ECE 445 – Senior Design Laboratory

Design Document

Smart AC Unit Device

Team 56

Vineeth Kalister

Xavier Oliva

Kevin Zhang

TA: Douglas Yu

May 1, 2024

Contents

1. Introduction 2
1.1 Purpose 2
1.1.1 Problem Statement 2
1.1.2 Solution 2
1.2 Functionality 2
1.2.1 High-Level Requirements ....
1.3 Subsystem Overview 3
1.3.1 Air Conditioner System (Smart AC device).......... 3
1.3.2 Mobile Device System (User Application).......... 4
1.3.3 Visual Aid 4
1.3.4 Block Diagram 7
2. Design 7
2.1 Equations & Simulations 7
2.2 Design Alternatives 7
2.3 Design Description & Justification 8
2.4 Subsystem Diagrams & Schematics 10
3. Cost & Schedule 10
3.1 Cost 10
3.2 Schedule 12
4. Requirements & Verification 12
4.1 Completeness of Requirements 12
5. Conclusion 18
5.1 Accomplishments 18
5.2 Uncertainties 18
5.3 Future Work / Alternatives 18
5.4 Ethical Considerations 18

1. Introduction

1.1 Purpose

1.1.1 Problem Statement

In the United States, 90% of homes and apartments built have an air-conditioning unit installed. However, about a third of homes lack a central air conditioning system [1]. While some homes are in climates where they do not need a robust air conditioning solution, homes that don't have central air conditioning typically use a window unit for climate control. This is especially true in communities with older homes, such as New York City and Boston [2]. Many of these older homes use "dumb" wall-mounted AC units that are inefficient and manually set, thus contributing to the national energy burden. Furthermore, there is the risk of over-cooling the living space, decreasing overall comfort for those who have a window unit. Therefore, there is a gap in the market for a more affordable and effective AC unit. As such, we want to target these homes and make them more efficient through "smart" AC control units.

"Smart" AC control units that make up most of the market are exclusively to be used with central air conditioning systems, many of which allow you to integrate voice assistants and other AI services. Although there exist "smart" wall-mounted units, these are often equipped with proprietary solutions that exclusively work with certain brands of expensive window units or are, themselves, expensive devices that simply modulate the voltage going inside the AC unit without changing the physical settings of the unit. With our Smart AC Unit system, we believe that we can accomplish a more efficient and equitable experience for those with window unit ACs and ensure optimal ease of access as well as a lower power bill. As the central air conditioning market advances in the technology available to make the air conditioning experience easier, such advances and improvements are lacking in homes that do not have central air conditioning.

1.1.2 Solution

Our proposal is a multipart system combining temperature and humidity sensors, servo motors, and central control units to allow for window-mounted ACs to be automatically controlled through an application on one's smart device. Our "smart" AC device will be able to latch on top of the knobs of a window unit AC, regardless of the brand, and, with the help of the User Application available on their mobile device, be able to accurately adjust the knobs remotely to the settings of the user's choosing. The main system relies on sensor units, control units, and mobile devices. The prototype device will be tested on a 5000 BTU Arctic King window air conditioner. Overall, our unit will offer significant energy and cost-saving capabilities as well as a user-friendly experience and convenience with its mobile device connectivity.

1.2 Functionality

1.2.1 High-Level Requirements

- **Temperature Sensor Integration:**

- The Smart AC Unit can be controlled and changed following signals from the temperature sensors. If the temperature sensed is lower than the desired temperature, the actuators turn the AC unit off, and if the temperature is too high, the actuators turn the AC Unit back on.

- **Accuracy and Precision of Temperature:**
  - Our system will accurately read and compute the current room's heat index with an accuracy of +-0.5 Celsius. This information will be displayed on the display to the nearest degree Fahrenheit.
- **Wifi Connectivity:**
  - Mobile Devices can communicate via Wifi with the Smart AC Device as well as relay instructions that will manipulate the Window Unit AC, whether that's knob adjustment, temperature adjustment, etc. Changes can be made through the user-friendly mobile application.
- **Manually Set Automation:**
  - Be able to manually automate instructions such that the Window Unit AC will increase efficiency when the user is not in the room and save energy and money. Also, be able to set "plans" in the UI system so that the AC unit can be turned off at certain intervals to save power.

1.3 Subsystem Overview

1.3.1 Air Conditioner System (Smart AC device)

1.3.1.1 Power Unit

The Smart AC itself will need to be powered with enough voltage to be able to power the two motors responsible for turning the knobs on a 5,000 BTU Arctic King window air conditioner as well as the temperature and air quality sensors.

The goal is to have the entire control and sensor units powered by USB power through a wall outlet. This necessitates an LM1117 voltage regulator to ensure the correct voltage to the microcontroller and the Servomotors, which will both operate at 4.8-5V. We will use USB-based power. To ensure correct operation, we will use a USB-PD two-wire communication system between a charger and our device.

1.3.1.2 Sensor Unit

The Smart AC device will be equipped with a temperature sensor to read the temperature of the room, and thus, regulate the temperature to the temperature selected by the User Application. The Smart AC device will also be equipped with an air quality sensor which enables the air quality of the room to be read and communicated to the user through the User Application.

For a more accurate sensor reading of the current temperature and humidity, we will use two SHT30 sensors, which will have a +-0.5% real humidity accuracy. These temperature sensors will be held by thin cables that connect them to the main AC unit such that they can be spaced far enough away from the Main AC unit to avoid regional cold spots near the AC unit.

