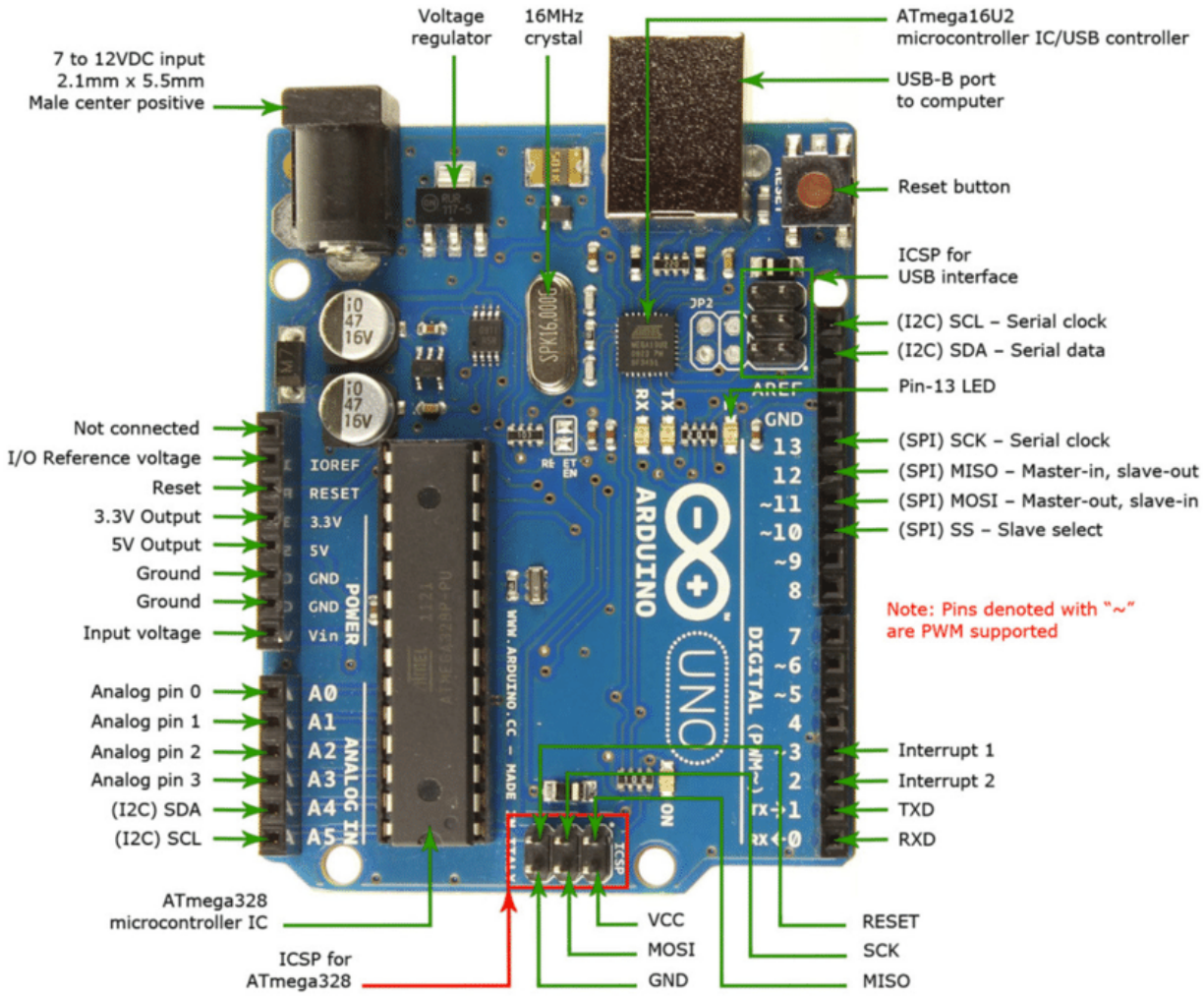
LUSIP-2023 Project

JUNE 2023

By - Harshita Fogat

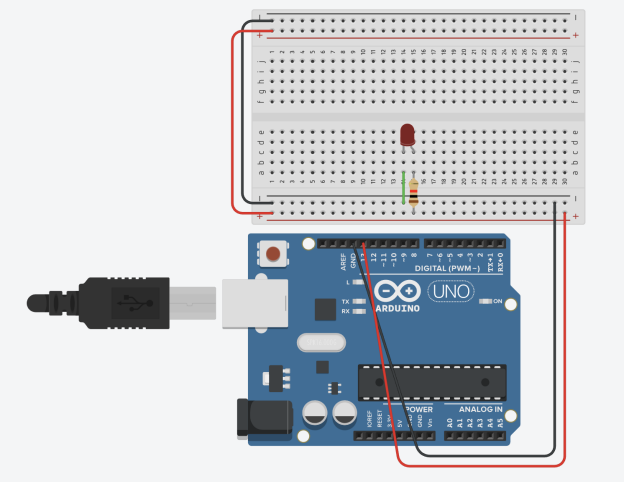
Manipal University Jaipur

ARDUINO



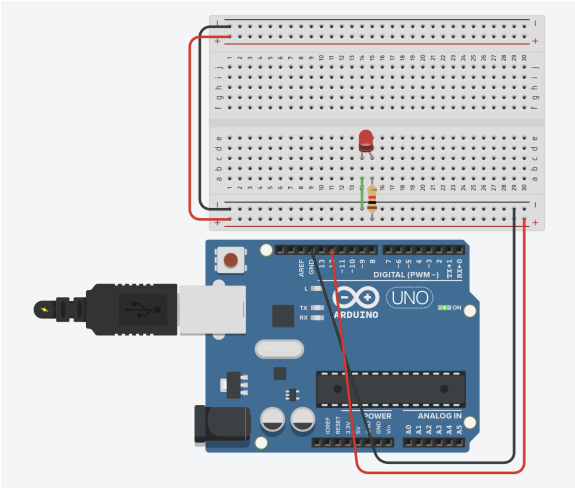
Arduino Programs

1.LED blink



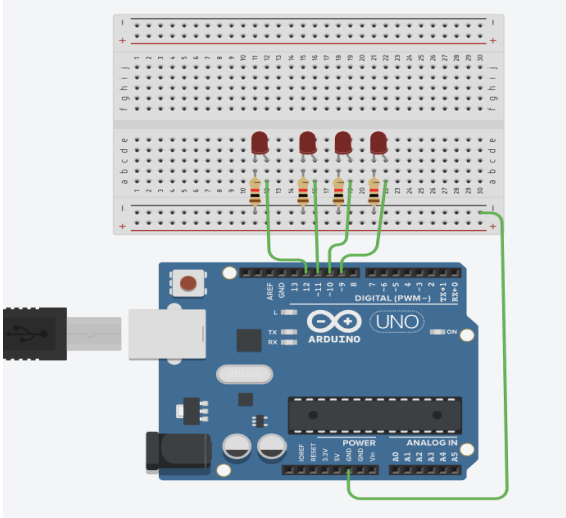

```
int LED =13; // The digital pin to
which the LED is connected

void setup ( )
{
pinMode (LED, OUTPUT);
//Declaring pin 13 as output pin
}

void loop( ) // The loop function runs again and again
{
digitalWrite (LED, HIGH); //Turn ON the LED
delay(1000); //Wait for 1 sec
digitalRead (LED, LOW); // Turn off the LED
delay(1000); // Wait for 1 sec
}
```

2. Fade-in and fade-out the LED




```
int led=12; // The digital pin to which the LED is connected

void setup()
{
pinMode(led, OUTPUT); //pin 10 is set as output pin
}

void loop() // The loop function runs again and again
{
for (int fade=0; fade<=255; fade=fade+5)
{
analogWrite (led, fade); // Change the brightness of LED by 5 points
delay (30);
}
}
```

