**PROJECT REPORT ON AIR QUALITY MONITORING**

**INTRODUCTION:**

As our project is based on IOT, let us throw some light on the topic of IOT itself. The Internet of Things (IOT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals, or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human-to-computer interaction.

**APPLICATION OF IOT:**

A growing portion of IOT devices are created for consumer use, including connected vehicles, home automation, wearable technology, connected health, and appliances with remote monitoring capabilities.

In consumer market, IOT technology is most synonymous with products pertaining to the concept of the “smart home”

Therefore, using IOT, the aim of this project is to monitor the air quality level in the surrounding of our device using Arduino and MQ135 gas sensor and show the results in PPM units.

**Problem Statement**

As urbanization causes the growth of suburban communities, the existing dependent on fossil fuels must expand. An increase in traffic-related pollutant emissions. According to science, the six common air pollutants are particulate matter, ground-level zone, carbon monoxides, and lead. These are called the criteria pollutants and thus are requited to be measured to tell us how healthy the air is to breathe. Vehicular emissions contribute carbon monoxide, carbon dioxide, and nitrogen oxides to air pollution.

The current pollution measurement methodology uses expensive equipment at fixed locations or dedicated mobile equipment. The raw data obtained in this manner is used to further extrapolate the extent and concentration of pollution through dispersion models. This is a coarse-grained system where the pollution measurements are few and far in-between. Widespread deployment of this measurement paradigm is constrained by its prohibitive cost. In addition, it is desirable to have access to real-time measurements to be able to quickly analyses and identify alarming levels of pollutants. Currently, access to such data is limited if not absent. It is available to and discern able by only a few who are well informed on the subject of pollution.

As opposed to a coarse-grained sensing system, a fine-grained approach would provide more frequent and spatially dense pollutant measurements. A scalable sensing platform could effectively disseminate pollution information to users in need. Today, the scarcity of fine-grained air quality information is hindering public awareness of health issues arising from pollution. Studies suggest that the health effects among asthmatics from short-term changes in air pollution levels are an important public health problem. We anticipate that, with the advised to take actions based on real-time pollution levels to accommodate individual health needs.

The availability of real-time air quality data could make drivers better educated about driving patterns and how it impacts the environment and increases pollution. Better driving habits will lead to reduced pollution. Also, more health-conscious citizens may choose alternate “healthy” routes based on pollution information. It will benefit them as well as others by reducing pollution concentration in peak roadways so everybody breathes cleaner air.

At the same time, the emergence of cheap commodity air pollution sensors and the increase of cellular bandwidth have made mobile sensing platforms capable of real-time air quality data collection increasingly feasible.

Several manufactures such as Auroral or Variable Technologies have recently introduced handheld pollution measurement devices. These devices are small enough to be carried by walking people for personal use and measure all the criteria pollutants contributed by vehicle emissions. But none of these off-the -shelf devices have been evaluated with respect to their real-time sensing performance when installed on mobile platforms such as vehicles. To the best of our knowledge, we have not come across any work that studies the long-term stability, reliability, and impact of real-time pollution monitoring systems using commodity sensors and the problems associated with deploying such systems.

**2.Proposed system:**

Now in this project, we are using the locally available gas sensors for observing polluted gases like Carbon monoxide (CO), Carbon dioxide (CO2), and parameters like temperature, humidity. By using this method people can view the level of pollution through a wireless system. It reduced cost, is reliable, and is comfortable for any place where we are monitoring the gases.

**Goal and Objectives**

* Quality of air can be checked indoors as well as outdoor.
* Detecting a wide range of physical parameters.
* Indoor air quality monitoring.
* Industrial perimeter monitoring.
* Roadside pollution monitoring.
* To make this data available to the common man.

