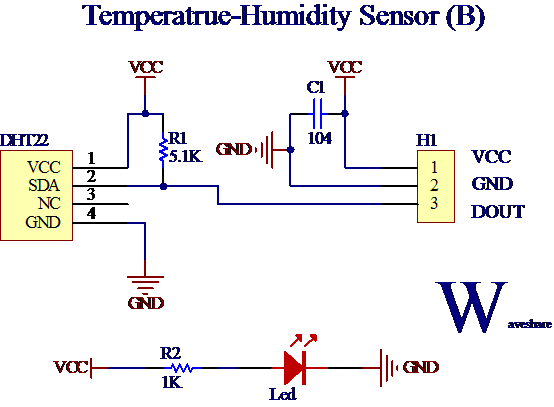
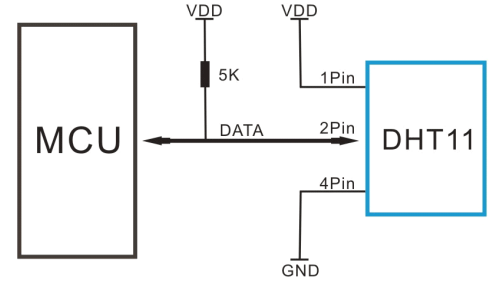
Temperature and humidity sensor



The **DHT11**is a commonly used **Temperature and humidity sensor that** comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data.



Vcc-Power supply 3.5V to 5.5V

Data-Outputs both Temperature and Humidity through serial Data

Ground: Connected to the ground of the circuit

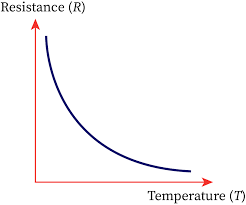
## Working Principle

They consist of a humidity sensing component, a NTC temperature sensor (or thermistor) and an IC on the back side of the sensor.

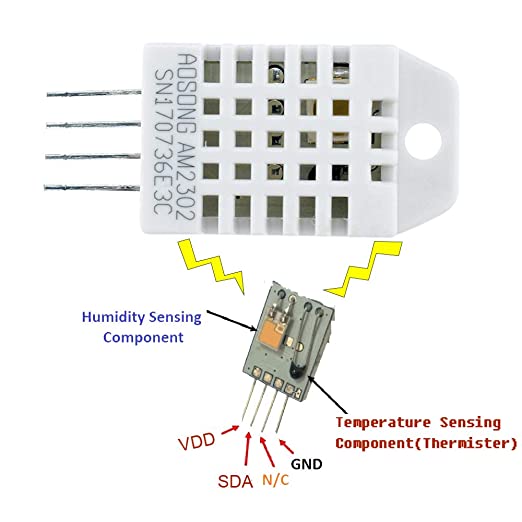
For measuring humidity they use the humidity sensing component which has two electrodes with moisture holding substrate between them. So as the humidity changes, the conductivity of the substrate changes or the resistance between these electrodes changes. This change in resistance is measured and processed by the IC which makes it ready to be read by a microcontroller.

On the other hand, for measuring temperature these sensors use a NTC temperature sensor or a thermistor.

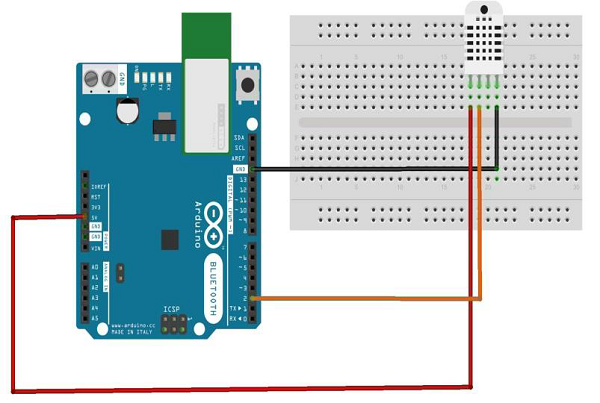
A thermistor is actually a variable resistor that changes its resistance with change of the temperature. These sensors are made by sintering of semiconductive materials such as ceramics or polymers in order to provide larger changes in the resistance with just small changes in temperature.



