Carbon Control

ECE445: Design Document

By:
Team 32

Mois Bourla (Bourla2)
Tanmay Goyal (Tanmayg2)
Vikram Belthur (belthur2)

TA: Daniel Ahn

Introduction **2**
1.1 Problem and Solution Overview 2
1.2 Visual Aid 3
1.3 High Level Requirements 4
Design **5**
2.1 Block Diagram 5
2.2 Physical Design 6
2.3 MCU Subsystem 7
2.4 Sensor Subsystem 8
2.4.1 CO2 Sensor 9
2.4.2 PIR Sensor 11
2.5 Alarm Subsystem 13
2.5.1 Amplifier, Speaker, LED 13
2.6 Power Subsystem 15
2.7 Server Subsystem 16
2.8 Tolerance Analysis 19
Requirements and Verification **21**
3.1 Power Subsystem 21
3.2 Sensor Subsystem 22
3.3 MCU 23
3.4 Alarm Subsystem 25
3.5 Server Subsystem 26
Cost Analysis and Project Scheduling **28**
4.1 Cost Analysis 28
4.1.1 Cost of Parts 28
4.1.2 Cost of Labor 29
4.2 Project Schedule 29
Ethics and Safety **31**
References **32**
Appendix A: Use Cases **34**
Appendix B: Circuit Schematic **40**

Introduction

1.1 Problem and Solution Overview

Air quality for indoor spaces is critically important, and universally needed. Whether it is an office, a school, or a hospital, buildings where large amounts of people congregate need to be safe to occupy. A key component of the safety of an indoor environment is ventilation. If a space is under ventilated the safety of its occupants is compromised. The Wisconsin Department of Health reports that indoor CO_2 concentrations between 1000-2000 parts per million (ppm) are associated with "complaints of drowsiness and poor air." While levels between 2,000–5,000 ppm indicate "stagnant, stale, stuffy air." Public health experts have recommended the use of CO_2 monitoring in "assessing ventilation … in an effort to reduce the risk of disease transmission" (Allen et al., 2020). Allen et al. recommend measuring the decay of CO_2 concentration to estimate how many times the air is replaced in a given room. This metric is known as the Air Changes $Hour^{-1}$ or ACH. These tests are conducted by artificially raising the indoor concentration and then removing the CO_2 source to monitor the rate of decay.

Commercially available CO_2 sensors can monitor CO_2 concentrations and alarms to alert occupants of high concentrations. However, these solutions do not provide functionality that is needed for large scale deployments across a variety of indoor environments. Many current solutions are not wirelessly accessible or use a bluetooth hub to enable a wireless interface. Devices without wireless interfaces cannot be used by facilities management personnel to monitor the ventilation quality of a room. While bluetooth enabled devices have a coverage area that is limited to within the bluetooth range of the hub. Moreover, current devices are not designed to monitor multiple zones within the same room. Lastly, commercially available devices cannot run automated ventilation tests to estimate the ACH of a space by taking advantage of changes in a room's occupancy.

Carbon Control is a WIFI enabled CO_2 and occupancy sensing node. It can not only trigger alarms based on CO_2 concentrations, but communicate its readings to a cloud based server for facilities personnel to track. Moreover, this device will support same room multi-zone deployments, where nodes in the same room will have synchronized alarms. Finally, the devices will leverage the occupancy sensor to detect recently emptied rooms. These rooms have high CO_2 concentrations from their previous occupants and have decaying concentrations. This allows the device to automatically estimate the ACH.

1.2 Visual Aid

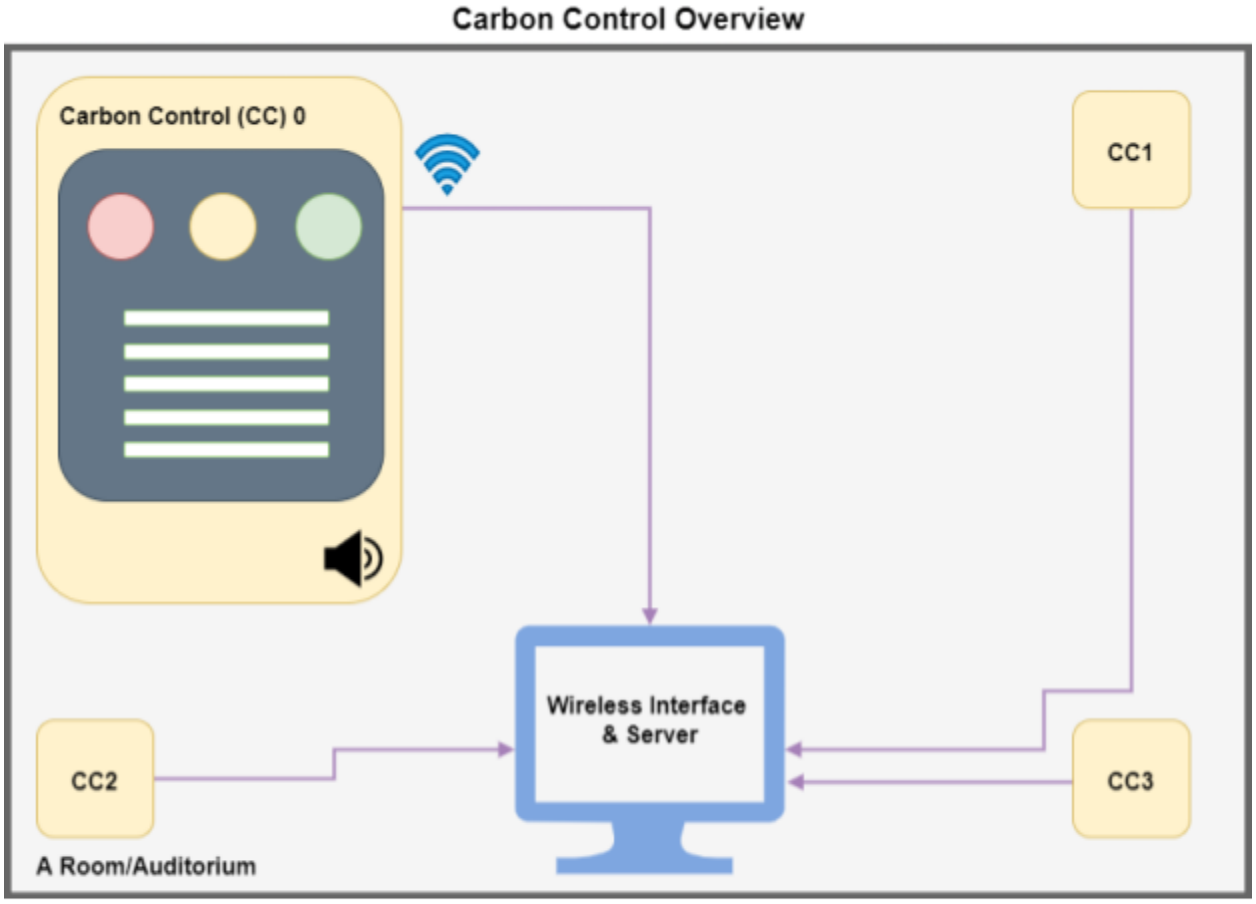

Figure 1: Visual Aid

1.3 High Level Requirements

1. The device must be capable of operating for at least eight hours on only battery power and also while charging.

2. The device will be able to monitor CO_2 concentrations and determine whether a room is occupied or not, and calculate the rate of CO_2 concentration decay in the given room to provide a well rounded view of safety.

3. The device should display for the users the relative safety of the room they are in through an LED capable of emitting different colors in the traffic light format. If a room is deemed unsafe for humans by having too high of a CO_2 concentration, occupants of that room shall be notified through an audible alarm.

4. The device shall connect to a 2.4 GHz WiFi network and it must send its sensor readings to a server so that users can monitor and analyze data through a web application. This application should be able to run on mobile devices.

