ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Automated Heated Bridge System

Team #52

KAHMIL HAMZAT
(khamza2@illinois.edu)
ADRIEL TAPARRA
(taparra2@illinois.edu)
JAMES RAUE (jdraue2@illinois.edu)

TA: Jiankun Yang

May 7, 2025

Abstract

The Automated Heated Bridge System aims to enhance safety in cold, wet environments by using heat to automatically prevent ice formation on bridge surfaces. An ESP32 microcontroller is used to monitor environmental conditions using two DS18B20 temperature sensors and a moisture sensor. When both low temperatures and moisture are detected, the system activates heating. This system is designed to be energy efficient and low maintenance, and it also features wireless monitoring capabilities, allowing real-time status updates and remote oversight. This provides a reliable solution for reducing accidents on icy bridges.

Contents

1 Introduction **1**
1.1 Problem Statement 1
1.2 Solution 1
1.2.1 Functionality 1

2 Design **2**
2.1 Block Diagram 2
2.2 Physical Design 2
2.3 PCB Layout 4
2.4 Power Subsystem 5
2.5 Heating Subsystem 7
2.6 Sensing Subsystem 8
2.7 Control Subsystem 9

3 Design Verification **13**
3.1 Power Subsystem 13
3.1.1 Requirement 13
3.1.2 Verification 13
3.2 Heating Subsystem 14
3.2.1 Requirement 14
3.2.2 Verification 14
3.3 Sensing Subsystem 15
3.3.1 Requirement 15
3.3.2 Verification 15
3.4 Control Subsystem 16
3.4.1 Requirement 16
3.4.2 Verification 16

4 Costs and Schedule **17**
4.1 Costs 17
4.2 Schedule 17

5 Conclusion **19**
5.1 Accomplishments 19
5.2 Uncertainties 19
5.3 Future Work 19
5.4 Ethical Considerations 19

References **20**

Appendix A **21**

1 Introduction

1.1 Problem Statement

The purpose of our project is to solve the problem of bridges freezing faster than the rest of the road. While a normal road is insulated from the sides and bottom, bridges are exposed to the cold air of the environment from all sides, making them freeze faster. Current solutions to this problem include using signs to inform drivers of the hazard, as well as manual salting and plowing of the roads. Both of these solutions are flawed. Using signs is a passive solution that doesn't actually solve the problem. Manual salting and plowing of the roads requires manual labor, which can introduce a significant lag time between the icing of the bridge and when the ice is cleared. This also requires the workers to enter potentially hazardous conditions, such as blizzards or freezing rain, in order to clear the roads.

1.2 Solution

Our solution to this problem was an automatic heated bridge. It will determine when there is ice or snow on the surface of the bridge. When it is determined that there is snow or ice, then it will melt the ice in order to clear it off the bridge surface and clear the road.

1.2.1 Functionality

The bridge system takes in temperature and precipitation data in order to determine when there are conditions when there is ice or snow on the surface of the bridge. This data is processed by an ESP32 microcontroller, which makes the decision of whether or not the bridge needs to be cleared of ice/snow. When it does need to be cleared, it activates heating cartridges in the bridge surface, generating heat and melting the ice.

2 Design

In this section, we will outline the various design choices of the bridge system, including various components and subsystem descriptions.

2.1 Block Diagram

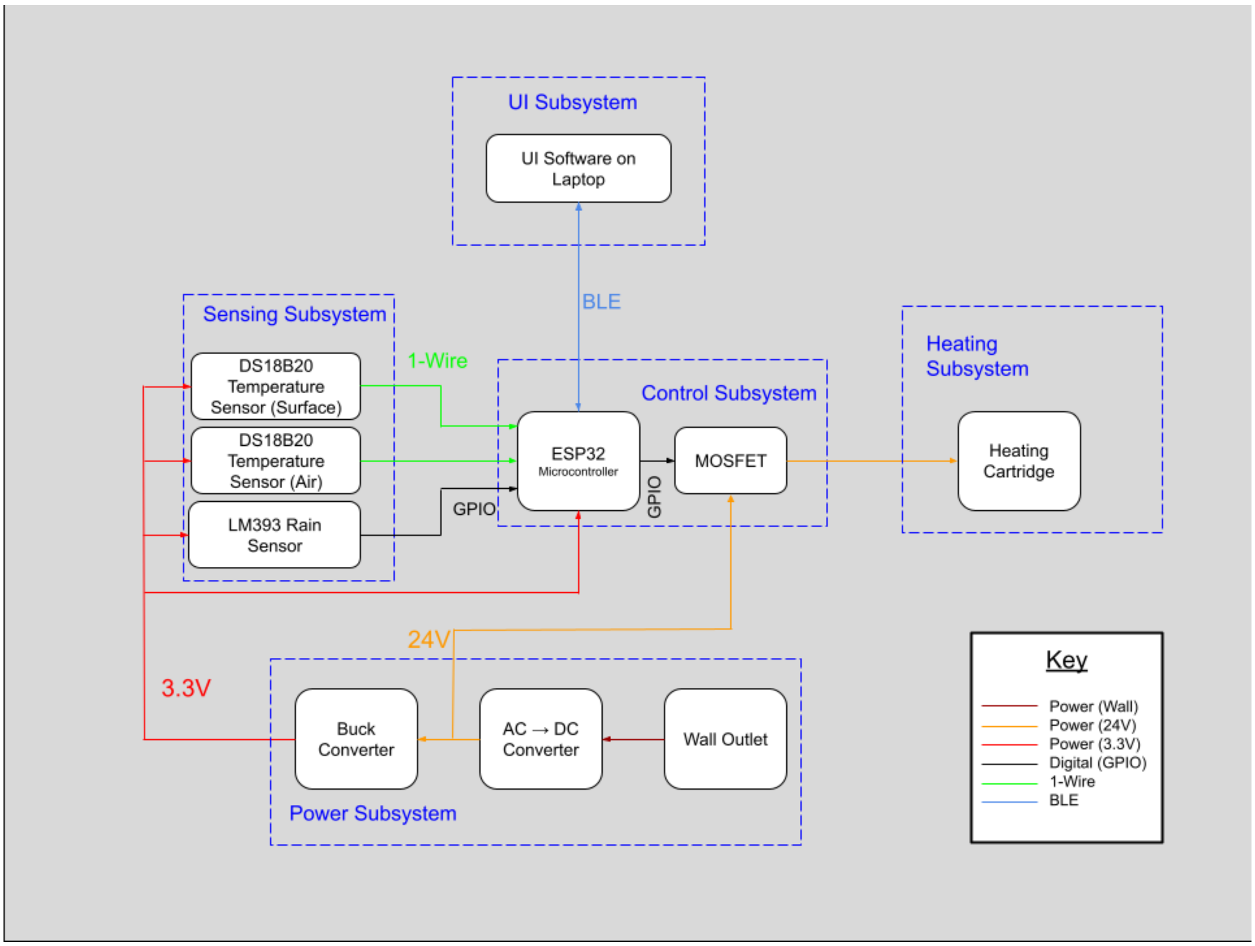

Figure 1: Block Diagram

Figure 1 shows the block diagram. This diagram shows the components within each subsystem, and how they interact with each other.

