Final Report for ECE 445, Senior Design, Fall 2022

TA: Hojoon Ryu

5 December 2022

Team 8

Jake Fava

Jinzhi Shen

Alex Yuan

By

HEARING DAMAGE DEtector and Alarm system

**Abstract**

Our hearing damage detector and alarm system takes in environmental noise, converts it to a decibel reading, and informs users via LED and a graphical interface about their current sound danger level, both moment to moment with instantaneous SPL and over a session with integrated SPL.

Overall, we achieved limited success with our project. On one hand, we were not able to construct our PCB properly and were not able to extract frequency out of our sound data. On the other hand, we were able to develop software that worked well enough on a development board to be able to send consistent decibel readings to a local computer via USB and trigger our LEDs appropriately. Due to the similarity between our development board and PCB, we reason that if we got our PCB connected properly, we could use almost identical software to get the PCB working correctly.

Contents

[1. Introduction 1](#_Toc121073332)

[2 Design 3](#_Toc121073333)

[2.1 Design Procedure 3](#_Toc121073334)

[2.1.1 Block Diagram 3](#_Toc121073335)

[2.1.2 Overall System 3](#_Toc121073336)

[2.1.3 Sound Processing Subsystem 4](#_Toc121073337)

[2.1.4 User Interface Subsystem 4](#_Toc121073338)

[2.1.5 Subsystem Interconnects 5](#_Toc121073339)

[2.2 Design Details 5](#_Toc121073340)

[2.2.1 Overall System 5](#_Toc121073341)

[2.2.2 Sound Processing Subsystem 6](#_Toc121073342)

[2.2.3 User Interface Subsystem 7](#_Toc121073343)

[2.2.4 Subsystem Interconnects 7](#_Toc121073344)

[3. Design Verification 8](#_Toc121073345)

[3.1 Sound Processing Subsystem 8](#_Toc121073346)

[3.2 User Interface Subsystem 9](#_Toc121073347)

[3.2.1 LEDs 9](#_Toc121073348)

[3.2.2 Sound Report 9](#_Toc121073349)

[3.3 Subsystem Interconnects 10](#_Toc121073350)

[4. Costs 11](#_Toc121073351)

[4.1 Parts 11](#_Toc121073352)

[4.2 Labor 11](#_Toc121073353)

[5. Conclusion 12](#_Toc121073354)

[5.1 Uncertainties 12](#_Toc121073355)

[5.2 Accomplishments 12](#_Toc121073356)

[5.3 Ethical considerations 13](#_Toc121073357)

[5.4 Future work 13](#_Toc121073358)

[References 14](#_Toc121073359)

[Appendix A Requirement and Verification Table 16](#_Toc121073360)

[Appendix B Schematics and Diagrams 18](#_Toc121073361)

[Appendix C Software Flow and Outputs 20](#_Toc121073362)

# 1. Introduction

Middle and high school musicians can be subjected to harmful levels of noise daily between rehearsals, practice sessions, and performances. Cheap and effective hearing protection is available, but many students neglect using it until they start noticing the effects of their hearing damage years later. Even without considering hearing loss, long-term hearing damage can disturb the normal “balance between excitation and inhibition in the central auditory system” that can last several times longer than the time required to cause the damage [1].

To address this issue, we developed the hearing damage detector and alarm system. As the name implies, the system detects potential sound hazards: if the system detects dangerous sound levels, the system alerts the user. To use the system, the user only needs to connect it to a local computer via USB. The system then alerts the user of two different sound related dangers: dangerous sound at a given moment, which we will refer to as instantaneous SPL, and dangerous sound over a given session, which we will refer to as either daily sound exposure or instantaneous SPL. The system alerts the user through easy-to-understand LEDs, one for daily sound exposure and one for instantaneous SPL. For daily sound exposure, if the LED is on at all, the user has exceeded their daily sound exposure limit, while if the LED is off, the user has not yet reached their daily sound exposure limit. For instantaneous SPL, a green-colored LED indicates safe levels, while a yellow-colored LED and a red-colored LED represent moderate and severe danger, respectively. These LEDs help a user decide if hearing protection is necessary or not. In addition to the LEDs, the system also transfers sound data to a local computer where users can view both the raw sound data and a sound report summarizing the raw data. We believe that this system will help users through the intuitive LED system as well as the more in-depth sound report that will give users a sense of how hazardous their environment is and if they need to wear ear protection.

