**Version 2**

(Smart city)

1. **Gas sensor – (mq-135)**



"C:\Users\User\Desktop\data science\MQ135 (Ver1.4) - Manual.pdf"



**Description:**

Gas sensors are devices that help us understand the amount of gas in the environment and the natural state of its movement. Gas sensors reveal the amount of gas in the environment and the nature of the gas composition with electrical signals and can provide its change [91–93].

**Test code :**

#define MQ135\_PIN 36 // Analog pin connected to the MQ-135 sensor

void setup() {

Serial.begin(9600); // Initialize serial communication

}

void loop() {

int sensorValue = analogRead(MQ135\_PIN); // Read analog value from sensor

Serial.print("MQ-135 Sensor Value: ");

Serial.println(sensorValue); // Print sensor value to serial monitor

}

**Code explanation:**

This line defines a constant named MQ135\_PIN and assigns it the value 36, which represents the analog pin on the Arduino board to which the MQ-135 sensor is connected.

The setup() function is called once when the Arduino board is powered on or reset.

Serial.begin(9600) initializes serial communication with a baud rate of 9600 bits per second. This allows the Arduino to communicate with a connected computer via the USB cable.

The loop() function is executed continuously after the setup() function.

analogRead(MQ135\_PIN) reads the analog voltage from the MQ-135 sensor connected to the pin specified by MQ135\_PIN. This value is converted to a digital value ranging from 0 to 1023.

The obtained sensor value is then printed to the serial monitor using Serial. println(). This allows you to observe the sensor readings in real-time on your computer.

**Issues we faced:** the sensors getting some random values while not connecting the sensor

**Solution:** added 2.2k resistor (pull-down resistor) power to ground

1. **PIR sensor – (HC-SR501)**

****

"C:\Users\User\Desktop\data science\HC-SR501-pirsensor.pdf"

****

**Description:**

PIR (passive infrared) sensors utilise the detection of infrared that is radiated from all objects that emit heat. This type of emission is not visible to the human eye, but sensors that operate using infrared wavelengths can detect such activity.

**Test code:**

#define PIR\_PIN 35 // Digital pin connected to the PIR sensor

void setup() {

Serial.begin(9600); // Initialize serial communication

pinMode(PIR\_PIN, INPUT); // Set PIR pin as input

}

void loop() {

int motionDetected = digitalRead(PIR\_PIN); // Read digital value from PIR sensor

if (motionDetected == HIGH) {

Serial.println("Motion Detected!");

} else {

Serial.println("No Motion Detected");

}

delay(1000); // Delay for stability

}

**Explanation of the code:**

#define PIR\_PIN 35

This line defines a constant named PIR\_PIN and assigns it the value 35, representing the digital pin on the Arduino board to which the PIR sensor is connected.

The setup() function is called once when the Arduino board is powered on or reset.

Serial.begin(9600) initializes serial communication with a baud rate of 9600 bits per second.

pinMode(PIR\_PIN, INPUT) configures the PIR pin as an input, indicating that it will be used to receive digital signals from the PIR sensor.

The loop() function is executed continuously after the setup() function.

digitalRead(PIR\_PIN) reads the digital state (HIGH or LOW) of the PIR sensor connected to the pin specified by PIR\_PIN.

If motion is detected (the digital value is HIGH), it prints "Motion Detected!" to the serial monitor; otherwise, it prints "No Motion Detected".

**Issues we faced:** the sensors getting some random values while not connecting the sensor

**Solution:** added 2.2k resistor (pull-down resistor) power to ground

1. **Vibration sensor – (sw-420):**

****

"C:\Users\User\Desktop\data science\vibrationSENSOR SW420 MODULE LM393\_Datasheet..pdf"



**Description:**

Vibration sensors are devices that detect vibration, shock, and sound. They can be used in machinery to detect problems before they happen. Vibration sensors work by detecting the motion of a material or object by sensing its frequency. The faster the movement, the higher the frequency detected on a vibration sensor.

**Test code:**

#define VIBRATION\_SENSOR\_PIN 34 // Analog pin connected to the vibration sensor

void setup() {

Serial.begin(9600); // Initialize serial communication

}

void loop() {

int sensorValue = analogRead(VIBRATION\_SENSOR\_PIN); // Read analog value from vibration sensor

Serial.print("Vibration Sensor Value: ");

Serial.println(sensorValue); // Print sensor value to serial monitor

delay(1000); // Delay for stability

}

**Explanation of the code:**

#define VIBRATION\_SENSOR\_PIN 34 // Analog pin connected to the vibration sensor

This line defines a constant named VIBRATION\_SENSOR\_PIN and assigns it the value A0, representing the analog pin on the Arduino board to which the vibration sensor is connected.

void setup() {

Serial.begin(9600); // Initialize serial communication

}

The setup() function is called once when the Arduino board is powered on or reset.

