Indoor smoke detection & air-quality monitoring system

TEK5110 BUILDING MOBILE AND WIRELESS NETWORKS

Department of Technology System

Stephen Kimogol & Abdulaziz Abdugani

Table of Contents

Introduction

Due to the global warming, population and transportation increase there is an increased need for monitoring the indoor air quality in the urban areas.

We stay most of our time indoors and the gases contained indoors can be harmful to health and can reduce productivity. They are studies that show indoor air quality have effect on health and some of the outcomes of poor indoor air quality include sick building syndrome,respiratory tract lesion and increase in symptoms of ischaemic heart disease [1, 9]. This is also true for the elderly and children who spend most of the time indoors. In this project we present a novel indoor-smoke detection and indoor air quality monitoring system for use in homes. Indoor air quality monitoring system can be used by individuals, public health professionals and also authorities responsible for designing of buildings and ventilation system manufactures that interested in understanding indoor gases so as to make air cleaning technologies that have minimum overall energy consumption [14]. This project uses analog sensors to collect gas levels in indoor setting. If the gas levels go beyond a certain threshold the users are notified enabling them to take necessary measures. Furthermore, we have use online platform to enable real-time monitoring of the gas levels.

The rest of the report is organized this way. In the first part we describe the hardware and software we used in collecting and analyzing data. The second part describes the procedure for data collection, notifications and visualization. The third and fourth part entails the challenges faced and evaluation respectively. The last part is the conclusion

Requirements & components

The main components used in this project are the following:

Raspberry Pi 3

There are different types of single-board computers that can be used in this implementation, we chose the raspberry pi 3 with raspbian OS installed. The 64 - bit quad core processor and the on board included wifi component make the raspberry pi easy to use and remote access. The raspberry pi board has GPIO (General-purpose input/output) pins which will be used in this project.

ADS1115

The output of the MQ sensors is an analog signal that has to be converted. To this end we have used ADS1115 – analog to digital converter [5]. ADS1115 is a small, low-powered 16 bit analog to digital converter with an i2C interface, where the data is is transported through an i2C compatible serial.

MQ sensors

We have used MQ series gas sensors to collect the data. These are the sensors that were at our disposal MQ2, MQ3, MQ4, MQ5, MQ6, MQ7, MQ8, MQ9 and MQ135. The sensors are sensitive to a range of gases. We have decided to work with only two sensors in these projects. MQ2 (Methane, Butane, LPG, smoke) and MQ7 (Carbon Monoxide) [2,10,11]. According to U.S. Environmental protection agency (EPA), the major air pollutants are O_3, PM, CO, SO_2, and NO_2 [3]. In our case we will monitor indoor CO levels (MQ7) and use MQ2 for smoke detection.

Twilio

Twilio is a cloud based communication platform that allows making and receiving of phone calls, send and receive text messages and other functions with its own APIs. We have used this platform to send the sms notifications to the user.

Here is the flow of the system we implemented.

