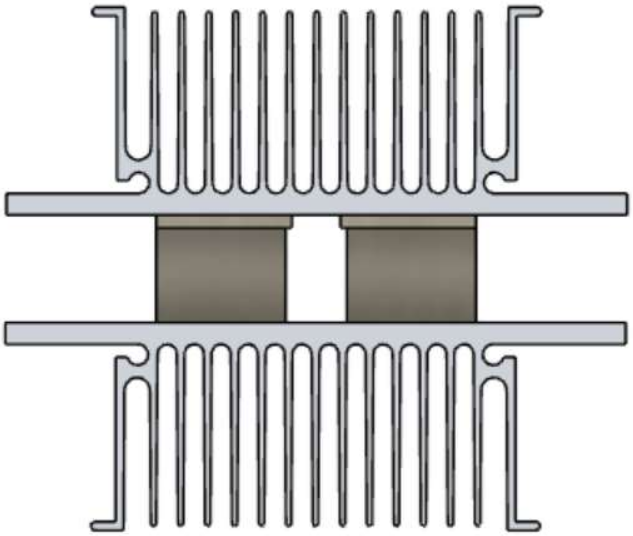


Figure 2828: Heatsink Assembly

Construction of the box involved cutting out each face of the box with a table saw and bandsaw and then using a drill press and jigsaw to make the necessary cutouts for each face. On the front face of the box, a router was used to decrease the thickness of the area where the LCD screen and buttons sit. The middle panel had a small cutout on the edge to allow wires to feed through. The box was then held together with wood screws, and the middle panel, which has the PCB attached to it, was screwed in last. Next, the Styrofoam was measured and hand-cut with a precision knife for each face of the box, with zig-zag edges cut to allow for inter-locking of pieces. On the Styrofoam layer for the heatsink face, four square holes were cut to hold each Peltier module and its accompanying aluminum extender block. It was important to err on making these holes slightly smaller than the actual Peltier modules to ensure that they would be held tightly in place by the compression of the Styrofoam. When assembling the heatsink assembly, a thin layer of thermal paste was applied to each side of the Peltier module and the side of the aluminum extender block that contacts the internal heatsink. Then, the screws and washers were used to hold the entire assembly together. The fans were then screwed on to the heatsinks. Finally, silicone sealant was applied to the edges of the Styrofoam in the chamber of the box as well as the outside cracks in the wood casing to further increase insulation. Photos of the final construction of the box can be found in Appendix F.

Project Time Line

The tasks for this project were divided into several main sections: user interface, temperature sensors, physical design, Peltier module and its associated fans and heatsinks, power supply and battery management, integration of microcontroller into the system, and the temperature control algorithm. The design for all the components, including the power supply,

Peltier module, buttons, and LCD screen, were accomplished in parallel to the overall system design. After the conclusion of the overall system design, we worked on the code for the microcontroller concurrently with the design stages of the components, leaving flexibility in the code to accommodate different ways of interfacing with each component. As the design stages for each individual component finished up, the integration of the hardware components came into play, deciding how and where the components interface with the microcontroller and overall system. Integration of hardware ran in parallel with the microcontroller integration to finalize the code. For each individual hardware component, there was a design, build, and test phase to ensure that they work as intended, for more targeted testing of the microcontroller code to ensure that the system works as outlined by the system design stage. Once all components were designed and built the assembly of the physical box was a serial task that had all team members contribute to since the only way to do whole system testing required a physical object.

Diana was primarily responsible for the power supply development and heatsink/fan design. This involved dealing with the energy and power calculations needed within the system and figuring out how to be able to power all the components and finding a heatsink/fan design that would move the energy within the system appropriately.

Eric was responsible for the Peltier/thermoelectric cooling element design and H-bridge switching circuit. This incorporated calculations for thermal energy requirements, and microcontroller/code interaction with the Peltier unit alongside the associated electrical components. There was also active interaction with Diana since the Peltier units draw a notable amount of power and the heatsink calculations are directly impacted by the Peltier system design.

Thu oversaw the microcontroller integration design to ensure the embedded software and hardware systems are working together properly. She will be responsible for ensuring the hardware components respond accordingly to the input from the front-end (users) via the microcontroller. She will also work closely with Maggie in integrating the user-interacting hardware components such as the LCD screen and buttons.

Maggie was primarily focused on the CAD design and figuring out how the different components would be assembled. She also was involved in the temperature sensors, LCD display screen, and buttons and worked closely with Thu in integrating these hardware components with the microcontroller.

Victor was primarily in charge of control systems and microcontroller programming work. He investigated and designed a fuzzy control system to address the non-linear behavior of the Peltier heating/cooling device and worked with all the other team members to make sure that their components are working as intended with the implemented program on the microcontroller.

From the original Gantt chart and the final there were several major changes. One task that changed was the time needed for the assembly. With assembly the timeline was shifted significantly due to difficulty in finding a place to do the wood and metal cutting. In particular, the heatsink drilling took longer than expected, getting the heatsink drilled was halted for almost two weeks due to difficulty in securing a location for the drilling and then having to wait for parts to be ordered in order to be able to make the specified holes. With the assembly being shifted the whole system testing was shifted as well. For whole system testing, the heatsink construction was needed, since the Peltier devices can't be used without a heatsink to deal with the energy/heat generated. There were other noticeable changes from the initial Gantt chart, this included that some component selection took longer than expected but things like the electrical testing and PCB component placement took significantly less time than initial anticipated.

