Urban Noise Pollution Monitoring System

By
CJ Kompare (kompare3)
Cornell Horne (chorne7)
Marc Rhymes (mrhymes2)

Final Report for ECE 445, Senior Design, Spring 2024
TA: Surya Vasanth

May 1, 2024

Project Team 50

Abstract

This report details the findings of the Urban Noise Pollution Monitoring System (UNPMS), a system designed to collect sound data from urban settings and plot the findings on a geographical heat map. It consists of wireless, battery-powered microphones to collect sound data, a central hub to process sound data, and a web application to plot the data. The report outlines design changes and the overall design process throughout the project. While unsuccessful in delivering a fully functional design, the project was capable of collecting sound data, differentiating between safe and dangerous sound levels, and plotting a heat map with simulated data with the QCGIS system.

Table of Contents

1 Introduction

1.1 Problem

Though expansion is great, as urban cities begin to grow in both population and commercial densities, they become riddled with overwhelming noise pollution from construction, traffic, businesses, and even people. Common noise ranges are 60 A-weighted decibels (dBA) for the human voice, 85 dBA for traffic, 100 dBA for a garbage truck, and 120 dBA for a jet takeoff [1]. Typical noise pollution for an urban city such as New York City ranges from 55-95 dBA [2]. Although this range is broad, it is estimated by the National Library of Medicine that 30 million people are exposed to damaging levels of noise pollution every year [2].

Quick amounts of noise pollution are manageable for humans, but as people are continuously exposed to it in urban settings, it becomes damaging to health. At low levels of exposure to noise people are said to experience resentment, sleep disorders, insomnia, and discomfort that interfere with one's thoughts [1]. At moderate exposure levels attention, memory, and concentration are negatively affected in young people [1]. At these levels and above, the body is physically affected by the release of stress hormones, uncontrollable body movements, increased heart rate causing heart problems, and irreversible inner ear damage over time [1]. In 2007, researchers did 200,000 hearing tests and found that city residents who were regularly exposed to moderate and high noise pollution had hearing that was as if they were 10-20 years older, with irreversible damage [3].

With the health of the general public in mind, there has become a need to actively fight noise pollution in urban settings. Some examples of combative measures are soundproofing homes and governmental policies to limit noise levels. Before the government or people can combat noise pollution, however, they need to be able to accurately locate areas of high and low noise. Current methods of monitoring noise pollution include handheld devices that measure at will or over a continuous period of time [4]. These systems are meant for more short-term monitoring of small areas. Larger solutions are complex cloud-based systems made for commercial use that do analyses of noise pollution over larger areas and periods of time [4]. The Urban Noise Pollution Monitoring System is meant to be an upgrade from the handheld systems, with broader location-based monitoring and a graphical representation of numerical data. Furthermore, in contrast to the larger systems, our project will be more cost-effective and easier to implement for a common user.

1.2 Solution

The UNPMS aims to create a solution through the use of wireless, battery-powered microphones strategically placed outdoors. This system will utilize a concentrator to collect and process data from distributed microphones, providing accurate and real-time noise pollution insights for urban planning and environmental conservation through the use of a geographical information system and web application. Through the web application, users will have the ability to see both numerical and graphical data in the form of a noise level heat map. Said heat map will use the colors green, yellow, and red to differentiate between various sound levels. Users may use this to accurately determine areas of low and high noise levels. With our project, the average person will be able to use the web application to determine where noise pollution combative methods would be most effective.

1.3 High-Level Requirements

1. **Real-time Monitoring and Hourly Data Reporting**: The central hub system should successfully report noise data to the central web application every hour.
2. **Updating of Noise Map**: Noise map updates with each hourly upload from the central hub, ensuring an accurate representation of the space being monitored. It will use the colors green, red and yellow to differentiate between various sound levels.
3. **Operation Longevity:** Our system should be capable of running for at least a month utilizing a combination of proper battery choice and activation protocols.

1.4 Block Diagram

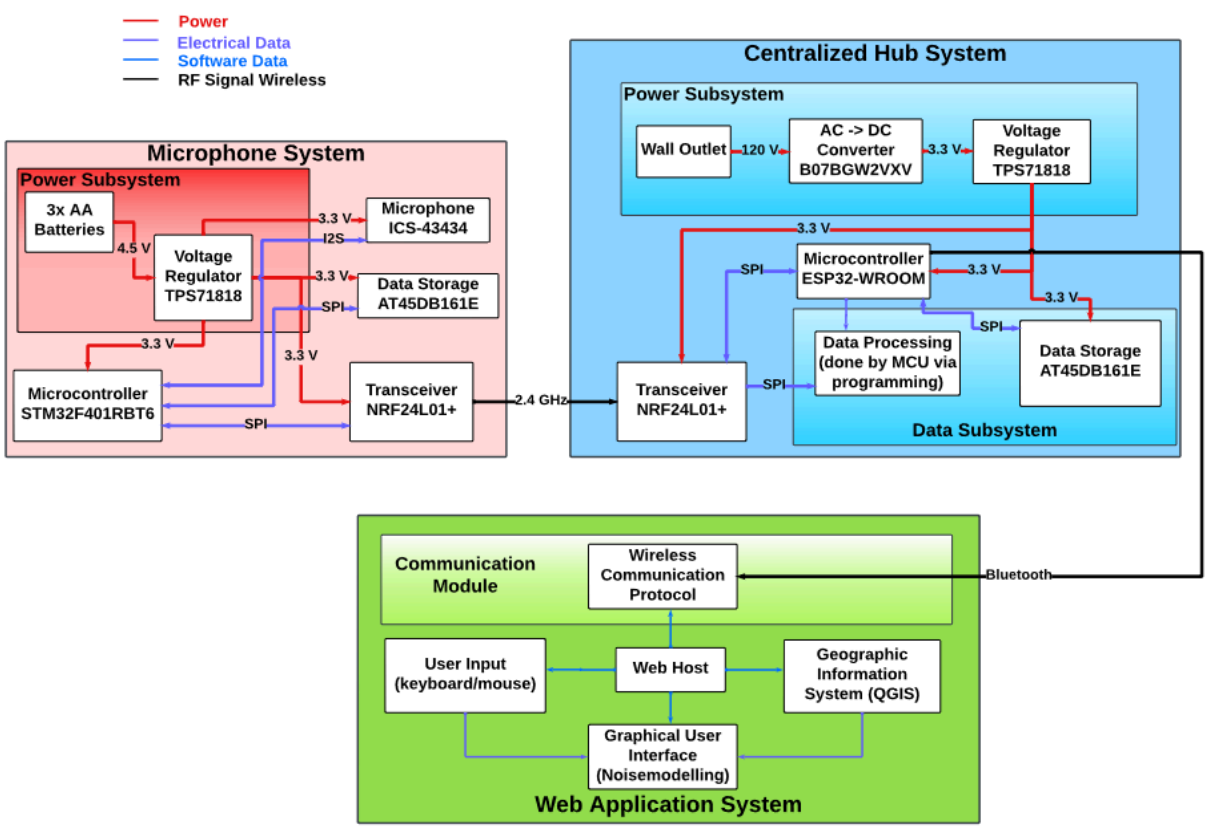

Figure 1: Block Diagram

1.5 Subsystem Overview

The design is broken up into three separate systems: the Microphone System, the Central Hub System, and the Web Application System. The Microphone System and the Central Hub System are both hardware-oriented with printed circuit boards (PCBs) created for each and the Web Application System is software-based.

The Microphone System is responsible for recording sound data from the microphone and sending it to the Central Hub via a transceiver where it can be processed. This allows for the creation of a noise map so that a user can recognize where noise pollution is present.

The Central Hub System has the main task of receiving the microphone data so that it can be processed into its respective decibel (dB) level and interfacing with the Web Application System. Along with these tasks, it must also manage the time each Microphone System is active by enacting power management protocols to ensure system longevity. By completing these, it acts as a median between the Web Application System and the Microphone System, ensuring proper function of the device.

Finally, the Web Application System is where the creation of the noise map takes place. This system must interface with the Central Hub on the host machine and be able to transfer the incoming dB data into the proper format to upload to the noise map. Without this component functioning, the user would not have a representation of the data collected.

