Heated Bridge System

ECE 445 Design Document – Spring 2025

Project #52

Adriel Taparra, Kahmil Hamzat, James Raue

Professor: Arne Fliflet

TA: Jiankun Yang

Date: March 6, 2025

1 Introduction

1.1 Problem

Bridges pose a significant safety hazard in the winter due to their increased susceptibility to ice compared to standard roadways. Unlike roads, which retain some heat from the ground, bridges are exposed to cold air from all sides, accelerating the freezing process. This causes hazardous driving conditions which can potentially lead to accidents, road closures, and traffic. Currently, the solutions are to put on passive warning signs such as "Bridge Ices Before Road". While the signs inform the driver, it does nothing to prevent ice accumulation. According to the U.S. Department of Transportation, Federal Highway Administration 21% of all vehicle crashes are weather-related [1]. Implementing an automated heating system to prevent ice formation on bridges can significantly enhance road safety by reducing the likelihood of ice-related incidents.

1.2 Solution

Our proposed solution is a heated bridge safety system that actively prevents ice buildup by using an array of heat cartridges embedded beneath the bridge's surface. The system will be controlled by a microcontroller that continuously monitors real-time weather conditions through temperature, moisture, and precipitation sensors. When freezing temperatures and moisture are detected, the system will activate the heating elements, ensuring that ice does not form.

To demonstrate this concept, we will construct a simulated bridge model with a metal sheet to represent the road surface, under which heating cartridges will generate heat. A MOSFET-based power switching circuit will efficiently regulate power delivery to the heating elements. The power supply will be sourced from a 12V or 24V DC adapter, with the potential for a rechargeable battery and DC-DC converter integration. By activating only when necessary, the system will minimize energy consumption while maintaining a safe, ice-free surface. The effectiveness of this prototype will be tested in a controlled environment to evaluate its heating capability and responsiveness to changing conditions.

