



**T.C.
GEBZE TEKNİK ÜNİVERSİTESİ**

Elektronik Mühendisliği Bölümü

**ELM 491
BİTİRME PROJESİ
I**

HAVA KALİTE KONTROL KİTİ

**Yusuf Hamdi Saylam 151024069
Ömer Konan 171024085**

**Danışman
Dr. Öğr. Üyesi Atilla Uygur**

**Haziran 2020
Gebze, KOCAELİ**

ÖNSÖZ

Bu kılavuzun ilk taslaklarının hazırlanmasında emeği geçenlere, projemizin geliştirilmesinde yol gösterici olan Sayın Dr. Ört. Üyesi Atilla Uygur hocamıza içten teşekkürlerimizi sunarız.

Haziran, 2020

**Ömer Konan
Yusuf Hamdi Saylam**

İÇİNDEKİLER

ÖNSÖZ	2
İÇİNDEKİLER	3
ŞEKİL LİSTESİ	4
ÖZET	5
GİRİŞ	6
1. MALZEMELER	7
1.1 Raspberry PI 4	7
1.2 5V 3A USB-C Güç Kaynağı	8
1.3 Arduino Uno	8
1.4 DHT 22 Sıcaklık ve Nem Sensörü	9
1.5 MH-Z19 Kızılötesi CO2 Ölçme Sensör Modülü	9
1.6 MQ-2 Yanıcı Gaz ve Duman Sensörü	10
1.7 MQ – 7 CO Gaz Sensörü	10
1.8 MQ – 135 Hava Kalitesi Ölçüm Sensörü	11
2. ELEKTRONİK ELEMANLARIN ÇIKTILARI VE BAĞLANTILARI ----	12
2.1 - DHT 22-----	12
2.1.1 – DHT 22 OUTPUT	12
2.1.2 – DHT 22 RASPBERRY PI BAĞLANTISI	12
2.2 – MH-Z19	13
2.2.1 – MH-Z19 OUTPUT-----	13
2.2.2 – MH-Z19 RASPBERRY PI BAĞLANTISI	13
2.3 – MQ-2, MQ-7, MQ-135-----	14
2.3.1 – MQ-2, MQ-7, MQ-135 ARDUINO BAĞLANTISI	14
2.4 – RASPBERRY PI 4 OUTPUT	14
2.5 – ARDUINO UNO OUTPUT-----	14
3. YAZILIM GEREKSİNİMLERİ-----	15
3.1 Arduino IDE-----	15
2.1.1MQ-2 Kodları-----	15
3.2 Python Modülleri-----	18
2.2.1 MH-Z19 Kodları-----	18
2.2.2 DHT-22-----	18
4.SONUÇ-----	19

ŞEKİL LİSTESİ

ŞEKİL 1 : RASPBERRY PI 4

ŞEKİL 2 : 5V 3A ADAPTÖR

ŞEKİL 3 : ARDUİNO UNO

ŞEKİL 4 : DHT-22

ŞEKİL 5 : MH-Z19

ŞEKİL 6 : MQ-2

ŞEKİL 7 : MQ-7

ŞEKİL 8 : MQ-135

ŞEKİL 9 : DHT22-OUTPUT

ŞEKİL 10 : DHT22 BAĞLANTI

ŞEKİL 11 : MHZ-19 OUTPUT

ŞEKİL 12 : MH-Z19 BAĞLANTI

ŞEKİL 13 : MQ-2,MQ-7, MQ-135 OUTPUT

ŞEKİL 14 : MQ-2, MQ-7, MQ-135 BAĞLANTI

ŞEKİL 15 : RASPBERRY PI 4 OUTPUT

ŞEKİL 16 : ARDUİNO UNO OUTPUT

ŞEKİL 17 : ARAÇ İSKELETİ

ŞEKİL 18 : ARAÇ AYRINTILI GÖSTERİM

ŞEKİL 19 : KUMANDANIN İSKELETİ

ŞEKİL 20 : KUMANDANIN KULLANIM GÖSTERİMİ

ÖZET

Günlük yaşamda bina içinde gaz kaçakları, yangın ve iç ortamın hava kalitesi bina işleticileri tarafından ele alınan problemler arasındadır. Bu sorunun çözümü olarak düşük güç tüketen, bulut sistemine bağlı, sensörlerden alınan verilerin bulut sistemine kaydedilerek merkezi sistemden hava kalitesinin kontrol edildiği bir kit geliştirilmeye karar verdik.