Design

2.1 Block Diagram

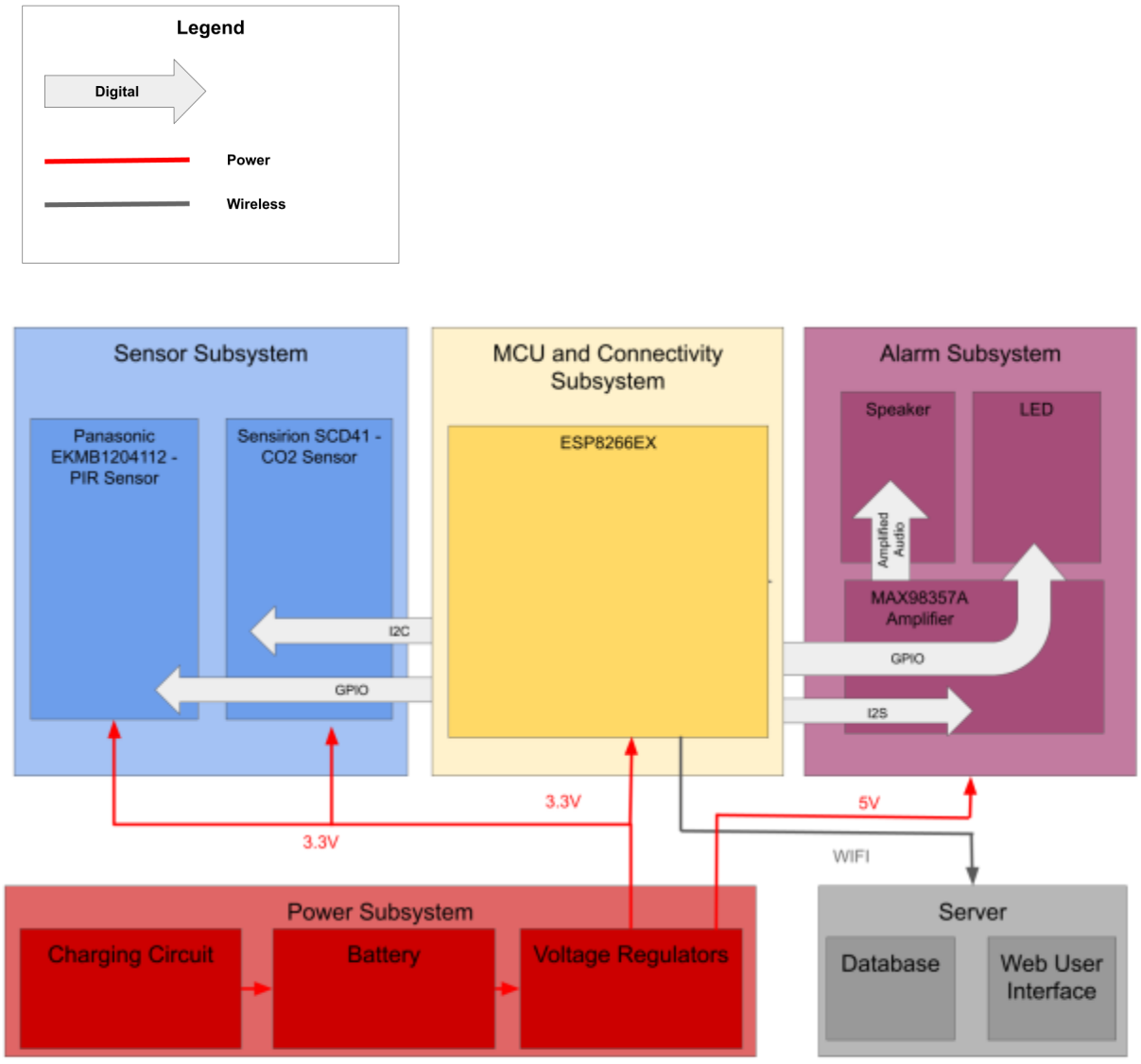

Figure 2: Block Diagram

2.2 Physical Design

The physical design of the project is shown above in Figure #, and it shows that the project will be a device that contains a PCB housed inside a rectangular enclosure. The front side of the project depicts the LED and air-vents, while the back side shows that there will be a battery pack. The USB symbol on the back side of the divide denotes that the device will have a USB charging port. Inside the enclosure there will be a PCB and a speaker. The PCB will contain the LED, a CO2 sensor, an Amplifier, and a PIR sensor, among other things. The PCB will be connected to a speaker. According to Espressif, the manufacturer of the ESP32, it is best to keep the MCU towards the edge of the PCB to reduce interference in the PCB antenna. It is also recommended to have the PCB antenna off the board for this same reason.

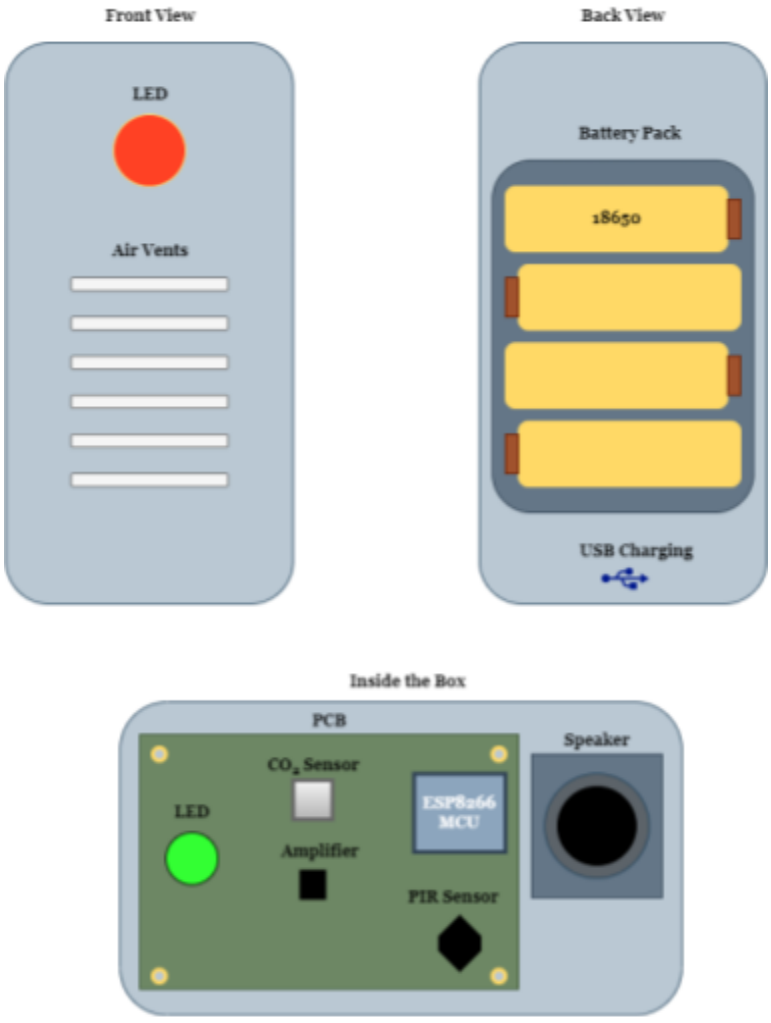

Figure 3: Physical Design

2.3 MCU Subsystem

A node's microcontroller must perform simple arithmetic computations as well as manage the nodes IO and connectivity needs. The primary criteria for the MCU selection was whether the MCU would have enough IO pins to support communication with both the sensor and alarm subsystems. Moreover, the MCU was selected to have wireless capabilities to be able to transmit and receive data from the server subsystem. Our proposal had originally detailed an STM32 paired with an ESP32-WROVER-E to provide enough IO capabilities and WIFI connectivity. This design was ultimately not pursued because it is duplicative and expensive. After considering what IO was necessary for the project, it was decided that the ESP32-WROVER-E had sufficient IO to connect to the other subsystems. It was also discovered that if the UART bus was used for the interconnect between the two MCUs, it would make in-board programming rather difficult.

The ESP32-WROVER-E has sufficient IO to maintain an I2C bus for the CO_2 sensor, an I2S bus for the sound signal of the alarm system, and two GPIO pins, one for the PIR output and one to trigger the LED. Additionally, the ACH computation and storage of historical data is now handled by the server subsystem, obviating the need for significant compute and storage resources.