**Project Resources**

HARDWARE REQUIRMENTS:

* Air Quality sensor (MQ 135)
* Potentiometer
* 16x2 LCD panel
* Arduino Uno
* Wires

SOFTWARE REQUIRMENTS:

* Arduino (Version 1.8.2)
* THINGSPEAK website

**SYSTEM ANAYSIS AND DESIGN**

**COMPONENT DESCRIPTION:**

**Air Quality Sensors (MQ 135):**

Product Description:

Air quality click is suitable for detecting ammonia (NH3), nitrogen oxides (NOx) benzene, smoke, CO2, and other harmful or poisonous gases that impact air quality. The MQ-135

Sensor unit has a sensor layer made of tin dioxide (SnO2), an inorganic compound that has lower conductivity in clean air than when polluting gases are present. To calibrate Air quality, use the onboard potentiometer to adjust the load resistance on the sensor circuit.

Pin Description:

* the VDD power supply 5V DC
* GND used to connect the module to system ground
* DIGITAL OUT, you can also use this pin, by setting a threshold value using the potentiometer

ANALOG OUT, this pin outputs 0-5Vanalog voltage based on the intensity of this gas.



**Potentiometer:**

Product Description:

A potentiometer is a three-terminal resistor with a sliding or rotating contact that forms an

Adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat. The measuring instrument called a potentiometer is essentially a voltage divider used for measuring electric potential (voltage);

The component is an implementation of the same principle, hence is name.



**16x2 LCD Panel:**

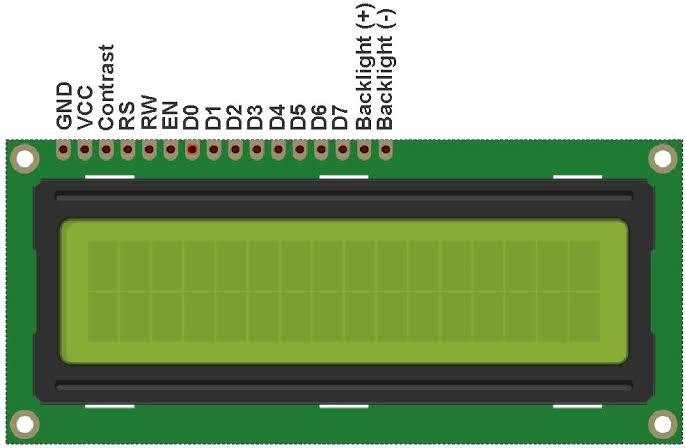
Product Description:

A liquid-crystal display (LCD) is a flat-panel display or another electronically modulated optical device that uses the light-modulating properties of liquid crystals. Liquid crystals do not emit [1]

Light directly, instead of using a backlight or reflector to produce images in colour or monochrome. LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayed or hidden, such as preset words, digits, and seven-segment displays

Pin Description:

* Connect pin1 (VEE) to the ground.
* Connect pin2 (VDD or VCC) to the 5V.
* Connect pin3 (V0) tot eh middle pin of the 10K potentiometer and connect the other two ends of the potentiometer is used to control the screen contrast of the LCD. A potentiometer of values other than 10K will work too.
* Connect pin4 (RS) to pin12 of the Arduino.
* Connect pin6 (E) to pin11 of the Arduino. The RS and E pin are the control pins that are used to send to send data and characters.
* The following four pins are data pins that are used to communicate with the Arduino.
* Connect pin11 (D4) to pin5 of Arduino.
* Connect pin12 (D5) to pin4 of Arduino.
* Connect pin13 (D6) to pin3 of Arduino.
* Connect pin14 (D7) to pin 2 of Arduino.
* Connect pin15 to the VCC through the 200-ohm resistor. The resistor will be used to set the backlight brightness. Larger values will make the backlight much darker.
* Connect pin16 to the Ground.