2.2 Physical Design

The physical model that was created for testing can be seen in Figure 2.

There are several notable design features that we can see in Figure 2. The first is the rain sensor, which is the black rectangle mounted in the top-left. We chose to tilt the sensor at a 45 degree angle. This design choice was made so that water would not pool on the

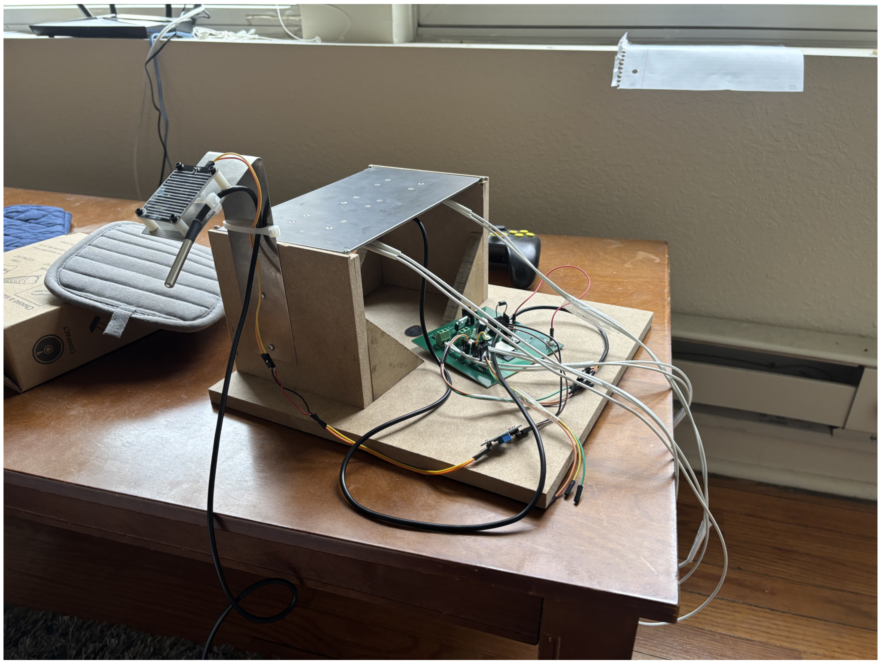

Figure 2: Model Design

sensor once snow and ice melt, and artificially extend the conditions for activating the heater.

Slightly beneath the rain sensor is the air temperature sensor. It was placed in this location because this was best location for keeping it away from the heating cartridges, or any material that is being heated by them. This results in the reported temperature only being influenced by the environment and not the system;s heating.

The surface of the bridge is made out of stainless steel. This material was chosen because it was the closest material to the steel that a full scale bridge would be made out of, as well as having a good thermal conductivity.

On the underside of the bridge, the heating cartridges and surface temperature sensor are mounted as seen in Figure 3. The physical model that was created for testing can be seen in Figure 3. Copper was chosen as the material to hold all of the material, due to its high thermal conductivity of 385 W/m·K [1]. This allows for the heat to conduct faster from the cartridges to the bridge surface and from the bridge surface to the temperature sensor.

The two heating cartridges are the components mounted on the two ends of the underside of the bridge with the white wires. We chose to use two because although one would generate sufficient heat for our purposes, it would take much longer to conduct to the farthest points away from the cartridges. With two, we can heat the surface faster and more uniformly. They are place 2.5″ from either end of the 10″ long bridge so that any point on the bridge is at most 2.5″ away from a heating cartridge (along the x axis).

The surface temperature sensor was placed in the center of the bridge so that it is at the

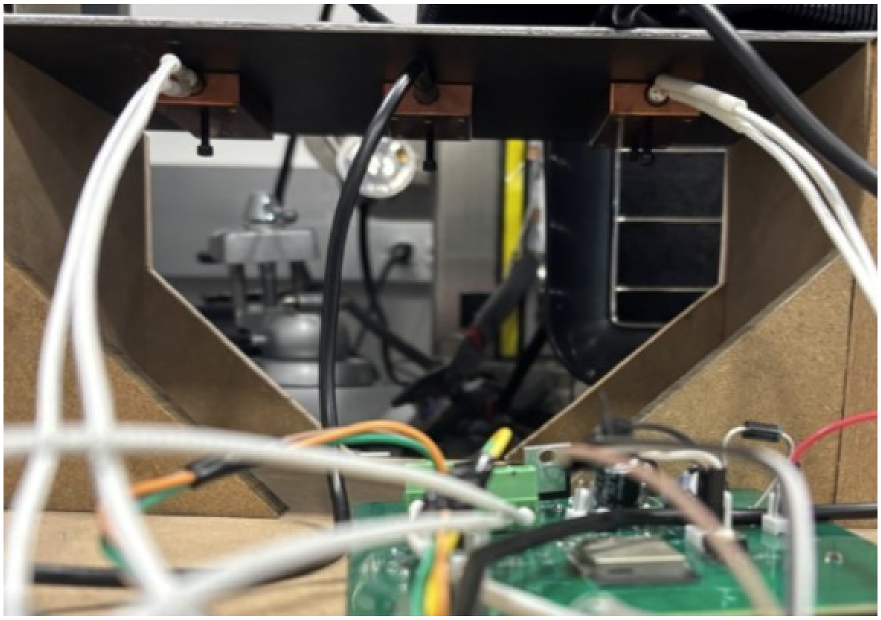

Figure 3: Model Design Underside

furthest point away from he heating cartridges. This allows for us to have the data from the temperature sensor report the coldest temperature on the bridge surface, so that it does not report that the bridge is sufficiently heated when some areas may not be.

2.3 PCB Layout

The PCB Layout in Figure 4 was constructed in a way that the subsystems that require 24V of power and larger trace widths were separated from the subsystems that require 3.3V of power. This is so it is easier to test each individual subsystem separately if needed and for safety as well. Once testing is finished, a jumper is used to connect the 3.3V output from the power subsystem (*BC_Out* connector in Figure 4) and use it to power the control and sensing subsystems (*Control Vin* connector in Figure 4).