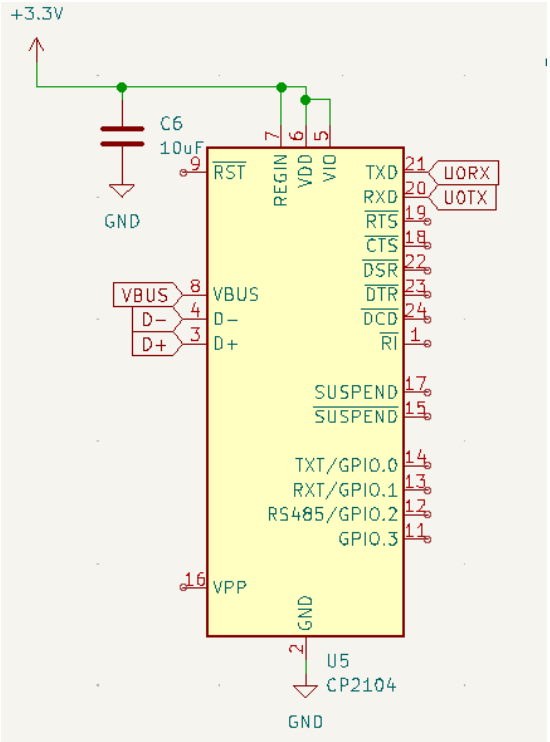

Figure 4: USB to Serial Converter for MCU

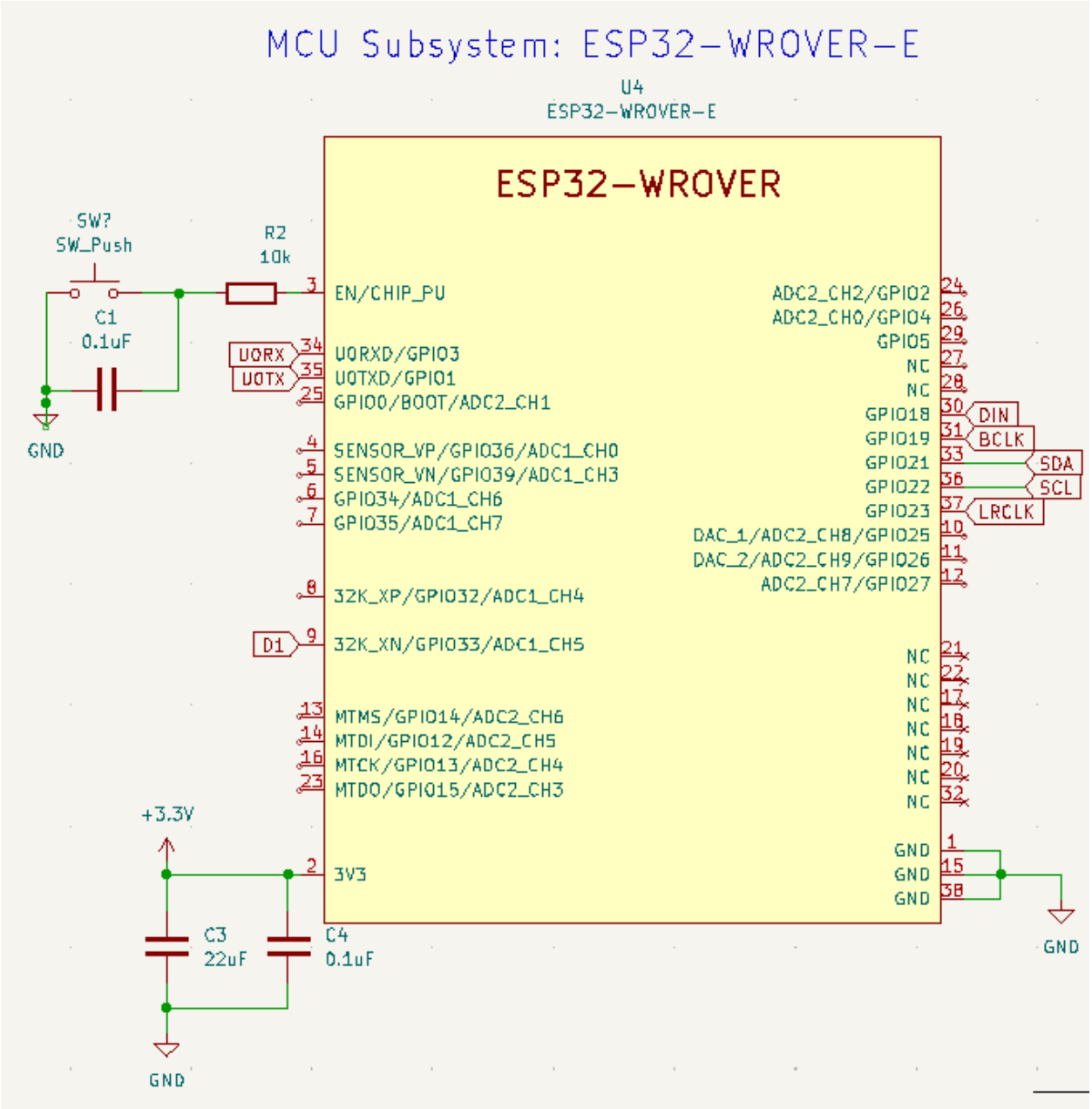

Figure 5 - MCU Subsystem Schematic

2.4 Sensor Subsystem

Our sensor subsystem contains two main components, a CO_2 air quality sensor and a PIR occupancy sensor. The CO_2 sensor will be used to keep track of CO_2

concentrations, measured in ppm, in a given space. Since we want to track the decay rate of CO_2 concentration when a room is unoccupied we have included the PIR sensor. The PIR sensor is used to detect motion, and it can tell us whether or not a room is empty. As mentioned in section 2.3, the ESP32-WROVER-E MCU is capable of I2C communication with various devices. It is also equipped with many GPIO pins. The CO_2 sensor will be connected to the I2C, while the PIR sensor is connected to one of the MCU's GPIO pins. Data from the sensor subsystem is used to activate the Alarm Subsystem and will be displayed to the user via the Server Subsystem.

2.4.1 CO_2 Sensor

Our device needs to be capable of monitoring CO_2 levels in an indoor space. To do this, we need a sensor which is accurate, has a low response time, and can detect CO_2 at a high enough concentration. CO_2 thresholding is important to our device, since we need to change LED colors and sound an alarm at specific CO_2 levels. Due to this sensitivity, our measurements must be accurate to at least $\pm$150 ppm. Because concentration is measured over a long time (an entire day, for example), we need a sensor that can produce updated values every few minutes. We also need a sensor capable of detecting values of up to 5000 ppm. This level is important because it is the OSHA mandated CO_2 exposure limit for indoor spaces.

For this project we will use the Senserion SCD41-D-R2 as our CO_2 sensor. This part was chosen in part due to its sensing range, accuracy, and sufficient response time. This design choice was also the result of a tradeoff between technical specifications and price. There are commercially CO_2 sensors that can sense more accurately and for a greater range than this sensor. However, there is a steep increase in price for these sensors compared to the SCD41-D-R2, which is already somewhat more expensive than sensors with less accuracy or a longer response time. Ultimately, this sensor offered our project the most features at a price point that would still allow our device to be affordable.

This sensor can detect CO_2 concentrations of upto 5000 ppm with an accuracy of around $\pm$40 ppm + 5% of the sensor readings. The response time of the sensor was found to be 60 seconds. As indicated by the manufacturer, the SCD41's average current use is typically 15 mA, and for low power periodic measurements is between 3.2-3.5 mA. As such, this sensor is not anticipated to heavily consume power, which is beneficial to the overall battery life (of at least 8 hours) for our device. As mentioned before, this sensor will communicate with the MCU through the I2C bus, as shown in Figure 6.

This sensor must be properly calibrated to ensure accurate CO_2 concentration values are being captured. This sensor operates based on a background concentration

reference value of 400 ppm, and its ppm readings are all relative to this value. The sensor needs to be recalibrated to this value over time. Over time, there is an accuracy drift, meaning the background CO_2 level will deviate from 400 ppm. This deviation will result in incorrect readings for the sensor. There are two ways to calibrate this sensor, automatic_self_calibration and forced_calibration, and these are programmed to the sensor via I2C. Automatic Self Calibration should be enabled to minimize accuracy drift. Forced Calibration should be used to reset the sensor's reference of 400 ppm. In order to maintain accuracy the sensor must be exposed to CO_2 concentrations of at least 400 ppm once per week. This concentration will be derived from a room that has been unoccupied for a long period of time.

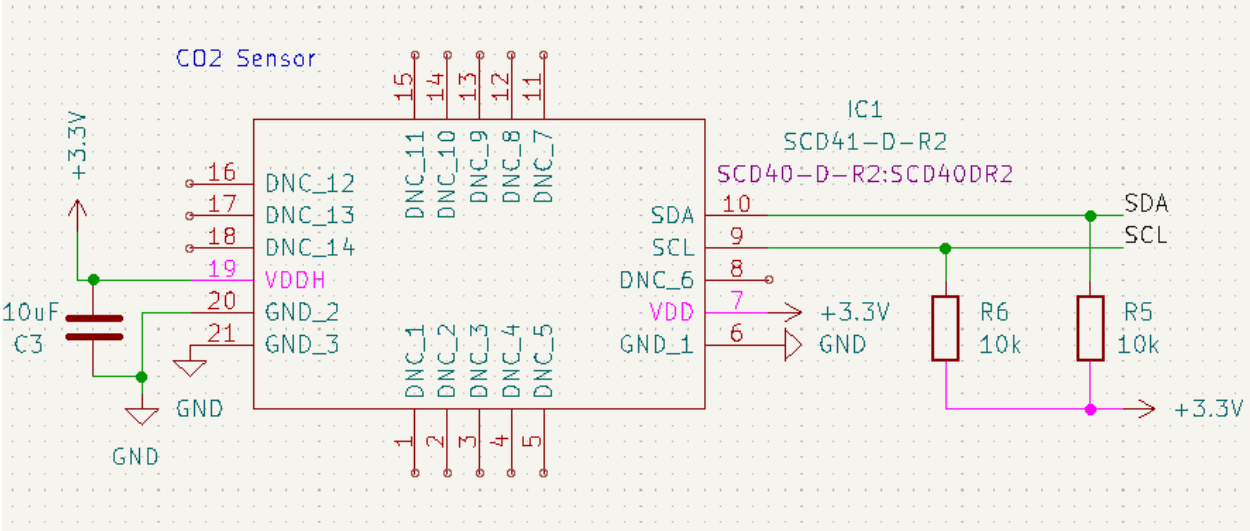

Figure 6 - Sensirion SCD-41-D and Connection to MCU via I2C

2.4.2 PIR Sensor

Our device must be capable of performing a ventilation test of a room by calculating the decay rate of CO_2 concentration. This measure depends on the time-constant of the decay. For these tests to take place in a room, it must be empty. The PIR sensor is important here because it works as a motion detector. If a room is empty, there is no motion within it; if that same room is occupied, there will be some motion detected by the PIR sensor. When selecting a sensor we aim to satisfy three criteria. First, the sensor must have a large enough detection range to encompass a room. Second, the sensor should be of a wall-mounted variety since the device will be so. Finally, the PIR sensor should be of a small form factor with low power consumption.

We ultimately chose the EKMB1204112 PIR sensor manufactured by Panasonic. In terms of cost effectiveness, this sensor is more expensive than other commercially available PIR sensors, but the price is compensated for by its capabilities. This sensor is specifically of the wall mounted type and will have a detection distance of up to 12m. A diagram depicting the range and detection of the PIR sensor is shown.

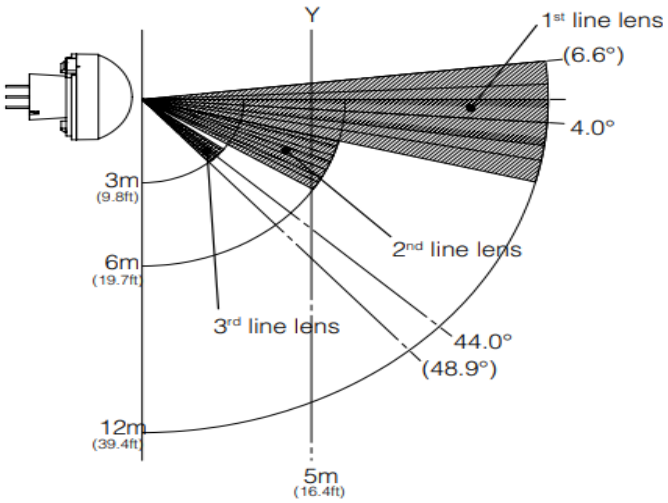

Figure 7 - Sensing Range of Panasonic EKMB1204112
Sourced from manufacturer's data sheet.

From the standpoint of current and power consumption, the PIR sensor uses 2 uA in standby and 100 uA when actively sensing. This is a miniscule current usage compared to other sensors and subsystems, and this sensor will not strain the overall power budget in any meaningful way. As shown in Figure 8, there is one output to this sensor which is connected to a GPIO pin on the MCU. This output will pulse high when motion is detected by the sensor, and will be low at all other times.