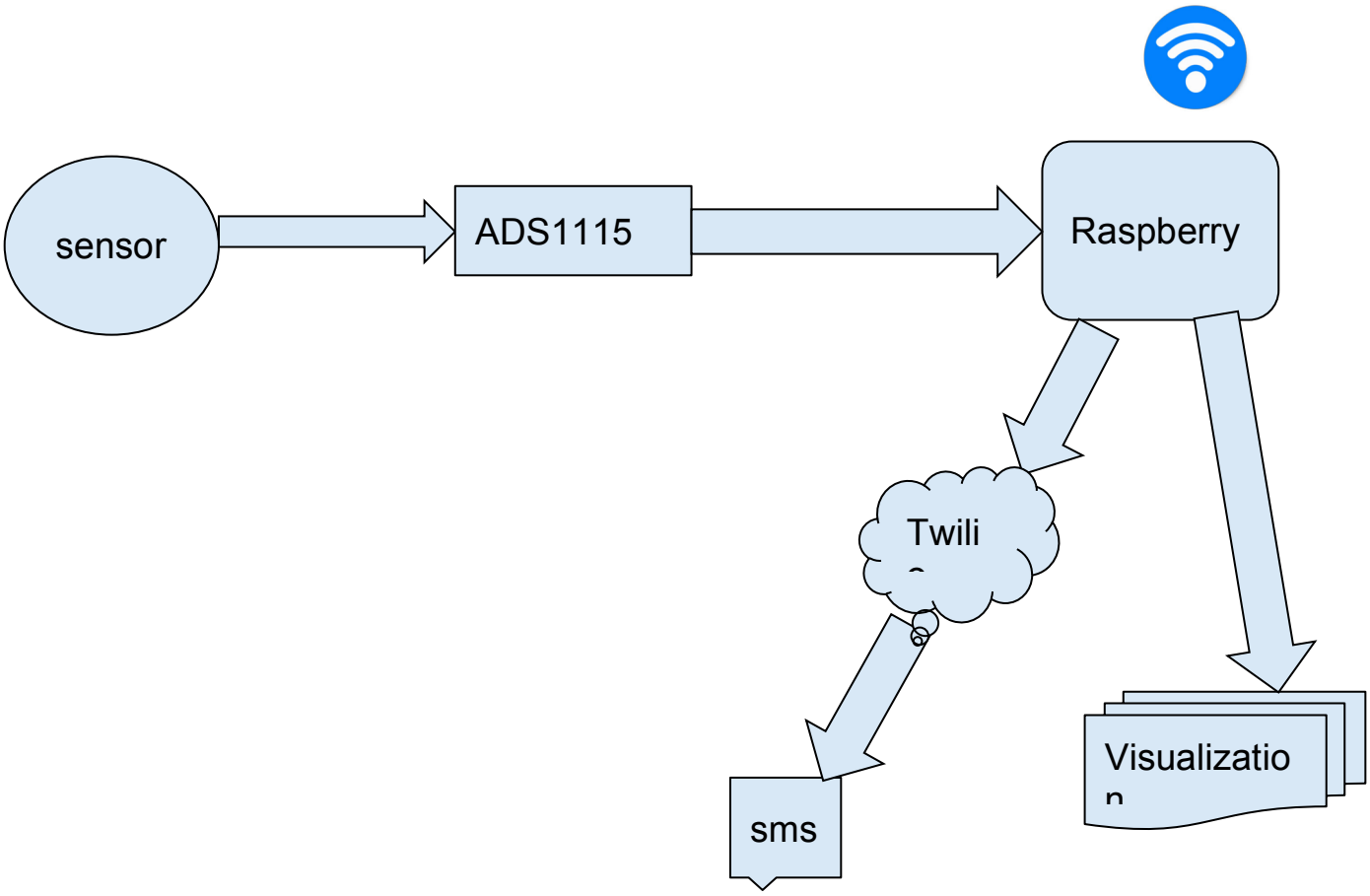

Figure 1 System flowchart

Procedure

We started by setting up the raspberry pi, installing the Raspberry OS and configuring connection to internet. We have to make sure that we have the right packages and tools for further implementation of the software part. The libraries installed were the *Adafruit ADS1x15 Python libraries* [5]. Next step is to learn how the sensors work and how we can extract/monitor data from the sensor.

The MQ-2 sensors voltage output increases if there is an increase in gas concentration in the air. Here is a simple illustration of the output of MQ-2 sensor.

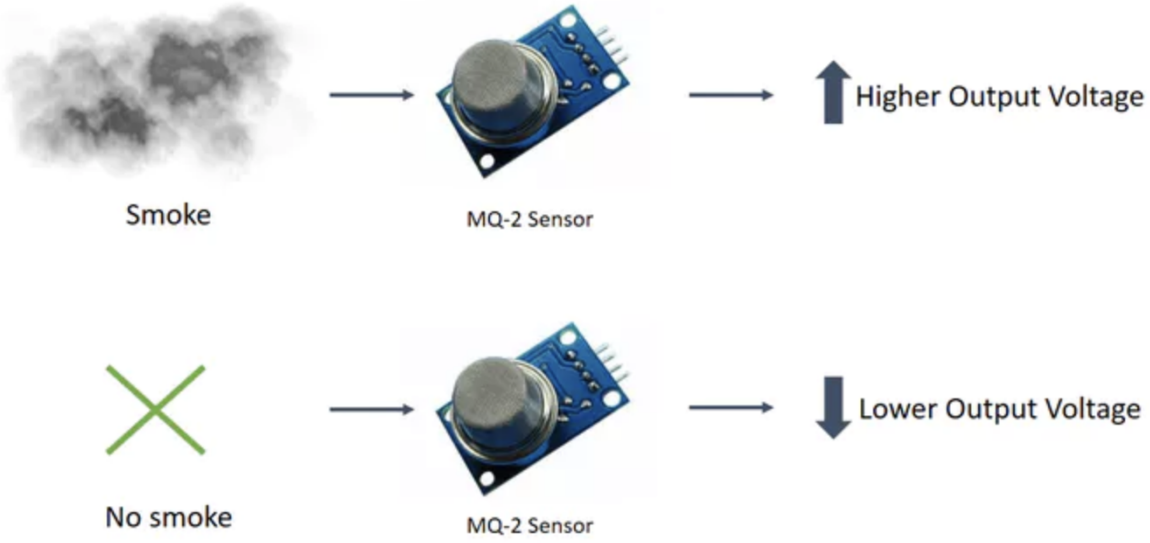

Figure 2 MQ-2 sensor

The sensor has 4 pins AO (analog out), DO (digital out, 1 or 0, acts like a threshold), VDD and GND. Most of the MQ sensors have a built-in potentiometer that can be adjusted to a certain threshold depending on how sensitive we need the sensor to be. The sensor has a comparator called LM-393 [7].

