Harman Swaich – Jason Li – Kiryll Parfyonov – Raj Sarai

INCS 3610: EMBEDDED SYSTEMS  PROJECT PROPOSAL

Attic cooling system project proposal

Table of Contents

[Rationale: 2](#_Toc118375211)

[Scope of the Project: 2](#_Toc118375212)

[In Scope: 2](#_Toc118375213)

[Building the Circuit 2](#_Toc118375214)

[Writing the Code 2](#_Toc118375215)

[Product Target 3](#_Toc118375216)

[Out of Scope: 3](#_Toc118375217)

[Testing Prototype 3](#_Toc118375218)

[Constraining Budget 3](#_Toc118375219)

[Ordering Equipment 3](#_Toc118375220)

[Data Analytics 3](#_Toc118375221)

[Restricting Temperatures 3](#_Toc118375222)

[Deliverables: 4](#_Toc118375223)

[Conceptual Framework 5](#_Toc118375224)

[Methodology 6](#_Toc118375225)

[Project Plan 10](#_Toc118375226)

[Work Breakdown Structure 10](#_Toc118375227)

[Gantt Charts 11](#_Toc118375228)

[Chart for Work Breakdown Structure (Figure 1) 11](#_Toc118375229)

[Chart for Work Breakdown for Individual Tasks 12](#_Toc118375230)

[Risk Analysis 4](#_Toc118375231)

[Hackers 4](#_Toc118375232)

[Wi-fi Loss 4](#_Toc118375233)

[Rodents 4](#_Toc118375234)

[Fire Hazard 4](#_Toc118375235)

[References 14](#_Toc118375236)

[Appendix 15](#_Toc118375237)

# Rationale:

Most houses have an unfinished area underneath the roof called an “attic”. This area can be problematic during the summer months as heat builds up and rises to this area. This area can reach up to 75°C. As a result, the temperature of the room(s) directly underneath the attic increases, causing discomfort for occupants inside. Central air conditioning units are a means to regulate the internal temperature inside a house. Specifically, a thermostat is used to sense and send a control signal to the cooling system to maintain a desired temperature. The air conditioner will constantly run since the thermostat is detecting a higher temperature reading. Consequently, overall energy consumption (and operational costs) increases during the summer months.

The purpose of our project is to create an automated cooling system. Once the temperature reaches a certain temperature, the fan will turn on. The sensor we will be using will be the DHT22 Temperature and Humidity sensor. There will be one sensor placed in the attic, and another one placed in the bedroom. The fan will continue to run until the attic temperature has reduced to the desired point.

An override condition has been implemented where the fan will turn off during nighttime. We will be using a photoresistor sensor located on a window ledge in the bedroom.

# Scope of the Project:

## In Scope:

### Building the Circuit

We will be using the following equipment to build the circuit:

* Arduino Uno Rev3
* DHT22 Temperature Sensor
* Photoresistor
* Relay
* ESP32
* Fan

Once the DHT22 sensor reaches a certain temperature, the fan will be turned on. The photoresistor is used to determine when it becomes dark outside. Once it is dark, the fan will automatically switch off.

### Writing the Code

The code will consist of a fan being controlled by a temperature sensor and a photoresistor. The two inputs will be the temperature and photoresistor readings. The output will be the fan itself.

We will have a condition where the fan will turn on once the specific temperature is reached. The fan will turn off once temperature is reduced to a desired point. Regardless of the temperature, the fan will be turned off once it is dark outside.

### Product Target

The project design is catered for residential use only. The sensor we will be using has a range of -40°C to 80°C. The Arduino operating temperature range is -40°C to 85°C with ±0.5% accuracy.

## Out of Scope:

### Testing Prototype

Since our project is primarily used during the summer months, we cannot test the prototype in an attic. We will be using two shoe boxes with a heat source like a hairdryer to warm up the inside of one of the shoe boxes.

### Constraining Budget

By setting a budget, our options for equipment are limited using inexpensive, and readily available equipment such as sensors.

### Ordering Equipment

Since there is a deadline for this project, much of the equipment wasn’t available online. The arrival dates of some of the equipment were longer than normal.

### Data Analytics

Due to time constraints, we will be lacking data analytics. We will not have a mobile app to control or monitor temperature activity.

### Restricting Temperatures

This is for residential use only because the DHT22 sensor can only reach a maximum reading of 80°C. Anything beyond 80°C will read the same.