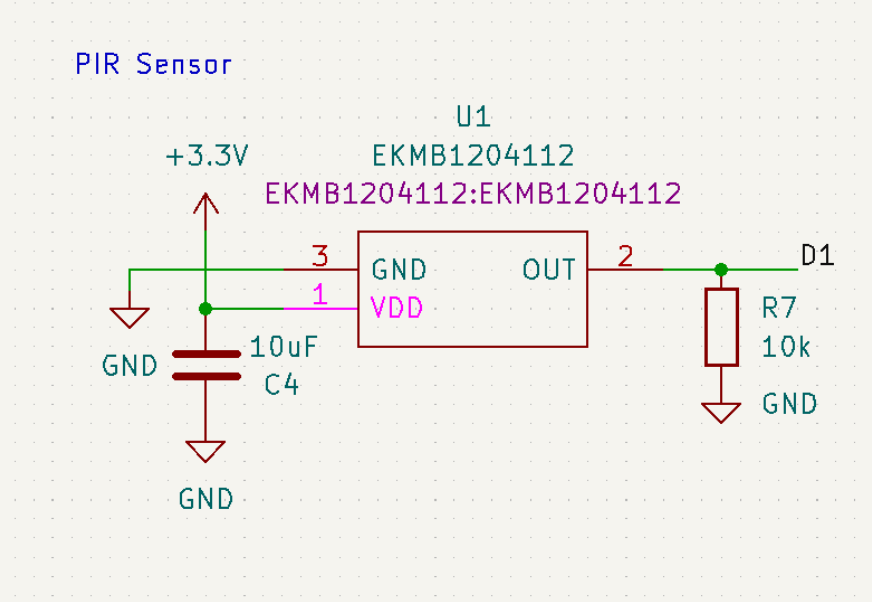

Figure 8 - Panasonic EKMB1204112 to MCU via GPIO - "D1"

2.5 Alarm Subsystem

The alarm subsystem serves the purpose of informing the occupants of a room if they are safe based on the concentration of CO_2 in the air. Our user interface will be both visual and auditory: an LED which changes color based CO_2 concentration, and an alarm will sound upon egregious levels of the gas. To accomplish these objectives we need bright LEDs and a loud sound, so that no notification is missed. In order to increase the amplitude of the sound, thereby making it louder, an amplifier is needed. The amplifier will connect to both the MCU and the speakers. It will use the I2S protocol to transmit information. There will be an LED as part of this system and it will be connected to the MCU as well. This subsystem will operate on 5V rather than 3.3 V, as the sensor subsystem, because the LEDs require 5V to appear at their brightest.

2.5.1 Amplifier, Speaker, LED

In order to play the sound and have it be audible, the speaker needs to receive enough power. Moreover, since our MCU is not equipped with an on-board DAC, one must be integrated onto our board. For the project, we opted to use the MAX3687A audio amplifier by Maxim. The amplifier integrates a digital audio interface, among other features, and a DAC on-chip. This is ultimately beneficial to the simplicity of the project. The amplifier is also relatively inexpensive, which is advantageous to the project from a budget standpoint. The operation of The MAX3687A is shown by the manufacturer in a block diagram below in Figure 9.

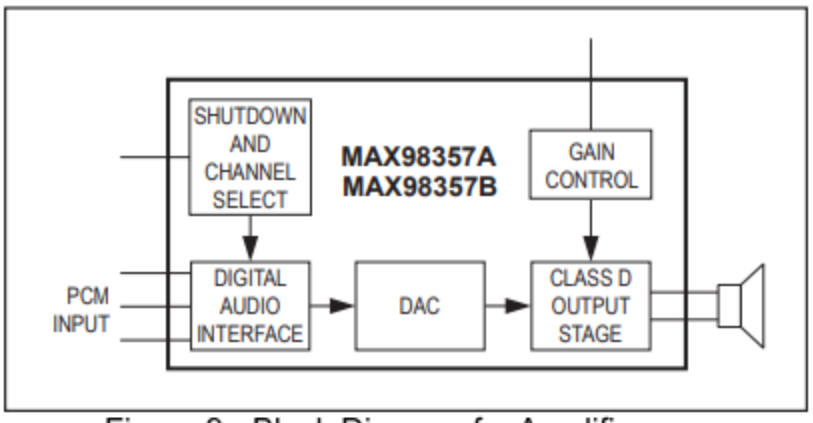

Figure 9 - Block Diagram for Amplifier

I2S is a three wire bus as indicated below in Figure 10. This audio will be played on a standard speaker and it will sound the alarm tone for a short time (15 seconds +/- 1 second). The LEDs will have three levels, red, yellow, and green corresponding to modes of operation for the device. The green and yellow LEDs represent low and high CO_2 concentrations respectively, and they will be colored (yellow and green) in the default operation mode. There are two alarm modes for the device; global and local. A local alarm has been triggered by the device because of a concentration of CO_2. A global alarm has been triggered by another device in the room. The red LED will be illuminated when the device is in the local alarm mode. If a device is in the global alarm mode, the LED will be off, and only the speaker will sound. The LED we chose for this project was the programmable Adafruit Flora because it is an RGB LED.

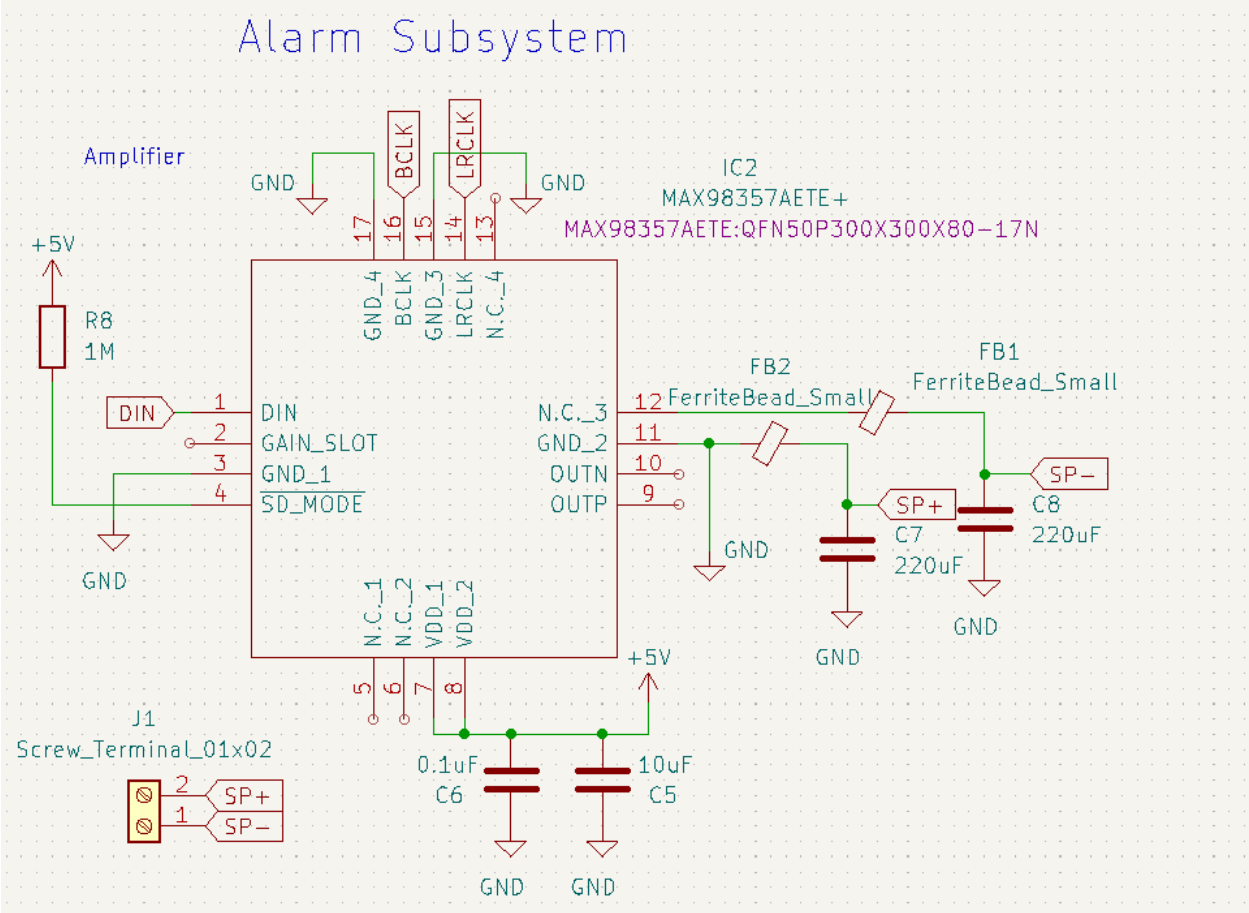

Figure 10 - Block Diagram for Amplifier

2.6 Power Subsystem

Our device must be able function for at least eight hours on battery power, delivering a stable voltage to the other subsystems of the project. The estimated power consumption of this device is approximately 5500 Wh of energy and possesses 1270 mAh of charge capacity. So that our device stays powered on for a sufficient amount of time, we have opted to use a 3.7V rechargeable lithium-ion battery. There are two aspects to this subsystem: supplying a constant voltage to the other subsystems, and charging the batteries via USB, once they are drained.