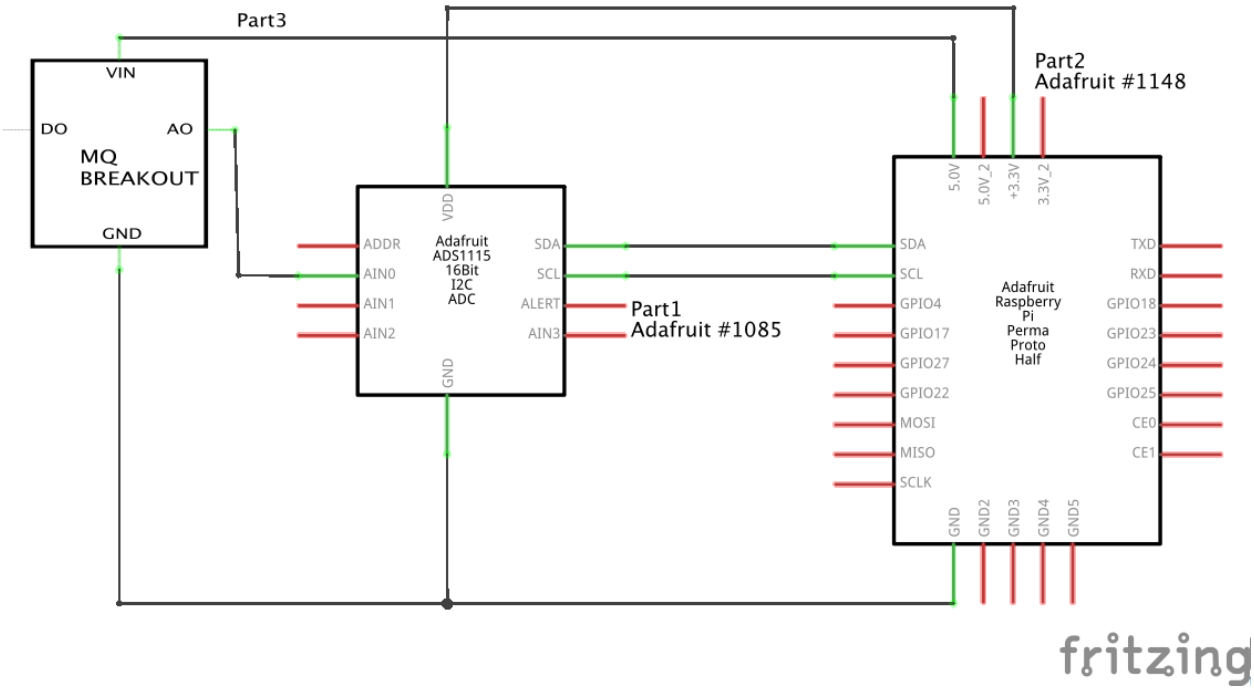

Figure 3 System scheme

The figure above shows the connections on our circuit. The important connections besides the Vdd and Gnd are the SDA, SCL and AO lines. SDA is the serial data line and SCL stands for serial clock line, these bidirectional lines are used in I2C (Inter-Integrated Circuit). After wiring the connections we had to configure and enable the ARM I2C interface by using “sudo raspi-config” command on the raspberry pi. Then we installed the I2C tools which allows us to check the output of the sensor. In the manual of i2c tool (man i2cdetect), states that we can use “i2c detect -y” command to see the output of the sensor. Notice that we have to use the “i2c detect -y 1” command since we are using the newest raspberry pi version, on older versions replace the 1 with 0, “i2c detect -y 0”. The output should produce the following:

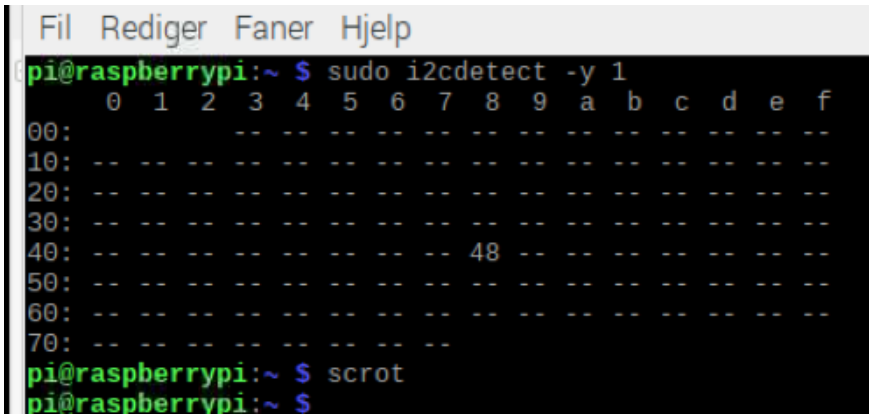

Figure 4 I2C detect -y 1 command

We can see that there is an allocated address for the sensor 0x48. Since we have the i2C and Adafruit libraries we can now put everything together and produce a python script that would do extract the sensor data, scale to a readable value and output visually.