The term “NTC” means “Negative Temperature Coefficient”, which means that the resistance decreases with increase of the temperature.



Circuit diagram



The DHTxx sensors have four pins, VCC, GND, data pin and a not connected pin which has no usage. A pull-up resistor from 5K to 10K Ohms is required to keep the data line high and in order to enable the communication between the sensor and the Arduino Board. There are some versions of these sensors that come with a breakout boards with built-in pull-up resistor and they have just 3 pins.

The DHTXX sensors have their own single wire protocol used for transferring the data. This protocol requires precise timing and the timing diagrams for getting the data from the sensors can be found from the datasheets of the sensors

Code:

**#include <dht.h>**

**#define dataPin 8 *// Defines pin number to which the sensor is connected***

dht DHT; *// Creats a DHT object*

void **setup**() {

Serial.begin(9600);

}

void **loop**() {

int readData = DHT.read22(dataPin); *// Reads the data from the sensor*

float t = DHT.temperature; *// Gets the values of the temperature*

float h = DHT.humidity; *// Gets the values of the humidity*

*// Printing the results on the serial monitor*

Serial.print("Temperature = ");

Serial.print(t);

Serial.print(" \*C ");

Serial.print(" Humidity = ");

Serial.print(h);

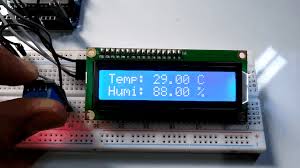
Serial.println(" % ");

delay(2000); *// Delays 2 secods, as the DHT22 sampling rate is 0.5Hz*

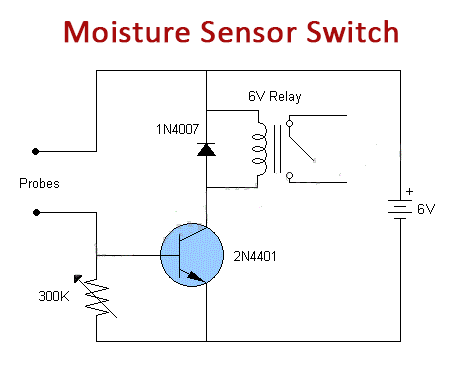
}

After we will upload this code to the Arduino board, the temperature and humidity results from the sensor can be seen on the Serial monitor.

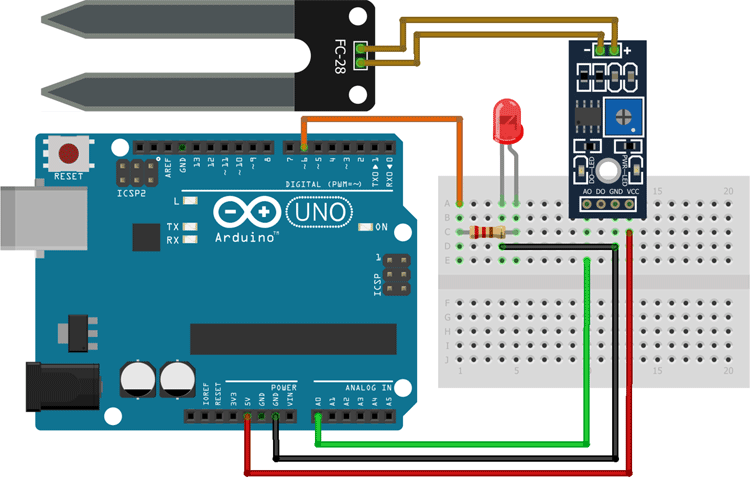
Output:



2) Moisture sensor:



Pin diagram:



The working of the soil moisture sensor is pretty simple and straightforward, as you can see in the image below. We just need to stick the fork-shaped conductive probe to the soil, as the probe has **two exposed conductive plates that will act as a variable resistor** whose resistance will vary depending on the water content in the soil.

This resistance of the probe is inversely proportional to the soil moisture of the device. The more water in the soil the better the conductivity which will result in lower resistance. The less the water in the soil the poor the conductivity which means higher resistance. This sensor produces an output voltage according to the resistance by measuring which we can determine the moisture level.