1.3.1.3 Control Unit

The control unit of the Smart AC device system will be capable of changing the settings of both the temperature and cooling knobs of the Arctic King window air conditioner. If the temperature set by the User Application is higher or lower than that measured by the Sensor Unit, the Control Unit is responsible for adjusting the air conditioner settings to ensure that the room temperature stays constant. These will interface with an ESP32-S3-Mini microcontroller to calculate the heat index.

The Control Unit should also be able to be WiFi interfaced with the mobile device and the user application.

The heat index calculation is well known [3]. However, if we are unable to get enough precision, we can always load a look-up table into the microcontroller. This should be well enough given the 16MB of flash memory the ESP32 has.

1.3.1.4 Knob Actuation

The knobs of the smart device should be able to toggle the settings of the AC unit. This will be accomplished by one MS24 20KG actuator which will manipulate the knobs on the AC unit.

1.3.1.5 Display Unit

Simple display to be able to display the desired temperature. Will use NHD-C0220BIZ-FSW-FBW from Newhaven Technologies to display the status and current set temperature of the room.

1.3.2 Mobile Device System (User Application)

1.3.2.1 UI Unit

The user applications contain all the necessary features to read the current room temperature, FS().

1.3.2.2 Control Unit

The user application will be able to communicate with the Smart AC device via Wi-Fi.