The Arduino operating temperature range is -40°C to 85°C. While the recommended temperature is a maximum of +70°C. We would need to make sure the temperature of the attic is below this point for the system to function to what it is designed to do.

## Risk Analysis

### Hackers

The protocol we will be using is HTTP. Our data will not be encrypted when it is in transit. The only risk this imposes is that hackers will be able to monitor our data. The control of the system will not be at risk.

This would be an unlikely risk to happen. We would take a risk acceptance approach to this.

### Wi-Fi Loss

Loss of connectivity to the wi-fi will cause the ESP32 to no longer upload data to the server. This will not stop the functionality of the system. During the time of the outage, the data will not be monitored.

### Rodents

Damage can be caused by rodents in the attic. May lead to parts of the system not working such as connectivity to the server. Can also lead the system to not work altogether.

Traps, poison, and rodent repellent will be used to mitigate this risk of occurring.

### Fire Hazard

A short circuit can cause harm to both the system, the dwelling, and most importantly, residents living within the home.

A dedicated Fire and Smoke alarm system will be present to prevent/mitigate a potential harm. Verifying that the attic stays a smoke-free area will also alleviate the risk of a fire.

## Deliverables:

At the end of the project, we will have these following deliverables:

* Email notification via ThingSpeak
* Live monitoring on a webserver via ThingSpeak
* Prototype of the cooling system
* Final report for Umme

# Conceptual Framework

Server Holding Temperature Data

As the temperature of the roof rises, it causes the temperature of the attic to rise. This process is the cause of the attic acting as a heat source for the floor below it.

As the fan operates it will cool the temperature of the attic, thus reducing the temperature of the floor beneath it.

Temperature of Floor Beneath Attic

Temperature of Attic

Sunrays Hitting Roof of House

Set Speed of Fan

As the sunrays hit the roof of the house, the rays will cause the roof of the house to heat up.

If the attic temperature falls within certain temperature characteristics, it will trigger the fan to turn on.

# Methodology

This project consists of an experiment aimed at reducing a house’s attic temperature to reduce its effect as a heating source for the floor(s) below it. During the hotter months of the year, this is problematic as the floor below it will heat up drastically due to the attic’s rise in temperature.

We have decided to build an embedded system that uses a microcontroller, a Wi-Fi module, a fan, and two temperature sensors to distinguish the temperature in the attic and the floor below it. The fan will turn on when the temperature in the attic rises past a certain temperature OR if the difference between the attic and the floor below it is more than five degrees. The fan will turn off when the above bounds have been resolved. We decided on five degrees because when dealing with high summer temperatures, a rise of five degrees is noticeably an uncomfortable change.

Our data will be sent out using a Wi-Fi module that will store the data in ThingSpace. Since we are using a DHT22 sensor, temperature data will be sent to the server every two seconds. When the fan gets triggered, a message is sent via email to the owner notifying them of the temperature change. This data can be accessed and graphed for visual representation.

Obstacles we have had include problems with the code, that include connecting to BCIT’s internet. Their authentication is not supported by our code. The circuit build was also based off trial and error, with no specific obstacle. Due to ongoing international supply chain issues, long than expected delays on certain equipment is expected.

We hope that our results will show that the embedded system works within our experimental environment and that it can be practically used in a real-life situation.

# Project Design

Diagram

Description automatically generated

In this project, we built a close loop on/off feed back control system. The purpose of this system is to keep attic area relatively cool during summertime. The attic temperature and top floor room temperature are measured, and the temperature data would be collected by the controller. When designed conditions are met, the fan would turn on and push hot air out through roof vents.

Power supply for the system can be any type of 5 volts power source. In the graph above, the computer is acting as a power source. The reason is in our final presentation, we used computer to supply power and watch the serial monitor simultaneously for better demonstration. It can be replaced by a regular 5VDC phone charger, or a mobile power bank if local power outlet is not available.

We have chosen ESP32 as our controller because it is a cost-effective micro-controller to be used in our project, which has BLUETOOTH an WIFI feature. Two DHT22 sensors as temperature sensors, and a Photoresistor to detect the light level. DHT22 can measure both temperature and humidity. Humidity reading is not used in this project, but it can be used for future optimization of this project.

There are three conditions that determine the fan status. If the photoresistor gives a reading higher than 1500 (can be adjusted based on user’s preference), if the temperature difference between attic and top floor is greater 5˚C, or the attic area temperature along is greater than 35˚C, then ESP32 will send a digital output signal to turn on the cooling fan.

