Fire And Smoke Detection With Real-Time Led Navigation

ECE 445 Design Document - Spring 2021

Project # 81
Abel Garcia, Alex Parafinczuk , Jainam Shah

Professor: Yang Zhao
TA: Surya Vasanth

Contents

1 Introduction

1.1 Problem

Fires come unexpectedly at times and can pose a risk in many homes. Whether you have guests over who aren't very familiar with your home, or maybe you are staying at an Airbnb rental home so you are less familiar with exits. In the night, a fire could possibly happen in the middle of your sleep and panic will set in. Commercial smoke detectors in the market currently only give users the ability to call first-responders immediately and sound an alarm where there is a hazard detected in the home. The problem with these smoke detectors is that help isn't immediate, responders take a little while to reach home and smoke can blind your vision during your escape to an exit. If there was a way to at least help the efforts of families getting to safety as quickly as possible, while also trying to combat the fire, then this is a step forward to combating fire. We say this because as of now, commercial smoke detectors only play an alarm when a hazard is detected, some modern alarms even notify first responders and can be integrated with your smart home systems such as Alexa. All of this is great, but there is still the lack of guiding families to the safest exit possible as well as weakening the growth of the fire present. Weakening the fire can be troublesome when you think about it. Depending on the type of fire that occurs, some options are now limited, for example, if an electrical fire occurs, you wouldn't want to throw water near that area to put the fire out. But what exactly causes the fires to grow and rapidly spread across houses? Well, places with a good ventilation system can easily promote the growth of the fire [1]. Sometimes we forget about the true influence that oxygen has on strengthening the spread of the fires, a fire is started on three things, heat, fuel, and oxygen. If any of these increase in concentration/supply, then the fire will increase in strength [2]. So, if we combine a ventilation system which is the way we bring fresh oxygen from the outdoor air into a house which is on fire, we will by default be increasing the growth of this fire and in turn making it harder to protect everything in the house. So, two things become apparent, one, there has to be a way to close off the ventilation system in order to stop the incoming flow of oxygen, and the other is that the safest route to an exit should easily be detectable by the families in the house.

1.2 Solution

In order to address the issues presented above, our solution will implement a twofold plan. First, we will integrate temperature sensors all across the rooms as well as a gas sensor on the main control unit in a central location in the house. The sensor data will be inputted into our ESP-32 Microcontroller which will be in control of reading all the values coming in and detecting, based on the parameters we give, whether or not there is a hazard detected. If there is a hazard, an alarm will get played alerting residents of the

trouble that is found. At the same time, the ESP-32 will be communicating this information to an application interface which will then be able to run an algorithm which will give the ESP-32 necessary information as to where the best path to take is. From this information, the ESP-32 will then light up LEDs on the temperature sensor boards which will indicate 3 of the following colors: Green will indicate the path you should follow, Yellow indicates that while the room is safe, there is no exit to be found in this room, and finally red which indicates to stay clear from this area as there is a fire here. This type of system that we have for the first plan will cover the problem of making sure that no matter how familiar you are with the house, you will always be guided to the best exit possible which will be updated in Real-Time if the fire starts to progress and block other exits which were once thought to be safe. We hope that with this solution, no matter what time of day it is, families will be able to calmly follow the best path possible to safety. The second plan will be to ensure that we are able to stop the supply of oxygen from the outside in order to prevent the spread of fire. This will be done by turning off the HVAC system once a fire is detected as well as closing vents at the same time in order to fully prevent oxygen from seeping into the rooms. Our implementation along with the capabilities that modern smoke detectors have should make for a much safer operation.

1.3 Visual Aid

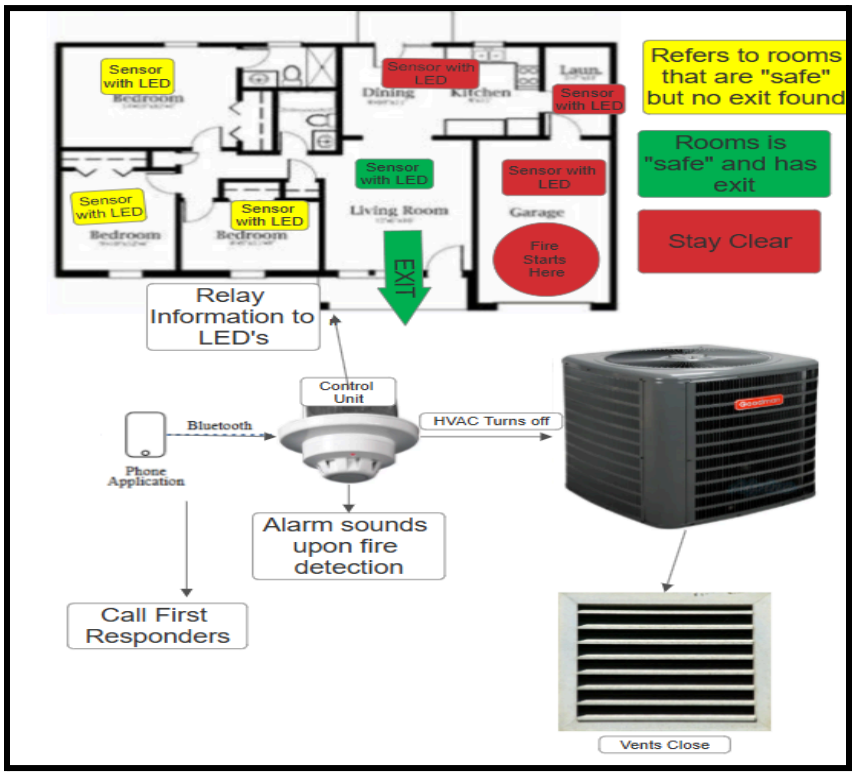

Figure 1: A Model of System Working when Fire is Present

As of now the project will be made to work with only single-story houses, in Figure 1, we have a main control unit which will be in a central location of the house. This Control Unit will be communicating with essentially every component. The application will be in charge of sending signals to our control unit on exactly where the fire or hazard started as well as calling first responders. From this, The control unit will then send the signal to turn the HVAC off as well as closing the vents, at the same time, the control system will be in charge of relaying the information to the LED's to light up the proper colors. For the purposes of this project we will be using three colors which will indicate different circumstances. For instance, green indicates the room is safe and has an exit, on the other hand, yellow will indicate that there is a room but it has no exit, and red indicates that you should stay away from this place as there is a fire present. The plan for assembling everything together will be to wire all our external temperature sensors to the main PCB board which will have the ESP-32 where we will input our analog output too as shown in Figure 2.

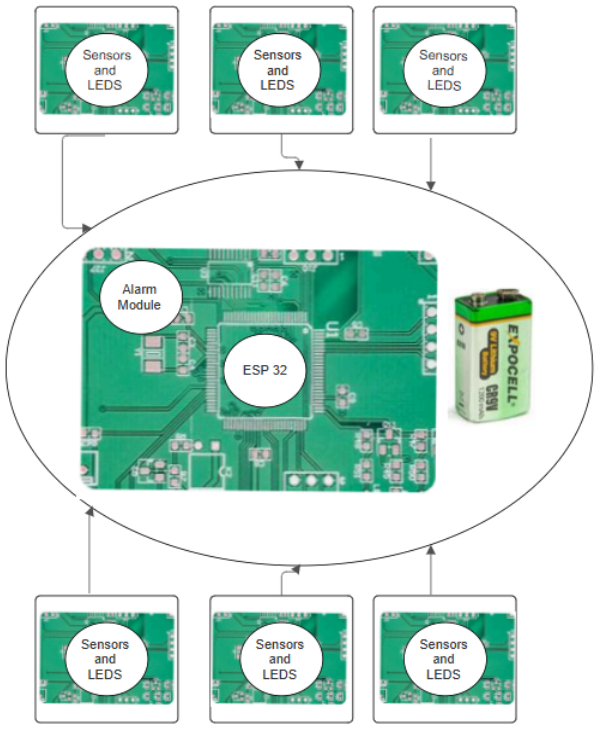

Figure 2: Layout and Connections for PCB Boards.

The general idea is to have one main PCB board which will contain our alarm module, microcontroller, power supply, gas sensor, LED, temperature sensor, and all the other PCB boards will only contain an LED as well as a temperature sensor. And the arrow is meant to display that the input from the temperature will be going into our ESP-32, and it's also worth noting that the ESP-32 itself will

be outputting to our LED in order to light the corresponding color for the situation. The reason for not including gas sensors on the other boards is simply because gas hazards like carbon monoxide, while dangerous, do not require immediate actions such as a fire. Therefore, having the main control unit in a central location in your house will be just fine when it comes to detecting harmful gases like carbon monoxide in a timely manner.

1.4 High Level Requirements

To consider our project successful, our safety suite must fulfill the following:

1. Application will consist of a place where you can layout your floor plan by designating rooms with connections such as doors that lead to other areas in the house along with the location of sensors and exits. The algorithm will isolate rooms that are affected and designate an exit that is furthest from the fire and all of this will be visible with LED's with guaranteed accuracy

2. Once a threshold of 90℃ is reached on the temperature sensors indicating a fire, a signal is sent to the control unit along with an alarm sounding and first responders being alerted to the hazard. The gas sensor will sound the alarm when a Carbon Monoxide (CO) level greater than 20 ppm is present.

3. When a fire is detected the HVAC will turn off and the vents will close at any time a fire is detected, upon clearance of the fire, operation of the HVAC and vents will function as normal.

2 Design

2.1 Block Diagram

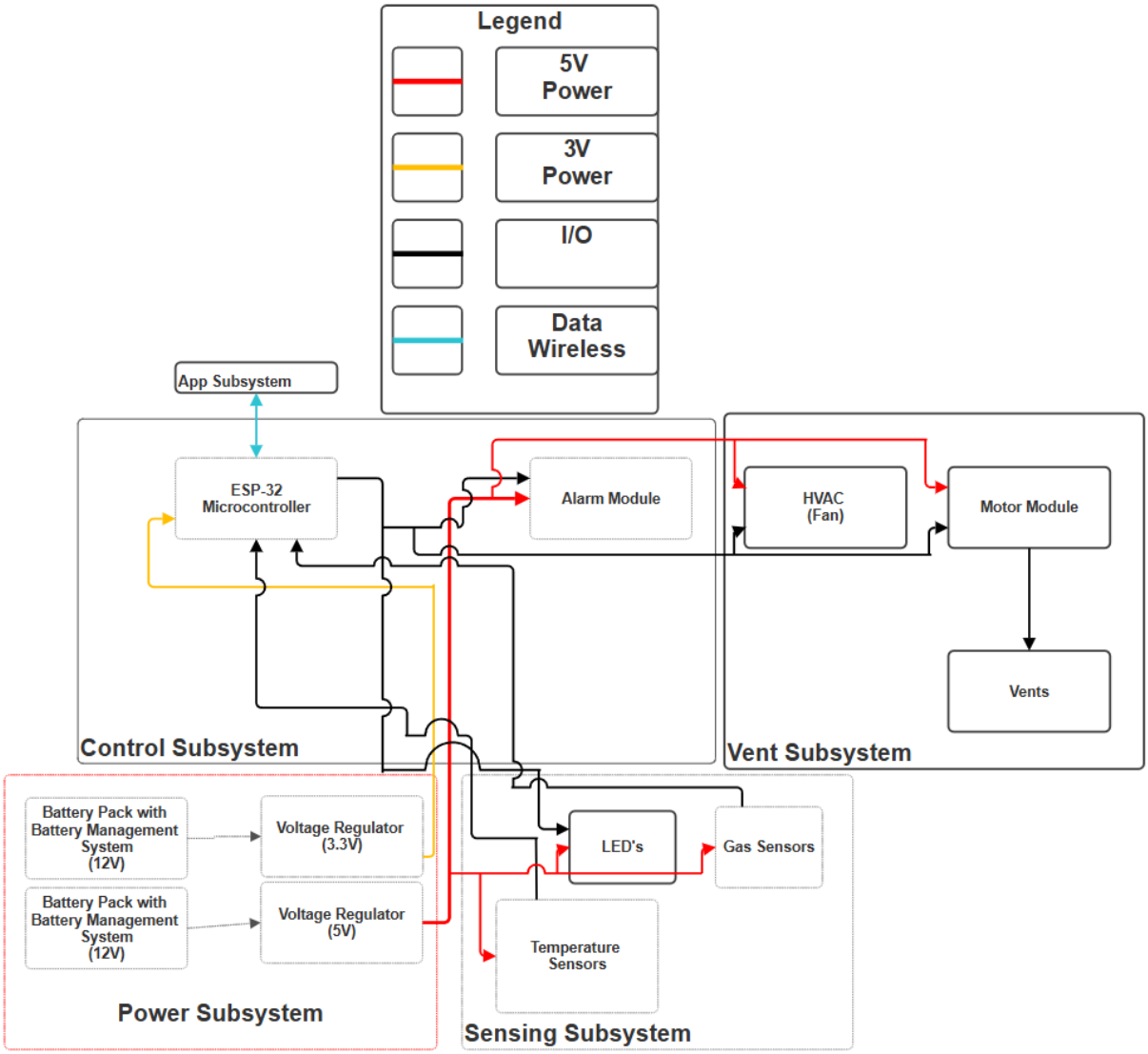

Figure 3: Block Diagram of Project

2.2 Functional Overview & Block Diagram Requirements

2.2.1 Sensor Subsystem

The Temperature and Gas Sensor that we will be using are the foundation of the project, these two sensor will be communicating with our ESP32-S3 by providing an analog voltage output, and we will be converting this ADC value into a readable voltage value that will give us information on the temperature as well as the gas concentration. More information will be provided on how exactly the value is read and