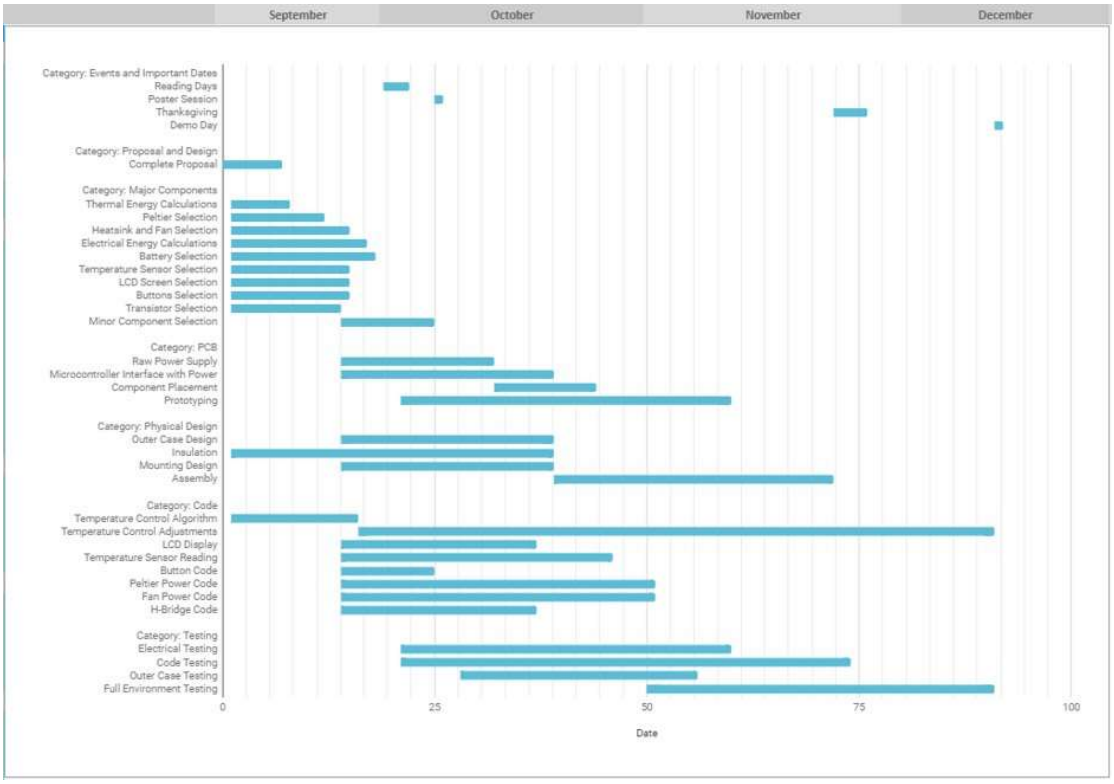


Figure 2929: Original Gantt Chart

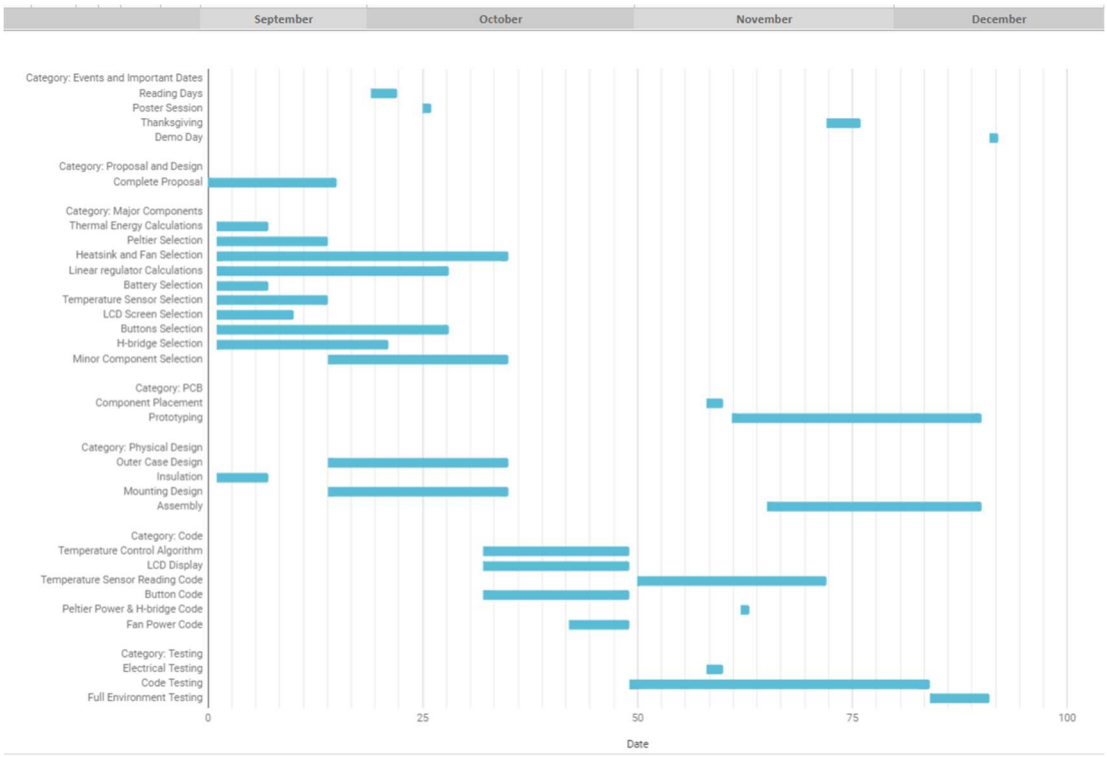


Figure 3030: Final Gantt Chart

Test Plan

The testing plan was divided into the power supply circuit, user interface and the temperature regulation. The first step was to test if we can supply the correct power needed for the different subunits, since without the supply nothing can function. The plan for testing the power supply can be seen in Figure 31. The testing procedure consisted of making sure that 12 volts were being supplied to the linear regulator and that the appropriate voltage, 3.3V or 5V, were being generated from the output, and that 12 V was being supplied to both the H-bridge and fans.

During the testing of power to the different components, the LCD screen was not displaying the same as when power was provided through the microcontroller. Through various voltage readings, it was determined that the problem was caused due to an interaction between the linear regulator and the potentiometer. The 5V output of the microcontroller, used for initial testing, produced the intended behavior for setting a voltage that adjusted the contrast of the screen. Presumably, this was because the microcontroller USB power supply had a much greater input impedance and/or maximum current output than the linear regulator, so the potentiometer was able to perform its role as a voltage divider. When switched to power using the PCB linear regulator, the LCD appeared to display nothing since voltage across the potentiometer was not accurate and improperly set the contrast of the screen. When the potentiometer had a higher impedance value, it always displayed 5V, but after decreasing the resistance enough, it

simultaneously dropped the voltage of the 5V source as well. This indicated that there was a low impedance from the 5V output of the linear regulator, since the input of the potentiometer was always 5V until the resistance was low enough to exceed the maximum output current. This was fixed by first measuring a voltage that produced the desired LCD contrast and then the potentiometer was replaced by two resistors that acted as a set voltage divider for the measured voltage.

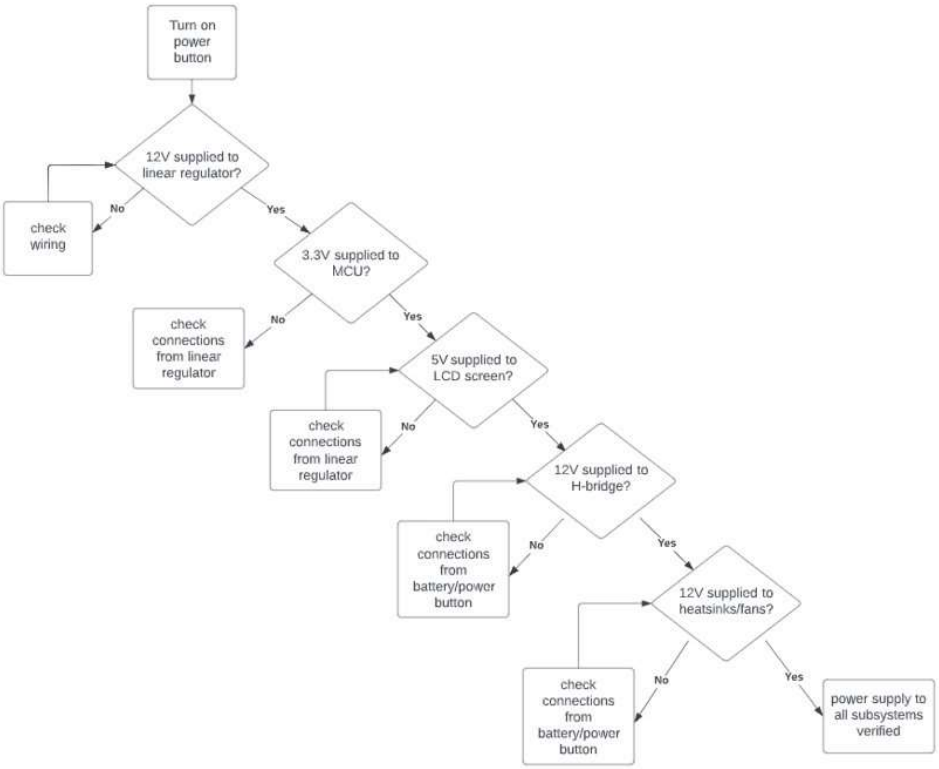


Figure 3131: Power Supply Testing

After testing the power supplies on the PCB we tested the buttons and LCD interaction where the test plan can be seen in Figure 32. The testing plan was to make sure that we could write to the LCD and that when either the increase or decrease button was pressed then the temperature on the screen would change accordingly. Other components like the fan and lid button were also tested to make sure that when the lid button was not pressed the internal fan would stop working, this is to make sure that when the box is opened the internal fan would not be running and pushing the cold/hot air out of the compartment.

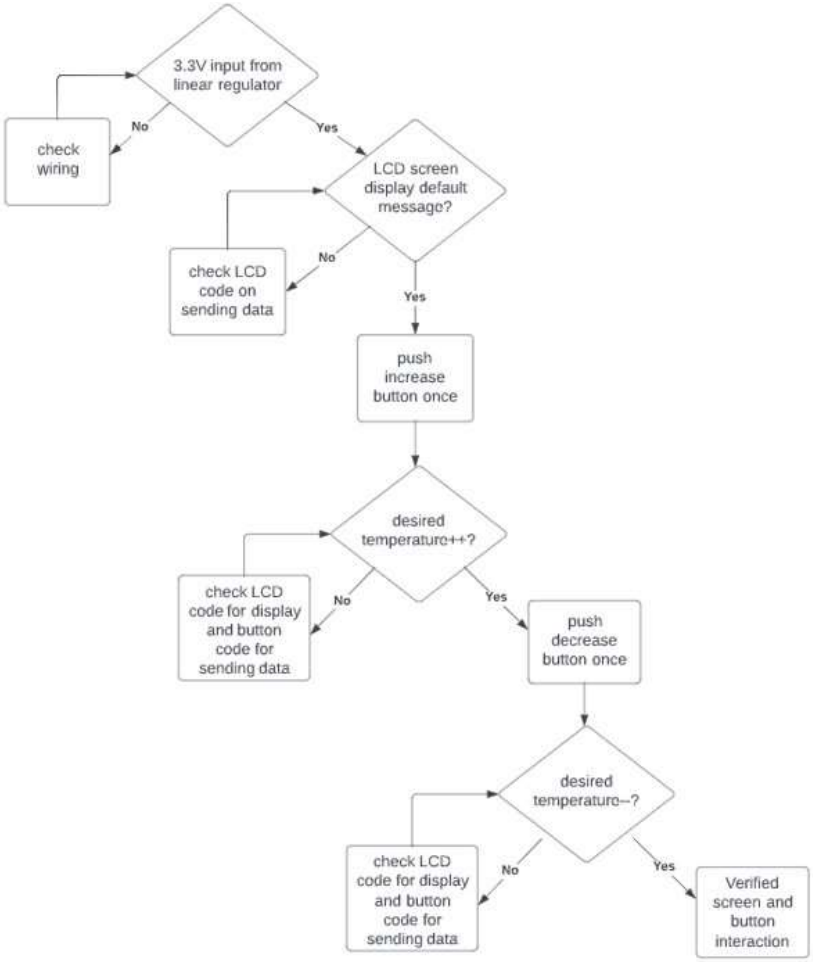


Figure 3232: LCD, Buttons, MSP interaction test plan

When subcomponents, including the PCB, LCD, buttons, and fans, were tested then whole system testing was conducted to see whether the Peltier devices could in fact cool or heat the compartment to a specific temperature. Testing the Peltier devices was broken down to several steps. These steps included first testing each of the four Peltier modules to determine if they all worked as expected. Once it was determined that the Peltier modules worked correctly the container was assembled with the Peltier modules within the heatsink/fan configuration. At this point the software, particularly the fuzzy algorithm and fan code, was tested. This testing included three specifications, if the container can cool or heat to the desired temperature, if it can reach the desired temperature within 2 hours and finally once it reaches the desired can it stay stable within a 4 degree Fahrenheit margin of error. When testing the system, the configuration of the Peltier devices was considered, the configuration that gave the best results was when there were two parallel branches of two in series max of 3A per branch. This ended up being the original design, so no redesign was necessary. Other configurations, such as putting the Peltier devices all in parallel, were tried, but they did not reach lower temperatures since increased current also increased the temperature of the outside heatsink; greater temperature differentials were achieved, but the minimum temperatures were not improved, so the increased power consumption was not desirable.