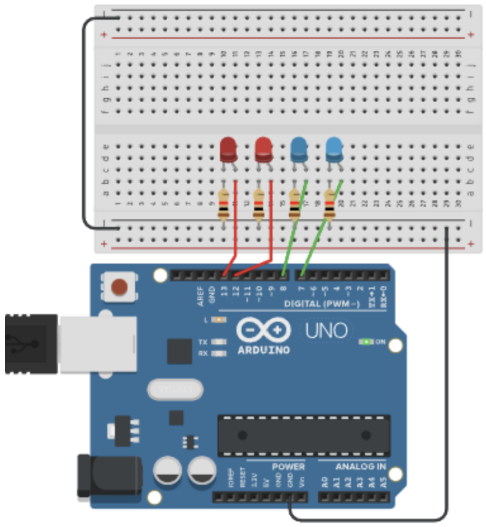
3. Series Blink




```
int pin9=9;
int pin10=10;
int pin11=11;
int pin12=12;
void setup ()
 {
pinMode (pin9, OUTPUT);
pinMode (pin10, OUTPUT);
pinMode (pin11, OUTPUT);
pinMode (pin12, OUTPUT);
}
void loop ()
{
digitalWrite (pin12, LOW);
digitalWrite (pin9, HIGH);
delay (500);
digitalWrite (pin9, LOW);
digitalWrite (pin10, HIGH);
delay (500);
digitalWrite (pin10, LOW);
digitalWrite (pin11, HIGH);
delay (500);
digitalWrite (pin11, LOW);
digitalWrite (pin12, HIGH);
delay (500);
}
```

4. Alternate Series Blink

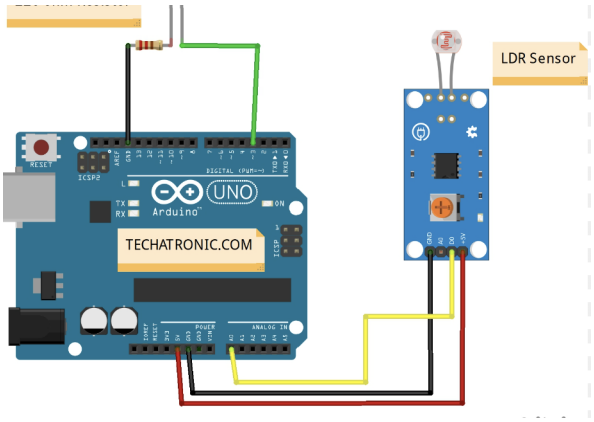
```
void setup ()
{
  pinMode (13, OUTPUT);
  pinMode (12, OUTPUT);
  pinMode (8, OUTPUT);
  pinMode (7, OUTPUT);
}

void loop ()
{
  digitalWrite (13, HIGH);
  digitalWrite (8, HIGH);
  digitalWrite (12, LOW);
  digitalWrite (7, LOW);
  delay (500); // Wait for 500 millisecond(s)
  digitalWrite (13, LOW);
  digitalWrite (8, LOW);
  digitalWrite (12, HIGH);
  digitalWrite (7, HIGH);
  delay (500); // Wait for 500 millisecond(s)
}
```