1.3 Visual Aid

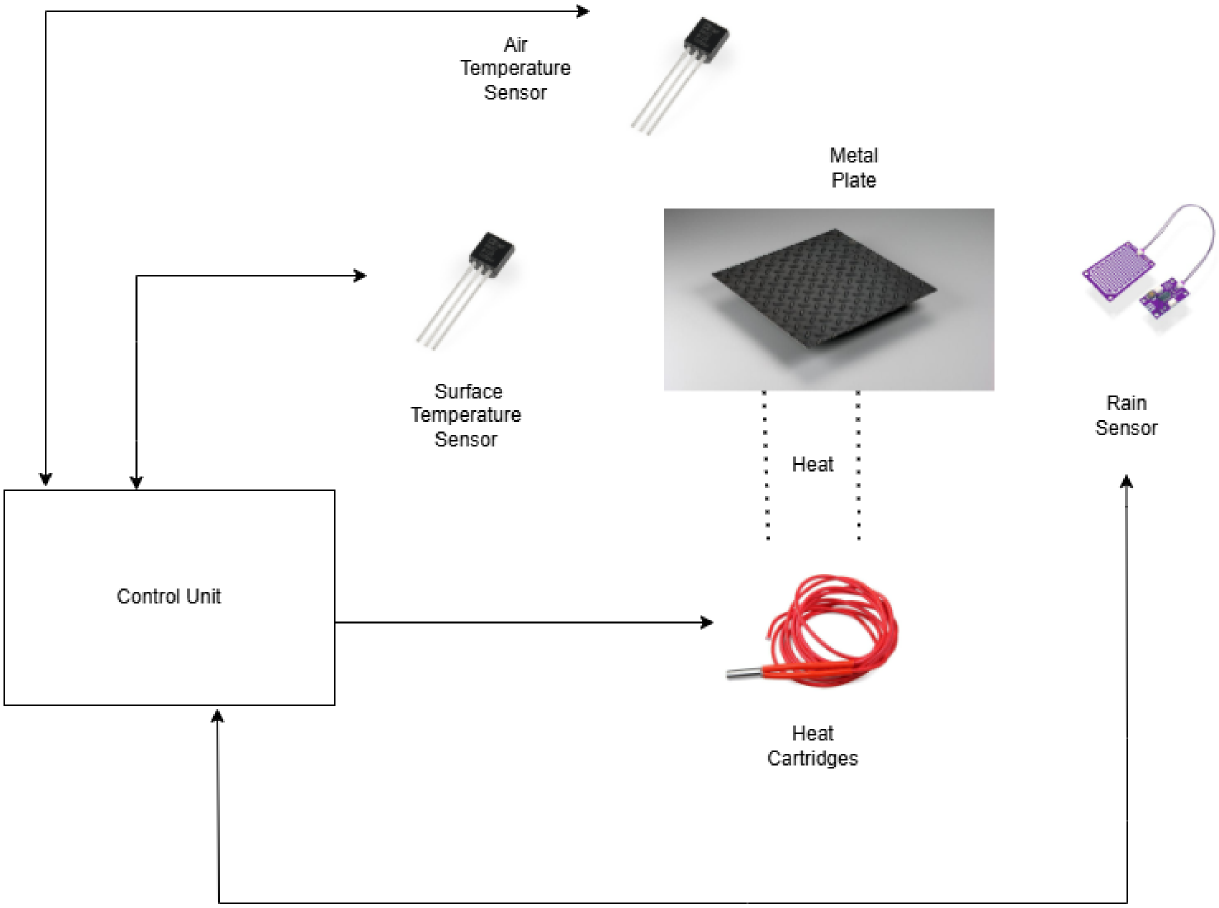

Figure 1: Visual Aid

1.4 High-level requirements list

- **Accurate environmental sensing** – The system must reliably detect surface temperature, moisture presence, and precipitation with an accuracy of at least ±1°C for temperature and a clear binary (wet/dry) output for moisture detection to ensure proper activation of the heating system.
- **Efficient heating capability** – The heating elements must generate enough heat to raise the bridge surface temperature above freezing (0°C) within five minutes of activation in simulated icy conditions.
- **Automated power regulation** – The heating system must activate only when freezing temperatures and moisture are detected and automatically deactivate once conditions are no longer hazardous, optimizing power usage and preventing unnecessary energy consumption.