The temperature sensors were observed to be working initially, albeit with much less precision than desired. Upon debugging, two issues were identified: the temperature sensors were set by default to have 9-bit resolution and 0.5°C precision, and the temperature being displayed was at 8-bit resolution, or 1°C precision. The precision of the temperature sensors was fixed by updating the sensors to use 12-bit resolution as described in the *Temperature Sensors* section of the technical description. The temperature conversion function for converting sensor data to degrees in Fahrenheit had an issue with truncating the lower 4 bits through an early divide operation, which was resolved through reordering the conversion expression.

During testing, the container was not reaching desired temperatures in cooling mode. The insulation was reinforced by adding additional Styrofoam to the interior of the box and using silicone adhesive to fill in any gaps/holes that were present. For heating mode, the container was also not reaching desired temperatures at a satisfactory rate. It was determined that the internal heatsink could not heat up effectively, due to the external fan cooling down the external heatsink. A change was made to the fan algorithm to turn off the external fan during heating mode to allow the combined heatsink system to heat up.

The fuzzy control algorithm was also adjusted to improve the performance of the system. During an initial run of the heating mode, it was observed that the system would overshoot the set temperature, then decrease in temperature until roughly 2 degrees below the set point, before rising again. The change in temperature membership function parameters were modified to respond to smaller changes in temperature. The output PWM parameters were also adjusted to slow down the decrease of the PWM signal, since debugging using the PWM monitor revealed that the PWM signal to the h-bridge was dropping too fast relative to the thermal inertia of the system. Lastly, the period at which the fuzzy control algorithm was run was changed from an initial 500ms to 1s, to make the algorithm run slower. The final parameter tuning is displayed in Figure 33.

Linguistic Term	TempDiff Input (°F)	Change in Temp Input (Δ°F)	Output (PWM)
NB	-20	-0.20	-4
NM	-5	-0.07	3
NS	-1	-0.02	1.5
ZE	0	0	0
PS	1	0.02	2
PM	5	0.07	6
PB	20	0.20	10

Figure 3333: Final fuzzy logic controller parameters

Throughout testing, there'd be circumstances where the system would freeze, and the system would need to be reset to allow the device to run again. Due to the complexity of debugging the entire codebase for areas where the system could fall into an infinite while loop, a watchdog system was implemented to automatically reset the MSP430. This watchdog timer was set with a period of around 2.68 seconds.

The starting set temperature was adjusted from a hardcoded starting set temperature of 50°F to the nearest degree of the current chamber temperature to allow for faster readjustment of the system back to the old set point. Conceptually, having the system instantiate the set temperature with the current chamber temperature also makes more sense than a hardcoded starting set point.

During the late stages of testing, one of the Peltier devices stopped working during testing and suddenly started operating like an open circuit. Given the configuration of the Peltier devices, this defect caused two Peltier devices to stop operating. Upon partial deconstruction of the Peltier module, it was determined that a row of semiconductors had a break in the circuit connection. This problem was resolved with a workaround involving the soldering of a wire to create a short circuit that bypasses the defective row of semiconductors, resulting in a working Peltier module with slightly degraded performance. This fix was the only viable solution outside of proceeding with only three Peltier devices mounted on the project.

Final Results

This project met all requirements specified during the proposal of the project (Figure 37). Thermo-Stasis is a box capable of performing both heating and cooling within a 4-degree margin of error to the set temperature. We conducted performance testing on the box to assess the cooling and heating characteristics of the system, as well as verify that the box would stay within the 4-degree margin of error specified. The capability of the box to increase and decrease the set temperature by 1°F was utilized to set the set point for the following tests. The data for the following tests was collected by recording the temperature of the box directly from what is shown on the device display.

The cooling characteristic of the box turned out to be slightly weaker than initially anticipated, being capable of driving a roughly 25° F difference from the outside ambient temperature. As seen in Figure 34, given a set temperature of 50° F, the box cools rapidly from an initial temperature of 75° F, then slows down significantly upon reaching a 20° F temperature differential between the chamber temperature and the outside ambient temperature. At the end of the 90-minute test, the chamber temperature did not actually reach 50° F, but was projected to reach the 50-51° F range by the end of the 2-hour period. There was a measured temperature differential across the heatsinks of roughly 40° F at the end of the test. Although the system did not manage to reach exactly set point temperature during the duration of this test, the system still falls within the 4-degree margin of error in under two hours, satisfying the initial specifications for the project.

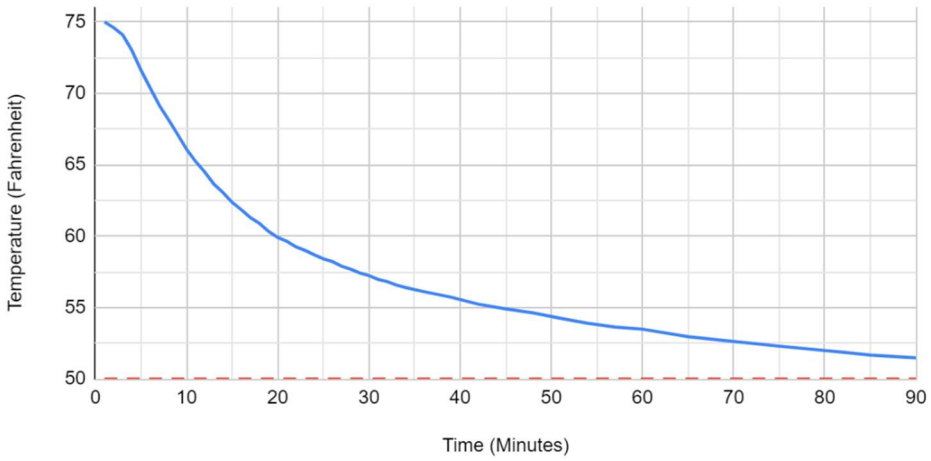


Figure 3434: Cooling Characteristic with Set Temp of 50F

The heating characteristic of the box proved to be more robust given ambient temperature conditions. This is not unexpected, given the heat loss of the system as well as the driven temperature differential by the Peltier modules both serving to increase the temperature of the system. The performance test was done using a set point temperature of 120°F, and a starting temperature of 78°F. As seen in Figure 35, the climbs rapidly before settling at a rate of +0.5° F/min. Set point temperature was achieved within an hour of turning on the device. At set point temperature, the heatsinks were maintaining a 50°F temperature difference, with both heatsinks measuring temperatures above ambient.

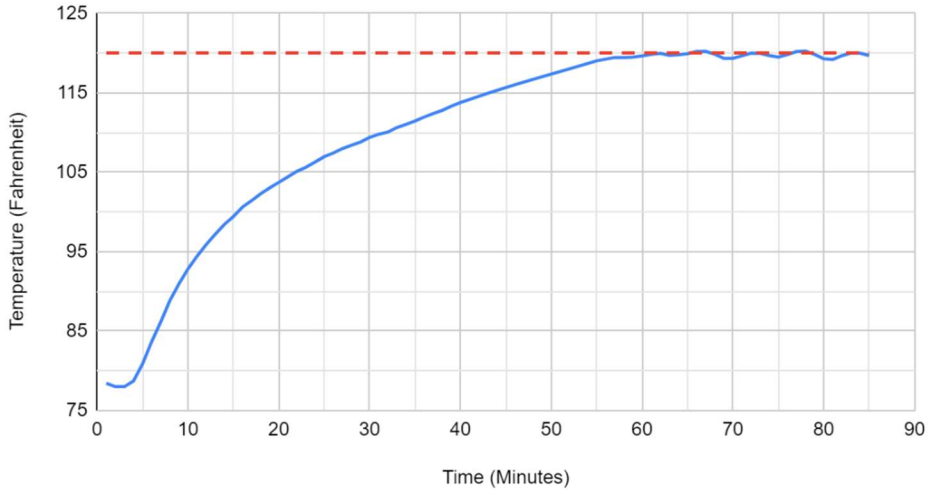


Figure 3535: Heating Characteristic with Set Temp of 120F

Upon reaching set point temperature, the fuzzy logic controller kicked in to keep the temperature of the device steady at the set point. The fuzzy logic controller was tuned to increase the PWM signal faster than it decreases, since the Peltier modules take longer to create a larger temperature differential than it does for the system temperature to naturally equilibrate back to

ambient temperature. There was no overshoot behavior present during the heating performance test, with the temperature rising smoothly to the set point temperature of 120°F. The steady-state margin of error from the set point temperature was measured to be roughly 1F pk-pk, with the temperature rarely exceeding the set point temperature. This steady-state behavior could be further improved through tuning of the fuzzy logic controller parameters. Overall, the chamber temperature is maintained well within the specified margin of error of ±4°F and achieves steady-state behavior comparable to tuned PID temperature controllers.

