Automated Sensor-Based Filtration System

ECE 445 Design Document

Project # 68

Prithvi Saravanan (prithvi3@illinois.edu)
Omar Koueider (oyk2@illinois.edu)
Karthik Talluri (talluri4@illinois.edu)

Professor: Arne Fliflet
TA: Selva Subramaniam

Contents

Introduction

1.1 Problem and Solution

As our environment continues to change for the worse with the presence of global warming and increased human consumption of resources like fossil fuels, the safety associated with breathing clean air is being threatened. In metropolitan areas across the world, there is an increase in the smog and toxic output, leading to increased respiratory problems. Currently, no building filtration systems adapt according to compounds present outside the building, like volatile organic compounds (VOCs), and, as a result, the implementation of a new, different filtration system becomes necessary. This ever-present problem will continue as the population of the world increases, and breathing clean air indoors is a fundamental right every individual should have.

The proposed solution to this vast and unending problem is a dynamic filtration system that adjusts according to the concentration of a specific outdoor particle. We have chosen to monitor CO2 and PM2.5 particles commonly found in dust. The goal is to keep the indoor particle concentrations despite any change in composition of the outside air. This will be done using a sensor subsystem, an ESP32 microcontroller, and an air blower for filtration testing. The electrochemical sensor system constantly monitors these factors and provides a reading that will activate the dynamic filtration subsystem to filter out particles in a more accurate manner. To keep the indoor concentration numbers constant, we must compare data from the outdoor particulate sensor system with one based inside the enclosure. Two separate electron chemical sensor systems will monitor outdoor and indoor particles, and a microcontroller will take the data from these sensors and determine what particles to filter out. The adaptation functionality of changing the directional flow of an external source of air will be implemented according to the results from the data acquisition subsystem and the responses of the microcontroller subsystem.

1.2 Visual Aid

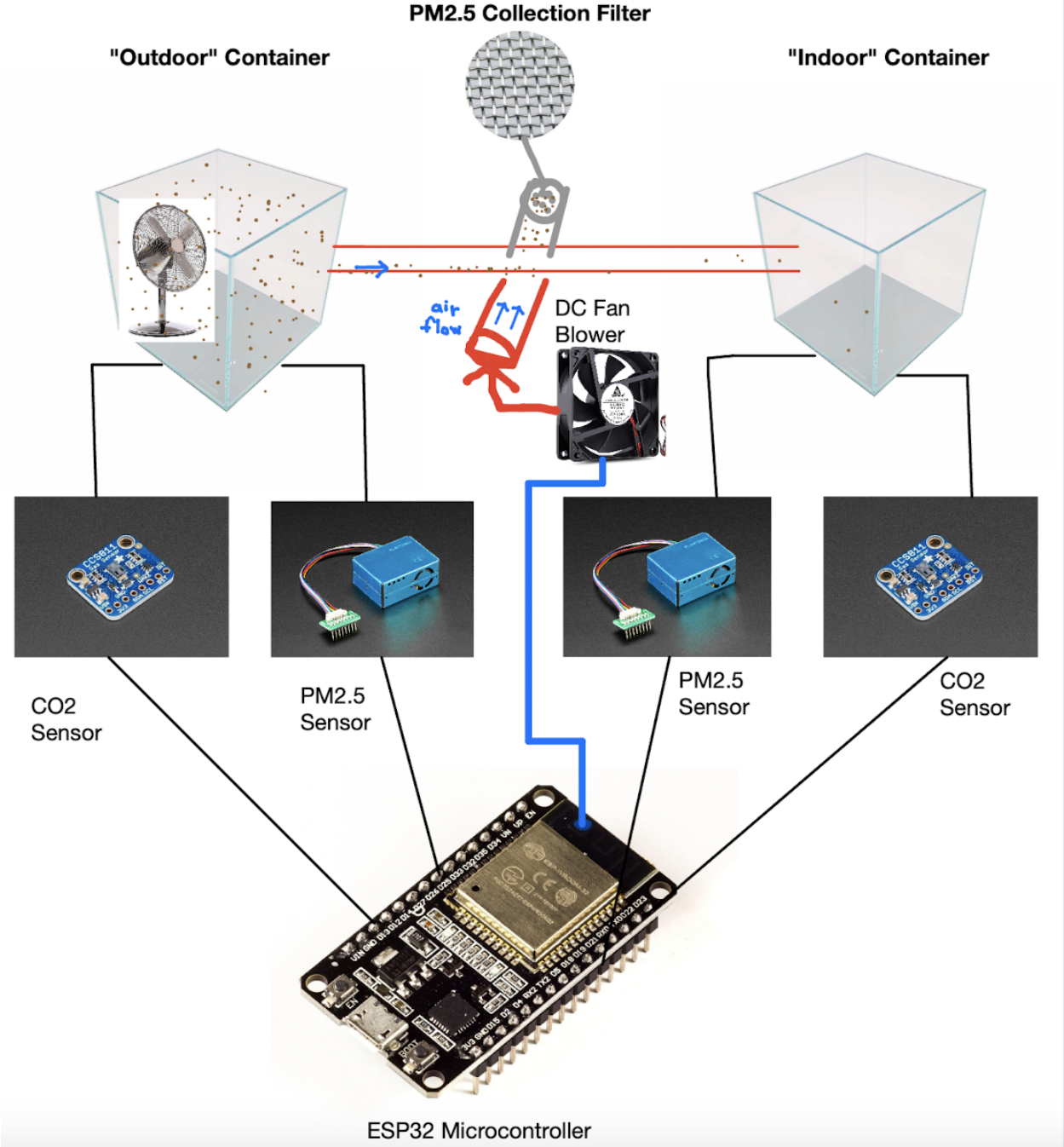
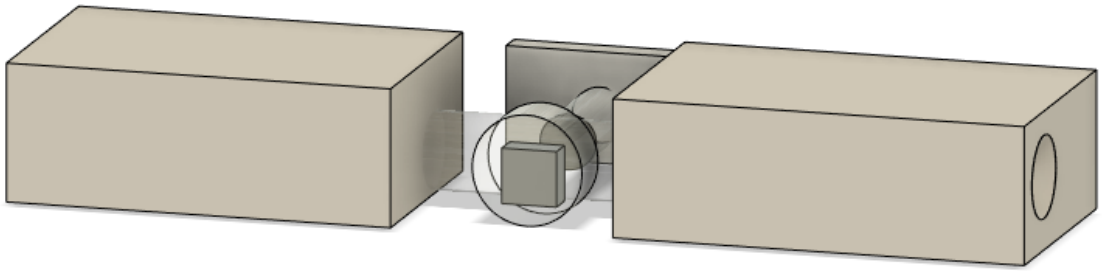