In this prototype, an Arduino Uno plus a 5V driven relay are used together to use the digital signal from ESP32 to control the cooling fan. Reason for this combination is the relay we bought along with all other equipment is 5V driven, but ESP32 runs 3.3V input/output only. Due to the global supply chain issue, we couldn’t but a 3.3V driven relay on time. However, this Arduino plus relay combination can be replaced by a single 3.3V driven relay in the future.

When the Arduino Uno receives a 3.3V signal from ESP32, it will send a 5V digital signal to the relay and active the relay. An extension cord is connected to the relay, and supply 120VAC power to the cooling fan once the relay is engaged. The cooling fan we chose for demonstration purpose is a table fan. In real world application, a 24-40 inches industrial fan can be used.

While the sensors measuring the temperature, light level, and the micro-controller turning on and off the cooling fan, all the data is sent to our web server for statistic and indication purpose. Users can log into their accounts and view the data anytime. We choose Thingspeak as our webserver because it supports MQTT protocol that used on ESP32.

We setup 5 graphs to show tendency of each data. Attic Temperature, Top Floor Temperature, Top Floor Light Level, Differential Temperature, and Fan Status. We can see the data record and relationship between them. We can also study the data to improve our control logic in the future. With Thingspeak free account, we can only upload data every 15 seconds. Paid membership is required for data uploading every second.

The last part is email notification. An email notification will be sent to users when the cooling fan is turned on. Similar to the data upload, free account can only receive 2 email notifications every half an hours. Paid member is required if more email notification is desired.

To sum up, in this project design, we’ve designed this system to be cost-efficient. More importantly, the system has to meet our functionality and performance expectation. Global supply chain issue impacted our design during equipment selecting process. We had to use the equipment that we could have in hand on time to build the system that performs reliable and desirable.

# Result Analysis

Our results meet our expectation. During demonstration, the attic area temperature was decreasing when the cooling fan was turned on. Fan started and stopped once the desired condition is reached as expected. Photoresistor provided very responsive reading and was able to override the fan control logic. Webserver receives data properly and sent email notification successfully.

The entire system was functioning properly. However, the high attic area temperature didn’t cause top floor area during our demonstration because heat transfer is a slow process that takes hours before it can be observed. The attic area temperature was decreasing slowly be cause the fan we used is not powerful enough, also the adapter we made with cardboard caused flow restriction.

Therefore, when we are implementing this system in real world scenario, fan selection and installation can be a key factor that affects ultimate the system performance.

# Lessons Learnt

Our code was written in the C++ language having used Arduino’s IDE. Even though Harman already knew how to program in C++, our group had to learn certain aspects of the syntax needed for the Arduino IDE. This included searching the official Arduino website for syntax information.

We learned how to use an ESP32 within the scope of our project and how to accommodate it in our project code.  
  
Having worked on our project at school, we soon came to realize that due to BCIT’s AAA framework, we were unable to access the internet with the ESP32. For that reason, our solution was using a cellular hotspot from one of our phones. This allowed the ESP32 to connect to the internet and send data to ThingSpeak.  
  
Having needed an online web server for our project, it caused our group to learn how to use ThingSpeak. ThingSpeak is an IoT analytics platform service that allowed us to visualize and analyze our data. We were able to use the graphs for our report and to help the class visualize our data during our presentation.  
  
A crucial lesson our group learnt that can apply to any situation where one is learning a new concept, is that it is important to do the proper research before trying to apply a new concept. For us, during different steps of the project, we often tried applying the concept without fully doing the proper research. This caused a lot of wasted time with troubleshooting that could have been avoided had we done the proper research. Researching every step of a concept is crucial for time management and successfully completing the task at hand.

# Conclusions and Future Direction

Due to the time constraints for this project, we only had a couple months to work on the code and thus a limited time to improve it and make it more efficient. For future applications, the code can be improved and made more efficient.

In a real-life application, it is important to have redundancy in the system. At least for potential points of failure. For this reason, we believe that having a redundant fan is important because if the fan stops working, the temperature will stop decreasing. Having a redundant power supply is just as important, because without power there is no working system.  
  
If the system were to gain popularity and was successful on the market, then creating our own app to connect to the system via Wi-Fi or Bluetooth could be helpful. A dedicated app could allow for different customization of the settings for the system based on individual needs of every home.