While commercial SPL meters exist at a comparable price point, we note that these SPL meters do not inform the users of which sound levels are dangerous in any fashion and many do not come with a USB connection to allow users to see raw sound data and a sound report. Our system instead gives users quick feedback as to how dangerous their sound environment is at any given moment.

In the design section, we will first describe the design of the system by looking at how the system functions. Also, we will look at the sound processing and user interface subsystem individually, going over the parts used, reasoning for each part, and how the subsystem contributes to the overall design. Additionally, we will go over all the changes made during the development period, including switching microcontrollers and removing the battery subsystem. In the verification section, we will describe the process of verifying our design on the development board as well as the PCB. On the development board, all the tests passed in some form or another, while we struggled a bit more on the PCB. Next, in the cost section, we will go over the costs of the parts and labor used to develop the project. Finally, in the conclusion, we will go over our concluding thoughts on the project, including what did not work, what we accomplished, ethics, and recommendations for further work.

To summarize our main conclusions, first, we found that although we were unable to get our system functioning on the PCB, we found success getting the system functioning well on the development board. While the PCB and development board are not identical, due to their myriad similarities in microcontroller, microphone, and overall functionality, we believe that if we constructed the PCB correctly, porting the software over from the development board would be trivial. Second, we believe that since the breadboard works well, and it should be easy to port over the software to the PCB, we would describe the project as a limited success since we were able to accomplish our main objective of building a hearing damage alarm system that quickly conveyed information about sound hazards.

# 2 Design

## 2.1 Design Procedure

### 2.1.1 Block Diagram

Figure 1 below gives a general block diagram overview of the system divided into subsystems and interconnections.

Diagram

Description automatically generated

Figure . Block diagram of system. See legend on bottom left for type of connection between blocks.

Next, we will go over our decision-making process for the overall design as well as for each subsystem.

### 2.1.2 Overall System

Due to our objective of converting sound hazards into warnings for users, we needed a sound processing subsystem to convert noise into digital signals and a user interface subsystem to convert digital signals into warnings.

Initially, we had a power subsystem including the voltage regulator and the USB port implicitly as well as a battery and a battery charger. The idea was that if the system had a battery, users could use the system could away from a computer, albeit without the raw sound readings and sound report. However, by the time we designed our schematic and PCB, we eventually decided against having a power subsystem with a battery and had power instead come directly from a local computer. We decided against a power subsystem with a battery because we realized that not having raw sound readings and sound report would be too limiting for users, using a battery would introduce unwanted safety hazards, and using a battery and charger would introduce additional points of failure.

Our general design philosophy for our system was to reduce points of failure as much as possible. First, we reason that less points of failure means that in the design process, we do not have to worry about as many components, so construction and debugging becomes much easier. Second, we reason that once our system is in use, less points of failure means that our system should be more reliable.

### 2.1.3 Sound Processing Subsystem

This subsystem needed to be able to convert sound into functional digital data. Due to our relative inexperience, our desire to reduce complexity and points of failure, and the objective of the project, we decided upon two main components: an I2S digital microphone and a microcontroller with I2S compatibility. We will go in-depth into our decisions below.

First, we had to decide what kind of microphone we wanted to use. We had to first decide between whether we were using an analog or digital microphone. Due to our desire to reduce complexity, we decided to use a digital microphone, as a physical microphone would require an ADC and filters when a digital microphone would not. Next, we had to decide between data formats, we could user either PDM or I2S. While PDM would require less signals from the microcontroller as well as higher audio quality, we decided to use an I2S microphone due to easier implementation as external components do not need to process I2S signals [2]. Additionally, audio quality is not as relevant to us as ease of implementation since our system does not need high audio quality as all we need are decibel and frequency values and ease of implementation will reduce the points of failure.

The specific I2S microphone we went with was the DMM-4026-B-I2S-R. The DMM-4026-B-I2S-R is an I2S microphone that supports the 20 Hz to 20 kHz range, which captures the same frequency range as the human ear [3].

Next, we had to decide on the type of microcontroller we would use. Our decision for the type of microcontroller mostly stemmed from our choice of microphone. If we were using an analog or PDM microphone, we could use a wide range of microcontrollers and pick an inexpensive and simpler option. However, since we decided that an I2S microphone would be better than any other microphone choice, we had no choice but to either choose a microcontroller with built-in I2S compatibility or a microcontroller with no built-in I2S support and try to configure I2S with our own software. Due to our inexperience with microcontroller code development, we decided that less software complexity was better than a simpler and/or less expensive microcontroller and chose to use a microcontroller with built-in I2S support.