İki aşama olarak ayrılan projede birinci aşamada sensörlerin ve kullanılacak kitlerin seçilmesi, bu kitlerle sensörlerden alınan verilerin okunması hedeflenmiştir. Hedeflenen verilerin buluta sisteme aktarılması ve sonuçlar çıkarılması projenin ikinci kısmında gerçekleştirilecektir.

Proje kapsamında kullanılan sensörlerden CO, CO₂, Sıcaklık, Nem, Duman verileri sayısal olarak toplanacak ve bu veriler ile hava kalitesine dair sonuçlar çıkarılacaktır. Yapılan bu kit sayesinde toplanan veriler ile daha sonrasında makine öğrenimi algoritmaları kullanılarak, oluşabilecek sorunlar önceden tespit edilebilir ve erken müdahale mümkün olabilir. Bu proje makine öğrenimi algoritmaları çalışmaları için veri ihtiyacını gidermede kullanılacaktır. Fakat proje kapsamında makine öğrenimi algoritmaları test edilmeyecektir.

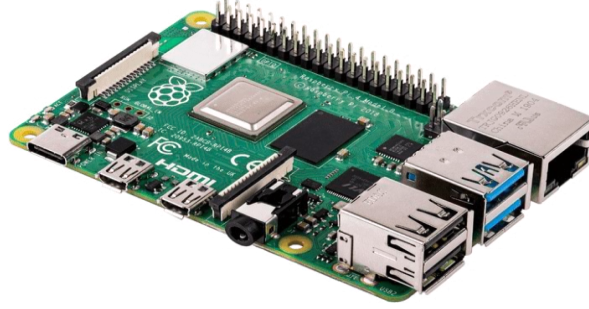
GİRİŞ

Bu proje GTÜ ELM 491 Bitirme Çalışması I dersi kapsamında hazırlanmaktadır. Projede uzaktan bulut tabanlı hava kalite kontrol kiti geliştirilecektir.. Proje sonucunda bulunulan ortamın hava kalitesini gösteren verileri toplanacak ve bu veriler bulut sistemine aktarılacaktır.

1. MALZEMELER

Bu bölümde projede kullanılan malzemeler tanıtılmış ve kullanım nedenleri açıklanmıştır.

1.1 - Raspberry PI Bilgisayar



Şekil 1

Özellikler

- 1.5 GHz dört çekirdekli ARM Cortex-A72 CPU
- 4GB LPDDR4 RAM SKU | 2GB Modeli
- VideoCore VI Grafikleri
- 4kp60 HEVC video
- Gerçek Gigabit Ethernet
- 2 × USB 3.0 ve 2 × USB 2.0 bağlantı noktalar
- 2 × mikro HDMI bağlantı noktası (1 × 4K@60Hz veya 2 × 4K@30Hz)

Geliştirilmesi hedeflenen kitin bütün işlemlerini kontrol eden düşük güç tüketen, güçlü ve uygun fiyatlı bir bilgisayar. En büyük özelliklerinden biri küçük hacimli olması ve digital girdi almak için yeterli pinlere sahip olması. Ayrıca içersinde barındırdığı UART, I2C gibi haberleşme protokolleri ile sensör bağlantılarını kolay gerçekleştirilebilmesi ve sıradan bir bilgisayar gibi bulut sistemlerine bağlanabilmesi için wifi ve işletim sistemi kolaylığı sağlaması proje için seçilmesine sebep olmuştur.