2 Design

2.1 Block Diagram

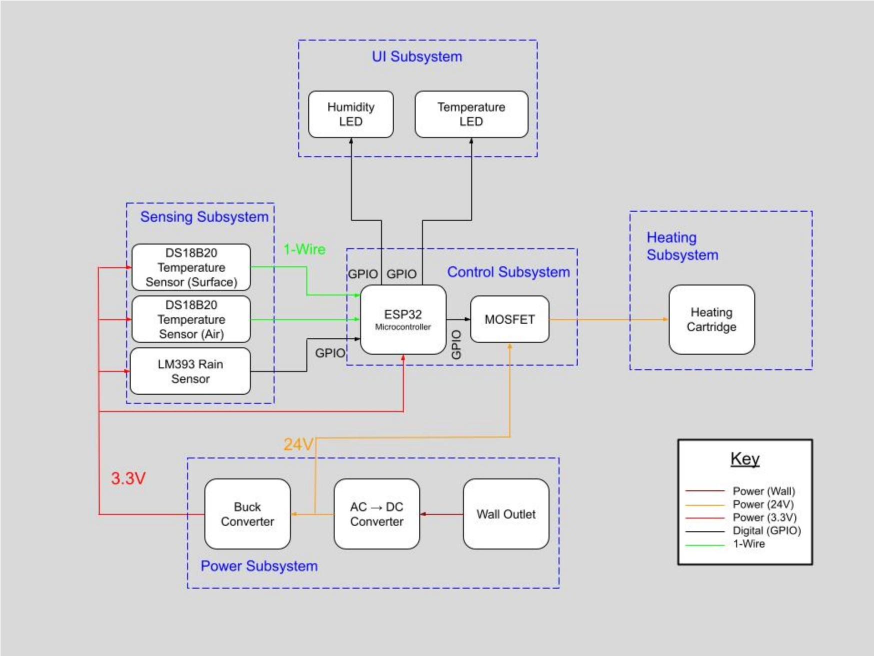

Figure 2: Block Diagram

The above figure shows the block diagram for our system. It is composed of 5 subsystems: The power, sensing, control, UI, and heating subsystems. The power subsystem is responsible for transforming the wall power into the 24V and 3.3V DC power needed by the rest of the system. The Sensing system is in charge of collecting environmental data. The Control subsystem is in charge of determining when to activate the bridge based on data from the sensing subsystem. The Heating subsystem is responsible for heating the bridge enough to melt away any potential snow or ice. The UI subsystem is in charge of giving some insight into the control subsystem by showing if the moisture and temperature thresholds have been passed.

2.2 Subsystem Overview/Requirements

2.2.1 Power Subsystem

The power subsystem is in charge of supplying power to all our components. It ensures that all components get the necessary power they need to function without getting shorted or blowing a fuse. This subsystem has connections to the control and sensing subsystem to ensure that we are able to sense weather conditions accurately 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 power from the wall into a regulated 24V DC power rail that we can use to power the heaters, ESP32, and the connected 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 AC/DC adapter also produces a maximum load current of 6A and an output wattage of 144W, which is sufficient to heat our bridge in a reasonable amount of time.

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 and maximum operating current of 6.0A, which can handle the output voltage and current 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 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 make the most of 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 load current of 2A:

Linear Regulator

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

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

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

Buck Converter

Assuming an efficiency of 75%:

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

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

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

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 its a specialized converter that converts to 3.3V exclusively. This helps simplify the subsystem because we don't need to specify the specific voltage to drop down to (which would introduce more complex circuitry). 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 below:

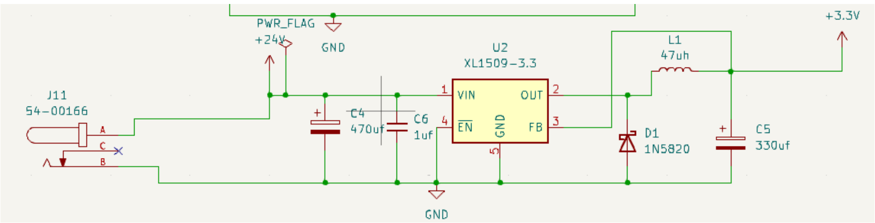

Figure 3: Block Diagram

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 is the current output from the adapter and the required current for the heater cartridges, respectively. Based on KiCad calculations, we need a 140mils track width and a 54mils for both current amounts. This is an extra consideration we need 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.

Table 1: 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%.

2.2.2 Heating Subsystem

We have chosen to use two 24V 70W heater cartridges [5] for our project. Ideally, these cartridges will be embedded on two sides of a copper heating element, which will be right below our 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 will allow us to raise the temperature by 10 degrees Fahrenheit in under 2 minutes, as explained in the Tolerance Analysis section. 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 about the same 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.

Table 2: 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
• 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.

Requirements	Verification
• 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.

2.2.3 Sensor Subsystem

The sensing subsystem is responsible for taking in environmental data such that the rest of the system may determine if the bridge needs to be heated or not. It consists of 2 DS18B20 temperature sensors, and a LM393 Water Sensor. Both types of sensors are powered via the 3.3V DC power outputted by the buck converter in the power subsystem. The outputs of the sensors are then used by the control subsystem in order to determine when to activate the heating system.

The temperature sensor should have an accuracy of ±1C and be able to detect temperatures below the freezing point (0C). The moisture sensor must also be able to detect when a sizable amount of liquid (1mL) falls on it so that we can activate the heater as needed. Both sensors should be operable at real-world temperatures of -30 ° C to 60 ° C.

Table 3: Sensor 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.

DS18B20 Temperature Sensors

Function: The DS18B20 Sensors are used in two instances in our project. The first is to monitor the surface temperature of the bridge, such that we are able to determine whether or not the surface is adequately heated. The second is to measure the temperature of the outside environment.

Connections:

- The sensor outputs data to the ESP32 Microcontroller in the control subsystem using the 1-Wire protocol.