The exact microcontroller we chose was the STM32F401RCT6. In addition to the required I2S support, the microcontroller also has USB support, which is important since it means that we can use a direct USB connection.

### 2.1.4 User Interface Subsystem

For this subsystem, we had to decide how exactly we wanted to warn the user when sound hazards were present. Following our overall design philosophy, we wanted to choose simple but effective components. To that end, we choose to use LEDs and a sound report on a local computer.

We chose to use LEDs because of their simplicity; they only require one or a few digital inputs from the microcontroller that can function as analog signals as the microcontroller only needs to update them every second.

We chose to use a sound report on a local computer because the report would only require the raw sound readings the microcontroller was already using to configure the LEDs. Additionally, the software could send these signals infrequently, e.g., every second, so long as the software accorded for the interval between signals. Additionally, while the inputs for the report are simple, we could make the eventual sound report arbitrarily complex, since computer software generates the report. Using a local computer also means that debugging becomes easier, as we could use the raw sound data on whatever USB communication line we were using.

Alternatively, we could have opted for other forms of visual feedback like a 7-segment display, sound report on a phone, or even a display module. However, we decided that the combination of LED and sound report would cover both simple instant warnings using the LED and more detailed data using the computer-generated sound report.

### 2.1.5 Subsystem Interconnects

Last, we had to decide how to connect the systems together. Due to the hardware requirements of the microphone and microcontroller, we needed to use a voltage regulator. Also, due to our user interface subsystem, we needed a way to communicate between the microcontroller and a computer. Eventually, we settled on using a linear voltage regulator and a USB port.

First, we choose a linear voltage regulator in lieu of a switching voltage regulator because while switching voltage regulators are more efficient, they require more complex circuits to use. Since we are trying to reduce points of failure and complexity and are not as concerned with power efficiency, we went with a linear regulator.

Next, we had to decide how to communicate between our microcontroller and computer. We would have chosen to use UART and use a UART to USB bridge, we decided that since our microcontroller had USB support, we decided that a simple USB port would suffice. While using UART and a UART to USB bridge would allow easier communication between our microcontroller and computer, again, it would introduce extra complexity and points of failure that we decided was not worth it given that our microcontroller had USB support.

## 2.2 Design Details

### 2.2.1 Overall System

For details on our overall system, see Appendix B for the final circuit schematic and PCB design.

As a quick aside, since we could not get our PCB working at an early stage, to evaluate and develop software as soon as possible, we used a breadboard with a development board and microphone module in lieu of testing and developing directly on the PCB. For reference, the breadboard containing the development board and microphone module behaves analogously to our schematic. For example, the development board contains a linear regulator to convert from 5 V down to 3.3 V, like our linear regulator, the TLV70230DBVR, converting from 5 V to 3 V [4]. Also, the board contains a pull-down switch to the BOOT0 pin like our PCB. Finally, the microphone module has the exact same pins as the microphone we would use on the PCB [5]. As such, we conclude that any tests done and any software developed for the breadboard would be analogous to any tests done and software developed for the actual PCB. We will go more in-depth on the similarities between the breadboard and PCB in the next subsections.

### 2.2.2 Sound Processing Subsystem

We have already mentioned the exact microcontroller and microphone we eventually went with, the STM32F401RCT6 and DMM-4026-B-I2S-R, respectively. Since this subsystem also encompasses the microcontroller software, we will briefly cover the main aspects of the software design and flow that Figure 6 from Appendix C illustrates.

First, we will cover the exact details of how the microcontroller was receiving and processing I2S data. As mentioned above, we specifically chose our microcontroller as it comes with I2S support. Thus, we used the built-in I2S functions to extract the I2S data, which is a 32-bit word, into a length 2 16-bit array. Because both the microphone we used in the PCB as well as on the breadboard only support 18-bit precision and testing on the breadboard mic, the INMP441, made it seem like the final 16 bits were noisy, we made the decision to only keep the first 16 bits of I2S data. Then, we had to filter the data through two criteria. First, the data needed to not be zero, since at least half of the data would be zero by virtue of us only using one microphone with only one of left or right enabled on the microphone. Next, the data needed to not be negative, otherwise the data would simply be noise. Here, since uint\_16, an unsigned integer of size 16, needed to process the I2S data, we rejected all data with an MSB of 1, i.e., any data greater than or equal to 32768. If the data was neither zero nor negative, then we accepted the sample. We then followed the software flow that Figure 6 illustrates to find peak values of the sample.