1.2 5V 3A Güç Kaynağı

Projenin güç isterlerinin karşılanabilmesi için gerekli gerilim ve akım değerlerini sağlayacak olan adaptör, Raspberry PI'nın kutu içeriğinde bulunan 5V 3A USB-C portlu adaptör kullanılmıştır.



Şekil 2

1.3 Mikrodenetleyici



Şekil 3

Arduino UNO

Çalışma Gerilimi: 5V

Dijital I/O Pinleri: 14 (6 tanesi PWM çıkışı)

Analog Giriş Pinleri: 6

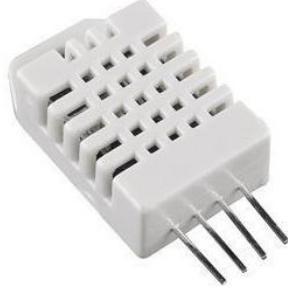
Her I/O için Akım: 40 mA

3.3V Çıkış için Akım: 50 mA

Boyut: 68.6*53.4 mm

Sensörler içerisinde analog çıkış veren sensörler olması ve kullandığımız ana bilgisayarımızın adc dönüştürücüye sahip olmaması, proje ekipman seçiminde adc dönüştürücü arayışına sebep oldu. Hali hazırda elimizde bulunan Arduino Uno'nun bu özelliğe sahip olması proje masraflarını kısmak ve proje geliştirme sürecinde dünyada yayılan COVID 19 nedeniyle aranan elektronik eşyaların temininde sıkıntılar yaşanması nedeniyle elimizde bulunan elemanlarla sorunları çözme kararı alındı. Mikrodenetleyicinin analog girişi ile okunan sensör verilerini ana işlemcimizi barındıran Raspberry PI'a iletmemiz gerekti. Bunu problem de seri haberleşme yöntemi ile çözüldü. Böylelikle Arduino ile Raspberry PI arasındaki USB bağlantısı, hem mikrodenetleyicinin gücünün Raspberry PI üzerinden sağlanması hem de ikisi arasındaki haberleşmenin gerçekleştirilmesi sağlandı.

1.4 - DHT22 Sıcaklık Nem Sensörü



Özellikleri:

Dijital çıkış vermektedir.
Çalışma Gerilimi: 3.3-5 VDC
Nem: 0-100%RH
Sıcaklık: -40 - 80 °C
Hassasiyet:
Nem: +/- %3 (Max %5) RH
Sıcaklık: < +/- 1°C
Ölçüm Periyodu: 2 s
Ürün Ölçüleri: 22x28x5 mm

Şekil- 4

DHT22 sıcaklık ve nem algılayıcı kalibre edilmiş dijital sinyal çıkışı veren gelişmiş bir sensör birimidir. Yüksek güvenilirliktedir ve uzun dönem çalışmalarda dengelidir. 8 bit mikroişlemci içerir, hızlı ve kaliteli tepki verir.

-40 ile 80°C arasında +/-1°C hata payı ile sıcaklık ölçen birim, 0-100% RH arasında +/-5% RH hata payı ile nem ölçümü yapabilmektedir. Sensör ölçümü olarak sensörün data toplama periyodundan kaynaklı olarak 2 saniyelik periyotlarla ölçüm sonuçları alınabilmektedir.

1.5 - MH-Z19 Kızılötesi CO2 Ölçme Sensör Modülü

Özellikleri:

- Hedef Gaz: CO2 / Karbondioksit
- Çalışma voltajı: 3.6 ~ 5.5 V DC
- Ortalama akım: <18 mA
- Arayüz seviyesi: 3,3 V
- Ölçüm aralığı: 0 ~% 0,5 VOL isteğe bağlı
- Çıkış sinyali: UART - PWM
- Ön ısıtma süresi: 3 dakika
- Yanıt Süresi: T90 < 60 s
- Çalışma sıcaklığı: 0 ~ 50 ° C
- Çalışma nemi: 0 ~ 95% RH (Yoğunlaşma yok)
- Boyut: 33 mm × 20 mm × 9 mm (L × G × Y)
- Ağırlık: 21 g
- Yüksek hassasiyet, yüksek çözünürlük
- Düşük güç tüketimi
- Kararlı ölçümleme