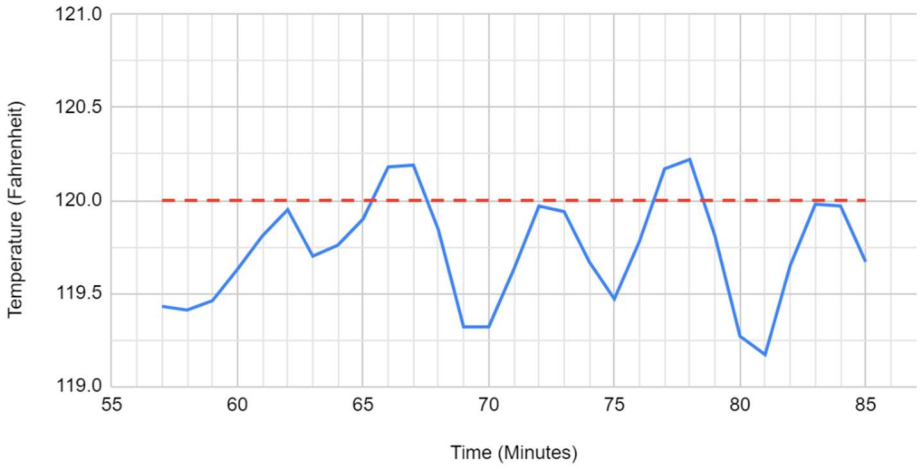


Figure 3636: Steady state response of system at 120F set point

The fans were running in response to the real time changes of temperature across the various components of the system. Without a functioning fan system, the chamber temperature would equilibrate with the internal heatsink temperature through passive air convection and change at a much slower rate than shown in the previous tests. Similarly, the cooling performance of the device would further degrade without the ventilation system, since the increasing temperature of the external heatsink would in turn increase the temperature of the internal heatsink, resulting in an average higher temperature across the system despite maintaining the temperature differential across the heatsinks. Since these characteristics were not observed in the above tests, it can be assumed that the ventilation systems were functional and working as intended.

The criteria set at the beginning of the project are as follows:

- The device can perform both heating and cooling within a specified margin of error from the designated temperature (4 degree Fahrenheit margin of error)
- The device can adjust the internal temperature to the specified temperature within 2 hours.
- The device can stay on for, at minimum, the specified battery life.
- The user can adjust the temperature by 1 degree Fahrenheit increments.
- The device display is capable of the following:
    - Responds to user inputs in a timely manner
    - Displays the current temperature and target temperature

- The system's heating, cooling, and ventilation systems respond to temperature changes and the physical state of the device

Grade	Condition
A	All the expectations mentioned are met.
B	All but 1 of the expectations are met.
C	All but 2 of the expectations are met.
D	All but 3 of the expectations are met.

Figure 3737: Grading Criteria

Costs

The total cost of the project was $622.64, table shows the distribution of cost between the subsystems. Full cost for individual components can be seen in Appendix G. The price is $122.64 over the project budget of $500. The high cost of this project was expected from the beginning since several key components usually cost a lot. In particular, the battery, heatsink/fans, aluminum bar, and Peltier devices were the biggest expenses within the project.

The cost for manufacturing in large quantities would decrease. One main reason is that many of the used items came in certain sized packages/sizes and so after the building of the project there was material left over. Some of the leftover material included Styrofoam, aluminum bar, nuts, washers, and bolts. When manufacturing large quantities there would be less material that would go to waste. Automated equipment would be needed for things like the heatsinks that needed to be customized to have screw holes in the correct location for the Peltier modules. The heatsinks would probably need to be customized entirely, since heatsinks that meet the thermal requirement are not only expensive, but overall low in stock which will make getting enough for large scale manufacturing difficult. This would add to an initial cost increase for manufacturing, but over a long period of time the cost might be recouped. To save cost, the component placement on the PCB was done manually instead of sending it to WWW electronics. Places like WWW electronics do single orders by hand but for mass production of the same PCB, they automate the prosses, so when transitioning to mass production the component placement would be able to be automated and the price per individual PCB would be lower than a one unit order.

As mentioned previously, the costliest components included the battery and the heatsinks and, while the costs could decrease when purchased in bulk, the quantity in stock is a major limitation. When purchasing 100 of the batteries, the price is reduced by 30 dollars. Purchasing 250 heatsinks reduces the price by 10 dollars per device. If the stock of these devices were greater, the price would likely be reduced even further, but there are not enough in stock to have a projected price for mass production.

Item	Cost
MSP	$31.35
PCB parts	$73.85

Power	$131.23
Thermo-related	$214.70
User Interface	$10.36
Assembly	$153.16
Miscellaneous	$7.99
Total	$622.64

Figure 3838: Total project cost by category

Future Work

The project could be expanded on in several ways. Firstly, an alternate source of power, such as solar, could be used to supply the device with power in addition to or instead of the battery. Similarly, the device could be altered to be able to operate when plugged into a power outlet, which would require circuitry for converting AC to DC. Additionally, an accompanying mobile app could be developed that would allow the user to change the set temperature and view the current temperature of the device remotely. For the temperatures displayed on the LCD screen, a feature could be added so that users can have the option to change the temperature reading from Fahrenheit to Celsius. A battery life indicator could also be displayed. Some improvements could also be made to the fabrication of the device. All the pieces were hand-cut using power tools, but the quality of the project could be improved by using computerized tools, such as a CNC router.

To improve the steady-state performance of the device, more tuning could be done on the fuzzy control parameters. In particular, the fuzzy input linguistic term parameters could be tuned to be narrower, to increase the sensitivity of the algorithm to small changes in system conditions. The output linguistic term parameters could also be tuned to further emphasize a faster increase and slower decrease of the output PWM signal, to keep the average steady-state temperature closer to that of the set point. For testing purposes, current sensors for the H-bridge could have been implemented, which could have aided in the tuning of the fuzzy control algorithm, as well as provided insights on the performance of the Peltier modules.

The external fan algorithm could be further optimized with the addition of an ambient temperature sensor. Currently, the external fan algorithm operates off fixed temperature thresholds. For example, the fan always runs at max RPM given a temperature reading of 90F. If the ambient temperature was 100F, the fan would only serve to heat up the heatsink, rather than cool the heatsink as intended. With an ambient temperature sensor, the fan algorithm can be adjusted to account for the difference between the ambient and the external heatsink temperature, allowing for more efficient power use concerning the external fan while not diminishing the heat dissipation performance of the system. The ambient temperature sensor could also be utilized to dynamically set bounds for the upper and lower set temperature bounds for the system based on the ambient temperature conditions.

As previously mentioned, the Peltier configuration limited the performance of the device. Due to the variable resistance of the Peltier modules and the added resistance from the inductor, the Peltier modules did not reach the expected input current. To compensate for this, the Peltier

devices would all need to be configured in parallel with a more effective current source implemented. These devices were tested at higher currents, providing an input current of 6 Amperes to each Peltier, but the hot side temperature increased too much for the heatsinks to dissipate; the temperature differential increased as expected, but the overall performance did not improve since the minimum temperature was greater. Additional modifications to the hardware would need to be made to measure the current through the Peltier devices and then send it to the microcontroller, which would also require modifications to ensure that the Peltier PWM produces the expected current. Drawing more current this way would require some parts, like the diodes and inductor, to be replaced with similar components that are rated for higher current; many inductors with a higher saturation current had lower inductance values, which would affect the ripple current for the device. The higher current draw would also reduce the battery life of the product. A careful evaluation of the economic and thermal performance tradeoffs would be required to implement many electrical hardware improvements.

Some difficulties that were not foreseen at the beginning of the project included the implementation details of an I2C communication, finding a heatsink that would perform to specifications while also being within budget, and the amount of time needed to fabricate the device and the level of precision it required, especially with machining the aluminum bar stock and heatsinks and cutting the Styrofoam panels. A major piece of advice would be to order extras of some of the components such as the linear regulators and Peltier modules. Since these components are relatively inexpensive but vital to the project, it is beneficial to have some extras on hand in case they become damaged during testing or arrive defective.

Projects pursuing the use of Peltier modules should make sure they are willing to dive into thermodynamics calculations and carefully consider power costs. Peltier modules are power hungry devices that can easily be damaged by overheating if they are mounted improperly, and the power consumption means that more care should be taken to ensure that the wire materials and connections on the PCB are designed to handle the high current load running through the system.

References

[1] CDC, "Fast Facts About Food Poisoning," *Centers for Disease Control and Prevention*, Feb. 22, 2022. [Online]. Available: https://www.cdc.gov/foodsafety/food-poisoning.html.

[2] "Building a temperature-controlled Peltier Mini Fridge," *Joseph Rautenbach*, Apr. 03, 2015. [Online]. Available: https://joeraut.com/posts/peltier-mini-fridge/.

