GOVERNMENT COLLEGE OF ENGINEERINGERODE

அரசினர் பொறியியல் கல்லூரி, ஈரோடு
Government College of Engineering, Erode
(Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai)

B.E Electronics and Communication Engineering

NOISE POLLUTION MONITORING

Name of the Student: Sahaana Priya C

Naan Mudhalvan Register no: au731121106040

Under the mentor of

Dr. M. Sathyakala

Assistant professor, Department of IT

Department of Electronics and Communication Engineering

Government College of Engineering

Erode, PO, near Vasavi College, TamilNadu-

638316,Affiliated to Anna University,

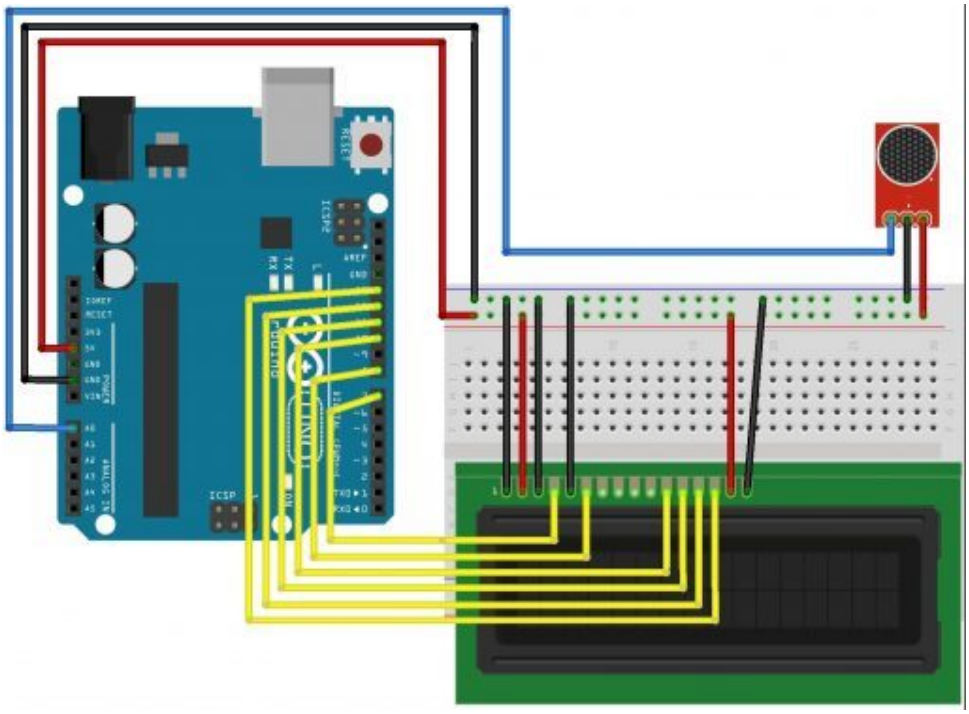
Chennai.

INTRODUCTION:

Noise pollution is the propagation of noise with ranging impacts on theactivity of human or animal life, most of which are harmful to a degree. Large amount of increasing noise pollution has made human life proneto large number of diseases. Therefore, it has now become necessary tocontrol the pollution to ensure healthy livelihood and better future.

HARDWARE APPROACH:

CIRCUIT DIAGRAM:



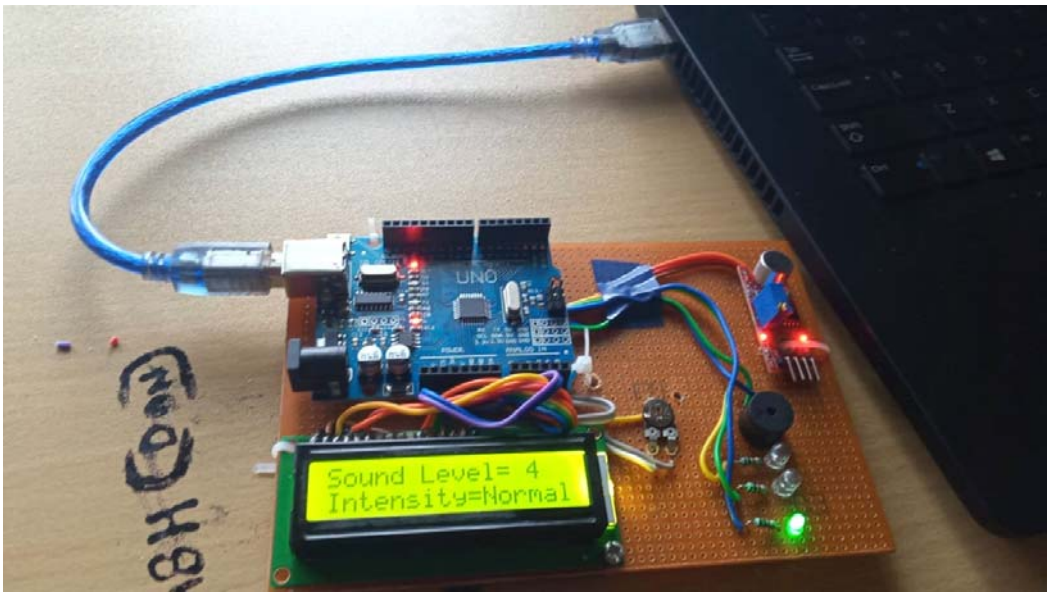
PROGRAM:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(7,8,10,11,12,13);
int num_Measure = 128 ; // Set the number of measurements
int pinSignal = A0; // pin connected to pin O module sound sensor
int greenLed = 4;
int blueLed = 5;
int redLed =  6;
int buzzer = 3;
long Sound_signal;    // Store the value read Sound Sensor
long sum = 0 ; // Store the total value of n measurements
```

```
long level = 0 ; // Store the average value
int soundlow = 40;
int soundmedium = 200;

void setup ()
{
 pinMode (pinSignal, INPUT); // Set the signal pin as input
 pinMode (greenLed,OUTPUT);
 pinMode (blueLed,OUTPUT);
 pinMode (redLed,OUTPUT);
 pinMode (buzzer,OUTPUT);
 Serial.begin (9600);
 lcd.begin(16,2);
 lcd.print("Noise Detector");
 delay(1000);
}

void loop ()
{
 // Performs 128 signal readings
 for ( int i = 0 ; i <num_Measure; i ++)
 {
  Sound_signal = analogRead (pinSignal);
   sum =sum + Sound_signal;
 }

 level = sum / num_Measure; // Calculate the average value
 Serial.print("Sound Level: ");
 lcd.print("Sound Level= ");
 Serial.println (level-33);
 lcd.print(level-33);
 if(level-33<soundlow)
 {
  lcd.setCursor(0,2);
  lcd.print("Intensity=Normal");
   digitalWrite(greenLed,HIGH);
    digitalWrite(blueLed,LOW);
     digitalWrite(redLed,LOW);
      digitalWrite(buzzer, LOW);
     delay(500);
 }
 if(level-33>soundlow && level-33<soundmedium)
 {
  lcd.setCursor(0,2);
  lcd.print("Intensity=Medium");
   digitalWrite(blueLed,HIGH);
    digitalWrite(redLed,LOW);
```

```
      digitalWrite(greenLed,LOW);
       digitalWrite(buzzer, LOW);
      delay(500);
  }
  if(level-33>soundmedium)
  {
    lcd.setCursor(0,2);
    lcd.print("Intensity= High");
    digitalWrite(redLed,HIGH);
     digitalWrite(greenLed,LOW);
     digitalWrite(blueLed,LOW);
     digitalWrite(buzzer, HIGH);
  }
  sum = 0 ; // Reset the sum of the measurement values
  delay(200);
  lcd.clear();
}
```

IMPLEMENTATION:

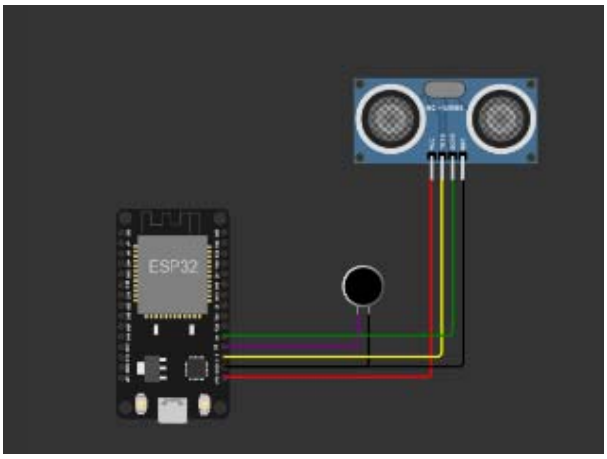


CONDITION:

- If sound level is less than 40dB, Green LED turns on, which denotes normal sound intensity.
- If sound level is in between 40-200dB, Blue LED turns on, which denotes medium noise intensity.
- If sound level is >200dB, Red LED turns on, which indicates very high level of noise.

SOFTWARE APPROACH:

CIRCUIT DIAGRAM:



PROGRAM:

```
import machine
import time
import urequests
import ujson
import network
import math

# Define your Wi-Fi credentials
wifi_ssid = 'Wokwi-GUEST'
wifi_password = ''  # Replace with the actual Wi-Fi password

# Connect to Wi-Fi
wifi = network.WLAN(network.STA_IF)
wifi.active(True)
wifi.connect(wifi_ssid, wifi_password)

# Wait for Wi-Fi connection
while not wifi.isconnected():
    pass

# Define ultrasonic sensor pins (Trig and Echo pins)
ultrasonic_trig = machine.Pin(15, machine.Pin.OUT)
ultrasonic_echo = machine.Pin(4, machine.Pin.IN)

# Define microphone pin
microphone = machine.ADC(2)
```

```
calibration_constant = 2.0
noise_threshold = 60  # Set your desired noise threshold in dB

# Firebase Realtime Database URL and secret
firebase_url = 'https://noise-pollution-monitori-9196a-default-rtdb.asia-
southeast1.firebasedatabase.app/'
firebase_secret = 'aANDKcKQEK3ky7G38Uuq5WDdvPP5OUmtBxsSMN2C'

def measure_distance():
    # Trigger the ultrasonic sensor
    ultrasonic_trig.value(1)
    time.sleep_us(10)
    ultrasonic_trig.value(0)

    # Measure the pulse width of the echo signal
    pulse_time = machine.time_pulse_us(ultrasonic_echo, 1, 30000)

    # Calculate distance in centimeters
    distance_cm = (pulse_time / 2) / 29.1
    return distance_cm

def measure_noise_level():
    # Read analog value from the microphone
    noise_level = microphone.read()

    noise_level_db = 20 * math.log10(noise_level / calibration_constant)
    return noise_level, noise_level_db

# Function to send data to Firebase
def send_data_to_firebase(distance, noise_level_db):
    data = {
        "Distance": distance,
        "NoiseLevelDB": noise_level_db
    }
    url = f'{firebase_url}/sensor_data.json?auth={firebase_secret}'

    try:
        response = urequests.patch(url, json=data)  # Use 'patch' instead of 'put'
        if response.status_code == 200:
            print("Data sent to Firebase")
        else:
            print(f"Failed to send data to Firebase. Status code:
{response.status_code}")
    except Exception as e:
        print(f"Error sending data to Firebase: {str(e)}")
```

```
try:
    while True:
        distance = measure_distance()
        noise_level, noise_level_db = measure_noise_level()

        print("Distance: {} cm, Noise Level: {:.2f} dB".format(distance,
noise_level_db))

        if noise_level_db > noise_threshold:
            print("Warning: Noise pollution exceeds threshold!")

        # Send data to Firebase
        send_data_to_firebase(distance, noise_level_db)

        time.sleep(1)  # Adjust the sleep duration as needed

except KeyboardInterrupt:
  print("Monitoring stopped")
```

WORKING:

The circuit is simulated using wokwi. The real time data base is created in firebase whose URL and secret ID is added to this program. Real time noise data measured using sensor is sent to the firebase for data storage.

SIMULATED OUTPUT:

CONCLUSION:

A low-budget smart sensor unit for environmental noise level measurement is designed using IoT. I ensure that my project will perform efficiently. With some advanced developments in this project, we can create smart cities and a healthy environment.