Air-Quality Balloon

Janaki Sheth	(100260005)
Sarthak Bagaria	(100260006)
Abhijeet Mukhekar	(100260009)
Yashasvi Alladi	(100260021)

Project Description:

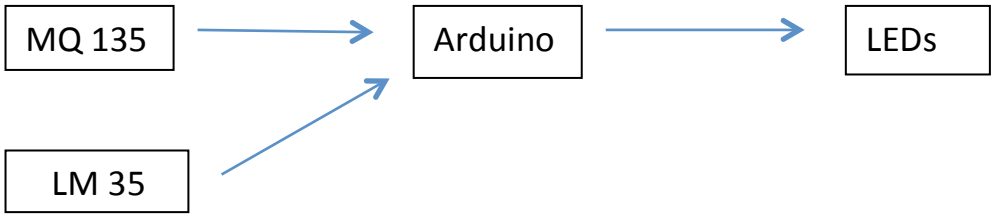
The circuit uses both an air-quality sensor (MQ 135) and a temperature sensor (LM 35) to measure the pollution levels and the temperature levels of the surroundings respectively. The air-quality sensor gives values in voltages, and taking the room value as the base level, we judge the pollution level of the surrounding. Values were taken in the room, on the roadside and at the main gate. They progressively show greater pollution levels. The LEDs display a binary condition of a polluted environment.

Original Plan:

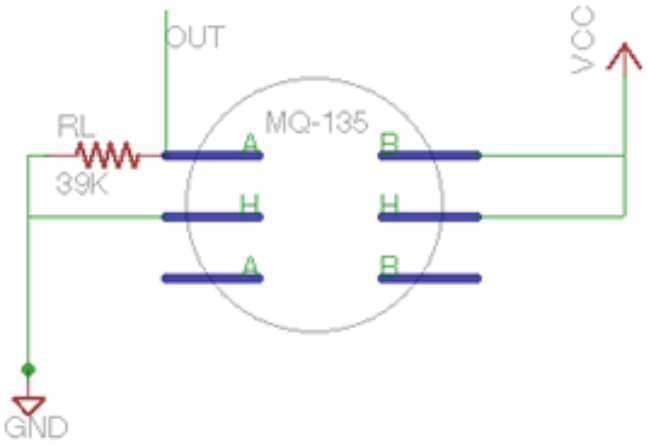
We intended to use AVR Atmega, which could be embedded on a small PCB along with the sensors. This board would have been much more compact, and lighter than Arduino and hence more suitable to be attached to a helium balloon later. For the power supply we planned on using a Li-Po 7V battery along with a 5V voltage regulator, as our sensors and micro-controller work on voltages close to 5V.

What we did:

We used Arduino Micro-controller to interpret the voltage readings from the Pollution and Temperature Sensor, and glow the LEDs according to whether the pollution was higher or lower than a threshold value.



MQ 135 (Pollution Sensor) Circuit:



Sensitivity Characteristics

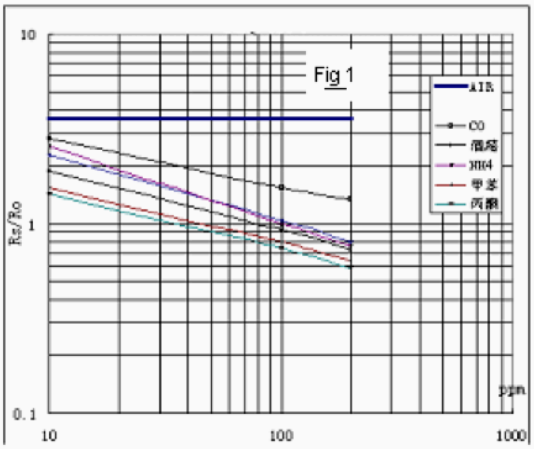


Fig.1 shows the typical sensitivity characteristics of the MQ135, ordinate means resistance ratio of the sensor (Rs/Ro), abscissa is concentration of gases. Rs means resistance in different gases, Ro means resistance of sensor in 100ppm Ammonia. All test are under standard test conditions.

