Pantastic

A Technical Report submitted to the Department of Computer Engineering

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Tyler Hendricks

Spring, 2022

Technical Project Team Members
Andrew Tam
Kai Wong
Thomas Forrester
Noal Zyglowitz

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Harry C. Powell, Department of Electrical and Computer Engineering

Pantastic - 5 Guys One Capstone Project

Andrew Tam, Kai Wong, Noal Zyglowicz, Thomas Forrester, Tyler Hendricks

December 17, 2021

Capstone Design ECE 4440 / ECE4991

Signatures

Statement of work:

Andrew Tam

My contributions to the project have been focused primarily on hardware. In the beginning of the project, I researched and selected the microcontroller for its in-built Wi-Fi support and capabilities. I assisted in the selection and decision of the components that were used on the PCB. These components include the solder mount device (SMD) components for the 1206 type package, phoenix connectors, header pins, LEDs, and IR sensor. In addition, I assisted in testing the Wi-Fi connection between the microcontroller, network, and communicating device using a mobile hotspot as an access point.

After we received the boards, I took it to be soldered at WWW Electronics Inc for the SMD package 1206 components. Afterwards, I soldered on the remaining components which consisted of the tester pins, the array of LEDs, a 1N4001 diode, IR sensor, and the phoenix connectors. Additionally, I handled most of the wiring configuration for ease of testing for components including the laser button, alarm silencer button, phoenix connectors, and the laser diode.

Besides the physical electrical hardware, I designed the casing CAD files that would be 3D printed. While designing the case, I took component sizing and wire management into consideration to reduce the bulk size of the project. Additionally, I designed a window for the array of LEDs and IR sensor as well as the slots for the buttons and laser diode.

Kai Wong

Throughout the semester, the core of what I worked on was the user interface (UI). Once a bare bones idea of our project was created, it was my job to find out how we wanted our device to interact with the users. Through much discussion with the team about what we wanted, it was determined that we wanted some sort of audio and visual indicator (piezo buzzer and LED array) for temperature and warnings. Along with this, I chose the two buttons to use; a black button for the laser and a red button to turn off the alarm. After we had the UI components chosen, I needed to develop hardware drivers for these external components to ensure stability. Once baseline driver circuits and components were created, I worked with Thomas Forrester to create a final schematic with all the components and drivers as well as a baseline PCB for testing.

Since we weren't entirely confident in the driver circuits due to a wide operating range and lack of detailed plots, specifically with the laser diode, I took a lead in the testing of the components to develop the correct BJT driver orientation for each of the UI components. This included the use of many variable resistors, voltage dividers, capacitors and diodes to ensure the components worked at a safe operating voltage and current as well as not pulling significant power from either the board or battery. I then altered the schematics in Multisim with Thomas to create an updated schematic and PCB for our final board send out.

Noal Zyglowicz

I primarily worked on the web application and network functionality. I created a web interface using React.js for the frontend framework and Express.js for the backend framework. I previously had two years of experience in both classes based and functional react. I coded all the components and state management. Lots of time was spent making the website responsive and handling updates efficiently. I used handlers to handle the functions of the buttons and found a chart api in chart.js to provide a good visual view of the data. I customize the website visual using css and html. I had no previous experience in

Express.js so a small learning curve was present in setting up the post and get apis as well as in parsing the data from incoming requests. I worked through the debugging process with the network code being received by the backend.

I developed the network functionality with code written for the CC3220s provided in the simplelink sdk. I set up an environment in the code composer studio and downloaded the httpget application for our board. I spent time learning how to send data from the board to a localhost domain. A lot of time was spent reviewing the correct relevant network protocols needed for the process and the URI needed to access a localhost network. It took many attempts to make the environment work. UART debugging software was very finicky and hard to run. Once the network code was merged by Tyler into a single code composer studio project, I spent a lot of time figuring out how to send multiple http requests given the simplelink apis and our driver code. I figured out how to convert a c double data type into a string literal and send it as part of a post body which needed a char pointer. Once this was functional, lots of time was spent debugging why the network connection would shut down after a short period of time or a given number of requests sent. With many attempts and word of advice from Kai who had some global knowledge from taking Operating Systems, I was able to find a way to clear up memory created from excess threads and get our device to repeatedly send sensor data to the web server application I created.

Thomas Forrester

I worked on several portions of our project. The first task I worked on was selecting a battery to power Pantastic. This included calculating the maximum voltage and current needed for each component in our device and selecting a battery that would be appropriate for our needs. Next, I helped design the circuitry. Specifically, I worked with Kai Wong to design the circuitry for the LED drivers and piezo buzzer and reviewed Tyler's circuits for the IR sensor and push buttons. In addition to designing the circuitry, I was in charge of finalizing our Multisim schematic. For our Multisim schematic, I reviewed all of our circuitry to make sure there were no mistakes in our final design. I reviewed all of the connections, made sure test points were placed in the correct locations, and mapped pins to get our design ready to forward annotate to Ultiboard.

Next, I was in charge of keeping track of the budget for our project. I ordered all of our parts and kept track of how much money we had left to spend. When our parts arrived, I took inventory to make sure we were not missing any components and organized our parts. Next, my main responsibility for the project was designing our printed circuit board (PCB). For this task, I reviewed the data sheet for our components to find dimensions and designed all of the custom component footprints in our Ultiboard database. After designing our components, I updated the packages in our Multisim schematic and forward annotated our design to Ultiboard to begin routing the traces. Finally, when we received our PCB, I helped test all of the circuitry and fix any issues we had.

Tyler Hendricks

I had several responsibilities for this project. First, I performed research to determine which IR sensor would work best with our project and determined which communication protocol would work best with the Launchpad. Next, I designed the circuits for the IR sensor and its voltage regulator, the two push buttons, and the final piezo circuit that was used to increase the volume of the alarm.

Next, the majority of my time was spent developing the embedded driver code. This involved designing the FSM in yED and coding the FSM as well as writing the driver code in Code Composer Studio for the: buttons, IR sensor, piezo, laser diode, and LED matrix. I also adapted an online version of I2C bit bang code to use with the IR sensor because the TI I2C code would not work for the sensor. Next, because I own a 3D printer, I was in charge of printing the printed parts of the project.

Finally, I tested the circuitry and driver code of the launchpad. The circuitry was tested using a voltmeter to ensure voltage was correct at the required points across the board. The driver code was tested by examining signals using an Analog Discovery 2 and Crockpot to ensure correct behavior when the device detects heat.

Table of Contents

Contents

Capstone Design ECE 4440 / ECE4991 1
Signatures 1
Statement of work: 2
Table of Contents 5
Table of Figures 6
Table of Tables 6
Abstract 8
Background 8
Constraints 9
- Design Constraints 9
- External Standards 10
- Tools Employed 11
- Ethical, Social, and Economic Concerns 15
- Environmental Impact 15
- Sustainability 15
- Health and Safety 16
- Intellectual Property Issues 16

Detailed Technical Description of Project 16
- Hardware 17
- Firmware 28
- Software 34

Project Timeline 34
Test Plan 36
Final Results 39
Costs 42
Future Work 43
References 45
Appendix 49
- Appendix A: Main and Test Programs Differences 49
- Appendix B: Bill of Materials 50
- Appendix C: Project Costs 51
- Appendix D: Images of Pantastic 53

Table of Figures

Figure 1: Web UI Interface 13
Figure 2: PuTTy Debugger 14
Figure 3: Postman API Tool 14
Figure 4: Pantastic Web Application Code........ 15
Figure 5: Simple Block Diagram of Pantastic 17
Figure 6: Full Block Schematic of Pantastic........ 18
Figure 7: FOV Chart 19
Figure 8: Preliminary LED Driver Circuit, Modified to the Final LED Driver Circuit 21
Figure 9: Zoomed out LED Block Containing the Driver Array with Attached LEDs........ 21
Figure 10: Push Button Subcircuit Schematic for both Laser Diode and Alarm........ 22
Figure 11: Alarm Button and Laser Button as Seen on the Main Schematic 23
Figure 12: Laser Diode Driver Schematic 24
Figure 13: Laser Diode Driver Block Diagram for Final Schematic........ 24
Figure 14: New Piezo Driver 25
Figure 15: Simplified Piezo Driver........ 25
Figure 16: Final Board PCB Layout 26
Figure 17: Final Board PCB Layout Reference Designators........ 26
Figure 18 Algorithmic Flowchart 28
Figure 19: Interrupts Flowchart 29
Figure 20: Pantastic Embedded Code 31
Figure 21: Rendered CAD Image of Casing........ 33
Figure 22: Rendered CAD Image of Mount System 33
Figure 23: Original Gantt Chart........ 35
Figure 24: Final Gantt Chart 36
Figure 25: Push Button Verification 37
Figure 26: Top View of Pantastic 39
Figure 27: Bottom View of Pantastic........ 39
Figure 28: Pantastic Mounted on Wall in Case 53
Figure 29: Pantastic in Case Side View 53
Figure 30: Pantastic in Case Top View (1)........ 54
Figure 31: Pantastic in Case Top View (2)........ 54
Figure 32: Pantastic in Case Front View 55
Figure 33: Pantastic In Case Top View Exposed........ 55
Figure 34: Pantastic Test PCB 56

Table of Tables

Table 1: Grading Template 41
Table 2: Points to Letter Grade 41

Table 3: Pantastic Cost 42
Table A1: Main vs. Test Program 49
Table B1: Digi-Key Bill of Materials for Single Unit of Pantastic 50
Table B2: Digi-Key Bill of Materials Per Unit of Pantastic for 10,000 Units Manufactured ...50
Table C1: First Part Order and Budget 51
Table C2: Second Part Order and Budget 51
Table C3: Third Part Order and Budget 51
Table C4: Bonus Part Order and Budget 52
Table C5: Extra Costs and Budget 52

Abstract

Pantastic is a kitchen safety device that uses temperature sensing technology to measure the ambient temperature of kitchen appliances. In the event that kitchen appliances are left on accidentally, the device will issue an alarm to notify the user and surrounding agents that the appliance has reached an unsafe temperature for an extended period of time. The device is wall mountable with a flexible case to allow the device to be mounted at various angles. The temperature sensing technology is achieved using a non-contact infrared (IR) temperature sensor controlled by a microcontroller. The device uses a piezo buzzer and an LED array to provide sensory alert signals to the user if their device is at an unsafe temperature. Pantastic integrates well with other smart home IoT devices that serve to protect the wellbeing of homeowners and households.

Background

This project was chosen for two main reasons. First, in the 21st century, many people, especially college students, live a busy lifestyle. Often individuals will multitask cooking and other activities. It is easy to become distracted and forget that something is running on the stove. Pantastic is the device that solves this issue by remotely sensing stove temperature and allowing the user to view the information on their device from another location. The second serious danger of cooking is stoves left on while unattended. This is a waste of energy and a dangerous fire hazard which can be mitigated with Pantastic.

We were not able to find any projects that were identical to the Pantastic but there are two products that follow a similar premise. The first is IR non-contact digital thermometers which measure temperature with an IR sensor [1]. The next similar project is the BurnerAlert which produces sound and light when the stove is on to prevent the user from forgetting to turn off the stove. This product works by attaching directly to the burner knob and detecting when the knob is turned [2].

Our project is different from these similar projects in two main ways. First, the IR temperature sensor is different from the non-contact thermometer because it is a stationary mounted device that continuously measures the temperature instead of measuring it in a single scan by the user. Secon, it is similar to the BurnerAlert in the sense that it can give light and sound alerts if the stove is left on, but it will not do this continuously like the BurnerAlert, and it will detect if the stove is on via temperature and not by the burner knob position. There are a few classes that helped tremendously with providing the knowledge and skills needed to complete this project.

First, content from EAR 1 and EAR 2 played directly into this project because we developed the code in Code Composer Studio [3] to run on a TI Launchpad device (CC3220S-LAUNCHXL) [4] which we learned how to do in the EAR series. Next, EAR 2 was useful because we learned how to generate interrupts for real time responses based on signals on certain pins. Second, content from the FUN series was applicable for several reasons. First, as part of the FUN series, we learned how to develop custom PCBs. Finally, FUN also helped because for Pantastic, we had to design the circuit in a manner that was power efficient to prevent running down the battery too quickly. Third, content from CS3240 was utilized as we had to do some minor web development to create an online interface for the user to connect to. Finally, content from ENGR 1610 was utilized because of the basic 3D printing and CAD skills showcased in part of the course.

Constraints

Design Constraints

Two Major related constraints were imposed on the project because the team had members from both Computer and Electrical Engineering majors. As such, it was required that the project have a custom designed printed circuit board (PCB) and use a microcontroller.

Microcontroller Limitations

The microcontroller chosen for the project was a Texas Instruments CC3220S Launchpad [4]. This microcontroller was chosen because of its onboard WiFi capabilities and enough GPIO pins to handle all the peripherals for the project. The board is constrained by its 80 MHz clock speed. This speed was fast enough to not affect the project. The microcontroller is also constrained by which pins can be used. Designers are disallowed from accessing over a quarter of the pins on the header which affected which header pins were used.

Another microcontroller limitation was the I2C bus required to communicate with the MLX90614 IR [5] sensor. The sensor communicates using SMBus and would timeout if the clock was low for more than 27 milliseconds or high for more than 45 microseconds.

Software Limitations

Circuit and PCB designs were done in National Instruments Multisim [6] and Ultiboard [7] respectively due to this software being provided to UVA ECE students and taught in ECE courses. Embedded software development was done in Code Composer Studio [3]. Design for 3D printed parts was modeled in AutoDesk Inventor [8] and sliced in Cura Ultimaker [9].

Manufacturing Limitations

The device will be relatively easy to manufacture. The system will require a printed circuit board (PCB) and a plastic case to protect the microcontroller.

The PCB was constrained by the requirements of the manufacturer, Advanced Circuits, and by the size of the case. The PCB needed to be small enough to avoid the need for a large case which would waste plastic and take a long time to print. Next, the following requirements had to be followed in order for the PCB order to be filled.

- Thickness: 62 mil
- Core: 39 mil
- Max size: 30 square inches
- Minimum 5 mil line per space

- Minimum 10 mil drill hole size
- Maximum 50 drilled holes per square inch

The case will be limited by the abilities of the Anycubic Mega Pro [10] used to print the parts. This includes materials, size, resolution, and print time. Some of these constraints are set because the printer is tuned to succeed at those specifications. The printer specifications are as follows:

- Material: PLA
- Speed: 40 mm/s
- Infil: Greater than 9%, less than 20% (15% used)
- Layer Height: Greater than 0.1 mm, less than 0.4 mm (0.3 mm used)
- Nozzle Temperature: 198℃
- Bed Temperature: 60℃

External Standards

These External Standards are followed by the capstone project

1. *FCC Regulations* - The TI CC3220S-LAUNCHXL complies with the FCC as an unlicensed intentional radiator, described as "a device that intentionally generates and emits radio frequency energy by radiation or induction that may be operated without an individual license" [11].
2. The TI CC3220S is 802.11 standard compliant security support [12]. Additionally, the TI CC3220S has been certified by the WiFi Alliance under the programs for WPA/WPA2/WPA3 Personal and Enterprise along with Wi-Fi Certified n/b/g.[13]
3. The PCB design adheres to the IPC standards IPC-2221A [14]. The criteria met for the designed board meets IPC-A-600J, which includes board edges, material, through holes, solder mask [15].
4. *SMD (Surface Mount Device) Standards* - For the PCB design, certain components were selected that meet SMT (Surface Mount Technology) industry standards. This allowed for a reduction in PCB sizing and a more convenient form factor. The majority of these components were 1206 package types for passive rectangular components which conforms to the JEDEC (Joint Electron Device Engineering Council) Specifications [16].
5. *UART (Universal Asynchronous Receiver-Transmitter)* - UART is utilized for serial communication implementation between devices. UART is implemented for the TI CC3220S-LAUNCHXL to send data and communicate to a computer terminal [17].
6. *SPI (Serial Peripheral Interface) Communication Protocol* - The TI CC3220S utilizes SPI, an interface between microcontroller and peripheral IC. SPI was utilized for sensor

data collection between the TI CC3220S-LAUNCHXL and the Melexis MLX90614 IR temperature sensor [18].

7. *I2C (Inter-Integrated Circuit)* - I2C communication protocol was employed for the TI CC3220S-LAUNCHXL for synchronous communication between the microcontroller and the MLX90614 IR sensor [19].
8. *IEEE 1149.1 JTAG Standard* - The JTAG (Joint Test Action Group) is standard under IEEE 1149.1 that defines a TAP(test access point) and boundary scan architecture for digital integrated circuits and provides a standardized serial interface to control the associated test logic [20].
9. *RoHS (Restriction of Hazardous Substances) Compliant* - The TI CC3220S-LAUNCHXL conforms with the legislation RoHS, a EU standard, that restricts the usage of specific hazardous materials that are found in electrical and electronic products [21].
10. *RED (Radio Equipment Directive) Compliant* - the TI CC3220S-LAUNCHXL conforms with the legislation RED, a EU standard, that provides the basis for further regulation governing the protection of privacy, personal data, interoperability, access to emergency services, compliance with the combination of radio equipment and software, and against fraud. [21]

Tools Employed

Hardware

Multisim [6] was used for circuit design, and Ultiboard [7] was used for PCB design. Both tools have been used in past courses so a large amount of usage information was already possessed by the team. A new skill that needed to be learned was how to add custom components to the database and specify the dimension of the item for PCB layout.

The circuitry was tested using a Craftsman Digital Multimeter [22] to perform continuity testing and voltage testing. Tyler Hendricks had experience with this tool in the past, so no new skills were needed. Next, signals for the I2C bus to read data from the I2C were tested using an Analog Discovery 2 [23] logic analyzer and oscilloscope. All team members had used this tool before; however, this project gave the opportunity to learn how to use the logic analyzer to parse I2C signals. Finally, Waveforms [24] was used as the software to view data from the Analog Discovery 2. Once again, the team had used this tool in the past in the FUN series.

Firmware

Code Composer Studio [3] was used to develop the embedded code for the project. This tool had been used by all team members in the Embedded Computing and Robotics course series, so many of the basics were known. The CC3220S Launchpad libraries cannot be compiled without being used in example code with the TI SysConfig tool, so Noal Zyglowicz and Tyler Hendricks had to learn how to use the SysConfig. SysConfig allows for a GUI to be used to configure module settings (GPIO, Timer, etc.) instead of finding the memory address of each pin to be used and then configuring that address.

Testing and development of the I2C code was done using an Arduino [25] and the Arduino IDE [26]. Tyler Hendricks had past experience with this, so no new skills were needed for these tools. The code was later transferred to Code Composer Studio for use on the Launchpad.

Software

The web application was written in JavaScript using ReactJs. ReactJs is a flexible JavaScript library that is utilized for building web user interfaces [27]. It was selected for its isolated component blocks for ease of use and development. Additionally, ReactJs employs a Virtual DOM (Document Object Model) [28] that dynamically removes or adds the data at the back end, and when modifications to the code are done, immediately sends the updates to the browser each time the web page is loaded. This feature aided in smoother development for a user interface. The web application was maintained using Git [29] and GitHub [30]. This made collaboration simple between the software developers and allowed us to have a place to store code for easy accessibility. Finally, yED [31] was used to generate the graphic charts which provide a visual representation of the firmware.