This device requires two separate voltage rails. The MCU and Sensor subsystems will need 3.3V and the alarm subsystem will need 5V delivered to it. We have selected the LD1117V33 as our 3.3v regulator. It is manufactured by STMicroelectronics, It is shown in Figure 11. The 5v rail is supplied by the TPS613222 boost converter. The device will be operable for up to eight hours without the need to be charged, but in the event that the battery is discharged, the device will need to be charged via USB. We will use a micro USB type-B connection to charge the 16850 battery. The system also has

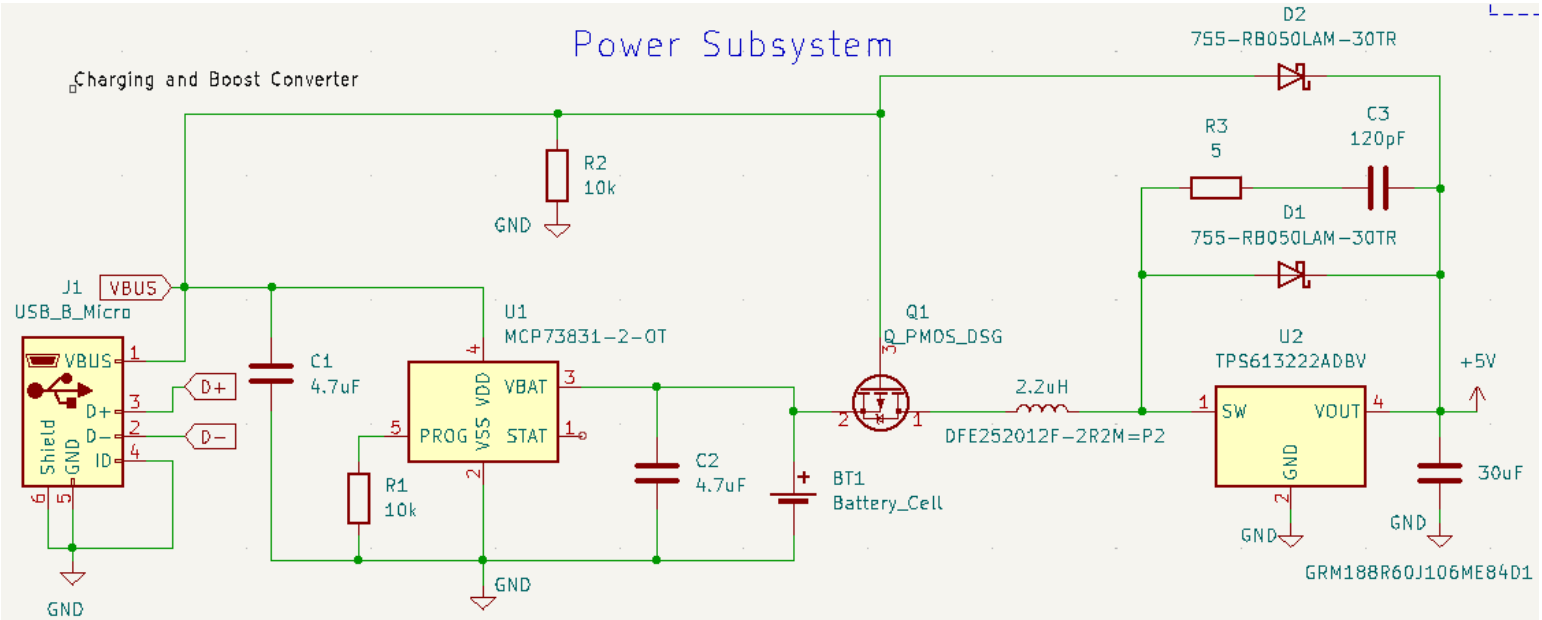

Figure 11 - Schematic for Power Subsystem

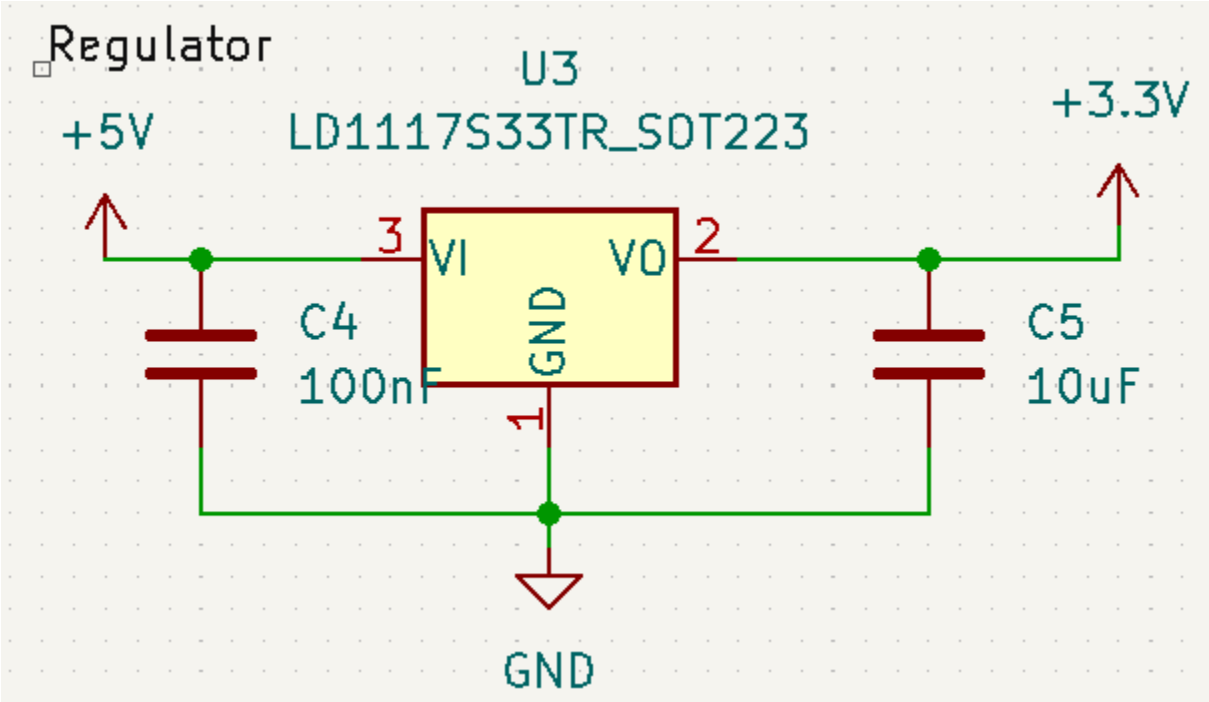

Figure 12 - Voltage Regulator Schematic

The charging system contains the MCP73831 USB charging IC, which ensures the 3.7V battery is properly charged. In order to provide a stable 5V rail, the battery voltage will be used as the input to a boost converter, the TPS613222ADBV, which will have 5V as its output. This will be used to then supply voltage to the Alarm Subsystem, particularly to ensure the LEDs are at their full brightness. The schematic for the USB charging and boost converter to 5V is shown in Figure 10.

2.7 Server Subsystem

In order to allow for remote monitoring of our deployed devices, we will maintain a web server that can be accessed by users to check both historical and current concentrations. The user will be able to view detailed information about any of their deployed devices. This server will expose a graphical user interface that will feature both a plot of historical co_2 concentrations, as well as the current status of the selected room. Room occupancy will also be reported.

The server will also permit the user to create zones within a given room in a multi-zone deployment. When the user tags nodes as being in the same room, our system will collate the data collected by the sensors into one view. Due to our data

being displayed to the user on a per-room basis, it is important to report the data in the context of it's room. The tagging process also enables us to synchronize alarms triggered in one room. If one device exceeds the preset CO_2 threshold, it will flash its red led and sound an alarm tone through its speaker. It will also send a POST request to an endpoint of our server, which will in turn, trigger the alarm tone in the other devices that share the same room. The alarm threshold will be modifiable through the web user interface.

Our cloud hosted server will receive and store data from sensor nodes that are deployed in the field. Our nodes will send POST requests to a server endpoint. The message payload will include the ten most recent data points from our sensor system. Each data point consists of the average CO_2 concentration over the proceeding minute as well as the occupancy as reported by the PIR sensor. The server will be a Flask server written in python. Flask is a web framework with support for a powerful templating engine, Jinja. This allows us to create HTML templates for our web pages that are populated at runtime with appropriate values. The Flask app will also include an SQLite database from a python package, SQLalchemy. This database will store our sensor readings.

The server will be hosted by Heroku, a commercially available cloud platform service provider. Heroku allows us to instantiate a dyno or host for our server. The dyno also provides us with significant compute capabilities, well in excess of the rather limited needs of our application. The storage capacity permitted by Heroku varies according to the tier of service selected. Our server will receive an array of 10 float values every 10 minutes while our device is in operation. We aim to store at least one week of historical data for our device. Assuming 8 hours of operation per day, we expect 3360 data samples of CO_2 concentration and occupancy. In SQLite, float values are stored within the Real storage class and consist of 8 bytes. Integers are variable in size depending on the value being stored, but for the purpose of representing a boolean value, we will use a single byte integer. This requires approximately 30 KB of storage. Additional information such as timestamps and other metadata may also need to be stored.

The server subsystem also calculates the air exchange rate of a space using the data collected by the sensor. There are a variety of formulas for estimating the air exchange rate. $ACH = \frac{-1 * ln(\frac{C_1}{C_0})}{t_1 - t_0}$ where $c_1$is the initial CO_2 concentration, $c_0$is the final CO_2 concentration, and $t_1 - t_0$ is the time taken for the decay to occur, provided by Batterman.