Finally, we processed the maximum of 16 peak values into decibel using the following conversion: , where the offset, which we found to be 70 but was originally 90, was set to be whatever value produced a reasonable value compared to other SPL meters; 2072 was chosen as it 90 dB right shifted by 7, ; and *Peak* and *PeakdB* and simply the raw peak value and peak value in decibel.

As a quick sidenote, the appropriate equation should be of the form ; however, since both equations are close to the same form and should be mostly resolved by the offset, we do not consider this difference much of an issue.

Finally, we will go over how we use the decibel value. In the case of the USB data, we simply pass along this decibel value by converting to a string and using a built-in function. For computing both instantaneous SPL, we first check against 90 dB, setting the pin of the red-colored LED high for samples that are higher than 90 dB. For samples under 90 dB, we do another check against 80 dB, setting the pin of the yellow-colored LED high for samples between 80 and 90 dB. For all other samples under 80 dB, we set the pin of the green-colored LED high. Next, to compute the daily dose exposure value, we start with an initial value of 0. Then, for each peak value, we add on a value equal to , where interval is the approximate interval between peak values in seconds. We derived this equation from OSHA’s daily dose equation and time interval equation [6]. Once the daily dose value has exceeded 1, we simply switch the value corresponding to the daily dose exposure LED from low to high.

We evaluated this software using the breadboard with the STM32F401CCU6 and INMP441. We have already mentioned some similarities between the microphones, namely their precision and data size. In addition, they also share the similar sensitivity with around -26 dB at 90 dB and a similar frequency range, 20 Hz to 20 kHz compared to 60 Hz to 15 kHz in the case of the PCB mic and breadboard mic respectively [7]. We note that both the breadboard and PCB microcontroller are from the STM32F401 line which uses the exact same software and share identical features aside from the amount of GPIO ports [8].

### 2.2.3 User Interface Subsystem

First, we chose the LEDs we ended up using, the LSM0805463V and the IN-PI554FCH, as they had our desired characteristics, namely they were easy to use and had reasonable voltage ranges. The one-color LED, LSM0805463V, accepts 3 V, which our microcontroller should be able to output and is thus perfect to show daily sound exposure [9]. Likewise, the IN-PI554FCH requires 3 ports with recommended minimum 3.4 V input, which our microcontroller should be able to output and is thus perfect to show instantaneous sound hazards [10].

As a brief sidenote, the instantaneous LED stays on one color for at least half a second to a second when the system detects a large sound so that users can notice hazardous sound easier.

Now, for the user interface, see Appendix C Figures 8 and 9 for examples of the sound report. The code accepts an array of decibel values and uses the equation to what was mentioned section 2.2.2, , to calculate the resulting daily dose value. While the code currently needs an array, we note that we could easily modify it to accept a .csv file of decibel values first and convert them into the appropriate array.

### 2.2.4 Subsystem Interconnects

For our linear regulator, the only needed features were a fixed output voltage of 3 V and accepting 5 V voltages, both things the TLV70030DCKT has [11].

Next, for our USB port, we wanted to choose a simple to use component. The 1001-001-01000 satisfies this requirement, as although it is only USB 1.0, it has few ports to worry about, which reduces the overall complexity and number of points of failure of the subsystem interconnects [12].

# 3. Design Verification

To clarify, while we were unable to verify our PCB due to various issues including improper connections and incorrect voltage to the microcontroller, we were able to verify the breadboard, which we have established to be equivalent to the PCB. Thus, all the verifications below detail tests done on the breadboard except for the sound report test, as we completed that test on a local computer independently of any PCB or breadboard.

See Appendix A for the exact specifications for each test.

## 3.1 Sound Processing Subsystem

To evaluate the sound processing subsystem, we verified the calibrated decibel readings from the microcontroller against a phone app at three different SPL levels, 60 dB, 80 dB, and 90 dB, and three different frequencies, 100 Hz, 1 kHz, 10 kHz.