equations used in the __Hardware Section__. The gas sensor that will be used is the MQ-9B sensor, it is equivalent to the MQ-9 sensor in terms of pin connection, the difference seems to be that the MQ-9B sensor is a lesser version of the other sensor in terms of ranges it can handle. The MQ-9 Sensor will be able to measure 10~500 ppm of carbon monoxide and 300~10000 ppm of Methane. The gas sensor will only be located in the main PCB board as action isn't immediately required compared to a fire, and the main PCB board will be in a central location in a single story home, therefore, the need of only 1 gas sensor is applicable for this project. The temperature sensor we will be using is a LMT84DCKR which is as stated before an Analog Temperature Sensor. This sensor has a wide temperature range of -50℃ to 150℃ and a typical accuracy of +-0.4℃. Now, our temperature sensors will be placed across several boards including the main PCB board, and we plan to mimic a standard family home with 7 rooms. Because of the need to hook each of these outputs to an ADC pin on the ESP32, and our need for more GPIO pins on the ESP32 for LED's, we decided to use an 8:1 multiplexer which will be where we input the data and then simply cycle through each address and read the values. Finally, the last thing that is a part of our sensor subsystem is our LED's, the LED that we will be using is RGB LED which will be capable of displaying yellow, red, and green which is what's needed for our operation. Each LED will require 2 GPIO pins, and we have 7 LED's, therefore in total we would require 14 GPIO pins from the ESP32 simply for the LED's. As of now we will proceed with using these GPIO pins instead of using a 2:1 multiplexer as we don't have a need for these GPIO pins for something else.

1. Temperature Sensor
    - Measures temperature from -50℃ to 150℃ with typical accuracy of +-0.4℃.
    - Ultra-low power consumption with maximum current at 5V = 8µA.
2. Gas Sensor
    - Measures CO concentration of 10~500 ppm and 300~10000 ppm of CH_4.
3. RGB LED
    - The Red LED will shine with an intensity of 300 mcd upon fire detection.
    - The Green LED will shine with an intensity of 700 mcd upon fire detection.
    - The Yellow LED will shine with an intensity of 600 mcd upon fire detection.

Table 1: Sensor subsystem - Requirements & Verification

Requirements	Verification

• The temperature sensor will be able to monitor temperature from -50℃ to 150℃ with an accuracy of +-0.4℃ as well as being an Ultra-low power consumption device.	• Power on the sensor with 5V or 3.3V on pin 4 if using a SOT (SC70) package and grounding pins 1, 2, 3. • With a voltmeter, place the positive probe on pin 3 which is our output pin and negative probe on ground. Ensure that the current temperature in the room is known. • Ensure that the sensor is in thermal equilibrium by reading the output voltage and then using the datasheet, you will find a corresponding temperature which you can then compare with the temperature you measured in the room. • Grab some heat source or cold source that you know the temperature of, and you can begin to take data points of the corresponding voltage. • The output voltage decreases as temperature increases at a rate of -5.3mV/℃, so if your data shows this relationship, the sensor is calibrated well. • Measure the current being drawn with a multimeter to ensure this operation is a low power consumption device.
• The MQ-9B gas sensor will be able to detect Carbon Monoxide ranges from 10~500 ppm, Methane ranges from 300~10000 ppm.	• For testing this sensor with a gas, use a well-ventilated area. • Power on the sensor with a DC voltage of 5V applied across the sensing pins as described in the data sheet and let the sensor preheat for 10 minutes. • Gas sensor has 2 heater pins which should cycle between 5V and 1.5V, for the sake of only testing the gas sensor (with lower reliability) you should only connect the 5V to the heater pin for testing. • Use a voltmeter between the load resistor and ground and monitor the voltage across the resistance. The voltage should be a value around 0.4-1V and should not be fluctuating • In a controlled environment, you can burn paper/wood nearby the sensor and you should see that your voltage output will increase as the concentration of CO increases. If this is the case, the sensor is working properly. • A similar test can be done with a Methane source to test if the sensor is able to detect that gas.

• The LED's will display Yellow, Green, and Red colors with different intensities due to limits in how bright each color can be.	• you will need a 3.3V power supply with a resistance of around 200Ω to place on the Cathode of the LED pins. • Grab a multimeter and set it to diode mode and probe the anode pine and one of the cathode pins and the LED should faintly glow for the corresponding color being tested. • Now you can power on the LED's anode with the series resistance on the cathode, once you connect the cathode to ground, the color should light up, repeat this step for the other color pins. • Changing the resistance will change the brightness of the LED, so this test can be performed to test if the LED can vary brightness.

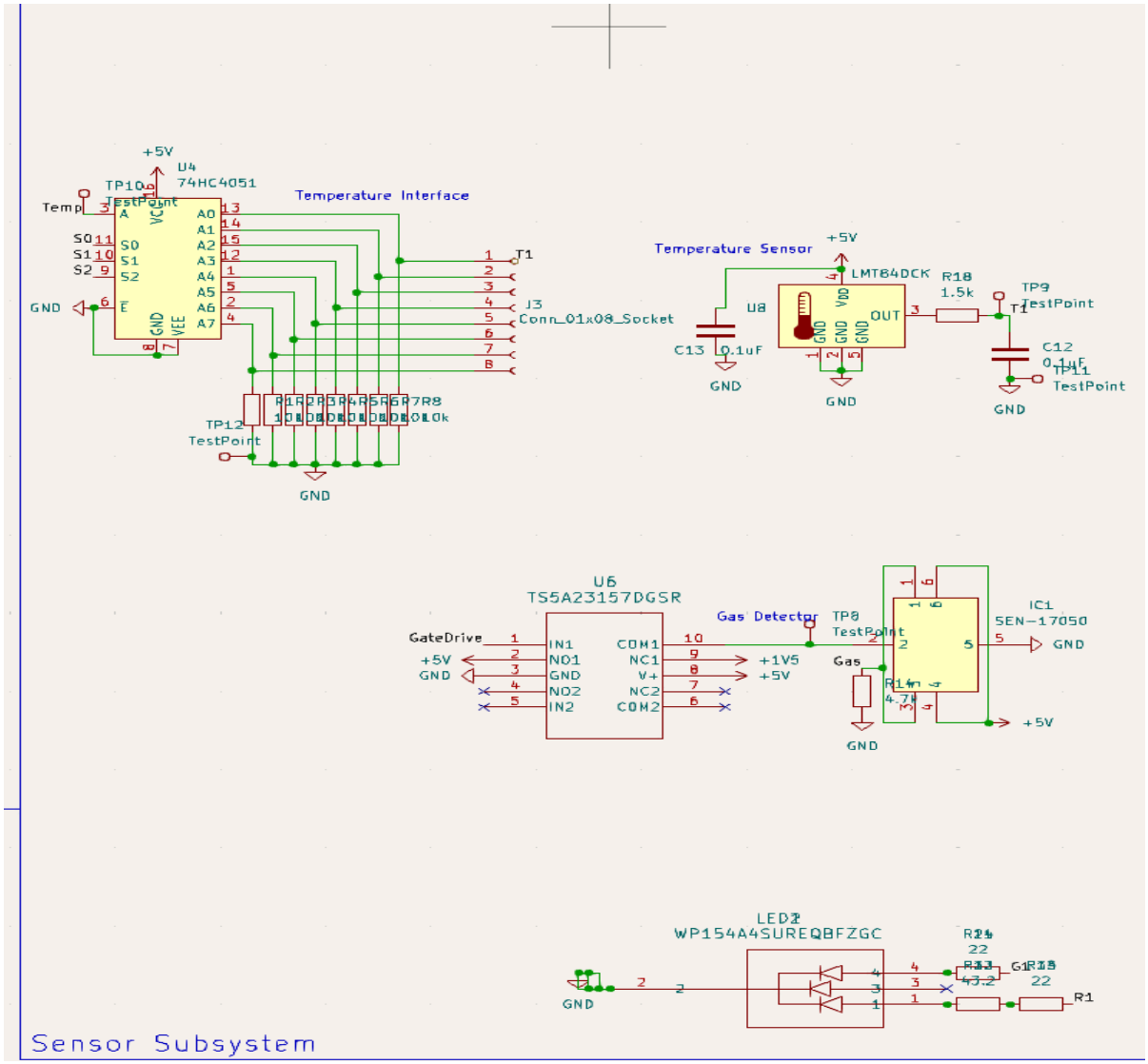

Figure 4: Sensor Subsystem Schematic

2.2.2 Vent Subsystem

The vent subsystem is responsible for automatically closing the vents and also shutting off the HVAC system when a fire is detected. This is crucial to preventing the spread of fire by limiting airflow and ensuring that smoke and toxic gases are not circulated throughout the house. To prevent a build-up of air pressure the HVAC fan will be turned off before closing the vents. The vent will be controlled by a 7V digital servo motor, which will be integrated into a standard vent cover by the ECE Machine Shop. The servo motor is capable of 20kg.cm of torque with 180-degree rotation which will ensure there are no issues opening or closing the vent. The motor will be directly controlled via PWM signals from the ESP32, allowing for incremental vent movement and ensuring that the vents can fully close when a fire is detected.

1. Vent/Servo Motor
    - The motor will be able to close vents at a speed in which the HVAC will turn off before the vent fully closes upon the detection of a fire.
    - The motor will be able to open vents once the fire is cleared, once the vents are opened, the HVAC will turn on again and resume normal operation.
2. HVAC
    - Upon receiving the signal that there is a fire, the HVAC should turn off immediately to limit the flow of oxygen.
    - Upon receiving the signal that there is no fire, the HVAC should continue normal operation.

Table 2: Vent Subsystem - Requirements & Verification

Requirements	Verification
• Taking the ESP32 PWM signal as input, ensure that the motor is positionally able to hold an angle of 0-180° as stated in the datasheet under no load.	• Power on the servo motor with a 7V input. • Send commands from the ESP32 which will hold the motor torque at different test positions such as 90°, 180°. • Using a protractor or some other technology, measure the angle at which the arms of the motor move to confirm if the motor is responding well to the positions given.