To test the sensor result we used the “simpletest.py” from Adafruit library that we previously installed. Here are the values we got by running “python simpletest.py”

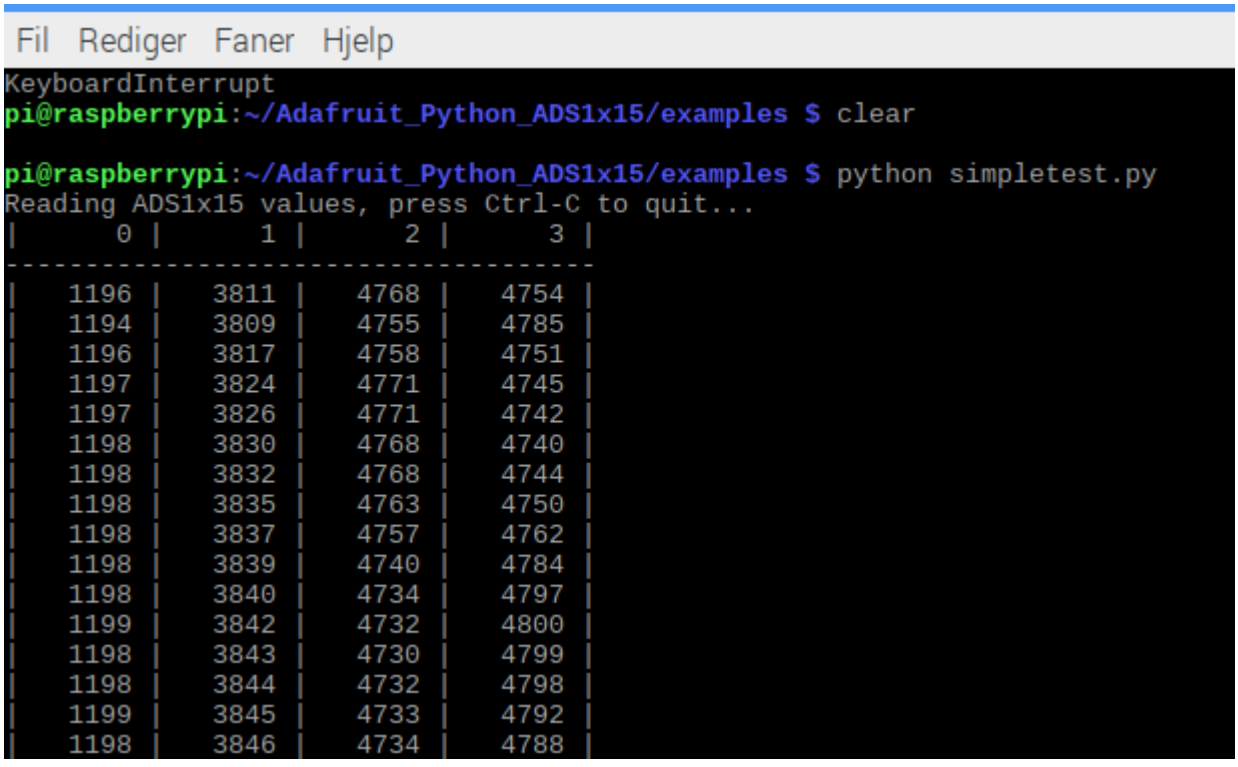

Figure 5 Adafruit library test

We can see that the script printed out values with corresponding column and rows representing four channels from 0 to 3. The value for each channel is the values coming from analog to digital converter. According to the datasheet of ADS1115, there is a maximum and a minimum number that represent the voltage. For example, the negative number -32768 represents voltage that is below the reference voltage and 32767 is the highest or the maximum voltage.

Reading from sensors and calculation of concentration in parts per million(ppm)

The MQ sensors give a single output value for the different gas concentration. We have to use the graph in the data sheet to calculate the concentration of each gas that is recorded by the sensor. To explain this we will use the MQ7 sensor that detects carbon monoxide gas. Even though the sensor is sentive to different gases, it only gives a single value. We have therefore used the senstivity curve to derive the concentration level for a specific gas.