React.js was used for the front end of the web application. It is a highly popular and well documented modern web framework maintained by Facebook and is used by thousands of companies worldwide [32]. It is lightweight, easy to learn, and helps make your web UI very efficient. React components, html, static images, and pure css helped to comprise the visual makeup of the UI. Additionally, chart.js was used to boost the visual appearance of the data represented. Chart.js allows for easily customizable visual elements and smoothly updated data when new data is received [33]. UseState was used to manage the data used by the client. State data that needed to be tracked included temp data, data unit, current color of the chart, current length of the data, and current temperature being represented. Useinterval was set up to poll the data from the backend via axios calls every few seconds. Polling serves as a great simple solution for updating the data with calls from the client. More robust techniques such as webhooks and websockets would be a more efficient system as they keep http connections open for two-way transmission, but would take a lot of time and effort to set up. It would not be practical to code a websocket connection for a capstone project when the same functionality could be achieved with polling and would cause no visible speed or resource disadvantage for the capstone demo given the amount of data.

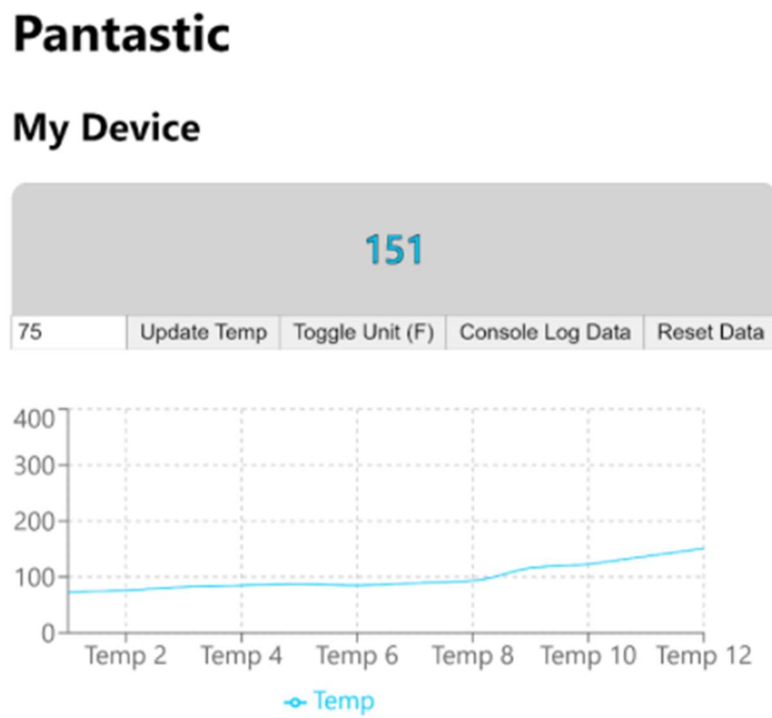


Figure 1: Web UI Interface

Express.js was chosen for the backend because it is also very popular and well documented. It has the additional bonus of being a javascript framework. Golang was almost chosen for the backend due to its high efficiency, but writing the code in one language made it easier for team collaboration when learning to write the code. App.post handlers were used as part of the express api to process incoming requests from the microcontroller to the server. Express body-parser was used to parse the string literal sent in the body of the request and then update the server data. App.get handlers were used to outline the functionality needed by buttons, as well as the polling feature the client uses to refresh the data.

PuTTy was used to debug the UART output from the microcontroller when response data was received by the microcontroller after the request was handled by the server [34]. The tool helped to debug calls when data was not sent in the right format or when it was not clear when the request was sent and received.

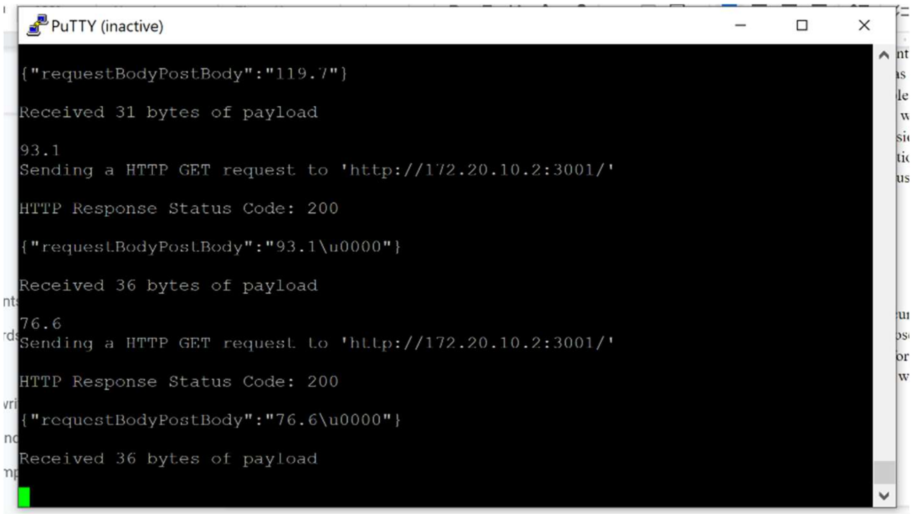


Figure 2: PuTTy Debugger

Postman was used as an API tool to ping the backend of the server for testing purposes. Postman was helpful because it can send http requests much faster than the microcontroller [35].

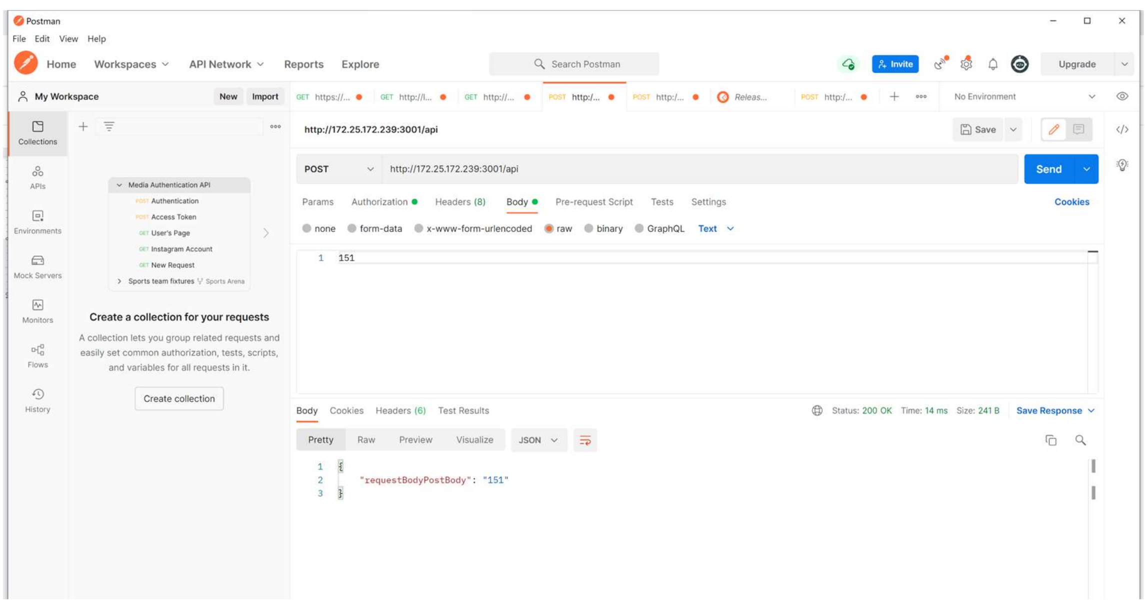


Figure 3: Postman API Tool

The code for this portion of the project can be found in the file *PantasticWebAppCode.zip* in the 5 Guys one Capstone Team Resources folder on Collab.

Figure 4: Pantastic Web Application Code

Ethical, Social, and Economic Concerns

Economic and Cost Constraints

The project is constrained by the set budget of $500. The team did not exceed the budget purchasing initial parts for the unit, so several backup parts were bought. This was done because of the ongoing parts shortage [36] and fears of not being able to get required replacement parts in the event of a broken part. The cost also limited what IR sensor could be purchased. There were several high-quality sensors that were passed on due to the cost constraints.

Environmental Impact

Due to the inherent electrical nature of our product, battery disposal will always be a major concern. In addition to batteries, the PCB may also have a negative impact on the environment if not recycled properly at its end of life. However, while using the product, there should not be any significant byproduct (besides batteries) that could possibly cause a significant environmental impact.

During manufacturing of the device's case, there will be some plastic waste generated. This will be mitigated by using tuned printer settings that will maximize the chance of a successful print and minimize the chance of generating a large piece of plastic waste.

Sustainability

Two major considerations for sustainability will be battery usage and waste as well as plastic usage and waste. These will be mitigated through practices to reduce resource consumption. This will be done by using efficient electrical designs to reduce battery usage. This will also be done by using the aforementioned tuned printer settings to reduce plastic waste. Finally, the printer will also be set to print at or less than 15% to reduce overall plastic use in the project.

Health and Safety

Pantastic is a safety device, but it is not a lifesaving device. Pantastic works in a similar capacity to a smoke alarm which senses a potential danger but is not equipped with the means to stop that danger. Humans are required to assess the danger or leave the handling of the situation to a professional with proper equipment if the situation becomes too severe. That being said, Pantastic is generally limited to the health and safety concerns of a normal IoT device. The device should be easy to attach to walls securely and safely. It should also not pose a risk of electric shock due to being battery powered.

The first ethical issue stems from the user misunderstanding their role in exiting-the-loop of supervision of appliances in the household kitchen. Pantastic is not a cooking companion which instructs the user when to remove or modify the state of food or cookware, but rather a safety device that serves as an extra backup when humans reveal their forgetful habits. The second ethical issue comes from the intrusion of IoT and bluetooth devices into the modern home.

Intellectual Property Issues

The device described in [37], is a contactless infrared thermometer. However, this is the only similarity to Pantastic. Both devices have an onboard IR sensor to measure temperature in a contactless manner. The device in [37] uses pulse width modulation (PWM) to record the temperature value, while Pantastic uses I2C to read the temperature value. The device in [37] also serves as a thermometer for the purpose of taking a human's temperature and not for recording the temperature of food or cooking appliances. Since the devices serve completely different purposes, this will not affect the ability of Pantastic to be patented. Further, the device in [37] was patented in Germany and the patent has since expired.

The iGuardStove described in [38] serves as a kitchen safety device. The device uses heat and motion detection to detect if a stove has been carelessly left on and has the capability to shut off the stove. This device could prevent the Pantastic from being patented because of the significant similarities and functions. Both devices use the MLX90614 IR temperature sensors, send the data over the network, use alarms to alert the user, and serve as kitchen safety devices. The benefit of the Pantastic is the significantly reduced price. Fortunately, the patent for the iGuardStove was abandoned on July 31, 2019 for failure to respond to office action.

The Stove Guard described in [39] also serves as a kitchen safety device. The device also uses an IR temperature sensor to detect the heat. It also has an onboard alarm that is capable of sounding if the temperature increases rapidly or high heat is detected. This similarity to Pantastic could potentially cause intellectual property issues if Pantastic were to be pantended. There is a chance that the visual indicators and the WiFi client app could make Pantastic different enough in function to allow it to be patented. Currently, the Stove Guard is still patent pending.

Detailed Technical Description of Project

The goal of this project was to develop an IoT device that measured the temperature of an object on a cooktop in a contactless manner, and notifies users when the object has been at a high temperature for an extended period of time. Temperature data is collected via the contactless IR sensor, and Wifi is used to send the data to a server which updates the user application with the current temperature. A laser diode is provided to help the user aim the IR sensor at the desired location. In addition to the graphical user interface on the web application, the device has an LED heat matrix to give the user a visual approximate the current temperature. Finally, an alarm is provided to audibly alert the user if a high temperature has been detected for a long period of time (one hour). A high-level diagram of the system is shown in Figure 5 and the full block schematic can be seen in Figure 6**.**

Hardware

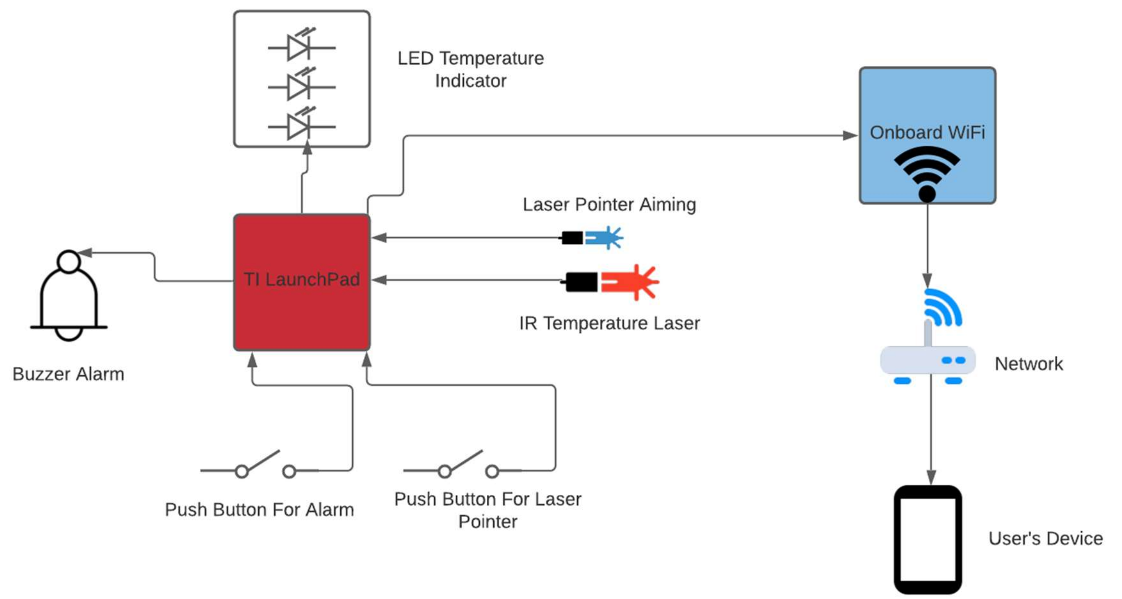


Figure 5: Simple Block Diagram of Pantastic

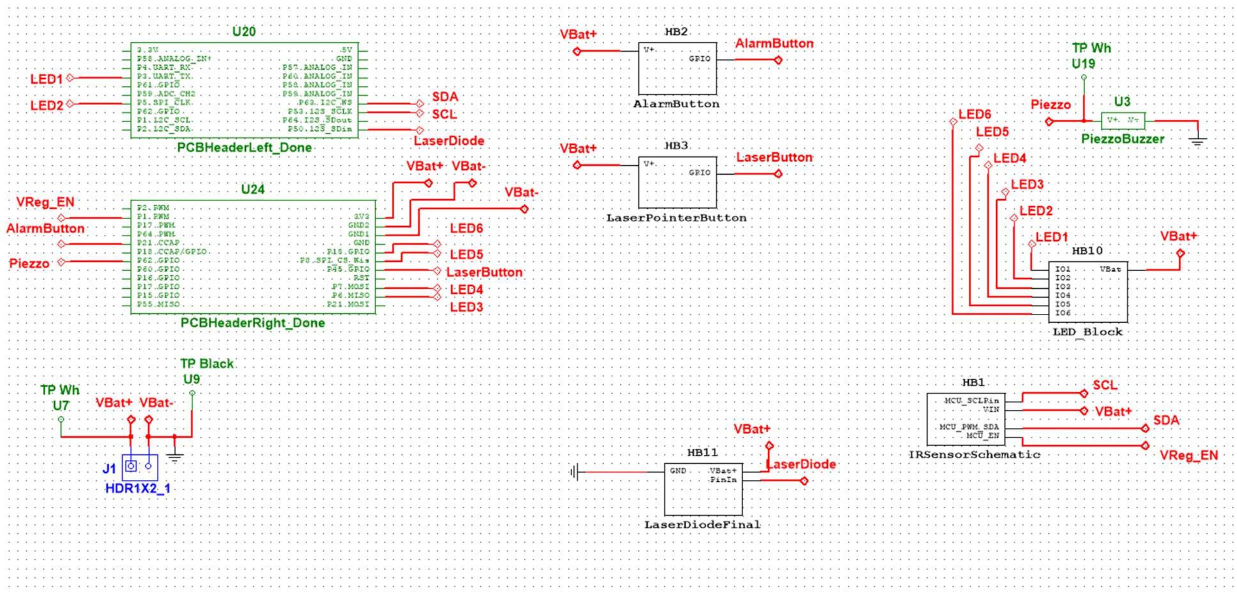


Figure 6: Full Block Schematic of Pantastic

Battery

The battery used in this project was the TL-2100/S 3.6V AA lithium battery [40]. The original design included a 9V battery, but it was recommended by Professor Powell to use a lower voltage battery. This battery was chosen because the maximum voltage required for our design was 3.3V. In addition to meeting the minimum voltage requirements, the power capacity and energy density met expectations. The calculation for these values can be seen below. The values used in these calculations came from the datasheet of the TL-2100/S [40].

$$Power\ Capacity = 2.1Ah * 3.6V$$

$$Power\ Capacity = 7.56\,Wh$$

$$Energy\ Density = \frac{7.56\,Wh}{0.0176\,kg}$$

$$Energy\ Density = 429.5\,\frac{Wh}{kg}$$

Finally, when searching for a battery, one of our main priorities was finding a battery that was inexpensive and met the above requirements. The TL-2100/S was chosen because it was a relatively inexpensive option compared to similar 3.6V batteries that were originally considered.

IR Sensor

The IR sensor used in this project was a Melexis MLX90614 IR temperature sensor [5]. This sensor was chosen because it combined high accuracy with a "low" price. Low is relative because the project required a sensor with a 5-degree field of view (FOV) and the MLX sensor was the cheapest option that could be found. The 5-degree FOV requirement will be discussed next.

The diameter of the zone of measurement can be computed with the following formula [41], where s is the zone of measurement, D is the distance from the source, and FOV is the field of view in degrees.

$$s = 2 * D * \tan\left(\frac{FOV}{2}\right)$$

This formula was used to generate a chart for various zones of measurements based on various values of D and FOV plotted in Figure 7.