2 Design

2.1 PCB and Housing Design

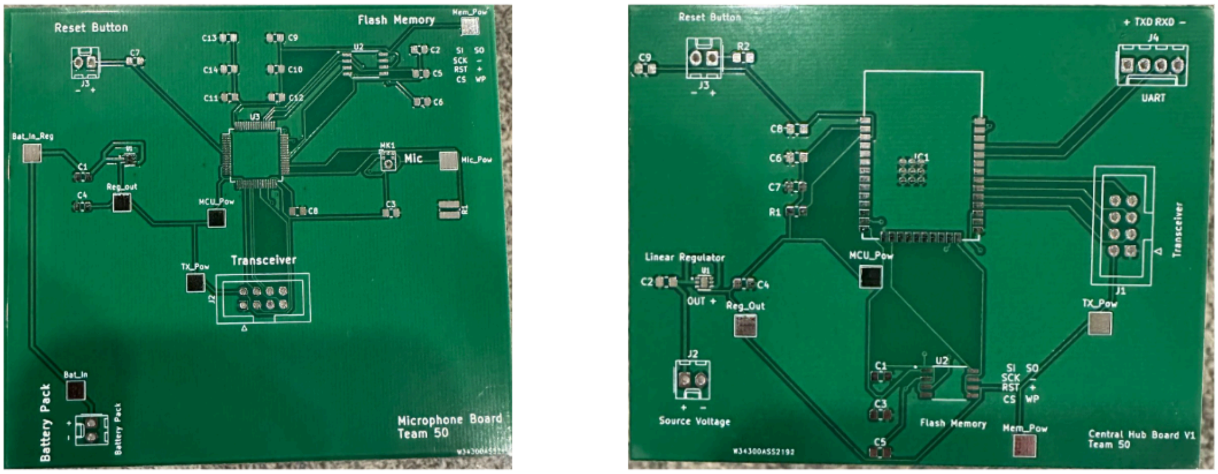

Figure 2: PCB Design for Microphone System (left) and Central Hub System (right)

The initial design for the PCBs of both hardware systems is seen above in Figure 2. The Central Hub System would use this board for the remainder of the project with slight modifications to components that would be used while the Microphone PCB would undergo changes. The new board featured the addition of a boot mode switch, crystal oscillator, debugger connections, and mounting holes. By adding these components, the MCU was able to be programmed and proper timing protocols could be enacted (*see Appendix Figure A.1*). Many extra components such as capacitors and resistors will be visible throughout the schematics and these are used to reduce noise seen by the components as well as to stabilize any inputs coming into the devices. For a more detailed view, the KiCad schematics can be viewed (*see Appendix Figures A.2-A.5*).

The housing design of the project is almost identical for both the Microphone System and the Central Hub System. Both are composed of a black rectangular case in which the PCB board was screwed into as demonstrated in Figure 3. The differences between the housings came from the Microphone System having the inclusion of a hole drilled where the microphone would be receiving sound data, along with differences in how power was delivered to each PCB (*see Appendix Figure A.6*). The Microphone System had a battery pack that was secured with velcro to the bottom of the housing and the Central Hub System had a hole drilled in where a female power adapter was installed to connect an AC/DC adapter.

Figure 3: Central Hub System housing (left) and Microphone System Housing (right)

2.2 Microphone System

2.2.1 Design Procedure

The microphone system consists of components designed to capture, process, and transmit sound data. Multiple microphone systems were included in the design to enhance the system's overall accuracy in noise mapping. The Microphone System consists of a power subsystem which is responsible for supplying the voltage to all components. All components outside of the power subsystem are connected to the microcontroller using wiring to transmit electrical signals. The signal transfer will be facilitated by the microcontroller, sending data to each component to complete their respective functions. For the purposes of this project, three Microphone Systems were created, with the idea that more could be manufactured to create a greater area of coverage.

In choosing the components for the Microphone System, considerations had to be made on the operating voltage, sensitivity, and cost-effectiveness of each device. The main consideration was to have each device operate at the same voltage to reduce the number of regulators on the board, thus reducing the current consumption. In choosing the microphone for recording the data, research was conducted to find a device that had a high sensitivity while consuming very little current. Many microphone options existed, but the ICS-43434 that was ultimately chosen provided optimal sound detection and minimal current draw. Finally, the choice of the transceiver followed a similar protocol. The NRF24L01+ was a common hobbyist component, lending itself to having a collection of open-source code to utilize to streamline the programming process. The microcontroller chosen was an STM32F401 due to its ability to interface with the microphone.

2.2.2 Design Details

To accomplish the tasks outlined in the design procedure, the MCU interfaced with the microphone via I^2S and the transceiver via SPI. The system had to transmit data to the Central Hub through the transceiver, communicating through the 2.4 GHz industrial, scientific, and medical (ISM) band. The board also had to operate at 3.3 V for one month to ensure the longevity of the system and reliable sound collection. The system is powered by 3 AA batteries with a nominal output of 4.5 V, indicating the need for the linear regulator to supply 3.3 V to each component. As the batteries only carry so much charge, considerations for power management took place. The power management strategy employed for the project was to wait for a "wake-up" message to be sent from the Central Hub's transceiver that indicated it was a specific Microphone System's time to record data. Once the message was received, the microphone would be powered on and the transceiver put into transmission mode. After a timer is cleared the system would go back to its original state with the microphone off and the transceiver in receive mode (*see Appendix Figure A.7 for flowchart*). By using the maximum current draw for each component on the board and the nominal milliamp-hour (mAh) output of a AA battery, it was shown that sending a wake-up protocol every 15 minutes to a Microphone System while recording data for 1 minute could achieve month-long longevity (*see Appendix Table A.1*). Furthermore, this wake up strategy allows for 48 hours of sound data to be recorded over the month, a large sample to create a reliable noise map.

In testing the design, it was found that the linear regulator's output was shorted. This was most likely due to its small footprint and the traces on the board being too large causing overlap of the regulator pins. To fix the regulator shorting the source voltage, a new device was utilized. The new regulator was the LP2951, which also had an output voltage of 3.3 V. The LP2951 was chosen due to its availability in the ECE Supply Shop and its larger footprint, making it easy to jerry-rig onto the PCB to correct the shorted voltage. Figure 4 demonstrates how the new regulator was soldered onto the board over the old regulator, and Figure 5 indicates the output of the regulator versus different input voltages. This regulator also boasted a lower current draw, making it more suitable for the project application.

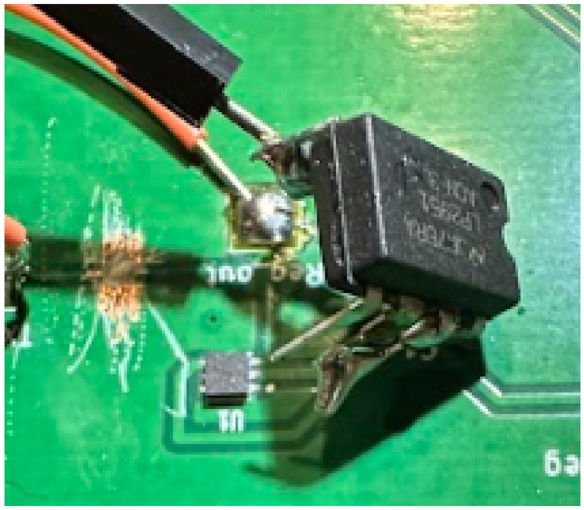

Figure 4: New Linear Regulator (right) soldered over old regulator (left)

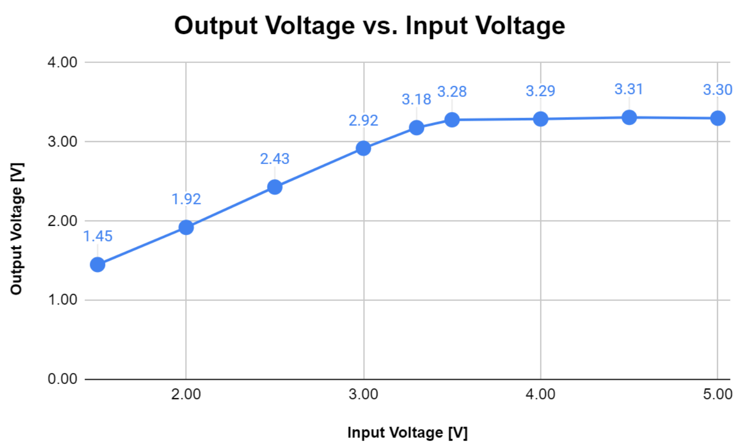

Figure 5: Plot of new linear regulator functional verification

2.3 Central Hub System

2.3.1 Design Procedure

This subsystem can be thought of as the median between the microphones and the web host. It must therefore be able to receive data from the microphones, process data, and then send it to the web host (*see Appendix Figure A.8 for code flowchart*). It consists of a power subsystem and a data flow subsystem. The power subsystem receives a 120 V input from a wall plug and outputs 3.3 V to the rest of the Central Hub via an AC/DC converter. The data flow subsystem consists of an MCU and transceiver. The transceiver is needed to collect data from the microphone and send it to the MCU. The MCU is then needed to convert byte data from the microphones to dB and send this information to the web host.

When choosing components for the Central Hub, it was necessary that all components be able to function at 3.3 V as that is the output from the AC/DC adapter. Considerations also needed to be made when choosing the proper transceiver and MCU. The transceiver chosen must be the same as the one used in the Microphone System, which was chosen due to its many open-source options and minimal current draw. The MCU chosen must be compatible with the transceiver and move fast enough for data processing. The MCU must also be able to communicate to a web host in some way meaning that it should have Bluetooth/WiFi communication abilities.

2.3.2 Design Details

The system receiving the proper power of 3.3 V was a concern for our project. Initially, the 3.3 V output for the AC/DC converter was sent to a linear regulator to ensure that the output voltage was indeed 3.3 V and that no extreme currents were being sent into our system. In the end, we found that our linear regulator shorted the input power to ground just like in the Microphone System. It was therefore removed and we decided to directly power our system via the AC/DC adapter by creating a short between the input power and the output of the regulator.