Figure 1: Setup Visual Aid



1.3 High Level Requirements

- The concentration of PM2.5 in the “indoor” enclosure should be lower than that of the “outdoor” by approximately 75-80%.
- The concentration of CO_2 will determine whether air flow will speed up or slow down based on circulation.
- In order to maintain power efficiency, the dynamic filtration mechanism must only start running when the PM2.5 or CO_2 particles reach a certain level.

Design

2.1 Block Diagram

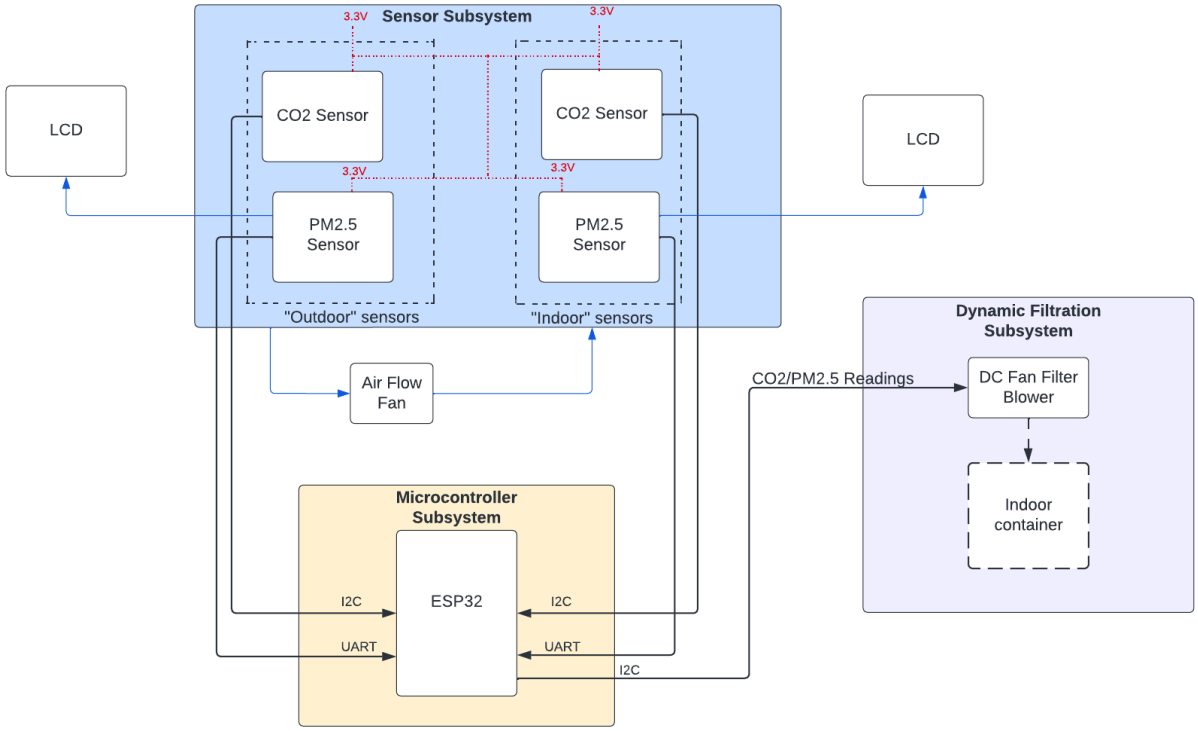


Figure 2: Block Diagram Subsystems

Our design consists of three subsystems, with the sensor and filtration subsystems housed within the enclosure of two containers joined by an intermediary tube. The ESP32 microcontroller will be located outside, hooked up to all the sensors inside the enclosure.

2.2 Physical Design

According to the visual aid above, our design will house the sensor and filtration subsystem within a clear glass enclosure. This is made up of two glass containers that are interconnected by a single tube allowing for the model of air transfer from outdoor to indoor environment. The entire enclosure will be specifically manufactured by the supply shop, and must be air-tight to capture and hold CO_2 and the PM2.5 particles without escaping into the air. To initiate the air flow between the “outdoor” and “indoor” containers, we manually switch on a fan inside the outdoor, which transfers the pollutants/CO_2 into the interconnecting tube. Within this tube there will be another DC fan that blows the dust particles into a pocket region, which will collect as many dust particles as it can through a lint filter. The rest of the filtered air will then channel into the “indoor” container,

monitored by another set of CO_2/PM2.5 sensors. The concept of inertial impaction is a proven process that is efficient with even the smallest of particles. The process itself has an overall effectiveness rate of 97.77% for particles in the range of 1 micrometer to 8 micrometers.

2.3 Data Acquisition Subsystem

The goal of the data acquisition subsystem is to feed real-time data accurately to the overall system to provide essential information for the correct implementation of the entire project. It will consist of an array of electrochemical sensors that will receive data from the enclosures' concentrations of PM2.5 particles and carbon compounds and transmit that data to the microcontroller subsystem. More specifically, we will utilize the PMSA003I PM2.5 sensor and the SGP30 VOC and CO_2 sensors, where each section of the entire enclosure will contain one of each. These are essential with regards to our high-level requirements since they will allow us to detect dust particles as well as CO2 concentrations, therefore enabling our dynamic filtration system to filter air more efficiently. We will also use the sensors to compare dust/CO2 levels in the contaminated environment with those in the "clean" environment to provide us with information with regards to the success of our filtration techniques. All four sensors (2 of each) will pass on data for analysis to the microcontroller subsystem as they will be on the same PCB and will communicate via I2C protocol. It is important to note that the sensors are already preinstalled on small PCBs, therefore we will need to connect those PCBs to our main PCB. Pictures of the sensors as well as a requirements & verification table have been provided below.