1.3.3 Visual Aid

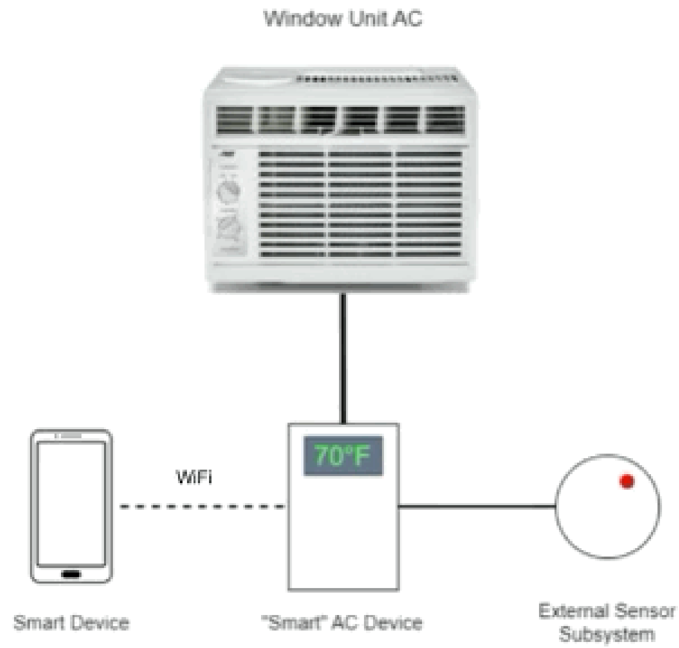


Figure 1 - System View

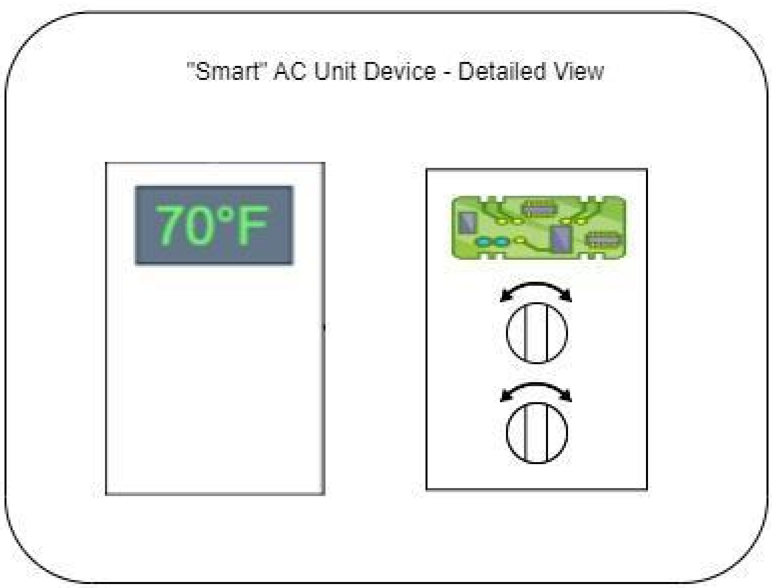


Figure 2 - Simple Detailed View of Smart AC Device

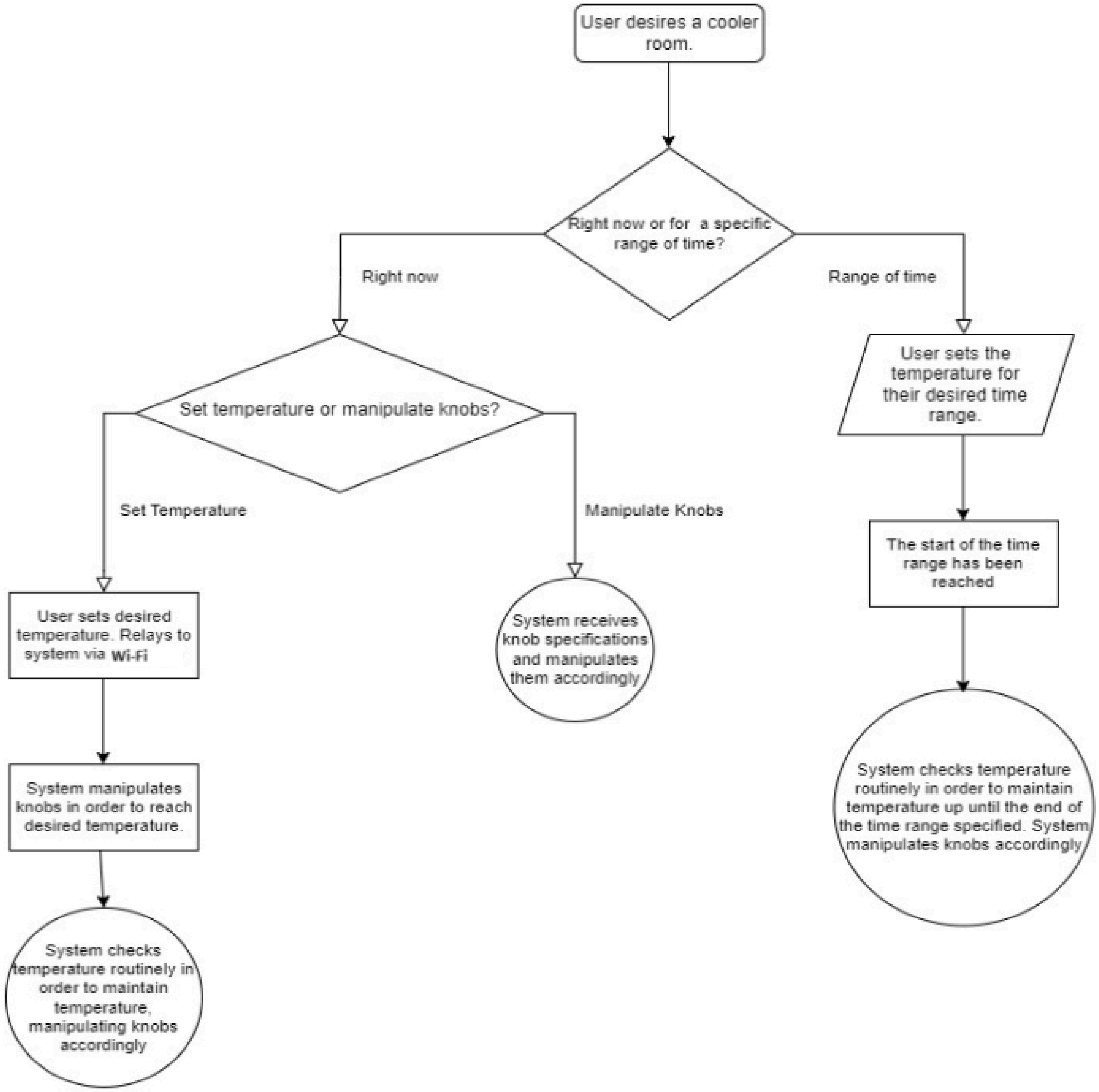


Figure 3 - User Application Flow Chart

1.3.4 Block Diagram

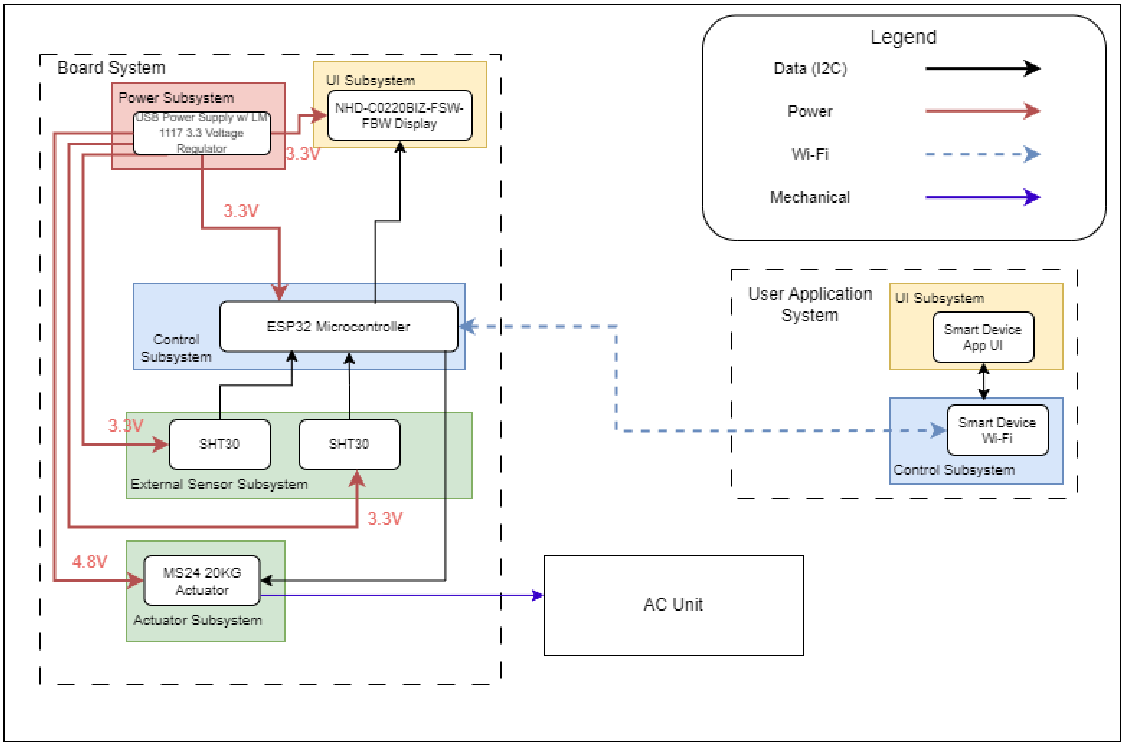


Figure 4 - Block Diagram of Smart AC Device

2. Design

2.1 Equations & Simulations

Heat Index = 0.047*H + 1.1*T - 10.3

2.2 Design Alternatives

Originally, the user application allows for complete control of the AC unit via Bluetooth which is how the information will be sent to the microcontroller. The ESP32 also receives data from the humidity sensor, HDC3020, and the temperature sensor, MCP9808, and will send the necessary data to the NHD-0108 HZ display and the FS90R actuator subsystem. The actuators communicate with the AC unit to rotate the knobs. Each subsystem is powered via USB.