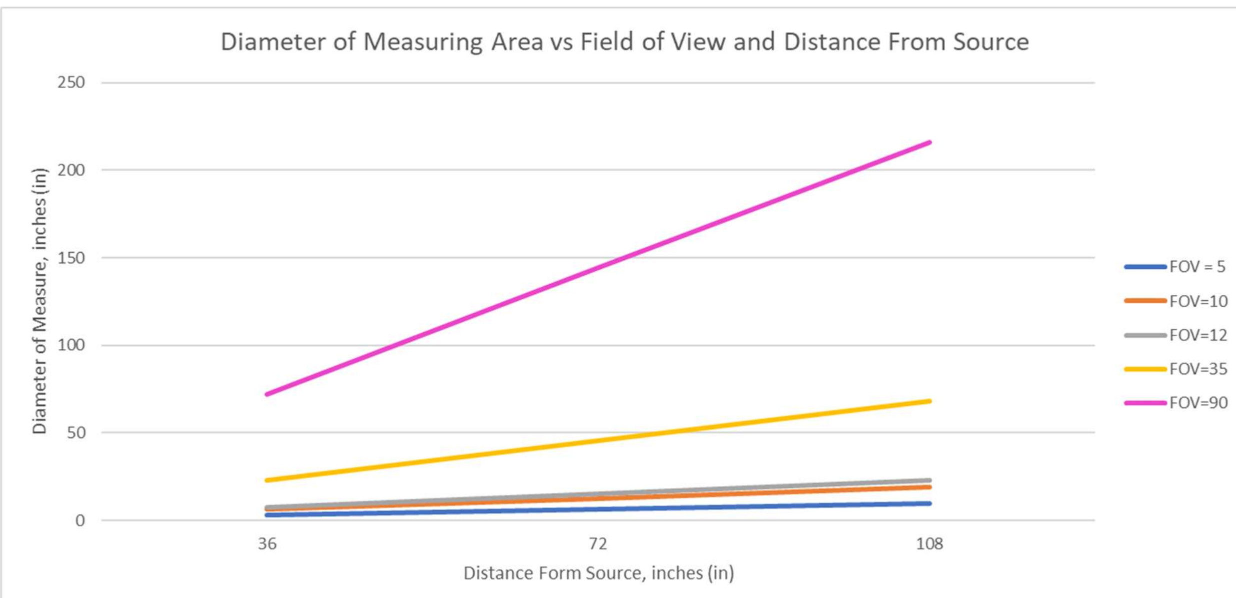


Figure 7: FOV Chart

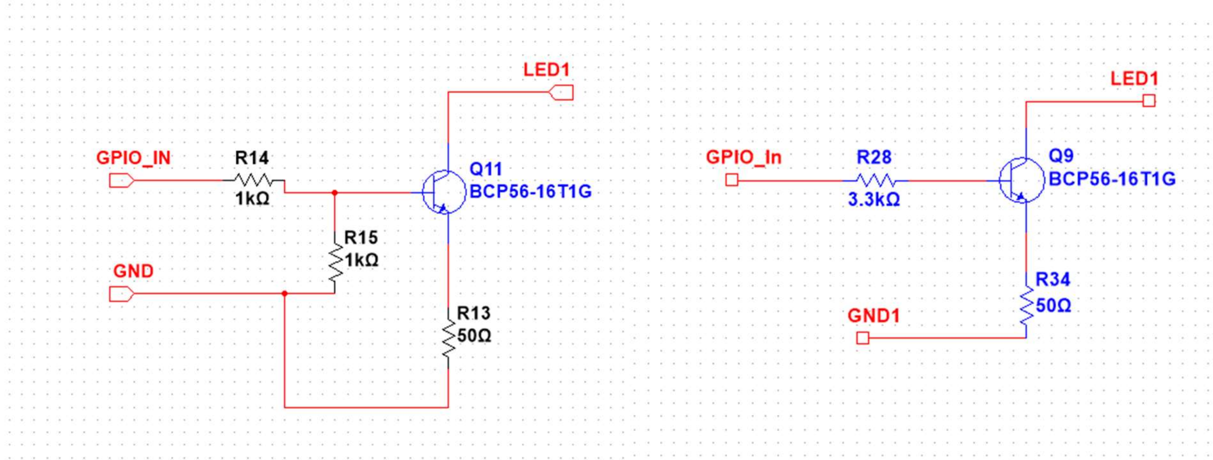
As seen in the chart, a 5-degree field of view produces the smallest zone of measurement. According to the data, 5 degrees is the largest FOV that can be used while keeping the zone of measurement under 6 inches at 6 feet of distance. This accuracy is the reason why a more expensive sensor with a smaller FOV had to be chosen in lieu of a cheaper sensor.

LED Array

Since the primary purpose of Pantastic was to alert a user if a stovetop was left on unattended, the LED Array was an additional component included to allow users to quickly estimate the temperature without having to look at the web application. The LED array includes 6 LEDS and was designed with the intention of giving the user an approximate pan temperature by progressively illuminating the LEDs in a vertical line when the temperature reaches a threshold. By assuming average room temperature to be 74°F, it was determined that the initial threshold for the bottommost LED to turn on would be 100°F and it would stay illuminated until the target temperature went under the threshold. Each progressive LED would turn on in temperature increments of 50°F until the entire 6 LED array was turned on at any temperature greater than or equal to 350°F.

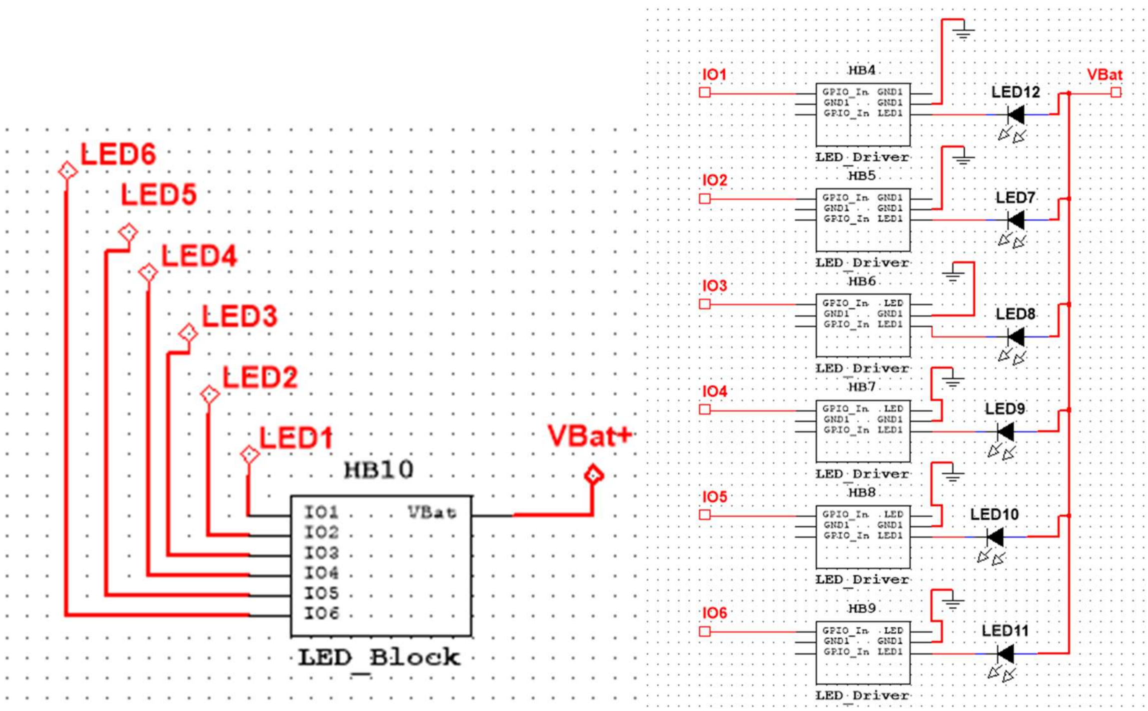
When choosing the LED's, it was decided that red would be the ideal color due to the attention-grabbing nature of it. The only electrical constraint that was required was to be able to operate in a range of 0-3.6V due to that being the voltage of the battery. Ideally, the LED would also be bright enough to catch attention and have a small current pull in order to maximize the battery life. Because of this, we chose the Red LED from Kingbright due to the ability to operate in the given voltage range as well as having a max current rating of 17.5mA at room temperature [42]. With the LEDs for the array chosen, the next step was to develop drivers that would allow the LED to be turned on from a GPIO output and battery voltages.

For each LED, a driver circuit was developed using the 1A, 80V NPN BJT from Rochester Electronics, LLC [43]. The idea behind this was to use the BJT as an electronic switch that would turn on when the designated pin was activated. The initial schematic that was sent out with the testing PCB can be seen in Figure 8(A). The idea was to use a voltage divider to lessen the voltage and current input from the GPIO pin to the base of the BJT with an additional resistor off of the emitter to control the current going through the LED. The values initially chosen were picked using Multisim's voltmeter and ammeter functions to assure an operating voltage and current of 3V and 13mA respectively. However, once testing began, it was determined that the LEDs were not getting sufficient current to achieve the desired brightness. To fix this, the voltage divider was replaced with a singular 3.3kΩ resistor, while maintaining the same emitter resistor. The schematic seen in Figure 8(B) provided a sufficient voltage difference and current through the LED to illuminate to the desired level. With the LED driver circuit completed, it was then added to the complete schematic in the LED block in Figure 9(A) which contains the six driver circuits as seen in Figure 9(B).



(A) **(B)**

Figure 8: (A) The Preliminary LED Driver Circuit, Modified to the (B) Final LED Driver Circuit



(A) **(B)**

Figure 9: (A) Zoomed out LED Block Containing the (B) Driver Array with Attached LEDs

Buttons

With the design of Pantastic and user interface components used, two buttons were required, one to toggle on and off the aiming laser diode, as well as one to turn off the alarm if it was ringing. While switches could be used, a user pressing a button to turn on or off an aiming device or turn off an alarm seemed more appropriate. For Pantastic's purpose, a simple black push button was used from E-Switch for the laser diode, while a similar red push button was used to turn off the alarm [44]. In similar fashion to the color of the LEDs, the push button used to turn off the alarm was chosen to be red to indicate an important alert. A black push button was chosen for toggling the laser diode on and off to distinguish the functionality of the two buttons. Since the only difference between the two buttons was the color, the circuits used to connect them to the microcontroller were identical and can be seen in Figure 10.

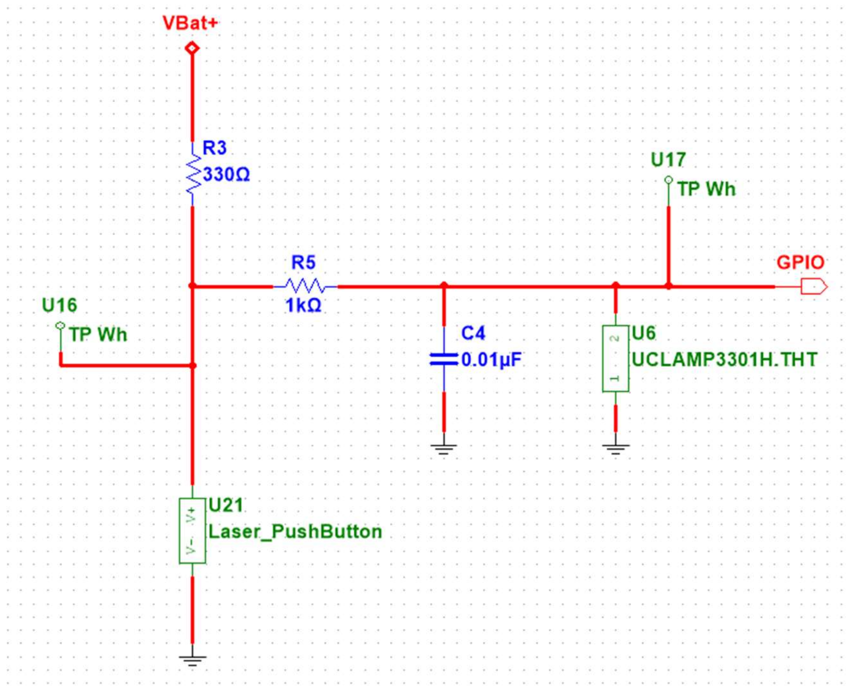


Figure 10: Push Button Subcircuit Schematic for both Laser Diode and Alarm Push Buttons

For the actual design of the button circuitry, the first and primary consideration was to assure overall protection of the CC33220S. This is where the UCLAMP diode (U6) [45] and 0.01uF surface mount capacitor (C4) were used. The capacitor was implemented to protect from any sort of high frequency electrical noise that could disrupt the input to the board and affect the GPIO pin reading. Along with this, the diode was included to prevent any voltage surges from the battery to the pin. Adding this diode protects the board from a possible fried pin due to a larger than anticipated voltage coming from the 3.6V battery. The resistors, R3 and R5, limit the current coming from the battery to prevent the board from getting destroyed while the button is not pressed. R3 is also in place to prevent a short circuit when the button is pressed, which could result in significant damage to both the battery and other components. The button schematics can be seen in Figure 11.

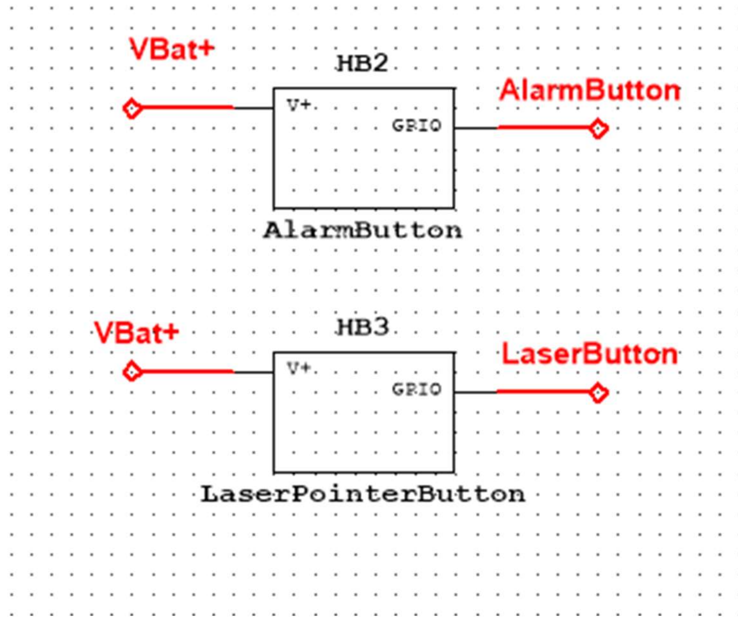


Figure 11: Alarm Button and Laser Button as Seen on the Main Schematic

Laser Diode

Since the IR sensor does not have its own aiming system, the Pantastic team decided to use a small red laser diode as an aiming device for the system. Similarly to the other components, the device was restricted to a max operating voltage of 3.6V. For this, the 650nm, 2.8-5.2V 25mA laser diode from Adafruit Industries LLC [46] was chosen. With these device constraints, a more specific driver had to be developed and tuned. Similarly, to the LED driver circuits, the same BJT was used as a sort of electrical switch to turn on the laser diode with a similar layout. While most driver circuits required a specific voltage and current regulator with a feedback potentiometer to tune the driver, two 10kΩ potentiometers were attached to the BJT. One of these potentiometers connected the GPIO pin on the microcontroller to the base of the BJT and the other connected the emitter to ground. These were organized to control the voltage and current through the laser diode to ensure that it would turn on but not receive too much power to the risk of destroying the device. After significant tinkering between the two, being careful not to overpower the laser, it was determined that a 50Ω resistor would work for the emitter resistor and a 1.5kΩ resistor would work for the base input as seen in Figure 12. Since the device could run into some operating issues with noise or reverse bias surges, a 0.1uF capacitor and a 1N4001 diode in a reverse bias orientation were used to ensure protection of the diode in those specific cases. The block diagram of the driver for the main schematic can be found in Figure 13.

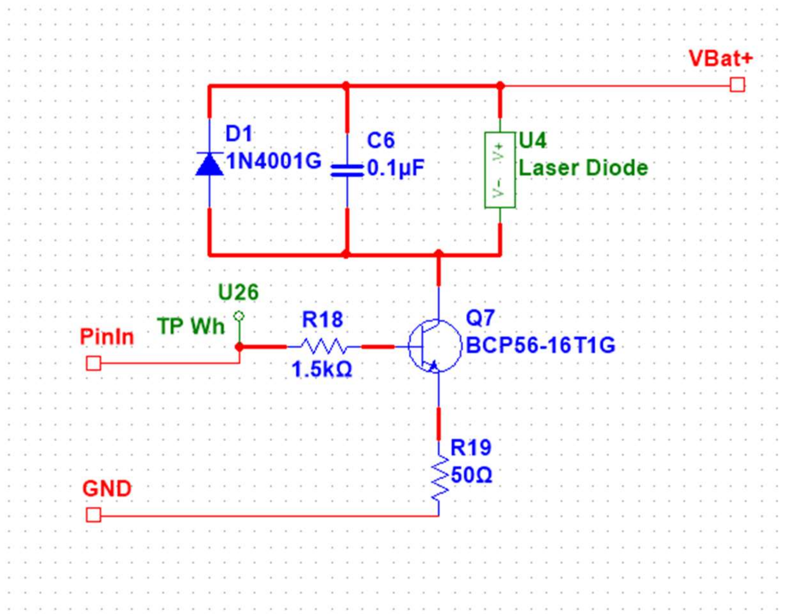


Figure 12: Laser Diode Driver Schematic

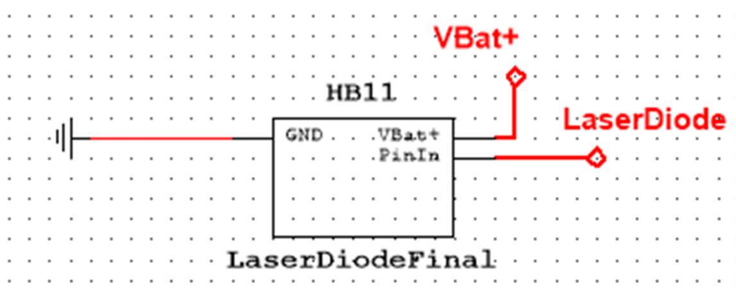


Figure 13: Laser Diode Driver Block Diagram for Final Schematic

Piezo Buzzer

A major struggle with the original piezo chosen for the project was getting it loud enough. Generating a square wave to ring the buzzer produced, at best, a quiet rattling noise. The piezo was capable of producing 85 dB, however, the switch from a 9V to a 3.6V battery made it so the voltage required to reach this sound level was no longer achievable. It was found that an active piezo buzzer from a team member's hobby electronics kit produced a better alarm sound and only required a DC input. This significantly reduced the difficulty of implementation so it was decided to go forward with this piezo. Because the active piezo that was used is a generic piezo with no identifier information, the closest alternative that could be found was cited and used in the BOM in Table B1 in Appendix B.

Next, it was realized that if the team created an AND circuit with the source voltage as one input, the GPIO pin as the other input, and with the connection being made between the source voltage and the piezo, a much louder sound could be generated. Since the PCB had already been created, this circuit was soldered on a breakout board and mounted with wires beneath the launchpad. The circuit is shown in figure 14**.**

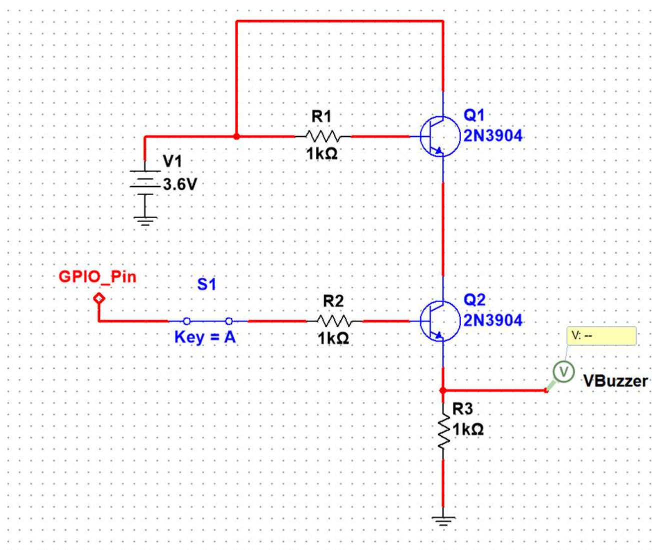


Figure 14: New Piezo Driver

It should be noted that the driver would produce the same result with less materials used if constructed in the manner shown in Figure 15.

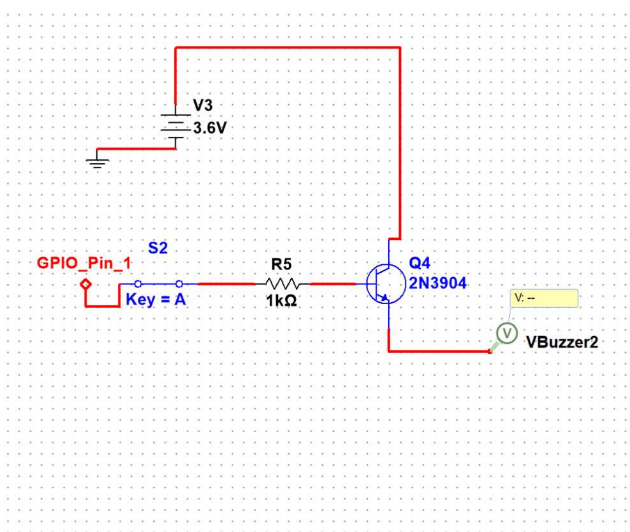


Figure 15: Simplified Piezo Driver

PCB Layout Decisions

After the final Multisim schematic had been created and exported to Ultiboard, the next decision that needed to be made was the layout of the components on the custom PCB. The final custom PCB schematic can be seen in Figure 16, and the layout with clearly defined reference designators can be seen in Figure 17.