Code:

#define ledPin 6

#define sensorPin A0

void setup() {

Serial.begin(9600);

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, LOW);

}

void loop() {

Serial.print("Analog output: ");

Serial.println(readSensor());

delay(500);

}

int readSensor() {

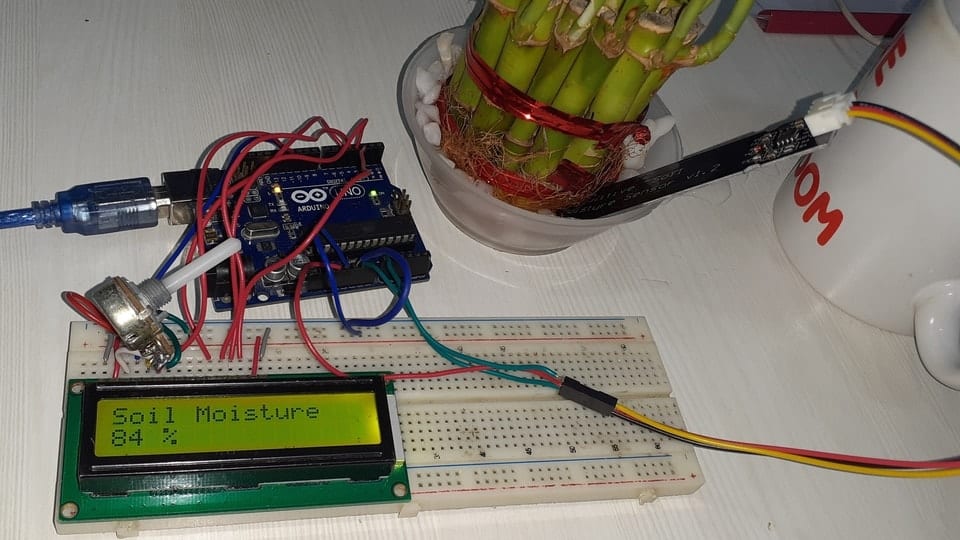
int sensorValue = analogRead(sensorPin); // Read the analog value from sensor

int outputValue = map(sensorValue, 0, 1023, 255, 0); // map the 10-bit data to 8-bit data

analogWrite(ledPin, outputValue); // generate PWM signal

return outputValue; // Return analog moisture value

}



3)Ultrasonic sensor:

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

An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object’s proximity.

High-frequency sound waves reflect from boundaries to produce distinct echo patterns.

Working:

Ultrasonic sensors work by sending out a sound wave at a frequency above the range of human hearing.  The transducer of the sensor acts as a microphone to receive and send the ultrasonic sound. Our [ultrasonic sensors](https://www.maxbotix.com/SelectionGuide/Selection-Guide.htm), like many others, use a single transducer to send a pulse and to receive the echo.  The sensor determines the distance to a target by measuring time lapses between the sending and receiving of the ultrasonic pulse.