Sensors

5. LDR sensor

A photoresistor or light dependent resistor is *an electronic component that is sensitive to light*




```
int LDR = A0;              // LDR input at A0 pin.
int LED = 3;               // LED is connected to PWM Pin 3.
int LDRReading = 0;        // to store input value of LDR
int lEDBrightness = 0;     // to store the value of LED Brightness
int threshold_val = 800; // Check your threshold and modify it.
void setup(){
  Serial.begin(9600);     // initializing serial communication.
  pinMode(LED, OUTPUT);   // Defining LED pin as output.
}
void loop(){
  LDRReading = analogRead(LDR);    // Reading LDR Input.
  Serial.println(LDRReading);      // Printing LDR input value.
  if (LDRReading >threshold_val){              // Condition to make LED ON.
 lEDBrightness = map(LDRReading, 0, 1023, 0, 255); // Converting LDR to LED Brightness.
  analogWrite(LED, lEDBrightness); // Writing Brightness to LED.
  }
  else{
  analogWrite(LED, 0);          // If LDR is below threshold make LED OFF.
  }
```

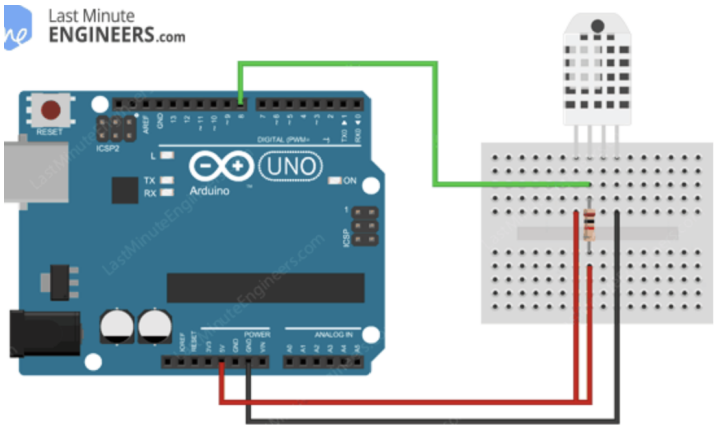
```
  delay(300);                  // delay to make output readable on serial monitor.
}
```

6. LDR using additional LED

```
int LDRInput = A0;            // Set Analog Input A0 for LDR.
int LED = 2;


void setup(){
  Serial.begin(9600);
pinMode(LEDInput, INPUT);
  pinMode(LED, OUTPUT);  }

void loop(){
  LDRReading = analogRead(LDR);    // Reading LDR Input.
  Serial.println(LDRReading);      // Printing LDR input value.

  if (LDRReading >threshold_val){                // Condition to make LED ON.
 lEDBrightness = map(LDRReading, 0, 1023, 0, 255); // Converting LDR to
LED Brightness.
  analogWrite(LED, lEDBrightness); // Writing Brightness to LED.
  }
  else{
  analogWrite(LED, 0);           // If LDR is below threshold make LED OFF.
  }

  delay(300);                  // delay to make output readable on serial monitor.
}
```

7. DHT22

DHT11 is a digital sensor for sensing temperature and humidity.

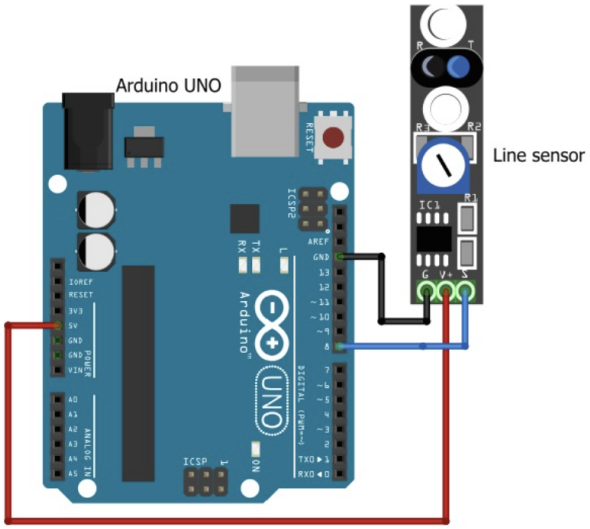



```
//Libraries
#include <DHT.h>;

//Variables
int chk;
float hum;  //Stores humidity value
Float temp;  //Stores temperature value

void setup()
{
   Serial.begin(9600);
      dht.begin();

}

void loop()
{
   //Read data and store it to variables hum and temp
   hum = dht.readHumidity();
   temp= dht.readTemperature();
   //Print temp and humidity values to serial monitor
   Serial.print("Humidity: ");
   Serial.print(hum);
   Serial.print(" %, Temp: ");
   Serial.print(temp);
   Serial.println(" Celsius");
   delay(2000); //Delay 2 sec.
}
```

8. Line Tracking Sensor

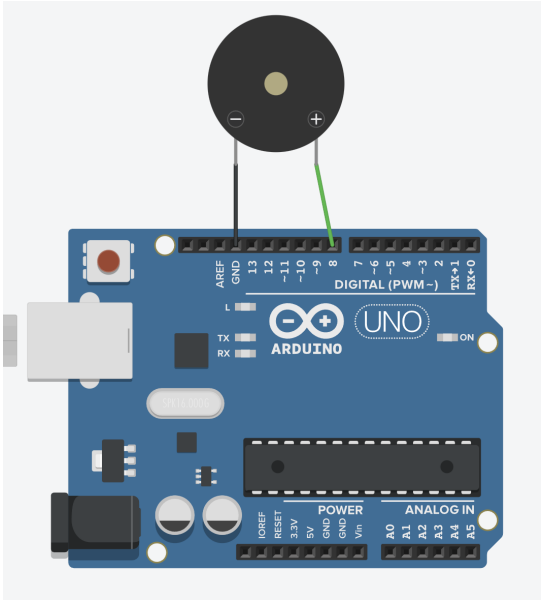
Line sensors detect the presence of a black line by emitting infrared (IR) light and detecting the light levels that return to the sensor

```
void setup()
{
Serial.begin(9600); //
activates Serial
Communication
}

void loop()
{
Serial.print(digitalRead(8)); //
Line Tracking sensor is
connected with pin 8 of the
Arduino
delay(500);
}
```