Another concern was choosing an MCU that was compatible with the transceiver chosen for the Microphone system, one that could process data fast and could communicate with the web application. The chosen MCU was the ESP32-WROOM microcontroller, which is capable of communicating with the transceiver via SPI protocol and is commonly used in conjunction with the transceiver. In terms of speed, the ESP32 transfers data at 12 Mbps, much more than any sound data received from the microphones. Lastly, it comes with built-in WiFi and Bluetooth (BT) communication abilities. The original plan was to use BT communication to our web host but it was later changed to WiFi. This design choice will be further explained in other sections of the report.

2.4 Web Application System

To import geographical data to the noise map QGIS was chosen because it is a free open-source software that allows us to easily map out geological locations. It allows options to include things such as elevation, temperature, sound, nature, and much more. It also has the ability to import data, a critical factor, because the Central Hub data needed to be converted to a noise map. Figure 6 demonstrates an example noise map of the geographic area around the ECE building.

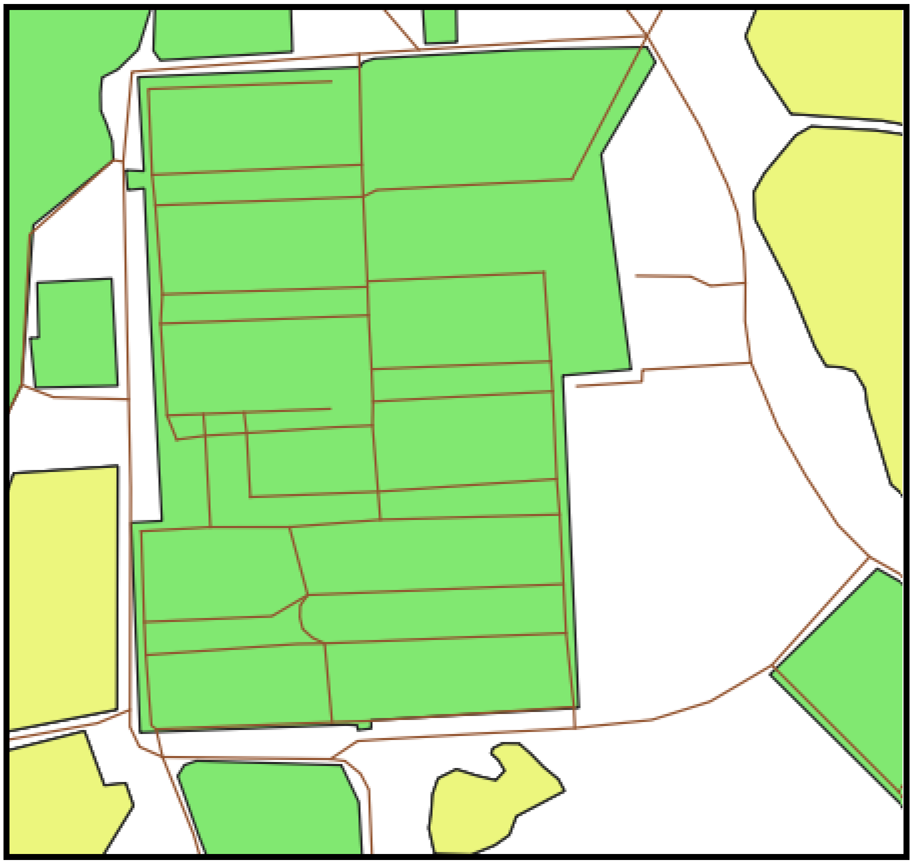

Figure 6: Noise Map of ECE/MNTL building area.

3 Design Verification

3.1 Microphone System

In order to test the proper functioning of the Microphone System, each component was to be tested on its own and verified before trying to connect everything together. This included running test programs for each component used and using a multimeter to verify voltage (*see Table A.2 for Microphone System Requirement and Verification Table*).

In the design implementation, only the voltage regulation and the transmission of data were able to be completed, as outlined by the proper terminal output in Figure 7 and regulator output from Figure 5.

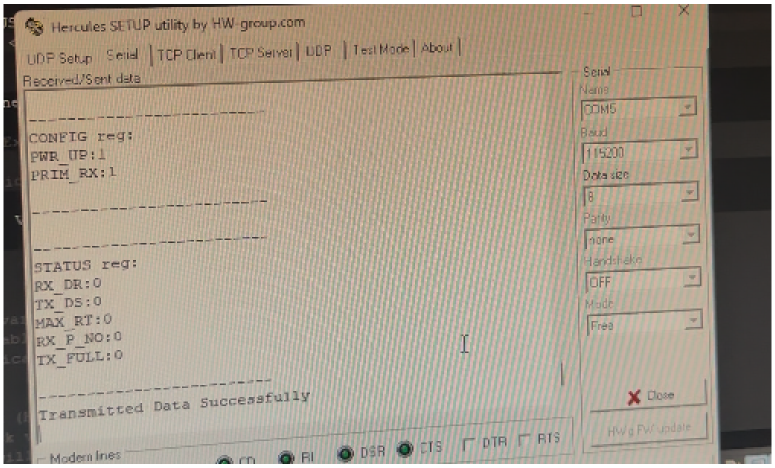

Figure 7: Transmission of transceiver working

The original microphone chosen was unable to be tested due to MCU connectivity complications, so an electret microphone was utilized to collect sound data. The circuit to test this new microphone features two amplifiers to boost the output from the microphone, as it produces very small voltage pulses in the range of 3 mV on its own. Capacitors were used to smooth the output of the microphone and resistors were used for voltage gain in the system. The final circuit for this setup is featured in Figure 8. Sound data collected with this setup is seen in Table 1, along with an iPhone recording of the same sound to get the dB value heard.

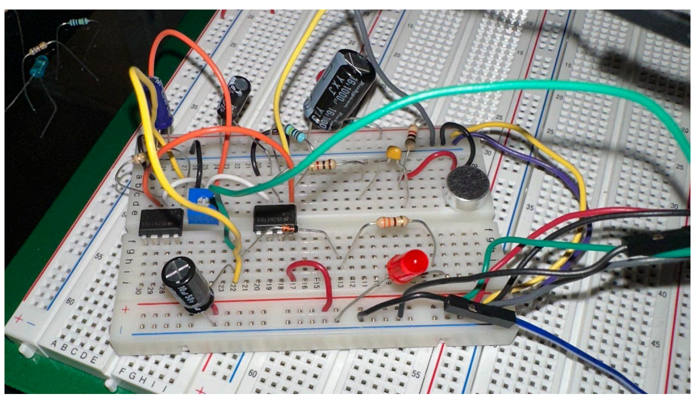

Figure 8: Final implementation of a working microphone

Table 1: Sound data recorded using the new microphone

Sound Input	Parameter	Voltage [V]	Swing [V]	Avg dB Recorded
Max Volume Music on Speaker	Voltage Min	1.321	0.382	86.2
	Voltage Max	1.703		
Loudly Talking	Voltage Min	1.427	0.208	67.3
	Voltage Max	1.635		
Talking at Conversation Level	Voltage Min	1.427	0.17	59.1
	Voltage Max	1.597		

To make this compatible with the project, the second amplifier output could have been connected to a set of 3 comparators. Each comparator would then have a reference voltage set to the voltage that corresponds to a set dB value's voltage threshold, such as 1.703 V for dangerous levels. The output of the comparators would then be sent as inputs to the MCU where an AND operation would be performed to decipher the recorded dB. This recorded dB level, categorized as low, medium or high would then be sent to the Central Hub. This implementation would offer advantages in terms of less data processing done by the Central Hub as well as the ability to test the microphone on the breadboard. On the contrary, disadvantages of setting up proper calibration on the comparators, larger board footprint due to more components, and a higher cost would be incurred.

The Microphone System was generally nonfunctional in completing the task of delivering sound data to the Central Hub. The main culprit of the inability to complete this task came from the inability to connect to the MCU. The first major issue with the microcontroller stemmed from too much solder paste being applied when using the soldering oven, essentially shorting all of its pins together as seen in Figure 9. A hazardous outcome arose from this pin shorting when the Microphone system was being transported when the battery pack was connected. One of the batteries began to short, melting the plastic of the pack and frying the PCB in the process.

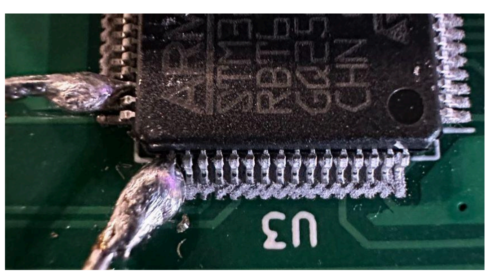

Figure 9: Microcontroller pins shorted together

This issue was able to be solved by applying less paste to the board. Unfortunately, this brought about the next issue with the microcontroller which was the debugging pins were connected to the wrong MCU pins on the PCB. As evident in Figure 9, wires had to be hand-soldered to the correct pins in order to get the debugger to connect. While success was achieved for a short moment, Figure 10 shows how the soldered wire actually broke the MCU pin off when disconnecting the debugger, rendering an unusable chip.