Şekil - 5

1.6 – MQ-2 Yanıcı Gaz ve Duman Sensörü

MQ-2 gaz sensörü ortamdaki yanıcı ve sigara dumanını algılayabilen bir gaz sensörüdür. Ortamdaki gaz yoğunluğuna göre bir analog çıkış verir. Duman algılama aralığı 300 – 10000 ppm'dir.

Sensör kullanımı kolaylaştırmak adına bir kart üzerine monte edilip modül haline getirilmiştir. 5V ile beslenir. Sensörden hem dijital hem de analog çıkış alınabilir. Böylece hem gaz var-yok şeklinde yada gazın yoğunluğu tespitleri yapılabilir. Modül arkasında bulunan pot ile hassasiyet ayarı yapılabilir. Arduino, Raspberry Pi yada diğer mikrokontrolcülerle rahatlıkla kullanılabilir. Raspberry Pi ile kullanmak istediğinizde eğer analog pini kullanmak isterseniz arada Mcp3008 gibi bir çevirici kullanmanız gerekir.



Şekil -6

1.7 – MQ – 7 CO Gaz Sensörü

MQ-7 Karbonmonoksit gazı sensörü 10ppm ve 10.000ppm konsantrasyonlarda Karbonmonoksit algılar. Diğer MQ sensörler gibi bu sensör de çıkış olarak gazın yoğunluğuna göre analog voltaj çıkışı verir. 10.000ppm ve 300ppm aralığında algılama yapabilmek gaz kaçağı için uygundur.

Özellikleri:

- Çalışma Sıcaklığı : -10°-50°C
- Çalışma Voltajı : 5V
- Çektiği Akım : 150 mA



Şekil – 7

1.8 – MQ-135

Çalışma voltajı: 5V DC

Özellikleri:

- Mikroprosesör uyumlu TTL çıkışı.
- Analog çıkış. Çıkış voltajı, havadaki gaz konsantrasyonuna orantısal olarak değişir.
- Yüksek hassasiyet. Sülfür, benzen, su buharı, duman ve diğer zararlı gazların(NH₃, NO_x, Alkol, CO₂ vb) konsantrasyonunu hassas bir şekilde ölçer.
- Uzun bir çalışma ömrü ve kararlılığa sahiptir.
- Hızlı cevap süresi.
- Soketli problarla test için tak/çıkar kolaylığı.