As the project progressed, however, changes to the design seemed necessary. To begin with, we changed the communication protocols from Bluetooth to WiFi. The Bluetooth implementation was found to be too tedious and inefficient to communicate between the ESP32 and the user application. Wifi, on the other hand, provided for more efficient integration within the app and therefore, was found to be easier to communicate with the ESP32. Secondly, we swapped our display from the NHD-0108 HZ display to the NHD-C0220BIZ. Although no improvements from a performance perspective, the C0220BIZ allowed for a wider and better resolution display than the aforementioned one. We then swapped the HDC3020 and MCP9808, our original Humidity and Temperature sensors respectively, for the SHT30 because this new sensor can obtain both the Humidity and Temperature. Therefore we can use two SHT30s to get a more accurate result. Finally, actuators were swapped from the FS90R to the MS24 20KG actuators because the original actuators were too weak to move knobs

2.3 Design Description & Justification

2.3.1 Power Subsystem

Our power subsystem consists of a USB Power Supply and a voltage regulator that would supply a 3.3V voltage source for everything on our board besides the actuator, which needed as much voltage as possible. We were able to also supply enough current to all components of the system, which was a problem and heavily influenced the confinements of what parts we were able to use since we ended up only having about an amp of headway after testing.

2.3.2 Control Subsystem

Our control subsystem was composed solely of the ESP32-S3 Mini. We chose the ESP32-S3 Mini because of its low-profile form factor as well as other reasons. One of the main reasons, admittedly, was because it was what was available to us at the time. We were working with a different developmental board for a week before we realized that the board was not functional. We just happened to source an ESP32-S3 Mini development board, and since it was small and worked just enough for what we wanted to work with, we stuck with it to not change any variables going into soldering onto the PCB. We also chose it because it had Wi-Fi capabilities, which rounded out all the requirements for the control subsystem after everything.

2.3.3 External Sensor Subsystem

Our external sensor subsystem consisted of our 2 SHT30 temperature and humidity sensors. We initially were going to go for a more barebones HDC3020 IC humidity sensor paired with an MCP9808 temperature sensor, but we opted for the SHT30 for a couple of reasons. One reason was that the two sensors we were going to go with initially ended up not being able to be implemented on a breadboard–to our knowledge. They were small sensors that would need to be soldered onto a PCB, something we were not ready to do in the prototyping phase when we received them. Instead, we opted for two SHT30s which would handle both temperature and humidity as well as be a lot easier to prototype with on the breadboard we were working with.

2.3.4 Actuator Subsystem

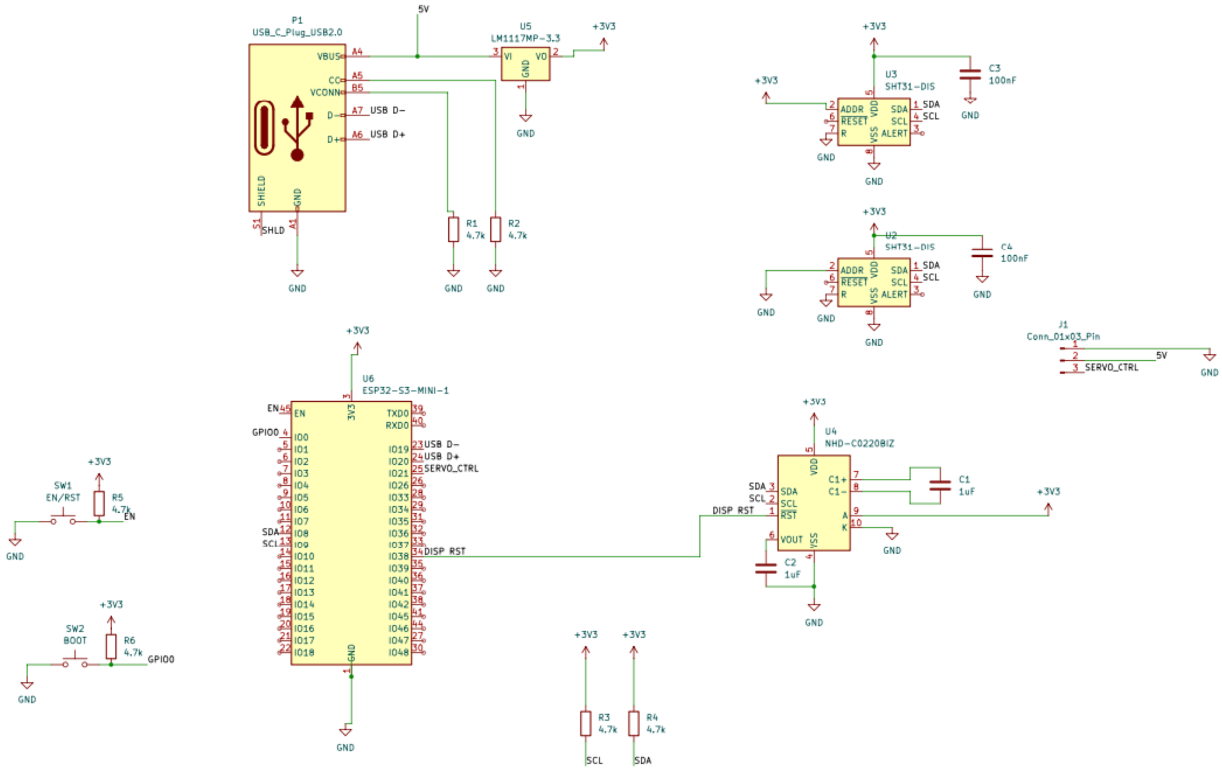
The actuator subsystem consisted of a single MS24 20KG servo motor which, along with some bearings, would attach to the knobs of our AC unit. The decision to opt for an MS24 20KG came late in our prototyping process since we had to wait for the procurement of our AC unit before we could decide on how much torque our actuator would need to successfully turn the knobs. However, after watching some demonstration videos of just how much even 10KG-m of torque could do, we opted for the highest torque available for less than 2.0A of current–since we ended up being low on available current after everything else was put together. We also had to opt for using only one servo motor due to the aforementioned issues with providing enough current for our entire system but also because of the issues with keeping two servo motors on the AC unit at once. However, opting for one servo ended up being alright since only one was needed to be able to manipulate the knob into going into all the settings of the servo.