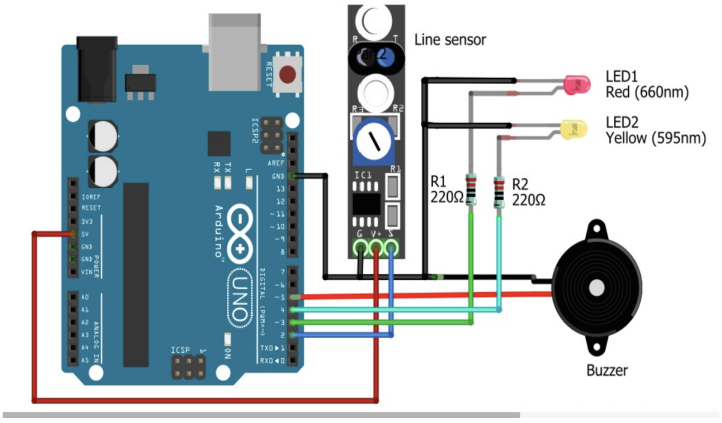
9. Buzzer Sensor

Buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric




```
#define NOTE_C4  262
#define NOTE_G3  196
#define NOTE_A3  220
#define NOTE_B3  247
#define NOTE_C4  262
// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3, NOTE_G3,
NOTE_A3, NOTE_G3, 0, NOTE_B3,
NOTE_C4
};
// note durations: 4 = quarter note, 8 =
eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};
void setup() {
    for (int thisNote = 0; thisNote < 8; thisNote++) {
        int noteDuration = 1000 / noteDurations[thisNote];
    tone(8, melody[thisNote], noteDuration);
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);

    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

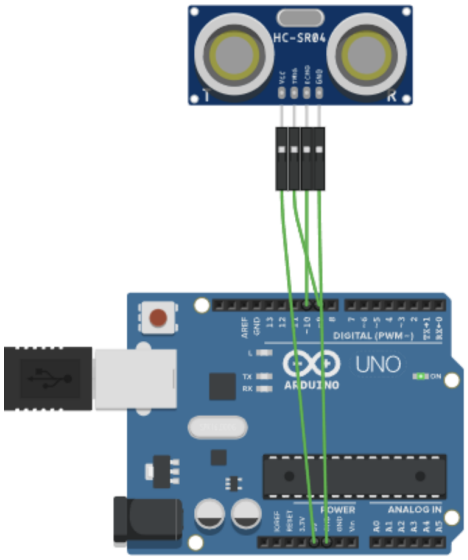
10. Line Tracking Sensor using Buzzer and LED.




```
#define sensor 2
#define buzzer 3
#define red 4
#define green 5

void setup() {
Serial.begin(9600);
pinMode(sensor,INPUT);
pinMode(buzzer,OUTPUT);
pinMode(green,OUTPUT);
pinMode(red,OUTPUT);


}

void loop() {
bool value = digitalRead(sensor);
if(value == 0)
{
  digitalWrite(buzzer, HIGH);
    digitalWrite(green, HIGH);
      digitalWrite(red, HIGH);

}
else
{
    digitalWrite(buzzer, LOW);
    digitalWrite(green, LOW);
      digitalWrite(red, LOW);
  }
}
```

11. UltraSonic Sensor

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves.



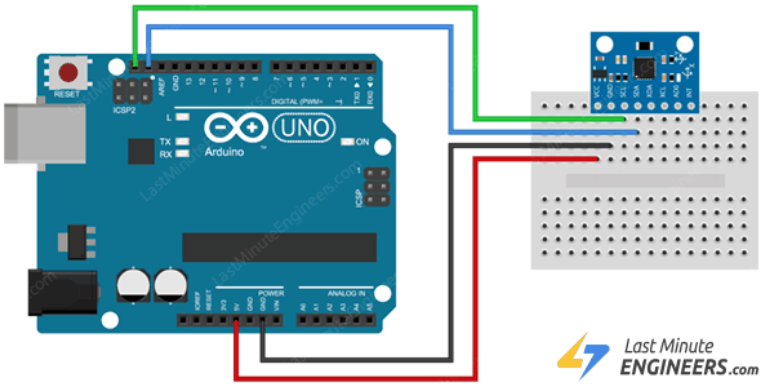

```
// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
}
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;
  // Prints the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);
}
```

RELAY

A relay is an electromagnetic switch; hence, its heart is the electromagnet, which is powered by a small current that acts as a lever or as the switch itself. This makes it possible to allow relatively small electric currents to leverage and control much larger electrical currents.

12. 3D Accelerometer

A three-dimensional (3D) accelerometer is an electromechanical device that detects and measures non-gravitational accelerations.

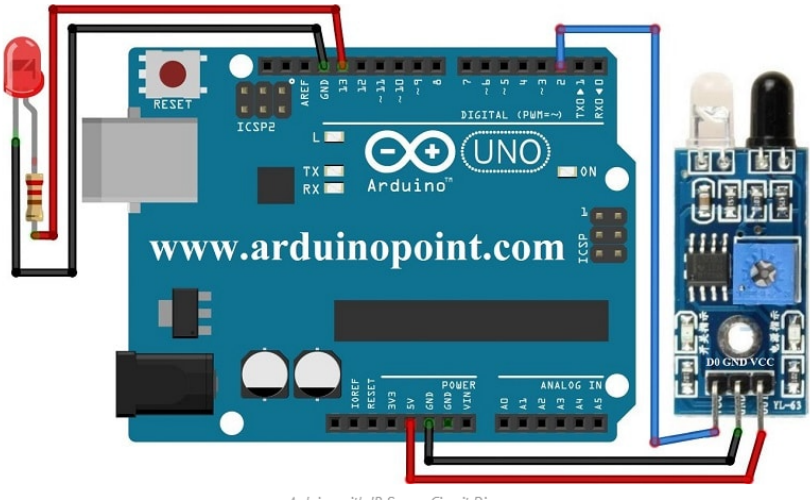



