## ULTRASONIC SENSOR:

## 

## 

## 

This sensor uses Ultrasonic waves as a medium to detect the distance or range. This sensor consists of a Ultrasonic wave transmitter which generates the wave. The transmitted wave bounces back by any medium or obstacle and returns to the sensor where it will be sensed by receiver. Now the distance between the sensor and obstacle is calculated based on the time it for the wave between transmission and reception. The further the obstacle is the longer the time delay between transmission and reception.

**Commonly used:** HC-SR04

### HC-SR04 Ultrasonic Sensor - Working

As shown above the HC-SR04 Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver.

**Distance = Speed × Time**

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module as shown in the picture below



## Now, to calculate the distance using the above formulae, we should know the Speed and time. Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is 330m/s. The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken. Now simply calculate the distance using a microcontroller or microprocessor.

## 

## Arduino Code

const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor

const int echoPin = 6; // Echo Pin of Ultrasonic Sensor

void setup() {

Serial.begin(9600); // Starting Serial Terminal

}

void loop() {

long duration, inches, cm;

pinMode(pingPin, OUTPUT);

digitalWrite(pingPin, LOW);

delayMicroseconds(2);

digitalWrite(pingPin, HIGH);

delayMicroseconds(10);

digitalWrite(pingPin, LOW);

pinMode(echoPin, INPUT);

duration = pulseIn(echoPin, HIGH);

inches = microsecondsToInches(duration);

cm = microsecondsToCentimeters(duration);

Serial.print(inches);

Serial.print("in, ");

Serial.print(cm);

Serial.print("cm");

Serial.println();

delay(100);

}

long microsecondsToInches(long microseconds) {

return microseconds / 74 / 2;

}

long microsecondsToCentimeters(long microseconds) {

return microseconds / 29 / 2;

}

## Result

You will see the distance measured by sensor in inches and cm on Arduino serial monitor.

# **MG996R Servo Motor**

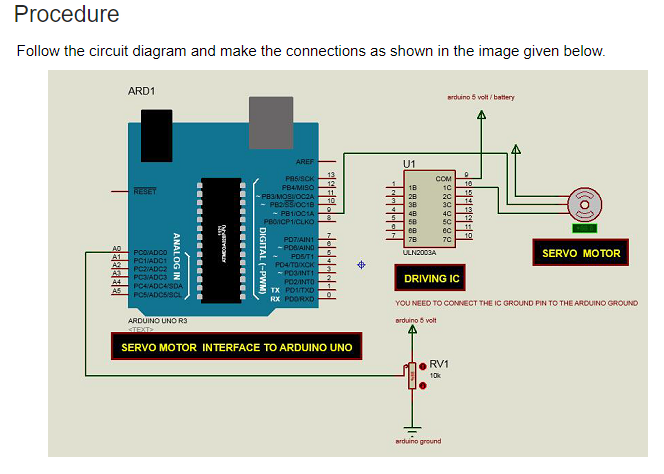
## 

A Servo Motor is a small device that has an output shaft. This shaft can be positioned to specific angular positions by sending the servo a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular position of the shaft. If the coded signal changes, the angular position of the shaft changes. In practice, servos are used in radio-controlled airplanes to position control surfaces like the elevators and rudders. They are also used in radio-controlled cars, puppets, and of course, robots.

Components Required

You will need the following components −

* 1 × Arduino UNO board
* 1 × Servo Motor
* 1 × ULN2003 driving IC
* 1 × 10 KΩ Resistor



## Arduino Code

#include <Servo.h>

Servo myservo; // create servo object to control a servo

int potpin = 0; // analog pin used to connect the potentiometer

int val; // variable to read the value from the analog pin

void setup() {

myservo.attach(9); // attaches the servo on pin 9 to the servo object

}

void loop() {

val = analogRead(potpin);

// reads the value of the potentiometer (value between 0 and 1023)

val = map(val, 0, 1023, 0, 180);

// scale it to use it with the servo (value between 0 and 180)

myservo.write(val); // sets the servo position according to the scaled value

delay(15);

}

Water sensor

Water sensor brick is designed for water detection, which can be widely used in sensing rainfall, water level, and even liquid leakage.



Connecting a water sensor to an Arduino is a great way to detect a leak, spill, flood, rain, etc. It can be used to detect the presence, the level, the volume and/or the absence of water. While this could be used to remind you to water your plants, there is a better Grove sensor for that. The sensor has an array of exposed traces, which read LOW when water is detected.

In this chapter, we will connect the water sensor to Digital Pin 8 on Arduino, and will enlist the very handy LED to help identify when the water sensor comes into contact with a source of water.