• Taking the ESP32 PWM signal as an input, ensure that the motor is able to hold a reasonable amount of torque under a given load.	• Power on the servo motor and send a command from the ESP32 which will hold the motor at a position of 180°. • Apply a small force on the servo arm and see if the servo is able to withstand this force. The force applied should be less than 25 kg*cm. • The test will be considered a success if the motor is able to handle this reasonable force and maintain its position without shaking.
• Taking the ESP32 PWM signal as an input, ensure that the motor can rotate the given position in a reasonable amount of time.	• Have a stopwatch or some other way to measure the amount of time it takes for the motor to spin to a given position. • Power on the servo motor and send a command from the ESP32 which will send a positional command for the motor to spin to 60°. During this time, ensure you record the start to finish time of this process. • Once you have the data, the datasheet says the rotation speed is 0.25sec/60° (7.4V), so if your calculated time was around this, you can ensure that the motor can rotate the given position in a reasonable amount of time.
• Taking the ESP32 PWM signal as input, ensure that the motor can cause the vent to both open and close.	• Power on the servo motor and apply a PWM signal with positional commands which will cause the vent to open. • Do the opposite now and send a PWM signal which is able to now close the vent,
• Ensure that the Digital Servo Motor does not draw excessive current.	• Power on the servo motor and do normal operation of opening and closing the vents, using a multimeter, measure the current draw of the motor to ensure the current is within datasheet specification, 2.4-3A is the stall current for this motor.

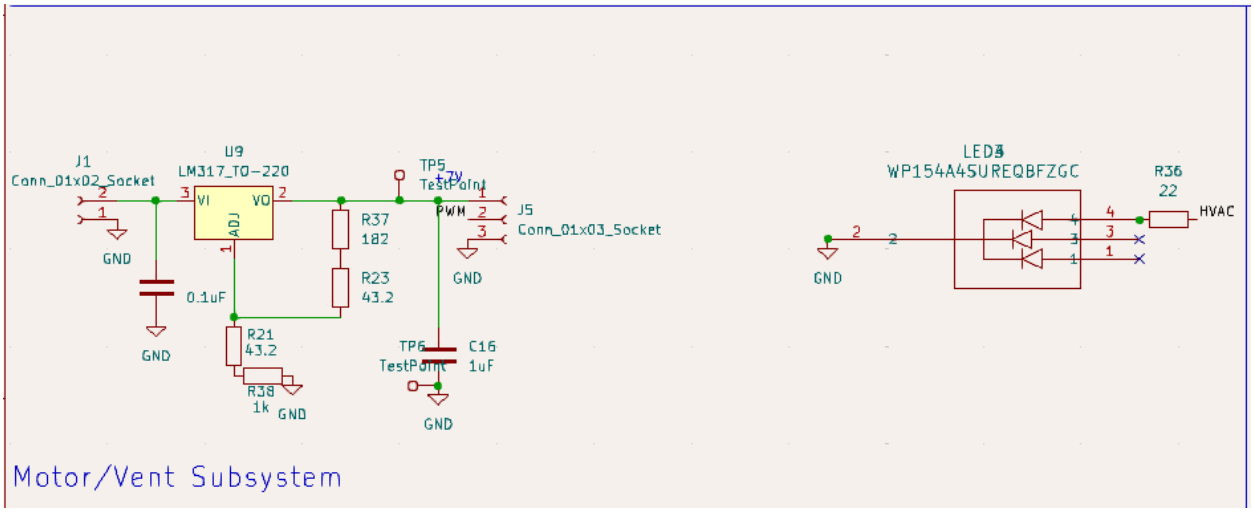

Figure 5: Vent Subsystem Schematic

2.2.3 Control Subsystem

The control subsystem acts as the interface between subsystems. It consists of the ESP-32 and the alarm that will ring in the case of hazards. The ESP-32 will process sensor data and control the LEDs and HVAC to ensure proper coordination between the hardware and the application. The ESP-32 will receive temperature readings from the temperature interface multiplexer by cycling through the select options within milliseconds using three GPIO pins for the select and one ADC capable pin for the reading. The gas sensor will be connected directly to the ESP-32 using one ADC capable GPIO pin. This data will be stored in real-time to Firebase for application use via WiFi as the ESP-32 has integrated WiFi connectivity. When a fire is detected the ESP-32 will use GPIO PWM signal to close the vent subsystem and turn off the HVAC fan. The ESP-32 will read LED configurations from Firebase and light up the respective LEDs directly using two GPIO pins for each LED. In the case of any hazard, the ESP-32 will sound the alarm which is a CMI-9605IC-0580T magnetic buzzer indicator capable of producing a sound of up to 80dB.

1. Buzzer Alarm
    - The alarm will sound upon receiving the signal from the ESP-32 that a hazard has been detected at 80dB.
    - The alarm will keep on sounding until there is no longer a hazard detected via the ESP-32.
2. ESP-32
    - The microcontroller will be able to cycle through and read the value of the multiplexer containing information about the temperature within milliseconds.

- The microcontroller will be able to send a signal to an alarm causing it to sound under the following conditions:
    - One of the temperature sensors has a value which exceeds the threshold of 90°C.
    - The gas sensor measures concentrations of CO > 10 ppm or CH4 > 5000ppm.
- The microcontroller will be able to send a signal to LED's which will light in certain colors depending on the situation:
    - Red: The source of the fire has originated here or in proximity to this place, not safe.
    - Yellow: The source of the fire is in a more distant location from this room but there is no exit to be found in this room, not safe.
    - Green: The path one should follow to the safest exit that exists in the house.
- The microcontroller must be able to use its Wi-Fi capabilities to store this data from the sensors in a firebase as well as receiving information from the application in order to send the relevant signals out to the other components in the case of fire.
- The microcontroller at maximum operation (Wi-Fi on) will draw no more than 500mA.

Table 3: Control Subsystem - Requirements & Verification

Requirements	Verification
• ESP32 is able to read ADC values using its ADC capable pins on Channel 1.	• Ensure that the ESP32 is powered on by a 3.3V supply. • Connect a potentiometer to an ADC capable GPIO pin. • Program the pin to print values corresponding to 2^{12} which is 4096 and is the resolution of the ADC pin. • Tune the potentiometer and you should see that the value being displayed will change ensuring that the ADC pin is ready to read.
• ESP32 should be able to send a PWM signal for controlling the motor.	• Program a PWM GPIO pin which will send a square wave at a given duty cycle in a period. • Connect an LED with some current limiting resistor to the output of this pin. • You should notice that the LED will go bright, and then it will dim down corresponding to the PWM waveform you send.
• ESP32 should be able to connect to the Wi-Fi.	• Using the Wi-Fi library, find a program that will enable the ESP-32 to connect to the Wi-Fi. • Once the ESP-32 connects to the Wi-Fi, the

	serial monitor should show an IP address indicating that there was a successful connection.
• ESP32 is able to read values from the multiplexer containing information about the temperature sensor and triggering an alarm once the threshold is passed. Should also be able to turn off the alarm once the hazard is cleared.	• Ensure that the ESP32 is powered on and the temperature sensors are powered on as well. • Connect the Temperature output to a ADC capable pin on the ESP-32 and program the ESP-32 to be able to convert the ADC value to a readable voltage value. • With some heat source such as a lighter, light the lighter and bring it near the sensor. • If the readable voltage value crosses the threshold (90°C = .543V), the ESP-32 will output a signal to the alarm causing it to sound. • Upon the hazard being cleared, the ESP-32 will no longer send a high signal to the switch controlling the alarm causing the alarm to turn off.
• ESP32 is able to read values from the Gas Sensor and will trigger an alarm once the threshold is passed as well as turning off the alarm once the concentration returns to normal levels.	• Ensure that both the ESP-32 and the gas sensor are powered on. • Connect the output of the gas sensor to an ADC capable pin on the ESP-32 and program the ESP-32 to be able to convert the ADC value to a voltage value that you will then be able to convert to gas concentration using the formula expressed in the datasheet. • Burn a piece of paper near the sensor to raise the concentration of CO. • If the CO concentration reaches 10 ppm, the ESP-32 will be programmed to send a signal to the alarm that a hazard has been detected. • Upon the clearance of this hazard, the ESP-32 will no longer send a high signal to the alarm and the alarm will shut off and normal operation is resumed.
• ESP-32 will be able to cycle through and read values from the multiplexer containing the temperature values within milliseconds.	• In the final step of the project, ensure that all temperature sensors across the different rooms are powered on and connected to the main PCB board containing the multiplexer. • Program the ESP-32 to begin reading these ADC values and convert these values to voltages and then into temperatures. • Place these sensors under different temperature conditions whether that be cold, or hot with varying intensity. • Ensure that the terminal output voltage is changing temperatures corresponding to these

	different temperatures, this will ensure that the ESP-32 is capable of reading and selecting the channels in a reasonable amount of time.
● ESP32 correctly stores sensor values to Firebase.	● Set up sensors with the ESP-32 connected to WiFi. ● Program the ESP-32 to store sensor data in Firebase. ● Manually change sensor values by increasing temperature and verify real-time changes in the database.
● ESP32 correctly reads LED configuration from Firebase and signals the right LEDs to turn on.	● Set up LEDs with the ESP-32 connected to WiFi. ● Program the ESP-32 to store sensor data in Firebase. ● Store an example LED configuration in Firebase and check to ensure the LEDs that turn on match the configuration.
● The buzzer should be able to play a sound of 80dB at 5V.	● Connect the buzzer to a 5V voltage source with a resistor for protection. ● A sound should be played at a noise level of 80dB. ● Use a phone app or a decibel meter to measure how close the sound being played is to 80dB. ● If the sound is within +-4 dB of 80dB, then the alarm is fine, if not, use a voltmeter to ensure that the alarm is indeed receiving a 5V input.