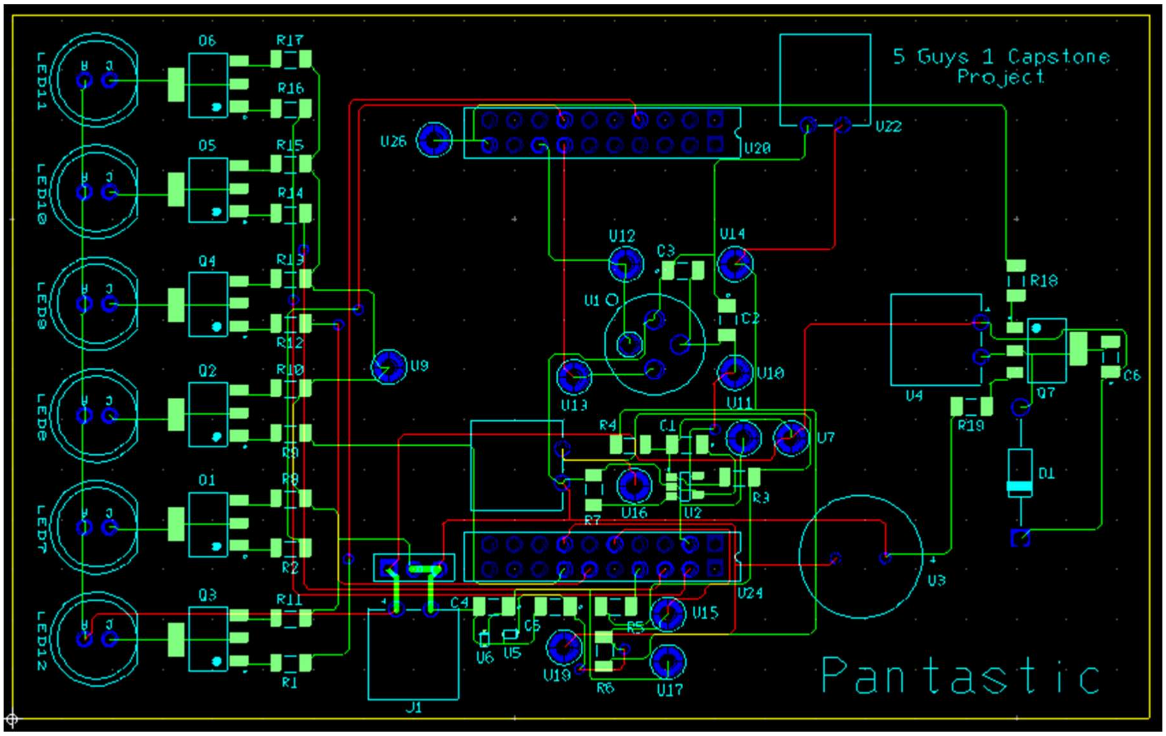


Figure 16: Final Board PCB Layout

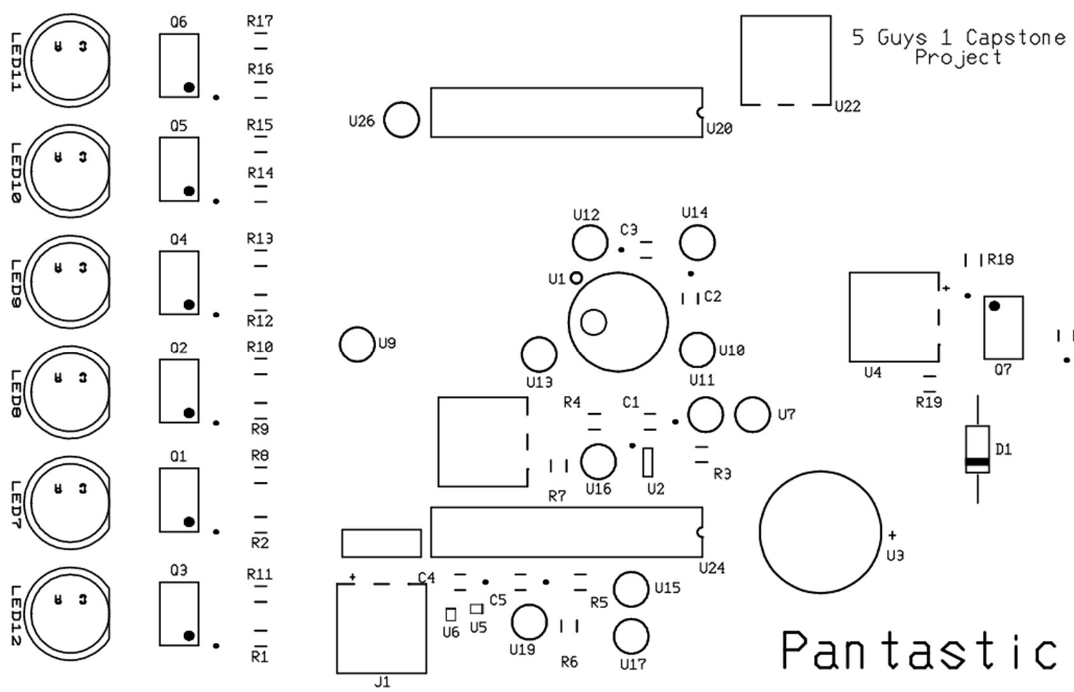


Figure 17: Final Board PCB Layout Reference Designators

The first consideration for the PCB that needed to be accounted for was how the launchpad was going to fit onto the board. Since the IR sensor needed to be unobstructed for proper functionality of the device, it was determined that the best orientation was for the launchpad to be under the PCB by fitting two sets of 2x10 headers [47] and a singular 1x3 header [48] onto the board to the exact dimensions of the Launchpad.

With the Launchpad fitting under the board, it also allowed for the LED array to be visible from the front of the device without having to elongate the PCB. Since the LED array's functionality was dependent on being in a vertical orientation, the left side of the PCB provided nearly the exact dimensions needed to align the 6 LEDs and their respective drivers. This allowed them to be out of the way of the rest of the components that needed to be directly interacted with (buttons, IR sensor, and laser) while still being able to be seen by the user.

The next consideration that was accounted for was the orientation of the battery. During development of the PCB, it was uncertain how the battery would be oriented with the final casing of the device. The most flexible design to account for this was to use a vertical phoenix connector to permit the battery to be moved anywhere in the casing of the device, while still being able to power the board without obstructing any other components. To prevent any possible resistance or capacitance from long power traces, the base of the phoenix connector was placed as close to the 1x3 pin as possible. This orientation and component attachment choice permitted the least obstructed power to the board, as well as gave flexibility to move the battery casing to a convenient space for a user to access.

Following the power to the board, the next most important component to orient on the board was the IR sensor which collected the entirety of the temperature data for the device. The most straightforward idea for this was to put it in the middle of the PCB to give the most intuitive way for aiming the sensor. Along with this, since there was some uncertainty to how we were going to set up the laser pointer, a phoenix connector was set up in close proximity to the IR sensor. This allowed for the laser to be attached directly to the casing next to where the IR sensor was pointing. Additionally, the buttons were connected to the board using phoenix connectors as well. This allows them to be movable when constructing the case to allow for the best possible placement without messing with the integrity of the design. Both of these vertical connectors were placed on the PCB next to their respective driver circuits and out of the way of any possible interference with the effectiveness of the IR sensor.

The last main component, the buzzer, was also placed with the intention of being semi-central to the board. Therefore, it was placed below and to the right of the IR sensor. This placement was used to allow the buzzer to be central to the board without displacing any of the other components that had stricter placing requirements. The driver for the buzzer was placed slightly to the right of the buzzer itself. Lastly, the test points were oriented as close to the device or driver as possible to give the most accurate reading of the voltage and current going to the components.

Firmware

General Firmware

The code functionality can be represented as an algorithmic flow chart which is shown in Figure 18 and Figure 19**.**

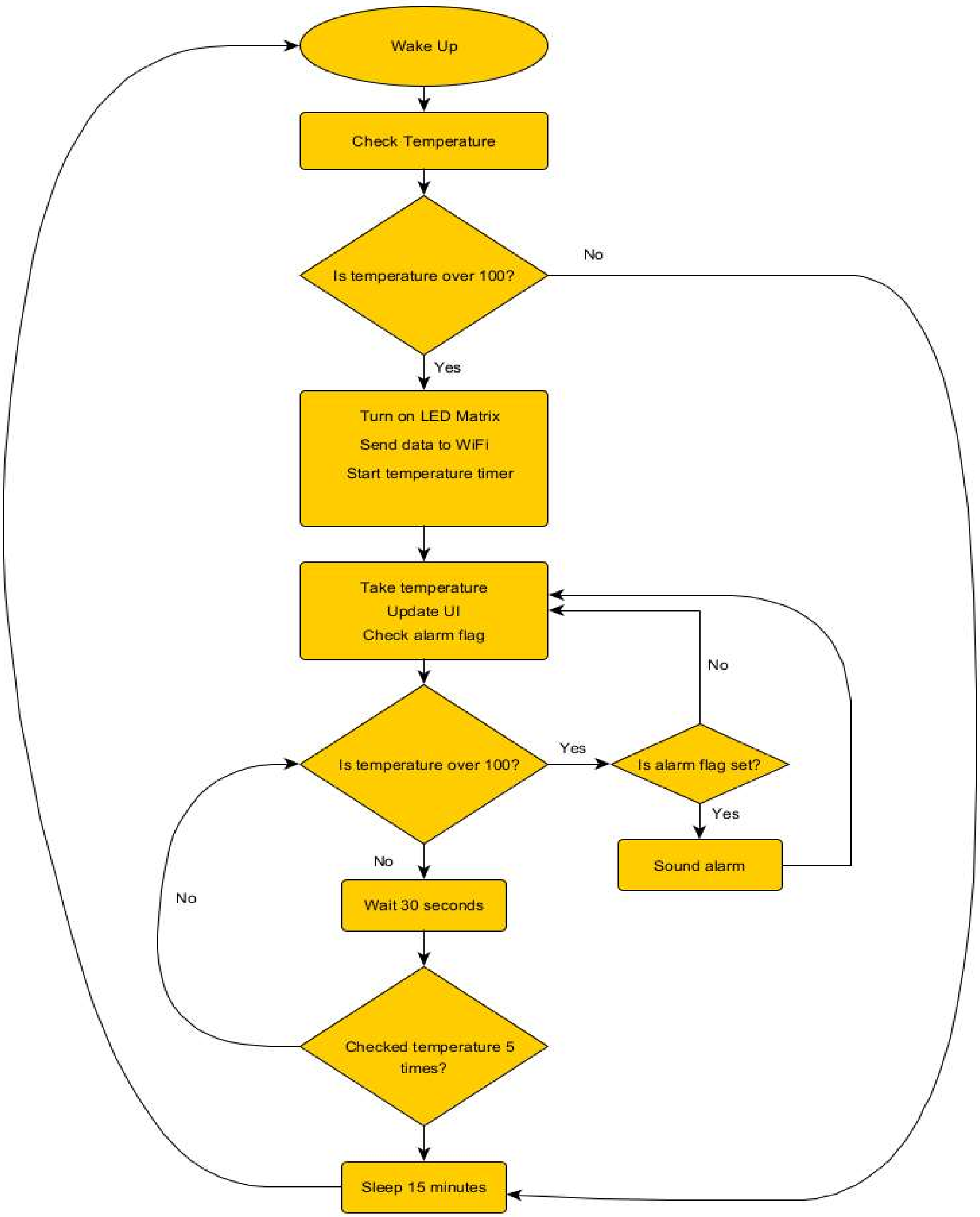


Figure 18: Algorithmic Flowchart

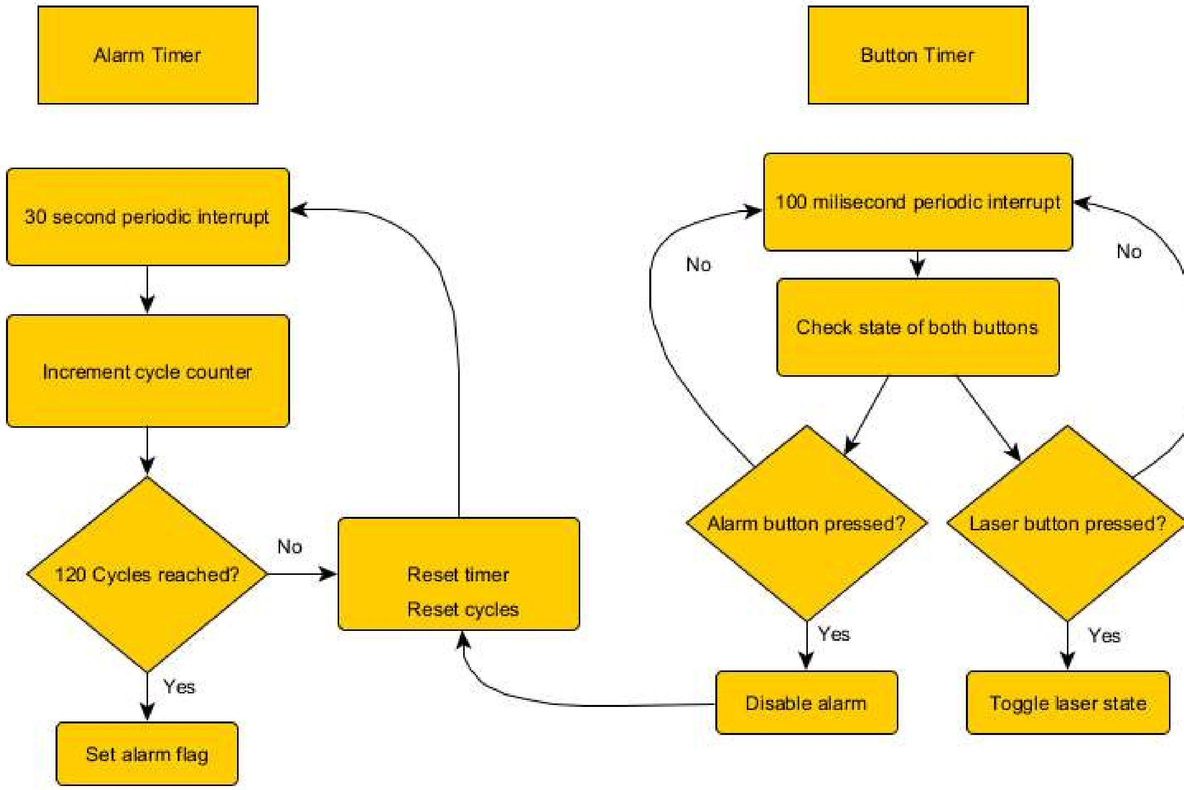


Figure 19: Interrupts Flowchart

As seen in the first algorithm flowchart in figure 18, the device works by waiting until the temperature crosses the threshold of 100°F. Fahrenheit was used over Celsius or kelvin because the device is meant to exist as an IoT helper device. Because it is not being used for scientific purposes, and because the device is being developed in America, Fahrenheit is the most appropriate unit to use.

If the temperature threshold has not been crossed, the device enters an idle state for 15 minutes. If the temperature threshold has been reached, then it will update the LED heat matrix to provide a visual approximation of the temperature. It will also send the temperature value to the web application via WiFi and start the one-hour timer. The timer is to alert the user that high temperature has been detected for a long period of time and serves as an extra safety feature.

Next, the program will repeatedly check the temperature and the alarm flag. The alarm flag is an indicator variable that indicates the one-hour timer has triggered. The alarm flag will be covered more later, but if it is set, the alarm will sound until the user disables it by pressing the red alarm push button. If the temperature has fallen below 100°F, then the device will verify that the temperature has actually fallen. It is possible that if the IR sensor is temporarily obstructed, it will detect a lower temperature. To avoid the device from falsely shutting off, it will check the temperature every 30 seconds for 5 minutes. If the temperature goes above 100°F during that time, normal operation will resume. If the temperature remains below 100°F, then the device will enter an idle sleep state.

The alarm timer and the button states are tracked with two periodic interrupt timers. The alarm timer is set to trigger every 30 seconds. Every time the alarm timer IRQ handler is called, it updates the number of alarm cycles that have been reached. When 120 cycles have been reached the alarm flag will be set (120 cycles = 1 hour). This enables the main program cycle to know the alarm needs to be activated.

The buttons are tracked with another periodic interrupt. By checking the state of the button every 100 ms, the state of the button is debounced so that there are no erroneous presses recorded. Originally, the buttons were tracked with interrupts which would register several button presses for each press. If the alarm button is pressed while the alarm is sounding, then it will reset the alarm timer and turn off the alarm. If the alarm is not sounding, then the alarm timer will be reset. If the laser button is pressed, then the state of the laser diode is toggled on or off.

IR Sensor

The IR sensor is a Melexis MLX90614 Infrared Temperature sensor. The sensor sends data using SMBus. The code for the CC3220S onboard I2C bus was not compatible with SMBus despite I2C and SMBus being very similar protocols. Due to this incompatibility, a custom "bit-bang" I2C code was implemented to access the data on the sensor. To save development time, an online example of I2C "bit-bang" code was found and used as a skeleton [49].

One GPIO pin was dedicated to toggling the voltage regulator for the IR sensor on and off. This enables the launchpad to turn off the sensor if needed. Two more GPIO pins were used for the I2C bus: one for SDA, one for SCL. The data and clock lines used the internal pullup resistors of the CC3220S. The pins were also reconfigured repeatedly to achieve the desired functionality. When the line needed to be pulled high to represent a logical 1, the pin was set as an input with a pullup resistor. When the line needed to be pushed low to represent a logical 0, the pin was reconfigured as an output and set to 0.

One issue that was encountered was the timeout functionality of SMBus. This caused the sensor to timeout and reset communication if the clock line was low for more than 27 milliseconds or high for more than 45 microseconds. The issue was that the SMBus would timeout when the microsecond delay function included with the CC3220S was used; this would happen even if a small enough delay was used. To mitigate this timing issue, a custom delay function was implemented. This was done by running a NOP assembly code 80 times to generate a 1 microsecond delay. The logic behind this is the CC3220S runs at 80MHz, so 80 clock cycles with no activity would result in 1 microsecond of no activity. Technically, due to the "for-loop" there would be more than 80 cycles, but the delay was accurate enough that this was not an issue.

The controller indicates to the sensor that it needs temperature data by writing to the address of the sensor(0x5A). It does this by sending a byte over the bus where the first 7 bits are the address and the last bit is a 0 to indicate it is writing a command. If the sensor ACKs the data, then the command 0x07 can be sent. This command indicates the register on the sensor of interest is 0x07. This is also the register where the IR temperature data is stored on the sensor. When the sensor ACKs the command, the final byte is sent to the sensor. This command is the address in the upper 7 bits and a 1 to indicate read in the least significant bit. The sensor will ACK this and send 3 bytes in reply. The first byte is the most significant bit of the temperature, the second byte is the least significant byte of the temperature, and the final byte is an error checking byte. A new integer is created and the first byte is bit shifted over by 1 byte and the second byte fills the lowest byte of the integer. Once this is done, the number is multiplied by 50.0. At this point, the value is the temperature in Kelvin. A Kelvin to Fahrenheit formula is followed to achieve the final temperature. The code for this portion of the project can be found in the file *PantasticFinalVersion.zip* in the 5 Guys one Capstone Team Resources folder on Collab.