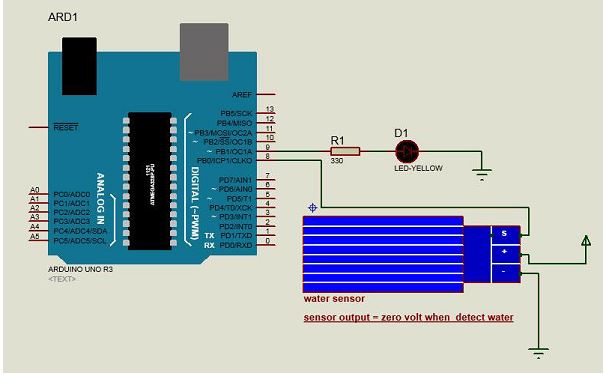
Components Required

You will need the following components −

* 1 × Breadboard
* 1 × Arduino Uno R3
* 1 × Water Sensor
* 1 × led
* 1 × 330 ohm resistor

## Procedure

Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.



## Arduino Code

#define Grove\_Water\_Sensor 8 // Attach Water sensor to Arduino Digital Pin 8

#define LED 9 // Attach an LED to Digital Pin 9 (or use onboard LED)

void setup() {

pinMode(Grove\_Water\_Sensor, INPUT); // The Water Sensor is an Input

pinMode(LED, OUTPUT); // The LED is an Output

}

void loop() {

/\* The water sensor will switch LOW when water is detected.

Get the Arduino to illuminate the LED and activate the buzzer

when water is detected, and switch both off when no water is present \*/

if( digitalRead(Grove\_Water\_Sensor) == LOW) {

digitalWrite(LED,HIGH);

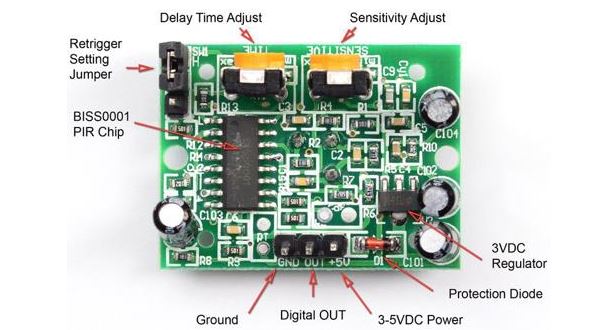
}else {

digitalWrite(LED,LOW);

}

}

PIR sensors:

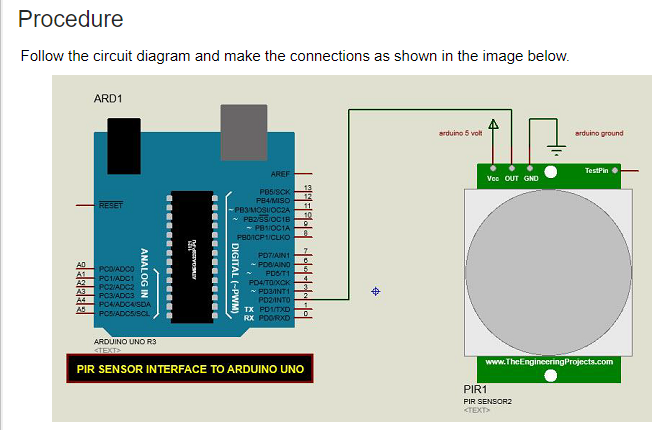


PIR sensors allow you to sense motion. They are used to detect whether a human has moved in or out of the sensor’s range. They are commonly found in appliances and gadgets used at home or for businesses. They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IR motion" sensors.

Components Required

You will need the following components −

* 1 × Breadboard
* 1 × Arduino Uno R3
* 1 × PIR Sensor (MQ3)



## Arduino Code

#define pirPin 2

int calibrationTime = 30;

long unsigned int lowIn;

long unsigned int pause = 5000;

boolean lockLow = true;

boolean takeLowTime;

int PIRValue = 0;

void setup() {

Serial.begin(9600);

pinMode(pirPin, INPUT);

}

void loop() {

PIRSensor();

}

void PIRSensor() {

if(digitalRead(pirPin) == HIGH) {

if(lockLow) {

PIRValue = 1;

lockLow = false;

Serial.println("Motion detected.");

delay(50);

}

takeLowTime = true;

}

if(digitalRead(pirPin) == LOW) {

if(takeLowTime){

lowIn = millis();takeLowTime = false;

}

if (!lockLow && millis() - lowIn > pause) {

PIRValue = 0;

lockLow = true;

Serial.println("Motion ended.");

delay(50);

}

}

}

## Result

You will see a message on your serial port if a motion is detected and another message when the motion stops.