**Arduino Uno:**

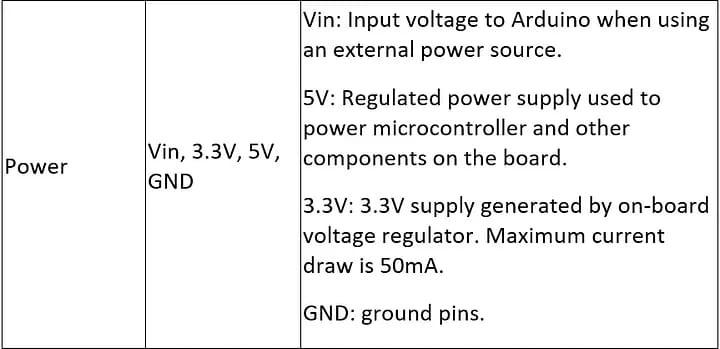
Product Description:

Arduino is an open-source computer hardware and software company, project, and user community that designs and manufactures single-board microcontrollers kills for building digital devices and interactive objects that can sense and control objects in the physical world.

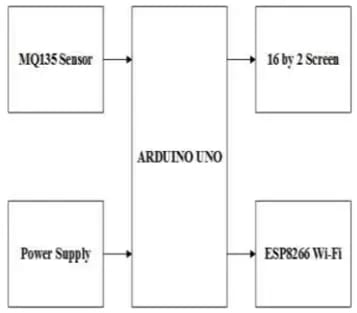


Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Series Bus (UDB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the processing language project.

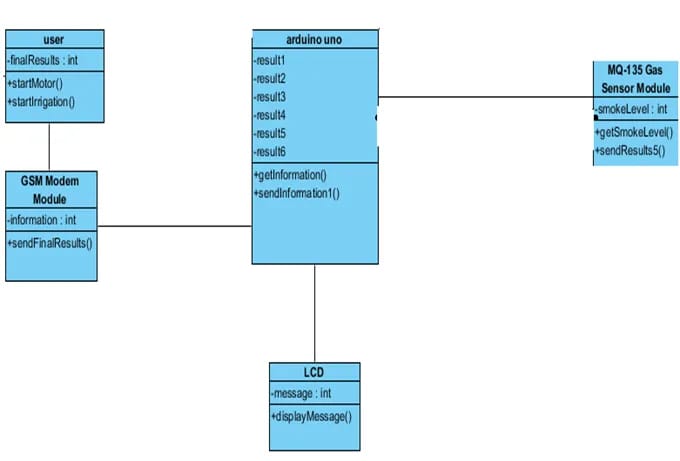
**Pin diagram: Pin Category**



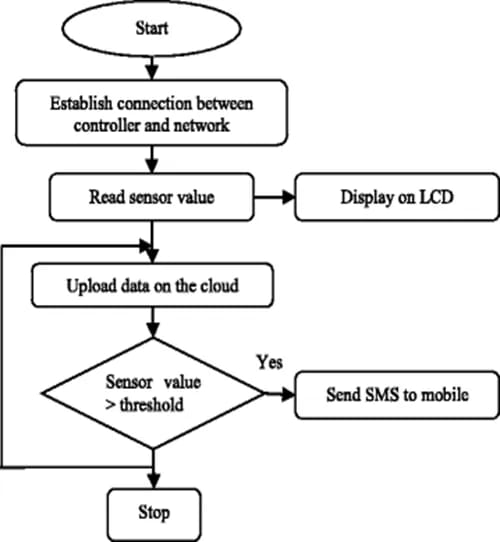
**Design diagrams/UML diagrams/Flow Charts/E-R diagrams**

****

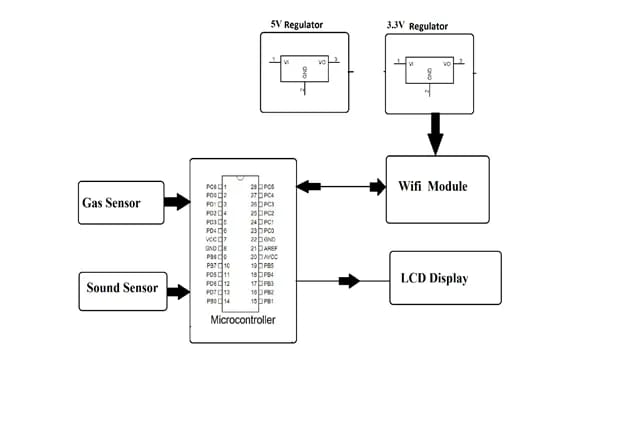
**Use case diagram**

****

**Class diagram**

****

**Activity diagram**

****

**Sequence diagram**

1. **Algorithms/pseudo code**

**Code Explanation:**

* Before beginning the coding for this project, we need to first Calibrate the MQ135 Gas sensor. There are lots of calculations involved in converting the outputs of the sensor into a PPM value. Using this library, you can directly get the PPM values, by just using the below two lines:

MQ135 gasSensor = MQ135(A0);

float air quality =

gassensors.getPPM();

* But before that, we need to **Calibrate the MQ135 sensors**, for calibrating the sensor upload the below-given code and let it run for 12 to 24 hours, and then get the RZERO value.

#include "MQ135.h"

void setup (){

Serial.begin (9600);

}

void loop() {

MQ135 gasSensor = MQ135(A0); // Attach sensor to pin A0

float rzero = gasSensor.getRZero();

Serial.println (rzero);

delay (1000);

}

After getting the RZERO value. Put the RZERO value in the library file you downloaded “MQ135.h”: #define RZERO 494.63

Now we can begin the actual code for our Air quality monitoring project.

In the code, first of all, we have defined the libraries and the variables for the Gas sensor and the LCD. By using the Software Serial Library, we can make any digital pin as TX and RX pin. In this code, we have made Pin 9 as the RX pin and pin 10 as the TX pin for the ESP8266. Then we have included the library for the LCD and have defined the pins for the same. We have also defined two more variables: one for the sensor analog pin and the other for storing air quality value.

#include <SoftwareSerial.h>

#define DEBUG true

SoftwareSerial esp8266(9,10);

#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11, 5, 4, 3, 2);

const int sensorPin= 0;

int air\_quality;

Then we will declare pin 8 as the output pin where we have connected the buzzer. lcd.begin(16,2) command will start the LCD to receive data and then we will set the cursor to first-line and will print the ‘ ’. Then we will set the cursor on the second line and will print ‘Sensor Warming’.

pinMode(8, OUTPUT);

lcd.begin(16,2);

lcd.setCursor (0,0);

lcd.print (" ");

lcd.setCursor (0,1);

lcd.print ("Sensor Warming ");

delay(1000);

Then we will set the baud rate for the serial communication. Different ESP’s have different baud rates so write it according to your ESP’s baud rate. Then we will send the commands to set the ESP to communicate with the Arduino and show the IP address on the serial monitor.

Serial.begin(115200);

esp8266.begin(115200);

sendData("AT+RST\r\n",2000,DEBUG);

sendData("AT+CWMODE=2\r\n",1000,DEBUG);

sendData("AT+CIFSR\r\n",1000,DEBUG);

sendData("AT+CIPMUair\_quality=1\r\n",1000,DEBUG);

sendData("AT+CIPSERVER=1,80\r\n",1000,DEBUG);

pinMode(sensorPin, INPUT);

lcd.clear();

The following code will print the data on the LCD. We have applied various conditions for checking air quality, and LCD will print the messages according to conditions and the buzzer will also beep if the pollution goes beyond 1000 PPM.

lcd.setCursor (0, 0);

lcd.print ("Air Quality is ");

lcd.print (air\_quality);

lcd.print (" PPM ");

lcd.setCursor (0,1);

if (air\_quality<=1000)

{

lcd.print("Fresh Air");

digitalWrite(8, LOW);

**Conclusion:**

The smart way to monitor environment and air as well as sound pollution being a low

cost but efficient and embedded system is presented in this paper. In the proposed architecture functions of different sensors and their working procedure were discussed. How they work, their functionality, their optimal uses and their data taking procedures and comparison with standard base data’s are also discussed here. The noise and air pollution monitoring system was tested for monitoring the gas levels on different parts of the country. It also sent the sensor parameters to the data server. Our project device showed that it is effective and cheap and with some highly working sensors it can really be a reliable one to everybody and its data’s will be a key to take some necessary steps for the betterment of the society as it will help to identify the affected area so that we can take early steps to reduce damages for the next generation.