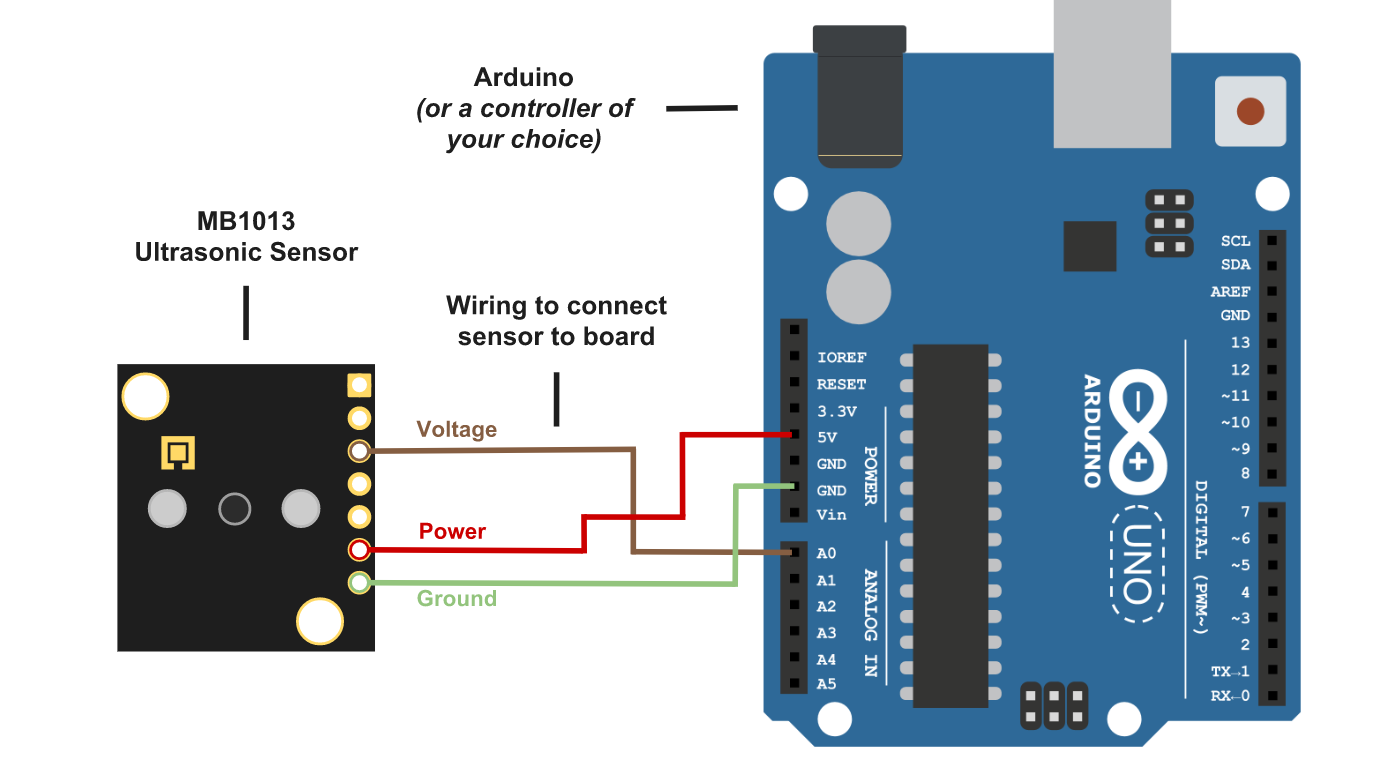
The working principle of this module is simple.  It sends an ultrasonic pulse out at 40kHz which travels through the air and if there is an obstacle or object, it will bounce back to the sensor.  By calculating the travel time and the speed of sound, the distance can be calculated.

Ultrasonic sensors are a great solution for the detection of clear objects.  For liquid level measurement, applications that use infrared sensors, for instance, struggle with this particular use case because of target translucence.

For presence detection, ultrasonic sensors detect objects regardless of the color, surface, or material (unless the material is very soft like wool, as it would absorb sound.)

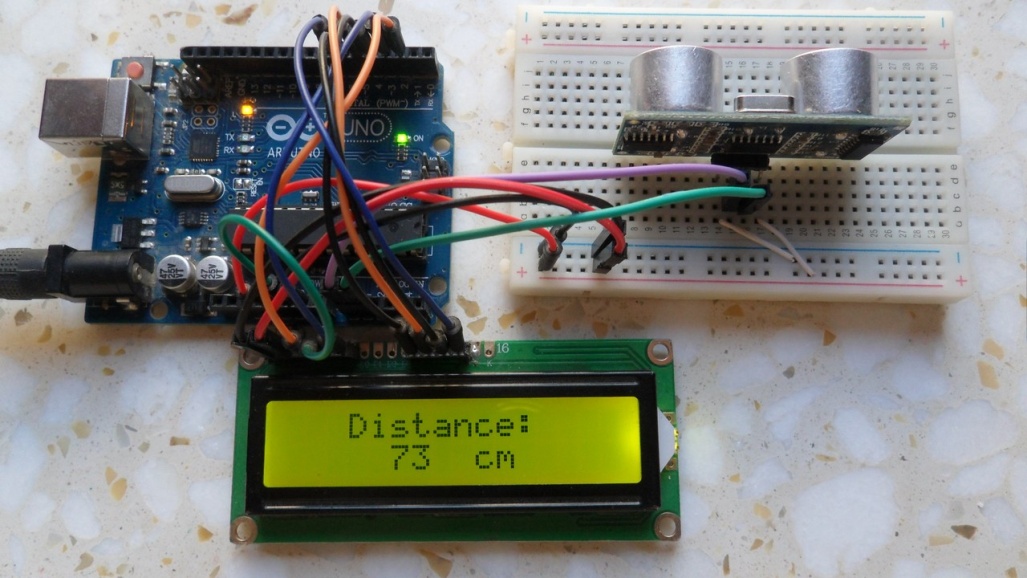
To detect transparent and other items where optical technologies may fail, ultrasonic sensors are a reliable choice.