The actual bearings for the servo motor were composed of two 3-D printed pieces that would connect to the servo wrench and snap onto the face of the knob of the AC unit. However, some unfortunate events occurred and the pieces were not strong enough to withstand the torque being put on the knob and bearings. Thus, we had to jerry-rig some solution that would be able to demonstrate our functionality–which looked like a nail taped and super-glued. The system worked such that it would prevent the servo motor from rotating and instead apply the torque onto the knob, thus making it turn.

2.3.5 User Application Subsystem

Our user application subsystem was composed of an Android application developed in Java in the IDE "Android Studio." This part of the system was rather arduous due to our lack of experience developing Android applications. We began with the idea of utilizing Bluetooth technologies to communicate with the ESP32 in the control system, however, we later chose to go with Wi-Fi due to its greater ease of development. Not only was the debugging easier for Wi-Fi as opposed to Bluetooth, but the documentation for Wi-Fi communication between the ESP32 and an Android application developed in Java was a lot more thorough and plentiful. There were an abundance of guides and demonstrations available online which made the development process a lot less painful. The way the application worked was it would go through various screens, acquiring the data necessary to set the AC unit system to the user's desired specifications. The application would thus translate this data into an 8 to 15-character string which would be sent to the ESP32 to be deciphered into a set of instructions that the actuator would thus carry out.

2.4 Subsystem Diagrams & Schematics



3. Cost & Schedule

3.1 Cost

3.1.1 Labor

For labor costs, we will assume an hourly wage of $40/hr and around 60 hours to complete. Therefore, ($40/hour) x 2.5 x 50 hours to complete = $5,000 per group member, so 3 members will bring the total cost to $15,000 in labor costs. The total cost of the products is $214.20. Therefore, the total cost would be

3.1.2 Parts

Description	Manufacturer	Part #/Product ID	Quantity	Price	Link
5,000 BTU Arctic King Window Air Conditioner	Arctic King	WWK05CR01N	1	$164.00	https://www.walmart.com/ip/Arctic-King-5-000-BTU-115V-Window-Air-Conditioner
ESP32S3-MINI Micro controller	Espressif Systems	3-01-1287	1	$9.99	https://www.amazon.com/HiLetgo-ESP-WROOM-32-Development-Microcontroller
FS90R Servo Motor	Feetech		2	$13.98	https://www.amazon.com/FS90R

MCP9808 sensor	Microchip Technology	1782	1	4.95	https://www.adafruit.com/product/1782
HDC3020 IC	Texas Instruments		1	$1.65	https://www.ti.com/product/HDC3020#order-quality
NHD-0108HZ-FSW-GBW	Newhaven Technologies	NHD-0108HZ-FSW-GBW-ND	1	$9.63	https://www.digikey.com/NHD-0108HZ-FSW-GBW
NHD-C0220BIZ-FSW	Newhaven Technologies	NHD-C0220BIZ-FSW-FBW-3V3M	1	$11.41	https://www.digikey.com/NHD-C0220BIZ-FSW-FBW-3V3M
SHT30	Sensiron	5064	5	$17.90	https://www.mouser.com/ProductDetail/403-SHT30-DIS-B
USB-C Headers	GCT	SB4115-03-C	10	8.85	https://www.mouser.com/ProductDetail/640-USB4115-03-C
MS24 20KG	Miuzei	B07HNTKSZT	2	$27.98	https://www.amazon.com/Miuzei-Torque-Digital-Waterproof-Control/dp/B07HNTKSZT
Thick Film Resistors - SMD 4.7 kOhms 250 mW 0805 1%	YAGEO	RC0805FR-104K7L	100	$1.00	https://www.mouser.com/RC0805FR-104K7L
Multilayer Ceramic Capacitors MLCC - SMD/SMT 100nF 25V 10%	Vishay	VJ0805H104KXXAT	10	$3.89	https://www.mouser.com/VJ0805H104KXXAT
Multilayer Ceramic Capacitors MLCC - SMD/SMT 25V 1uF X7R 0805 20%	KEMET	C0805C105M3REC7210	10	$1.03	https://www.mouser.com/C0805C105M3REC7210
ESP32-S3-Mini					
Switches LDO Voltage Regulators 800mA Low Dropout Positive Regulators Adjustable and Fixed 2.85V,3.3V, 5V	Analog Devices Inc.	584-LT1117IST#TRPBF	1	$6.39	https://www.mouser.com/ProductDetail/LT1117