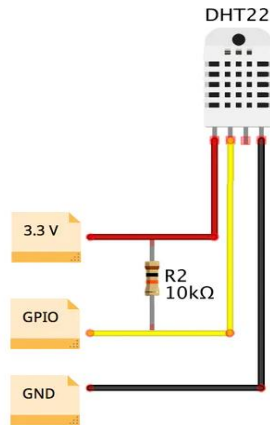


Şekil – 8

2. ELEKTRONİK ELEMANLARIN ÇIKTI VE BAĞLANTILARI

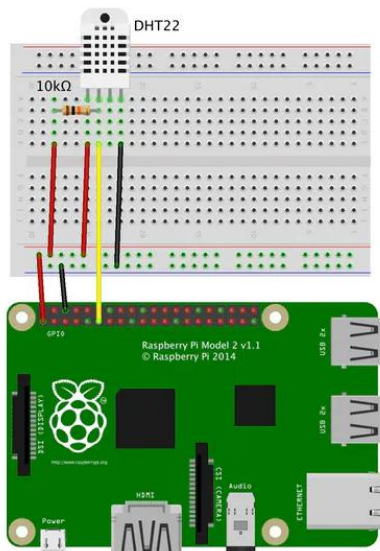
2.1 DHT-22

2.1.1-DHT-22 OUTPUT



Şekil -9

2.1.2- DHT-22 RASPBERRY PI BAĞLANTISI



Şekil – 10

2.2 - MH-Z19

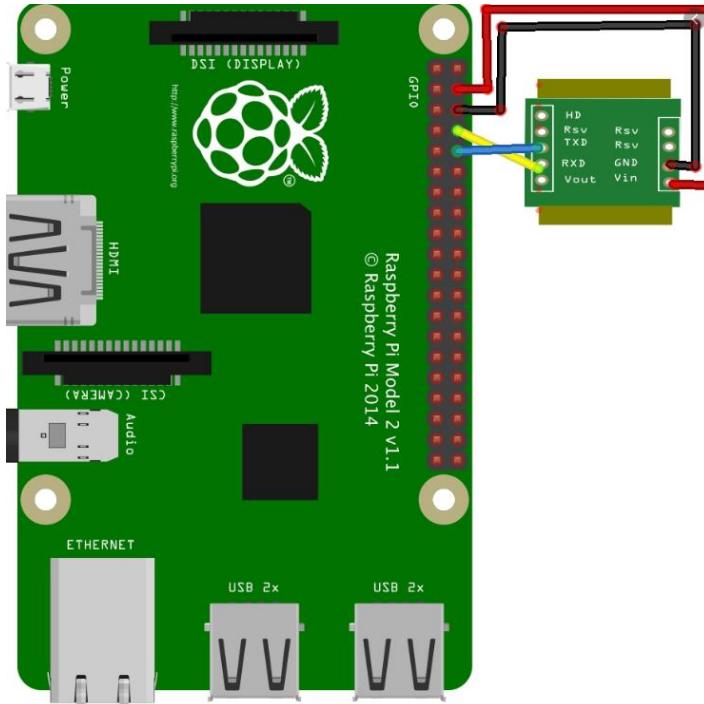
2.2.2-MH-Z19 OUTPUT

PIN	Description
Pin 6	Vin (voltage input)
Pin 7	GND
Pin 1	Vout (output voltage 3.3V, output current lower than 10mA)
Pin 9	PWM
Pin 5	HD (zero calibration, low level above 7 seconds) (Factory Reserved)
Pin 2	UART (RXD) 0~3.3V digital input
Pin 3	UART (TXD) 0~3.3V digital output
Pin 4	SR (Factory Reserved)
Pin 8	AOT (Factory Reserved)