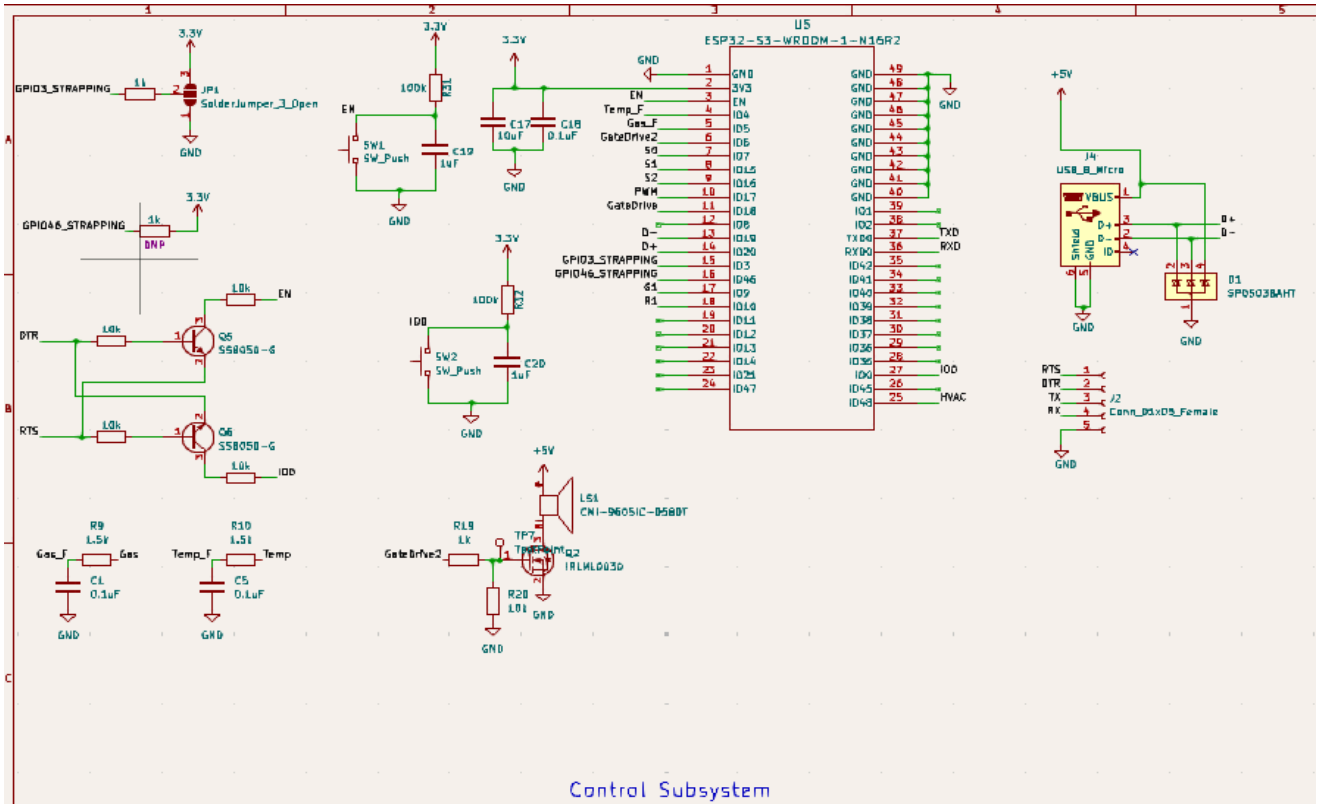

Figure 6: Control Subsystem Schematic

2.2.4 Application Subsystem

The application subsystem will serve as the primary interface between the user and the fire detection system by allowing the user to input their floor plan. The application will have a Firebase backend and a React frontend using the React DnD library to create a 2D grid layout where the user can drag and drop components such as rooms, doors, windows, stairs, sensors, and vents. There will be buttons to navigate to higher and lower floors which will replace the grid layout with the respective floor. The application will use this data to construct a graph representation of the home layout which will be stored in Firebase. The ESP-32 will be storing sensor data and reading LED commands to and from Firebase in real time using WiFi; the application will read this sensor data and store the LED commands. When a fire is detected, the application will execute a pathfinding algorithm which will be a modified A* that uses the sensor data to assign the cost of each area. The algorithm will compute all possible exits and pick the optimal route, this information will be stored as an LED configuration to Firebase, which the ESP-32 will read. The optimal route will be continuously updated until the user is no longer in danger.

1. Application

- The app must determine and communicate the escape route to the control unit within 5 seconds of hazard detection.
- The app must consistently receive sensor data while connected to WiFi in real time.
- The application must correctly display and store the user-defined floor plan.

Table 4: Application Subsystem - Requirements & Verification

Requirements	Verification
• The application must correctly display and store the user-defined floor plan.	• Test drag and drop functionality and create example floor plans. • After constructing graph representations, use visualization tools to print the graph and manually ensure it is correct. • Store to Firebase and manually ensure the graph is stored correctly.
• The app must determine and communicate the escape route to the control unit within 5 seconds of hazard detection.	• Use logging within the application to determine the time taken to calculate the optimal route and the time taken to store the LED configuration. • Run the algorithm with dummy sensor data and an example floor plan to ensure the time taken to calculate the route and store the LED configuration is less than 5 seconds.
• The app must consistently receive sensor data while connected to WiFi in real time.	• Display sensor data in the application read from Firebase. • Connect the ESP-32 to Firebase to store sensor data. • Manually change sensor values by increasing temperature and verify real-time changes in the application.

2.2.5 Power Subsystem

This system will be in charge of supplying us the correct levels of voltages across all our components. We will be utilizing a 9V high-capacity 1200mAh non-rechargeable lithium-ion battery pack. We will have 3 different voltage regulators. The first two voltage regulators will be a fixed output voltage regulator. So, one of these regulators will produce 5V with an output current of 1A, the other regulator will then produce 3.3V with an output current of 1A as well. Then, we will use an LM317T LDO voltage regulator that will give us an output voltage of 1.5V with an output current of 1A as well. The 5V will be used to power our gas sensor, buzzer alarm, temperature sensor as well as the multiplexer, and finally the USB

port. The 3.3V will be used to power on the ESP32-S3, and the 1.5V will be used only for the gas sensor heating pins.

1. ESP32-S3
    - The Power Supply must be able to supply a voltage of 3.3V with a +-0.3 tolerance.
    - The Power Supply must be able to supply at maximum 400mA to the ESP32.
2. Gas Sensor
    - The Power Supply must be able to supply a voltage of 5V and 1.5V with a +-0.2 tolerance.
    - The Power Supply must be able to supply a maximum current of 200mA to the Sensor.
3. Temperature Sensor
    - The Power Supply must be able to supply a voltage of 5V with a +-0.5 tolerance.
    - The Power Supply must be able to supply a maximum current of 50μA.
4. Buzzer Alarm
    - The Power Supply must be able to supply a voltage of 5V with a +-0.5 tolerance.
    - The Power Supply must be able to supply a maximum current of 50mA when active.
5. 8:1 Multiplexer
    - The Power Supply must be able to supply a maximum current of 50mA due to switching.

Table 5: Power Supply Subsystem - Requirements & Verification

Requirements	Verification
• The Fixed Voltage Regulator outputs will have at maximum a +-0.2 tolerance on the voltage output.	• Power on the power supply with the Voltage Regulators connected. • With a voltmeter, measure the output of each regulator and ensure that the voltage output is within the required threshold.
• The Fixed Voltage Regulator output will have 1A +-0.2 across all the voltage regulators.	• Power on the power supply with the Voltage Regulators connected. • With a multimeter, use the current measurement setting and probe the output of the regulator to ensure the expected current is measured.

• The battery must be able to provide 9V +-3V during the 1200mAh operation	• Set up a no-load test in order to measure the voltage across the battery, a fresh battery should be 9V or higher. • Connect a light load resistance such as 1kΩ and measure the voltage while under this load. The battery should have a voltage of 9 +-1V under this lighter load. • With the light load resistance, you can find the internal resistance of the battery to ensure that it is small as it should be (<1Ω typically). • If an extra battery is available, you can use a resistor that draws 300mA. Our battery is 1200mAh operation so you should expect the battery to be discharged in 4 hours at a 300mA load. If the battery lasts significantly less than 4 hours, the battery could be degraded.

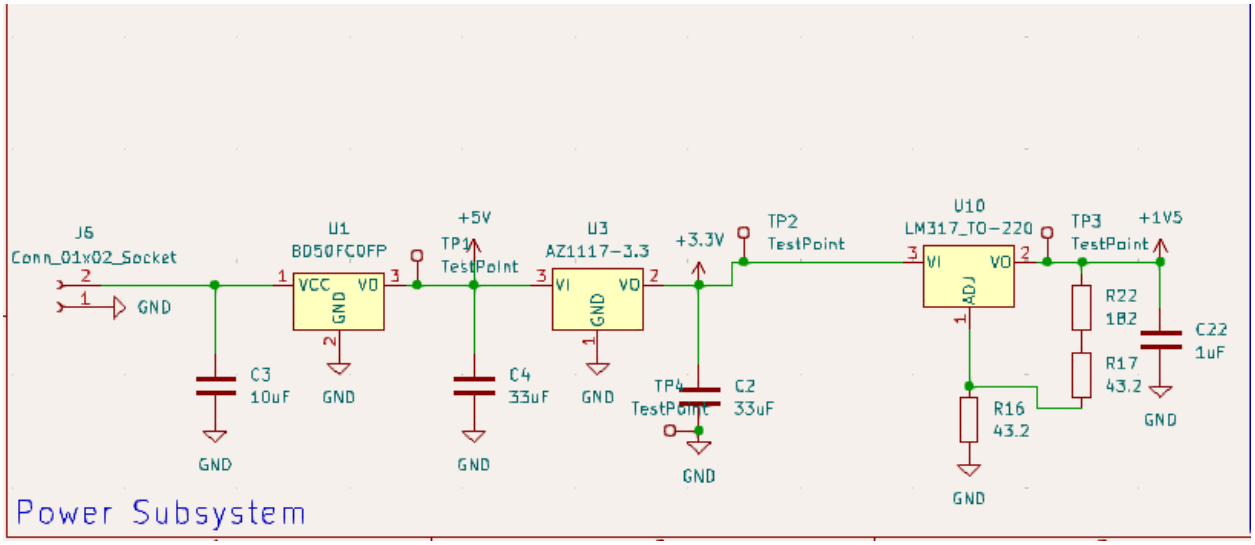

Figure 7: Power Subsystem Schematic

2.3 Hardware Design

2.3.1 Temperature Sensor

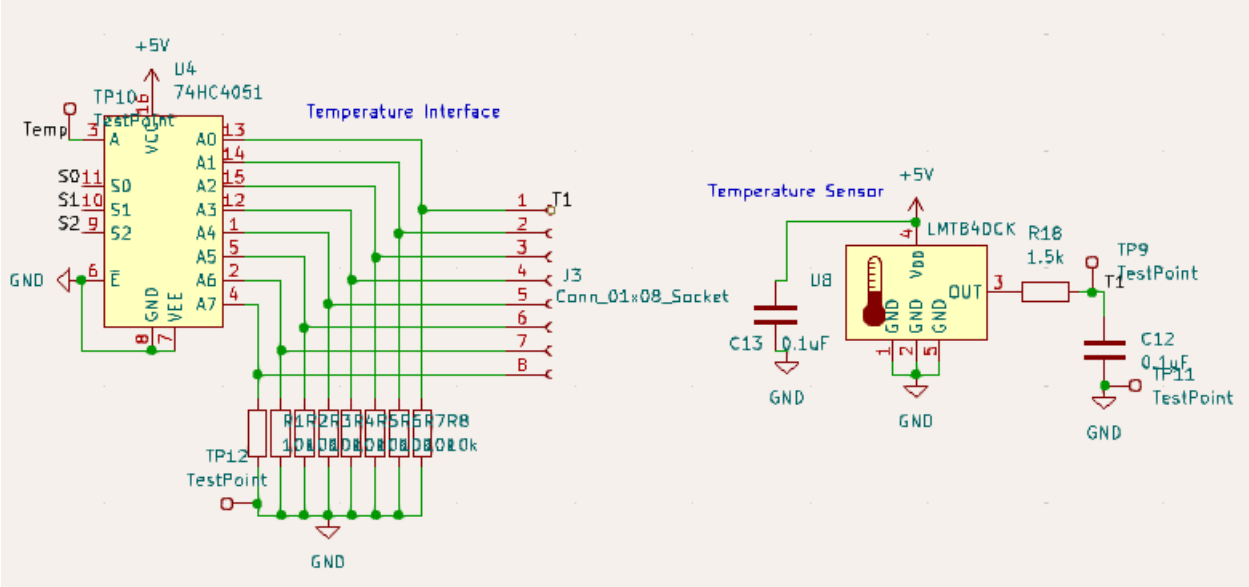

Figure 8: Schematic of Temperature Sensor

The following schematic diagram is of our temperature sensor we will be using along with a 8:1 mux. The temperature sensor has simple connections, a 5V input voltage source which will come from our voltage regulator, along with an input capacitor to filter out voltage fluctuations. The output will also have an RC filter which will be used to smooth out our output to maximize the accuracy of our output for the ESP32 to read. As shown in Figure 9, the datasheet provides the LMT84 transfer table lookup sheet which tells you what temperature you should expect given the voltage you are reading from the output.