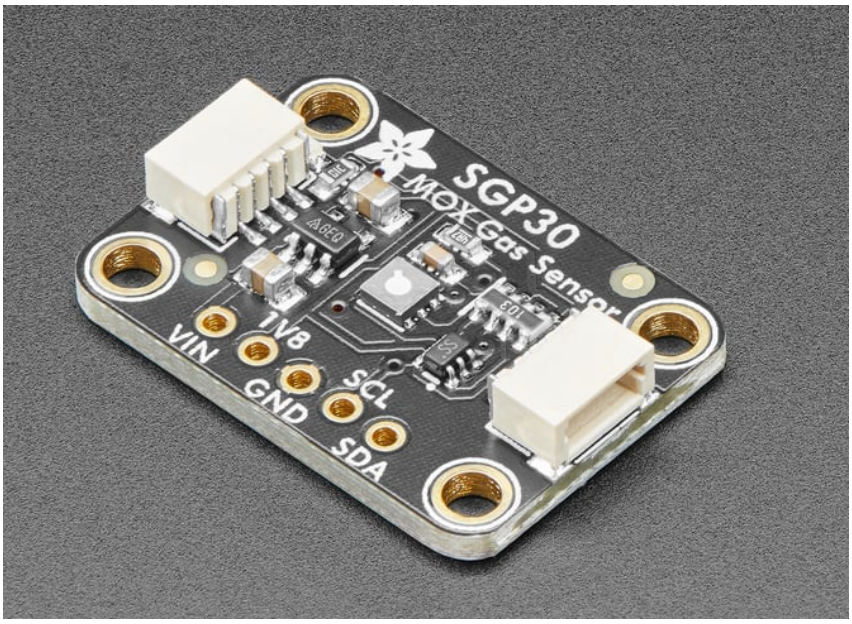


Figure 3: SGP30 VOC and eCO2 Sensor, implemented on its own PCB.

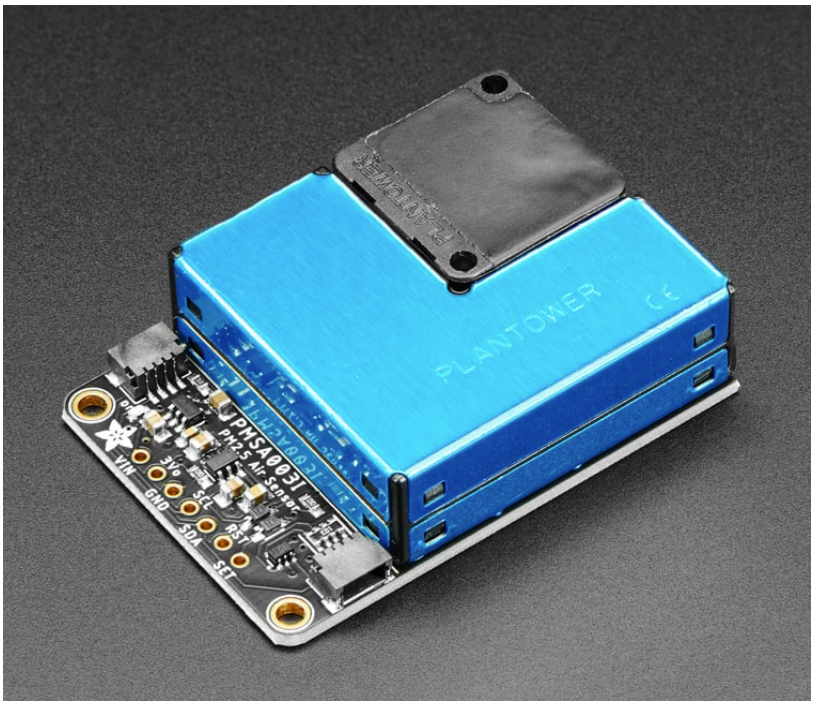


Figure 4: PMSA003I PM2.5 Sensor, implemented on its own PCB.

Requirements	Verification
• The necessary 3.3V to power both CO2 and PM2.5 sensors are	• Power supply must not be touched and should be capable of powering

constantly received regardless of sensor status.	up sensors for long periods of time.
● PM2.5 and CO2 sensors give accurate data and are not reading junk	● I2C protocol will be tested on all sensors first through a verilog file to ensure data being read is accurate consistently and constantly changes. ● Different modes and accuracies will be written onto the sensor via I2C to test the sensor and pick the most efficient environment. ● Sensors will be placed in an area of high dust and CO2 concentrations and significant changes in readings must be achieved.
● Communication between PM2.5 and CO2 sensors with microcontroller	● Ensure microcontroller and sensors are connected to each other in the PCB board. Further verification will be handled in the microcontroller subsystem section.
● If sensors stop reading data, user must be informed. If the display fails, then the Arduino IDE can print a stream of sensor data on the terminal window.	● Interface will be implemented to display the data from the sensors. This will allow us to spot if data is inaccurate or if no data is being read at all due to glitches.

