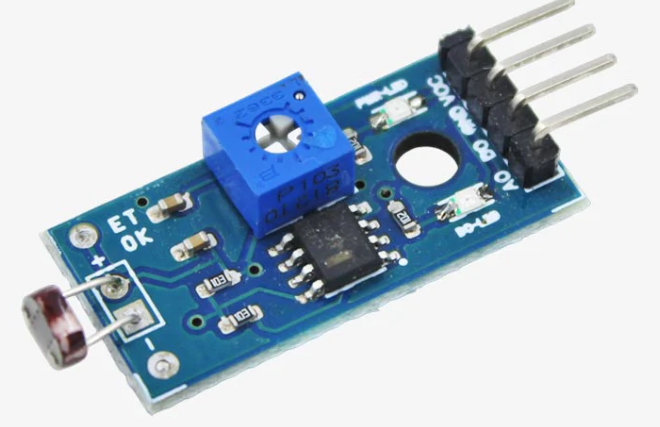
**Version 3**

(good health and well-being)

1. Ldr sensor:





**Description:**

An LDR is a resistor whose resistance changes as the amount of light falling on it changes. The resistance of the LDR decreases with an increase in light intensity, and vice-versa.

**Test code:**

const int ldrPin = A0; // LDR connected to analog pin A0

void setup() {

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

}

void loop() {

int sensorValue = analogRead(ldrPin); // Read LDR sensor value

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

Serial.println(sensorValue); // Print the sensor value

delay(1000); // Wait for 1 second before reading again

}

**Explanation of the code:**

const int ldrPin = A0;: This line defines a constant variable ldrPin and assigns it to the analog pin A0 of the Arduino board where the LDR sensor is connected.

void setup() { ... }: The setup() function is called once when the Arduino is powered on or reset. In this function, we initialize serial communication with a baud rate of 9600 bits per second using Serial.begin(9600);.

void loop() { ... }: The loop() function is executed repeatedly as long as the Arduino is powered on. In this function:

int sensorValue = analogRead(ldrPin);: This line reads the analog value from the LDR sensor connected to pin A0 and stores it in the variable sensorValue.

Serial.print("LDR Sensor Value: ");: This line prints the text "LDR Sensor Value: " to the serial monitor.

Serial.println(sensorValue);: This line prints the actual sensor value to the serial monitor followed by a newline character, so each reading appears on a new line.

delay(1000);: This line adds a delay of 1000 milliseconds (1 second) before the next iteration of the loop.

**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. **Gas sensor:**





**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.

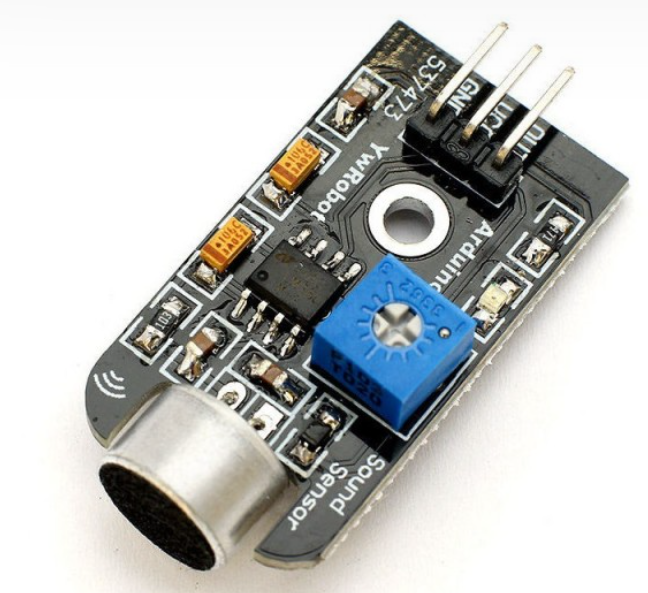
analogRead(MQ135\_PIN) reads the analog voltage from the MQ-135 sensor connected to the pin specified by MQ135\_PIN. This value is converted to a digital value ranging from 0 to 1023.

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. **sound sensor:**

****



**Description:**

A sound sensor is defined as a module that detects sound waves through its intensity and converting it to electrical signals. sound sensor consists of an in-built capacitive microphone, peak detector and an amplifier (LM386, LM393, etc.) that's highly sensitive to sound.