Şekil – 11

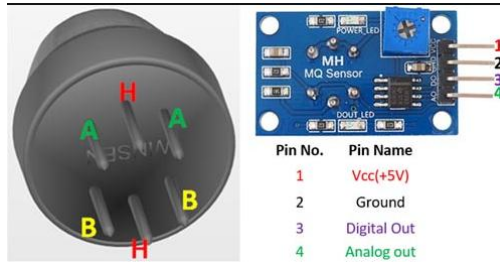
2.2.2- MH-Z19 RASPBERRY PI BAĞLANTISI



Şekil – 12

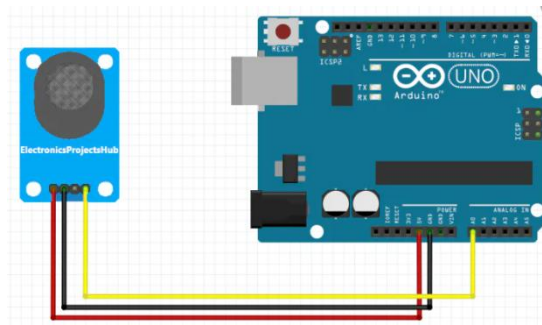
2.3 MQ-2, MQ-7, MQ-135

2.3.1 MQ-2, MQ-7, MQ-135 OUTPUT



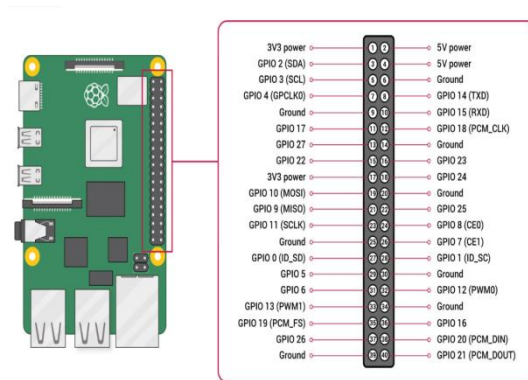
Şekil-13

2.3.2- MQ-2, MQ-7, MQ-135 ARDUİNO UNO BAĞLANTISI



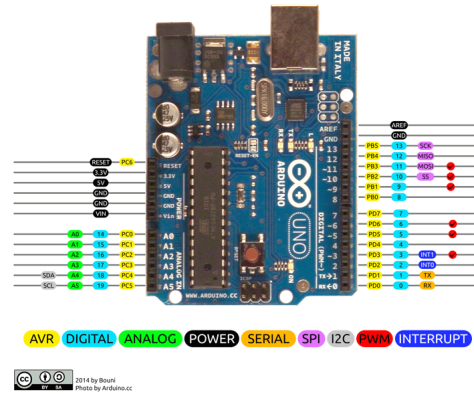
Şekil-14

2.4 RASPBERRY PI-4 OUTPUT



Şekil – 15

2.5- ARDUINO UNO OUTPUT



Şekil-16

3. YAZILIM

3.1 Arduino IDE

Arduino IDE, arduino kitleri için geliştirdiği; komutların yazılmasına, derleme işleminin yapılmasına ve son olarak da derlenen kodları doğrudan arduino kitine yüklenmesine olanak sağlayan yazılım geliştirme platformudur.