Table 1. Sound processing subsystem test results

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Average from Board  (dB) | Average from Phone App (dB) | dB Difference |
| Low Frequency Tests (100 Hz Test Tone) | | | |
| Low Volume | **54.25** | **60.1** | **5.85** |
| Mid Volume | **78.8** | **80.2** | **1.4** |
| High Volume | **92.4** | **90.1** | **-2.3** |
| Mid Frequency Tests (1 kHz Test Tone) | | | |
| Low Volume | **57.25** | **60.2** | **2.95** |
| Mid Volume | **80.65** | **80.4** | **-0.25** |
| High Volume | **92.7** | **90.3** | **-2.4** |
| High Frequency Tests (10 kHz Test Tone) | | | |
| Low Volume | **54.5** | **59.9** | **5.4** |
| Mid Volume | **82.25** | **79.8** | **-2.45** |
| High Volume | **96.1** | **89.8** | **-6.3** |

In general, our subsystem was most accurate in the mid-volume range, 80 decibels. The subsystem tended to measure low volume sounds as lower than the actual value and high-volume sounds as higher than the actual value. Our subsystem also tended to measure low frequency sounds as lower than the actual value, leading to a failed verification on the low frequency and low volume test.

Our system was also most accurate in the mid-frequency range, around 1 kHz, passing verification at all three volume levels. The high-frequency range, around 10 kHz produced the most varied and least accurate results, passing the mid-volume verification but just failing the low and high-volume verifications.

Due to the nature of our project, we overall accept the results and verify that the subsystem is functioning well. While we would like the subsystem to work at all frequencies and volumes, the fact that it gives an accurate level at mid volumes and only overestimates high SPL is acceptable. Overestimating high SPL is more acceptable than underestimating high since the latter causes underreporting of dangerous SPL. Likewise, since low SPL is not dangerous and does not significantly contribute to integrated SPL, the subsystem being relatively inaccurate for low SPL is not a concern for us. Thus, we verify that the sound processing subsystem works.

## 3.2 User Interface Subsystem

### 3.2.1 LEDs

First, to evaluate the instantaneous SPL LED, we connected the LED to the output of the microcontroller related to instantaneous SPL danger. Then, we ran the microcontroller and timed the response of the LED. We found that the LED switching from a safe sound environment to an unsafe sound environment took at most 1 second. Then, from unsafe sound to a safe sound, we found that the LED also took less than 1 second to switch, averaging around the 0.7 to 0.8 second range. Since we see that it takes less than 2 seconds to go from both safe to unsafe and unsafe to safe, we verify that the instantaneous SPL LED works on the breadboard.

Next, to evaluate the integrated SPL LED, like for the instantaneous SPL LED, we connected the LED to the output of the microcontroller, this time related to daily sound exposure danger. Then, we ran the microcontroller and monitored the behavior of the LED. We were able to perform one test, playing a 1 kHz pure tone at a constant 97dB, according to the data from our microcontroller, and using a stopwatch to measure how long it took for the integrated exposure LED to turn on.

Table 2. Integrated sound exposure LED test results

|  |  |  |  |
| --- | --- | --- | --- |
| Recorded dB | CDC Recommended Exposure Time | Measured Time Until LED Turn-On | % Difference |
| **97dB** | **30 minutes** | **27 minutes, 42 seconds** | **7.6%** |

Since the CDC recommends only 30 minutes of exposure to 97dB sound and our device turned on the integrated exposure LED after 27 minutes and 42 seconds, we could verify that our system is working within the tolerance specified in our verification table, 10%.

### 3.2.2 Sound Report

To evaluate the sound report, we feed it mock values and a mock time duration for the set of values. See Appendix C for the sound report’s GUI for the mock values. Regardless, we will go over one case, where we feed the program values of 89, 91, 80, and 70 dB with a time duration of 2 hours. According to OSHA’s time interval equation , the samples have an adjusted 8-hour exposure limit of 9.19 hours, 6.96 hours, 32 hours, and 128 hours respectively rounded to the nearest hundredth [6]. Then, using OSHA’s daily sound exposure equation with a time duration of 2 hours for each sample and the given exposure limits, we calculate daily exposure to be 58.29% rounded to the nearest hundredth. Since our sound report from Figure 8. reports the exact same value rounded to the nearest hundredth, we see that the maximum error is at most 0.005%, which is less than 10%. Additionally, we see that the sound reports the average SPL to be 82.5 dB, which is exactly the average of 89, 91, 80, and 70. Thus, since we have satisfactory amounts of error in both daily sound exposure and average SPL, we verify that the sound report works.

## 3.3 Subsystem Interconnects

To evaluate the USB data transfer between our microcontroller and computer, we had the microcontroller send the decibel readings it proceeded to the microcontroller. For each sample, the data needed to be in the format of “,XXXXX”, where “X” represents a digit. Looking at Figure 7 from Appendix C, aside from the initial error, where an error message splits a sample, we see that the remaining samples are all in the correct format. We see that there are approximately 13 columns of data and 24 rows. Since there was only one error and we have at least 100 samples, we have no more than 1% transmission error, which is less than 5%. Thus, we verify that the USB subsystem on the breadboard works.