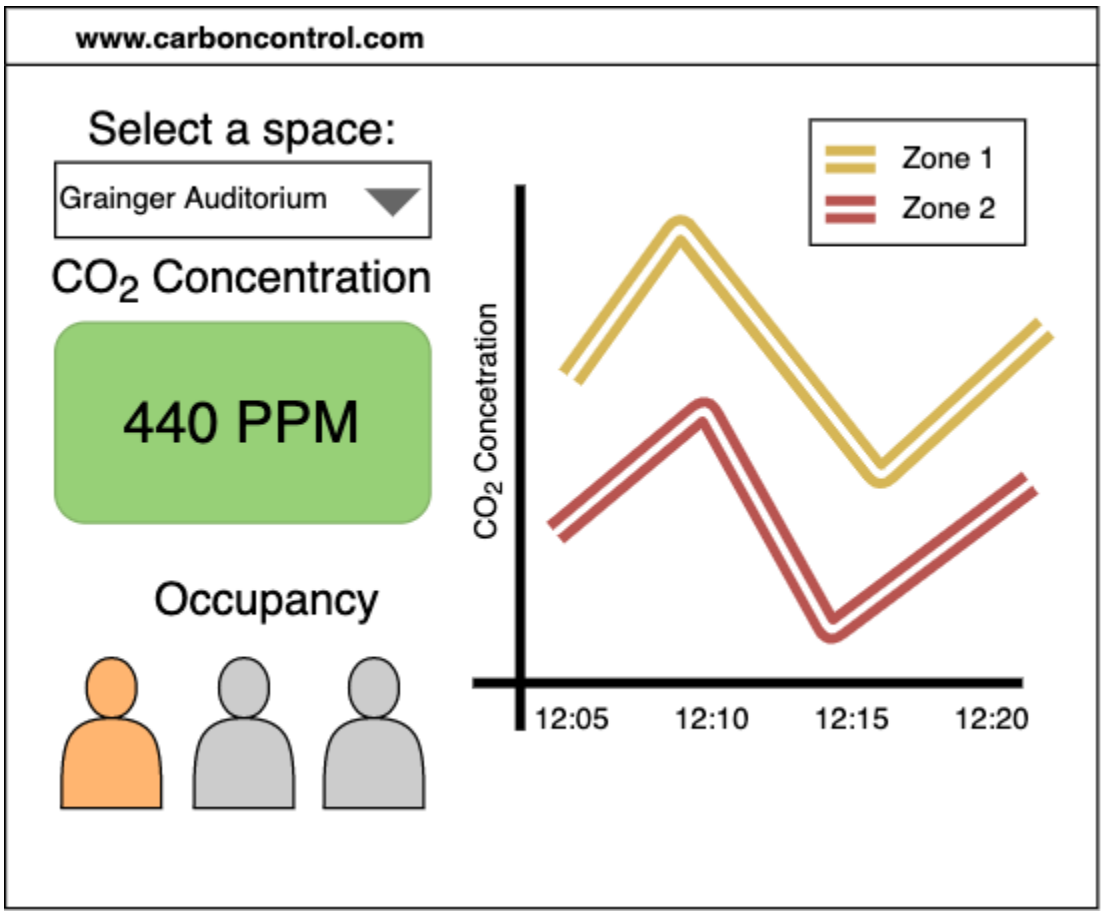

Figure 13 - Web App Concept

2.8 Tolerance Analysis

Our project is primarily built on the sensitivity of our sensing package. The PIR sensor enables us to automate the testing of a ventilation system, by sensing when a room does not have occupants. The primary specification is the quoted range of the sensor, which is stated to be at least 3m by the manufacturer. While this specification is important to our product's overall performance, it is not critical to our product's viability. The sensor's specification translates to how large of a room or zone our device can effectively monitor. While a bigger coverage area is commercially attractive, it is not necessary for our devices core functionality.

The CO_2 sensor's accuracy and responsiveness is far more important to the performance of our device. The accuracy is only important for triggering the alarm at the correct threshold value, which are themselves rather arbitrary. While the accuracy of the sensor is directly proportional to the accuracy of our device, we do not require a low absolute error, rather we require that changes in the measurement be proportional to changes in the true concentration for our device to be effective. Given that the absolute value is strongly affected by calibration, usually performed by adding a constant to the reading so an outdoor sample registers at ~400ppm, we focus on accurate relative changes. If our sensor error on any given sample is 50ppm, and we assume that a room's concentration rises from 500ppm to 1500ppm, the error constitutes only 5% of our measured change. This is insignificant for the purposes of triggering an alarm.

The sensor's responsiveness is very important to the computation of the air exchange rate, which depends on the rate of decay of the concentration. If the decay time is incorrectly measured, then the error on the ACH is rather large. Figure XX shows three separate scenarios where the decay in CO_2 is being measured for 10, 15, and 20 minutes respectively. The ACH varies inversely with the decay time, but even small errors can cause significant differences in estimated ACH. Increasing the length of the test does mitigate the impact of a given timing error. However, we would like a response time of two minutes or less so that the test can generate accurate (within 10% of actuals) results in under 15 minutes.

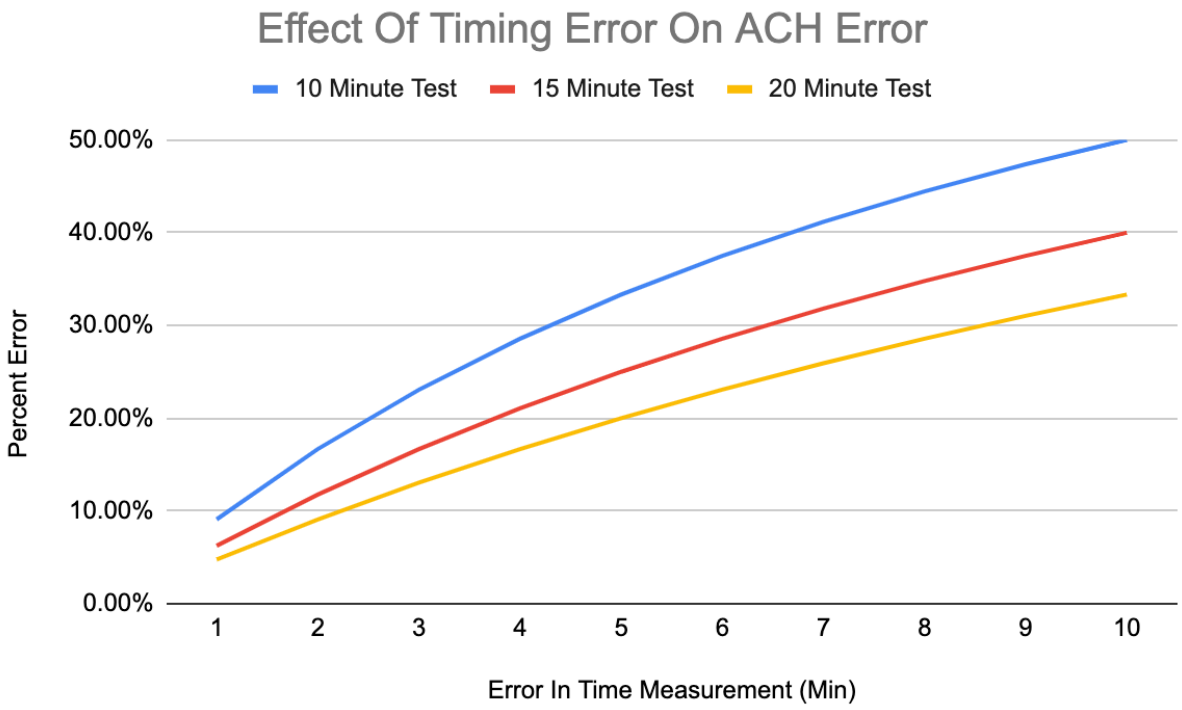

Figure 14 - Tolerance of ACH computation to Error in Time Measurement

Requirements and Verification

The following requirements and associated verification methods were derived from the associated use cases of our device. They are reported at the subsystem level.

3.1 Power Subsystem

Requirements	Verification
1. The device will maintain a low voltage power rail (3.3v +/- 0.3v) and a high voltage power rail (5v +/- 0.3v), when powered by an input voltage between 3.5v to 4.2v.	1A. Connect the battery input terminals to a lab based power supply. Connect a multimeter to the output of the 3.3v supply rail. 1B. Set the output voltage of the power supply to 6.90v +/- 0.05v and turn on the output. 1C. Verify that the regulator output voltage is between 3v and 3.6v. Increase the power supply voltage by 0.1v increments until output voltage is 7.50v +/- 0.05v, confirming that the rail voltage is between 3v and 3.6v at each increment. 1D. Repeat the procedure for the 5v supply rail, verifying that the rail voltage is between 4.7v and 5.3v.
2. The device will power on its power status LED red while the battery is charging, until the battery reaches its maximum voltage, 4.1v +/- 0.15v.	2A. Attach multimeter probes to the terminals of a battery with voltage <3.95v. Initiate charging by plugging in usb power. 2B. Verify that the device's status LED is red while the device's terminal voltage is below 4.1v +/- 0.15v. For this test to fail, the battery would need to exceed 4.25v and the LED continues to be red, alternatively, it would need to turn off the red LED while the battery voltage is below 3.95v.
3. The device's green power LED will be	3A. Attach multimeter probes to the

turned on after the battery is finished charging, i.e. battery voltage is 4.1v +/- 0.15v.	terminals of a battery with voltage <3.95v and green power status LED is off. Initiate charging by plugging in usb power. 3B. Monitor the device until the green power LED is on. Ensure that the battery voltage is between 3.95v and 4.25v when this occurs.
4. The device can operate for 8 hours on a single battery charge.	4A. Power on a fully charged device and begin a stopwatch. 4B.Trigger a local alarm through breath or dry ice. 4C. After the alarm has been active for 2 minutes (either through speaker tone or LED indication), remove the alarm stimulant by ventilating the area with a fan. 4D. Monitor the device every 30 minutes to ensure that the device is still powered on until it is not powered on or eight hours have elapsed, whichever occurs first.

3.2 Sensor Subsystem

Requirements	Verification
1. The device must be able to detect a human body in motion (0.8 - 1.5 m/s), within two meters of the device.	1A. Configure the MCU to print “motion identified” to the command line if the PIR sensor is activated. 1B. Walk in an arc around the device such that the distance to the device is approximately two meters. The distance will be measured by measuring tape. 1C. Verify that motion was detected by inspecting the command line.
2. The CO_2 sensor will have a response time of two minutes or less. The response time is defined as a greater than 25% change in concentration.	2A. The device will be configured to report the CO_2 concentration every five seconds to the command line. 2B. A baseline concentration will be taken as an average concentration

	2C. A sample of dry ice will be brought in close proximity (10 - 30cm) to the device. The distance will be measured by a ruler. 2D. A stopwatch will be started when the sample is released to its final position. 2E The stopwatch will be stopped when the command line reports a deviation from the starting value of 25% or more. 2F. We will verify that the time taken is less than or equal to two minutes.

3.3 MCU

Requirements	Verification
1. The MCU shall monitor battery voltage to within 10% of the true value.	1A. Use a multimeter (DMM) to monitor battery voltage ground truth. 1B. Configure MCU firmware to print estimated battery voltage to the serial bus. 1C. Verify that the difference is within 10% of the measured value on DMM.
2. The MCU GPIO outputs shall be able provide at least 10 mA of current, with output voltage staying within 20% of the open circuit voltage.	2A. Configure MCU to drive an output pin high. Measure the output voltage using a DMM. 2B. Select a resistor, such that, when used as a load, it sinks more than 10 mA of current. 2C. Verify that the output voltage is still within 20% of its initial output voltage under load.
3. The MCU shall send a POST request to a web server through WIFI.	3A. Configure the web server to have an endpoint that can accept POST requests, which displays incoming request payloads. 3B. Program the MCU to make a POST request with a known message up to 25 characters. Execute the program.