Influence of Temperature/Humidity

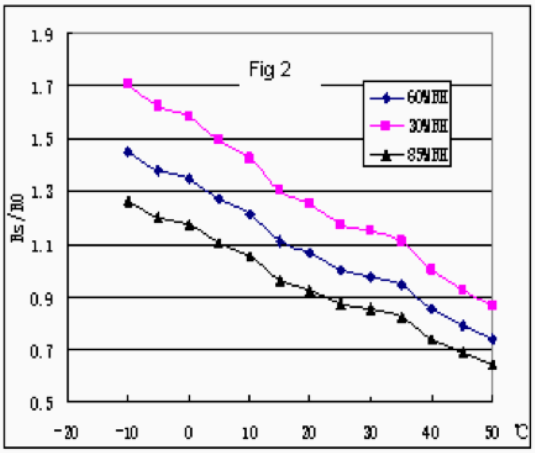


Fig.2 shows the typical temperature and humidity characteristics. Ordinate means resistance ratio of the sensor (Rs/Ro), Rs means resistance of sensor in 100ppm Ammonia under different tem. and humidity. Ro means resistance of the sensor in environment of 100ppm Ammonia, 20℃/65%RH

For testing, Arduino was connected to a computer, which provided power to the Arduino and allowed us to see the voltage readings from the sensor on the Serial Monitor.

Three sets of Measurements were taken:

Main Road	Roadside	Hostel Room
0.03	0.75	1.93
29.79	30.76	30.27
0.08	0.86	1.93
32.23	27.83	29.79
0.14	0.83	1.93
28.81	27.83	28.32
0.11	0.85	1.92
29.30	27.34	30.76
0.28	0.86	1.92
33.20	27.83	30.76
0.61	0.85	1.92
32.71	27.34	31.25
0.83	0.85	1.92
32.71	27.83	31.25
0.76	0.86	1.93
32.23	26.86	31.25
0.83	0.86	1.92
29.30	27.83	30.76
0.60	0.85	1.93
29.79	27.83	31.25
0.42	0.85	1.92
29.30	28.32	30.27
0.32	0.86	1.92
29.79	27.83	31.25
0.24	0.89	1.92
30.27	27.34	30.27
0.19	0.84	1.91
29.30	27.83	31.25
0.15	0.85	1.91
29.30	26.86	30.27
0.12	0.86	1.91
28.81	28.32	31.25
0.10	0.86	1.92
29.30	28.32	30.76
0.07	0.86	1.90
29.30	27.83	30.27
0.07	0.86	1.92
29.30	29.30	31.25
		1.91

The values around 1 are the voltage readings from the pollution sensor.

The values around 30 are the temperature readings.

Accordingly we set our threshold voltage from Air Sensor to 1.5 V.

Arduino Code:

```
int pollutionpin = 3;
int temppin = 5;
float a = 0;
float b = 0;
float pvalue = 0;
float tvalue = 0;
float v[100];                 //this array will store last 100 readings from pollution sensor
int pos = 0;
float mean = 0;               //mean of elements of the array v

void setup() {
   Serial.begin (9600);
   pinMode(5,OUTPUT);
   pinMode(6,OUTPUT);
   for (int i=0; i<100; i++) v[i]=0;
}

void loop() {
   delay(10);
   a = analogRead(pollutionpin);
   pvalue = float(a*5)/float(1024);
   b = analogRead(temppin);
   tvalue = float(b*5)/float(10.24);
   mean = mean - v[pos]/100;                //subtracts the reading taken 100 cycles earlier
                                            from the mean
   v[pos] = pvalue;                         //replaces the oldest value in the array with the
                                            newest
   mean = mean + v[pos]/100;                //adds the new reading to the mean
   pos = (pos+1)%100;                       //moves ahead in the array, which is now the
                                            position of the oldest value in the array

   if (pos == 99){
     Serial.println(tvalue);
```

```
      if (mean<1.5) {
          Serial.println("Air is Polluted");
          digitalWrite(6,HIGH);                       //LED 6 is red (polluted air)
          digitalWrite(5,LOW);                        //LED 5 is green (good air)
      }
      else{
          Serial.println("Air is Good");
          digitalWrite(6,LOW);
          digitalWrite(5,HIGH);
      }
  }
}
```

To go ahead with our plan of using the Atmega, we got access to an AVR Atmega programmer circuit and learned how to program Atmega for our project. We also tested our Atmega code by using an LED to distinguish high voltage inputs from low voltage inputs.

Problems:

Since, the air-quality sensor had to be continuously heated for at-least 24 hours before measurement, it had to be kept connected to the power supply. Abrupt power cuts wasted a lot of heating time.

Arduino board is quite huge to implement the actual balloon idea.

Proposed Work:

Given some more time, we could use the AVR coding that we learnt and implement the entire circuit on a smaller board and use the balloons. What was left was to place an AVR Atmega in-place of Arduino, attach a Li-Po battery with a voltage regulator to power the circuit and solder it onto a small circuit board.

Who did What:

Janaki Sheth – Project planning, coding and sensor data collection/testing

Sarthak Bagaria – Project planning, circuit designing/building and coding

Abhijeet Mukhekar – Sensor data collection/testing

Yashasvi Alladi (joined late) – Circuit building and soldering (could not be done for shortage of time)