# 4. Costs

Overall, from our parts and labor, we find that our project costs $36832.17.

## 4.1 Parts

Note that the parts we have listed below are only parts used in the final PCB design. Any other parts that we either did not use for any reason or we used exclusively on the development board we do not list. Also, note that the “miscellaneous resistors and capacitors” are resistors and capacitors from either our own possession laying around or from the lab. Since we did not pay for them nor know the exact part number or manufacturer, we label them as miscellaneous.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Table 3. Parts Costs | | | | |
| **Part** | **Part Number** | **Manufacturer** | **Unit Cost ($)** | **Quantity** | **Actual Cost ($)** |
| Linear Regulator | TLV70030DCKT | Texas Instruments | $0.98 | 1 | $0.98 |
| USB 1.0 Type A Port | 1001-001-01001 | CNC Tech | $0.72 | 1 | $0.72 |
| Microcontroller | STM32F401RCT6 | STMicroelectronics | $7.44 | 1 | $7.44 |
| Blue LED | LSM0805463V | Visual Communications Company - VCC | $0.43 | 1 | $0.43 |
| RGB LED | IN-PI554FCH | Inolux | $0.67 | 1 | $0.67 |
| I2S Digital Microphone | DMM-4026-B-I2S-R | PUI Audio, Inc. | $2.18 | 1 | $2.18 |
| Miscellaneous Resistors and Capacitors | N/A | N/A | N/A | N/A | $0.00 |
| **Total** |  |  |  |  | $12.42 |

## 4.2 Labor

We did not use the machine shop, so we have no labor costs from them. For ourselves, we use the average starting salary of an ECE graduate of $92,824 from the latest Grainger salary release [13]. Then, dividing that salary by 52 weeks and 40 hours per week yields an ideal hourly rate of $44.63. Using the given formula, an average of 10 hours per week for each teammate, and 11 weeks of work, we calculate our labor cost as .

# 5. Conclusion

## 5.1 Uncertainties

For us, the main uncertainty relates to our PCB. Since we could not get our PCB programmed and working, we suggest taking a closer look at the circuit schematic and PCB design to ensure that no mistakes are present. While the PCB should be analogous to the breadboard, the fact that we had our breadboard functioning but not our PCB means that the PCB has a flaw in either its design or construction. From testing, we found that on the PCB, the microcontroller’s VDD was 5 V, which is much too high, meaning that the linear regulator is not working or a connection from the USB port to the microcontroller is overriding the linear regulator. Regardless, we suggest that one update the PCB to ensure each used port has a suitable reading.

One other main uncertainty relates to our sound processing. While we were able to extract SPL/decibel from the I2S data, we were unable to extract frequency from the I2S data. While we tried using ARM math functions, we found that the frequency value was either noise or infeasible. Thus, since we were unable to extract frequency, we could not transfer dBA nor frequency to the computer nor use dBA in our calculations. While we do not believe that missing frequency and dBA impacts our goal of warning users of sound hazards since we can always adjust the offset used to calculate dB and whether a sound is hazardous, missing frequency and dBA means that our current project does not measure sound in relation to the human ear. To remedy this issue, we recommend revisiting our data filtering method, as it could be the case that we are incorrectly filtering out useful data as noise. In addition, we also suggest changing the function we use to collect I2S data, as the function simply may not be able to capture frequency.

One last uncertainty relates to logging the transferred USB data. While we verified that our sound report works with mock data, for real use, our sound report should be reading from the raw sound readings from the microcontroller. The software we used, PuTTY, should allow us to save the sound readings from the microcontroller with the logging feature; however, we found that often, PuTTY gave an error and refused to log any of the readings for the sound report’s use. To resolve this uncertainty, we suggest that using another USB communication software analogous to PuTTY for USB data transfer to log the data for the sound report.

## 5.2 Accomplishments

While we did not manage to get our PCB working, we did manage to implement our project on a breadboard. To accomplish this feat, we needed to correctly solder our development board and mic module, wire our breadboard together, program the microcontroller, get the microcontroller software working, and get the computer software working.

To elaborate on the microcontroller software, as it was likely the most complex feature to get working, we correctly configured the software properly to match the ports of the development board, configured the USB and I2S configurations properly to send and receive sound readings and I2S data respectively, and processed the I2S data to give reasonable sound readings and sound hazard warnings.