2.4 Microcontroller Subsystem

The microcontroller subsystem consists of a single microcontroller that will communicate with both the data acquisition subsystem and the dynamic filtration subsystem to provide accurate instructions to the directional air flow about when to activate. Using an external power source with a measured voltage, this subsystem will be the core impetus for the functionality of this entire project.

Requirements	**Verification**
• The necessary 3.3V to power the ESP32 is constantly received whether the system is on or off	• The ESP32 should be continuously checking for particulate readings to ensure that system is performing as instructed • We will measure the voltage of the subsystem by linking the device to a multimeter, which will produce the output. • The voltage source should be a constant provider to the ESP32 without any fail
• The ESP32 microcontroller will receive data from the CO_2 and PM2.5 sensors and communicate with the sensor that is controlling the amount of air received from the external blower	• Perform the experiment by starting the initial air flow that transfers the dust/CO_2 mixture from one container to the other. • The CO_2 and PM2.5 readings should be clearly displayed when requested • Check whether the readings are reflecting the purpose of the project
• The system should adjust based on the numerical readings processed and returned by the ESP32.	• Create a program where the ESP32 receives and responds to the particulate readings • Provide contingencies on when the ESP32 should send signals to the system to turn on or off with thresholds of 40000 ppm of CO_2 or 35 micrograms/m^3.

```
void setup() {
  // our debugging output
  Serial.begin(115200);

  // Set up UART connection
  Serial1.begin(9600, SERIAL_8N1, RXD2, TXD2);
}

struct pms5003data {
  uint16_t framelen;
  uint16_t pm10_standard, pm25_standard, pm100_standard;
  uint16_t pm10_env, pm25_env, pm100_env;
  uint16_t particles_03um, particles_05um, particles_10um, particles_25um, particles_50um, part
  uint16_t unused;
  uint16_t checksum;
};

struct pms5003data data;

void loop() {
  if (readPMSdata(&Serial1)) {
    // reading data was successful!
    Serial.println();
    Serial.println("---------------------------------------");
    Serial.println("Concentration Units (standard)");
    Serial.print("PM 1.0: "); Serial.print(data.pm10_standard);
    Serial.print("\t\tPM 2.5: "); Serial.print(data.pm25_standard);
    Serial.print("\t\tPM 10: "); Serial.println(data.pm100_standard);
    Serial.println("---------------------------------------");
    Serial.println("Concentration Units (environmental)");
    Serial.print("PM 1.0: "); Serial.print(data.pm10_env);
    Serial.print("\t\tPM 2.5: "); Serial.print(data.pm25_env);
    Serial.print("\t\tPM 10: "); Serial.println(data.pm100_env);
    Serial.println("---------------------------------------");
    Serial.print("Particles > 0.3um / 0.1L air:"); Serial.println(data.particles_03um);
    Serial.print("Particles > 0.5um / 0.1L air:"); Serial.println(data.particles_05um);
    Serial.print("Particles > 1.0um / 0.1L air:"); Serial.println(data.particles_10um);
    Serial.print("Particles > 2.5um / 0.1L air:"); Serial.println(data.particles_25um);
    Serial.print("Particles > 5.0um / 0.1L air:"); Serial.println(data.particles_50um);
    Serial.print("Particles > 10.0 um / 0.1L air:"); Serial.println(data.particles_100um);
    Serial.println("---------------------------------------");
```

Figure 5: Example ESP32 Code for Sensor Hookup

2.5 Dynamic Filtration Subsystem

The Dynamic Filtration subsystem is responsible for using a programmable DC fan to filter out the PM2.5 particles. The RPM of the fan is controlled by the ESP32 output signal containing CO_2 and PM2.5 data. Based on the readings of the two concentration levels, the fan will use the concept of inertial impaction to filter out the PM2.5 dust into a separate pocket in the tube. The particles will be collected using a lint filter, and the filtered air with carbon dioxide will continue to flow along the pipe.