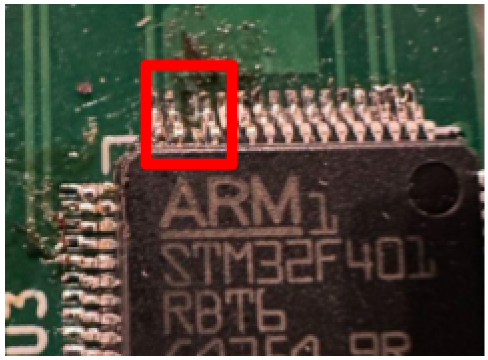

Figure 10: MCU pin broken off

A final attempt was made to hand-solder the chip, but the technique used only served to bridge many of the pins together and create another unusable chip, as indicated in Figure 11.

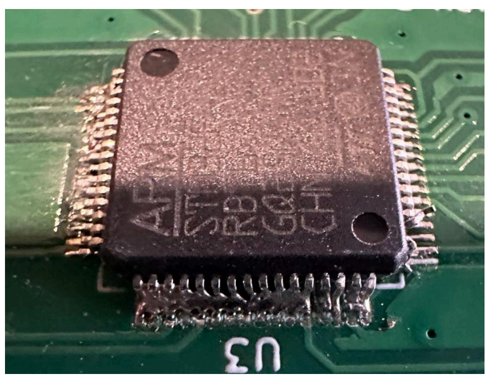

Figure 11: Hand solder failure on MCU

Due to the inability to connect a microcontroller to the debugger, the original microphone was not able to be tested. A solution to testing the microphone could have been to create a breakout board with pin connections to connect to a breadboard although a change in microphone was adopted as demonstrated.

As mentioned, the receive function of the transceiver was nonfunctional as well. It was able to output data to the terminal sporadically, however, it was not consistent and only output junk data. This output of junk data was due to the STATUS register of the transceiver being stuck in the RESET state. By not changing from this state, the data received never made it to the correct data pipe of the transceiver, and default junk data was reported as a false positive of functionality. Figure 12 outlines the junk data reported.

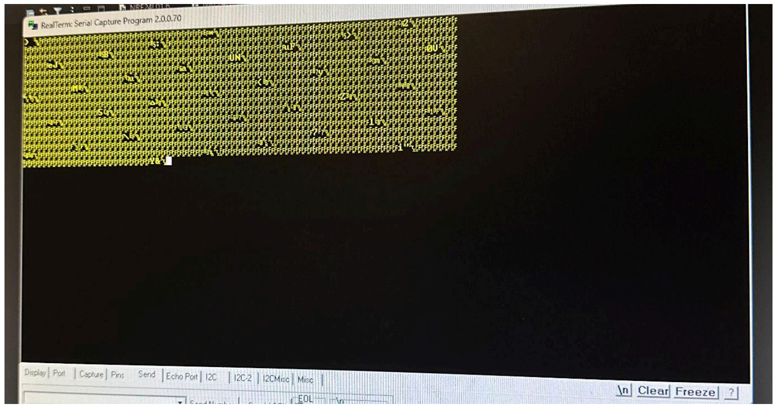

Figure 12: Receive function of transceiver outputting junk data

3.2 Central Hub System

Design verification for the Central Hub was focused on power into the system and individual components. The first verification was done on the AC/DC converter to measure the output voltage. This was tested using an oscilloscope on the PCB input pins and needed to maintain a constant voltage of around 3.3 V to be considered functional. Although removed, a similar form of testing would have been used on the linear regulator to ensure that the output voltage was indeed 3.3 V. After passing the previous test, all other components had their input voltages tested using test points and GND. Through this process, we discovered the issues with the linear regulator, which led to its removal. The same transceiver test was completed for the Central Hub as the Microphone transceiver (*see Appendix Table A.3 for Central Hub Requirement and Verification Table*).

Some functionality issues with the Central Hub System were its inability to process received data, connect to the Microphone Systems, and interface with its PCB. The ESP32 Dev Board was able to receive data and print it, but it failed in displaying the transmitted data properly. The displayed data was consistent but displayed a series of unknown characters with single-character strings mixed in. We believe that this issue was due to a failure to convert the transmitted data properly or a mix match in bandwidth between the serial monitor and transmitter. In regards to the inability to connect to the Microphone system, we believe that this was due to the different initialization processes between the STM32 and ESP32. Due to the differences in initialization, there was likely a mismatch of vital settings for communication. Lastly, our laptops were unable to connect to our Central Hub MCU and data was not able to be sent. The connection between the two was done via UART to USB. The issue was likely a hardware error in the UART to USB adapter or simply pin connections to the MCU.

3.3 Web Application System

We were never able to test the reception and updating of the data with QGIS because the Central Hub was not fully functional, however, we simulated data of the surrounding area. To do so, QGIS's tool Geo-Locator, which uses Google Maps, was utilized to find Urbana Champaign and zoomed into the ECE building area. Things such as elevation, buildings, streets, and other details that the software allowed were added. Also attempted was adding sample noise data by downloading the sample form online, and uploading it into the QGIS folder; however, the error "Open Sans Font installation" occurred. From online research it was found that this was a deeply rooted Python issue. It turned out that the hardware used was incompatible with QGIS's drivers so the full Python integers were not able to be registered, thus, not allowing the upload of the sound data. The good thing is, that it was a common issue that ended up being fixed by a tool called openNoise which allowed us to input our own controllable sound data, and add color to different regions that represented the sound. We observed that next to the ECE building there is that mound of grass where people are usually walking, throwing frisbees, highschoolers running for gym class, and more. This lent itself to depicting that area as a yellow region of noise (intermediate) and depicting the ECE building itself as the green region (lowest) because not much sound propagates from it.

4 Costs and Schedule

4.1 Cost & Labor

The projected cost for the creation of the UNPMS with three Microphone Systems and one Central Hub System is $162.80. A breakdown of the cost of each component can be found in Table 2.

Table 2: Breakdown of cost of each component

Part Description	Model Number	Quantity	Price Per Item	Extended Price	Part Link
Microphone	ICS-43434	3	$3.13	$9.39	Mic
Linear Regulator	LP2951	3	$0.77	$2.31	Lin Reg
Batteries	AA Batteries	1	$23.49	$23.49	Batteries
Microcontroller for Mic	STM32F401RBT6	3	$6.40	$19.20	MCU for Mic
Microcontroller	ESP32-WROOM-32E	1	$2.50	$2.50	Micro
Transceiver	NRF24L01+	1	$13.99	$13.99	Transceiver
AC -> DC Converter	B07BGW2VXV	1	$6.99	$6.99	AC-DC
DC Universal Output Pin	DC Female Power Plug	1	$6.83	$6.83	DC Adapter
Reset Button	GPB024A05BR	4	$3.39	$13.56	Button
Male Header	D01-9922046	1	$4.90	$4.90	Male Header
Female Header	PPPC101LFBN-RC	1	$0.68	$0.68	Header
Battery Pack	36-2464-ND	3	$1.81	$5.43	Battery Pack
Housing Case	CU-3283	4	$9.86	$39.44	Housing
PCB	PCB	4	$1.00	$4.00	PCBWay
Resistor	100 kΩ	3	$0.23	$0.69	Resistor1
Resistor	10 kΩ	5	$0.10	$0.50	Resistor2
Capacitor	10 pF	6	$0.80	$4.80	Cap1
Capacitor	0.1 µF	38	$0.10	$3.80	Cap2
Capacitor	4.7 µF	3	$0.10	$0.30	Cap3
Capacitor	22 µF	1	$0.04	$0.04	Cap4
Crystal	ABM8-16.000MHZ-10-1-U-T	3	$0.62	$1.86	Crystal
			Total Part Cost:	$162.80	

The total labor cost, seen in Table 3, for the project is based on the 265 hours total put into the project by the team. It rests on the assumption that an Illinois ECE graduate would be compensated on average $43.64/hour and machine shop workers $55/hour. The total labor cost comes out to $29,025.10. This brings the total design cost to $29,187.90.