[3] S. Kumar, A. Gupta, G. Yadav and H. P. Singh, "Peltier module for refrigeration and heating using embedded system," *2015 International Conference on Recent Developments in Control, Automation and Power Engineering (RDCAPE)*, 2015, pp. 314-319, doi: 10.1109/RDCAPE.2015.7281416.

[4] "MSP-EXP430F5529LP Development kit | TI.com." [Online]. Available: https://www.ti.com/tool/MSP-EXP430F5529LP.

[5] "MSP430G2553 | MSP430G2x/i2x | MSP430 ultra-low-power MCUs | Description & parametrics." [Online]. Available: http://www.ti.com/product/MSP430G2553.

[6] "KiCad EDA." [Online]. Available: https://www.kicad.org/.

[7] "CCSTUDIO IDE, configuration, compiler or debugger | TI.com." [Online]. Available: https://www.ti.com/tool/CCSTUDIO.

[8] "Fusion 360 | 3D CAD, CAM, CAE, & PCB Cloud-Based Software | Autodesk." [Online]. Available: https://www.autodesk.com/products/fusion-360/overview.

[9] "CADLAB.io | Visual collaboration and version control platform for your PCB." [Online]. Available: https://cadlab.io/.

[10] "GitHub: Where the world builds software," *GitHub*. [Online]. Available: https://github.com/.

[11] J. Smoot, "Choosing and Using Advanced Peltier Modules for Thermoelectric Cooling," *DigiKey*. [Online]. Available: https://www.digikey.com/en/articles/choosing-using-advanced-peltier-modules-thermoelectric-cooling/.

[12] "Hazardous Substance Fact Sheet," *New Jersey Department of Health and Senior Services*. [Online]. Available: https://nj.gov/health/eoh/rtkweb/documents/fs/0239.pdf.

[13] "Bismuth," *Minerals Education Coalition* [Online]. Available: https://mineralseducationcoalition.org/elements/bismuth/.

[14] "Tellurium - The Bright Future of Solar Energy," *USGS*. [Online]. Available: https://pubs.usgs.gov/fs/2014/3077/pdf/fs2014-3077.pdf.

[15] O. US EPA, "Frequent Questions on Lithium-ion Batteries," Sep. 16, 2020. [Online]. Available: https://www.epa.gov/recycle/frequent-questions-lithium-ion-batteries.

[16] OSHA. “Preventing Fire and/or Explosion Injury from Small and Wearable Lithium Battery Powered Devices.” *Occupational Safety and Health Administration*, [Online]. Available: https://www.osha.gov/sites/default/files/publications/shib011819.pdf.

[17] “Embedded C Coding Standard,” May 26, 2016. [Online]. Available: https://barrgroup.com/embedded-systems/books/embedded-c-coding-standard.

[18] “Meet Your Standards,” *IPC International, Inc.*, Sep. 21, 2021. [Online]. Available: https://www.ipc.org/meet-your-standards.

[19] “I2C Bus Specification,” *I2C Info – I2C Bus, Interface and Protocol*. [Online]. Available: https://i2c.info/i2c-bus-specification.

[20] B. C. Kennedy, “Cooker utilizing a peltier device,” US7174720B2, Feb. 13, 2007 [Online]. Available: https://patents.google.com/patent/US7174720B2/en?q=peltier+device&oq=peltier+device.

[21] V. Abramov, “Peltier cooler with integrated electronic device(s),” US20060237730A1, Oct. 26, 2006 [Online]. Available: https://patents.google.com/patent/US20060237730A1/en?q=peltier+cooler&oq=peltier+cooler.

[22] J. T. Welch, “Peltier temperature control system for electronic components,” US7082772B2, Aug. 01, 2006 [Online]. Available: https://patents.google.com/patent/US7082772B2/en?q=peltier&oq=peltier.

[23] “FreeDFM - A Service of Advanced Circuits.” [Online]. Available: https://www.my4pcb.com/net35/FreeDFMNet/FreeDFMHome.aspx.

[24] “Molded, 50 mil Pitch, Dual-In-Line Thin Film Resistor, Precision Automotive, AEC-Q200 Qualified, Networks.” [Online]. Available: https://www.vishay.com/docs/60127/aorn.pdf.

[25] “BP20-12-B1: Digi-key electronics,” *Digikey*. [Online]. Available: https://www.digikey.com/en/products/detail/b-b-battery/BP20-12-B1/1090958.

[26] “CFM-A225B-015-287 Datasheet - Axial Fans | Dc Fans | CUI Devices.” [Online]. Available: https://www.cuidevices.com/product/resource/cfm-120b.pdf.

[27] “Thermoelectric Cooler TEC1-12706.” [Online]. Available: https://peltiermodules.com/peltier.datasheet/TEC1-12706.pdf.

[28] “Sealed rocker switch - ZF switches & Sensors.” [Online]. Available: https://switches-sensors.zf.com/us/wp-content/uploads/sites/7/2012/10/Rocker_KC_Datasheet_08-11-17.pdf.

[29] "120VIN, 25MA, ultra-low IQ, high-PSRR linear regulator." [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MIC5281.pdf.

[30] "BTN8962TA - Infineon Technologies." [Online]. Available: https://www.infineon.com/dgdl/Infineon-BTN8962TA-DS-v01_00-EN.pdf?fileId=db3a30433fa9412f013fbe2d247a7bf5&ack=t.

[31] "High current chokes." [Online]. Available: https://4donline.ihs.com/images/VipMasterIC/IC/BOUR/BOUR-S-A0006777528/BOUR-S-A0006777567-1.pdf?hkey=6D3A4C79FDBF58556ACFDE234799DDF0.

[31] "Schottky Barrier Diode RB225T-40NZ," *RB225T-40NZ : Diodes*. [Online]. Available: https://rohmfs-rohm-com-cn.oss-cn-shanghai.aliyuncs.com/en/products/databook/datasheet/discrete/diode/schottky_barrier/rb225t-40nz-e.pdf.

[32] "Farnell | Electronic Component Distributors." [Online]. Available: https://www.farnell.com/datasheets/2561871.pdf.

[33] "HD44780U (LCD-II), (Dot Matrix Liquid Crystal Display Controller/Driver)," *Sparkfun Electronics*. [Online]. Available: https://www.sparkfun.com/datasheets/LCD/HD44780.pdf.

[34] "TS02-66-43-BK-100-LCR-D datasheet," *CUI Devices*. [Online]. Available: https://www.cuidevices.com/product/resource/ts02.pdf.

[35] "DS Series Detect Switch." [Online]. Available: https://www.ckswitches.com/media/1304/dsdetect.pdf.

[36] "uClamp3301H," *Salesforce*. [Online]. Available: https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/44000000MDWf/s0KJxyOHQyYOcvBXVBK9X3X8Cxj1gNltZ2vd2k1S0Wo.

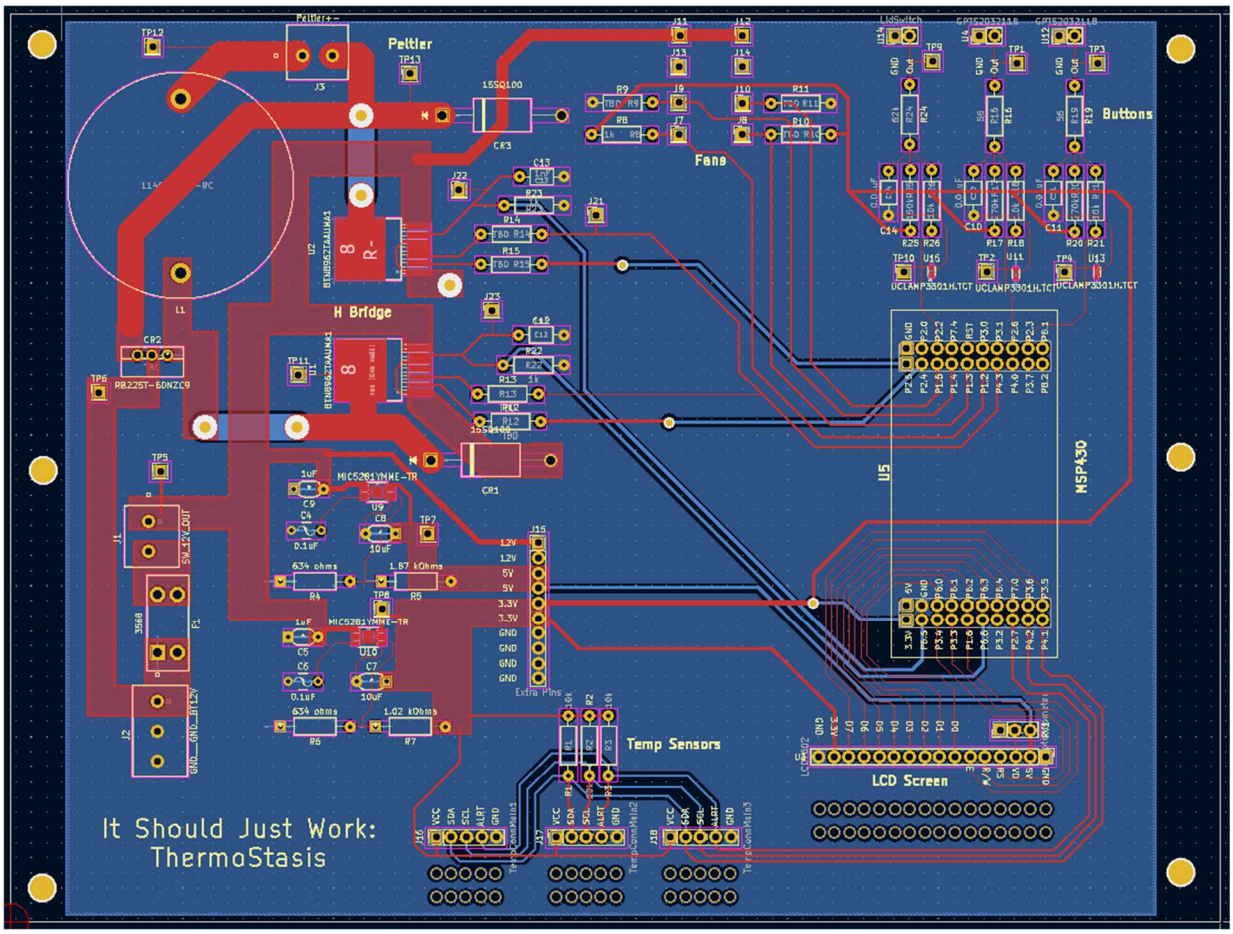
[37] "AT30TSE752A, AT30TSE754A, AT30TSE758A Digital Temperature Sensor with Nonvolatile Registers and Serial EEPROM." [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-8854-DTS-AT30TSE752A-754A-758A-Datasheet.pdf.