In the end, we were able to build a breadboard analogous to our PCB that was able to accomplish our main objective of warning users of sound hazards through LEDs and a sound report made from raw sound readings.

## 5.3 Ethical considerations

The main ethical consideration to keep in mind relates to that of IEEE Code of Ethics I.1. [14]. Since our project right now has not been thoroughly tested yet to find the correct offsets or to find the correct dBA values to a high degree of precision, users need to understand that our project in its current state may not warn them accurately of all sound hazards. In addition, in the project’s current state, the tolerance for 90 dB is low at high frequencies, so the project may be overreporting the amount of sound hazards, which the user should keep in mind.

Regardless, the user should also understand for their health and safety, the project only reports sound hazards. If the project accurately reports a sound hazard, for their own safety, a user must take it upon themselves to use ear protection.

## 5.4 Future work

Last, we consider potential avenues for future work. Aside from the suggestions made in section 5.1 to get rid of uncertainty in our current project, we have recommendations for future work.

First, we believe one should expand on our project by making it more portable while retaining the same features. To that end, we believe that one should add a power subsystem including a battery and a battery charger so that the project is not dependent on a USB connection. Additionally, we recommend adding a display to show instantaneous SPL and daily sound exposure explicitly instead of just through the LED.

If one makes the system portable through the above recommendations, we also believe one could go further by making the system wearable either in-ear or somewhere else on the body. If one made the system wearable, alerts would be much more obvious and thus make warning the user much easier and more effective.

We could see one developing this project further through partnership with the CDC and/or other sound danger regulatory bodies. We believe this device, especially in a portable form, could be a cheaper and more functional alternative to their current SPL and sound dosage measurement instruments.

# References

[1] J. J. Eggermont, “Effects of long-term non-traumatic noise exposure on the adult central auditory system. hearing problems without hearing loss,” *Hearing Research*, vol. 352, pp. 12–22, 2017.

[2] V. P. of A. E. and M. C. at C. U. I. D. Jeff Smoot, “A comparison of digital PDM and I²S interfaces in MEMS microphones,” *Digi*, 11-Jul-2022. [Online]. Available: https://www.digikey.com/en/articles/a-comparison-of-digital-pdm-and-i2s-interfaces-in-mems-microphones. [Accessed: 25-Nov-2022].

[3] “The audible spectrum - neuroscience - NCBI bookshelf.” [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK10924/. [Accessed: 25-Nov-2022].

[4] “STM32F401 Mini Dev Board, STM32F401CCU6,” *ElectroDragon*, 28-Jul-2019. [Online]. Available: https://www.electrodragon.com/product/stm32f401-mini-dev-board-stm32f401ccu6/. [Accessed: 27-Nov-2022].

[5] “INMP441 omnidirectional microphone module I2S interface ... - amazon.com.” [Online]. Available: https://www.amazon.com/INMP441-Omnidirectional-Microphone-Interface-Precision/dp/B07QFBVY84. [Accessed: 28-Nov-2022].

[6] “Department of Labor Logo United Statesdepartment of Labor,” *1910.95 App A - Noise Exposure Computation | Occupational Safety and Health Administration*. [Online]. Available: https://www.osha.gov/laws-regs/regulations/standardnumber/1910/1910.95AppA. [Accessed: 27-Nov-2022].

[7] “DMM-4026-B-I2S-R: Digi-key electronics,” *Digi*. [Online]. Available: https://www.digikey.com/en/products/detail/pui-audio-inc/DMM-4026-B-I2S-R/11587483?s=N4IgjCBcoCwdIDGUBmBDANgZwKYBoQB7KAbRAGZyAOKgdgCYQBdAgBwBcoQBldgJwCWAOwDmIAL4F6YGAFYooZJHTZ8RUiHoAGAAQAJAF46Afju06A1oeZtOkEACU0onAFUhA9gHkUAWRxoWACufDgS4uJAA. [Accessed: 27-Nov-2022].

[8] “Home - stmicroelectronics.” [Online]. Available: https://www.st.com/content/ccc/resource/technical/document/datasheet/9e/50/b1/5a/5f/ae/4d/c1/DM00086815.pdf/files/DM00086815.pdf/jcr:content/translations/en.DM00086815.pdf. [Accessed: 28-Nov-2022].

[9] “Department of Labor Logo United Statesdepartment of Labor,” *1910.95 App A - Noise Exposure Computation | Occupational Safety and Health Administration*. [Online]. Available: https://www.osha.gov/laws-regs/regulations/standardnumber/1910/1910.95AppA. [Accessed: 27-Nov-2022].