	3C. Verify the received data is what was transmitted on the server.
4. The MCU shall receive a POST request from the web server through Wifi.	4A. Configure the web server to send data through a POST request payload. 4B. Program the MCU to receive a POST request and print the payload to the command line. Execute the program. 4C. Verify the received data is what was transmitted from the server.
5. The MCU shall read the CO_2 sensor readings over an I2C bus.	5. Configure the MCU to read the sensor value and print the value read. Verify that a value between 400 and 5000ppm is read.
6. The MCU shall be able to transmit a sine wave tone with a frequency above 80Hz over I2S for at least 15 seconds.	6A. The MCU will be programmed to play a sine wave tone with a frequency above 80Hz over its I2S bus. 6B. An oscilloscope will have its probes on the positive and negative terminals of the speaker output. 6C. Verify that a sinusoidal wave is visible on the oscilloscope for at least 15 seconds.
7. The MCU shall toggle its GPIO Pins at a frequency of at 1Hz +/- 0.5Hz.	7A. Configure the MCU to toggle a GPIO pin at 1Hz. Verify that the frequency is within 0.5 to 1.5Hz by placing an oscilloscope probe on the output pin.
8. The MCU shall maintain a timer with millisecond ticks (+/- 0.5ms).	8A. The MCU will have a timer initiated with 1 millisecond increments and set to run for 10^5 iterations. As the program is triggered to run (within 2 seconds), a secondary timer will be initiated on a stopwatch. 8B. Verify that the elapsed time is within 50-150 seconds +/- 2 seconds. The additional increment accounts for synchronization errors.
9. The MCU shall classify the current CO_2 concentration into either low, medium, or	9A. The MCU will be set to monitor CO_2 concentration. The MCU will print current

alarm, based on user presets from the server.	concentration and current classification (low, medium, or alarm). 9B. The server will be updated with new thresholds for each class of reading, such that the previous readings would be classified in different categories. The new thresholds are transmitted to the MCU. 9C. Verify that the CO_2 readings have been reclassified. Note: if the CO_2 concentration changes such that the new thresholds would no longer cause a different classification, the test is invalid.

3.4 Alarm Subsystem

Requirements	Verification
1. The alarm system shall be able to emit Red, Yellow, and Green light from it's LED.	1A. The MCU will be programmed to turn the LED to Red, Yellow, and Green in sequence. Each color should be held for 20 seconds. Verify that all colors are visible and solid in appearance.
2. The alarm system shall be able to emit a tone from its speaker for at least 15 seconds.	2A. The MCU will be programmed to send a tone to the amplifier. 2B. The speaker will be plugged into the amplifier output. Verify that a tone can be heard for at least 15 seconds.
3. The amplifier shall provide at least 2 watts of power to a 4 ohm load +/- 0.5 ohm.	3A. Place a 4 ohm resistor rated for at least 4 watts across the amplifier output. 3B. Confirm the resistance of the resistor using a DMM to be within 3.5-4.5 ohms. 3C. Drive the amplifier to produce maximum power. 3D. Confirm that $\frac{V^2}{R} \geq 2W$ using a DMM with probes across the resistor measuring voltage.

3.5 Server Subsystem

Requirements	Verification
1. The server must maintain an endpoint that the device can POST to that it is in a local alarm condition.	1A. Configure the server to receive POST requests and display the output. 1B. Transmit a local alarm condition from the MCU. 1C. Verify that the server displays the local alarm condition.
2. The server can make POST requests to the MCU with a data payload.	2A. Configure the MCU to have an endpoint that can receive POST requests. The request contents should be printed. 2B. Program the server to make a POST request with a data payload. 2C. Verify that the request is printed by the MCU to the serial port.
3. The server can store at least 30 KB of data in its database.	3A. Create a text file (.txt) with a 30 KB size or greater. 3B. Save the contents of the text file to an entry in the database. 3C. Retrieve the entry and verify that the entry is complete.
4. The server will be able to update the web interface within 60 seconds of receiving updated data.	4A. Configure the server to display the time of the last inbound request and current time difference. 4B. Run the Carbon Control device until it makes an outbound transmission. 4C. Verify that the difference is less than or equal to 60 seconds.
5. The server can update the alarm thresholds on the device.	5A. Use the server to set the alarm threshold to 10ppm on a device. 5B. Verify that the alarm is triggering.
6. When one device has a local alarm, the server will activate a global alarm in all devices that are tagged to be in the same room.	6A. Configure the server such that two devices are tagged in the same room. 6B. Trigger an alarm in one device with dry ice or breath. 6C. Ensure that the other device also triggers with only the speaker, but not the LED, active.

7. The server's web UI can display at least 20 minutes of historical CO_2 data from each device in a room.	7A. Configure two devices to be in the same room on the server. Power up the devices. 7B. Run the devices for 30 minutes to permit at least 20 minutes of data to be uploaded to the server. 7C. Verify that the server can display the recorded data on a plot.
8. The server will compute the ACH to within 30%.	8A. View the computed ACH on the server's web UI. 8B. The ACH will be manually computed using the historical data graph on the web UI. The $ACH = \frac{-1 * ln(\frac{C_1}{C_0})}{t_1 - t_0}$. The points used will correspond to the first point of noticeable decay and an arbitrary final point during the decay. 8C. The manually computed ACH will be compared.

Cost Analysis and Project Scheduling

4.1 Cost Analysis

To determine the cost of this project we need to sum the total cost of all the parts as well as the cost of our labor.

4.1.1 Cost of Parts

All prices in USD.

Part	Name	Price ($)
MCU [6]	ESP32-WROVER-E	3.90
CO2 Sensor [7]	SCD41-D-R2	52.18
PIR Sensor [8]	EKMB1204112	28.29
Amplifier [9]	MAX98357A	2.84
RGB LED [10]	Adafruit Flora	7.95
Boost Converter [11]	TPS613222ADBV	0.61
Capacitor * 15	GRM188R60J106ME84D1	30.15
Voltage Regulator [12]	LD1117V33	0.77
Resistors * 8	MCU0805MD7504BP500	4.60
USB Charging IC	MCP73831T-2ACI/OT	0.69
USB to Serial Bridge	CP2104-F03-GMR	2.66

Total Part Cost = $134.64

4.1.2 Cost of Labor

All three of the members of this group are in electrical engineering, and according to the latest data available to us from the ECE department (2019), the average salary was $79,714 USD/year. Assuming we work a standard 40 hrs/week in a 52 week year, with no overtime pay, our annual wage corresponds to 38.32 USD/hr. We anticipate working on thi project for 10 hours for the remaining 10 weeks of the semester. This is a total of 100 hours. Using the formula given to us:

$$38.32\,[\tfrac{USD}{hr}]\ \times\ 2.5\ \times 100\,[hrs]\ \ = 9580\,[\tfrac{USD}{person}]\ *\ 3\ =\ \$28,740$$

To get the total cost we sum the parts and the labor to get a cost of $28,750. This cost can potentially be offset by buying in bulk which would generate a net savings in parts.

Therefore the total cost is $28,874.64.

4.2 Project Schedule

Week	Mois	Tanmay	Vikram
2/20 - 2/26	Design MCU & server subsystem	Design power subsystem	Design sensor & alarm subsystem
2/27 - 3/05	Work on PCB layout	Start testing the sensors	Start microcontroller programming
3/06 - 3/12	Test parts for validation	Finalize PCB design	Order finalized PCB design
3/20 - 3/26	Assemble the box enclosure	Test the alarm subsystem	Test the power subsystem
3/27 - 4/02	Start soldering PCB	Start coding the server subsystem	Finish coding the MCU subsystem
4/03 - 4/09	Finish coding and testing the server	Start testing PCB	Assemble the project
4/10 - 4/16	Debug all issues	Start testing the system together	Optimize the final project
4/17 - 4/23	Start writing final paper	Finish working on the project model	Start making final presentation

4/24 - 4/30	Finish making final presentation	Do mock demo presentation	Finish writing final paper
5/01 - 5/07	Final Demo and Presentation	Final Demo and Presentation	Final Demo and Presentation

Ethics and Safety

There are some safety issues that may be encountered during the course of this project. First, working with atmospheric gasses like CO2 could include exposure to high levels of the gas for prolonged periods of time during the testing phase of this project. This could be prevented by having multiple people monitor and update the potentially exposed party. If the test involves using sources of carbon dioxide like dry ice proper safety equipment should be used to handle the chemical in a secure manner and in the proper location in the lab. Another safety concern would be the electrical hazard. We will be wearing protective equipment when utilizing the electrical test equipment, as needed, when we are present in the lab. Other safety issues may arise during the mechanical assembly for our project. Proper safety precautions will be followed and safety equipment worn while working with equipment capable of injury, particularly sharp objects. Another potential issue relating to safety could be in our device's alarm. We will isolate its testing to prevent confusion and unnecessary panic that could ensue if it were to be sounded in public.

In addition to the aforementioned safety concerns there are some ethical considerations to be made. Data privacy regarding the number of people in a room is important and will be safeguarded, so as to comply with the ACM's codes of privacy (1.6) and security (2.9). We will also ensure that this project complies with the ACM's policy on fairness and descrimination (1.4) by giving an unbiased reading, irrespective of anyone who may be occupying a room. To conform to the IEEE code of ethics (1.1), [13] this project should make improvements on the existing infrastructure, and results from the project will be handled fairly and openly so as to not harm anyone if its implementation is erroneous.