Figure 20: Pantastic Embedded Code

Buttons

The push buttons are configured GPIO pins set as inputs. In order to debounce the button pushes, the status of the buttons is checked through a periodic interrupt every 100 milliseconds. This time was chosen because under 100 did not properly debounce the button presses, and over 100 decreased the quality of the user experience by not registering every button push.

The interrupt reads the state of the GPIO pin and calls the button handlers. If the button has not been pushed, then the handler will exit without performing any actions. If the alarm button has been pressed, the alarm will turn off if it is ringing. If the alarm is not ringing but the alarm button is pressed then the alarm timer will reset to 0. If the laser button is pressed, then the state of the aiming laser is toggled on or off depending on the current state.

Timers

Two hardware timers are used for the project. The first is a timer that triggers a periodic interrupt to check the state of the buttons. As previously mentioned, this timer is set to a 100,000-microsecond period (100 milliseconds). It also runs in continuous mode where the timer automatically restarts. The second timer is the alarm timer. This timer is set to a period of 30,000,000 microseconds (30 seconds). This timer runs in one-shot mode where the timer must be manually restarted.

The IRQ handler of the timer that checks the button has already been discussed in the button section. The handler of the alarm timer increments a variable to indicate the number of interrupts that have been generated. This is how the alarm measures how much time has passed to reach timer values greater than the timer modules can tolerate. When 120 interrupt cycles have been recorded, one hour has been reached. This sets a flag which will turn on the alarm on the next FSM cycle. The alarm is activated in the FSM cycle to prevent the alarm timer from turning on the alarm in the state where the device is confirming no heat is present.

Laser Diode

The laser diode is configured as an output GPIO with high drive strength. It is toggled on or off by the laser button.

Piezo

The piezo is configured as an output GPIO with high drive strength. It is turned on by the alarm timer and turned off by the alarm button. An active piezo buzzer was used so only DC power needs to be provided for the alarm to sound.

Casing

The casing is 3D printed using PLA plastic. It has a rectangular prism structure with an open window at the bottom to allow the IR sensor transparency to capture data. On the right side of the structure, there are two holes for buttons. The front of the structure is left open for easy access to the printed circuit board with an additional end piece to contain the device. The window on the top is for the visibility of the IR sensor with a slot for the laser pointer to be mounted on. Additionally, the inside of the case has a rail system for sliding the electrical components attached to the PCB. The bottom of the side has screw holes for the mounting component to attach to. Figure 21 displays these design implementations as a rendered CAD image.

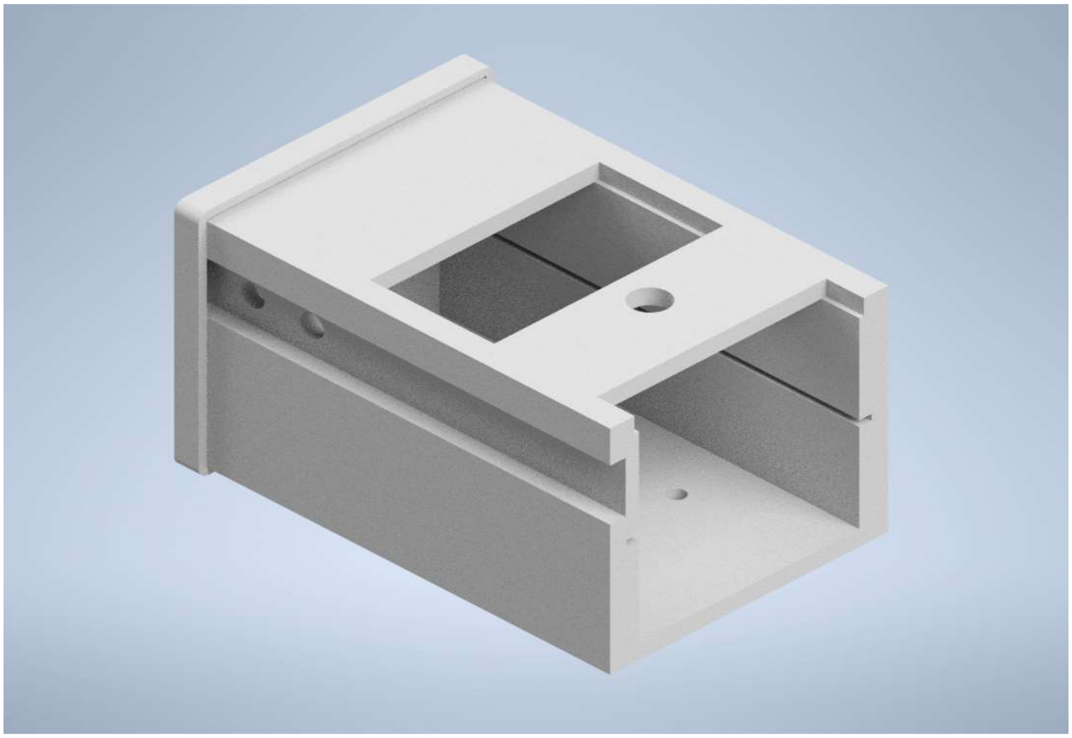


Figure 21: Rendered CAD Image of Casing

Additionally, the mounting system used was an online free design found on thingiverse.com [50], an online community for sharing 3D print files. The arm used to hang the case was found for free on Thingiverse [51]. Further, M5 screws were needed to mount the arm to the case as seen in Figure 22. These screws and hex nuts were also found on Thingiverse [52].

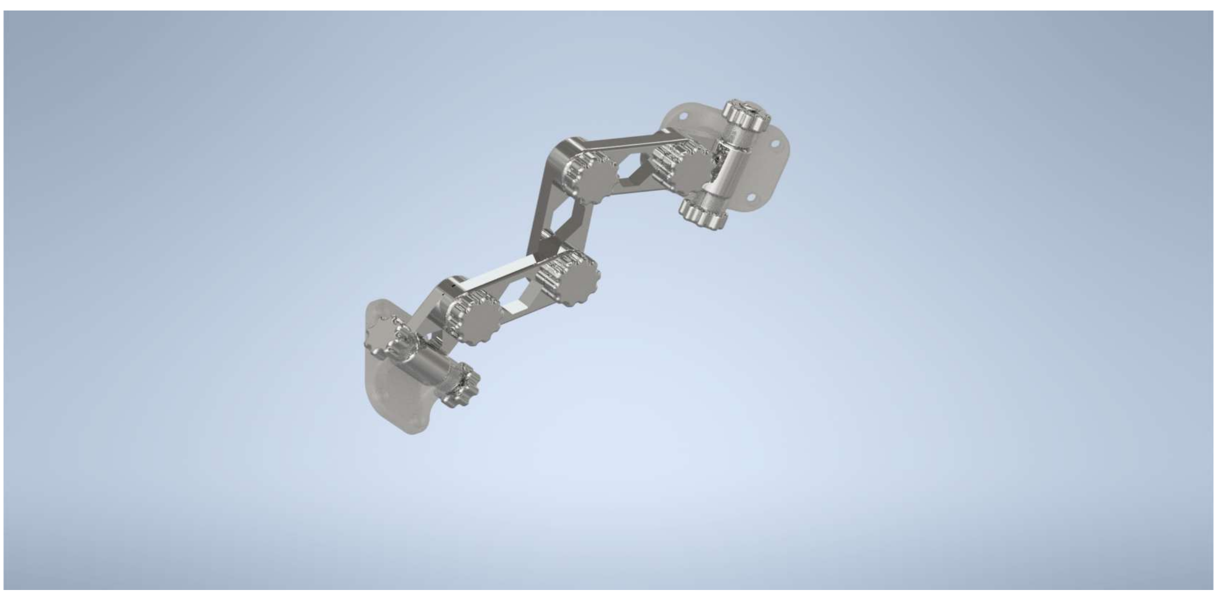


Figure 22: Rendered CAD Image of Mount System

Software

The final software deliverable is a local host server which can continuously receive data from the Pantastic. The TI httpget application for the CC3220s was used and modified to suit the needs of the application. The hostname was updated to the ip address of the laptop on the iphone hotspot so that the microcontroller could find the server. The request URI was updated to the post api so that data would be processed by the server. The header in the request was changed to post so that the api would be processed as a post request. The simplelink network handler was modified to set a flag so that the fsm in the driver code would wait for the network configuration to complete. The fsm then converted double temperature data into a string literal which was pointed to and sent along using the http task function in a thread. The http threads then were ended using pthread detach to make sure the microcontroller didn't crash by running out of memory. In stress testing, the device and server were able to continuously communicate without issue for 3 hours and sent around 3,000 data points. The software also includes a UI for the user to view the exact current temperature and the temperature history in the form of a graph.

Project Timeline

Original Gantt Chart

In our original project timeline, we were very optimistic about how swiftly we would go from task to task and be efficient about our project. We would start immediately with setting up the web UI while we figured out what parts we needed to order and started schematic design. Once parts were received, drivers could be tested on breadboards, PCBs could be designed and ordered, and embedded code could begin being created. Then, the network functionality could be added and the project could be tested and reformed to create a quality final product.

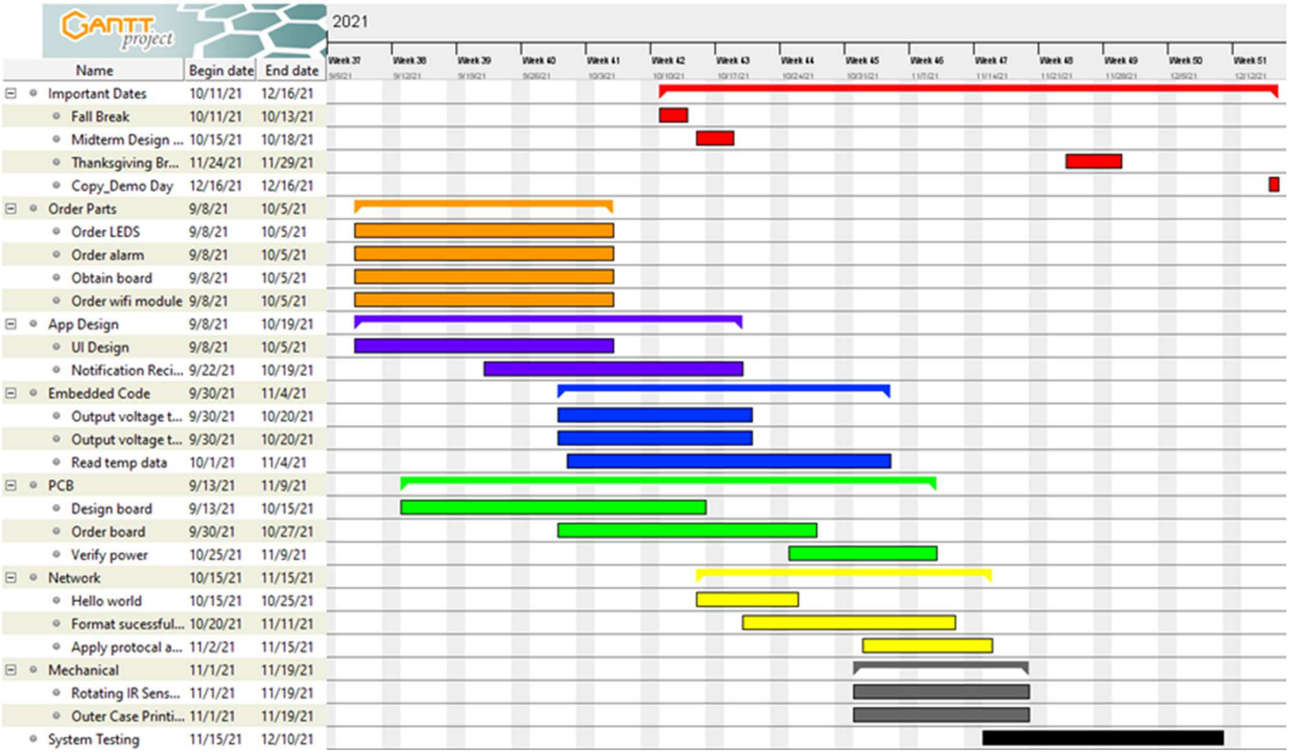


Figure 23: Original Gantt Chart

Revised Gantt Chart

The final resultant gantt chart timeline is a better depiction of reality, as the dates of actual work have been shifted back and the length of tasks has dramatically extended. The timeline did not accurately reflect that work would start slow, have to improve over time, and balance work with our classes and schedule. We had to file many part orders to get backup parts. Our board died weeks into our project and there was a moment when our development was stalled as we had to wait for the two new CC3220s boards. Testing and verifying measurements and results had to be pushed back as we tried to fix our circuits. It is evident that much work has been pushed back after Thanksgiving, when we previously thought that we would spend most of this time testing and refining. Network code functionality took a tremendous amount of time to debug and pushed all the way to demo day. Additionally, we finished assembly into our 3D printed casing near the end of the semester.

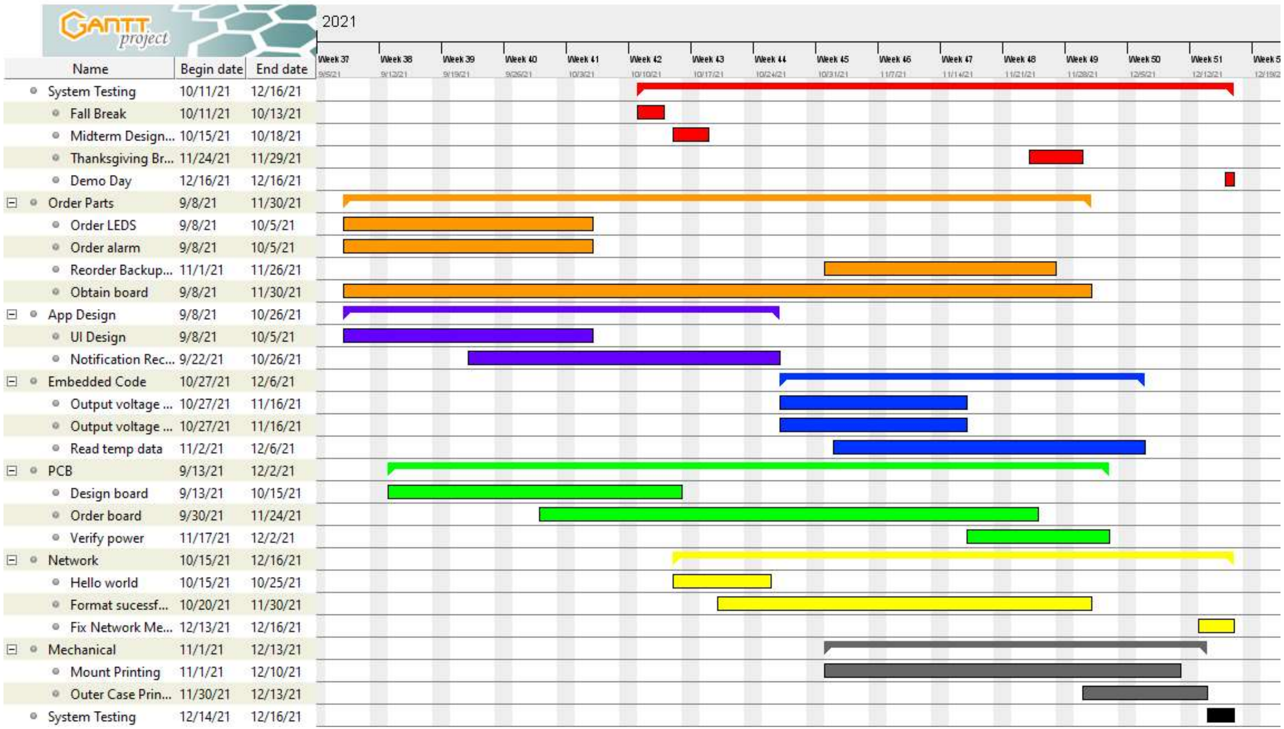


Figure 24: Revised Gantt Chart

Test Plan

To begin testing, the voltage and current output of the battery needed to be tested. It was tested using the voltmeter and ammeter available on the Craftsman Digital Multimeter. The voltage read was 3.65V. This value was expected because the battery was a 3.6V battery.

Next, to ensure that the board would be properly powered, U7 was tested to ensure that the board was properly connected to the power supply. The voltage measured was 3.102V. This immediately raised red flags as this is not the expected voltage of the battery. Further, the IR sensor regulator was not receiving enough power to turn on and the LEDs were too difficult to see. Considering this was at 2:00 am the night before the demo session, it was decided to drill a hole in the side of the case and use usb power to complete the project.

The voltage at U7 was 3.3V. U11 was then tested to ensure the IR sensor voltage regulator was enabled. The voltage was 3.23V. This was expected because this value enables the regulator. U10 was then tested to ensure that the voltage regulator was properly powering the IR sensor. The voltage measured was 3.002V. This value was expected because the regulator is supposed to provide an even 3.0V to ensure a stable IR sensor voltage for more accurate measurements.

Next, the buttons were tested by examining the signal on the pin. Since the buttons are connected to pullup resistors, it is expected that the voltage would fall to 0 when the button is pressed. An example of this is shown in figure 25 below. As seen in the figure, the default voltage is high and the voltage drops to low when the button is pressed.

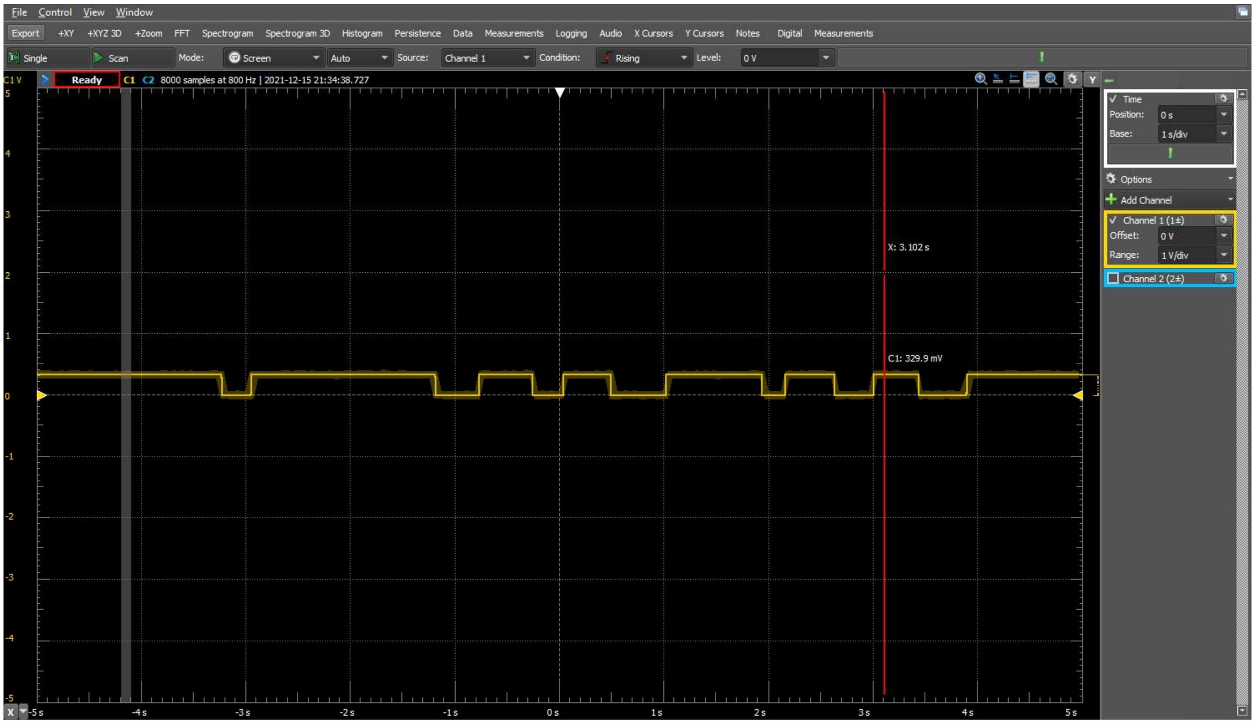


Figure 25: Push Button Verification

It was also important to test the temperature output from the IR sensor and verify the accuracy. This was done by comparing the temperature reading from the IR sensor to readings from a GDEALER digital thermometer probe [53] and the temperature from the temperature sensor on the Craftsman Digital Multimeter. The only issue with this testing setup was that the IR sensor has a greater accuracy than these two methods for measuring temperature. First, ambient air temperature was measured. Since the probe thermometer needs to be inserted into an object to record an accurate temperature, this device was excluded from this measurement. The Craftsman thermometer read 72.2°F and the IR sensor read 73.77°F**.** The proximity of these values to each other and to expected room temperature value verifies the accuracy of the device at room temperature. To measure higher temperatures, a Walmart Crockpot [54] was used to heat water. The GDEALER probe measured the temperature as 139.3°F**,** the Craftsman multimeter measured the temperature as 140.8°F, and the IR sensor measured the temperature as 135.84°F. The proximity of these readings allows for the conclusion that the IR sensor functions accurately.