3.2 Schedule

Week	Task	Person
2/26	Research Bluetooth Connectivity	Kevin
	Order Parts for Prototype	Xavier
	Begin Assembly Board	Vineeth
3/4	Begin Prototype Build	Kevin
	Continue Board Assembly	Xavier
3/11	Begin Development of User Application	Vineeth
	Ensure Power Unit USB-PD Communication System Works	Kevin
	Finish Board Assembly	Xavier
	Begin PCB Designs	Vineeth
3/18	Start Work with ESP32 for Heat Index	Kevin
	Connect Actuators to AC Unit Knobs	Xavier
	Continue Web Application	Vineeth
	Build the sensor unit	Kevin
	Finish PCB & Pass Audit	Xavier
4/1	Incorporate the UI Subsystem Display Unit	Vineeth
	Finish Control Unit	Kevin
	Continue Web Application	Xavier
	PCB Revisions	Vineeth
4/8	Finish Mobile App with Bluetooth capabilities	Kevin
	PCB Orders	Xavier
	PCB Revisions	Vineeth
4/15	Integration testing	Kevin
	Finalize Product	Xavier
	PCB Revisions	Vineeth
4/22	Fix Last Minute Bugs	Everyone
	PCB Revisions	
4/29	Demo	Everyone

4. Requirements & Verification

4.1 Completeness of Requirements

4.1.1 Requirements & Verification Table

Power Unit The system's power unit must have a consistent supply voltage that can consistently power the entire system at standby and full power.	Connect the voltmeter to the VCC pin, and ensure that the voltage displayed is consistent and is within ±5% of the 5V VCC supply. Check output voltage from the voltage regulator using a voltmeter to make sure that	Completeness: Success FINISHED: USB has automatically regulates voltage to between 4.4V-5.5V. Hence, our consistency in supply was found and measured. We measured

	ESP32, Display, and external sensors are all receiving 3.3V. Check the output voltage to the MS24 20KG actuator, and ensure it is at 5V. To test full power we shall run both actuation units at full speed, have the display run, as well as be actively transmitting data from sensor units to a mobile device, and ensure that all parts are fully powered. Repeat the initial test to ensure stable voltage.	supply voltage through a Multimeter provided in the ECE445 lab. We observed consistently between 4.85-5.2V on the output of the USB. 3V3 regulator regulated to around 3.46V. Was able to transmit and operate the actuation units at the same time. Voltage from the regulator remained constant at 3.46V, USB was constant at around 5.1V.
Sensor Unit The system's sensors will successfully measure humidity within ±0.5% of the room's Real Humidity as well as the temperature of the room within ±0.25℃. Thus, the system should be able to calculate the heat index within ±0.2881℃.	Put the sensor unit in 4 different locations: In the refrigerator, freezer, near a humidifier, and out on a desk as a control. Acquire an accurate thermometer and hygrometer to verify temperature. Output air from a humidifier should be close to 100% humidity, the sensor unit should read 95 ± 5% humidity. Moving the sensor unit from the desk to the humidifier should not significantly change temperature measurements Put the sensor unit in the refrigerator with the accurate thermometer and hygrometer, and close the refrigerator door for 20 minutes. The temperature and humidity readout from the sensor unit should be within 2F from the	Partially Complete: Due to budgetary conditions, the best sensors we could buy only had a guaranteed accuracy of ± 2% and ± 2 C in temperature. When exposed to air from a humidifier, the humidity detected quickly increased to over 90% humidity. However, the humidity sensors we used cannot accurately measure humidities > 90%. When moving the sensor to a lower humidity area, humidity seemed to "stick" at high humidity for a few minutes before coming down. This behavior is likely caused by water droplets entering the sensor and interfering with the humidity detection. Sensor testing was marred by issues of heat production within the ESP32 module.

	the true value of the temperature measured by the thermometer, and ± 5% from the true humidity value of the hygrometer.	Hence, on the final PCB, none of the sensors could accurately read temperatures below 30 C. However, during prototyping, we were able to measure the temperature inside a refrigerator. The temperature was within 2F of the true temperature of the refrigerator (sensors measured 38.17 F with a true value of 39 F). Hence, we listed this as a partial success
Knob Actuation The system will manipulate the knobs of an Arctic King window unit to change its cooling settings according to the specifications of the user.	Connect Knob Actuator to the Air conditioning control knob. Repeatedly turn the knobs from off to on, and repeat for 100 cycles. The knobs should be able to turn the window unit consistently and be reliable for a long time.	Partial Success: The knob was able to be consistently turned. However, the actuation broke the mount keeping it on the AC Unit. Hence, a stronger frame is needed to connect the knob to the AC Unit.
Display Unit The system will visibly display the data received from the microcontroller, including, temperature, humidity, and heat index.	Ensure the temperature displayed by the display unit is accurate to the one displayed through the sensor system.	Success: Compare values displayed by the unit and the values returned through serial control. Values displayed by the display and serial match exactly.

UI Unit The system will allow the user to seamlessly input their desired settings for a room's temperature. The UI unit allows for instantaneous temperature adjustment The UI Unit allows for the ability to plan temperature schedules like a smart thermostat. The UI unit should also display the current temperature and plan.	Start with the AC Unit off. Turn the AC Unit on and turn the desired temperature higher than the temperature of the room. The AC Unit should then turn off. Set the desired temperature very low, the AC Unit should then turn on at a very high setting to get the room to the desired temperature. Set a schedule that has the desired temperature fluctuate depending on the time of day. In the daytime, it is set to temperature A and at night time it is set to temperature B. At the transition point, the AC should suddenly turn on or off to mark the transition point. Compare the displayed temperature on the UI Unit and Display Unit, and make sure they are identical.	Success Testing was done by placing bags of frozen peas on the sensor and alternating it with a heat gun. The bag of frozen peas would lower the temperature and the heat gun would increase it. The program was able to respond to changes in temperature, and adjust the knobs to turn on or off the AC unit accordingly.
Mobile Control Unit The user application will be able to communicate with the Smart AC device via Wi-Fi. Changing settings on the app should also change the settings on the AC. If we toggle the units from F to C, the AC should reflect that and now display all temps in C.	The UI App should be able to communicate with the ESP32 and change settings via WiFi. Ensure all settings changes are reflected accurately within the device.	Success. Device settings were able to be changed, allowing for different "transition temperatures" to occur.