```
#include <Adafruit_MPU6050.h>
#include
<Adafruit_Sensor.h>
#include <Wire.h>

Adafruit_MPU6050
mpu;

void setup(void) {
  Serial.begin(115200);

  // Try to initialize!
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 chip");
    while (1) {
      delay(10);
    }
  }
  Serial.println("MPU6050 Found!");

  // set accelerometer range to +-8G
  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
```

```
  // set gyro range to +- 500 deg/s
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);

  // set filter bandwidth to 21 Hz
  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);

  delay(100);
}
void loop() {
  /* Get new sensor events with the readings */
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  /* Print out the values */
  Serial.print("Acceleration X: ");
  Serial.print(a.acceleration.x);
  Serial.print(", Y: ");
  Serial.print(a.acceleration.y);
  Serial.print(", Z: ");
  Serial.print(a.acceleration.z);
  Serial.println(" m/s^2");
  Serial.print("Rotation X: ");
  Serial.print(g.gyro.x);
  Serial.print(", Y: ");
  Serial.print(g.gyro.y);
  Serial.print(", Z: ");
  Serial.print(g.gyro.z);
  Serial.println(" rad/s");

  Serial.print("Temperature: ");
  Serial.print(temp.temperature);
  Serial.println(" degC");

  Serial.println("");
  delay(500);}
```

13. IR Sensor

An infrared (IR) sensor is *an electronic device that measures and detects infrared radiation in its surrounding environment*.

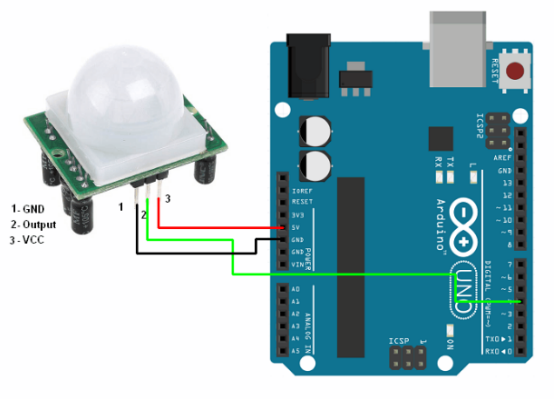



```
int SensorPin = 2;
int OutputPin = 13;

void setup() {
  pinMode(OutputPin,
OUTPUT);
  pinMode(SensorPin,
INPUT);
  Serial.begin(9600);
}

void loop() {
  int SensorValue = digitalRead(SensorPin);

  Serial.print("SensorPin Value: ");
  Serial.println(SensorValue);
  delay(1000);
   if (SensorValue==LOW){ // LOW MEANS Object Detected
    digitalWrite(OutputPin, HIGH);
  }
  else
  {
    digitalWrite(OutputPin, LOW);
  }
}
```

14. PIR Sensor

A device used to detect motion by receiving infrared radiation

```
const int PIR_SENSOR_OUTPUT_PIN =
4;
int warm_up;
void setup() {
  pinMode(PIR_SENSOR_OUTPUT_PIN,
INPUT);
  Serial.begin(9600);     /* Define baud
rate for serial communication */
  delay(20000);    /* Power On Warm Up
Delay */
}
```



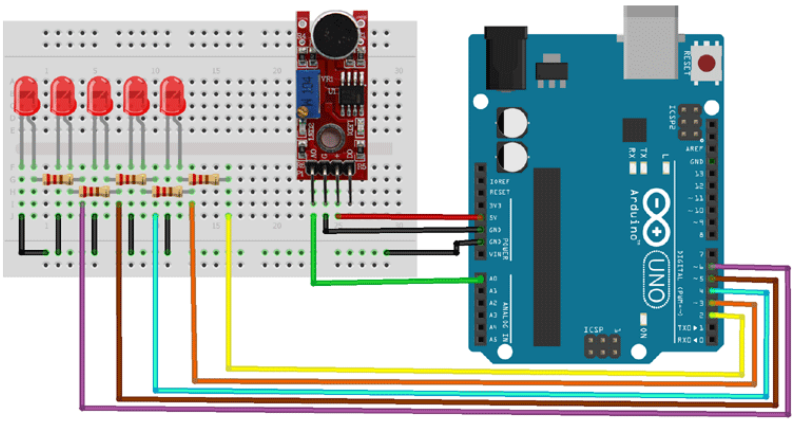
Interfacing PIR Sensor with Arduino UNO

```
void loop() {
  int sensor_output;
  sensor_output = digitalRead(PIR_SENSOR_OUTPUT_PIN);
  if( sensor_output == LOW )
  {
    if( warm_up == 1 )
     {
      Serial.print("Warming Up\n\n");
      warm_up = 0;
      delay(2000);
    }
    Serial.print("No object in sight\n\n");
    delay(1000);
  }
  else
  {
    Serial.print("Object detected\n\n");
    warm_up = 1;
    delay(1000);
  }}
```

15. Sound Detector Sensor Module

A module that detects sound waves via the sound's intensity and then converts it to electric signals.