- The sensor takes in 3.3V DC power, which is supplied by the buck converter in the power subsystem.

Design Justification: We chose this model because it fit our requirements for the power and control subsystems as well as being within the error tolerance threshold. Its operating voltage of 3-5.5V and its 1-Wire Protocol [6] fit with the buck converter's output voltage of 3.3V and the ESP32's compatibility with 1-Wire. In addition to this, it has a ±0.5°C Accuracy from -10°C to +85°C[6], which fits within our expected tolerance of ±1°C.

LM393 Water Sensor

Function: Detects water on the surface of the bridge in order to detect rain. This data, combined with environmental temperature, will allow the system to predict freezing rain.

Connections:

- The sensor outputs data in digital format to the ESP32 once it detects water above a certain threshold.
- The sensor receives 3.3V DC power from the buck converter in the power subsystem.

Design Justification: We chose this sensor because of its high compatibility with the other subsystems, as well as its 2 parts, one with the sensor and one with the output. This way, we would not need to have the sensor be near the rest of the PCB circuit.

2.2.4 Control Subsystem

The control subsystem is responsible for processing environmental data from the sensing subsystem and activating the heating cartridge when appropriate. It consists of an and a IRLZ44NPBF MOSFET [7], and ESP32-WROOM-32E-N4 Microcontroller [8].

The ESP32 acts as the central processing unit for the entire system. The ESP32-WROOM-32E-N4 will interface with the two DS18B20 Digital Temperature Sensors using 1-Wire [6]. For the LM393 Water Sensor Evaluation Expansion Board / Rain Sensor [9], the ESP32-WROOM-32E-N4 will interface with the sensor using the digital output for one of the pins with GPIO. For the IRLZ44NPBF MOSFET, the ESP32-WROOM-32E-N4 will interface with it using GPIO to signal it to turn on or off.

The IRLZ44NPBF MOSFET acts as an electronic switch to control the power supplied to the heating cartridge. The ESP32-WROOM-32E-N4 will drive the gate of the IRLZ44NPBF 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 cartridge. 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.

Table 4: Control Subsystem - Requirements & Verification

Requirements	**Verification**
• The IRLZ44NPBF 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 IRLZ44NPBF 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 IRLZ44NPBF MOSFET to turn ON when water is detected and the surface temperature is below 2°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 2°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 IRLZ44NPBF MOSFET to turn ON when water is detected and the air temperature is below 2°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 2°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 IRLZ44NPBF 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 2°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 IRLZ44NPBF 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 5°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.

ESP32-WROOM-32E-N4

The ESP32-WROOM-32E-N4 handles the processing of sensor data, including reading moisture and temperature inputs, and determining whether to activate the heater. It sends a signal to the IRLZ44NPBF MOSFET based on the data from the sensors, ensuring that the heater is only turned on under the correct conditions.

Connections:

- Reads data from the sensing subsystem and controls whether to signal to the MOSFET to turn on the heater or not.

Design Justification: We chose this model because it is well-suited for handling multiple sensor inputs while managing control signals for high-power components like the MOSFET. It is also highly customizable, with integrated Wi-Fi and Bluetooth, and low power consumption [8] for future expansion of the project. .

IRLZ44NPBF

Function: The IRLZ44NPBF is a logic-level MOSFET [7] used to switch the 24V supply to the heater based on a signal from the ESP32-WROOM-32E-N4. When the ESP32 outputs a 3.3V signal to the MOSFET gate, the MOSFET allows current to flow from the 24V supply to the heater, thereby powering the heater when necessary.

Connections:

- Receives a 3.3V signal from the ESP32-WROOM-32E-N4 at its gate.
- Uses the drain-source channel to control the flow of current to the heater based on the gate signal.

Design Justification: We chose this model because it is a logic-level MOSFET, meaning it can be fully activated with a low voltage (3.3V) from the ESP32, according to the datasheet [7]. The IRLZ44NPBF also has a high current rating (47A) and low on-resistance (Rds(on)), ensuring efficient switching and reliable operation in our system, especially given the high-power requirements of the heater.