References

[1] G. Pei, D. Rim, S. Schiavon, and M. Vannucci, “Effect of sensor position on the performance of CO2-based demand controlled ventilation,” *Energy and Buildings*, 13-Aug-2019. [Online]. Available:https://www.sciencedirect.com/science/article/abs/pii/S0378778819305377. [Accessed: 22-Feb-2022].

[2] Wisconsin department of health services, “Carbon dioxide guidelines” [Online] Available: https://www.dhs.wisconsin.gov/chemical/carbondioxide.htm

[3]Joseph Allen , Jack Spengler, Emily Jones, Jose Cedeno- Laurent, “Guide to measuring ventilation rates in schools,” Harvard T.H Chan School of Public Health [Online] Available: https://schools.forhealth.org/wp-content/uploads/sites/19/2020/08/Harvard-Healthy-Buildings-program-How-to-assess-classroom-ventilation-08-28-2020.pdf [Accessed : August 2020]

[4] Mike B. Schell, Stephen C. Turner , R. Omar Shim “Application of CO2 based demand control ventilation using standard 62: Optimizing energy use and ventilation.” Air test technologies [Online].Available: https://www.airtesttechnologies.com/support/reference/paper1.pdf

[5] Minnesota department of health, “Carbon dioxide guidelines” [Online] Available: https://www.health.state.mn.us/communities/environment/air/toxins/co2.html

[6] “Microcontroller unit ESP32-WROVER-E datasheet”, Adafruit [Online] Available: espressif.com/sites/default/files/documentation/esp32-wrover-e_esp32-wrover-ie_datasheet_en.pdf

[7]“CO2 Sensor unit SCD41-D-R2 datasheet”, Sensirion.com [Online] Available: https://sensirion.com/media/documents/C4B87CE6/61652F80/Sensirion_CO2_Sensors_SCD4x_Datasheet.pdf

[8]”PIR Sensor EKMB1204112 datasheet”, media.digikey.com [Online], Available:https://media.digikey.com/pdf/Data%20Sheets/Panasonic%20Sensors%20PDFs/EKMB_MC_AMN2_3_Rev_Sep_2012.pdf

[9] “Amplifier MAX98357A datasheet”, Adafruit.com [Online] Availablehttps://cdn-shop.adafruit.com/product-files/3006/MAX98357A-MAX98357B.pdf

[10] “RGB LED ADAFRUIT FLORA”, Adafruit.com [Online] Available : https://www.adafruit.com/product/1260#technical-details

[11] “Boost converter TPS613222ADBV datasheet”, Texas instruments [Online]Available:https://www.ti.com/general/docs/suppproductinfo.tsp?distId=10&gotoUrl=http%253A%252F%252Fwww.ti.com%252Flit%252Fgpn%252Ftps61322

[12] “ Voltage regulator LD1117V33 datasheet”, STMicroelectronics. [Online]Available: https://www.mouser.com/datasheet/2/389/cd00000544-1795431.pdf

[13] “IEEE Code of Ethics” IEEE [Online] Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[14] United States Department of Agriculture, “Carbon dioxide guideline”. [Online]Available: https://www.fsis.usda.gov/sites/default/files/media-file/2020-08/Carbon-dioxide.pdf

Appendix A: Use Cases

The use cases enumerated below explain how our system is expected to function. They explain what sequence of operations the system is expected to execute. They are a specification of all functions our device is guaranteed to perform. There are also triggers that describe when the use case is executed. The use cases are divided into five classes of possible use cases.

A note on syntax:

1. Optional explanation
    a. Option A
    b. Option B

Should be read as either option A is true or option B is true.

Use Case Class 1: Power

Use Case 1: Battery Charging

Main Flow:

1. The CC is plugged into a wall adapter by a micro USB connector.
2. The CC side LED begins to shine the power status LED red, while the device is charging by USB-MicroB connection.
3. When the CC is fully charged, its power status LED will become solid green while it is plugged in. This ends the use case.

Use Case 2: Device Runs on Battery Power

Main Flow:

1. The device checks the battery voltage on power up.
    a. If there is a sufficient battery voltage present for device operation (>= 3v), continue to step 3.
    b. If the battery voltage is insufficient for device operation, battery voltage (<3v), continue to step 2.
2. The device will enter “a fail to power on state” and all LEDs will remain dark. Exit

3. The device will enter a “power on successful state” and all LEDs will flash on for 3 seconds.
4. The device monitors the battery voltage
    a. If the battery voltage is below the low battery condition (3v <= battery voltage < 3.4v), continue to step 5.
    b. If the battery voltage is at or above the low battery condition (>= 3v), continue to step 6.
5. The device will blink its power status LED red while battery voltage is sufficient for device operation (>=3v).
    a. If the battery voltage is sufficient for device operation, the power status LED will blink red. Continue to step 6.
    b. If the battery voltage is insufficient for device operation, continue to step 2.
6. The device will maintain a low voltage power rail (3.3v +/- 0.3v) and a high voltage power rail (5v +/- 0.3v). Continue to step 4.

Use Case Class 2 : Large Scale Deployments

Use Case 1: Upload Data to Cloud

Trigger: A new data point is available.

Main Flow:

1.
    a. If a local alarm condition is triggered by the device, upload a local notification to a preset server alarm endpoint. Continue to step 2.
    b. If a local alarm condition is not triggered by the device, continue to step 2.
2.
    a. When more than a user specified number of data points is collected by the device, the data is posted to a preset cloud server data endpoint. Continue to step 3.
    b. When less than a user specified number of data points is collected by the device, Continue to step 1.
3. The number of samples collected is reset to zero. Continue to step 1.

Use Case 2: Alarm Synchronization

Trigger: An alarm notification is received from the server.

Main Flow:

1.
    a. If a global alarm notification is received from the server and the device is already in a local alarm condition, continue to step 2.
    b. If a global alarm notification is received from the server and the device is not already in a local alarm condition, continue to step 3.
2. The device will perform a local alarm notification to the users. This ends the use case.
3. The device will perform a global alarm notification to the users. This ends the use case.

Use Case Class 3 : Alarm Notifications

Use Case 1: Local Alarm Notification

Trigger**:** A Local Alarm Notification is triggered by the device.

Main Flow:

1. Default LED operation is suspended.
2. The speaker will sound the alarm tone for a short time (15 seconds +/- 1 second).
3. The LED will flash red at a fast frequency (1Hz +/- 0.5Hz).
4.
    a. If the local alarm condition is still active, continue to step 2.
    b. If the local alarm condition is inactive, the alarm system will resume default operation. This ends the use case.

Use Case 2: Global Alarm Notification

Trigger**:** A Global Alarm Notification is triggered by the device.

Main Flow:

1. Default LED operation is suspended and LEDs turn off.
2. The speaker will sound the alarm tone for a short time (15 seconds +/- 1 second).
3. The LED will resume default operation. This ends the use case.

Use Case 3: Default Operation

Main Flow:

1.
    a. The green LED will be on if the system is in the low concentration mode. Continue to step 1.
    b. The yellow LED will be on if the system is in the high concentration mode. Continue to step 1.

Use Case Class 4: Data Collection and Analytics

Trigger: Device is on.

Use Case 1: Read data from sensors

Main Flow:

1. A timer is initiated with a 1 millisecond tick interval (+/- 0.5ms).
2. If a system-defined or greater duration between CO_2 measurements has elapsed, a new CO_2 measurement will be obtained.
3. If a system-defined or greater duration between occupancy measurements has elapsed, a new occupancy measurement will be obtained.
4.
    a. If more than or equal to a minute has elapsed between the previously averaged data point, a new averaged CO_2 value and the boolean value of estimated occupancy is computed.
    b. If less than a minute has elapsed between the previously averaged data point, continue to step 2.
5. The averaged CO_2 value and the estimated occupancy will be saved as a new data point.
6.
    a. The data point CO_2 value is compared to the high concentration threshold. If the value is below the threshold, the low concentration mode is set.
    b. The data point CO_2 value is compared to the high concentration threshold. If the value is above the threshold, but below the alarm concentration threshold, the high concentration mode is set.

c. The data point CO_2 value is compared to the alarm concentration threshold. If the value is above the threshold, a local alarm condition is set.

7. The device declares that a new data point is available. Continue to step 1.

Use Case 2: Ventilation Testing

Trigger: New data has been uploaded to the cloud.

Main Flow:

1. Verify that the length of historical data is greater than or equal to the user defined test duration.
    a. If the length of historical data is greater than or equal to the user defined test duration, continue to step 2.
    b. If the length of historical data is less than the user defined test duration, raise insufficient data error to the user. This ends the use case.
2.
    a. If the room is estimated to be unoccupied for the proceeding user defined test duration, continue to step 3.
    b. If the room is estimated to be occupied for any datapoint in the proceeding user defined test duration, this ends the use case.
3.
    a. If the CO_2 concentration across the test window has decayed by more than 25% from it's initial data point, continue to step 5.
    b. If the CO_2 concentration across the test window has not decayed by more than 25% from it's initial data point, continue to step 4.
4. Wait for a new data point to be available, continue to step 2.
5. Compute the air changes per hour (ACH).
6. Display ACH to the user on the web user interface. This ends the use case.

Use Case Class 5: Server Operations

Use Case 1: Updating user defined settings

Trigger**:** The user enters the update settings webpage.

Main Flow:

1. The user updates fields in web form corresponding to individual user-defined settings.
2. The user submits the form.
3. The server locally updates server settings.
4. The server transmits user-defined device presets or settings to devices.
5.
    a. The user is returned to the homepage of the server if the update is acknowledged by all devices. This ends the use case.
    b. The user is notified of the appropriate error with an error page. This ends the use case.

Use Case 2: Display analytics

Trigger**:**

1. The user enters the home page.
2. The selected room is changed.

Main Flow:

1. If no room is currently selected, the first room added to the system is the default room, otherwise the selected room is used.
2. Server will display the most recent concentration, occupancy, ACH estimate, as well as historical CO_2 levels by zone.
3. If any metric is not available it will inform the user it is unavailable through the web ui.

Appendix B: Circuit Schematic