Requirements	Verification
• The DC fan must have enough RPM to redirect dust particles from the horizontal air flow to the collection pocket of the tube.	• Perform the experiment by starting the initial air flow that transfers the dust/CO_2 mixture from one container to the other. • During the transfer process, switch the filtering DC fan in the tube on,

	and test different RPM speeds. Observe which one successfully redirects the dust particles. ● Calculate the RPM by connecting the fan output to the oscilloscope. Observe any repetitive waveforms/spikes on the display to determine the period, then use it to calculate the frequency of rotation. ● Note down the minimum successful RPM and check if the dust concentration in the second container is reduced.
● The RPM/speed must adjust according to the PM2.5 and CO_2 concentrations detected by the sensors in the initial container, which are then transmitted to the Data Acquisition subsystem. The ESP32 must send the readings to the circuit controlling the DC fan speed.	● Create the fan speed controller circuit separately on a breadboard, using the ESP32, relay, and a pushbutton. ● Program the ESP32 to switch the relay on once the pollutants reach a certain level. Then it must alternate between on/off as real-time sensor data is collected. ● Supply 6V to the microcontroller and relay. If the fan turns when the button is pressed, then the ESP32 has proper control over the fan.
● Fan should reduce the amount of particulate matter and dangerous compounds significantly between the initial and final enclosures.	● Take the output of both sensor subsystems in the two containers sent through the I2C/UART. Hook the readings to 2 separate LCD displays, one for each container. ● Compare the concentrations between the initial and final container and verify if the final is lower.

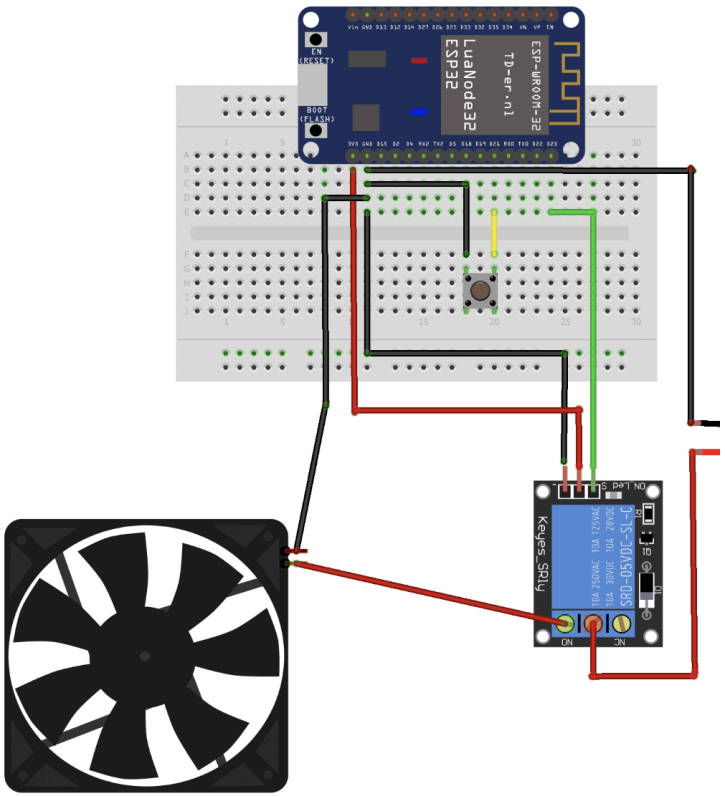


Figure 6: DC Fan Controller Circuit

As shown in the figure above, the circuit that controls the speed of the blower(DC fan) consists of a relay, the ESP32 microcontroller, a pushbutton, and the fan itself. The pushbutton is mainly used to test if the speed control works and is properly connected to the microcontroller. Once this is verified, the pushbutton will be removed from the breadboard. The fan requires 5V DC voltage, and is controlled using a PWM(pulse-width modulation) signal. The relay is connected between an ESP32 pin and the fan. It acts as a switch that turns on/off depending on the signal sent from the ESP32. In our case, it must turn on as soon as a certain PM2.5/CO_2 concentration is reached. This is done by flashing the C++ program to the microcontroller.

2.6 Tolerance Analysis

The most critical part of this project is the DC fan that functions as the programmable blower filtering the PM2.5 particles. The rated voltage is 12V, with the operation range being 4.5-13.8 VDC. The speed/airflow of the fan is proportional to the supplied voltage; if the fan VDC decreases, so does the RPM.