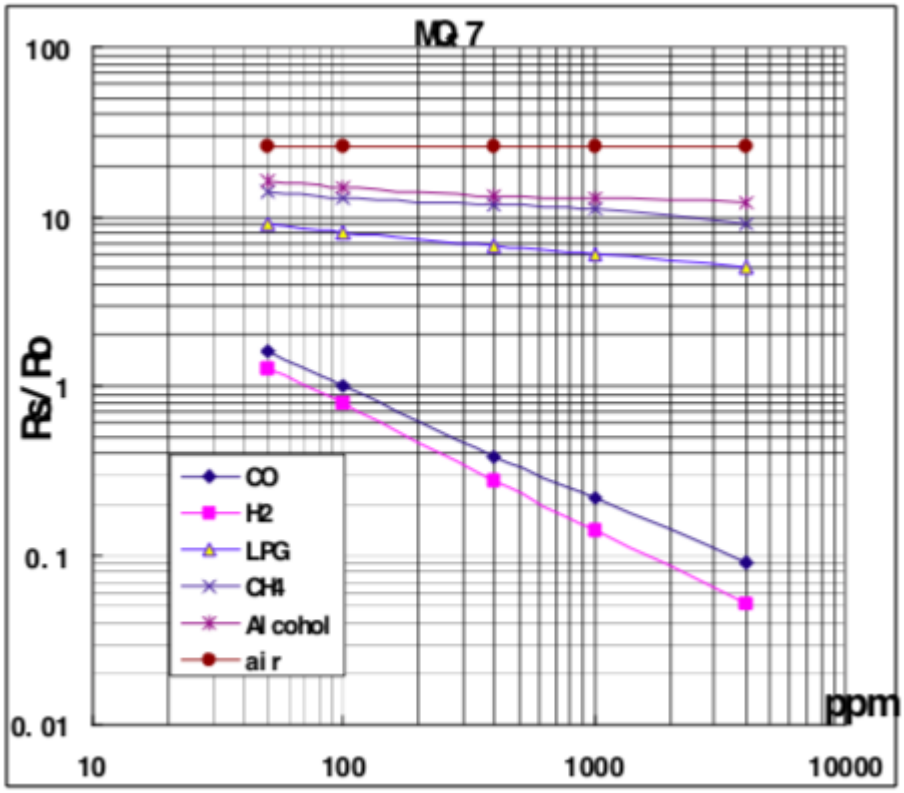

Figure 6 shows the sensitivity characteristic curve of MQ7 sensor.

The scale on the above graph is logarithmic and to read the concentration we need to get a straight line. First, we will have to take two points from the graph and calculate the slope of the curve. We pick to points from the curve P1 (x1, y1) and P2(x2, y2). Since the scaling is in log10 we need to convert it to real values. We calculate the slope first and then the y-intercept. We implement the above procedure and calculation of ppm in subsequent python code [12,13].

```
#calculate slope
slope = ( math.log10(self.y2) -  math.log10(self.y1) ) / ( math.log10(self.x2) -
math.log10(self.x1) )
 #calculate the y intercept c from linear equation
 c = math.log10 (self.y1) - m*math.log10(self.x1)
```

To get the gas concentration, we have to first calculate the gas sensing resistance RS. To get this we have read 10 samples from the sensor after every 5 seconds and then calculated the average. The sensor and the load resistor RL (adjustable) form a voltage divider and we can derive the resistance of sensor (RS) from this. RO is the Sensor resistance in 100 ppm of CO. We can find the ratio of RS/RO for clean air in the data sheet. Since we have RS and Clean Air Sensor Resistance from the data, we can derive the value of RO. RO = RS/Rs_Clean_Air.

```
sensorReading = sensorReading/numReadings; #average sensor value
sensorVoltage = ((sensorReading *6.144)/32767)
#The sensor and the load resistor forms a voltage divider
RS = LR*((5-sensorVoltage)/sensorVoltage);
RO = RS/self.Rs_Clean_Air; #Clean airResistance = RS/R0 --  from datasheet
```

We have now obtained the RO, slope and y-intercept of the CO curve and we can compute the gas concentration.

```
voltage = (adc.read_adc(channel, gain =2/3)*6.144/32767)
RS_gas = LR*((5-voltage)/voltage)
ppmRatio = RS_gas/RO
ppm_log = (math.log10(ppmRatio)-c)/m
ppm = pow(10, ppm_log)
```

Visualization & Monitoring

We have used thingsspeak platform to monitor gas levels [15]. The information can be used to make informed decision regarding ventilation and take necessary measures to improve indoor air quality.