```
* Arduino pins where the LED is attached*/

#define LED_1 2

#define LED_2 3

#define LED_3 4

#define LED_4 5

#define LED_5 6
```



```
#define sensorPin A0 // Analog input pin that the Sensor is attached to

/* boolean variables to hold the status of the pins*/

bool ledPin1Status;
+

bool ledPin2Status;

bool ledPin3Status;

bool ledPin4Status;

bool ledPin5Status;

void setup() {

  pinMode(LED_1, OUTPUT);
```

```
  pinMode(LED_2, OUTPUT);
  pinMode(LED_3, OUTPUT);
  pinMode(LED_4, OUTPUT);
  pinMode(LED_5, OUTPUT);
  pinMode(sensorPin, INPUT);
  Serial.begin(9600);// initialize serial communications at 9600 bps:
}
void loop() {
  int sensorValue = analogRead(sensorPin);
  Serial.println(sensorValue);
  if (sensorValue > 555  )
    ledPin1Status = 1;
   if (sensorValue > 558  )
    ledPin2Status = 1;
   if (sensorValue > 560  )
    ledPin3Status = 1;
   if (sensorValue > 562  )
    ledPin4Status = 1;
```

```
   if (sensorValue > 564 )

    ledPin5Status = 1;

  if (ledPin1Status == 1 || ledPin2Status == 1 || ledPin3Status == 1 ||
ledPin4Status == 1 || ledPin5Status == 1)

  {

    if (sensorValue > 555 || sensorValue < 537  )

      digitalWrite(LED_1, HIGH);

     if (sensorValue > 558 || sensorValue < 534  )

      digitalWrite(LED_2, HIGH);

     if (sensorValue > 560 || sensorValue < 534  )

      digitalWrite(LED_3, HIGH);

     if (sensorValue > 562 || sensorValue < 531 )

      digitalWrite(LED_3, HIGH);

     if (sensorValue > 564 || sensorValue < 528)

      digitalWrite(LED_4, HIGH);

     if (sensorValue > 568 || sensorValue < 525)

      digitalWrite(LED_5, HIGH);

    delay(200);
```

```
    ledPin5Status = 0;
    ledPin4Status = 0;
    ledPin3Status = 0;
    ledPin2Status = 0;
    ledPin1Status = 0;
  }
  digitalWrite(LED_1, LOW);
  digitalWrite(LED_2, LOW);
  digitalWrite(LED_3, LOW);
  digitalWrite(LED_4, LOW);
  digitalWrite(LED_5, LOW);
}
```

16. PVC Vibration Sensor

A vibration sensor, or vibration detector, measures vibration levels in machinery for screening and analysis.

```
int vib_sensor = A0;
int vib_data = 0;

void setup() {
  // put your setup code here, to run once:
Serial.begin(9600);
pinMode(vib_sensor, INPUT);

}

void loop() {
  // put your main code here, to run repeatedly:
vib_data = analogRead(vib_sensor);
Serial.println(vib_data);
delay(100);
}
```

17. Serial Interface sensor

A serial interface is one in which data is sent in a single stream of bits, usually on a single wire-plus-ground, wire-pair, or single wireless channel.

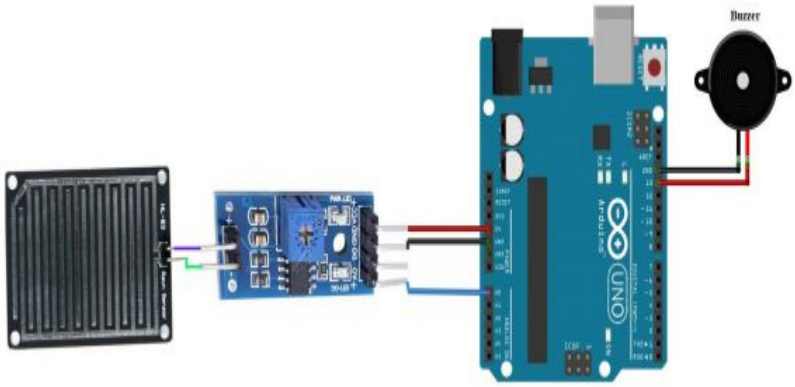
18. Gas Sensor

Gas sensors are *electronic devices that detect and identify different types of gasses*.

19. Rain Drops Module Sensor with Rain Alarm

A raindrop sensor is a board on which nickel is coated in the form of lines. It works on the principle of resistance. The raindrop sensor measures the moisture via analog output pins and it provides a digital output when a threshold of moisture exceeds.

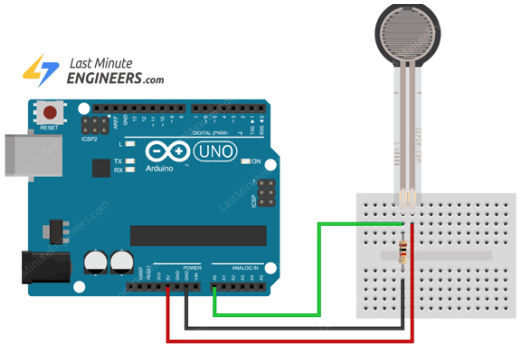
```
const int buzzer = 13;
void setup() {
  // initialize serial
communication at
9600 bits per second:
  Serial.begin(9600);
```




```
pinMode(buzzer,OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  if (sensorValue<900)
  {
      Serial.println("it is raining");
      digitalWrite(buzzer,HIGH);
    }
   else
   {
       digitalWrite(buzzer,LOW);
   }
  delay(1);}        // delay in between reads for stability
```

20. Force Sensor

A Force Sensor is defined as *a transducer that converts an input mechanical load, weight, tension, compression or pressure into an electrical output signal.*




```
int fsrPin = 0;     // the FSR and 10K pulldown are connected to a0
int fsrReading;     // the analog reading from the FSR resistor divider

void setup(void) {
  Serial.begin(9600);
}

void loop(void) {
  fsrReading = analogRead(fsrPin);