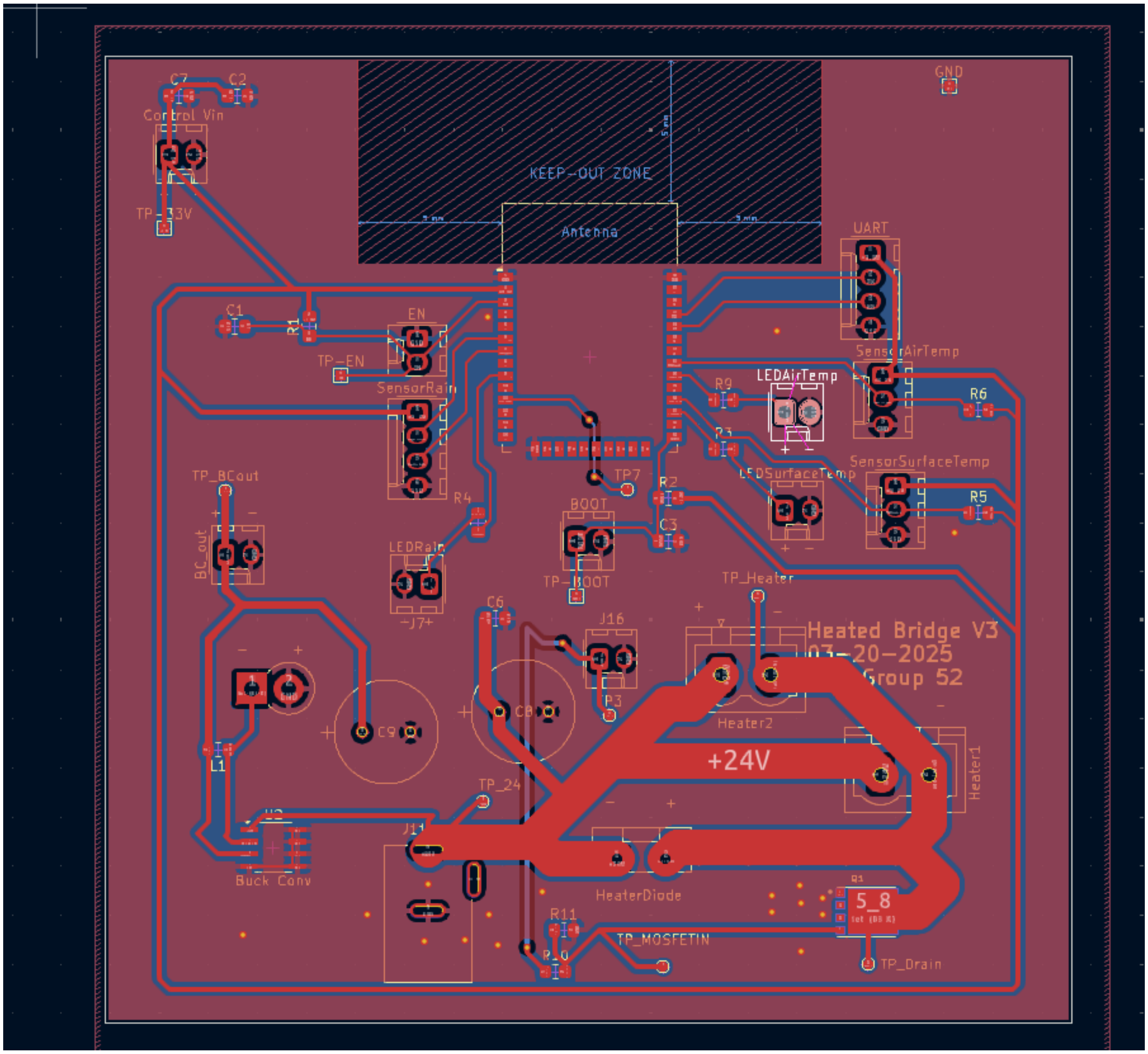

Figure 4: PCB Layout

2.4 Power Subsystem

The power subsystem supplies power to all our components, ensuring that all components get the right amount of power for pristine functionality. This subsystem has connections to the control and sensing subsystem to ensure that we can accurately sense weather conditions and turn on the heater as needed.

As illustrated in the block diagram, the power subsystem starts from the wall outlet where we plug in our AC-to-DC adapter. The adapter converts the 120V AC from the wall into a regulated 24V DC power rail that we use to power the heaters, ESP32 processor, and the sensors. One branch of the 24V DC from the adapter goes through a MOSFET switching circuit (controlled by the ESP32) to the heater cartridges. The other branch goes through a buck converter to supply 3.3V to the microcontroller and the sensors connected to it. The goal of this subsystem is to supply stable power to drive two heater cartridges rated at 70W, 24V each, while also supplying a low-voltage 3.3V for the microcontroller and its peripherals.

For safety reasons, we have chosen an AC/DC adapter [2] that can automatically convert the 100/220V AC from the wall to a regulated 24V DC rail to avoid dealing with dangerous alternating currents and high voltages. We have also chosen this AC/DC adapter because of its efficiency, protection features (overvoltage, undervoltage, overcurrent, overtemperature, short-circuit, and overload protection), anti-interference magnetic ring, and its compact appearance. The adapter produces a maximum load current of 8A and an output wattage of 192W, which provides us with ample power to heat our bridge in a reasonable amount of time. Initially, we planned to use a 6A, 144W adapter, but since the heaters alone draw about 5.6A of current, we thought it would be best to create a buffer, thereby ensuring that the cartridges heat up fully and all other components get enough power.

Additionally, the adapter needs a way to connect to our PCB, so we added a connector [3] to our designs. The connector we have chosen is a 5.5 by 2.1 mm barrel connector because it matches the connector of the adapter (also 5.5 by 2.1 mm). It also has a maximum operating voltage of 48V, which means it can handle the output voltage from the adapter.

As previously stated, our project will be using a buck converter to drop the 24V DC from the power adapter to the 3.3V required to power the microcontroller and associated circuitry. Here, we had to decide whether to use a buck converter or a voltage regulator, and we chose a buck converter because of a few advantages. First, Buck converters operate by switching a MOSFET on and off, thereby transferring energy through an inductor. This allows them to achieve efficiencies that can exceed 80%. A linear regulator, on the other hand, would drop excess voltage as heat, leading to extreme heat and wasted energy in our design. Secondly, a buck converter keeps our system cooler and helps us maximize the power we get from the adapter rather than wasting it as heat, especially since we need a lot of power for our heaters. The following calculations show the difference in efficiency between a voltage (linear) regulator and a buck converter, assuming a 2A load:

Linear Regulator

For a linear regulator stepping down from 24 V to 3.3 V at a load current of 2 A:

$$V_{\text{drop}} = 24\,\text{V} - 3.3\,\text{V} = 20.7\,\text{V}, \tag{1}$$

$$P_{\text{loss}} = V_{\text{drop}} \times I = 20.7\,\text{V} \times 2\,\text{A} = 41.4\,\text{W}. \tag{2}$$

Buck Converter

Assuming an efficiency of 75%:

$$P_{\text{out}} = 3.3\,\text{V} \times 2\,\text{A} = 6.6\,\text{W}, \tag{3}$$

$$P_{\text{in}} \approx \frac{P_{\text{out}}}{\eta} = \frac{6.6\,\text{W}}{0.75} \approx 8.8\,\text{W}, \tag{4}$$

$$P_{\text{loss}} = P_{\text{in}} - P_{\text{out}} \approx 8.8\,\text{W} - 6.6\,\text{W} \approx 2.2\,\text{W}. \tag{5}$$

From these calculations, we can observe that the buck converter is about 18 times more efficient, which is why we have chosen it for our project.