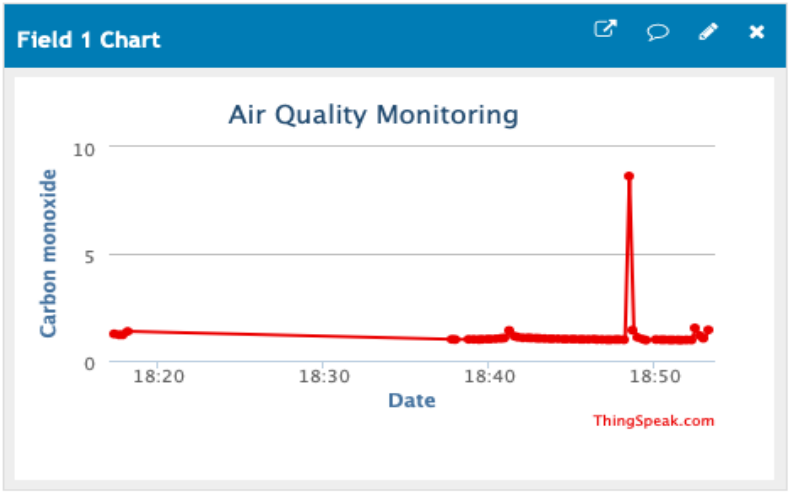

Figure 7 Chart: Carbon monoxide level (ppm) from thingsspeak.com. The spike in the ppm value is due a simulation.

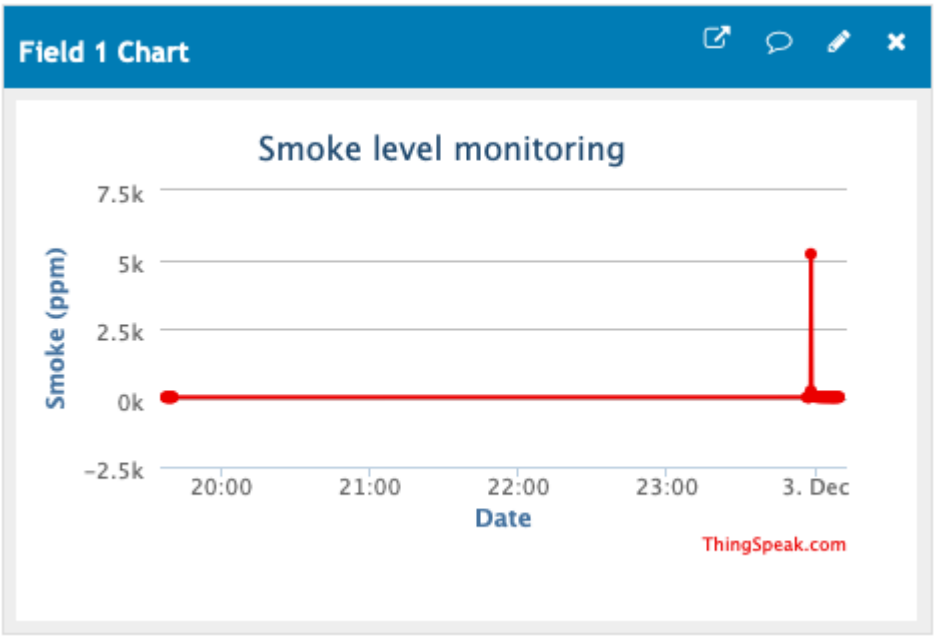

Figure 8 Smoke chart level monitoring. The spike in the smoke value is due a simulation and users are notified once smoke level gets beyond 1000 ppm.

Notification - sms/whatsapp- Twilio

To notify the users, we have used Twilio sms service. We have also considered sending WhatsApp messages from twilio but due to the limitations on the trial account we couldn't use it because Twilio assigns two different numbers for sms and WhatsApp. In our implementation we have decided to send sms to users after respective gas concentration gets to a certain level. We have set a threshold that will trigger sending of notifications to the users. For smoke detection we have set a threshold of 1000 ppm beyond which a message is sent to the user. For indoor air quality, we have used the Air Quality Index (AQI) developed by the U.S. Environmental protection agency (EPA) [4]. From their website we could enter the ppm value we have generated and get air quality index and category. AQI index has three sections: range, level of health concern(good, moderate, unhealthy for sensitive group, unhealthy, very unhealthy and to hazardous) and six colour symbolizing different levels. The range runs from 0-500 with 0-50 representing good air quality and 300 - 500 represents hazardous air quality. We have therefore based our threshold on this index. In our case we have set a ppm of 5 measured by MQ7 sensor for CO to be the threshold beyond which we send a notification (sms) to user through Twilio. A ppm of 5 will give a range of 56 and AQI category 'moderate' for level of health concern. We could have used a greater threshold but because we couldn't simulate environment needed to generate 15 ppm to get to category unhealthy, we decided to settle for 5 ppm as our threshold. We have not focused on advising the customers on what to do when the indoor air quality drops as we consider this to beyond the scope of this project.