2.2.5 User Interface Subsystem

The user interface system exists to provide insight into the inner workings of the bridge. It is meant to be used both in debugging the circuit and repairing the system in the future if something goes wrong. For example, if the heating cartridge is not activating when it is in conditions that it is supposed to, the information provided by the subsystem can be a great help in debugging. If the LED's indicate that the microcontroller determined that the heating cartridge should be on, then the problem would be with the MOSFET or heating cartridge. If not, then it would be in the sensing subsystem. It consists of two LED's, a blue led which will activate when adequate water has been detected and a red LED to indicate when the temperature goes below the threshold.

Table 5: User Interface Subsystem - Requirements & Verification

Requirements	**Verification**
• The cold threshold LED must turn on when the detected temperature is less than or equal to 0°C, and must turn off when the temperature rises above 0°C	• Use ice water to create an environment slightly colder than freezing (-2°C). • Submerge the temperature sensors in the ice water. • Ensure that the temperature LED lights up.
• The rain LED must turn on when the rain sensor detects rain and turn off when the moisture level returns to normal	• Submerge the rain sensor in water. • Ensure the rain LED turns on. • Dry off the rain sensor. • Ensure the rain LED is off.
• Each LED must have full functionality in extreme conditions of [-30°C, 60°C]	• Create a simulated extreme climate. Using dry ice and a cooler for a cold climate, and an oven for a hot climate. • Place the system in the simulated environment. If it is a cold environment, ensure the cold LED works. • Test the moisture LED indicator in the environment by putting water on the sensor. • Remove the system from the environment, and test to make sure the circuit has full functionality afterwards.

In order to use the LEDs, we must also pair them with resistors to limit current and prevent them from burning out. We can find the values of resistance using the following equations:

$$V_{\text{src}} - V_{\text{LED,Red}} - V_R = 0$$

$$3.3 - 2.2 - 10mA * R = 0$$

$$R_{\text{Red}} = 120$$

$$V_{\text{src}} - V_{\text{LED,Blue}} - V_R = 0$$

$$3.3 - 2.9 - 15mA * R = 0$$

$$R_{\text{Blue}} = 27$$

Kingbright WP7113ID (Red 5mm LED) Function: This LED activates when the microcontroller detects sufficiently low temperature such that the heating cartridge should be activated

Connections

1. A digital signal from the ESP32 at 3.3V to activate the LED.
2. A series connection with a resistor to limit current.

Justification: This was chosen because it is a very standard LED found in many electronics kits, and it is very cheap. We chose to use LEDs over a more advanced display method, because LEDs get the desired information across just as well as a display, while being cheaper and easier to implement.

Kingbright WP7113CBD (Blue 5mm LED)

Function: This LED indicates when the ESP32 detects enough rain/snow from the environment such that snow and ice may be possible.

Connections

1. A digital signal from the ESP32 at 3.3V to activate the LED.
2. A series connection with a resistor to limit current.

Justification: We chose this for the same reason as the red LED. It is a very standard LED found in many electronics kits, and it is very cheap.

2.3 Tolerance Analysis

For our project to be successful, we need to make sure the heating element produces enough heat to prevent ice formation while also being energy efficient. A potential risk is that the heating element may not reach the required temperature to melt the ice, or we might not be able to produce enough power for heating. We also need to consider how long it will take to raise the surface temperature by a certain amount to make sure that the amount of power we supply can raise the bridge temperature in a reasonable amount of time.

Specifically, we need to be able to heat the copper block by 10°F (approximately 5.56°C) within a specified time frame. Based on the diameter of the heater cartridges (6mm), the dimensions of copper block will be $10'' \times 5'' \times 1''$ so that the cartridges can be safely lodged inside the copper block. To calculate the power needed, we first need to calculate the energy needed using the mass of the block and the constant specific heat of copper. The specific heat capacity of copper is $c \approx 385\,\mathrm{J/(kg\,^\circ C)}$. The mass will be calculated below:

The copper block is designed to have dimensions of 10 inches by 5 inches by 1 inch. We first convert these dimensions to centimeters:

$$10\,\text{inches} \approx 25.4\,\text{cm}$$

$$5\,\text{inches} \approx 12.7\,\text{cm}$$

$$1\,\text{inch} \approx 2.54\,\text{cm}$$

The volume V of the block is then:

$$V = \text{length} \times \text{width} \times \text{height} \approx 25.4\,\text{cm} \times 12.7\,\text{cm} \times 2.54\,\text{cm} \approx 819\,\text{cm}^3$$

Since $1\,\text{m}^3 = 1\,000\,000\,\text{cm}^3$, we convert the volume to cubic meters:

$$V \approx \frac{819\,\text{cm}^3}{1\,000\,000} \approx 0.000819\,\text{m}^3$$

The density ρ of copper is approximately:

$$\rho \approx 8960\,\text{kg/m}^3$$

Thus, the mass m of the copper block is given by:

$$m = \rho \times V \approx 8960\,\text{kg/m}^3 \times 0.000819\,\text{m}^3 \approx 7.35\,\text{kg}$$

This calculation shows that the copper block has a mass of approximately 7.35 kg. We can now calculate the energy required.

The energy required to raise the temperature of the copper block by ΔT is given by:

$$Q = m \cdot c \cdot \Delta T \tag{1}$$

For a temperature increase of $\Delta T = 5.56°\text{C}$, we have:

$$Q = 7.35\,\text{kg} \times 385\,\text{J}/(\text{kg}\,°\text{C}) \times 5.56°\text{C} \approx 15\,735\,\text{J} \tag{2}$$

To determine the required power P to achieve this temperature increase in a time t, we use:

$$P = \frac{Q}{t} \tag{3}$$

We can now calculate the power required for various time intervals:

- **In 1 minute (60 s):**
$$P = \frac{15\,735\,\text{J}}{60\,\text{s}} \approx 262.25\,\text{W}$$
- **In 2 minutes (120 s):**
$$P = \frac{15\,735\,\text{J}}{120\,\text{s}} \approx 131.13\,\text{W}$$
- **In 3 minutes (180 s):**
$$P = \frac{15\,735\,\text{J}}{180\,\text{s}} \approx 87.42\,\text{W}$$
- **In 5 minutes (300 s):**
$$P = \frac{15\,735\,\text{J}}{300\,\text{s}} \approx 52.45\,\text{W}$$
- **In 10 minutes (600 s):**
$$P = \frac{15\,735\,\text{J}}{600\,\text{s}} \approx 26.23\,\text{W}$$

Discussion: These calculations demonstrate that if we want to achieve a rapid temperature increase (e.g., 10°F in 2 minutes), our system must deliver roughly 132 W to the copper block. For slower heating, the power requirement drops accordingly. This analysis is useful for selecting the appropriate heater cartridges and ensuring our power supply can handle the power demands of our project. Furthermore, if the heater is left on for extended periods, the block will continue to heat until thermal equilibrium is reached with its environment.

This analysis confirms that our design must account for significant power when rapid heating is desired, and it guides the selection of power components and heater specifications to meet our project requirements.

3 Cost and Schedule

3.1 Cost

In order to estimate the cost of designing and building this project, we must first estimate the salary of the engineers. Assuming an annual salary of $109,176 [10], and an average of 1,976 working hours in a year[11], we can estimate an hourly salary of $55 an hour. We can then calculate the cost of wages for engineering for the project as such:

3 people × 55 $/hour × person × 100 hrs = 16,500 $

Item	**Cost ($)**	**Purchase Link**
Temperature Sensor (DS18B20)	5.99	Link
Water Sensor (LM393)	4.55	Link
ESP32 Microcontroller	4.84	Link
MOSFET	1.38	Link
Buck Converter	1.21	Link
AC/DC Converter	14.89	Link
Heater Cartridges	13.99	Link
PCB Manufacturing	70 (estimate)	Link
LED (red)	0.20	Link
LED (blue)	0.53	Link
Total	**117.58**	

Table 6: Cost Breakdown of Components

We must also account for the cost of the machine shop. According to Greggory Bennet, the cost of 3 days of work by the machine shop would total to $1,262.70. This leaves us with the total sum of cost from labor and parts to be **$17,880.28**.