We have specifically chosen the XL1509-3.3 [4] because it's a specialized converter that converts to 3.3V exclusively, regardless of input voltage. The XL1509 buck converter has a fixed 150kHz switching frequency, low dropout, and a maximum current of 2A, which is enough to drive the microcontroller and sensors. It also has an efficiency of 75% which is very high compared to a voltage regulator and allows us enough power for our heater cartridges. The schematic for the XL1509-3.3, constructed based on the typical application from the datasheet, is shown in Figure 5:

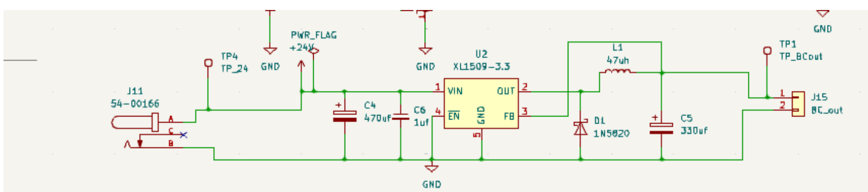

Figure 5: Buck Converter Schematic

Additionally, the power subsystem requires that we consider the PCB traces and make sure that they are large enough to carry the required current and prevent unexpected voltage drop and overheating. Based on our current design, we need PCB traces that can handle both 6A and 3A, which are the current output from the adapter and the required current for the heater cartridges, respectively. Based on KiCad calculations, we need a 140-mils track width and a 54-mils for both current amounts. This is an extra consideration we had to make for our PCB design.

Finally, note that the power subsystem also connects to a high-power switching block, which controls power delivery to the heater cartridges using an N-Channel MOSFET. The functionality of this block will be explained in the Control Subsystem.

2.5 Heating Subsystem

We have chosen two 24V, 70W heater cartridges [5] for our project. The cartridges are embedded inside a copper heating block which are placed on the two opposing sides of the stainless steel bridge surface. We chose copper because it is a good heat conductor and because of the expert advice of the machine shop. We also had the option of 24V, 40W heater cartridges, but we chose 70W because it helps us achieve the same amount of power using fewer cartridges and fewer connectors on our PCB.

Using a combined 140W power to heat the bridge allows us to raise the temperature of the surface by 10 degrees Fahrenheit in under 2 minutes. Each heater will draw about 2.9A of current at 24V, so the two heaters in parallel will draw about 6A of current, which is less than the amount of current supplied by our adapter. A MOSFET will be used to switch the 24V power supply on/off as described in the Control Subsystem section. Figure 6 illustrates the schematic setup of the heating subsystem as well as the MOSFET connection.

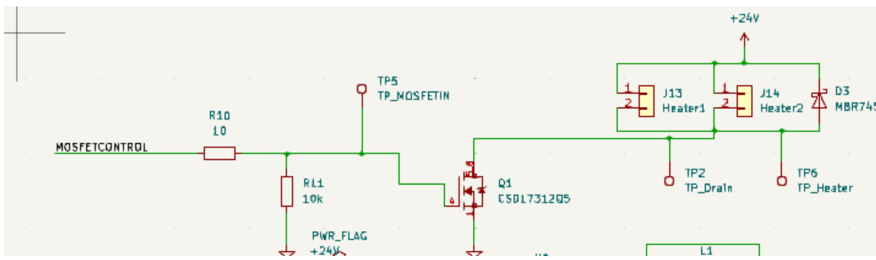

Figure 6: Heating Subsystem Schematic

2.6 Sensing Subsystem

The sensing subsystem processes the environmental data that is relevant to the detection of freezing and hazardous conditions. The environmental data that was deemed necessary to detect was the temperature of the air, the temperature of the surface of the bridge, and the surrounding moisture of the environment. The DS18B20 temperature sensor [6] was chosen for both the air temperature and the surface temperature, and the 333044 Simple Rain Sensor [7] was chosen to detect the surrounding moisture of the environment.

DS18B20 Temperature Sensor

The DS18B20 Temperature Sensor was chosen due to its 3.3V power requirement, which can easily be integrated with the power and control subsystems. For the control subsystem to interface with the temperature sensor, only one GPIO Pin on the ESP32 is needed per sensor due to the 1-Wire Interface protocol. The sensor is also waterproof, which is helpful due to the system being in a cold and moist environment, where water can accumulate or condense on components.

333044 Simple Rain Sensor

The 333044 Simple Rain Sensor was chosen due to its 3.3V power requirement which can easily be integrated with the power and control subsystems as well. For the control subsystem to interface with the rain sensor, only one ADC channel is needed on the ESP32.

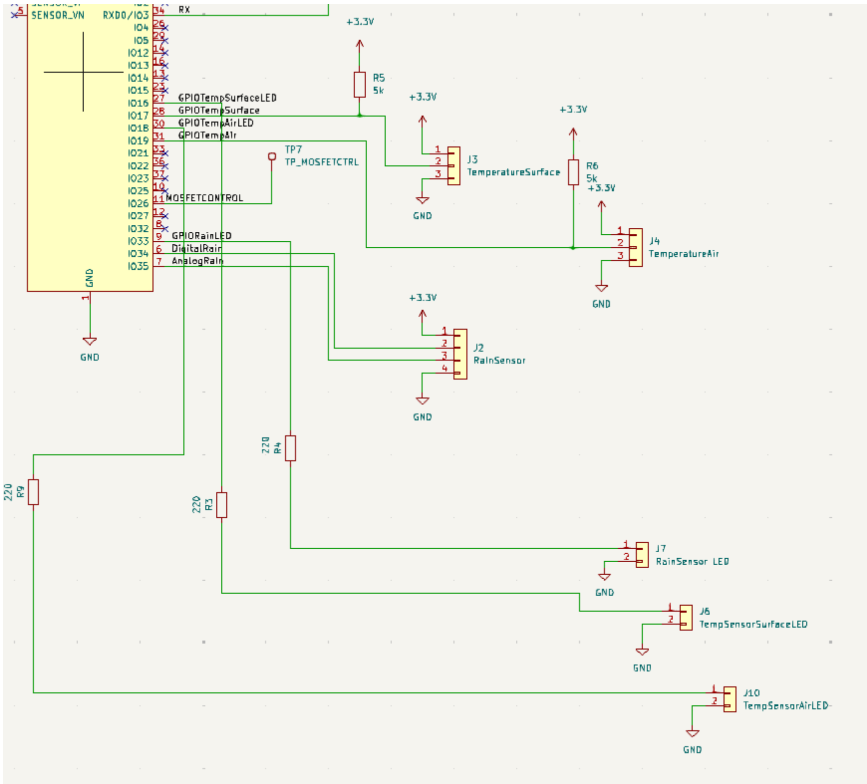

Figure 7: Sensing Subsystem Schematic

The rain sensor will output analogous values from 0 to 4095 depending on the amount of water detected.