ITEM	DESCRIPTION
RATED VOLTAGE	12 VDC
OPERATION VOLTAGE	4.5 – 13.8 VDC
INPUT CURRENT	0.11 (MAX. 0.17) A
INPUT POWER	1.32 (MAX. 2.04) W
SPEED	8000 R.P.M. (REF.)
MAX. AIR FLOW (AT ZERO STATIC PRESSURE)	0.375 (MIN. 0.338) M^3/MIN. 13.24 (MIN. 11.94) CFM
MAX.AIR PRESSURE (AT ZERO AIR FLOW)	9.39 (MIN. 7.61) mmH_20 0.370 (MIN. 0.300) $inchH_20$
ACOUSTICAL NOISE (AVG.)	41.0 (MAX. 45.0) dB-A
INSULATION TYPE	UL: CLASS A

Figure 7: DC Fan Datasheet

According to the datasheet, the nominal speed at 12V is 8000 RPM. The tolerance level associated with this value is typically +-10%. The average peak current draw is 0.17 A, which can also be displayed as a periodic waveform of a certain frequency rather than a single value.

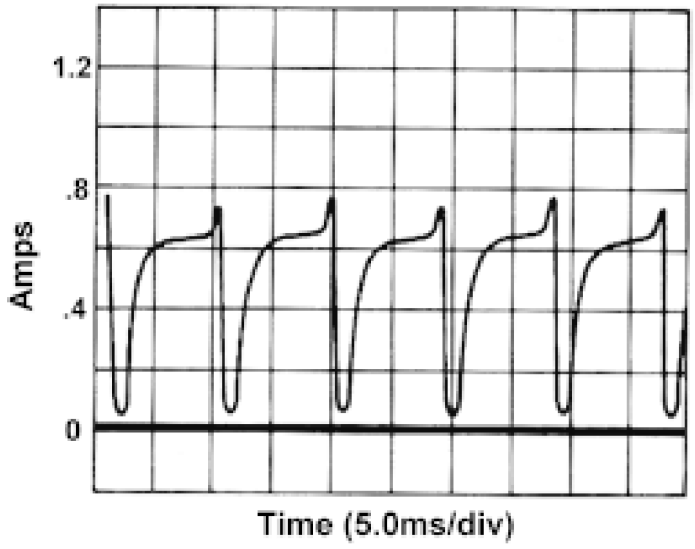


Figure 8:DC Fan Current Ripple

We can measure the running current using the true root mean square formula:

$$TRMS = (DC^2 + AC^2)^{1/2}$$

For measuring the current RPM of the fan, we need to monitor the DC ripple current. Using an oscilloscope, we can display the waveform of the current and determine the period of the current graph. This will have some slight uncertainty, since the measurement will not be exact. Then we determine RPM with:

$$f = \frac{1}{T}$$

f = frequency
T = Period

The applied voltage can never exceed the operation range 4.5-13.8 V, which means the RPM should be 8000 maximum.

Cost and Schedule

For labor costs, we can expect to earn $40/hour, therefore a salary of $40/hour x 2.5 x 50 hours = $5000 per group member. The total labor cost for all three members would be $15000.

Cost for parts:

Item	Manufacturer	Quantity	Cost	Total Cost
Adafruit SGP30 Air Quality Sensor Breakout - VOC and eCO2 - STEMMA QT / Qwiic	Adafruit Industries	2	$17.5	$35
PM2.5 Air Quality Sensor with I2C Interface - PMSA003I	Adafruit Industries	2	$39.95	$79.9
DC Fan with Speed Sensor and PWM Speed Control	Deltra Electronics	1	$9.99	$9.99

Machine Shop Help with Enclosure	ECE Machine Shop	1 (1 hour of labor)	$30	$30
Norge Dryer Lint Trap Screen Clothes Dryer Lint Filter Replacement.	Norge Dyer	1	$13.87	$13.87
Total Cost of Parts			$168.76	

The total cost of labor and parts is: $15168.76.