  Serial.print("Analog reading = ");
  Serial.print(fsrReading);     // print the raw analog reading

  if (fsrReading < 10) {
    Serial.println(" - No pressure");
  } else if (fsrReading < 200) {
    Serial.println(" - Light touch");
  } else if (fsrReading < 500) {
    Serial.println(" - Light squeeze");
  } else if (fsrReading < 800) {
    Serial.println(" - Medium squeeze");
  } else {
    Serial.println(" - Big squeeze");
  }
  delay(1000);
}
```

Motor

DC Motor

```
void setup() {
  // put your setup code here, to run once:
 pinMode (10, OUTPUT);
 pinMode (8, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
digitalWrite(10,LOW);
digitalWrite(8,HIGH);
delay(5000);
digitalWrite(10,HIGH);
digitalWrite(8,HIGH);
delay(5000);
digitalWrite(10,HIGH);
digitalWrite(8,LOW);
delay(5000);
digitalWrite(10,LOW);
digitalWrite(8,LOW);
delay(5000);
}
```

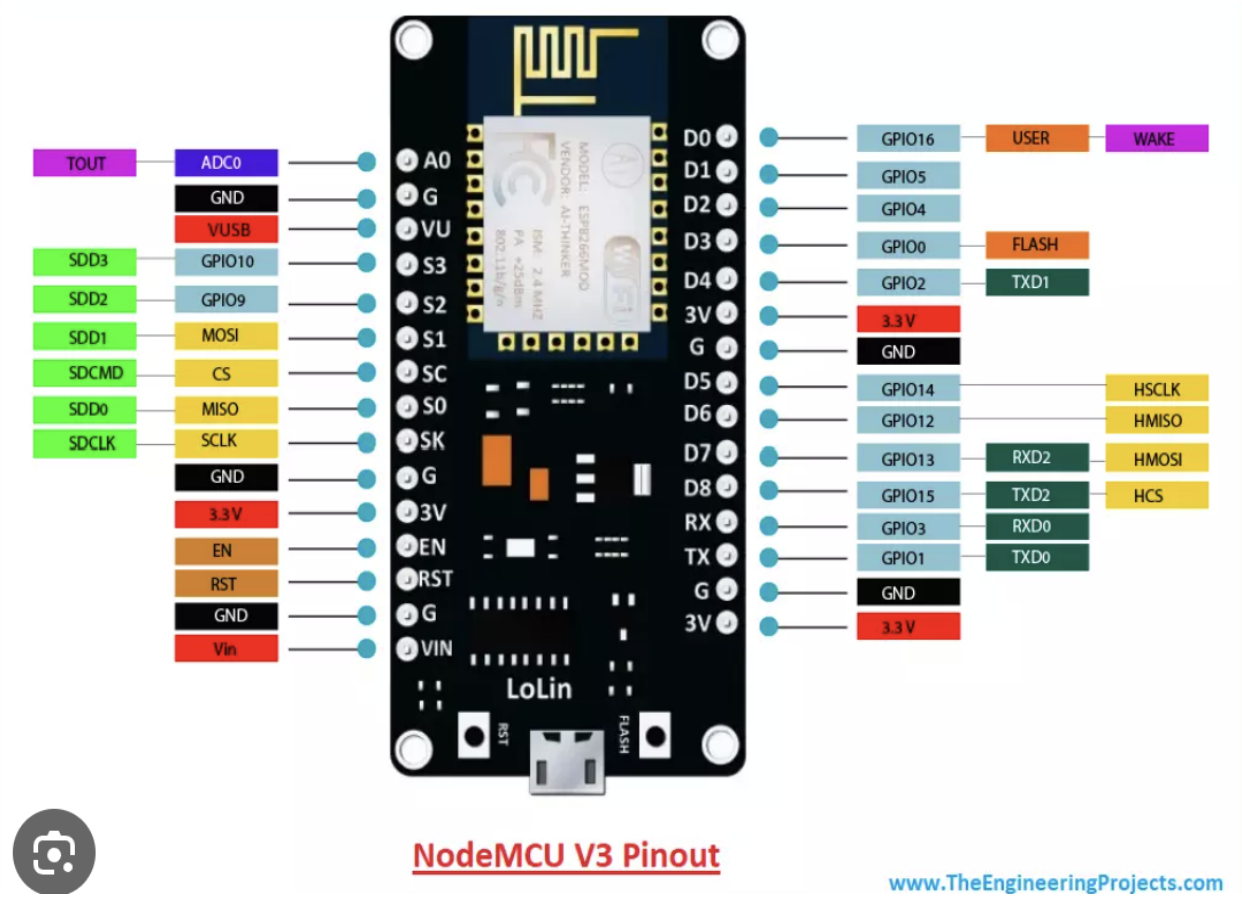
Servo Motor (till a finite angle)

```
//move the servo motor like a wiper

  //access the built-in servo library
  #include <Servo.h>
  Servo Serv1; //create a servo object named Serv1

  //use a PWM pin (3,5,6,9,10,11)
  int pinServo1=3;

  void setup(){
    Serv1.attach(pinServo1);
  }

  void loop(){
    //move the servo 20 degrees
    Serv1.write(90);
    //allow enough time to move
    delay(1000);