2.7 Control Subsystem

The control subsystem is in charge of reading the environmental data from the sensing subsystem and deciding when to change the heater state in the heating subsystem to ON or OFF. The control subsystem consists of an ESP32-WROOM-32E-N4 [8] and a CSD17312Q5 MOSFET [9].

ESP32-WROOM-32E-N4

The ESP32-WROOM-32E-N4 was chosen due to its Bluetooth and / or wireless capabilities. Since our entire system will be operating in extreme weather conditions, it is helpful to have an MCU that is capable of communication without the need for physical connections, which may be unreliable in these environments. Furthermore, the ESP32 has multiple GPIO Pins and ADC channels which will serve as connections for our sensing subsystem so that the ESP32 can read the values and for our heating subsystem so that the ESP32 has the means to change the state of the heaters.

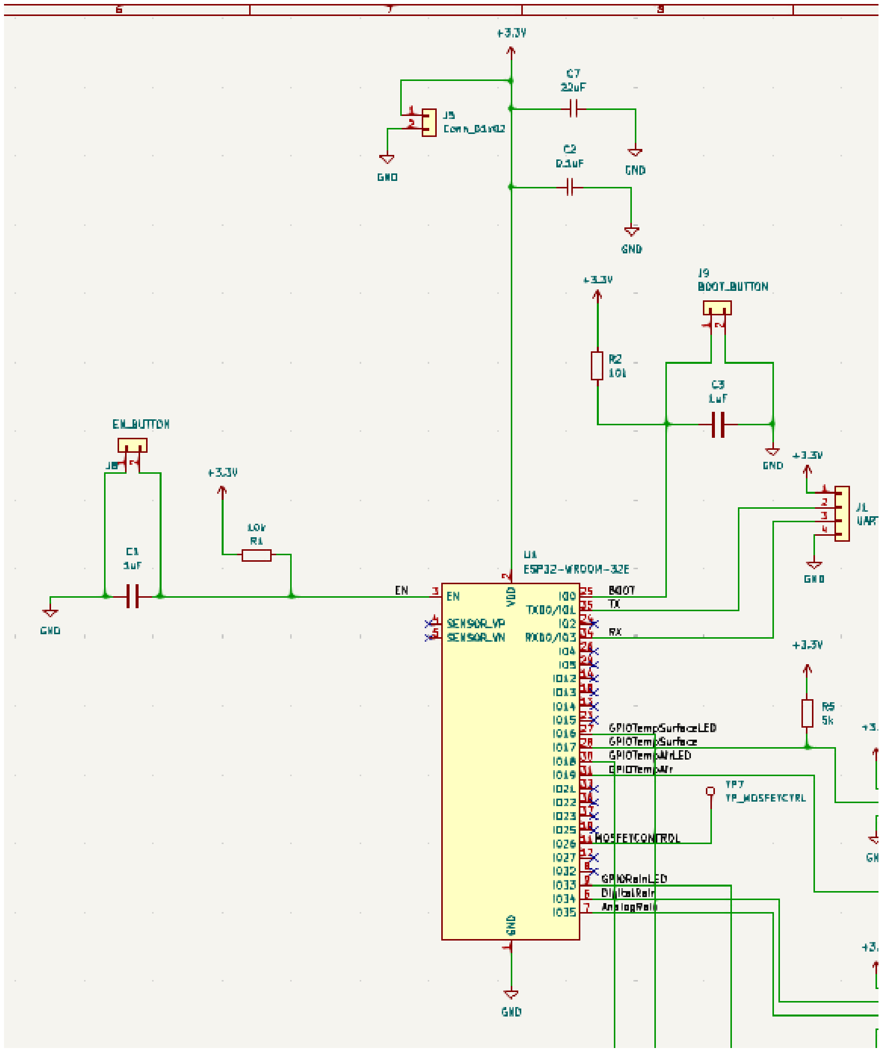

Figure 8: Control Subsystem Schematic

CSD17312Q5

The MOSFET acts as an electronic switch to control the power supplied to the heating cartridges. The ESP32-WROOM-32E-N4 will drive the gate of the MOSFET using a GPIO pin, applying a voltage signal to turn the MOSFET on or off. When the GPIO pin outputs a HIGH signal (3.3V), the gate voltage will surpass the MOSFET's gate threshold voltage (Vgs(th)), allowing current to flow from the drain to the source and powering the heating cartridges. When the GPIO pin outputs a LOW signal (0V), the MOSFET will turn off, cutting off power to the heating cartridge. This design allows the control subsystem to efficiently and safely regulate the heating cartridge based on sensor data from the sensing subsystem.

To satisfy the requirements of our project, the chosen MOSFET has to be fully operational when a 3.3V signal is sent to its gate, wide enough for 6A, and cool under pressure (low heat dissipation). Thus, the CSD17312Q5 MOSFET was chosen for its excellence in these areas. The CSD17312Q5 MOSFET comes in the 8-VSON-CLIP package, where four of the eight available pins are allocated to the MOSFET drain. This is useful because it allows us to create wide traces that can carry 6A on the PCB. We initially considered using the regular MOSFETs that come in the TO-220AB package (IRLB8743PBF and IRLZ44NPBF), however, the pins were very close together, which meant that we wouldn't be able to connect a 6A current line (141 mils) to its drain pin. The following snippet from the PCB illustrates the width of the trace widths and why a wide MOSFET is required.

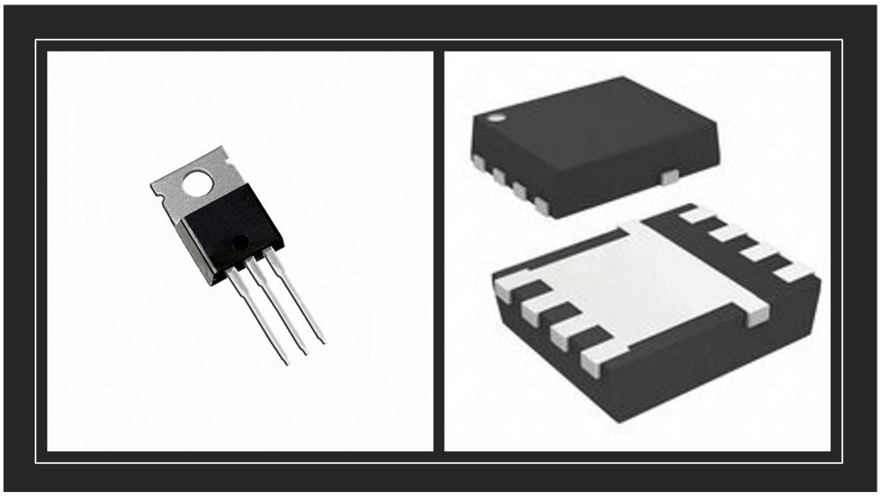

Figure 9: Comparison between the packages for the IRLB8743PBF (left) and the CSD17312Q5 (right) signifying the width of the drain.