Table 3. LMT84 Transfer Table (continued)

TEMP (°C)	V_{OUT} (mV)	TEMP (°C)	V_{OUT} (mV)	TEMP (°C)	V_{OUT} (mV)	TEMP (°C)	V_{OUT} (mV)	TEMP (°C)	V_{OUT} (mV)
–37	1231	3	1017	43	799	83	574	123	343
–36	1226	4	1012	44	793	84	568	124	337
–35	1221	5	1007	45	788	85	562	125	332
–34	1215	6	1001	46	782	86	557	126	326
–33	1210	7	996	47	777	87	551	127	320
–32	1205	8	990	48	771	88	545	128	314
–31	1200	9	985	49	766	89	539	129	308
–30	1194	10	980	50	760	90	534	130	302
–29	1189	11	974	51	754	91	528	131	296
–28	1184	12	969	52	749	92	522	132	291
–27	1178	13	963	53	743	93	517	133	285
–26	1173	14	958	54	738	94	511	134	279
–25	1168	15	952	55	732	95	505	135	273
–24	1162	16	947	56	726	96	499	136	267
–23	1157	17	941	57	721	97	494	137	261
–22	1152	18	936	58	715	98	488	138	255
–21	1146	19	931	59	710	99	482	139	249
–20	1141	20	925	60	704	100	476	140	243
–19	1136	21	920	61	698	101	471	141	237
–18	1130	22	914	62	693	102	465	142	231
–17	1125	23	909	63	687	103	459	143	225
–16	1120	24	903	64	681	104	453	144	219
–15	1114	25	898	65	676	105	448	145	213
–14	1109	26	892	66	670	106	442	146	207
–13	1104	27	887	67	664	107	436	147	201
–12	1098	28	882	68	659	108	430	148	195
–11	1093	29	876	69	653	109	425	149	189
								150	183

Figure 9: Voltage to Temperature Data of the Sensor

While the sensor output is very linear, there is still a slight parabolic shape to it, so the overall equation you should use to calculate the data as given in Figure 9 is the following:

$$V_{temp}(mV) = 870.6mV - [5.506(mV/℃)(T - 30℃)] - [0.00176(mV/℃^2)(T - 30℃)^2]$$

So, if you are looking at a voltage output, input the temperature you are observing into the following equation and you will find an output voltage that corresponds to that temperature. It is important for us to have a prompt response if a fire is detected, and the 8:1 multiplexer which was chosen should allow for the ESP32 to cycle through each channel and read the values correspondingly in a reasonable amount of time. The Multiplexer has a switching time of 18ns at 5V, which is much faster than the ADC sampling speed of the ESP32 which is at a maximum 2MHz. We will be using Wi-Fi operation, therefore, this speed will drop, therefore we will say the ESP32 ADC sampling speed will be 10kHz, or 10ksps per channel. We will also be adding a 0.1µs delay between switching. Therefore, if we want to read values from 8 different channels at a speed of 10ksps per channel, as well as our delay of 0.1µs, it will take an estimated time of 1.6ms for our ESP32 to read from all 8 channels of the multiplexer which is still a fast response. The only difference between connecting the temperature sensor output directly to the input of the ADC pin of the ESP32 is simply the delay that we should introduce for the switching. The choice of

this sensor compared to other temperature sensors was simply that it measured -50℃ to 150℃ which is perfect for a testing environment. In practice you would want sensors that could handle higher temperature ranges, it doesn't necessarily have to detect temperatures at a high range such as 300℃, but being able to withstand these temperatures and still be able to operate would be nice but more pricey which is not needed in our case.

2.3.2 Gas Sensor

The gas sensor chosen was the MQ-9B SEN17050 which is a 6 pin device and has the following schematic as shown below in Figure 10:

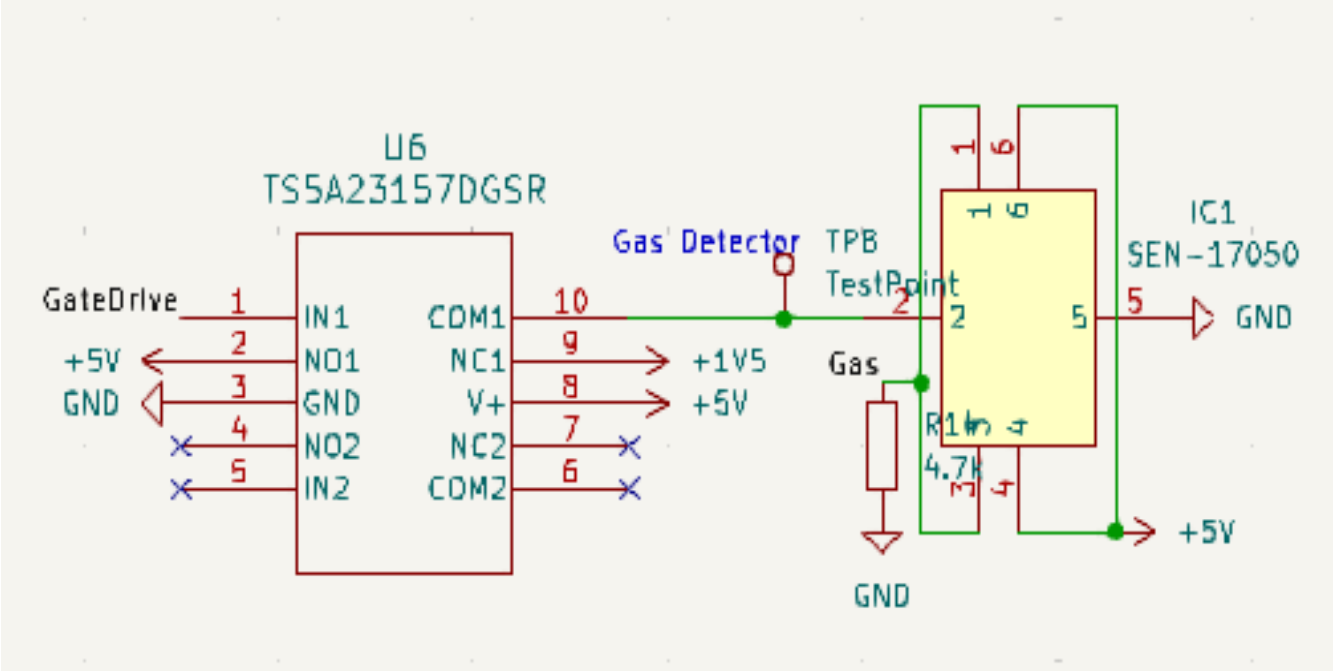

Figure 10: Gas Sensor Schematic

From the schematic diagram, pin 2 and pin 5 are our heater pins, pin 2 will be receiving either 5V or 1.5V from our 2:1 multiplexer. The sensor requires that you cycle the heater pins as follows: you want to hold 5V for 60s and then switch to 1.5V for 90s. Before taking measurements, you want to make sure that you preheat the sensor using the 5V which is attached between pins 6 and 4 for around 10 minutes, this will ensure that you get more accurate readings. A load resistance is attached between pins 1 and 3 which is our sensing pin, from the datasheet, a load resistance of 4.7kΩ was used between the sensing pins and provided an expected output as shown below in Figure 11:

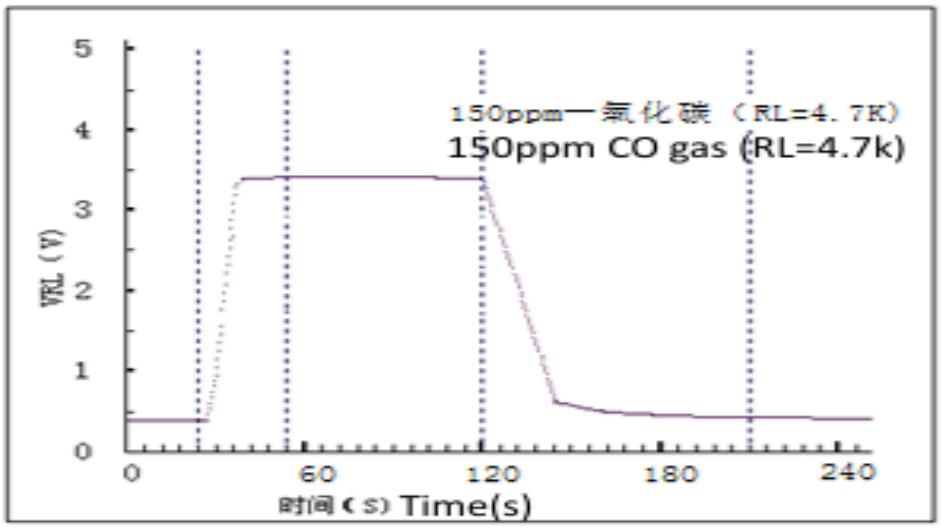

Figure 11: Load Voltage Changing with Gas Concentration

We can see from this graph that we start at a low voltage of around 0.4V with a load resistance of 4.7kΩ and a supply voltage of 5V. Carbon Monoxide is then introduced to the sensor at around 30s and we see a fast rise to a much higher voltage in the presence of the carbon monoxide. The gas concentration is held constant for around 100 seconds, and after this time the gas is removed and we can see that the expected voltage output will return back to the nominal value of around 0.4V. To get more insight as to how the signal will look like, Figure 12 shows more details as shown below:

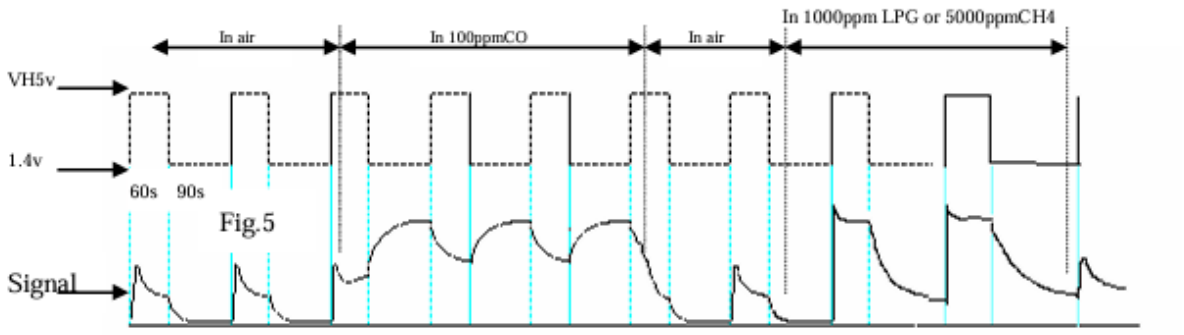

Figure 12: Operation Principle of Gas Sensor

Here we see that we hold 5V for 60s and 1.5V for 90s. The 5V is specifically there to measure only the flammable gasses such as CH4, while the 1.5V will be able to measure the CO more accurately then the 5V. In Figure 12, when looking at the CO concentration, the 5V concentration will provide us a less accurate reading of CO compared to the 1.5V which gives us a much more consistent reading of how much concentration is actually present. The choice of the MQ-9B sensor is that it is commonly used in a lot of applications involving gasses as there are plenty of models to choose from which serve different

purposes with which gasses it detects. We chose the MQ-9B as it detects Carbon Monoxide which is one of the most important gasses to be able to detect. We were unable to find a MQ-9 standalone gas sensor, they were always sold in Breakout Boards, so we found a replica known as the MQ-9B sensor which has the same operation principle.