**Test code:**

const int soundPin = A0; // Sound sensor connected to analog pin A0

void setup() {

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

}

void loop() {

int sensorValue = analogRead(soundPin); // Read sound sensor value

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

Serial.println(sensorValue); // Print the sensor value

delay(1000); // Wait for 1 second before reading again

}

**Explanation of the code:**

const int soundPin = A0;: This line defines a constant variable soundPin and assigns it to the analog pin A0 of the Arduino board where the sound sensor is connected.

void setup() { ... }: The setup() function is called once when the Arduino is powered on or reset. In this function, we initialize serial communication with a baud rate of 9600 bits per second using Serial.begin(9600);.

void loop() { ... }: The loop() function is executed repeatedly as long as the Arduino is powered on. In this function:

int sensorValue = analogRead(soundPin);: This line reads the analog value from the sound sensor connected to pin A0 and stores it in the variable sensorValue.

Serial.print("Sound Sensor Value: ");: This line prints the text "Sound Sensor Value: " to the serial monitor.

Serial.println(sensorValue);: This line prints the actual sensor value to the serial monitor followed by a newline character, so each reading appears on a new line.

delay(1000);: This line adds a delay of 1000 milliseconds (1 second) before the next iteration of the loop.

**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. **heart rate sensor:**

****



**Description:**

These devices use electrical detection to track your heart rate. They detect electrical activity through a band that wraps around your chest. For most of these devices to work as designed, the band must be wet, or you need to use a conductive gel where the sensors touch your skin.

**Test code:**

#include <Wire.h>

#include "MAX30105.h"

#include "heartRate.h"

MAX30105 particleSensor;

const byte RATE\_SIZE = 4; //Increase this for more averaging. 4 is good.

byte rates[RATE\_SIZE]; //Array of heart rates

byte rateSpot = 0;

long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;

int beatAvg;

void setup()

{

Serial.begin(115200);

Serial.println("Initializing...");

// Initialize sensor

if (!particleSensor.begin(Wire, I2C\_SPEED\_FAST)) //Use default I2C port, 400kHz speed

{

Serial.println("MAX30105 was not found. Please check wiring/power. ");

while (1);

}

Serial.println("Place your index finger on the sensor with steady pressure.");

particleSensor.setup(); //Configure sensor with default settings

particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running

particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED

}

void loop()

{

long irValue = particleSensor.getIR();

if (checkForBeat(irValue) == true)

{

//We sensed a beat!

long delta = millis() - lastBeat;

lastBeat = millis();

beatsPerMinute = 60 / (delta / 1000.0);

if (beatsPerMinute < 255 && beatsPerMinute > 20)

{

rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array

rateSpot %= RATE\_SIZE; //Wrap variable

//Take average of readings

beatAvg = 0;

for (byte x = 0 ; x < RATE\_SIZE ; x++)

beatAvg += rates[x];

beatAvg /= RATE\_SIZE;

}

}

Serial.print("IR=");

Serial.print(irValue);

Serial.print(", BPM=");

Serial.print(beatsPerMinute);

Serial.print(", Avg BPM=");

Serial.print(beatAvg);

if (irValue < 50000)

Serial.print(" No finger?");

Serial.println();

}

1. **speed sensor**





**Description:**

The Infrared Speed Sensor Module is an IR counter that has an IR transmitter and receiver. If any obstacle is placed between these sensors, a signal is sent to the microcontroller. The module can be used in association with a microcontroller for motor speed detection, pulse count, position limit

**Test code:**

#define sensorPin1 2

#define sensorPin2 23

const float dist = 0.3;

bool sensor1State = false;

bool sensor2State = false;

unsigned long interval = 0;

unsigned long startTime = 0;

double speed = 0;

const int timeout = 3000;

void setup() {

pinMode(sensorPin1, INPUT);

pinMode(sensorPin2, INPUT);

Serial.begin(9600);

}

void loop() {

sensor1State = digitalRead(sensorPin1);

sensor2State = digitalRead(sensorPin2);

startTime = 0;

if (sensor1State==0) {

startTime = millis();

while (digitalRead(sensorPin2)) {

// Serial.print(sensor2State);

if (millis() - startTime > timeout) {

Serial.println("Timeout");

return;

}

delay(3);

}

interval = millis() - startTime;

if (interval > 3) {

speed = ((dist) / (interval / 1000.0)) \* (18 / 5.0);

Serial.print("Speed in kmph = ");

Serial.println(speed);

// if (speed > 1) {

// digitalWrite(buzzerPin, HIGH);

// }

}

delay(1000);

}

**Block diagram of version 3:**