Pin diagram:



Code:

|  |
| --- |
| const int anPin = 0; |
|  | long anVolt, cm; |
|  |  |
|  | void setup() { |
|  | Serial.begin(9600); |
|  | } |
|  |  |
|  | void read\_sensor(){ |
|  | anVolt = analogRead(anPin); |
|  | cm = anVolt/2; |
|  | } |
|  |  |
|  | void print\_range(){ |
|  | Serial.print(“Range = ”); |
|  | Serial.print(cm); |
|  | Serial.print(” cm “); |
|  | Serial.print('\n'); |
|  | } |
|  |  |
|  | void loop() { |
|  | read\_sensor(); |
|  | print\_range(); |
|  | delay(100); |
|  | } |



4)IR sensor:



IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. An [**IR sensor**](https://robu.in/product-category/sensor/ir-and-pir-sensor/) can measure the heat of an object as well as detects the motion. Usually, in the [**infrared spectrum**](https://en.wikipedia.org/wiki/Infrared_spectroscopy), all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.

The emitter is simply an IR LED [**(Light Emitting Diode**](https://robu.in/product-category/display-boards/led/)) and the detector is simply an IR photodiode . Photodiode is sensitive to IR light of the same wavelength which is emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

Working:

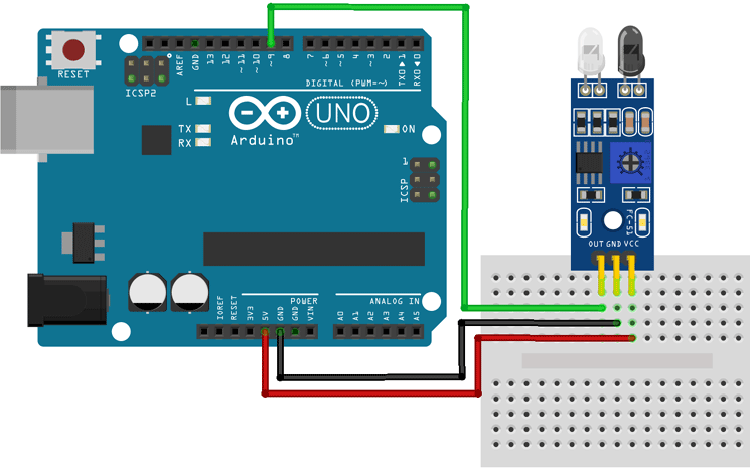
The working principle of an infrared sensor is similar to the object detection sensor. This sensor includes an IR LED & an IR Photodiode, so by combining these two can be formed as a photo-coupler otherwise optocoupler. The physics laws used in this sensor are planks radiation, Stephan Boltzmann & weins displacement.

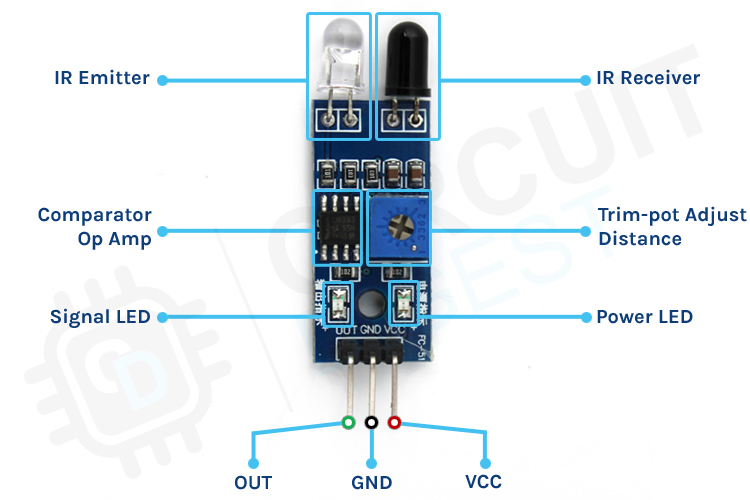
IR LED is one kind of transmitter that emits IR radiations. This LED looks similar to a standard LED and the radiation which is generated by this is not visible to the human eye. Infrared receivers mainly detect the radiation using an infrared transmitter. These infrared receivers are available in photodiodes form. IR Photodiodes are dissimilar as compared with usual photodiodes because they detect simply IR radiation. Different kinds of infrared receivers mainly exist depending on the voltage, wavelength, package, etc.