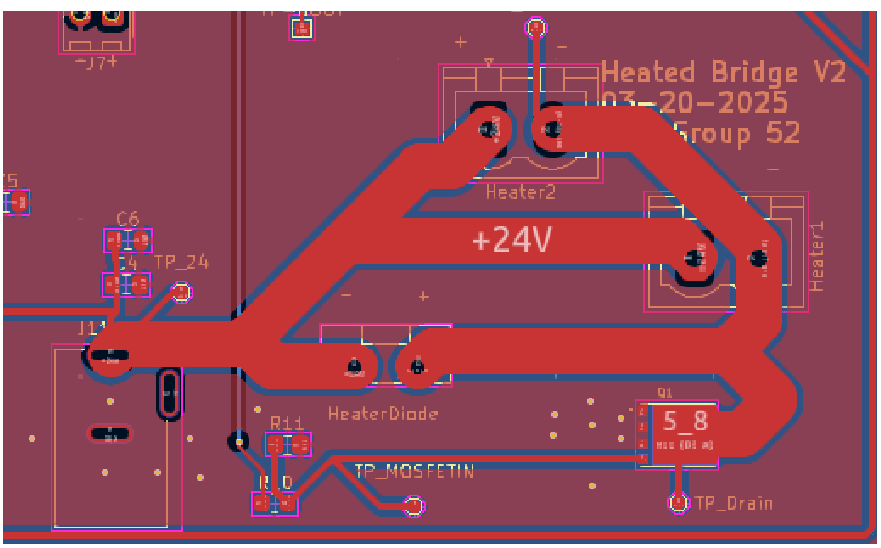

Figure 10: MOSFET PCB design

Additionally, The CSD17312Q5 MOSFET [9]has a maximum drain current (I_D) of 38 A, maximum drain-source voltage (V_{DS}) of 30 V, on-resistance ($R_{\mathrm{DS(on)}}$) of 1.2 mΩ at a gate-source voltage (V_{GS}) of 8 V (much lower than IRLZ44NPBF), and $R_{\mathrm{DS(on)}}$ of approximately

$1.8\,\mathrm{m}\Omega$ at $V_{\mathrm{GS}} = 3\,\mathrm{V}$. It also has a low gate threshold voltage ($V_{\mathrm{GS(th)}}$) since it starts turning on at $0.9\,\mathrm{V}$–$1.5\,\mathrm{V}$ (typical $1.1\,\mathrm{V}$), ensuring full operation at $3.3\,\mathrm{V}$. These specifications ensure that it generates very little heat (about 0.06W), eliminating the need for a heatsink and making it perfectly suitable for our project.

3 Design Verification

In this section, the requirements and verification tables will be expalined in detail for each subsystem.

3.1 Power Subsystem

3.1.1 Requirement

There are two main requirements to verify the functionality of the power subsystem as presented in the Requirements and Verification (R&V) tables in Appendix A. First, the AC/DC adapter must provide 24V ±5% DC. Secondly, the buck converter must output 3.3V ±2%. To facilitate these tests, we added many test points to our PCB so we can easily measure the voltage with a probe. We also isolated the power subsystem by using wired connectors instead of a direct connection through the copper traces of the PCB. This ensures that we can verify the output of the buck converter before feeding it to the ESP32 and sensors.

3.1.2 Verification

To verify the functionality of the power subsystem, we can follow the following procedure:
1. Plug the power adapter into a wall outlet.
2. Insert the barrel jack into the barrel connector, ensuring a secure connection.
3. Take the red and black multimeter test probes.
4. Touch the ground test point with the black test probe and touch the 24V test point with the red probe.
5. Confirm that the voltage is approximately 24V
6. Follow the same probing procedure for the buck converter output test point, verifying that it's approximately 3.3V.
7. Unplug the adapter for safety.

Table 1 has some results:

Table 1: Results of Voltage Measurements

Trial #	Adapter Voltage (V)	Buck Converter Voltage(V)
1	24.028	3.348
2	24.035	3.372
3	24.138	3.335

These numbers all fall within our margin of error, signifying that the power subsystem works as expected.

3.2 Heating Subsystem

3.2.1 Requirement

The main requirement for the heating subsystem is that it raises the temperature of the bridge by 10 degrees Fahrenheit (approximately 5°C) within 5 minutes. This requirement can be verified in two main ways: conducting the overall system test, showing how the temperature of the bridge changes over time, or measuring the voltage across the heaters when they are turned on. The first testing method shows the overall functionality of the heaters, while the second shows that the heaters are operating to the best of their abilities. These test results will be discussed in the verification section.

3.2.2 Verification

For the first test, we can present some data points that show how fast the bridge surface heats up when a full system test is done. This test can also be done without the ESP32 by sending a 3.3V signal to the MOSFET gate through the alternative connectors (just for testing purposes), which will activate the heaters. Once the heaters are on, we can measure the temperature at the center of the bridge using a thermocouple. The center is a safe place for measurement because it's farthest from the two heaters, so we know that when the middle is hot, the other parts of the bridge will be hotter. Table 2 presents some of the results.

Table 2: Surface temperature increase over 5 minutes after heater activation

Elapsed Time (s)	Surface Temp (°C)
0	0.88
30	1.75
60	2.69
90	3.81
120	5.12
150	6.62
180	8.81
210	11.56
240	13.88
270	16.44
300	24.81

These data points show that our heaters surpass the requirement and work as expected.

The second method is to simply verify that the heaters are operating at the appropriate voltage, which can be done by following a similar procedure to the power subsystem verification:

1. Power the PCB through the barrel jack.
2. Use the voltage supply machines to send 3.3V to the MOSFET gate.
3. Take the red and black multimeter test probes.
4. Touch the ground test point with the black test probe and touch the metallic part of the heater with the red probe.
5. Confirm that the voltage is approximately 24V
6. Unplug the adapter for safety.

Some test results in Table 3:

Table 3: Results of Heater Voltage Measurements

Trial #	Heater Voltage (V)
1	23.992
2	24.044
3	24.024

The second set of results further strengthens the hypothesis that the heating subsystem works as expected.

3.3 Sensing Subsystem

3.3.1 Requirement

The sensor subsystem requirement in the R&V table in Appendix A mainly involves making sure that the temperature sensors are accurate within ±1°C in extreme conditions, this is important because this system will need to operate under hazardous conditions and making sure that the rain sensor is able to detect when water is present in the environment.

3.3.2 Verification

Both temperature sensors were connected to the ESP32, which would read and serial print their values every 2 seconds. These values were used for our verifications.

The temperature sensors were verified using ice for the lower bound, the higher bound was not tested since our overheat threshold was 10°C (the heater turns off when the surface temperature reaches 10°C). Furthermore, according to the datasheet [6] the sensor has ±0.5°C accuracy when the temperatures are within -10°C and 85°C. While we could not get to temperatures below -2°C, we were still getting accurate readings using a store-bought thermometer for reference.