Table 3: Breakdown of labor costs

Project Component	**Description**	**Time**	**Cost**
Central Hub	Research	40	$1,745.60
	Design	40	$1,745.60
	Assembly	6	$261.84
Microphone System	Research	60	$2,618.40
	Design	60	$2,618.40
	Assembly	15	$654.60
Web Application	Research	20	$872.80
	Creation	20	$872.80
Machine Shop	Meetings	1	$55.00
	Design	3	$165.00
		Labor Cost	$29,025.10

4.2 Schedule

Table 4 outlines the schedule taken to complete the project as well as outlines the general workload distribution

Table 4: Schedule for working on the project

Week	Task	Person
February 22nd - March 2nd	Finalize Proposal	Everyone
March 3rd - March 9th	KiCad for subsystems	Marc & CJ
	Web Application Research	Cornell
March 10th - March 16th	SPRING BREAK	
March 17th - March 23rd	Coding work for Microcontrollers	CJ & Marc
	Start Web Development for Noise Map March 18th	Cornell
	PCB Order	Everyone
March 24th - March 30th	Continue Web Application Development	Cornell
	Coding/Testing of Microcontroller Code	CJ & Marc
March 31st - April 6th	Debugging Microcontroller Code	CJ & Marc
	Continue Web Application Development	Cornell
April 7th - April 13th	KiCad Error Corrections	CJ
	Web Application Development	Cornell
April 14th - April 20th	Soldering Boards/Board Fixes	CJ
	Code Debugging for Microcontrollers	CJ & Marc
	Prepping for Final Demo	Everyone
April 21st - April 27th	Last Minute Soldering Fixes	CJ
	Prepping for Final Demo	Everyone
April 29th - May 4th	Final Presentation Prep	Everyone
	Getting Microphone Data	CJ
	Final Paper Writing	Everyone

5 Conclusion

5.1 Accomplishments

While our project was unsuccessful overall, there was progress made to demonstrate some functionality of the design, with each subsystem completing some initial design goals.

5.1.1 Microphone System

The Microphone System was successful in transmitting data with a 100% success rate. Also, with the implementation of the electret microphone, sound data was able to be collected and shown to differentiate between noise levels based on the voltage output of the amplifier. The linear regulator functioned as specified, delivering 3.3 V to each component on the PCB as well.

5.1.2 Central Hub System

The Central Hub System was successful in transmitting data with an 80% success rate, receiving data, and connecting via WiFi to a laptop. Testing for these accomplishments was done through the use of an ESP32 Dev Board and an Arduino Uno, each paired with a transceiver. In terms of transmission and reception, both boards were used as transmitters and receivers. Transmission was verified through the radio.write() function, which outputs a boolean value of 0 or 1 depending on if data was successfully sent. If the value was 1, then "sent" would be printed on the serial monitor, and "failure" would be printed on the serial monitor otherwise. Reception was verified by the printing of some variations of the characters on the serial monitor consistently. While received data was not processed properly, changing the transmission characters also changed the characters printed by the receiver, indicating some functionality. Connecting via Wifi to a laptop was done using the ESP32 Dev Board in its access point mode. In this mode, the board acts as an access point that other devices may connect to and generate a web server. The information displayed on the web server was able to be changed by the user manual through changes in code.

5.1.3 Web Application System

The Web Application System was a 100% success as its own unit. It proved to work and was capable of mapping the town of Urbana as well as the capability to map sound data as color in its respective regions. It is user-friendly and allows the user to move around the entire city, zoom in, zoom out, and much more. Although coding and firmware issues occured, solutions were found using external sources to understand our bugs and to implement the correct tools to output what was needed.

5.2 Uncertainties

A major uncertainty for this project is the reliability of the transmission distance on the transceivers. Due to our PCBs failing to connect to the microcontroller and trouble with receiving data, no data was collected on the range of the transceivers. While the datasheets for the NRF24L01+ do mention a transmission distance of 300 m with the base model, proper testing would need to be done in order to confirm this. Testing under different conditions with various lines of sight would prove to reveal the true communication distance between the transceivers.

5.3 Ethical Considerations

When it comes to the use of microphones, privacy and/or misuse of information collected is a concern. Our group acts in alignment with the IEEE Code of Ethics adopted by the IEEE Board of Directors along with revisions through June 2020 [11, 12]. We aim to uphold and abide by the standards of:

1. **Holding in high value the safety, and health of the public making sure our design complies with ethical and contained practices, and securing the privacy of others.**

   To avoid inadvertently collecting and using private conversations or voice data, our urban noise pollution monitoring system will employ a sound filtering and processing approach. The system will be designed to filter out low frequencies such as speech, as that is not dangerous to human hearing, and focus on capturing and analyzing environmental noise levels that can cause harm. This will involve implementing digital signal processing algorithms to differentiate speech and other types of environmental noise, such as traffic, construction, or industrial sounds to assure the public conversations are not recorded.

2. **Avoiding the illegal behavior of professional conduct and accepting any forms of bribery.**

   Our team understands that technology with the ability to listen in on others is very sought after in politics, the finance business, and many other professions for the aim to benefit financially or politically over another. We strive to uphold integrity by undeniably refusing to sell or be bribed by anyone to modify our idea to store the content of the conversation in order to be bought or sold.

5.4 Future Work

In our project, a major issue was communication between the Microphone and Central Hub Systems. As stated previously, the issue may have been due to different initialization processes between the two MCUs and our transceivers. While both MCUs were compatible with our transceivers, they used vastly different coding systems. For example, Arduino IDE, used for programming the Central Hub, automatically initialized radio channels while STMCubeIDE, used for coding the STM32, required the user to directly code radio channels. In the future, we would like to use a microphone compatible with the ESP32 so that we may use its WiFi/Bluetooth communication system to transmit/receive data instead of a transceiver. Not only would this method limit code but it would also lower power consumption. Ideally speaking this microphone would have accessible pins so that sound collection may be tested independently as well.

Another concern was with our plotting system which was application-based instead of web-based. Later in the design process, we realized that directly sending information synchronously was extremely difficult. A temporary solution would be to manually import the data hourly, but this takes away from complete automation. To remedy this we would have to find a cloud-based geo-mapping system that can plot sound data so that web information can be directly sent to it.

An improvement to the design would be the implementation of small solar panels on the Microphone System. Solar panels, paired with rechargeable batteries, would allow our system to run for a much longer period without a user having to manually replace batteries every month. Final improvements would be toward the transceiver and the casing. If we were to keep the transceivers in our design, adding antenna attachments could increase the overall range from roughly 300 m to up to 1000 m. In terms of casing, we would opt for a smaller casing that would allow easier placement as it would take up less area when being placed outside.