[10] “INolux 5050 RGB led 4-pin with integrated IC in-PI554FCH.” [Online]. Available: http://www.inolux-corp.com/datasheet/SMDLED/Addressable%20LED/IN-PI554FCH.pdf. [Accessed: 28-Nov-2022].

[11] “TLV70030DCKT: Digi-Key Electronics,” *Digi*. [Online]. Available: https://www.digikey.com/en/products/detail/texas-instruments/TLV70030DCKT/2176436. [Accessed: 27-Nov-2022].

[12] “1001-001-01000: Digi-key electronics,” *Digi*. [Online]. Available: https://www.digikey.com/en/products/detail/cnc-tech/1001-001-01000/3064730. [Accessed: 27-Nov-2022].

[13] “Salary averages,” *Electrical & Computer Engineering | UIUC*. [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages. [Accessed: 28-Nov-2022].

[14] “IEEE code of Ethics,” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 28-Nov-2022].

# Appendix A Requirement and Verification Table

Note that the verification status below, as previously mentioned in section 3, is referring to the verification of the breadboard except in the sound report test. In the case of the PCB, we were unable to verify the requirements due to our microcontroller not connecting our computers due to a variety of previously mentioned reasons.

|  |  |  |
| --- | --- | --- |
| Table 4. Overall System Requirements and Verifications | |  |
| Requirement | Verification | Verification status  (Y or N) |
| **USB Data Transfer.**   1. The sound processing subsystem needs to be able to communicate with the user interface subsystem via USB with at most 5% transmission error ratio — incorrect transmissions/all transmission. | 1. Set the data transmitted by the microcontroller to be either a certain known value or a value with a certain format. 2. Verify that the value seen on the computer is correct or has the correct format 95% of time. | Y |
| **Sound Processing**.   1. Instantaneous SPL readings are correct within a ±3 dB margin of error. | 1. Feed curated sound data to the sound processing subsystem and to a commercially available SPL meter. 2. Either directly measure the output of the sound processing subsystem using an oscilloscope/logic analyzer or through data files on a local computer if the USB connection is functional. 3. Compare outputs, verify that the sound processing subsystem does not vary more than ±3 dB from the SPL meter. | Y |
| **Instantaneous SPL LED.**   1. The instantaneous SPL LED switches between a certain value to another within 2 s. | 1. Use either real or mock sound intensity data to send a valid safe dB reading and unsafe reading. 2. Verify that once the dB intensity changes ranges, the LEDs switch within 2 s. | Y |
| **Integrated SPL LED.**   1. Integrated SPL LED switches from off to on within 2 s only after receiving enough sound exposure. | 1. Use either real or mock sound intensity data to send decibel readings to the microcontroller. 2. Verify that once the decibel readings have crossed a certain threshold, the LED lights up. The verification fails if the LED lights up too quickly or turns off after lighting up. | Y |
| **Sound Report.**   1. The sound report software needs to be correct within 3 dB tolerance intensity wise (dB) for average sound level and within 10% for daily sound exposure. | 1. Send mock sound data simulating background noise in the human frequency range with a given decibel range. 2. Verify that the software conveys the correct overall average sound level and sound exposure within 1 dB and 10% tolerance respectively by checking against the raw data. | Y |

# Appendix B Schematics and Diagrams

Diagram, schematic

Description automatically generated

Figure . Circuit schematic.

A screenshot of a video game

Description automatically generated

Figure . PCB schematic. Note that the version number and date is incorrect, but we kept them due to the text being there in the final physical PCB. Likewise for the incorrect name for the USB port.

# Appendix C Software Flow and Outputs

Diagram

Description automatically generated

Figure . Flowchart of software on microcontroller. Note that this diagram does not include blocks corresponding to frequency, as we were unable to get frequency working by the final demo.

Text

Description automatically generated

Figure . Example of raw sound data input in dB from microcontroller to computer. Note that the top error message was one of the reasons we struggled to connect the raw data readings to the sound report.

Graphical user interface, text, application, website

Description automatically generated

Figure . Example of sound report interface where we found safe daily sound exposure. Data used was sound values of 89, 91, 80, and 70 dB with a time duration of 2 hours for each sample.

Graphical user interface, text, email

Description automatically generated

Figure . Example of sound report interface where we found unsafe daily sound exposure. Data used was sound values of 89, 91, 80, and 70 dB with a time duration of 4 hours for each sample.