    //move the servo 100 degrees
    Serv1.write(180);
    //allow enough time to move
    delay(1000);
  }
```

Network Connectivity

Node MCU—WIFI Module

NodeMCU is an open source development board and firmware based on the widely used ESP8266 -12E WiFi module. It allows you to program the ESP8266 WiFi module with the simple and powerful LUA programming language or Arduino IDE.



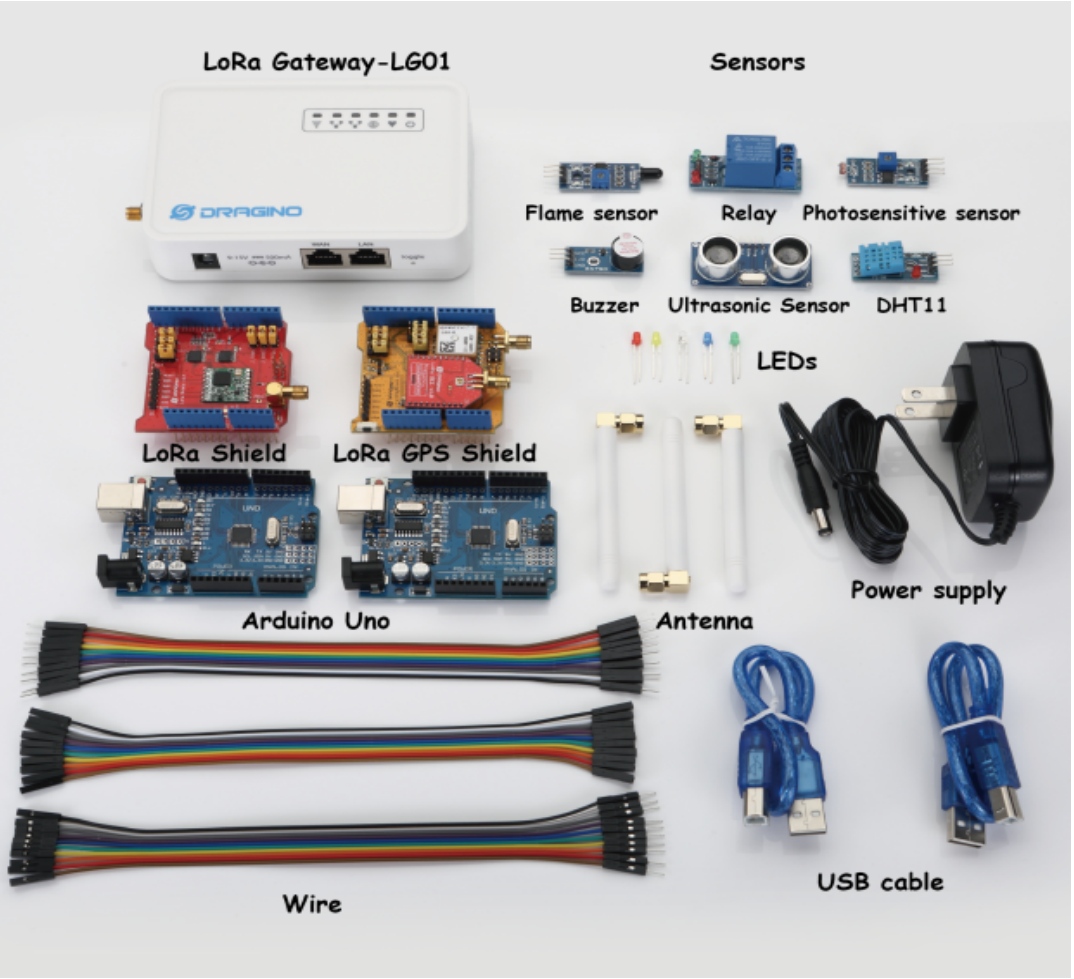
```
#include <ESP8266WiFi.h>   // Include the Wi-Fi library
#include<WiFiClient.h>
#include<ThingSpeak.h>
const char* ssid     = "1+";         // The SSID (name) of the Wi-Fi network
you want to connect to
const char* password = "harshita";     // The password of the Wi-Fi network
int light;
WiFiClient client;
unsigned long mychannelNumber =2181252;
const char *myWriteAPIKey="5EFAH3H7VDOMJ7WF";

void setup() {
  Serial.begin(9600);         // Start the Serial communication to send
messages to the computer
  pinMode(A0,INPUT);
  delay(10);
  WiFi.begin(ssid,password);
  ThingSpeak.begin(client);// Connect to the network
}

void loop() {
  light=analogRead(A0);
  Serial.print("light intensity value=");
  Serial.print(light);
  ThingSpeak.writeField(mychannelNumber,1,light,myWriteAPIKey);
  delay(100);
}
```

Using DHT

```
#include"DHTesp.h"
#include <ESP8266WiFi.h>   // Include the Wi-Fi library
#include<WiFiClient.h>
#include<ThingSpeak.h>
const char* ssid     = "1+";         // The SSID (name) of the Wi-Fi network
you want to connect to
const char* password = "harshita";     // The password of the Wi-Fi network
int temperature;
WiFiClient client;
unsigned long mychannelNumber = 2182749;
const char *myWriteAPIKey="7V1JA747VC0TDNJ2";
DHTesp dht;
void setup() {
  Serial.begin(115200);         // Start the Serial communication to send
messages to the computer
  Serial.println();
  WiFi.begin(ssid,password);
  ThingSpeak.begin(client);// Connect to the network
  dht.setup(D1, DHTesp::DHT22);
}
void loop() {
   Serial.println("temperature value=");
    Serial.println(temperature);
delay(200);
float humidity=dht.getHumidity();
float temperature=dht.getTemperature();
  ThingSpeak.writeField(mychannelNumber, 1, temperature,
myWriteAPIKey);
  delay(2000);
  ThingSpeak.writeField(mychannelNumber, 2, humidity, myWriteAPIKey);
  delay(2000);
}
```

LoRa

LoRa is a wireless technology for low-power, wide-area networks (LPWANs), enabling long-range, low-power communication in challenging environments. LoRa networks support numerous devices, facilitating extensive IoT deployments.



ZigBee

Zigbee is a standards-based wireless technology developed to enable low-cost, low-power wireless machine-to-machine (M2M) and internet of things (IoT) networks. Zigbee is for low-data rate, low-power applications and is an open standard.