3.2 Schedule

We hope to follow the following schedule for the successful implementation of our project:

Week	Tasks and Milestones
Week of 3/10	James: Assemble PCB, debug circuit traces. Kahmil: Verify component placements. Adriel: Run initial firmware tests.
	Spring Break (No Work)
Week of 3/17 Week of 3/24	James: Test PCB #2 for power issues. Kahmil: Validate sensor functionality. Adriel: Get bridge model from machine shop.
Week of 3/31	James: Finalize third PCB order. Kahmil: Integrate PCB with bridge model. Adriel: Begin software development for ESP32.
Week of 4/7	James: Debug third PCB (if needed). Kahmil: Develop sensor communication code. Adriel: Implement MOSFET control logic.
Week of 4/14	James: Conduct full system integration test. Kahmil: Debug sensor readings. Adriel: Implement UI subsystem (LEDs, alerts).
Week of 4/21	James: Mock demo prep: refine hardware. Kahmil: Mock demo prep: finalize software. Adriel: Conduct stress testing.
Week of 4/28	James: Final Demo: Test system under real conditions. Kahmil: Fix any last-minute bugs. Adriel: Document system performance.
Week of 5/5	James: Final Presentation: Create slides. Kahmil: Write final report sections. Adriel: Lab checkout and packaging.

Table 7: Week-by-Week Task Distribution for Each Team Member

4 Ethics and Safety

4.1 Dangerous Conduct

Due to the nature of our project relating to the use of large amounts of energy and heat, it is especially important that we avoid dangerous conduct, as described in the student code 1-302-a. If we are reckless during the design and testing process of our project, we could start a fire which could cause bodily harm or property damage. Because of this, we must take all necessary safety measures when designing, building, and testing our project, which would include:

1. Perform all testing within a controlled lab environment, where fire extinguishers are readily available.
2. Monitor the temperature of circuit elements and the heating cartridge, such that it does not produce excessive heat.

4.2 Public Safety

Because our project is intended for the public, it is important that we incorporate the highest level of safety into our design in accordance with the IEEE Code of Ethics, Section I.a. Failure to do so may cause injury to the public. In order to prevent this, we will be taking the following measures:

1. Testing under various environments with varying temperature and humidity to ensure that our design is reliable in real world use.
2. Properly insulating of dangerous electrical or heating components such that they are inaccessible to the public.
3. Compliance with relevant codes such as the AASHTO LRFD bridge design specifications manual for structural integrity.

4.3 Work Only in Areas of Competence

According to the ACM Code of Ethics Section 2.6, engineers must work only in areas of competence. Our team specializes strictly in electrical and computer engineering. We recognize that there are more areas of knowledge needed to make a safe and functional bridge, such as structural engineering, material science, and thermodynamics expertise. In order to not overstep outside of our area of expertise, we will consult with qualified engineers or faculty for any of these areas of knowledge.

References

[1] U. D. of Transportation Federal Highway Administration. “How do weather events impact roads?” (2024), [Online]. Available: `https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm#:~:text=Weather%20Impacts%20on%20Safety,1%2C235%2C000%20%2D%20are%20weather%2Drelated.`.

[2] AOYADAISU, *24v 6a dc power supply adapter 144w, universal ac to dc converter with 5.5 mm x 2.5 mm & 2.1 mm output jack*, 2022. [Online]. Available: `https://www.amazon.com/dp/B0CFFBMQ6W?ref=ppx_yo2ov_dt_b_fed_asin_title`.

[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] I. Rectifier, *Irlz44npbf power mosfet datasheet*, 2003. [Online]. Available: `https://www.infineon.com/dgdl/irlz44npbf.pdf?fileId=5546d462533600a40153567217c32725`.

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

[10] U. D. of Electrical and C. Engineering. “Salary averages.” (2022), [Online]. Available: `https://ece.illinois.edu/admissions/why-ece/salary-averages`.

[11] Clockify. “Average working hours (statistical data 2025).” (2025), [Online]. Available: `https://clockify.me/working-hours`.