Once it is used as the combination of an IR transmitter & receiver, then the receiver’s wavelength must equal the transmitter. Here, the transmitter is IR LED whereas the receiver is IR photodiode. The infrared photodiode is responsive to the infrared light that is generated through an infrared LED. The resistance of photo-diode & the change in output voltage is in proportion to the infrared light obtained. This is the IR sensor’s fundamental working principle.

Once the infrared transmitter generates emission, then it arrives at the object & some of the emission will reflect back toward the infrared receiver. The sensor output can be decided by the IR receiver depending on the intensity of the response.

Pin diagram:





Code:

int IRSensor = 9; // connect IR sensor module to Arduino pin D9

int LED = 13; // connect LED to Arduino pin 13

void setup(){

Serial.begin(115200); // Init Serial at 115200 Baud Rate.

Serial.println("Serial Working"); // Test to check if serial is working or not

pinMode(IRSensor, INPUT); // IR Sensor pin INPUT

pinMode(LED, OUTPUT); // LED Pin Output

}

void loop(){

int sensorStatus = digitalRead(IRSensor); // Set the GPIO as Input

if (sensorStatus == 1) // Check if the pin high or not

{

// if the pin is high turn off the onboard Led

digitalWrite(LED, LOW); // LED LOW

Serial.println("Motion Detected!"); // print Motion Detected! on the serial monitor window

}

else {

//else turn on the onboard LED

digitalWrite(LED, HIGH); // LED High

Serial.println("Motion Ended!"); // print Motion Ended! on the serial monitor window

}

}

//Constants:

int IRPin = 3; // declaring pin 3 to read digital input

//Variables:

int value; //save the digital value sent by ir led

void setup(){

pinMode(IRPin, INPUT); //Set pin 3 as 'input'

Serial.begin(115200); //Begin serial communication

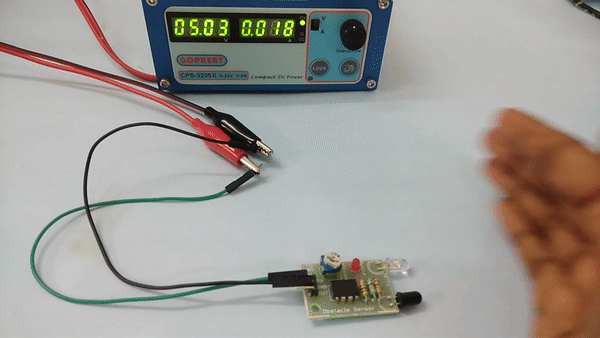
}

void loop(){

value = digitalRead(IRPin); //Read and save digital value from ir sensor

Serial.println(value); //Print value

}



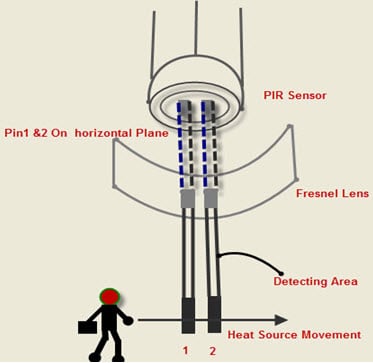
PIR sensor:

PIR sensors mostly used in PIR-based motion detectors. Also, it used in security alarms and automatic lighting applications.