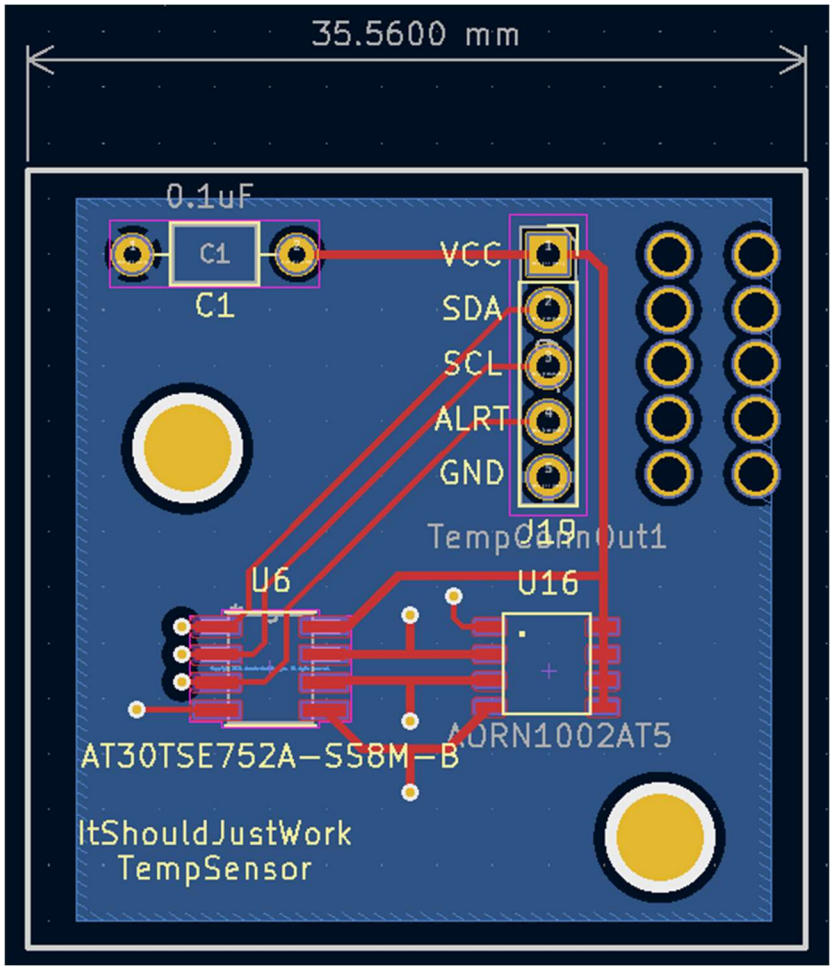
[38] Mamdani, E. H. (1974). *Application of fuzzy algorithms for control of simple dynamic plant*. [Print]. In: Proc .. Inst. Elect. Eng. Contr. · Sci., 121 , 1585-1588.

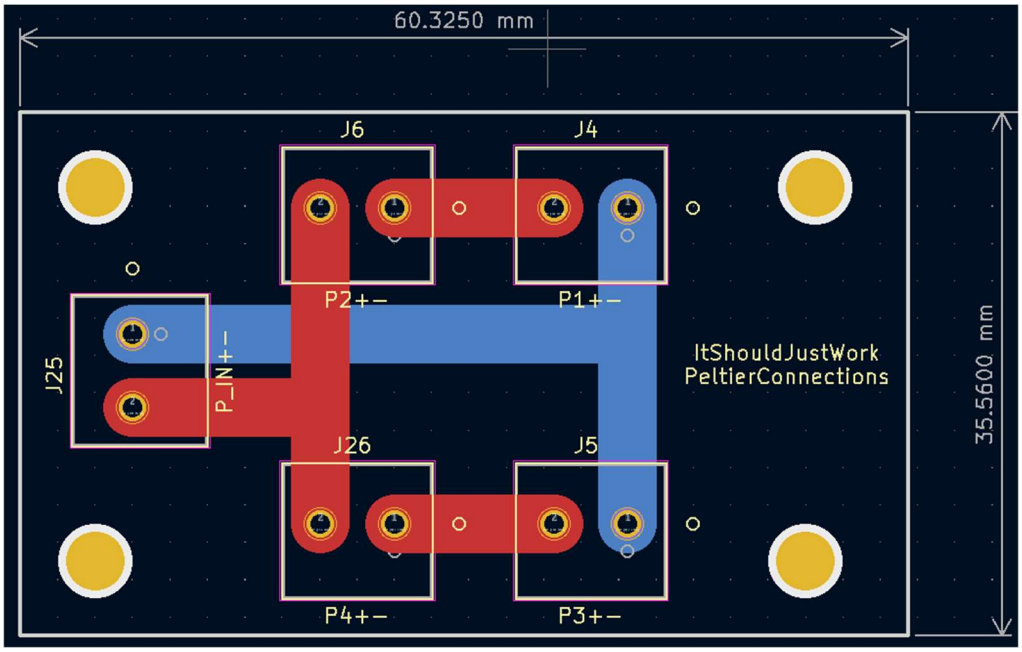
[39] M. M. Gouda, S. Danaher, and C. P. Underwood, in *Fuzzy Logic Control Versus Conventional PID Control for Controlling Indoor Temperature of a Building Space.* [Print]. vol. 33, pp. 249–254.