3.1.1 MQ2 Kodu

```
1  #define      MQ_PIN              (0)      //define which analog input channel you are going to use
2  #define      RL_VALUE            (5)      //define the load resistance on the board, in kilo ohms
3  #define      RO_CLEAN_AIR_FACTOR (9.83)  //RO_CLEAN_AIR_FACTOR=(Sensor resistance in clean air)/RO,
4                                          //which is derived from the chart in datasheet
5
6  /*****Software Related Macros*****/
7  #define      CALIBRATION_SAMPLE_TIMES (50) //define how many samples you are going to take in the calibration phase
8  #define      CALIBRATION_SAMPLE_INTERVAL (500) //define the time interval(in milisecond) between each samples in the
9                                          //calibration phase
10 #define      READ_SAMPLE_INTERVAL (50) //define how many samples you are going to take in normal operation
11 #define      READ_SAMPLE_TIMES (5) //define the time interval(in milisecond) between each samples in
12 #include <LiquidCrystal.h>
13
14 const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
15 LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //normal operation
16
17 /*****Application Related Macros*****/
18 #define      GAS_LPG              (0)
19 #define      GAS_CO                (1)
20 #define      GAS_SMOKE            (2)
21
22 /*****Globals*****/
23 float LPGCurve[3] = {2.3,0.21,-0.47}; //two points are taken from the curve.
24 //with these two points, a line is formed which is "approximately equivalent"
25 //to the original curve.
26 //data format:{ x, y, slope}; point1: (lg200, 0.21), point2: (lg10000, -0.59)
27 float COCurve[3] = {2.3,0.72,-0.34}; //two points are taken from the curve.
28 //with these two points, a line is formed which is "approximately equivalent"
29 //to the original curve.
30 //data format:{ x, y, slope}; point1: (lg200, 0.72), point2: (lg10000, 0.15)
31 float SmokeCurve[3] = {2.3,0.53,-0.44}; //two points are taken from the curve.
32 //with these two points, a line is formed which is "approximately equivalent"
33 //to the original curve.
34 //data format:{ x, y, slope}; point1: (lg200, 0.53), point2: (lg10000, -0.22)
35 float Ro = 10; //Ro is initialized to 10 kilo ohms
36
37 void setup()
38 {
39     Serial.begin(9600); //UART setup, baudrate = 9600bps
40     Serial.print("Calibrating...\n");
41     Ro = MQCalibration(MQ_PIN); //Calibrating the sensor. Please make sure the sensor is in clean air
42     lcd.begin(16, 2); //when you perform the calibration
43     Serial.print("Calibration is done...\n");
44     Serial.print("Ro=");
45     Serial.print(Ro);
46     Serial.print("kohm");
47     Serial.print("\n");
48     lcd.print("Calibration is done...\n");
49     lcd.print("Ro=");
50     lcd.print(Ro);
51     lcd.print("kohm");
52     lcd.print("\n");
53 }
```

```

55 void loop()
56 {
57     Serial.print("LPG:");
58     Serial.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_LPG) );
59     Serial.print( "ppm" );
60     Serial.print("    ");
61     Serial.print("CO:");
62     Serial.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_CO) );
63     Serial.print( "ppm" );
64     Serial.print("    ");
65     Serial.print("SMOKE:");
66     Serial.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_SMOKE) );
67     Serial.pr
68     int( "ppm" );
69     Serial.print("\n");
70     lcd.setCursor(0, 0);
71     lcd.print("LPG:");
72     lcd.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_LPG) );
73     //lcd.print( "ppm" );
74     lcd.print("    ");
75     lcd.setCursor(9, 0);
76     lcd.print("CO:");
77     lcd.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_CO) );
78     //lcd.print( "ppm" );
79     lcd.print("    ");
80     lcd.setCursor(0, 1);
81     lcd.print("SMOKE:");
82     lcd.print(MQGetGasPercentage(MQRead(MQ_PIN)/Ro,GAS_SMOKE) );
83     //lcd.print( "ppm" );
84     lcd.print("    ");
85     delay(200);
86 }
87
88 /***** MQResistanceCalculation *****/
89 Input:  raw_adc - raw value read from adc, which represents the voltage
90 Output: the calculated sensor resistance
91 Remarks: The sensor and the load resistor forms a voltage divider. Given the voltage
92          across the load resistor and its resistance, the resistance of the sensor
93          could be derived.
94 *****/
95 float MQResistanceCalculation(int raw_adc)
96 {
97     return ( ((float)RL_VALUE*(1023-raw_adc)/raw_adc));
98 }
99
100 /***** MQCalibration *****/
101 Input:  mq_pin - analog channel
102 Output: Ro of the sensor
103 Remarks: This function assumes that the sensor is in clean air. It use
104          MQResistanceCalculation to calculates the sensor resistance in clean air
105          and then divides it with RO_CLEAN_AIR_FACTOR. RO_CLEAN_AIR_FACTOR is about
106          10, which differs slightly between different sensors.
107 *****/
108 float MQCalibration(int mq_pin)
109 {
110     int i;
111     float val=0;
112
113     for (i=0;i<CALIBARAION_SAMPLE_TIMES;i++) { //take multiple samples
114         val += MQResistanceCalculation(analogRead(mq_pin));
115         delay(CALIBRATION_SAMPLE_INTERVAL);
116     }
117     val = val/CALIBARAION_SAMPLE_TIMES; //calculate the average value
118
119     val = val/RO_CLEAN_AIR_FACTOR; //divided by RO_CLEAN_AIR_FACTOR yields the Ro
120                                     //according to the chart in the datasheet
121
122     return val;
123 }