2.3.3 Battery Usage

We will be using a 4 pack 9V Lithium battery with a capacity of 1200mAh, this battery will be non-rechargeable but offers high performance as shown in Figure 13:

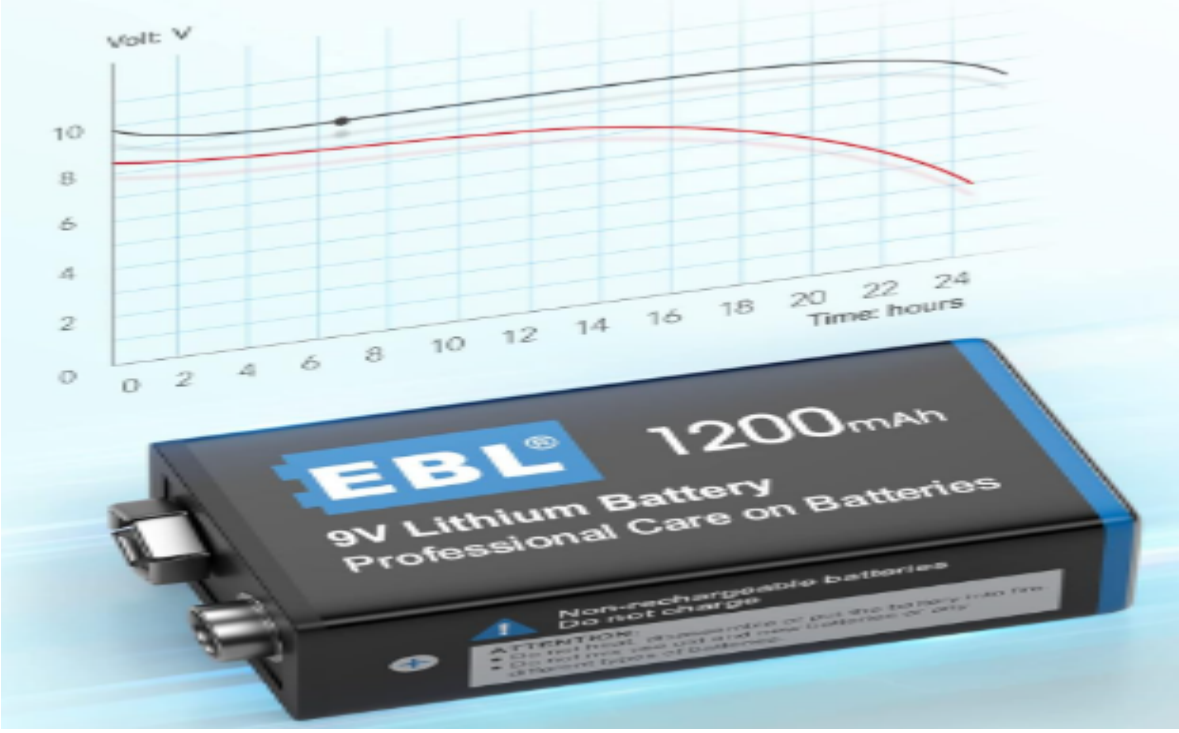

Figure 13: Curve Representing the Life of the Battery

The red curve represents a typical Alkaline 9V battery, the black curve on the other hand represents the battery we have chosen and that's shown in Figure 13 as well. From this image we can see that the battery is able to hold a much more constant output at 9V for about the whole duration of the battery. We see that it starts dropping down after 24 hours which would be the expected time of the battery to discharge assuming that the load the company used to test the battery was 50mA, which, if we compare with the red curve, shows a much better output and battery life. In our application, we are already expecting to discharge this battery much faster than what is shown here. Fortunately, this battery holds a more consistent voltage when compared with other batteries of the same non-rechargeable type. Although our battery is rated for 1200mAh, the voltage regulators will limit this current output to be 1A. So, it is important to check the maximum total current that could be drawn by our components.

- ESP32 has maximum current draw of 500mA assuming ESP32 is using Wi-Fi only as well as utilizing its GPIO pins for reading signals and outputting.
- The Gas sensor has a maximum current draw of 200mA assuming a high concentration of gas is introduced to the system.
- Temperature sensors have negligible current draw (designed for ultra-low power).
- The LEDs that we will be utilizing will have a maximum current draw of 10mA, and there are 8 in total in our design so this would amount to 80mA.
- The buzzer alarm when active will consume at maximum 50mA.
- The 8:1 multiplexer due to active switching can consume at maximum 50mA.

The rest of the components not mentioned above are meant to be low current consumption therefore negligible in this calculation. Adding all of these currents up we get a total current draw of 880mA, for the purposes of all the other negligible currents, we will round this number up to a total current draw of 900mA from our batteries. So, our battery will at maximum testing conditions can discharge as fast as $Battery\ Life = \frac{Battery\ Capacity}{Current\ Drawn} = \frac{1200mAh}{900mA} = 1.33h$. So, this number could be optimistic and our battery might indeed end up discharging much faster, our hope with this battery is that it maintains a fairly constant output as shown in Figure 13. And even if there is a deviation, our circuit can accept a $\pm$3V deviation due to our voltage regulators being fixed outputs at their respective voltages as seen in Figure 7 where we have 2 Fixed voltage regulators in our Power Subsystem. The only LDO voltage regulator (LM317) will be the one that outputs 1.5V to send to the heater pin of our gas sensor.

2.3.4 PCB Layout

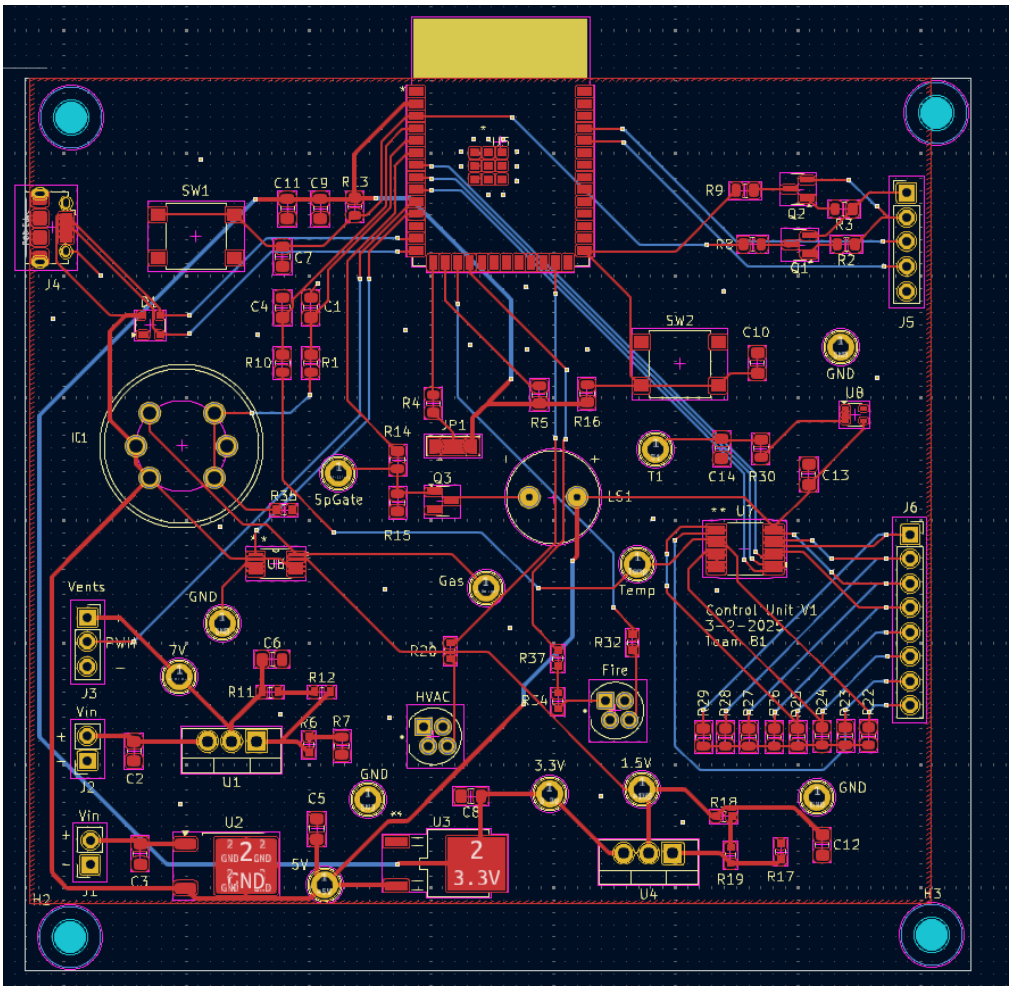

Figure 14: Control Unit PCB Layout

2.4 Software Design

2.4.1 Database

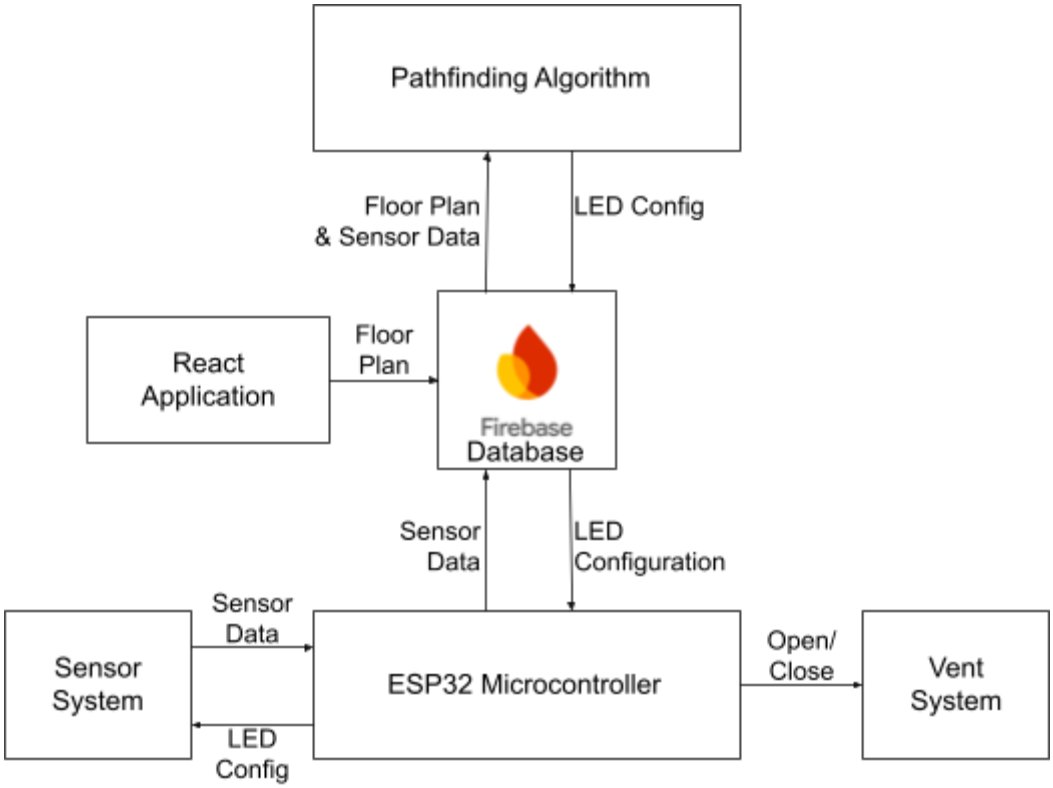

Figure 15: Flowchart of Software Design

The database will be structured as shown on the right to handle storing floor plans, sensor data, led configurations, and HVAC status:

```
/fire-detection-system
  /users
    /userID1
      /floor-plans
        /plan1 { rooms: {...}, doors: {...}, exits: {...} }
        /plan2 { rooms: {...}, doors: {...}, exits: {...} }
      /sensor-data
        /sensor1 { temperature: 72, gas: 5 }
        /sensor2 { temperature: 120, gas: 0 }
      /led-configurations
        /room1 { status: "green" }
        /room2 { status: "red" }
      /hvac-status { status: "off" }
```

Figure 16: Structure of Firebase Database

2.4.2 Pathfinding Algorithm

Pathfinding Implementation Steps:

1. Construct a graph representation of the home layout using user-defined room connections.

2. Assign weights to each path based on distance and hazard levels.
3. Use *A** to determine the lowest-cost path to every exit.
4. Update Firebase with LED configuration data to guide the escape route.
5. Continuously recompute the route if hazard conditions change.

Pathfinding Algorithm Overview:

1. **Graph Representation:**
    - The floor plan is converted into a graph, where:
        - Nodes represent rooms.
        - Edges represent valid pathways between rooms (i.e., doors and stairs).
        - Each edge has a default weight based on distance.
    - The stored floor plan in Firebase is parsed to construct this graph dynamically.
2. **Hazard Integration:**
    - Sensors provide real-time hazard levels (temperature, gas concentration).
    - Rooms with high hazard levels are marked as dangerous and either:
        - Have their edge weights increased to discourage pathfinding through them.
        - Are removed entirely from the graph if they exceed a critical safety threshold.
3. **Escape Route Calculation:**
    - The system runs a *modified A (A-star) pathfinding algorithm** to determine the safest exit route.
    - The algorithm prioritizes paths based on:
        - Shortest distance to exit
        - Lowest hazard levels (avoiding high-risk areas)
        - Real-time sensor updates (continuously recalculates routes as conditions change)
    - The algorithm continuously updates Firebase with the new escape path and LED configurations.