Appendix

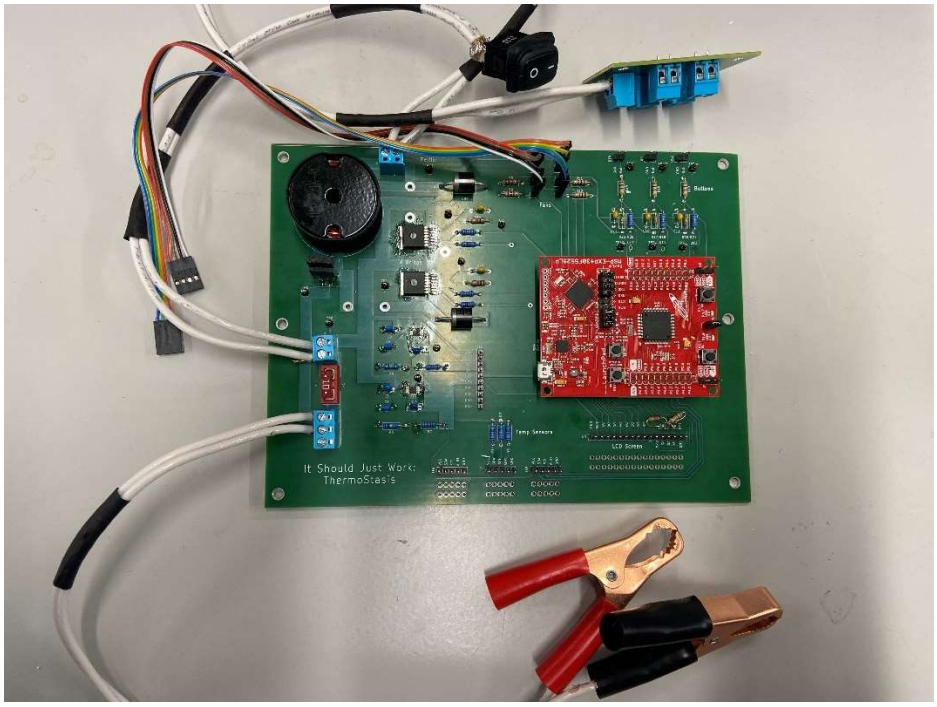
Appendix A: PCB Design

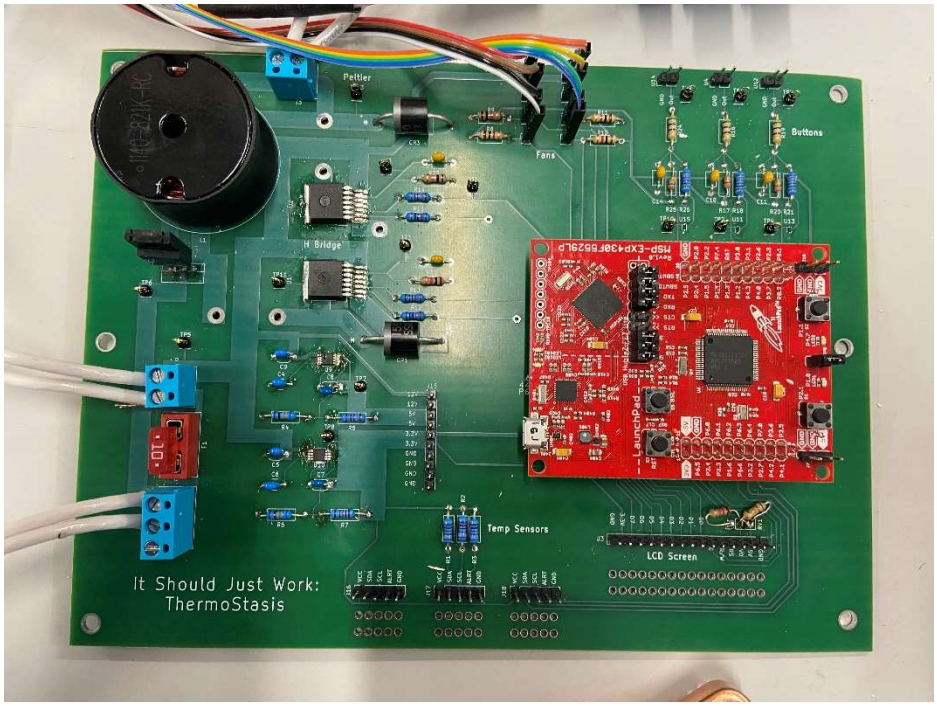


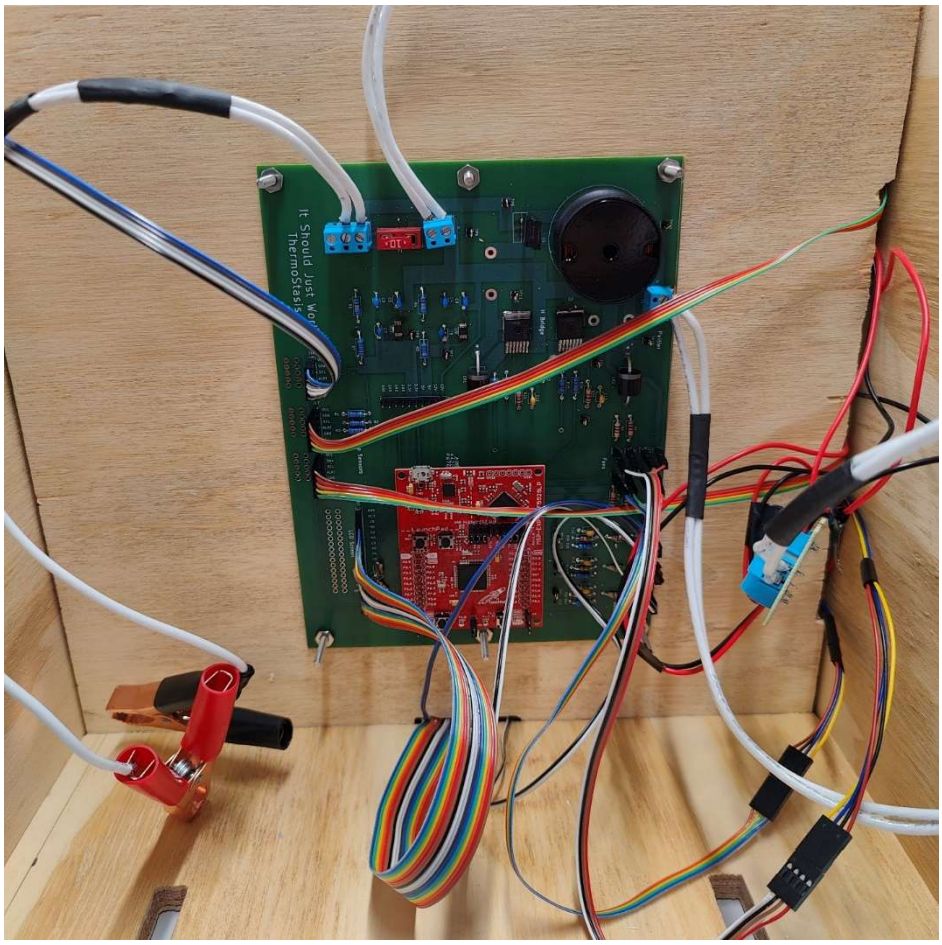

35.5600 mm
0.1uF
C1
C1
VCC
SDA
SCL
ALRT
GND
J19
U6
U16
AT30TSE752A-S8M-B
ItShouldJustWork
TempSensor


60.3250 mm
J6
J4
P2+-
P1+-
J25
P_IN+-
ItShouldJustWork
PeltierConnections
J26
J5
P4+-
P3+-
35.5600 mm

Appendix B: PCB with Components







Appendix C: Thermal Calculations
Required Energy and Power:

1BTU is the amount of energy required to raise 1 pound of water by 1 degree Fahrenheit

Energy (BTU) =T * Weight of Water

1 BTU1 hour=0.29307107 Watts

1 BTU = 0.29307107 Watt Hours

0.5 Liters of Water = 1.10229 pounds of water

T = 42 degrees Fahrenheit

Required Energy = 1.10229 *42 = 46.29618 BTU = 13.56807 Watt Hours

Required Power = Energy (Wh)Time (h)=13.56807 Wh0.5 h=27.136 W

Heat loss can also be included, but that requires a case design. Tentatively, the heat loss for a theoretical case would be around 27.68W, so the Peltier device would need to compensate with additional energy, making Qc = 27.136 + 27.68 = 54.816 Wh

The theoretical container size is also 1080 in3 = 17.698 liters which means that 17.198 liters would just be air. This can also be included to modify the initial Q requirement.

1 BTU = 10.41 Liters Atmosphere

17.198 Liters Atmosphere = 17.19810.41=1.652 BTU per degree Fahrenheit

Required Energy = (1.10229 + 1.652) *42 = 115.682 BTU = 33.9033 Watt Hours

Thermal Power required for 30 minute time frame:

Required Power = Energy (Wh)Time (h)=33.9033 Wh0.5 h=67.807 W

67.807+27.68=95.487W

Thermal Power required for 2 hour time frame:

Required Power = Energy (Wh)Time (h)=33.9033 Wh2 h= 16.952W

Qc = 16.952+27.68=44.632 W

Thermal Power required for 1 hour time frame:

Required Power = Energy (Wh)Time (h)=33.9033 Wh1 h=33.9033 W

Worst case would require Qc = 33.9033 + 27.68 = 61.583W

61.583W / 4 Peltiers = 15.396W per Peltier device

Appendix D: Inductor Calculations

Max Clock Freq = 16MHz

Max PWM Switching time = 10.1 us → 99.01 kHz

Sample PWM freq: f=20kHz

Timer sample period = 160 (when setting this in CCS, remember that it should be n-1 = 159)

Maximum current ripple = 0.05*I_target at the target duty cycle of 25%

Ideally, the average current at a 25% duty cycle should be 25% of the maximum current
Imax = Vmax/R = 12V/2Ohms = 6A

I_target 25% = 1.5A

ΔI = 0.05*I_target 25% = 0.05 *1.5 = 0.075A

ΔI = 0.075A=I_high-I_low=1.5375-1.4625

T_on = Period * 0.25 = (1/f)*0.25 = 1/20kHz*0.25 = 1.25*10^-5 s = 12.5 us

T_off = Period * (1-0.25) = (1f/)*0.75= 3.75 *10^-5 = 37.5 us

V_Load High Current = I*R = 1.575*2 = 3.075V; V_Load Low Current=I*R=1.425*2=2.925V

12=V_inductor+V_Load

V_ind high current=8.925; V_ind low current=9.075

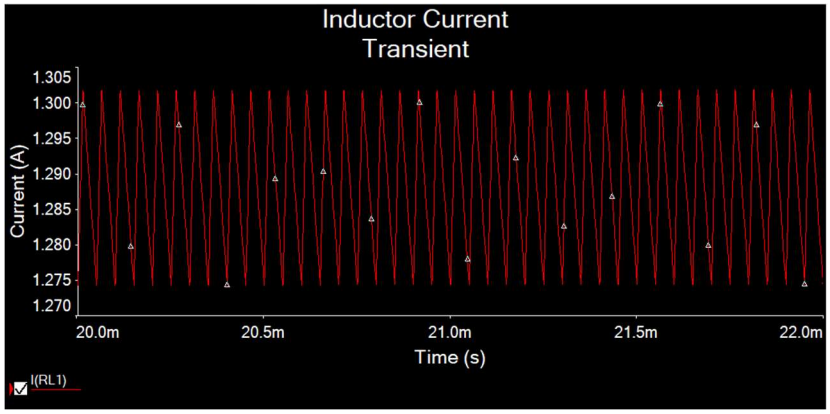
$1.5375 = 1.4625+\frac{1}{L}\int_0^{t_{off}} \frac{(9.075-8.925)}{t_{off}} t+8.925\ dt$