```



```

125 /***** MQRead *****/
126 Input:  mq_pin - analog channel
127 Output: Rs of the sensor
128 Remarks: This function use MQResistanceCalculation to caculate the sensor resistenc (Rs).
129          The Rs changes as the sensor is in the different consentration of the target
130          gas. The sample times and the time interval between samples could be configured
131          by changing the definition of the macros.
132 *****/
133 float MQRead(int mq_pin)
134 {
135     int i;
136     float rs=0;
137
138     for (i=0;i<READ_SAMPLE_TIMES;i++) {
139         rs += MQResistanceCalculation(analogRead(mq_pin));
140         delay(READ_SAMPLE_INTERVAL);
141     }
142
143     rs = rs/READ_SAMPLE_TIMES;
144
145     return rs;
146 }

148 /***** MQGetGasPercentage *****/
149 Input:  rs_ro_ratio - Rs divided by Ro
150          gas_id      - target gas type
151 Output: ppm of the target gas
152 Remarks: This function passes different curves to the MQGetPercentage function which
153          calculates the ppm (parts per million) of the target gas.
154 *****/
155 int MQGetGasPercentage(float rs_ro_ratio, int gas_id)
156 {
157     if ( gas_id == GAS_LPG ) {
158         return MQGetPercentage(rs_ro_ratio,LPGCurve);
159     } else if ( gas_id == GAS_CO ) {
160         return MQGetPercentage(rs_ro_ratio,COCurve);
161     } else if ( gas_id == GAS_SMOKE ) {
162         return MQGetPercentage(rs_ro_ratio,SmokeCurve);
163     }
164
165     return 0;
166 }

168 /***** MQGetPercentage *****/
169 Input:  rs_ro_ratio - Rs divided by Ro
170          pcurve      - pointer to the curve of the target gas
171 Output: ppm of the target gas
172 Remarks: By using the slope and a point of the line. The x(logarithmic value of ppm)
173          of the line could be derived if y(rs_ro_ratio) is provided. As it is a
174          logarithmic coordinate, power of 10 is used to convert the result to non-logarithmic
175          value.
176 *****/
177 int MQGetPercentage(float rs_ro_ratio, float *pcurve)
178 {
179     return (pow(10,((log(rs_ro_ratio)-pcurve[1])/pcurve[2]) + pcurve[0]]));
180 }

```

3.2.1 MH-Z19

- `pip install mh-z19`

Python'un Adafruit_DHT Kütüphanesi ile digital pinden sıcaklık ve nem değerleri okundu. Kütüphaneyi indirmek için aşağıdaki komut satırını terminalde çalıştırmak yeterlidir.

```
#!/usr/bin/python

#pre requirement
#pip3 install Adafruit_DHT
import Adafruit_DHT

def read_dht(pin=4, sensor = Adafruit_DHT.DHT22):

    while(True):
        humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
        if humidity is not None and temperature is not None:
            print('Temp={0:0.1f}*C Humidity={1:0.1f}%'.format(temperature, humidity))
        else:
            print('Failed to get reading. Try again!')

###Test###
"""
# Sensor should be set to Adafruit_DHT.DHT11,
# Adafruit_DHT.DHT22, or Adafruit_DHT.AM2302.
sensor = Adafruit_DHT.DHT22

# Example using a Beaglebone Black with DHT sensor
# connected to pin P8_11.
pin = '4'

read_dht()
"""
###Test####
```

4.SONUÇ

İki ayrı aşamada ele alınacak projenin ilk kısmı olan bu projede gerekli sensörlerin bulunması, gerekli işlemci ve mikrodenetleyicilerle kullanımı üzerine araştırmaların yapılması ve elde edilen bilgiler doğrultusunda proje için en uygun ekipmanların kullanımı amaçlanmıştır. Dünya genelinde kargo süreçleri ve ekipman temini konusundaki sıkıntılar nedeniyle yurtdışından temin edilmesi planlanan, hassasiyeti daha yüksek sensörlerin kullanımı gerçekleştirilemese de var olan sensörler ile ilk aşama tamamlanmıştır. Böylelikle her bir sensörün seçimi ve bu sensörlerin kullanımı konusundaki gerekli araştırmalar tamamlanmış. İkinci aşama olan bu sensörlerden alınan verilerin bir dosyaya kaydedilip bulut sistemine aktarılması na hazır hale getirilmiştir.