Next, testing was done to ensure that the device FSM functioned correctly. This was done by looping through every possible direction on the FSM and ensuring the correct functionality was performed. This allowed numerous driver bugs to be revealed. For example, it was discovered that the initial time values set for the periodic timers caused the entire system to crash due to an overflow error. Other issues regarding resetting timers, toggling pins, and entering correct states were fixed during this debugging stage. In the end, the FSM correctly looped based on the requirements in the FSM flowchart and it was concluded the drivers and main code functioned correctly.

The LED testing plan was fairly simple due to the fact that it was impossible to kill the LED from too much voltage, so the only thing that needed to be altered was the current. Unfortunately, when initially testing the available max voltage and current that could be drawn from one of the GPIO pins, the values were smaller than initially expected from the datasheets. The recorded value of the voltage from the GPIO pin was approximately 3V when it was supposed to be 3.3V and the current was in the range of about 4mA when it should have been 6mA. This caused us to alter our driver circuits to provide for the lesser output power from the microcontroller. The emitter resistor was kept to limit the current going through the diode which proved to be sufficient. However, just doing a brightness check, we determined that the LED looked to be an average brightness with a 3.3kΩ base resistor. A 3V voltage was read over the diode, as well as 18mA of current going through the LEDs which was also right in the operating range of the devices.

To test the laser diode, two 10kΩ potentiometers were attached to the base of the BJT and the emitter of the BJT to adjust the current draw of the laser diode. After turning up both the potentiometers to their max resistance, they were slowly lowered until the laser diode turned on. After the threshold was reached, the potentiometers resistances were turned down to adjust for any smaller voltages than the battery was listed at. The resistances were then measured with an ohmmeter and they were measured to be 1.5kΩ for the base resistor and 50Ω for the emitter. With this information, the circuit was built and tested with the launchpad plugged running off both computer and battery power and the laser was still able to be turned on.

The WiFi testing was done in several stages. First it was tested to see if it could connect to an iPhone hotspot. This was done by seeing a connection appear on the iPhone and by seeing the connection information printing in the serial terminal. Next, http get methods were sent and it could be seen in the terminal that the data was leaving the device. On the server side, the data could be seen arriving. Once the data was correctly processed it was determined that the data was correctly arriving and updating the UI. It was then tested to ensure that this could run continuously and restart after a power loss. If the device was unplugged, it automatically reconnected to the hotspot and resumed sending data to the server. A final stress test was performed where the Pantastic continuously sent data to the server for almost three hours. Throughout this time, the Pantastic sent about 3,000 data points to the server without issue. This was considered a huge success.

Final Results

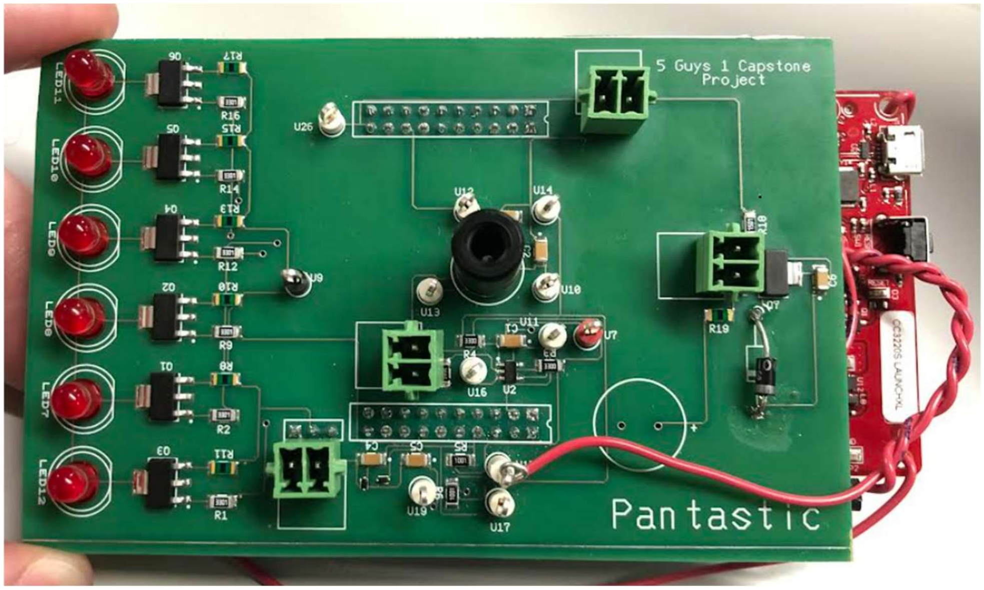


Figure 26: Top View of Pantastic

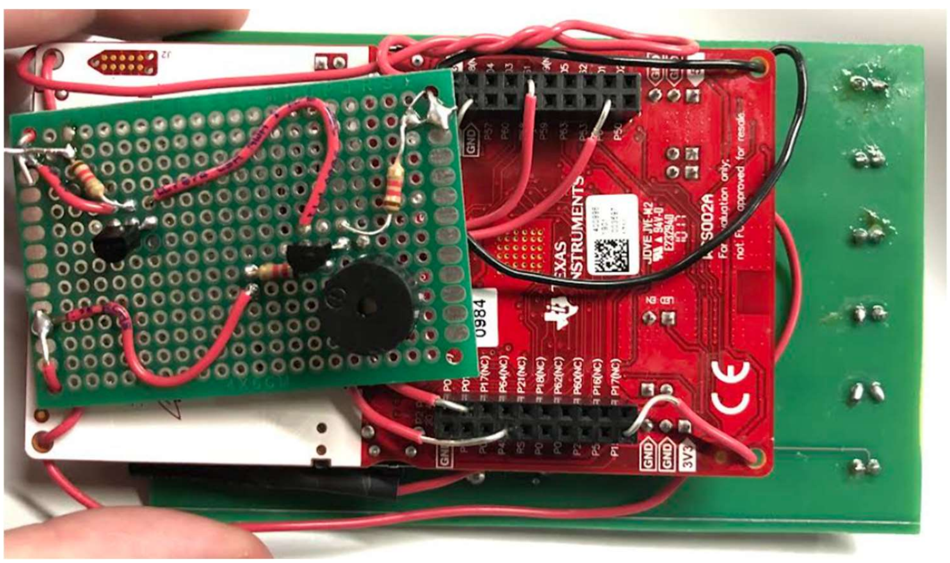


Figure 27: Bottom View of Pantastic

The final results of pantastic included a fully functioning kitchen safety device that is capable of continuously gathering temperature sensor data and alerting users when the temperature remains at an unsafe value for an extended period of time. The temperature data is sent to a server over wifi and the user is able to view a real time graph of the temperature readings through an intuitive user interface. A large portion of our project relied on the IR temperature sensor recording accurate temperatures. As shown in the demo video, pantastic was able to record the temperature accurately and send the data to the server for the user to view. As temperature readings are consistently collected, the graph in the user interface updates to include all temperatures recorded allowing the user to monitor temperature fluctuation over an extended period of time. In addition to monitoring the temperature through the user interface, the user is able to estimate the current temperature visually through the 6 LED array on the device. Because the functionality of Panstastic depends on the amount of time a cooking device has been at an unsafe temperature, in order to easily demonstrate the functionality of our project, we created two code programs for our device, each with different constraints.

We have a main project which includes constraints that would be used if Pantastic was in production, and a test program with constraints that we set to allow us to easily demonstrate the full functionality of our project. The full list of constraints for each program can be seen in Appendix A. For the main program, The LEDs light up in 50-degree increments beginning at 100°F. When the temperature reaches 350°F, all 6 LEDs would be lit up. For the test program, which is the program that was used in our demo video, the LEDs light up beginning at 100°F, but light up in 10-degree increments. When the temperature reaches 150°F, all of the LEDs are lit up. In addition to giving the user a visual representation of the temperature, Pantastic successfully gave users an audible queue when the temperature had been at an unsafe value for too long by sounding the piezo buzzer. The piezo buzzer continues to ring until the user turns it off by pressing the red push button. In the main program, the buzzer sounds if it detects temperatures equal to or over 100°F for more than 1 hour. In the test program, the buzzer still sounds if it detects temperatures equal to or over 100°F, but instead of the time being 1 hour, it sounds after 2 minutes. Pantastic satisfies the proposed functionality shown in table 1.

Table 1: Grading Template

Points	Records IR Temp.	Application Software	Embedded Code	Network Development	PCB/Mechanical
3	Device accurately records current temperature within a responsible range. Accuracy within 5 degrees Celsius of the given accuracy listed in the sensor data sheet	Software is complete and allows user to monitor the correct temperature through user interface	Code is complete and functions as intended	Network functionality is complete, and data is transferred as intended consistently	PCB and Mechanical components are excellent quality and function as intended
2	Records temperature with slight error. Accuracy within 10-20 degrees Celsius of the given accuracy listed in the sensor data sheet	Software is complete but the user interface does not automatically update temperature values	Code is complete but has a few minor errors	Network functions as expected most of the time, but still has slight issues	PCB and mechanical components are acceptable quality and function as intended
1	Records temperature very inaccurately. Accuracy within 30-40 degrees Celsius of the given accuracy listed in the sensor data sheet.	Software is complete but the user interface does not work	Code is finished but does not function correctly	Network functions but is very unreliable and inconsistent	PCB and mechanical components are finished but poor quality
0	Unable to record current temperature	Software is unusable or unfinished	Code doesn't run or is unfinished	Network connection does not function at all	PCB and Mechanical components are unfinished or don't work

Table 2: Points to Letter Grade

Accumulated Points	Letter Grade
12-15	A or A+
7-11	B or B+
4-6	C or C+
0-3	D or D+

Most importantly, Pantastic accurately records the current temperature. As seen in the test plan section, the IR sensor measured temperature within 5 degrees Celsius of the alternative temperature testing methods. The application and network software are complete and data can be transferred continuously through WiFi to a web application. This allows the user to monitor the current temperature, as well as temperature history through the Pantastic website interface. In addition, the embedded code is complete and functions as intended. The only portion of the above table that was not as successful as we would have liked it to be is the PCB/Mechanical section. We are happy with the connections on the top of our PCB shown in figure 26. but because we had issues with our launchpad pins, we had to create a separate circuit for the piezo buzzer and attach it to the bottom of our device as shown in figure 27. As a result, our PCB does not look as professional as it would have if we did not have this issue. Overall, the final version of Pantastic works very well and the team is satisfied with the results. The first four sections receive 3 points for achieving the full functionality that was decided on. The final section received a 2 due to the issues with the PCB that caused the unprofessional external wires to have to be run on the underside of the board. Totaling the points results in a 14 and an A+ for achieving all the goals that were decided on in the project proposal. Images of the final version of Pantastic can be seen in Appendix D.

Costs

The cost to manufacture 1 Pantastic device is $200.86. This price will fluctuate slightly depending on the availability and cost of parts. Table 3 below shows a high-level breakdown of the total cost to manufacture a single unit of Pantastic. In addition, the table shows a breakdown of the per-unit cost for Pantastic if 10,000 units were being manufactured.

Table 3: Pantastic Cost

Digi-Key Part Number	Cost for 1 Unit	Cost Per Unit for 10,000 Units
CC3220S-LAUNCHXL	$47.99	$47.99
MLX90614ESF-BCI-000-TU (Temperature Sensor)	$49.15	$39.85
1528-1391-ND (Laser Diode)	$5.95	$5.95
458-1433-ND (Piezo Buzzer)	$13.24	$8.73
TL-2100/S (Battery)	$6.78	$4.08
PS1024ARED (Alarm Push Button)	$1.63	$1.08
PS1024ABLK (Laser Diode Push Button)	$1.63	$1.08
PCB	$33.00	$7.25
PCB Components	$41.49	$19.17
Total	**$200.86**	**$135.18**

Looking at table 3, the two most expensive components in the Pantastic design are the CC3220S-LAUNCHXL and MLX90614ESF-BCI-000-TU. If 10,000 units of Pantasic were produced, the price of the MLX90614ESF-BCI-000-TU would decrease from $49.15 to $39.85, but the price of the CC3220S-LAUNCHXL would remain the same. Compared to the total price of $200.86 to manufacture a single unit of Pantastic, the price would decrease by 32.7% to $135.18 if 10,000 units were manufactured. The product did not utilize the full functionality of the CC3220S-

LAUNCHXL. If the product were to be mass produced, replacing the CC3220S with a cheaper alternative launchpad or a single IC chip would decrease costs significantly. The PCB for Pantastic is fairly simple but includes very small components that are difficult to solder by hand. Automating PCB production for the device would need to be considered which could increase costs. In addition to Automating PCB production, a new protective case would need to be designed and manufactured. Automated equipment would be used to manufacture the case and assemble the device.

Future Work

Casing

The case designed for the project was functional, however, it was a bit large and bulky. A redesigned case with a smaller, sleeker design that completely encloses the device while leaving the buttons and LEDs on the outside would improve the visual appearance and possibly the functionality of the case. Polycarbonate should be utilized to protect the IR sensor while leaving visibility for the laser to pass through the transparent material. Furthermore, Polycarbonate has a temperature rating of 155°C or 311°F which is suitable for kitchen environments [55]. Additionally, more contouring to the case such as filets and chamfers would be useful in reducing the material cost while improving the aesthetic and ergonomics of the design. The mounting device should be significantly shrunken and reduced to two swivel joints for 360° articulation. The actual mounting can be further improved through adding other options for how it can be stationed. Including options for suction, adhesives, or wall mountable screws are areas that the mounting device could acquire to increase the versatility of the product in a kitchen environment. Additionally, the assembly of the case could improve through the replacement of super glue as the foundation to a more robust alternative. Some alternatives include a peg-in-hole system or internal brackets that hold the casing together. Finally, the casing should be a closed system to ensure that no tampering is done to the electrical components. It would also be desirable to mount an LCD screen onto the device to display the exact temperature.

Circuitry

Moving forward, there are some significant areas for optimization that were not able to be implemented in the time allotted. First, in regards to the circuitry, the design utilized the maximum pins allotted for general purpose input/output (GPIO) which provides much room to expand. There are two possible improvements that could be made regarding the pin allotment. The first of these would be the utilization of a different microcontroller. Since we were using the maximum number of GPIO pins, it doesn't leave room for additional sensors or on-board user interfacing. Using a different board with more available GPIO pins would allow for additional components to be added seamlessly. Another solution to this that would be possible is to utilize external IC chips to handle the LED array off of a singular GPIO pin instead of the six pins that it is currently taking up.

Next, some portions of the project required the PCB in order to run tests due to the use of surface mount components. Because of this, it was not realized until the final PCB arrived that several pins were either unusable or non-functional. Due to this, it would be helpful to have a re-routed PCB with connections to the new pins that had to be used.

Firmware

Due to the numerous issues involved with developing with the CC3220S, a beneficial remake of the project would be one that uses a different WiFi capable device. Next, future iterations of the project could include expanded features such as an onboard LCD to display the exact temperature along with the ability to adjust certain settings such as timer values or temperature ranges. Another idea that was given to the team was to add the ability to sound an alarm if the temperature increases rapidly as an extra layer of safety.

Software

The first improvement to make to the software would be to send the data over an encrypted connection with an SSL certificate and https secure connection. More research would need to be done learning the depths of the TI simplelink SDK which is not very well documented and did not have any starter code for an https application. Other boards such as MSP variants had code developed for such a process. A better solution may be to switch over to a separate booster device such as an ESP board which is much more robust and well documented. Additionally, it would provide a better bridge to switching to a custom board that exclusively has the CC3220s chip for cheaper large scale production costs. Then it would make sense to host the application on a cloud server so that the server would be reliable and accessible from locations on different networks. At that point, a database would need to be developed so that the data would be the same and persistent from different network connections. A future feature of the UI could be to add a control that would send requests to a server on the board that would shut off the buzzer when the user has acknowledged the warning.

References

[1] “How Does a Contactless Thermometer Work?,” Aug. 11, 2020. [Online].

Available: http://www.precisioncut.org.uk/how-does-a-contactless-thermometer-work. [Accessed: 08-Sep-2021].

[2] B. S. S.-S. Alert, “BurnerAlert Stove Reminder,” *BurnerAlert Stove Alarm*. [Online]. Available: https://burneralert.com/. [Accessed 08-Sep-2021].

[3] “CCSTUDIO IDE, configuration, compiler or debugger | TI.com.” [Online].

Available: https://www.ti.com/tool/CCSTUDIO. [Accessed 09-Sep-2021].

[4] “CC3220s-LAUNCHXL,” *CC3220S-LAUNCHXL Development kit | TI.com*. [Online]. Available: https://www.ti.com/tool/CC3220S-LAUNCHXL. [Accessed: 19-Sep-2021].

[5] “Digital plug & play infrared thermometer in a TO-can,” *Melexis*. [Online]. Available: https://www.melexis.com/en/product/MLX90614/Digital-Plug-Play-Infrared-Thermometer-TO-Can. [Accessed 14-Sep-2021].

[6] “Multisim Live Online Circuit Simulator,” *NI Multisim Live*. [Online]. Available: https://www.multisim.com/. [Accessed 09-Sep-2021].

[7] “Ultiboard - NI.” https://www.ni.com/en-us/shop/software/products/ultiboard.html. [Accessed 09-Sep-2021].

[8] “Inventor Software | Get Prices & Buy Official Inventor 2022 | Autodesk.” https://www.autodesk.com/products/inventor/overview. [Accessed 14-Dec-2021].

[9] “Ultimaker Cura: Powerful, easy-to-use 3D printing software,” *ultimaker.com*. https://ultimaker.com/software/ultimaker-cura. [Accessed 14-Dec-2021].

[10] “Anycubic Mega Pro,” *ANYCUBIC 3D Printing*. https://www.anycubic.com/products/mega-pro. [Accessed 14-Dec-2021].

[11] “Equipment authorization – rf device,” *Federal Communications Commission*, 20-Mar-2018. [Online]. Available: https://www.fcc.gov/oet/ea/rfdevice.

[Accessed: 10-Sep-2021].

[12] “IEEE 802.11-2020 - IEEE standard for Information Technology--Telecommunications and information exchange between systems - local and metropolitan Area Networks--Specific requirements - Part 11: Wireless LAN medium access Control (mac) and physical Layer (PHY) SPECIFICATIONS,” *IEEE SA - The IEEE Standards Association - Home*. [Online]. Available: https://standards.ieee.org/standard/802_11-2020.html. [Accessed: 20-Sep-2021].