$0.075L = \int_0^{toff} \frac{0.15}{t_{off}} t +8.925\ dt$

0.075L = (0.15/2)t_off + 8.925_toff

L > ((0.075+8.925) * t_off 0)/.075 = (9/0.075)*t_off=120*t_off= 0.0045H = 4.5 mH

L > (1-Duty)(RL))/((Max ripple ratio)(Duty)) * toff=120*t_off= 0.0045H = 4.5 mH



Ripple amplitude of about 0.025A which is less than the expected 0.075A

Attempt with frequency of 50kHz: L > ((1-Duty)(RL))/((Max ripple ratio)(Duty)) * t_off = ((0.75)(2))/((0.05)(0.25)) * (1/50kHz*0.75)120 * t_off= 0.0018H = 1.8 mH

This is not any cheaper than a 7.9mH inductor (tolerance of +50% -30% or ~5.53-11.85mH)

Chokes do not work for this because it reaches a saturation current too soon

A large ripple ratio (rr) would need to used to get an actual inductance value.

At 50kHz and max current of 6A:

rr = 0.10 --> 900 uH

rr = 0.15 --> 600 uH

rr = 0.20 --> 450 uH

rr = 0.25 --> 360 uH

rr = 0.30 --> 300 uH

At 20kHz and max current of 6A:

rr = 0.10 --> 225 0uH

rr = 0.15 --> 1500 uH

rr = 0.20 --> 1125 uH

rr = 0.25 --> 900 uH

rr = 0.30 --> 750 uH

Appendix E: Heatsink Calculations

Qctotal= 27.136 to 61.583W

Qc= 15.396 W

Four Peltier devices so we can split the work. Each device needs to remove around 15 W from the container.

Qmax= 50 W at 25c

dT =35

COP = Qc/Pel

COP 0.5 - from graph on datasheet

Pel=Qc/COP

Pel =15/0.5 = 30

I = 3A - from graph on datasheet

V = Pel/I = 30/3 = 10V

Qh =Qc+Pel = 15+ 30 = 45W

Qh=30W - from graph on datasheet

RthHS = THS/Qh= 10/30 (0.33) or 10/45 (0.22)

R = T / P

R - thermal resistance in c/watt

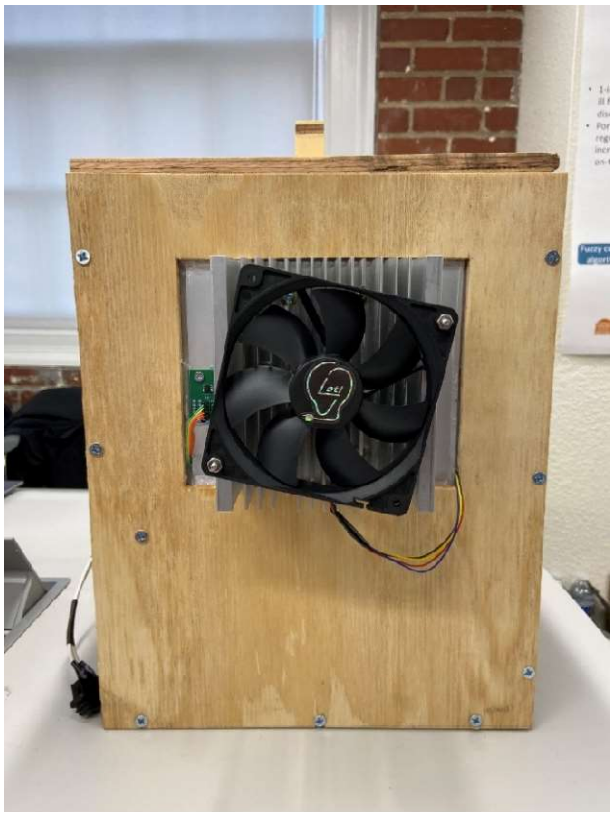
T - temperature difference in C

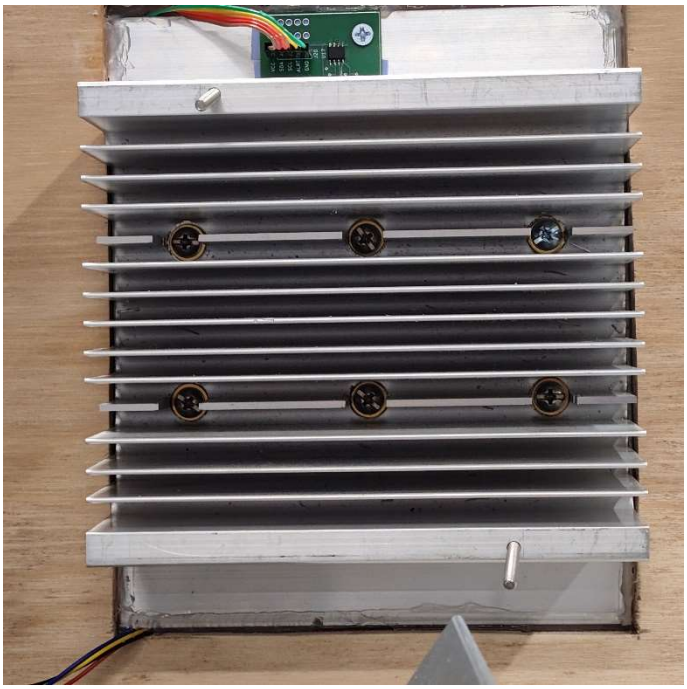
P - power dissipated

R = (30-20) / 30 (or 45) = 0.33 or 0.22 c/watt -- for external heatsink

R = (5-0) / 15=0.33 c/watt -- for internal heatsink

Appendix F: Final Project Images





Appendix G: Project Costs

Category	Item	Cost	Cost per category
MSP	MSP430F5529	$ 18.06	$ 31.35
	MSP430G2553	$ 13.29	
Power	Rocker Switch	$ 3.20	$ 131.23
	Battery	$ 102.74	
	Battery Charger	$ 20.99	
	Clamps	$ 3.49	
	Lid Button	$ 0.81	
UI	LCD Screen	$ 6.94	$ 10.36
	Buttons (GPTS203211B) (x2)	$ 3.42	
PCB	PCB Manufacturing	$ 33.00	$ 73.85
	Diode (UCLAMP3301HCT-ND)	$ 1.92	
	Linear Regulator (MIC5281YMME-TR)	$ 9.65	
	1uF capacitor	$ 0.70	
	0.1uF capacitor	$ 0.70	
	10uF capacitor	$ 1.00	
	634 Ohm resistor	$ 0.20	
	1.87kOhm resistor	$ 0.10	
	1.02kOhm resistor	$ 0.10	
	Schottky Diode (3 pin)	$ 2.98	
	10A Fuse	$ 1.16	
	820 uH Inductor	$ 12.00	
	Screw Terminal Block (3 ports)	$ 0.86	
	Screw Terminal Block (2 ports) (10 ct)	$ 2.38	
	PCB fuse socket	$ 1.33	
	Schottky Diode (singles) (x4)	$ 3.28	
	Wire Crimp	$ 2.49	

Thermo-related Components	Peltier plates (x4)	$ 36.00	$ 214.70
	Heatsinks (x2)	$ 94.20	
	Aluminum Bar	$ 27.70	
	Gray Thermal Pad	$ 1.14	
	blue thermopads	$ 9.47	
	Fan (x2)	$ 29.54	
	Temperature sensor (x3)	$ 4.89	
	Resistor Networks	$ 11.76	
Assembly	beleville washers	$ 7.96	$ 153.16
	Thermal Paste	$ 31.56	
	Plywood	$ 29.86	
	Styrofoam	$ 10.98	
	Silicone Sealant	$ 8.40	
	Nylon Shoulder Washers	$ 21.15	
	#10S Brass Flat Washers (8 ct) (x2)	$ 3.16	
	#10-24 Hex Nuts (20 ct)	$ 1.38	
	10-24, 2" Screws	$ 5.59	
	#6 32 Hex Nut (x20)	$ 2.78	
	#6-32, 1/2", Flat Phillips with Nuts, (14 ct)	$ 1.38	
	#10, 1", wood screws (100 ct)	$ 5.48	
	12" copper wires	$ 13.38	
	#4-40, 1/2" machine screws (14 ct)	$ 1.38	
	#6-32, 1.5", machine screws (10 ct)	$ 2.78	
	Tax	$ 5.94	
Miscellaneous	Mouser Shipping	$ 7.99	$ 7.99
Total			$ 622.64
Amount Over Budget			$ 122.64