Appendix A

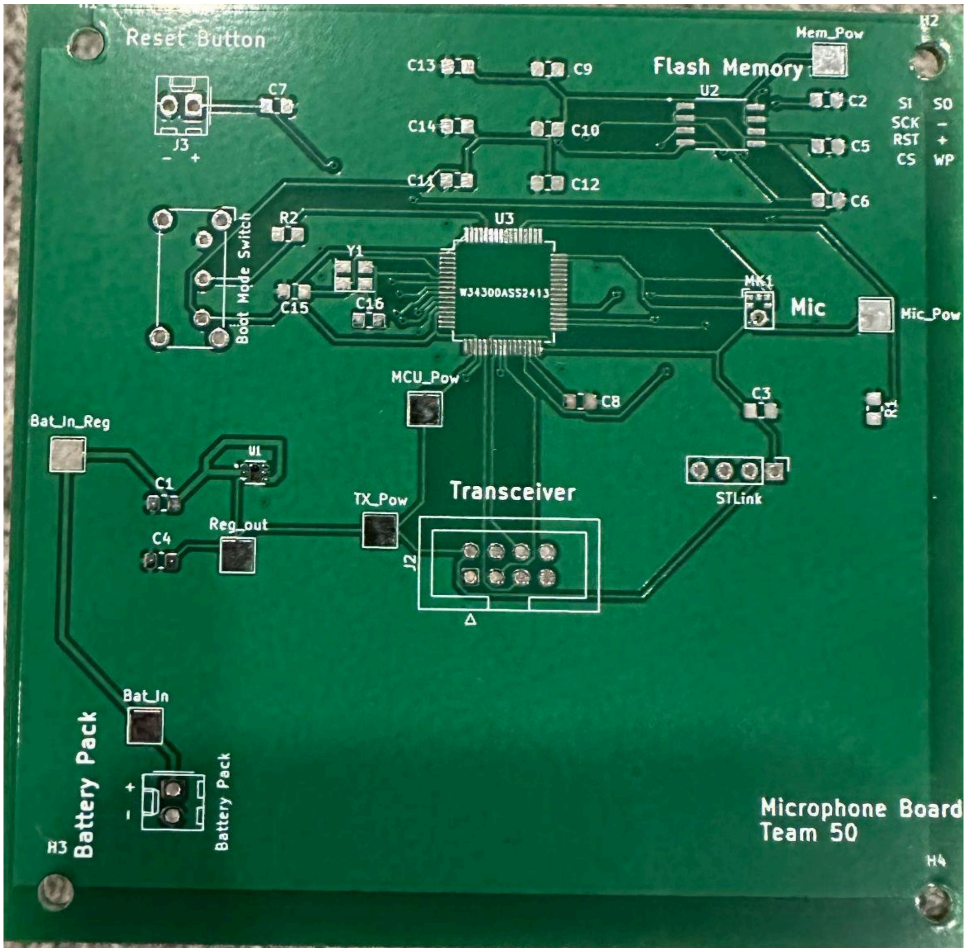

Figure A.1: Updated Microphone System PCB

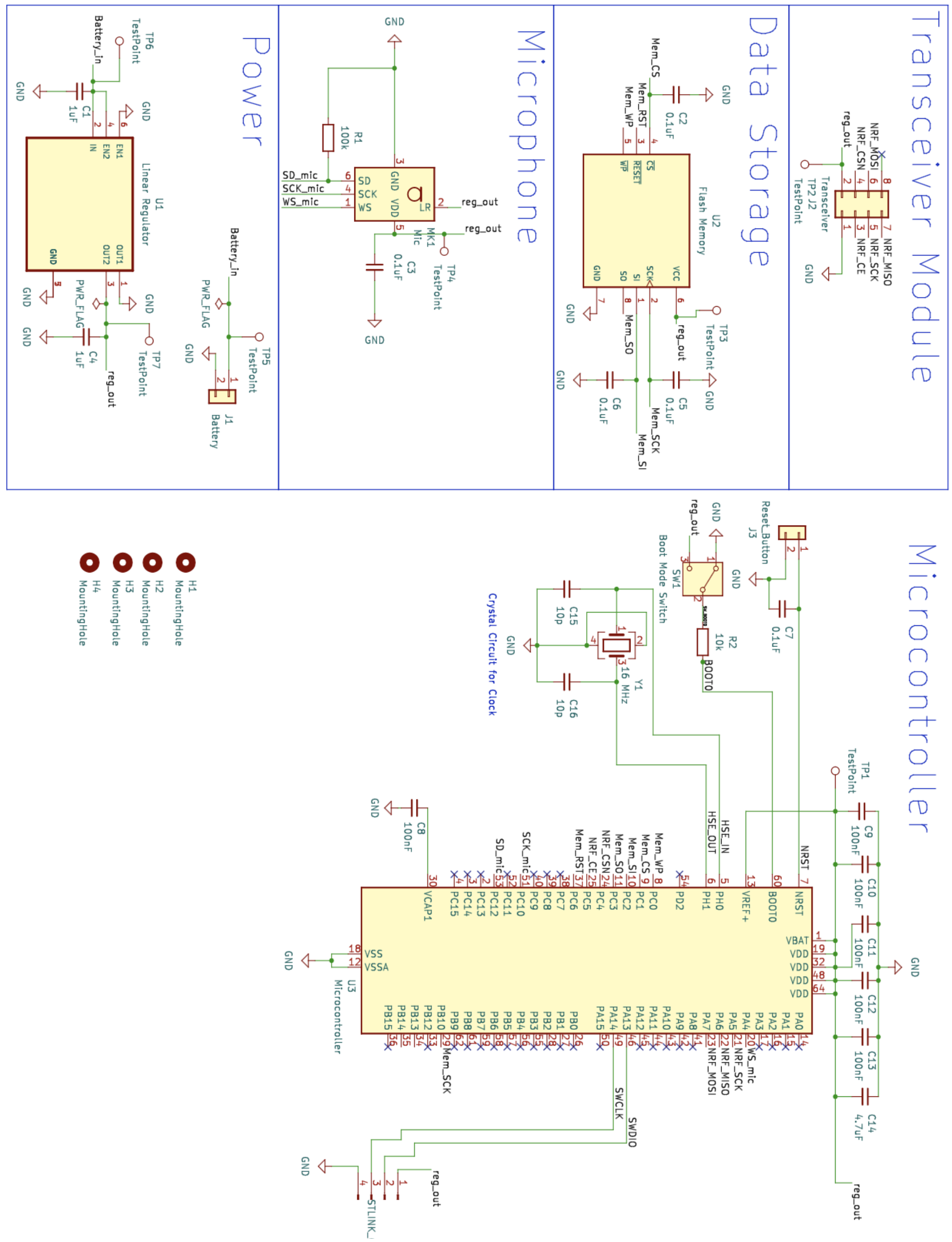

Figure A.2: Microphone PCB Schematic

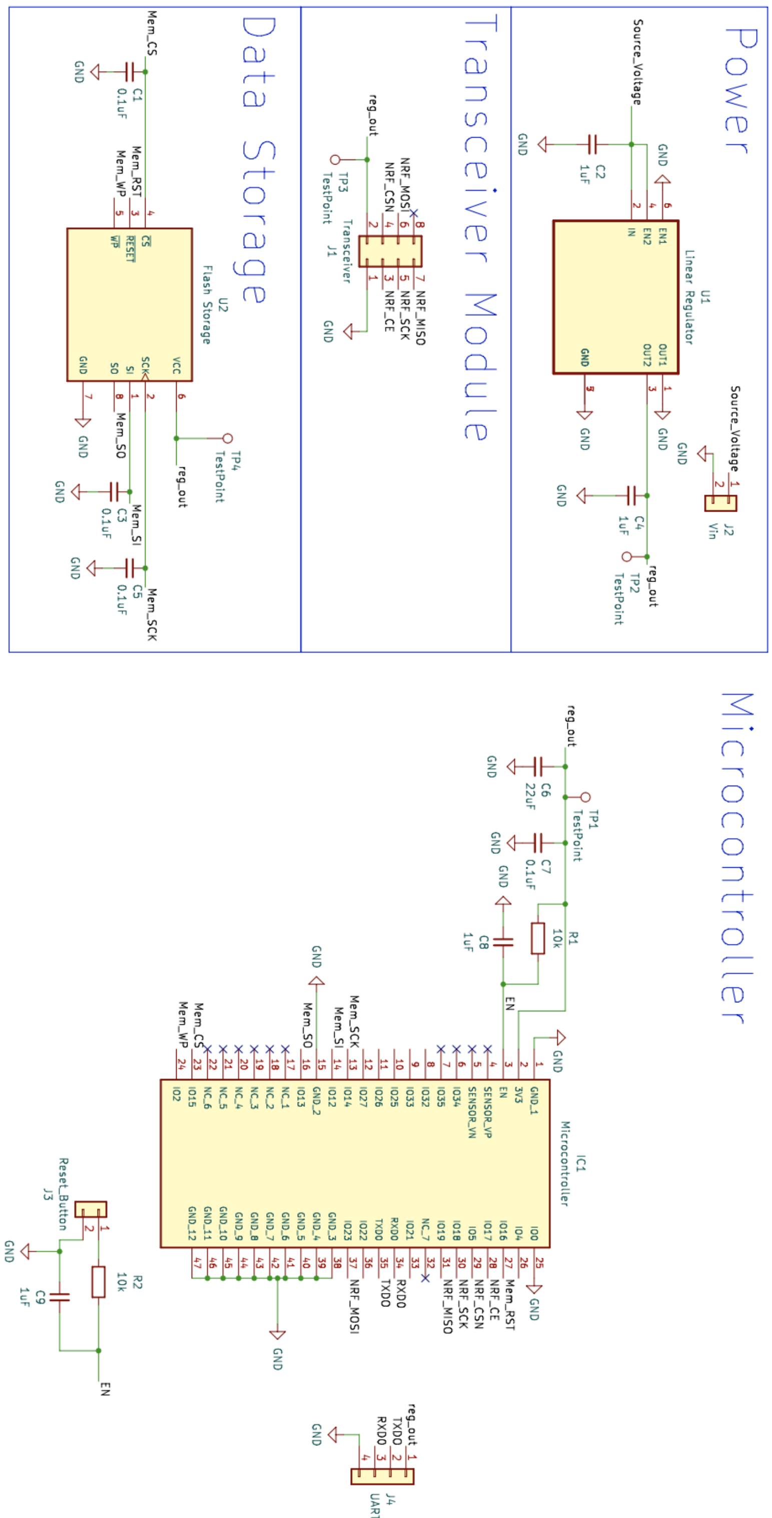

Figure A.3: Central Hub System Schematic

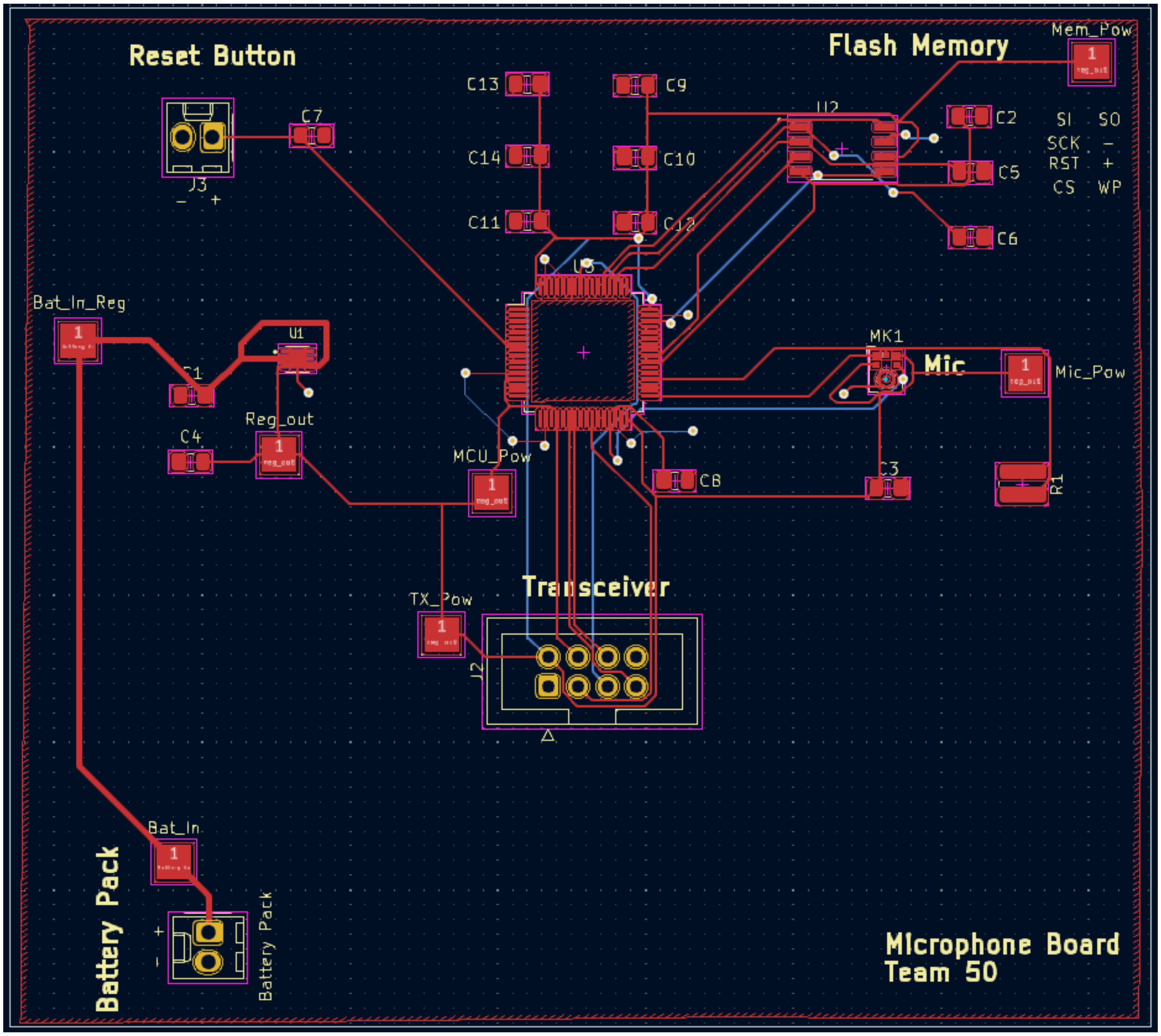

Figure A.4: Microphone PCB Layout

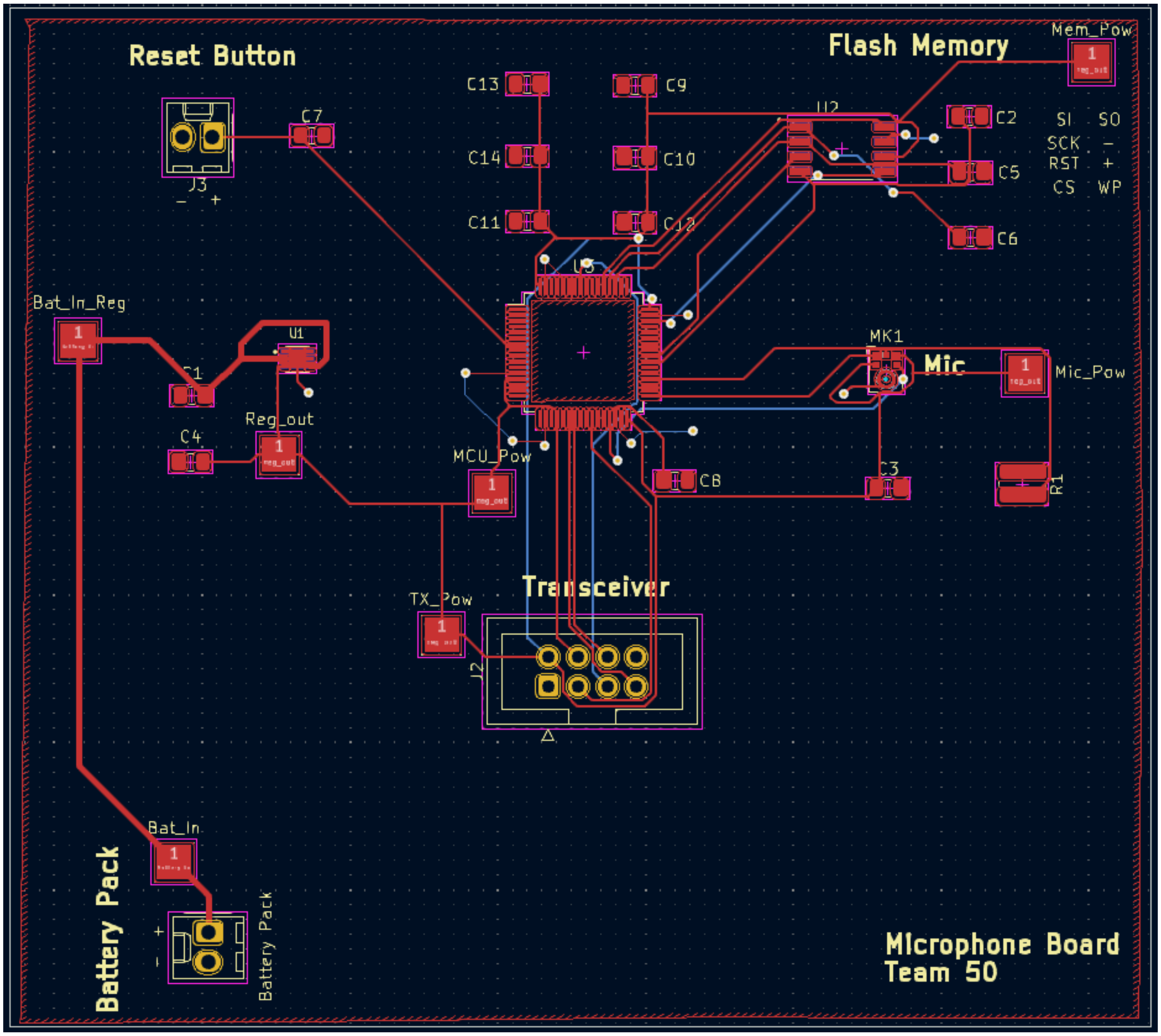

Figure A.5: Central Hub System PCB Layout

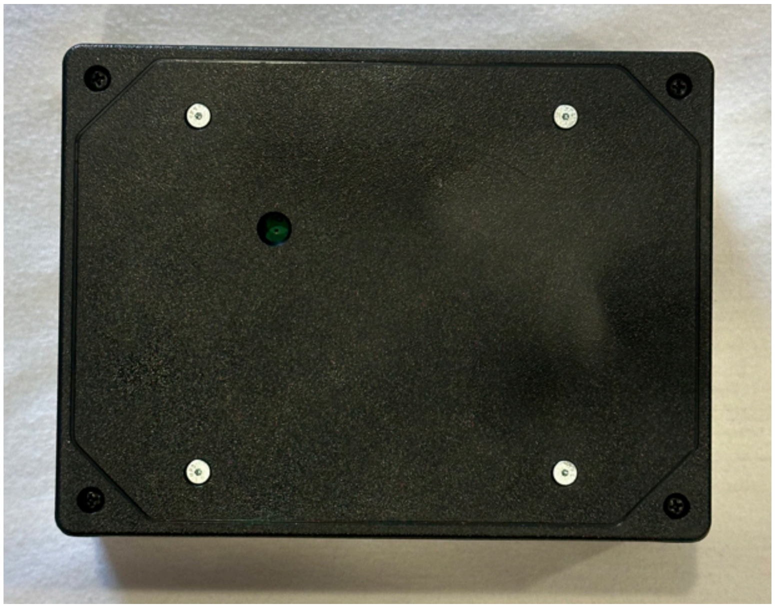

Figure A.6: Microphone System housing showing hole drilled for microphone

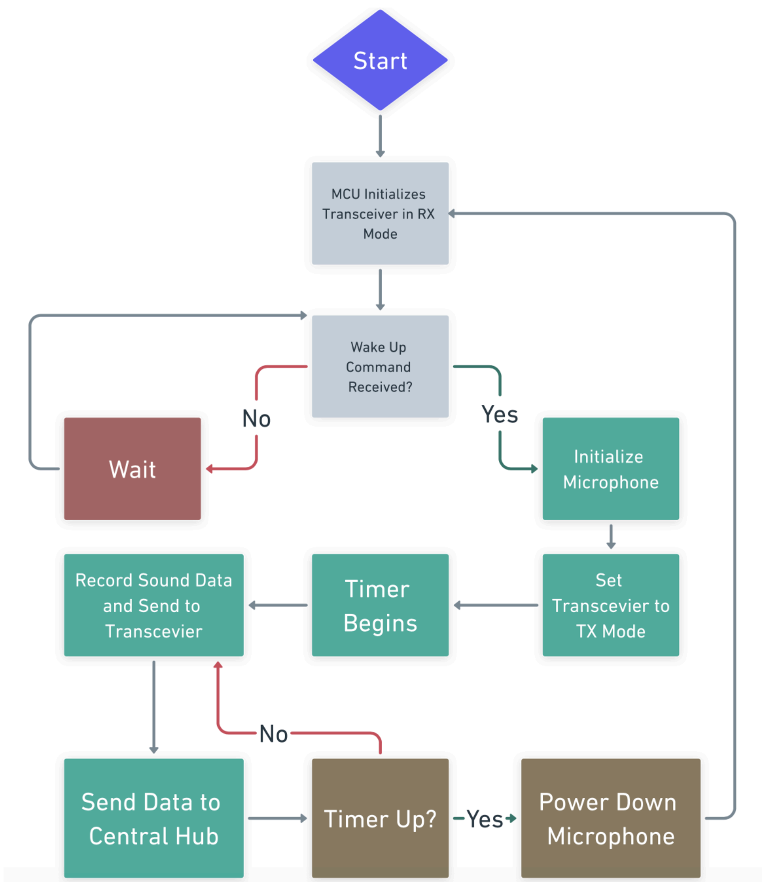

Figure A.7: Code used for Microphone

Table A.1: Current consumption of Microphone board with max current consumption

Typical AA Battery mAh:	2850 mAh

Component	**Max Current Consumption**	**Duration**	**Current Consumption in hours**
Microphone (Active) [5]	550 uA	48 hours	26.4 mAh
Microphone (Sleep) [5]	20 uA	672 hours	13.44 mAh
Transceiver (Active) [6]	8.5 mA	48 hours	408 mAh
Transceiver (Standby) [6]	26 uA	672 hours	17.472 mAh
Linear Regulator [7]	75 uA	730 hours	54.75 mAh
Microcontroller (Sleep Mode) [8]	2.9 mA	672 hours	1968 mAh
Microcontroller (Active Mode) [8]	8.0 mA	48 hours	384 mAh
		Total	2862.112 mAh

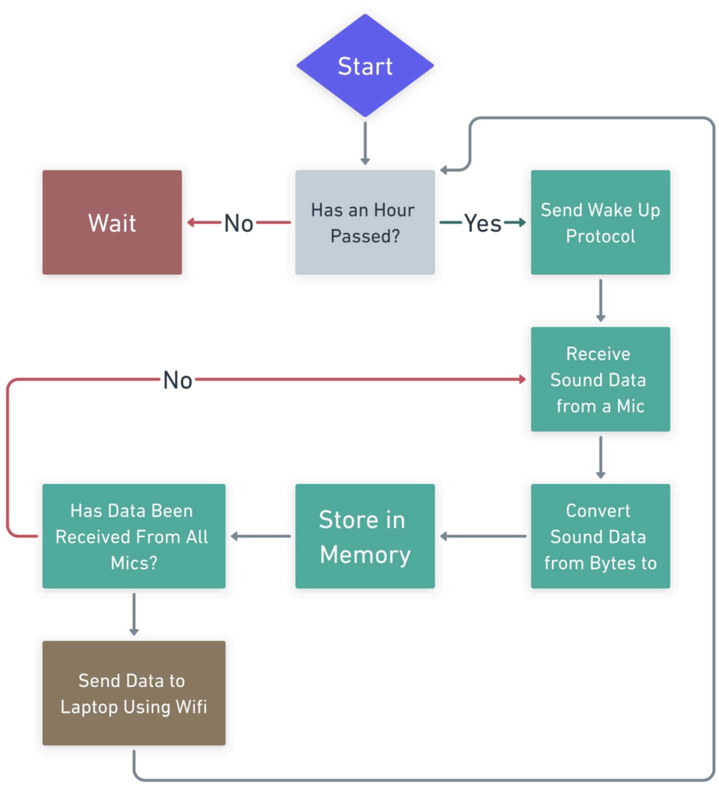

Figure A.8: Central Hub Code Flow Chart

Table A.2: Requirement and verification table for Microphone System

Requirement	Verification
1.) Must supply power continuously for approximately one month	1.) Turn on the digital multimeter 2.) Set the multimeter to DC mode 3.) Connect the positive probe to the V_{DD} pin of the linear regulator and connect the negative probe to ground 4.) Verify that the voltage is being supplied at 3.3 V. 5.) Repeat steps 1-4 weekly for one month
2.) Must regulate and correctly drop the supply voltage to each chip's respective operating point	1.) Turn on the digital multimeter 2.) Set the multimeter to DC mode 3.) Connect the positive probe to the V_{DD} pin on the component you are testing and connect the negative probe to ground 4.) Verify that the voltage is at the correct operating point