[13] “Product finder results | Wi-Fi Alliance,” *Wi-Fi*, 05-Nov-2020. [Online]. Available: https://www.wi-fi.org/product-finder-results?keywords=CC3220. [Accessed: 14-Dec-2021].

[14] "Association connecting Electronics Industries ... - IPC." [Online]. Available: https://www.ipc.org/TOC/IPC-2221A.pdf. [Accessed: 14-Dec-2021].

[15] "IPC-A-600J: Acceptability of printed boards." [Online]. Available: https://www.ipc.org/TOC/IPC-A-600J.pdf. [Accessed: 14-Dec-2021].

[16] E. Notes, "SMT / SMD components & packages, sizes, dimensions, details," *Electronics Notes*, 2021. [Online]. Available: https://www.electronics-notes.com/articles/electronic_components/surface-mount-technology-smd-smt/packages.php. [Accessed: 14-Dec-2021].

[17] E. Eric Peňa and M. G. Legaspi, "UART: A hardware communication protocol understanding universal asynchronous receiver/transmitter," *UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter | Analog Devices*, Dec-2020. [Online]. Available: https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html#:~:text=UART%2C%20or%20universal%20asynchronous%20receiver,%2Dto%2Ddevice%20communication%20protocols.&text=In%20serial%20communication%2C%20data%20is,for%20successful%20serial%20data%20transfer. [Accessed: 14-Dec-2021].

[18] "Introduction to SPI Interface | Analog Devices." https://www.analog.com/en/analogdialogue/articles/introduction-to-spi-interface.html# (accessed Dec. 14, 2021).

[19] J.-M. Irazabal and S. Blozis, "AN10216-01 I2C Manual." Philips Semiconductors, Mar. 24, 2003. [Online]. Available: https://www.nxp.com/docs/en/application-note/AN10216.pdf.

[20] "IEEE 1149.1," *JTAG*, 12-Jun-2019. [Online]. Available: https://www.jtag.com/ieee-1149-1/. [Accessed: 14-Dec-2021].

[21] "EU Declaration of Conformity (Doc) - CC3220S-LAUNCHXL," *TI*, 02-Dec-2017. [Online]. Available: https://www.ti.com/lit/cr/swrq055b/swrq055b.pdf?ts=1639525762743&ref_url=https%253A%252F%252Fwww.ti.com%252Ftool%252FCC3220S-LAUNCHXL. [Accessed: 14-Dec-2021].

[22] "Craftsman 82007 | Test Meter PRO." http://testmeterpro.com/craftsman-82007/. [Accessed 14-Dec-2021].

[23] "USB Oscilloscope and Logic Analyzer - Digilent Analog Discovery 2." https://digilent.com/shop/analog-discovery-2-100ms-s-usb-oscilloscope-logic-analyzer-and-variable-power-supply/. [Accessed 14-Dec-2021].

[24] "Digilent WaveForms - Digilent." https://digilent.com/shop/software/digilent-waveforms/. [Accessed 14-Dec-2021].

[25] "Arduino - Home." https://www.arduino.cc/. [Accessed 14-Dec-2021].

[26] "Arduino IDE." https://www.arduino.cc/en/software. [Accessed 14-Dec-2021].

[27] "React – A JavaScript library for building user interfaces." https://reactjs.org/. [Accessed 14-Dec-2021].

[28] "Virtual DOM and Internals – React." https://reactjs.org/docs/faq-internals.html. [Accessed 14-Dec-2021].

[29] "Git." https://git-scm.com/. [Accessed 14-Dec-2021].

[30] "GitHub: Where the world builds software · GitHub." https://github.com/. [Accessed 14-Dec-2021].

[31] yWorks company the diagramming, "yEd Graph Editor," *yWorks, the diagramming experts*. https://www.yworks.com/products/yed. [Accessed 14-Dec-2021].

[32] "Introducing hooks," *React*. [Online]. Available: https://reactjs.org/docs/hooks-intro.html. [Accessed: 16-Dec-2021]. https://reactjs.org/docs/hooks-intro.html

[33] "Chart.js," *Chart.js | Open source HTML5 Charts for your website*. [Online]. Available: https://www.chartjs.org/. [Accessed: 16-Dec-2021].

[34] "Download Putty - a free SSH and telnet client for windows," *Download PuTTY - a free SSH and telnet client for Windows*. [Online]. Available: https://www.putty.org/. [Accessed: 16-Dec-2021].

[35] *Postman*. [Online]. Available: https://www.postman.com/. [Accessed: 16-Dec-2021].

[36] S. Cass, "Chip shortage challenges maker manufacturers," *IEEE Spectrum*, 10-Aug-2021. [Online]. Available: https://spectrum.ieee.org/maker-manufacturers-squeezed-chip-shortage. [Accessed: 19-Sep-2021].

[37] M. M. Kimura and T. A. Kudo, "Contactless thermometer," DE69721886T2 [Online]. Available: https://patents.google.com/patent/DE69721886T2/en?q=contactless+thermometer&oq=contactless+thermometer

[38] N. Kopp, "Stove Sensor," US20190238958A1 [Online]. Available: https://patents.google.com/patent/US20190238958A1/en?oq=US20190238958A1

[39] H. V. Bagh and S. Lintonen, "Stove Guard Utilizing Different Wavelengths," US20190360866 [Online]. Available: https://patents.google.com/patent/US20190360866A1/en?assignee=safera&oq=safera

[40] "TL-2100/S Tadiran Batteries | Battery Products | DigiKey." https://www.digikey.com/en/products/detail/tadiran-batteries/tl-2100-s/512512 [Accessed: 10-Sep-2021].

[41] G. Hollows and N. James, “Understanding focal length and field of View: Edmund Optics,” *Edmund Optics Worldwide*. [Online]. Available: https://www.edmundoptics.com/knowledge-center/application-notes/imaging/understanding-focal-length-and-field-of-view/. [Accessed: 10-Sep-2021].

[42] “WP7113ID5V Kingbright | Optoelectronics | DigiKey.” https://www.digikey.com/en/products/detail/kingbright/WP7113ID5V/3084220?s=N4IgTCBcDaIOwFYAsBaAjHAzGFA5AIiALoC%2BQA. [accessed Dec. 15, 2021].

[43] “BCP56-16T1G onsemi | Discrete Semiconductor Products | DigiKey.” https://www.digikey.com/en/products/detail/onsemi/BCP56-16T1G/918356. [accessed Dec. 15, 2021].

[44] “PS1024ABLK E-Switch | Switches | DigiKey.” https://www.digikey.com/en/products/detail/e-switch/PS1024ABLK/81742?s=N4IgTCBcDaIKIHEwAYCMqC0A5AIiAugL5A [accessed Dec. 15, 2021].

[45] “UCLAMP3301H.TCT Semtech Corporation | Circuit Protection | DigiKey.” https://www.digikey.com/en/products/detail/semtech-corporation/UCLAMP3301H-TCT/1000952?s=N4IgTCBcDaIKoGEAyBBAsgBQMxYAwEYAJBAFQFoA5AERAF0BfIA [accessed Dec. 15, 2021].

[46] “1054 Adafruit Industries LLC | Optoelectronics | DigiKey.” https://www.digikey.com/en/products/detail/adafruit-industries-llc/1054/5629439 [accessed Dec. 15, 2021].

[47] “PPPC102LFBN-RC Sullins Connector Solutions | Connectors, Interconnects | DigiKey.” https://www.digikey.com/en/products/detail/sullins-connector-solutions/pppc102lfbn-rc/807245 [accessed Dec. 15, 2021].

[48] “PPPC031LFBN-RC Sullins Connector Solutions | Connectors, Interconnects | DigiKey.” https://www.digikey.com/en/products/detail/sullins-connector-solutions/pppc031lfbn-rc/810175 [accessed Dec. 15, 2021].

[49] 262588213843476, “I2C Bit-Banging - MSP430,” *Gist*. https://gist.github.com/ksvbka/fec1f28ce40c276334fb. [accessed Dec. 16, 2021].

[50] Thingiverse.com, “Thingiverse - Digital Designs for Physical Objects.” https://www.thingiverse.com/. [accessed Dec. 16, 2021].

[51] Thingiverse.com, “Articulating, Wall-Mounted, Magnetic Phone Mount by doctriam.” https://www.thingiverse.com/thing:2448971. [accessed Dec. 15, 2021].

[52] Thingiverse.com, “3D printable M5 Hex Screw and Nut by mrisco.” https://www.thingiverse.com/thing:4677349. [accessed Dec. 15, 2021].

[53] “Amazon.com: GDEALER DT09 Waterproof Digital Instant Read Meat Thermometer with 4.6” Folding Probe Calibration Function for Cooking Food Candy, BBQ Grill, Smokers: Home & Kitchen.” https://www.amazon.com/GDEALER-DT09-Waterproof-Thermometer-Calibration/dp/B077N9T555 [accessed Dec. 15, 2021].

[54] “Crock-Pot 2 Quart Round Manual Slow Cooker, Black,” *Walmart.com*. https://www.walmart.com/ip/Crock-Pot-2-Quart-Round-Manual-Slow-Cooker-Black/50213889 [accessed Dec. 15, 2021].

[55] “Polycarbonate - an overview,” *Prospector Knowledge Center*, Feb. 26, 2021. https://knowledge.ulprospector.com/11442/pe-polycarbonate-an-overview/ (accessed Dec. 15, 2021).

[56] “Amazon.com: ELEGOO UNO Project Super Starter Kit with Tutorial and UNO R3 Compatible with Arduino IDE : Electronics.” https://www.amazon.com/gp/product/B01D8KOZF4/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1 (accessed Dec. 15, 2021).

Appendix

Appendix A: Main and Test Programs Differences

Table A1: Main vs. Test Program

Functionality	**Main Program**	**Test Program**
Piezo Buzzer	Piezo buzzer sounds when temperature > 100 degrees fahrenheit for longer than 1 hour.	Piezo buzzer sounds when temperature > 100 degrees fahrenheit for longer than 2 minutes.
LED Array	The 6 LEDs begin to light up when temperature > 100 degrees fahrenheit. LEDs light up in 50 degree increments up to 350 degrees fahrenheit.	The 6 LEDs begin to light up when temperature > 100 degrees fahrenheit. LEDs light up in 10 degree increments up to 150 degrees fahrenheit.

Appendix B: Bill of Materials

There are a few important things to note about the bill of materials shown in Table B1. The piezo buzzer (458-1433-ND) listed in the BOM is not the piezo buzzer used in the project. The buzzer used was an active piezo buzzer that Tyler Hendricks owned. The buzzer used in Pantastic can be purchased as part of an electronic starter kit on Amazon.com [56]. The buzzer listed in the BOM is an alternative active piezo buzzer that would work well and can be purchased from Digi-Key. Similarly, the rectifier diode (Reference Designator D1) used in the project was taken from Kai Wong's lab kit that he received in Fundamentals of Electrical Engineering I. The rectifier listed in the BOM is the same diode that was included in the lab kit.

Table B1: Digi-Key Bill of Materials for Single Unit of Pantastic

Index	Quantity	Digi-Key Part Number	Manufacturer Part Number	Description	Package	Reference Designator	Unit Price USD	Total Price USD
1	1	296-45432-ND	CC3220S-LAUNCHXL	LAUNCHPAD DEV BOARD FOR CC3220S	Bulk	N/A	$47.99	$47.99
2	1	MLX90614ESF-BCI-000-TU-ND	MLX90614ESF-BCI-000-TU	SENSOR DGTL -40C-85C TO39	Tube	U1	$49.15	$49.15
3	1	1528-1391-ND	1054	LASER DIODE 650NM 5MW 10MM DIA	Bulk	U4	$5.95	$5.95
4	1	458-1433-ND	SBT200PC	BUZZER PIEZO 44.5MM TH	Bulk	U3	$13.24	$13.24
5	1	439-1010-ND	TL-2100/S	BATTERY LITHIUM 3.6V AA	Bulk	J1	$6.78	$6.78
6	1	EG2015-ND	PS1024ARED	SWITCH PUSH SPST-NO 3A 125V	Bulk	U21	$1.63	$1.63
7	1	EG2011-ND	PS1024ABLK	SWITCH PUSH SPST-NO 3A 125V	Bulk	U22	$1.63	$1.63
8	7	BCP56-16T1GOSCT-ND	BCP56-16T1G	TRANS NPN 80V 1A SOT223	Cut Tape (CT)	Q1, Q2, Q3, Q4, Q5, Q6, Q7	$0.42	$2.94
9	1	ADP122AUJZ-3.0-R7CT-ND	ADP122AUJZ-3.0-R7	IC REG LINEAR 3V 300MA TSOT5	Cut Tape (CT)	U2	$1.47	$1.47
10	2	UCLAMP3301HCT-ND	UCLAMP3301H.TCT	TVS DIODE 3.3VWM 8VC SOD523	Cut Tape (CT)	U5, U6	$0.70	$1.40
11	6	754-1732-ND	WP7113ID5V	LED RED DIFFUSED T-1 3/4 T/H	Bulk	LED7, LED8, LED9, LED10, LED11, LED12	$0.36	$2.16
12	1	S7036-ND	PPPC031LFBN-RC	CONN HDR 3POS 0.1 GOLD PCB	Tray	U24	$0.37	$0.37
13	2	S6106-ND	PPPC102LFBN-RC	CONN HDR 20POS 0.1 GOLD PCB	Tray	U20, U24	$1.26	$2.52
14	4	277-8632-ND	1731471	TERM BLOCK HDR 2POS VERT 3.5MM	Bulk	U4, U22, U21, J1	$0.90	$3.60
15	4	277-5790-ND	1863152	TERM BLOCK PLUG 2POS 90DEG 3.5MM	Bulk	U4, U22, U21, J1	$2.52	$10.08
16	7	764-1183-1-ND	PATT1206E50R0BGT1	RES SMD 50 OHM 0.1% 0.4W 1206	Cut Tape (CT)	R8, R10, R11, R13, R15, R17, R19	$1.69	$11.83
17	2	311-2987-1-ND	RT1206DRE07330RL	RES SMD 330 OHM 0.5% 1/4W 1206	Cut Tape (CT)	R3, R4	$0.25	$0.50
18	2	CRM1206-FX-1001ELFCT-ND	CRM1206-FX-1001ELF	RES SMD 1K OHM 1% 1/2W 1206	Cut Tape (CT)	R5, R6	$0.19	$0.38
19	1	P1.50KFCT-ND	ERJ-8ENF1501V	RES SMD 1.5K OHM 1% 1/4W 1206	Cut Tape (CT)	R18	$0.19	$0.19
20	6	P3.30KFCT-ND	ERJ-8ENF3301V	RES SMD 3.3K OHM 1% 1/4W 1206	Cut Tape (CT)	R1, R2, R9, R12, R14, R16	$0.16	$0.96
21	1	541-3983-1-ND	CRCW120610K0FKEAC	RES SMD 10K OHM 1% 1/4W 1206	Cut Tape (CT)	R7	$0.10	$0.10
22	2	339-C1206C153K5RAC7800CT-ND	C1206C153K5RAC7800	CAP CER 0.015UF 50V X7R 1206	Cut Tape (CT)	C4, C5	$0.34	$0.68
23	2	720-1802-1-ND	VJ1206Y104KXBTW1BC	CAP CER 0.1UF 100V X7R 1206	Cut Tape (CT)	C3, C6	$0.54	$1.08
24	2	399-13374-1-ND	C1206C105K1RACAUTO	CAP CER 1UF 100V X7R 1206	Cut Tape (CT)	C1, C2	$0.51	$1.02
25	1	641-1310-1-ND	1N4001-G	DIODE GEN PURP 50V 1A DO41	Cut Tape(CT)	D1	$0.21	$0.21
26	1	N/A	N/A	PCB	N/A	N/A	$33.00	$33.00
							TOTAL:	$200.86

Table B2: Digi-Key Bill of Materials Per Unit of Pantastic for 10,000 Units Manufactured

Index	Quantity	Digi-Key Part Number	Manufacturer Part Number	Description	Package	Reference Designator	Unit Price USD	Total Price USD
1	1	296-45432-ND	CC3220S-LAUNCHXL	LAUNCHPAD DEV BOARD FOR CC3220S	N/A	N/A	$47.99	$47.99
2	1	MLX90614ESF-BCI-000-TU-ND	MLX90614ESF-BCI-000-TU	SENSOR DGTL -40C-85C TO39	Tube	U1	$39.85	$39.85
3	1	1528-1391-ND	1054	LASER DIODE 650NM 5MW 10MM DIA	Bulk	U4	$5.95	$5.95
4	1	458-1433-ND	SBT200PC	BUZZER PIEZO 44.5MM TH	Bulk	U3	$8.73	$8.73
5	1	439-1010-ND	TL-2100/S	BATTERY LITHIUM 3.6V AA	Bulk	J1	$4.08	$4.08
6	1	EG2015-ND	PS1024ARED	SWITCH PUSH SPST-NO 3A 125V	Bulk	U21	$1.08	$1.08
7	1	EG2011-ND	PS1024ABLK	SWITCH PUSH SPST-NO 3A 125V	Bulk	U22	$1.08	$1.08
8	7	BCP56-16T1GOSCT-ND	BCP56-16T1G	TRANS NPN 80V 1A SOT223	Tape & Reel (TR)	Q1, Q2, Q3, Q4, Q5, Q6, Q7	$0.08	$0.56
9	1	ADP122AUJZ-3.0-R7CT-ND	ADP122AUJZ-3.0-R7	IC REG LINEAR 3V 300MA TSOT5	Cut Tape (CT)	U2	$0.69	$0.69
10	2	UCLAMP3301HCT-ND	UCLAMP3301H.TCT	TVS DIODE 3.3VWM 8VC SOD523	Tape & Reel (TR)	U5, U6	$0.25	$0.50
11	6	754-1732-ND	WP7113ID5V	LED RED DIFFUSED T-1 3/4 T/H	Bulk	LED7, LED8, LED9, LED10, LED11, LED12	$0.16	$0.96
12	1	S7036-ND	PPPC031LFBN-RC	CONN HDR 3POS 0.1 GOLD PCB	Tray	U24	$0.17	$0.17
13	2	S6106-ND	PPPC102LFBN-RC	CONN HDR 20POS 0.1 GOLD PCB	Tray	U20, U24	$0.67	$1.34
14	4	277-8632-ND	1731471	TERM BLOCK HDR 2POS VERT 3.5MM	Bulk	U4, U22, U21, J1	$0.63	$2.52
15	4	277-5790-ND	1863152	TERM BLOCK PLUG 2POS 90DEG 3.5MM	Bulk	U4, U22, U21, J1	$1.76	$7.04
16	7	764-1183-1-ND	PATT1206E50R0BGT1	RES SMD 50 OHM 0.1% 0.4W 1206	Tape & Reel (TR)	R8, R10, R11, R13, R15, R17, R19	$0.64	$4.48
17	2	311-2987-1-ND	RT1206DRE07330RL	RES SMD 330 OHM 0.5% 1/4W 1206	Tape & Reel (TR)	R3, R4	$0.03	$0.06
18	2	CRM1206-FX-1001ELFCT-ND	CRM1206-FX-1001ELF	RES SMD 1K OHM 1% 1/2W 1206	Tape & Reel (TR)	R5, R6	$0.02	$0.04
19	1	P1.50KFCT-ND	ERJ-8ENF1501V	RES SMD 1.5K OHM 1% 1/4W 1206	Tape & Reel (TR)	R18	$0.02	$0.02
20	6	P3.30KFCT-ND	ERJ-8ENF3301V	RES SMD 3.3K OHM 1% 1/4W 1206	Tape & Reel (TR)	R1, R2, R9, R12, R14, R16	$0.02	$0.12
21	1	541-3983-1-ND	CRCW120610K0FKEAC	RES SMD 10K OHM 1% 1/4W 1206	Tape & Reel (TR)	R7	$0.01	$0.01
22	2	339-C1206C153K5RAC7800CT-ND	C1206C153K5RAC7800	CAP CER 0.015UF 50V X7R 1206	Tape & Reel (TR)	C4, C5	$0.07	$0.14
23	2	720-1802-1-ND	VJ1206Y104KXBTW1BC	CAP CER 0.1UF 100V X7R 1206	Tape & Reel (TR)	C3, C6	$0.13	$0.26
24	2	399-13374-1-ND	C1206C105K1RACAUTO	CAP CER 1UF 100V X7R 1206	Tape & Reel (TR)	C1, C2	$0.13	$0.26
25	1	641-1310-1-ND	1N4001-G	DIODE GEN PURP 50V 1A DO41	Tape & Box (TB)	D1	$0.03	$0.03
26	1	N/A	N/A	PCB	N/A	N/A	$7.25	$7.25
							TOTAL:	$135.21