# Project Plan

## Work Breakdown Structure

Figure

## Gantt Charts

### Chart for Work Breakdown Structure

Figure (Chart related to Figure 1)

### Chart for Work Breakdown for Individual Tasks

Figure 3

# References

“CP210x USB to Uart Bridge VCP Drivers.” *Silicon Labs*, 14 Sept. 2021, https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads.

diyhacking. “How to Use ThingSpeak and Arduino to Develop a Temperature Sensor.” *YouTube*, YouTube, 13 Aug. 2018, https://www.youtube.com/watch?v=ssiHSCmTylQ.

Mafukidze, Harry. “How to Write Arduino Sensor Data to the Cloud.” *Circuit Basics*, 24 Nov. 2021, https://www.circuitbasics.com/how-to-write-data-to-the-cloud-with-an-arduino/.

Mathworks. “Mathworks/Thingspeak-Arduino: Thingspeak Communication Library for Arduino, ESP8266 and ESP32.” *GitHub*, https://github.com/mathworks/thingspeak-arduino.git.

Taylor, W. Joe, et al. “ESP32 Publish Sensor Readings to ThingSpeak (Easiest Way).” *Random Nerd Tutorials*, 17 June 2021, https://randomnerdtutorials.com/esp32-thingspeak-publish-arduino/#:~:text=After%20inserting%20your%20network%20credentials,publishing%20the%20readings%20to%20ThingSpeak.

Tony, et al. “ESP32/ESP8266 Send Email Notification Using PHP Script.” *Random Nerd Tutorials*, 2 July 2021, https://randomnerdtutorials.com/esp32-esp8266-send-email-notification/.

# Appendix

Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level | WBS Code | WBS | Definition | WBS to Members |
| 1 | 1 | Attic Cooling System | All the work to implement new project |  |
| 2 | **1.1** | **Initiation** | **The work to initiate the project** | -------------- |
| 3 | 1.1.1 | Brainstorm Ideas | Collaborating as a group to find a project | All |
| 3 | 1.1.2 | Allocate Workload | Distribute and assign task's for each member | All |
| 3 | 1.1.3 | Project Scope | Explaining what is in the scope of our project | RS, KP |
| 3 | 1.1.4 | Project Risks | Potential risks | All |
| 3 | 1.1.5 | Project Constraints | Not being able to achieve certain tasks due to constraints | All |
| 2 | **1.2** | **Planning** | **The work to plan the project** | -------------- |
| 3 | 1.2.1 | Make a Schedule | Make a Gantt Chart to schedule specific tasks | RS, KP |
| 3 | 1.2.2 | Research Equipment | List out equipment needed for project | JL, HS |
| 3 | 1.2.3 | Set Budget | Set a budget for equipment needing | All |
| 3 | 1.2.4 | Project Design | Designing the prototype for the appropriate use | All |
| 2 | **1.3** | **Execution** | **The work to execute the project** | -------------- |
| 3 | 1.3.1 | Writing Code | Code to embed onto Arduino | JL, HS |
| 3 | 1.3.2 | Building Circuit | Connecting the Arduino with the sensors, and ESP32 | JL, HS |
| 3 | 1.3.3 | Test Prototype | Make sure the code, and circuit itself are functional | JL, HS |
| 3 | 1.3.4 | Troubleshoot Prototype | Fix any issues we come across while testing the prototype | All |
| 2 | **1.4** | **Closure** | **The work to close out the project** | -------------- |
| 3 | 1.4.1 | Writing Final Report | Project overview with the required criteria | All |
| 3 | 1.4.2 | Delivering Prototype | Demonstrating project via video | JL or HS |
| 3 | 1.4.3 | Oral Presentation | Presenting with a PowerPoint to class | All |

Table

|  |  |  |  |
| --- | --- | --- | --- |
| Start Date | End Date | Description | Duration (Days) |
| 10-Oct | 14-Oct | Compile List of Parts Needed | 4 |
| 14-Oct | 19-Oct | Complete PowerPoint | 5 |
| 15-Oct | 21-Oct | Buy Needed Equipment | 6 |
| 20-Oct | 27-Oct | Code - Fan & Sensors | 7 |
| 25-Oct | 28-Oct | Code - Uploading to Server | 3 |
| 26-Oct | 30-Oct | Building the Circuit | 4 |
| 28-Oct | 04-Nov | Project Proposal and Work Plan | 7 |
| 31-Oct | 18-Nov | Testing Phase | 18 |
| 21-Nov | 02-Dec | Oral Presentation | 10 |
| 25-Nov | 02-Dec | Final Project Report and Video | 7 |
| 02-Dec | 11-Dec | Revise Final Project Report | 9 |