3.) Transceiver must transmit and receive data from another transceiver	1.) Make the proper SPI1 pin connections of the first transceiver to the microcontroller 2.) Make the proper SPI2 pin connections of the second transceiver to the microcontroller 3.) Supply power to both the transceivers and the microcontroller 4.) Flash the test program *NRF_test* to the STM microcontroller using STMCubeIDE by connecting the microcontroller to a serial wire debugger 5.) Open RealTerm and connect to the port that the microcontroller is connected to 6.) View the output of the terminal a.) "Transmitted Data Successfully" will print for transmission functioning correctly b.) "Hello World" will transmit for receiving functioning properly

4.) Microphone must collect and transmit sound data	1.) Connect the microphone to microcontroller by soldering the microphone to the PCB 2.) Power on the PCB by connecting the power pins of the battery pack to their respective + and - connections 3.) Flash the test program *Mic_test* to the STM microcontroller using STMCubeIDE by connecting the microcontroller to a serial wire debugger 4.) Open the serial wire debugger view 5.) Under view, select “view trace” 6.) Make noises next to the microphone and observe changes in the trace

Table A.3: Central Hub Requirement and Verification Table

Requirement	Verification
1.) Voltage must be converted from AC to DC from the wall outlet and reduced from 120 V to 3.3 +/- 0.2 V before being delivered to system components	1.) Make sure device is connected to the wall outlet 2.) Turn on an oscilloscope 3.) Connect channel one of the oscilloscope to the AC portion of the circuit 4.) Connect channel two of the oscilloscope to the DC portion of the circuit 5.) Set both channels to read the output voltage of the probes 6.) Ensure that channel one reads an AC sine wave signal and that channel two is a constant DC signal and that the voltage falls to the right tolerance limits
2.) Voltage output must be reduced within the specified operating voltages for internal components.	1.) Make sure device is on 2.) Turn on digital multimeter 3.) Set the multimeter to DC mode 4.) Connect the positive probe to the V_{DD} pin on the component you are testing and connect the negative probe to ground 5.) Verify that the voltage is at the correct operating point