Schedule:

Week	Task	Group Member
2/20 - 2/27	• Order remaining parts • Complete design document • Start PCB design • Prepare for design review on 02/27 • Finish team contract (due 02/24)	• All
2/27 - 3/06	• Complete PCB design and pass audit (03/07 due date) • Meet with machine shop to manufacture enclosure • Test sensors with Verilog code via I2C protocol • Test the blower control circuit • Test microcontroller	• All • Omar • Karthik • Prithvi
3/06 - 3/13	• Assemble blower • Test efficiency of dust filter by utilizing blower and PM2.5 sensors • Complete teamwork evaluation (due 03/08) • Make sure nothing else is needed of machine shop (revisions due 03/10)	• Karthik • Omar • All
3/13 - 3/20	• Spring break	

3/20 - 3/27	• Integrate sensors with PCB board • Make sure microcontroller is capable of receiving data from sensors	• Prithvi and Omar
3/27 - 4/03	• Complete individual progress reports (due 03/29) • Make sure microcontroller is able to communicate with blower	• All • Karthik and Prithvi
4/03 - 4/10	• Program microcontroller to change blower speed depending on readings from sensors (PM2.5 and CO2) • Start assembling enclosure with PCB, blower, and filter	• All
4/10 - 4/17	• Complete team contract fulfillment (due 04/14) • Continue maximizing efficiency for data collection, blower control, dust filtration, and air circulation • Start on presentation powerpoint	• All
4/17 - 4/24	• Mock demo • Make changes based on feedback from mock demo • Complete presentation preparations	• All
4/24 - 05/01	• Final demo and mock presentation	• All
05/01 - 05/08	• Final presentation • Finish final paper (due 05/03)	• All

Ethics/Safety

There are general risks pertaining to PCB assembly and the use of electronic components. Solder will be used to stick our sensors and microcontroller onto one PCB, therefore we must beware of burn hazards as well as chemical hazards. We aim to proceed cautiously with the use of gloves as well as safety goggles. One main safety hazard related to our project specifically would be excessive inhalation of dust. Our goal is to demo our project by creating an environment filled with dust particles in order to see if our clean environment is capable of filtering all of it out. When creating this dust-infested environment, we must wear masks and control

the dispersion of dust particles as the buildup of it in our body can be very dangerous, consequences of which include lung infection and even more serious complications for those with asthma. [1]

In addition, there are IEEE safety regulations associated with the usage of blowers and their application in this project. Blowers are high demanding, high quality machines that require precise measurements and accurate usage to ensure the best possible performance. However, common issues that happen to these blowers are symmetry irregularities, rotor malfunctions, and speed. These can occur given specific instances of a manufacturing issue, a power surge, or even a power failure. Symmetrical variations in the blower may cause either incorrect directional air flow or even an internal issue in the functionality of the device. [7] Voltage dips may cause the blower to function improperly, either at low capacity or turn off, depending on the power input. The IEEE code of ethics dictates that fans can be somewhat unpredictable in their measurements, but the readings are usually accurate. [7] The variation is associated with the variability in the electrical parts themselves. Power surges may cause the capacitors within the blower to overheat and malfunction, making the entire device worthless. Given the various safety hazards associated with working with blowers, we plan on being very safe when using them. Such actions include using surge protectors to ensure minimal power surges and ensuring a battery supply to provide a constant current into the system. We also plan on purchasing a blower from a well-known manufacturer and running basic tests on the blower to test its performance and its symmetrical properties.

References

[1] “Control a Fan Using the ESP32 Card.” *Robotique Tech*, https://www.robotique.tech/robotics/control-a-fan-using-the-esp32-card/.

[2] Department of Health, “*Health Effects of Dust*”, https://www.healthywa.wa.gov.au/Articles/F_I/Health-effects-of-dust#:~:text=Currently%20it%20cannot%20be%20confirmed,and%20heart%20and%20lung%20disorders.

[3] DroneBot Workshop. “Measure Air Quality with Microcontrollers - Air Quality Sensors.” *DroneBot Workshop*, Publisher Name DroneBot Workshop Publisher Logo, 23 Sept. 2022, https://dronebotworkshop.com/air-quality/.

[4] Industries, Adafruit. “PM2.5 Air Quality Sensor with I2C Interface - PMSA003I.” *Adafruit Industries Blog RSS*, https://www.adafruit.com/product/4505.

[5] Miller, Dean. “Adafruit SGP30 Air Quality Sensor.” *Adafruit Learning System*, https://learn.adafruit.com/adafruit-ccs811-air-quality-sensor.

[6] Zhang, Xiaowei, et al. “High-Performance Inertial Impaction Filters for Particulate Matter Removal.” *Nature News*, Nature Publishing Group, 19 Mar. 2018, https://www.nature.com/articles/s41598-018-23257-x.

[7] IEEE Board of Directors, "IEEE Code of Ethics," Institute of Electrical and Electronics Engineers, 2020. Accessed: Sep. 14, 2022. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html