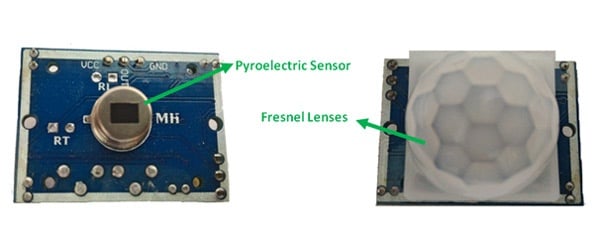
* Pin1 corresponds to the drain terminal of the device, which connected to the positive supply 5V DC.
* Pin2 corresponds to the source terminal of the device, which connects to the ground terminal via a 100K or 47K resistor. The Pin2 is the output pin of the sensor. The pin 2 of the sensor carries the detected IR signal to an amplifier from the
* Pin3 of the sensor connected to the ground.

Working



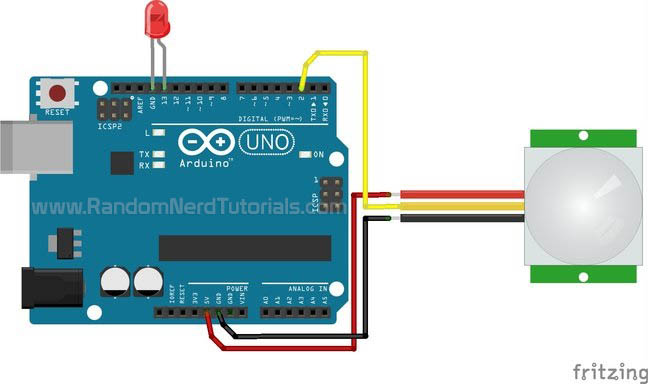
The passive infrared sensor does not radiate energy to space. It receives the  infrared radiation from the human body to make an alarm. Any object with temperature is constantly radiating infrared rays to the outside world. The surface temperature of the human body is between 36° C - 27 ° C and most of its radiant energy concentrated in the wavelength range of 8 um-12 um.

Passive infrared alarms classified into[**infrared detectors**](https://robu.in/product-category/sensor/ir-and-pir-sensor/) (infrared probes) and alarm control sections. The most widely used infrared detector is a pyroelectric detector. It uses as a sensor for converting human infrared radiation into electricity. If the human infrared radiation is directly irradiated on the detector, it will, of course, cause a temperature change to output a signal. But in doing all this, the detection distance will not be more. In order to lengthen the detection distance of the detector, an optical system  must be added to collect the infrared radiation. Usually, plastic optical reflection system or plastic **Fresnel lens** used as a focusing system for infrared radiation.



In the detection area, the lens of the detector receives the infrared radiation energy of the human body through the clothing and focused on the pyroelectric sensor. When the human body moves in this surveillance mode, it enters a certain field of view in sequence and then walks out of the field of view. The[**pyroelectric sensor**](https://www.sciencedirect.com/topics/engineering/pyroelectric-sensor) sees the moving human body for a while and then does not see it, so the infrared radiation of human body constantly changes the temperature of the pyroelectric material. So that it outputs a corresponding signal, which is the alarm signal.

Pin Diagram:



Code:

/\*

Arduino with PIR motion sensor

For complete project details, visit: http://RandomNerdTutorials.com/pirsensor

Modified by Rui Santos based on PIR sensor by Limor Fried

\*/

int led = 13; // the pin that the LED is atteched to

int sensor = 2; // the pin that the sensor is atteched to

int state = LOW; // by default, no motion detected

int val = 0; // variable to store the sensor status (value)

void setup() {

pinMode(led, OUTPUT); // initalize LED as an output

pinMode(sensor, INPUT); // initialize sensor as an input

Serial.begin(9600); // initialize serial

}

void loop(){

val = digitalRead(sensor); // read sensor value

if (val == HIGH) { // check if the sensor is HIGH

digitalWrite(led, HIGH); // turn LED ON

delay(100); // delay 100 milliseconds

if (state == LOW) {

Serial.println("Motion detected!");

state = HIGH; // update variable state to HIGH

}

}

else {

digitalWrite(led, LOW); // turn LED OFF

delay(200); // delay 200 milliseconds

if (state == HIGH){

Serial.println("Motion stopped!");

state = LOW; // update variable state to LOW

}

}

}