3.) The system must provide short-circuit, undervoltage, overvoltage, and overcurrent protection	1.) Connect an external power supply to the input of the device where the wall outlet normally supplies power 2.) Using the external supply, provide AC voltage outside the range of the standard wall outlet values 3.) Using a multimeter, measure voltage, current, and resistance and connect the probes to the power and ground pins of each component to check that all their values are within their allowed range.
3.) Transceiver must transmit and receive data from another transceiver	7.) Make the proper SPI1 pin connections of the first transceiver to the microcontroller

	8.) Make the proper SPI2 pin connections of the second transceiver to the microcontroller 9.) Supply power to both the transceivers and the microcontroller 10.) Flash the test program *NRF_test* to the STM microcontroller using STMCubeIDE by connecting the microcontroller to a serial wire debugger 11.) Open RealTerm and connect to the port that the microcontroller is connected to 12.) View the output of the terminal a.) "Transmitted Data Successfully" will print for transmission functioning correctly b.) "Hello World" will transmit for receiving functioning properly

References

[1] R.Bhatia., " Noise pollution: Managing the challenge of urban sounds," *Earth Journalism Network,* May 2014. [Online]. Available: https://earthjournalism.net/resources/noise-pollution-managing-the-challenge-of-urban-sounds.

[2] T. McAlexander, R. R. M. Gershon, and R. L. Neitzel, "Street-level noise in an urban setting: assessment and contribution to personal exposure," *Environmental Health*, vol. 14, no. 1, Feb. 2015, doi: 10.1186/s12940-015-0006-y.

[3] I. Tech, "Noise pollution is much worse for you than you think," YouTube, https://youtu.be/5jfmzufa8qo?si=4zDZ4S9_UaKtEAN4 (accessed Feb. 27, 2024).

[4] Perfect Pollucon Services - Environmental Lab, "Noise level monitoring & testing - Perfect Pollucon Services," *Perfect Pollucon Services - Environmental Lab*, Aug. 11, 2023. https://www.ppsthane.com/noise-level-monitoring-testing.

[5] "TDK InvenSense ICS-43434," *Digi-Key Electronics*. https://www.digikey.com/en/products/detail/tdk-invensense/ICS-43434/6140312.

[6] "2.4GHz Transceiver IC - NRF24L01+," *COM-00690 - SparkFun Electronics*. https://www.sparkfun.com/products/690.

[7] "LP2951 data sheet, product information and support | TI.com." https://www.ti.com/product/LP2951