4.1.2 Tolerance Analysis

4.1.2.1 Heat Index

When dealing with temperatures lower than 80 degrees Fahrenheit, the typical Rothfusz regression is replaced by the Steadman formula [3].

$$\text{Heat Index} = 0.047*H + 1.1*T - 10.3$$

In this representation of the formula, H stands for the relative humidity in percent and T is the temperature in degrees Fahrenheit.

This, combined with the perceptible difference in temperature a human can feel is 2F, makes it such that our total margin of error for temperatures and AC operation is 2F[4].

4.1.2.2 MS24 20KG Actuators

Components:

1. Operating Voltage Range: 5-6.8V
2. Operating speed at 5V: 0.16 sec/60°
3. Operating speed at 6.8V: 0.14 sec/60°
4. Stall torque at 5V: 18 kg-cm
5. Stall Torque at 6.8V: 21.5 kg-cm
6. Stall current at 5V: 1.8A
7. Stall current at 6.8V: 2.2A
8. Dead band width: 3 µsec
9. Pulse width range: 500～2500 µsec (0 ~ 270°)
10. Neutral position: 1500 µsec
11. Control angle: 270°
12. Operating Frequency: 50-330 Hz
13. Neutral position: 1500 µsec
14. Stall current: 2100mA±10% (4.8 V) -2700mA±10% (6.8V)
15. Peak stall torque: 20.5kg·cm (284.7oz·in) -22.8kg·cm (316.63oz·in)

Descriptions:

- The provided voltages range of the actuators provide with necessary supply voltage to operate and twist the knobs of the AC unit
- The dead bandwidth allows for the control of different rotational positions and speeds
- The 270˚ rotational angle allows for wide range of positioning that the servo may need to access
- The stall torques show that the servos can hold heavier objects, control mechanisms with greater friction, and counteract opposing forces. The Higher voltage of 6.8V compared to the 5V indicates that the servo can increase its torque by 19.4%. [((21.5-18)/18)*100 = 19.4%]

- The stall current shows that the servo can increase power consumption. The difference from 5V to 6.8V in terms of current increase is 22.2%, [((2.2-1.8)/1.8)*100 = 22.2%]
- Servo specifications such as gear backlash and bearing slop are not specified in the documentation.

The MS24 actuators have proper requirements that would enable them to meet all the mechanical requirements necessary for this project to be able to operate successfully.

4.1.2.3 Total Room Size Cleared

At full power, the Arctic King air conditioner consumes 5000 BTU of energy to cool your home. This typically means that the AC unit can remove 5000 BTU of heat per hour. To give a power estimate, this becomes 5000 BTU = 5275279 J. We have 5275279J/3600 s = 1465 W.

The average US bedroom is about 3.7m x 3.7m x 2.5m with a window of around 1 square meter in size. We can use the known thermal transmittances table from Wikipedia[9]. We will assume the worst-case scenario: with poorly insulated walls, floors, windows, and roofs. We have roughly (3.7*3.7) = 13.69 m^2 of roofs with the same amount of floor. We also have (4*2.5*3.7) - 1 = 29 m^2 of wall, and 1 m^2 of window. We know our AC unit has enough power to cool 1465W. Hence, putting this all together with a temperature delta of T, we have (1 + 1) * 13.69*T + 29*2*T + 4.5*1*T = 1465. From this, we get a temperature delta of 16.3. Hence, there can be at max a 16.3 degree Celsius temperature difference between the outside of the room and the inside assuming very poor insulation. As normal room temperature is around 22 C, we can reliably cool a single room to a comfortable temperature for all outdoor temperatures below 38C. However, this is assuming a worst-case scenario. We have to remember that in most situations all walls, roofs, and floors don’t have direct access to the outside of the building.

Typically, rooms are connected to other rooms with similar temperatures, hence, we can expect our AC unit to run relatively quietly even if it is very hot outside

4.1.2.4 WiFi Signal Dependencies

While the WiFi capabilities of the ESP32 allow for 50m-200m of range [12], most mobile devices only support up to 45m of WiFi range. Thus, the Smart AC Unit Device should be operated within that allotted range unless there is a concrete wall or some other impedances, such as metal or other electronic devices, inhibit the transmission of Bluetooth signals. In this case, it would be better served to be in the same room as the device to avoid packet loss or other data discrepancies that will hinder the ability of the

Smart AC Unit Device to manipulate the AC Unit itself. Moreover, it also depends on what obstructions that might block the signal [13].

5. Conclusion

5.1 Accomplishments

The project as a whole was a big success, and as a result, came with multiple accomplishments. We were able to efficiently design an app that communicated to the ESP32 via WiFi to turn the servo motors to adjust the designed AC settings. This was possible because of a fully functioning PCB that contained ESP32-S3 Mini, LM1117 voltage regulators, USB-C port, and SHT30 humidity and temperature sensors. Another big accomplishment was due to the display setback. Even though the NHD-C0220BIZ-FSW-FBW display was fried, we were still able to figure out a way to show our humidity and temperature, showcasing that our PCB works flawlessly.