Appendix C: Project Costs

Table C1: First Part Order and Budget

Index	Quantity	Part Number	Manufacturer Part Number	Description	Customer Reference	Unit Price	Total Price
1	2	MLX90614ESF-BCI-000-TU-ND	MLX90614ESF-BCI-000-TU	SENSOR DGTL -40C-85C TO39	5 Guys One Capstone Project	$49.15	$98.30
2	1	EG4699-ND	PR144C1900	SWITCH PUSH SPST-NO 16A 125V	5 Guys One Capstone Project	$2.30	$2.30
3	1	EG2011-ND	PS1024ABLK	SWITCH PUSH SPST-NO 3A 125V	5 Guys One Capstone Project	$1.63	$1.63
4	1	102-3752-ND	CPT-1255C-090	BUZZER PIEZO 12MM TH	5 Guys One Capstone Project	$1.01	$1.01
5	1	1528-1391-ND	1054	LASER DIODE 650NM 5MW 10MM DIA	5 Guys One Capstone Project	$5.95	$5.95
6	2	1.5KE6.8ALFCT-ND	1.5KE6.8A	TVS DIODE 5.8VWM 10.5VC DO201	5 Guys One Capstone Project	$0.46	$0.92
7	2	439-1010-ND	TL-2100/S	BATTERY LITHIUM 3.6V AA	5 Guys One Capstone Project	$6.78	$13.56
8	12	A106011CT-ND	CFR100J1K0	RES 1K OHM 5% 1W AXIAL	5 Guys One Capstone Project	$0.28	$3.36
9	6	BCP56-16T1GOSCT-ND	BCP56-16T1G	TRANS NPN 80V 1A SOT223	5 Guys One Capstone Project	$0.42	$2.52
10	2	ADP122AUJZ-3.0-R7CT-ND	ADP122AUJZ-3.0-R7	IC REG LINEAR 3V 300MA TSOT5	5 Guys One Capstone Project	$1.22	$2.44
11	6	53J50RE-ND	53J50RE	RES 50 OHM 5% 3W AXIAL	5 Guys One Capstone Project	$0.67	$4.02
12	2	399-17286-ND	C440C105M5U5TA	CAP CER 1UF 50V Z5U AXIAL	5 Guys One Capstone Project	$0.39	$0.78
13	1	399-9878-1-ND	C322C104M1U5TA7301	CAP CER 0.1UF 100V Z5U RADIAL	5 Guys One Capstone Project	$0.54	$0.54
14	2	RSMF3JT10K0CT-ND	RSMF3JT10K0	RES 10K OHM 5% 3W AXIAL	5 Guys One Capstone Project	$0.40	$0.80
15	10	754-1732-ND	WP7113ID5V	LED RED DIFFUSED T-1 3/4 T/H	5 Guys One Capstone Project	$0.36	$3.58
16	2	399-C410C153M5R5TA-ND	C410C153M5R5TA	CAP CER 0.015UF 50V X7R AXIAL	5 Guys One Capstone Project	$0.44	$0.88
17	2	RSMF2JT330RCT-ND	RSMF2JT330R	RES 330 OHM 5% 2W AXIAL	5 Guys One Capstone Project	$0.25	$0.50
18	1	296-45432-ND	CC3220S-LAUNCHXL	LAUNCHPAD DEV BOARD FOR CC3220S	5 Guys One Capstone Project	$47.99	$47.99
						TOTAL:	**$191.08**
						Budget:	**$500**
						Budget Remaining:	**$308.92**

Table C2: Second Part Order and Budget

Index	Quantity	Part Number	Manufacturer Part Number	Description	Customer Reference	Unit Price	Total Price
1	3	S7036-ND	PPPC031LFBN-RC	CONN HDR 3POS 0.1 GOLD PCB	5 Guys One Capstone Project	$0.37	$1.11
2	3	S7035-ND	PPPC021LFBN-RC	CONN HDR 2POS 0.1 GOLD PCB	5 Guys One Capstone Project	$0.33	$0.99
3	4	S6106-ND	PPPC102LFBN-RC	CONN HDR 20POS 0.1 GOLD PCB	5 Guys One Capstone Project	$1.26	$5.04
4	5	277-8632-ND	1731471	TERM BLOCK HDR 2POS VERT 3.5MM	5 Guys One Capstone Project	$0.90	$4.50
5	4	277-2416-ND	1844210	TERM BLOCK HDR 2POS 90DEG 3.5MM	5 Guys One Capstone Project	$0.98	$3.92
6	4	ED10554-ND	OSTTJ0211530	TERM BLOCK PLUG 2POS STR 3.5MM	5 Guys One Capstone Project	$1.25	$5.00
7	5	277-5790-ND	1863152	TERM BLOCK PLUG 2POS 90DEG 3.5MM	5 Guys One Capstone Project	$2.52	$12.65
8	5	541-3983-1-ND	CRCW120610K0FKEAC	RES SMD 10K OHM 1% 1/4W 1206	5 Guys One Capstone Project	$0.10	$0.50
9	15	CRM1206-FX-1001ELFCT-ND	CRM1206-FX-1001ELF	RES SMD 1K OHM 1% 1/2W 1206	5 Guys One Capstone Project	$0.16	$2.46
10	5	311-2987-1-ND	RT1206DRE07330RL	RES SMD 330 OHM 0.5% 1/4W 1206	5 Guys One Capstone Project	$0.25	$1.25
11	10	764-1183-1-ND	PATT1206E50R0BGT1	RES SMD 50 OHM 0.1% 0.4W 1206	5 Guys One Capstone Project	$1.50	$14.97
12	4	399-13374-1-ND	C1206C105K1RACAUTO	CAP CER 1UF 100V X7R 1206	5 Guys One Capstone Project	$0.51	$2.04
13	3	720-1802-1-ND	VJ1206Y104KXBTW1BC	CAP CER 0.1UF 100V X7R 1206	5 Guys One Capstone Project	$0.57	$1.71
14	4	399-1240-1-ND	C1206C153K5RACTU	CAP CER 0.015UF 50V X7R 1206	5 Guys One Capstone Project	$0.35	$1.40
						TOTAL:	**$57.54**
						Budget:	**$308.92**
						Budget Remaining:	**$251.38**

Table C3: Third Part Order and Budget

Index	Quantity	Digi-Key Part Number	Manufacturer Part Number	Description	Customer Reference	Unit Price	Total Price
1	1	1528-1391-ND	1054	LASER DIODE 650NM 5MW 10MM DIA	5 Guys One Capstone Project	$5.95	$5.95
2	10	339-C1206C153K5RAC7800CT-ND	C1206C153K5RAC7800	CAP CER 0.015UF 50V X7R 1206	5 Guys One Capstone Project	$0.24	$2.44
3	10	P3.30KFCT-ND	ERJ-8ENF3301V	RES SMD 3.3K OHM 1% 1/4W 1206	5 Guys One Capstone Project	$0.16	$1.57
4	10	P1.50KFCT-ND	ERJ-8ENF1501V	RES SMD 1.5K OHM 1% 1/4W 1206	5 Guys One Capstone Project	$0.16	$1.57
						TOTAL:	**$11.53**
Index	Quantity	Mouser Part Number	Manufacturer Part Number	Description	Customer Reference	Unit Price	Total Price
1	1	595-CC3220S-LAUNCHXL	CC3220S-LAUNCHXL	WiFi Development Tools - 802.11 SimpleLink CC3220 Wi-Fi Launchxl	5 Guys One Capstone Project	$53.19	$106.38
						TOTAL:	**$117.91**
						Budget:	**$251.38**
						Budget Remaining:	**$133.47**

Table C4: Bonus Part Order and Budget

Index	Quantity	Part Number	Manufacturer Part Number	Description	Customer Reference	Unit Price	Total Price
1	5	UCLAMP3301HCT-ND	UCLAMP3301H.TCT	TVS DIODE 3.3VWM 8VC SOD523	5 Guys One Capstone Project	$0.70	$3.50
2	5	BCP56-16T1GOSCT-ND	BCP56-16T1G	TRANS NPN 80V 1A SOT223	5 Guys One Capstone Project	$0.38	$1.90
3	1	EG2015-ND	PS1024ARED	SWITCH PUSH SPST-NO 3A 125V	5 Guys One Capstone Project	$1.63	$1.63
4	3	ADP122AUJZ-3.0-R7CT-ND	ADP122AUJZ-3.0-R7	IC REG LINEAR 3V 300MA TSOT5	5 Guys One Capstone Project	$1.22	$3.66
5	2	296-LM317T/LF01-ND	LM317T/LF01	IC REG LIN POS ADJ 1.5A TO220-3	5 Guys One Capstone Project	$2.32	$4.64
6	5	505-ADP1710AUJZ-3.0-R7CT-ND	ADP1710AUJZ-3.0-R7	IC REG LINEAR 3V 150MA TSOT5	5 Guys One Capstone Project	$1.39	$6.95
						TOTAL:	$22.28
						Budget:	$133.47
						Budget Remaining:	$111.19

Looking at Table C5, the test points used on the PCB were provided to us at no cost. The 3D printed PLA case for the project was printed by Tyler Hendricks on his 3D printer, and it is estimated that the case costs $5.50 to produce. The two PCBs developed for the project were purchased from Advanced Circuits for $33 each. Finally, It is estimated that the PCB assembly from WWW Electronics, Inc (3W) cost $20.20. This is assuming that it costs $5 up-front and $0.40 for each component soldered. In total we got 3W to solder 38 components for our project:

$$Total\ Cost = 5 + (0.4 * 38)$$

$$Total\ Cost = \$20.20$$

Table C5: Extra Costs and Budget

Item	Quantity	Description	Cost	Total Cost
Test Points	12	Test Points for PCB	Free	Free
3D Printed Case	1	3D Printed PLA Case to Protect Pantastic	$5.50	$5.50
PCB	2	PCB from Advanced Circuits	$33	$66
PCB Assembly	1	3W PCB Assembly	$20.20	$20.20
			TOTAL:	$91.70
			Budget:	$111.19
			Budget Remaining:	$19.49

Appendix D: Images of Pantastic

Figure 28: Pantastic Mounted on Wall In case

Figure 29: Pantastic in Case Side View

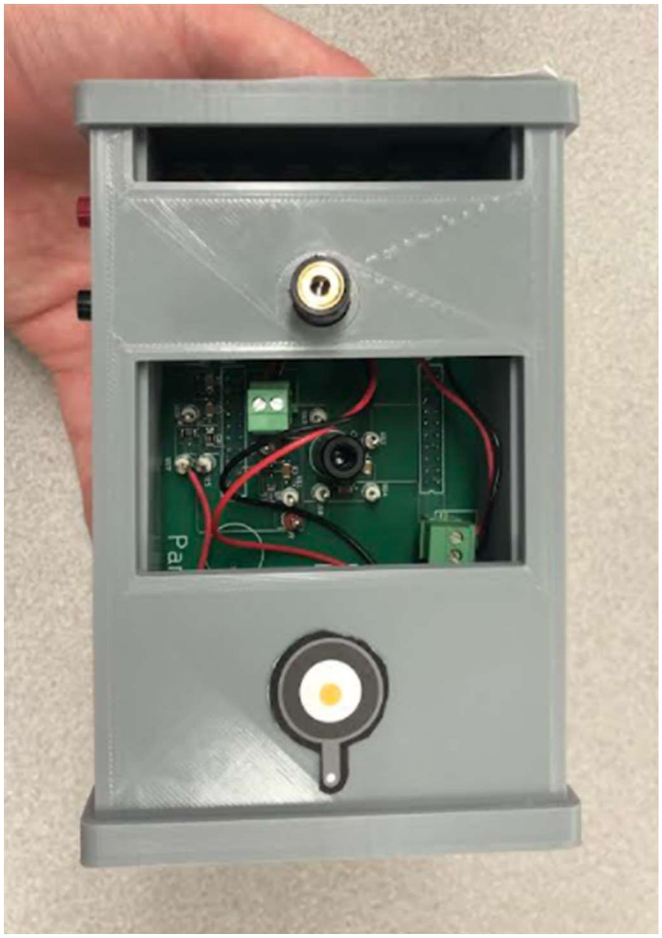


Figure 30: Pantastic in Case Top View (1)

Figure 31: Pantastic In Case Top View (2)

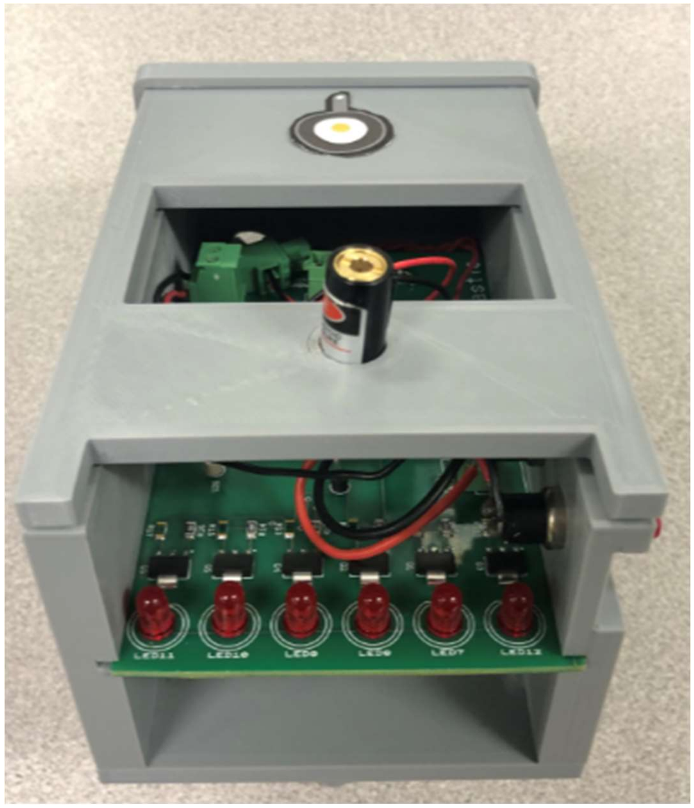


Figure 32: Pantastic in Case Front View

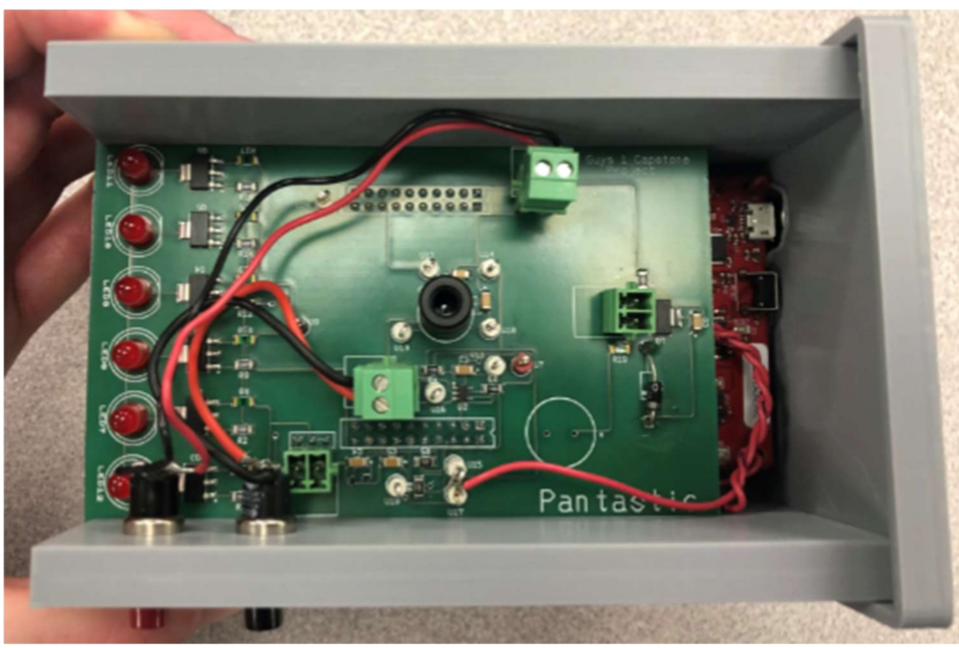


Figure 33: Pantastic in Case Top View Exposed

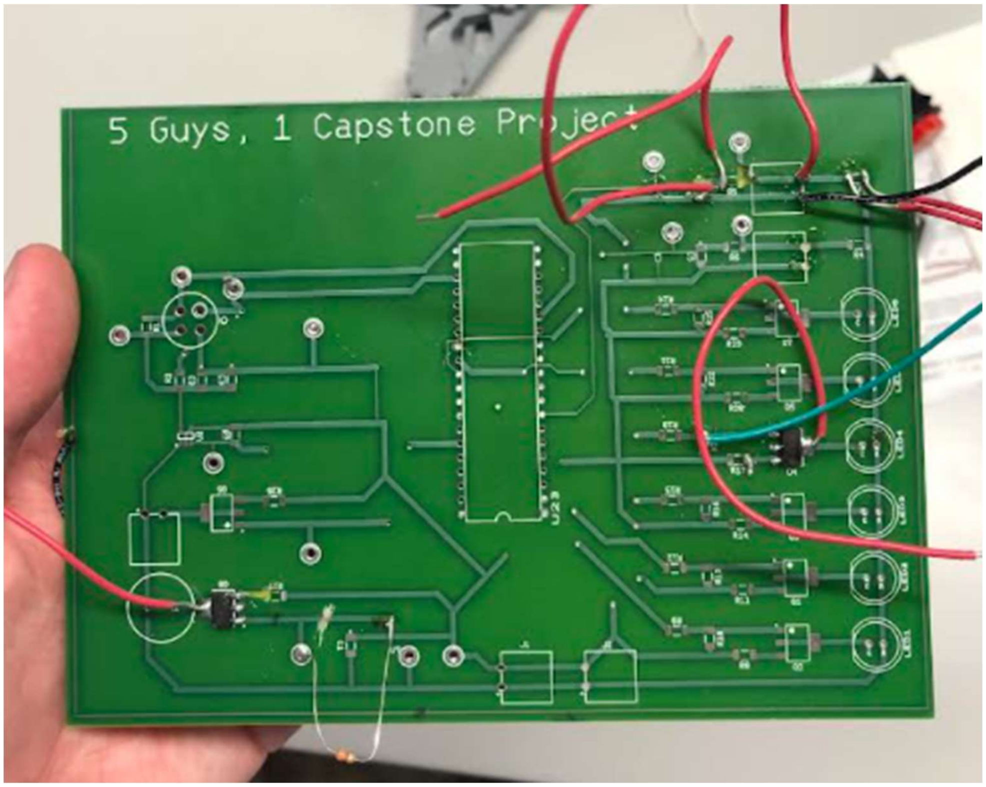


Figure 34: Pantastic Test PCB