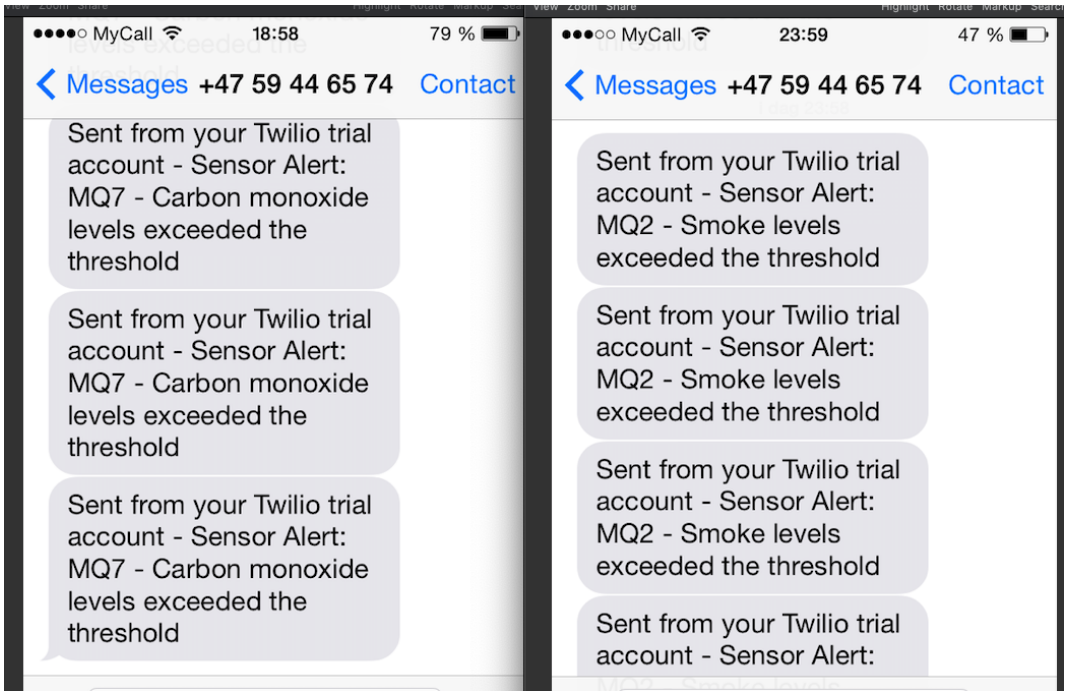

Figure 9 Sms from Twilio for carbon monoxide and Smoke.

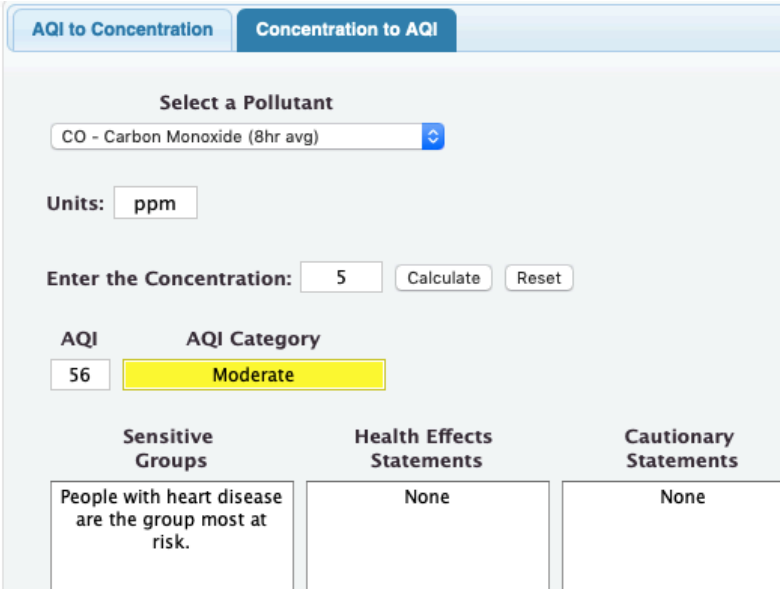

Figure 10 Air quality index calculator from the U.S. Environmental protection agency (EPA). Conversion of 5 ppm to AQI index and category.

Air Quality Index (AQI) Values	Levels of Health Concern	Colors
When the AQI is in this range:	*..air quality conditions are:*	*...as symbolized by this color:*
0 to 50	Good	Green
51 to 100	Moderate	Yellow
101 to 150	Unhealthy for Sensitive Groups	Orange
151 to 200	Unhealthy	Red
201 to 300	Very Unhealthy	Purple
301 to 500	Hazardous	Maroon

Figure 11 Air quality index and category.