5.2 Uncertainties

5.2.1 Challenges

Throughout the project, we faced a few issues that caused some major alterations or concerns for the final product. To begin with, our NHD-C0220BIZ-FSW was fried after sending 4V into it when the maximum operating voltage was 3.3V. Therefore, the data the ESP32 received from the SHT30 couldn't be displayed on the display. To work around this, we utilized the dev board ESP32 to receive data from the ESP32 on the PCB to receive the humidity and temperature data and then display the results on a display attached to the dev board.

Another unsatisfactory result was the temperature output by the SHT30 sensors. The heat caused by the operating ESP32 heated the PCB board, and due to the poor placement of the sensors on the PCB, the heat reached the sensors and provided inaccurate temperature readings.

In our original design, the user application was set to communicate with the ESP32 via Bluetooth, which was changed to WiFi due to complications with Bluetooth. However, another reason for this change was due to the ESP32-S3 Mini lacking Bluetooth capabilities, causing the switch.

5.3 Future Work / Alternatives

In the future, there are many changes that we'd probably make to our design to improve the usefulness and accuracy of the Smart AC Unit. To begin with, one of our main issues came down to what seemed to be inaccuracies within the SHT30 sensors for giving inaccurate readings of the current temperature. However, it was found that the reason the temperature readings were higher than expected was because the heat generated from the ESP32 microcontroller was causing the temperature to also increase. Therefore, we'd redesign the PCB to keep the SHT30 sensors far away from the ESP32 to avoid the microcontroller heating up and providing inaccurate temperature readings. Additionally, if we plan to mass produce this product, we should expect to expand our app's compatibility with iOS devices rather than just Android. We will also add our display to the outside of the AC unit so users will be able to view the current humidity and temperature of the room. Our last design change will be to 3-D print parts that will be used to manipulate the knobs, to make our design more visually appealing to our consumers.

5.4 Ethical Considerations

While developing the Smart AC Unit, safety, and ethical considerations are very important factors to keep in mind while proceeding. Regarding safety, we plan to stick to the strict protocols and regulations and follow the correct procedures as we learned in our lab safety training. We plan on always having sufficient lab members present and carefully following all instructions and rules for safety. Moreover, our project shouldn't require anything that would require additional training to complete. In terms of ethics, we plan to uphold all standards in the 7.8 IEEE Code of Ethics sections laid out to ensure a safe and healthy professional working environment that would allow for a respectful, comfortable, and supportive working environment for the members of this team and towards other groups. Moreover, according to the ACM Code of Ethics, we plan on complying with all general ethical principles, professional responsibilities, and professional leadership principles. One specific code that we hold dearly is 7.8 IEEE Code of Ethics Section II where we respect and support our colleagues and fellow group members. Moreover, section 2.9 of the ACM Code of Ethics is also very important, as to protect all future user's information and ensure that there is no potential misuse of our product. We can avoid these breaches by ensuring a safe and helpful workplace and proper testing, monitoring, and patching to avoid potential safety concerns within our product.

References

[1] Beall, R., McNary, B. (2022). "Nearly 90% of U.S. households used air conditioning in 2020", *U.S. Energy Information Administration,* [Online]. Available: https://www.eia.gov/todayinenergy/detail.php?id=52558#:~:text=Two%2Dthirds%20of%20U.S.%20households,%25%20and%2093%25%2C%20respectively.. Accessed 20 Mar. 2024.

[2] Goldbach, K. (2023). "Many CT homes do not have air conditioning. Here are some reasons why", *Connecticut Public,* [Online]. Available: https://www.ctpublic.org/news/2023-07-26/many-ct-homes-do-not-have-air-conditioning-here-are-some-reasons-why. Accessed 20 Mar. 2024.

[3] NOAA/ National Weather Service. "Heat index equation." (2022), [Online]. Available: https://www.wpc.ncep.noaa.gov/html/heatindex_equation.shtml

[4] Kollman, R., & Betten, J. (2005). Powering electronics from the USB port. Available: https://www.ti.com/lit/ds/symlink/opa548.pdf

[5] Mazzai, A. (2023, August 3). *Yes. you can feel one degree*. Foresight. https://www.climateforesight.eu/interview/yes-you-can-feel-one-degree

[6] Industries, Adafruit. "Continuous Rotation Micro Servo." *Adafruit Industries Blog RSS*, www.adafruit.com/product/2442. Accessed 20 Mar. 2024.

[7] Espressif Systems. (2024). "ESP32 Series Datasheet", [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf. Accessed 20 Mar. 2024.

[8] Blackview. (2024). "How far does Bluetooth reach? And if through walls?", [Online]. Available: https://www.blackview.hk/blog/guides/how-far-can-bluetooth-reach. Accessed 20 Mar. 2024.

[9] Thermal transmittance. (2024). In Wikipedia. https://en.wikipedia.org/wiki/Thermal_transmittance

[10] Mousdell, D. (2024). "Calculating Heat Loss: A Simple and Understandable Guide", *H2X Engineering,* [Online]. Available: https://www.h2xengineering.com/blogs/calculating-heat-loss-simple-understandable-guide/#:~:text=Heat%20loss%20through%20walls%2C%20windows,24%C2%B0C%20%3D%2072%20W. Accessed 20 Mar. 2024.

[11] "Miuzei 6V 20kg RC Digital Servo." Available: https://images-na.ssl-images-amazon.com/images/I/81Lbgu+nG6L.pdf

[12]"ESP32 WiFi telemetry — Blimp documentation," *ardupilot.org*. https://ardupilot.org/blimp/docs/common-esp32-telemetry.html

[13] E. P. B. Chattanooga, "How Far Will Your Wi-Fi Signal Reach?," *epb.com*. https://epb.com/get-connected/gig-internet/how-far-will-your-wi-fi-signal-reach/#:~:text=Wi%2DFi%20signals%20will%20usually