2.5 Tolerance Analysis

In our project, we are using 2 fixed output voltage regulators. But, we are also using an adjustable voltage regulator, this means that the output voltage will not always be guaranteed to give us the desired voltage that we want. As resistors have tolerances that will cause some difference in what we calculate, as well as the dropout voltage from the internal resistance of the LM317, a perfect output cannot be guaranteed. The voltage regulator output can be expressed as $V_o = V_{adj}(1 + \frac{R2}{R1}) + (I_{adj}R_1)$. Here I_{adj} is the current that will flow from the middle pin of the regulator and this current is generally very small so the effects can be neglected for our low-power application. Our application for this adjustable voltage regulator will be to take a 3.3V input and output a 1.5V to our gas sensor heater pin. This 1.5V is very important as it allows our gas sensor to accurately detect CO concentrations. R_2 will be set at 225.2 Ω, the typical V_{ref} value is set to be 1.25V. We can then come up with a value of R1 that should give us an output voltage of 1.5V, in this case we would select R_1 =43.2Ω. Upon simulating for a no load case, you would see the following results:

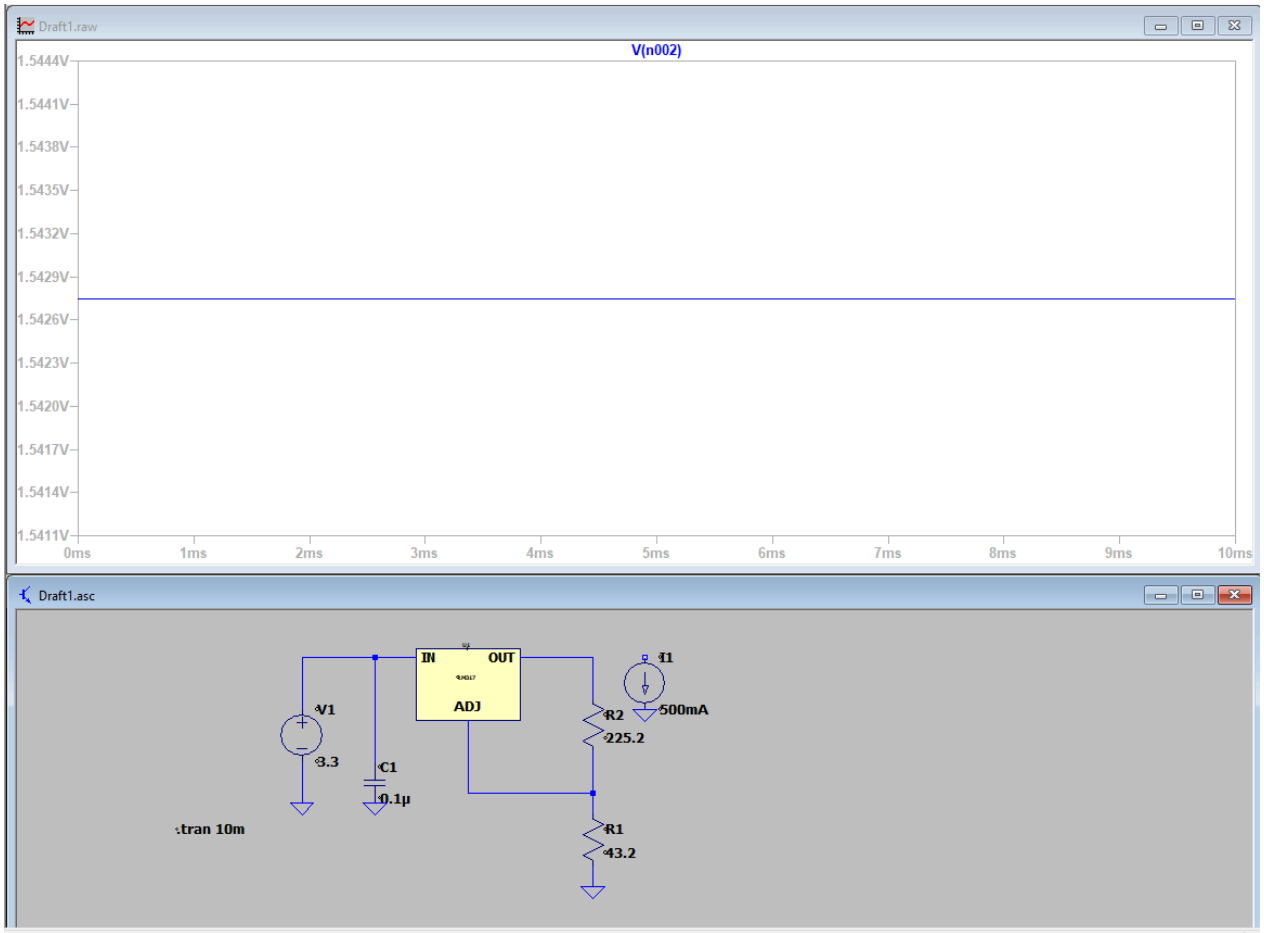

Figure 17: Simulation of LM317 Voltage Regulator with no Load

We see here that the value we see isn't exactly 1.5V, and the reason for this is because we are neglecting the current that we assumed to be small coming out from the ADJ pin. Also, while the reference voltage is typical 1.25V, this value could range from 1.2V - 1.35V depending on the actual reference voltage measured, so, with this we can find the current coming out of the pin ADJ by using ohm's law across R_2. In doing so, we get that I_2 min $= \frac{1.2}{225.2} = 5.33\text{mA}$, and I_2 max $= \frac{1.3}{225.2} = 6\text{mA}$. This value is pretty significant and would be enough to cause our voltage to vary from what we expect to see. the voltage produced across R_1 would be 720*5mA = 0.23V minimum and 0.26V maximum. And when we add these values to our reference voltage across R_1, we can see that the output can actually range from 1.24V-1.76V. Now, this voltage value is a pretty big range which is not ideal, fortunately, the gas sensor does have a tolerance of $\pm 0.2\text{V}$. And we expect a voltage of 1.5V, therefore our gas sensor has a voltage range of 1.3-1.7V. So, with this we can see that the operation would be overall fine, the output voltage that is actually produced by the LM317 is constant, so once you are able to reach the desired output voltage, you will be fine. Now, in order to actually test whether or not the regulator will be able to work under actual testing conditions, we will have to simulate the regulator with a load. The reason for this is because the regulator has an internal resistance, and with higher loads, a higher voltage dropout will be seen which will vary the actual output voltage from what is expected. So, we know that overall we will at max be expected to supply 1A of current as that is the highest amount of current the regulators are able to supply to our circuit. Therefore we can take a load of 500mA to test our circuit response:

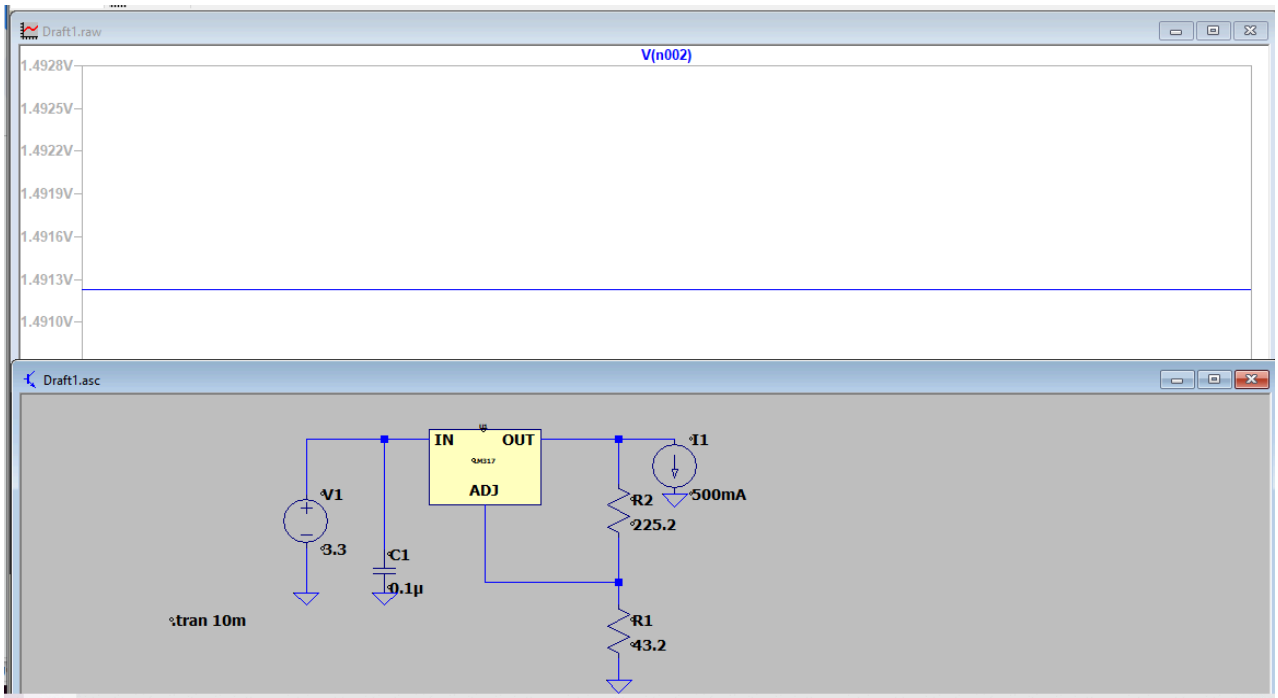

Figure 18: Simulation of LM317 with 500mA load

From the Figure 18 above we can see that our regulator is able to output a nice and constant ~1.5V with a reasonable load attached. We can see that the voltage dropout has drastically increased from when we were dealing with no load, almost a 0.05V difference when comparing with the no load case. As mentioned above, the max load current we will ever see is 1A as it is the only current we are able to supply, but the actual maximum current we should expect to see is 900mA. Unfortunately, the LM317 seems to really fall apart once we hit a load value of 800mA as shown below:

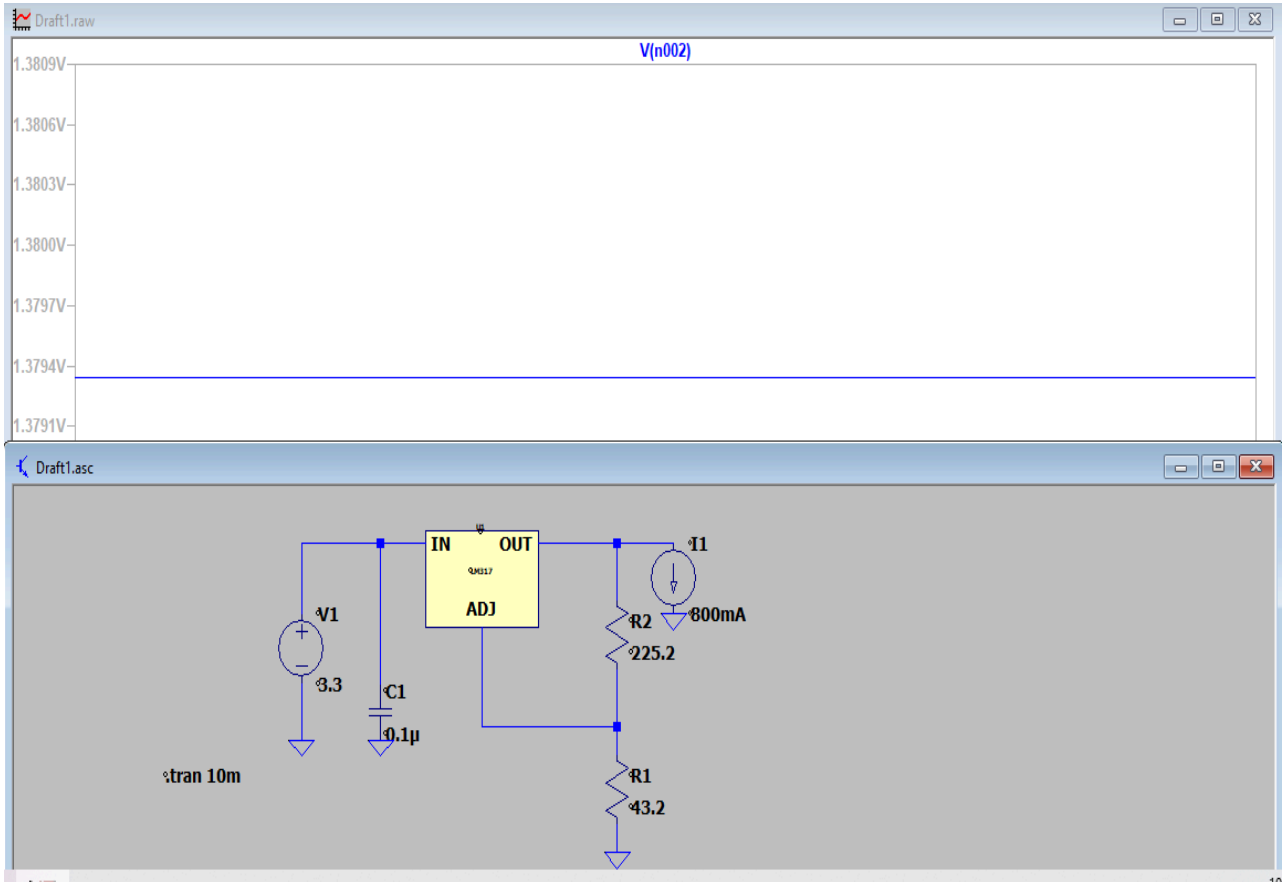

Figure 19: Simulation of LM317 with 800mA load

We see now that our voltage will almost be at the minimum voltage we should be feeding into the gas sensor. So, although the voltage output is lower, it's still in range and is still constant, and as we can see the voltage dropout will continue to increase, and the amount of heat we are dissipating will also increase, at this load we are dissipating a total of 1.5W due to heat. Now, we will continue to do actual testing with this device, but if testing ends up showing inconsistent results, we will switch to another LDO which offers a lower voltage dropout to ensure we can tolerate a load of maximum 1A. For our case, we believe we will be fine as we really won't be getting to this high of current during normal operation, the current estimate of 900mA was for worse case scenario, and even if we do get too high, our only concern would actually be the overheating of the LM317 as the gas sensor tolerance is only there to ensure that the actual

readings of the sensor are accurate. So, falling below this rating will mean that the measurements are less accurate, but the device should still function properly.

3 Cost and Schedule

3.1 Cost Analysis

The total cost for parts as seen below in Table 6 before shipping is $117.4. Adding a 10% shipping cost adds another $11.74 and 8.85% sales tax adds another $10.39. We can expect a salary of $17/hr*2.5hr*90 = $3,825 per team member. We have three members so overall, our labor cost is $3,825*3 = $11,475. Adding everything together, our total cost is $11,475+$11.74+$10.39+$117.4 = $11,614.53.

Description	Manufacturer	Quantity	Extended Price	Link
MQ-9 Gas Sensor	Amazon	2	$7.99	Link
Non Rechargeable 9V Li Battery 1200mAh	Amazon	1	$20.39	Link
4-inch by 10-inch Vent with Opening	Amazon	1	$8.95	Link
25KG Digital Servo Motor High Torque	Amazon	1	$13.99	Link
Buzzer 5V	Digi-Key	2	$2.06	Link
N-Channel Mosfet Switch	Digi-Key	5	$7.25	Link
01x03 Socket Connector	Digi-Key	2	$0.64	Link
01x02 Socket Connector	Digi-Key	4	$1.44	Link
01x05 Socket Connector	Digi-Key	2	$0.65	Link
01x08 Socket Connector	Digi-Key	2	$0.76	Link
LED RGB	Digi-Key	10	$12.53	Link
Analog Temperature Sensor	Digi-Key	10	$3.86	Link
2:1 Dual Mux	Digi-Key	2	$1.00	Link
8:1 Mux	Digi-Key	2	$0.64	Link
Fixed 5V Voltage Regulator	ECE E-Shop	2	$3.26	Link
Fixed 3.3V Voltage Regulator	ECE E-Shop	2	$1.28	Link
Mosfet -IRLML0030TRPBF	ECE E-Shop	2	$0.78	Link
Capacitor .1µF (0805)	ECE E-Shop	8	$1.2	Link

Capacitor 33µF (0805)	ECE E-Shop	5	$3.85	Link
Capacitor 10µF (0805)	ECE E-Shop	5	$5.1	Link
Capacitor 1µF (0805)	ECE E-Shop	6	$0.48	Link
Transistor - SS8050-G	ECE E-Shop	2	$0.48	Link
Resistor 10kΩ (0805)	ECE E-Shop	15	$1.50	Link
Resistor 1.5kΩ (0805)	ECE E-Shop	5	$0.50	Link
Micro USB-B Connector	ECE E-Shop	1	$5.99	Link
Resistor 22Ω (0603)	ECE E-Shop	8	$2.00	Link
Resistor 43.2Ω (0603)	ECE E-Shop	8	$0.80	Link
Resistor 4.7kΩ (0603)	ECE E-Shop	3	$0.30	Link
Resistor 1kΩ (0805)	ECE E-Shop	4	$1.04	Link
Resistor 182Ω (0603)	ECE E-Shop	4	$0.44	Link
Resistor 100kΩ (0805)	ECE E-Shop	4	$0.40	Link
Switch Tactile	ECE E-Shop	3	$0.36	Link
Test Points	ECE E-Shop	15	$4.88	Link
Diode - SP0503BAHTG	ECE E-Shop	1	$0.61	Link

Table 6: Bill of Materials

3.2 Schedule

Week	Task	Person
March 2nd - March 9th	Finish 1st round PCB orders	Alex
	Finish Design Document	Everyone
	Start Breadboard Testing	
	Teamwork Evaluation 1	
March 9th - March 16th	**2nd round PCB orders (3/13/25)**	Everyone
	Breadboard Demo (3/11/25)	
	Start Developing Application	Jainam
March 16th - March 23rd	Spring Break	Everyone
	Start the Individual Progress Report	
March 23rd - March 30th	Start Soldering/Debugging	Abel & Alex

	Start Testing Application	Jainam
March 30th - April 6th	**Third Round PCBway orders (3/31/25)**	Everyone
	Individual Progress Report (4/2/25)	
	Start to integrate both Software and Hardware	
April 6th - April 13th	**Final Round PCBway Orders (4/7/25)**	Everyone
	3D print enclosure for the PCB boards	
	Have the Model of the floorplan ready	
	Continue Testing the System under different circumstances	
April 13th - April 20th	Start working on Final Paper	Abel
	Continue Testing with full modelled system	Everyone
	Prepare for Mock Demo	
April 20th - April 27th	**Mock Demo (4/22/25)**	Everyone
	Finalize Testing for the Final Demo	
	Start work on Final Presentation	
April 27th - May 4th	Week of the Final Demo	Everyone
	Finalize Presentation	
May 4th - 11th	Week of the Presentation	Everyone
	Finish the Final Paper (5/7/25)	
	Lab Notebook Due (5/8/25)	

Table 7: Schedule for Project Progression

4 Ethics and Safety

In our project, we strive to meet the requirements laid out in the IEEE Code of Ethics to ensure that our fire and gas detection system will help to improve the lives of its users by keeping them safe in dangerous situations. Our main emphasis is on the health and wellbeing of our project's users, so we intend to strictly adhere to Section I.1 of the IEEE Code of Ethics which instructs to keep the safety and health of the public as our highest priority, which includes the disclosure of any problems our project may face that would jeopardize the safety of its users. [11]

In addition, there is the ethical concern of our project's use of personal data to function as intended. This includes the use of an application that will communicate with our project's control unit and prompt the

user to input their floor plan for our algorithm to work in determining the safest exit. According to the ACM Code of Ethics Section 1.6, we will ensure our user's right to personal privacy by only using the date relevant to the operation of our project as well as ensuring the data is secure and only accessed by the control unit of our project. [12]

For the safety and wellbeing of our users, one concern is the use of 9V batteries to power our project. We will make sure that our batteries are protected from accidental faults in our project in order to protect against any damage that may be done to our other subsystems, the batteries themselves, and to the user.

In the lab, there is another safety concern with the testing and operation of our project. We will be testing our sensors' response to hazards such as fire and carbon monoxide, so it is important to follow lab safety guidelines for our safety, and for the safety of everyone else in the lab. All tests will therefore be conducted in a well ventilated environment, especially with the carbon monoxide sensor, for everyone's protection. Extra precautions will also be taken when transporting dangerous materials we may be testing to ensure no accidents occur.

Finally, the operation of the automatic vent system poses a safety concern as it may be dangerous for the user to handle while it is operational. To ensure the safe handling of the automatic vent closure, considerations will be made to encapsulate as much of the moving parts as possible to minimize the risk of any accidents. In addition to ensuring safe handling of the vent closer, we must also ensure the deactivation of the HVAC system to protect not only the HVAC unit itself, but for the protection of the user from any issues that may arise if it doesn't turn off.

5 References

[1] “Ventilation-Limited Fires and the Influence of Oxygen,” 2017. [Online]. Available: https://www.fireengineering.com/firefighting/ian-bolton-ventilation-limited-fires-and-the-influence-of-oxygen/

[2] H. Naidoo, “Oxygen Enrichment and Fire Hazards,” 2024. [Online]. Available: https://www.co2meter.com/blogs/news/oxygen-enrichment-hazards?srsltid=AfmBOoqvc_zfwub4eY8siFmFGKAD62cOa29HnswcrraIdGanyEZ2TTF1

[3] *Magnetic Buzzer Indicator,* Same Sky, 2024, rev. 1.02. [Online]. Available: CMI-9605IC-0580T Datasheet - Audio Indicators | Buzzers | Same Sky

[4] *IRLZ34NPbf Power Mosfet*, International Rectifier, rev. 5. [Online]. Available: IRLZ34NPbF.pmd

[5] *Full Color LED Lamp*, Kingbright, 2021, rev. V.9A. [Online]. Available: WP154A4SUREQBFZGC(Ver.9A)

[6] *Analog Temperature Sensors*, Texas Instruments, 2017, rev. 2.0. [Online]. Available: LMT84 1.5-V, SC70/TO-92/TO-92S, Analog Temperature Sensors datasheet (Rev. E)

[7] *High-Speed CMOS Logic Analog Multiplexer and Demultiplexer*, Texas Instruments, 2024, rev. 4.0. [Online]. Available: CDx4HC405x, CD4HCT405x High-Speed CMOS Logic Analog Multiplexer and Demultiplexer datasheet (Rev. N)

[8] *Toxic Gas Sensor*, Winsen, 2015, rev. 1.4. [Online]. Available: MQ-9B_Ver1.4__-_Manual.pdf

[9] *4 Pack 9V Lithium Batteries 1200mAh*, EBL, 2022. [Online]. Available: Amazon.com: EBL 4 Pack 9V Lithium Batteries 1200mAh, Non Rechargeable 9V Batteries - High Performance 9 Volt Lithium Metal Battery Longer Lasting Constant Volt for Smoke Alarm Detector : Health & Household

[10] *25KG Digital Servo Motor High Torque PWM Servo*, Hiwonder. [Online]. Available: Amazon.com: 25KG Digital Servo Motor High Torque PWM Servo, Full Metal Gear Steering Servo for Robotic Arms and DIY Robot Making, Digital Servo with 25T Servo Horn, Control Angle 180° : Toys & Games

[11] IEEE, “IEEE Code of Ethics,” ieee.org, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html
[12] ACM, “ACM Code of Ethics,” ACM, [Online]. Available: https://www.acm.org/code-of-ethics