Challenges

In this section we will describe some challenges we encountered with, these complications made us rethink some approaches to the project. One of the problems that took a lot of time researching was the following error:

```
pi@raspberrypi:~/Adafruit_Python_ADS1x15/examples $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Figure 12 UU error

Figure 5: Gives an error “UU” on address 0x48.

This “UU” error means that the address is already loaded or occupied by another driver, to bypass this we used command “sudo modprobe -r” which unloads the driver and “sudo modprobe i2c-dev” to load it back. We also had to configure the module file which is located at “etc/modules” to make sure that the drivers startup on boot and uncomment i2c module so that they are loaded into memory. This ensures that the module is not “busy” and can be used.

Another challenge we encountered was the hardware connections on the ADS1115 chip. It turns out the ADS1115 chip is very sensitive and has to be very well soldered to make sure good contact with the breadboard, sensors and the raspberry pi.
The resistance of the sensor value varies depending on the concentration of the gas measured. The sensors are required to be preheated for about 48 hrs. We have managed to preheat them only for 10hrs. Since sensitivity curves of the sensors are in logarithmic scale, picking the points on the curve to calculate slope and y-intercept might not be precise.
Another challenge we faced is connecting different sensors to ADS1115 at the same time. When only one sensor is connected, the ADS works fine but when another sensor is connected the values of first sensor change even though we used different channels for each sensor. We have therefore made connections for each sensor separately.

Evaluation

This project proves that it is possible to monitor the air and smoke with a simple user notification in a cost-efficient way. The project can be modified with an Air sampler that continuously draws air from a specific monitored area and makes samples to analyse. This project gives an indication of quality of air in a room and the accuracy of measurement of air quality could have been improved if other sensors that measure different pollutants are used. For smoke detection we have set a 1000 ppm to be threshold so as the notify the user. To store data for visualization and analysis, it could have been better to use Elastic Stack monitoring platform [16] instead of thingsspeak platform. If this system is installed in large number of homes, the air quality data collected can also be useful for public health officials in urban ecosystem.

Conclusion

To implement this project, it is required knowledge in electronics both hardware and software. This product could be further realized to achieve other goals with accurate measurement from the sensors. Smoke detection system combined with an air monitoring system are already widely used in different environments and scenarios. This project was realized with few components which we think is important in environments that require limited resources.

The goal of the project is to warn users about the air quality in the room, our product will additionally act as a smoke detector.

References

1. Rui Pitarma, Gonçalo Marques, Bárbara Roque Ferreira 2016 “Monitoring Indoor air quality for enhanced occupational health” [online] available: *https://link.springer.com/article/10.1007/s10916-016-0667-2* [Accessed on 14.11.2018]

2. Sparkfun “Technical data MQ-7 gas sensor” [online] available: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf [Accessed on 10.11.2018]

3. H. Wang*, C. Tseng and T. Hsieh “Developing an indoor air quality index system based on the health risk assessment” 2008 [online] available: https://www.isiaq.org/docs/papers/749.pdf [Accessed on 22.11.2018]

4. U.S. Environmental protection agency (EPA) “Air Quality Index (AQI) calculator” Available:https://www.airnow.gov/index.cfm?action=airnow.calculator [Accessed on 20.11.2018]

5. ADS1015/ADS1115. https://learn.adafruit.com/raspberry-pi-analog-to-digital-converters/ads1015-slash-ads1115 [Accessed on 10.11.2018]

6. Ultra-Small, Low-Power, 16-Bit Analog-To-Digital Converter with Internal Reference: https://cdn-shop.adafruit.com/datasheets/ads1115.pdf [Accessed on 15.11.2018]

7. LMx93-N, LM2903-N Low-Power, Low-Offset Voltage, Dual Comparators: http://www.ti.com/lit/ds/symlink/lm393-n.pdf [Accessed on 15.11.2018]

8. I2C Interface Raspberry Pi: https://www.raspberrypi-spy.co.uk/2014/11/enabling-the-i2c-interface-on-the-raspberry-pi/ [Accessed on 16.11.2018]

9. Selected pollutants: WHO guidelines for indoor air quality 2010,[online] Available: http://www.euro.who.int/__data/assets/pdf_file/0009/128169/e94535.pdf [Accessed on 28.11.2018]

10. Hanwei Electronics “MQ2 Gas Sensor Technical data” [online] Available: https://www.mouser.com/ds/2/321/605-00008-MQ-2-Datasheet-370464.pdf [Accessed on 20.11.2018]

11. Olimex “MQ135 Gas Sensor Technical data” [online] https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf [Accessed on 20.11.2018]

12. Donghoon Shin, Elisha Earley, Pranshu Trivedi and Matthew Kern, 2017 “IoT Gas Sensing” Available : https://os.mbed.com/users/laughatm2/notebook/iot-gas-sensing/ [Accessed on 20.11.2018]

13. Mysensors “Gas Sensors” [online] Available: https://www.mysensors.org/build/gas [Accessed on 15.11.2018]

14. Chris Muller “ASHRAE standards 62.1 IAQ procedure and LEED” 2014 [online], available: https://www.epa.gov/sites/production/files/2014-08/documents/ciaq-webinar-muller.pdf [Accessed on 01.12.2018]

15. The open IoT platform with MATLAB analytics https://thingspeak.com [Accessed 15.11.2018]

16. Elastic Stack monitoring platform https://www.elastic.co/products/kibana [Accessed 02.12.18]