To verify the rain sensor, we used a pipette, an ESP32 that prints out the value of the analog output using an ADC pin. Since we used the analog output of the rain sensor, we adjusted the threshold depending on how much water was on the rain sensor. When there is no water on the sensor, the output would be 4095. When dropping 1 mL of water onto the sensor, the output would be around 1900. The threshold of 1800 was chosen based on this value.

3.4 Control Subsystem

3.4.1 Requirement

The main requirement in order to ensure functionality of the control subsystem is that, given the temperature and water level thresholds, the ESP32 shall decide whether to turn the heater ON or OFF after reading the values from the sensor subsystem. The R&V for the Control Subsystem in Appendix A mainly goes through the different conditions that are needed to turn the heaters ON or to turn them OFF.

3.4.2 Verification

We were able to verify this test by showing that the heating signal for the MOSFET was activated only when the temperature crossed the lower threshold and the rain sensor output crossed the lower threshold, and deactivated once the surface temperature surpassed the upper threshold (10 °C) as shown in Table 4. Note that the ... rows indicate a long stretch of values in which the heater state does not change, and no thresholds are passed. As you can see, the heater turns on once the temperature reaches the lower threshold of 3 °C, and turns off once it hits the upper threshold of 10 °C.

Time	Surface Temp (°C)	Rain Sensor Value	Heater State
...	...	...	OFF
1148.9	3.25	2144	OFF
1152.9	2.94	2119	ON
...	...	...	ON
1376.3	9.38	2212	ON
1380.3	10	2219	OFF
...	...	...	OFF

Table 4: Control Subsystem Test Values

4 Costs and Schedule

4.1 Costs

Item	Cost ($)	Purchase Link
Temperature Sensor (DS18B20)	5.99	Link
Water Sensor (LM393)	4.55	Link
ESP32 Microcontroller (5)	21.8	Link
Buttons (13)	3.06	Link
Capacitors (20)	6.17	Link
Connectors (52)	49.17	Link
Diodes (10)	10.95	Link
Inductors (7)	0.63	Link
Resistors (41)	2.25	Link
MOSFETs	9.6	Link
Buck Converter (6)	7.5	Link
AC/DC Converter	14.89	Link
Heater Cartridges	13.99	Link
PCB Orders	70 (estimate)	Link
Dry Ice	60	N/A
Ice	10	N/A
Cooler	100	link
Total	**390.56**	

Table 5: Cost Breakdown of Components

4.2 Schedule

Table 6: Weekly Accomplishments per Team Member

Week	Tasks and Milestones
Week of 3/10	James: Finalized physical model design and got the model made. Kahmil: Initial design of power subsystem schematic. Adriel: Initial PCB design of Control and Sensing Subsystems.
Week of 3/17	Spring Break (No Work)
Week of 3/24	James: Investigated how to make our model waterproof/resistant.. Kahmil: Initial design of Power subsystem PCB. Adriel: Soldered components on to PCB for Control and Sensing Subsystem.
Week of 3/31	James: Investigated how to test and simulate a subzero environment. Kahmil: Worked on choosing the appropriate components on PCB Adriel: Worked on programming the PCB and writing software for ESP32.
Week of 4/7	James: Achieved basic BLE communication with ESP32 from laptop. Kahmil: Fixed capacitor symbols and reordered PCB. Adriel: Verified main event loop is running on the ESP32 .
Week of 4/14	James: Refined the custom BLE protocol and added duplex communication. Kahmil: Soldered the power subsystem section of the new PCB. Adriel: Several independent tests of correct sensor readings and GPIO output of ESP32.
Week of 4/21	James: Began creating UI on laptop Kahmil: Debugged an issue where the heater wasn't working because it was overvolted. Adriel: Soldered new PCB, a few pull up resistors were missing which caused temperature readings to be incorrect.
Week of 4/28	James: Finalized UI. Kahmil: Helped test the functionality of the overall project. Adriel: Tested Control and Sensor Subsystems operate when connecting jumper to the power subsystem's 3.3V output.
Week of 5/5	James: Worked on Final Presentation and Report. Kahmil: Worked on Final Presentation and Report. Adriel: Worked on Final Presentation and Report.

5 Conclusion

5.1 Accomplishments

Our project was successful in its main objective of being able to melt ice based on the value of the sensor input. We were able to raise the temperature from near freezing (1 °C) up to (25 °C) in 5 minutes while triggering the start and stopping of the heaters purely with the sensor value.

5.2 Uncertainties

Although we were successful in the overall goal of our project, there are a few refinements to be made. For one, faulty wiring made the temperature sensor occasionally cut out, and report its default value of -127 °C. In our test run with 478 reports for each of the two temperature sensors, this happened three times. This means that it happened in 0.3% of the total reported values. Another issue we had during this project was simulating a sub-zero environment for testing. We were only able to get a temperature for testing of around 10 °C using dry ice. This prevented us from being able to test our project in the temperatures that it would need to work in the real world.

5.3 Future Work

There are many aspects of our bridge model and system that can be improved in order to take the project further. The PCB that runs the system does not have a permanent design to protect it from the elements, so solving that would be a top priority. We could also test the bridge's decision-making ability by incorporating wireless communication with some sort of weather forecast. We also believe that adding additional sensors would improve measurement accuracy. In our setup, the surface temperature sensor was placed at the center of the bridge, while the heaters were located at both ends. As a result, the sensor could not immediately reflect the temperature changes occurring at the heated ends, since heat needs time to propagate toward the center.

5.4 Ethical Considerations

Since this project is meant as a tool for the general public, the most important ethical consideration is to ensure that our design is sufficiently safe in order to keep in accordance with the IEEE Code of Ethics, Section I.a. [10]. We cannot have any dangerous electrical equipment exposed to the public, have the bridge working incorrectly and creating black ice when melted ice re-freezes, or have the base bridge be unsafe. In addition to this, since our expertise lies solely in electrical and computer engineering, it is important that we consult qualified professionals for aspects of bridge design outside our domain, per the ACM Code of Ethics section 2.6 [11]

References

[1] The Engineering ToolBox. "Thermal conductivity of metals and alloys: Data table & reference guide." (2005), [Online]. Available: https://www.engineeringtoolbox.com/thermal-conductivity-metals-d_858.html.

[2] HUEMIHIU, *24v 8a power supply adapter, ac to dc converter ac 100v-240v to dc 24 volt 8 amp 192w switching transformer led driver with 5.5mm x 2.1/2.5mm dc jack connector*, 2022. [Online]. Available: https://www.amazon.com/dp/B0B1CZYQF8?ref=ppx_yo2ov_dt_b_fed_asin_title&th=1.

[3] G. Electronics, *5.5 mm x 2.1 mm barrel connector datasheet*, Used for interfacing the AC/DC adapter output to the external heater circuitry., 2023. [Online]. Available: https://tensility.s3.us-west-2.amazonaws.com/uploads/pdffiles/54-00166.pdf.