Serial.begin(9600) initializes serial communication with a baud rate of 9600 bits per second.

The loop() function is executed continuously after the setup() function.

analogRead(VIBRATION\_SENSOR\_PIN) reads the analog voltage from the vibration sensor connected to the pin specified by VIBRATION\_SENSOR\_PIN. This value is converted to a digital value ranging from 0 to 4095.

The obtained sensor value is then printed to the serial monitor using Serial.println(). This allows you to observe the vibration sensor readings in real-time on your computer.

**Issues we faced:** the sensors getting some random values while not connecting the sensor

**Solution:** added 2.2k resistor (pull-down resistor) power to ground

1. **Flame sensor**



"C:\Users\User\Desktop\data science\A132002\_Flame sensor Module LM393\_Datasheet.pdf"



**Description:**

A flame detector is a type of sensor that can detect and respond to the presence of a flame. These detectors have the ability to identify smokeless liquid and smoke that can create open fire. For example, in boiler furnaces flame detectors are widely used, as a flame detector can detect heat, smoke, and fire.

**Test code:**

#define FLAME\_SENSOR\_PIN 27 // Analog pin connected to the flame sensor

void setup() {

Serial.begin(9600); // Initialize serial communication

}

void loop() {

int sensorValue = analogRead(FLAME\_SENSOR\_PIN); // Read analog value from flame sensor

Serial.print("Flame Sensor Value: ");

Serial.println(sensorValue); // Print sensor value to serial monitor

delay(1000); // Delay for stability

}

**Explanation of the code:**

#define FLAME\_SENSOR\_PIN 27 // Analog pin connected to the flame sensor

This line defines a constant named FLAME\_SENSOR\_PIN and assigns it the value A0, representing the analog pin on the Arduino board to which the flame sensor is connected.

The setup() function is called once when the Arduino board is powered on or reset.

Serial.begin(9600) initializes serial communication with a baud rate of 9600 bits per second.

The loop() function is executed continuously after the setup() function.

analogRead(FLAME\_SENSOR\_PIN) reads the analog voltage from the flame sensor connected to the pin specified by FLAME\_SENSOR\_PIN. This value is converted to a digital value ranging from 0 to 4095.

The obtained sensor value is then printed to the serial monitor using Serial.println(). This allows you to observe the flame sensor readings in real-time on your computer.

**Firmware for ( smart city)**

****

**Code explanation:**

**Sensor Pin Definitions:**

const int gasSensorPin = 36;: Defines the analog pin for the gas sensor.

const int pirSensorPin = 35;: Defines the analog pin for the PIR sensor.

const int vibrationSensorPin = 34;: Defines the analog pin for the vibration sensor.

const int flame\_PIN = 27;: Defines the digital pin for the flame sensor.

**Setup Function:**

Initializes serial communication at a baud rate of 9600.

Sets the sensor pins as input pins.

Prints messages indicating the beginning of sensor tests.

**Loop Function:**

Calls functions to read values from each sensor in a loop with a delay of 100 milliseconds.

Sensor Reading Functions:

gassensor(): Reads analog value from the gas sensor pin, calculates the gas ratio, and prints it as parts per million (ppm).

pir\_sensor(): Reads analog value from the PIR sensor pin, maps it to a range from 0 to 100, and prints it as a PIR value.

vibration\_sensor(): Reads the analog value from the vibration sensor pin, maps it to a range from 0 to 10, and prints it as a vibration magnitude.

**Issues we faced:** the sensors getting some random values while not connecting the sensor

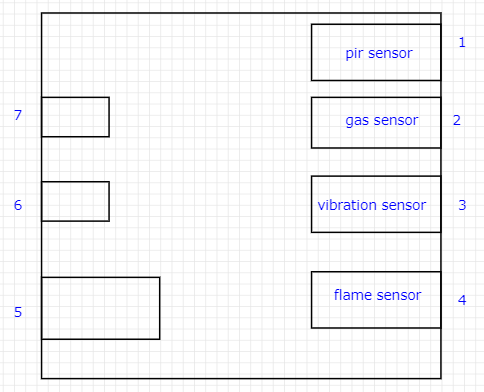
**Solution:** added 2.2k resistor (pull-down resistor) power to ground

**Board modification:**

All the sensors need a 5-volt supply, so we removed the esp32 3.3v pin.

Board, we removed the 3.3 v regulator shorted with 2 legs and removed sensor side components

**Sensor connection:**



How to connect sensors