[4] XLSEMI, *Xl1509 2a 150 khz 40v buck dc-dc converter datasheet*, 2007. [Online]. Available: https://www.xlsemi.com/datasheet/XL1509-EN.pdf.

[5] G. Electronics, *24v 70w high temperature heater cartridge (ht-ntc100k thermistor, 1m) for 3d printer sensor heater block v6 j-head hotend (5 pcs)*, 2022. [Online]. Available: https://www.amazon.com/dp/B09MYYP1YS?ref=ppx_yo2ov_dt_b_fed_asin_title&th=1.

[6] E. Systems, *Ds18b2o programmable resolution 1-wire digital thermometer datasheet*, 2019. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/DS18B20.pdf.

[7] Soldered, *Simple rain sensor*. [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/5858/333044%20Simple%20rain%20sensor%20datasheet.pdf.

[8] E. Systems, *Esp32-wroom-32e datasheet*, 2023. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf.

[9] T. Instruments, *Csd17312q5 30v n-channel nexfet power mosfet datasheet*, 2010. [Online]. Available: https://www.ti.com/lit/ds/symlink/csd17312q5.pdf?ts=1743687211123.

[10] Institute of Electrical and Electronics Engineers, *IEEE Code of Ethics*, Accessed: April 3, 2025. [Online]. Available: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf.

[11] Association for Computing Machinery. "Acm code of ethics and professional conduct." (2018), [Online]. Available: https://www.acm.org/code-of-ethics.

Appendix A

Table 7: Power Subsystem – Requirements & Verification

Requirements	Verification
• The AC/DC adapter must provide 24V ±5% DC at loads of up to 6A.	• Use a programmable load or high-power resistor bank to draw up to 6A. • Measure the output voltage with a DMM (digital multimeter) under each load increment. • Confirm that the voltage remains within 24V ±5%.
• The buck converter must output 3.3V ±2% at loads up to 2A.	• Once the 24V is verified, vary input from 20V to 24V (since the output from the adapter will not be 24V exactly). • Measure output with the multimeter at no load and 2A load. • Verify 3.3V ± 2%.

Table 8: Heating Subsystem – Requirements & Verification

Requirements	Verification
• Each heater cartridge must consume 70W ±10% at 24V (2.9A ±10%).	• Power each heater from a 24V supply. • Use a multimeter to measure the current draw. • Calculate Power and verify it falls within 63-77W (±10

Requirements	Verification
• The two heater cartridges must collectively raise the temperature of the heating element by 10°F in ¡2min.	• Embed the heaters in the copper block as planned. • Use the temperature sensor or thermacouple on the copper surface. • Apply 24V to both heaters; start timing when they turn on. • Record the temperature at 15s intervals. • Verify 10°F rise by the 2min mark.
• The heater connections must remain secure and safe (no short circuits).	• Check to make sure the connectors are secure and have correct polarity. • Ensure no shorts to ground prior to powering.

Table 9: Sensing Subsystem – Requirements & Verification

Requirements	Verification
• Temperature sensors must accurately sense the temperature to an accuracy of ±1°C within extreme conditions. Specifically, -30°C to 60°C.	• Place the DS18B20 sensors in a simulated environment. Dry Ice and heated water can be used to achieve extreme environments. • Compare the DS18B20 readings with readings from a calibrated thermometer for reference.

Requirements	Verification
• Rain sensor should be able to detect water in quantities $\geq$ 1mL.	• Using a pipette, drop up to 1mL of water onto the sensor. • Using the esp32 programming, read the signal value in constant time and ensure that it turns high before you use 1mL of water. • If not, adjust the threshold on the sensor.
• Sensors must update frequently, at least once a second.	• Create a subcircuit containing the ESP32, sensors, and UART. • Connect a computer to the ESP32 through the UART. • Verify that the ESP32 is receiving signals at least once a second.
• The system must provide a user interface for monitoring temperature and system status.	• Connect the system to a monitoring display or serial output. • Verify that temperature and status updates are shown correctly.

Table 10: Control Subsystem - Requirements & Verification

Requirements	Verification
• The CSD17312Q5 MOSFET must turn ON the heater cartridge if and only if it receives a 3.3V signal from the ESP32-WROOM-32E-N4.	• Set up a unit test by applying a 3.3V signal to the gate of the CSD17312Q5 MOSFET from the ESP32-WROOM-32E-N4. • Verify the MOSFET is switching properly by measuring the voltage across the heater cartridge when the gate signal is HIGH (3.3V). • Confirm that the heater cartridge turns on only when the gate signal is 3.3V and remains off when the gate signal is 0V.
• The ESP32-WROOM-32E-N4 must signal the CSD17312Q5 MOSFET to turn ON when water is detected and the surface temperature is below 3°C.	• Simulate precipitation using a spray bottle on the rain sensor. • Use dry ice near the surface temperature sensor so that the temperature reading goes below 3°C. • Verify that the GPIO pin of the ESP32-WROOM-32E-N4 connected to the MOSFET gate generates 3.3 V. • Verify that the MOSFET drain voltage is low, indicating that the MOSFET is allowing current to flow through.

Requirements	Verification
• The ESP32-WROOM-32E-N4 must signal the CSD17312Q5 MOSFET to turn ON when water is detected and the air temperature is below 3°C.	• Simulate precipitation using a spray bottle on the rain sensor. • Use dry ice near the air temperature sensor so that the temperature reading goes below 3°C. • Verify that the GPIO pin of the ESP32-WROOM-32E-N4 connected to the MOSFET gate generates 3.3 V. • Verify that the MOSFET drain voltage is low, indicating that the MOSFET is allowing current to flow through.
• The ESP32-WROOM-32E-N4 must signal the CSD17312Q5 MOSFET to turn OFF when moisture is not detected, regardless of the air and surface temperature.	• Simulate dry conditions by ensuring the rain sensor is not exposed to moisture. • Verify that the GPIO pin of the ESP32-WROOM-32E-N4 connected to the MOSFET gate generates 0 V when no moisture is detected. • Test with varying surface and air temperatures (both above and below 3°C) to confirm that GPIO pin of the ESP32-WROOM-32E-N4 connected to the MOSFET gate generates 0 V when no moisture is detected.

Requirements	Verification
• The ESP32-WROOM-32E-N4 must signal the CSD17312Q5 MOSFET to turn OFF when the surface temp is above a 5°C threshold, regardless of the air temperature and rain sensor readings.	• Simulate dry conditions by ensuring the rain sensor is not exposed to moisture. • Set the surface temperature above the 10°C, using a controlled heat source. • Verify that the GPIO pin of the ESP32-WROOM-32E-N4 connected to the MOSFET gate generates 0 V when the surface temperature is above the threshold. • Verify that the MOSFET drain voltage is high, indicating that the MOSFET is OFF and no current is flowing through the heater cartridge. • Test with varying air temperatures and rain sensor states to ensure that the MOSFET remains off solely when the surface temperature is above the threshold.