**Ex-1 : Blinking a LED**

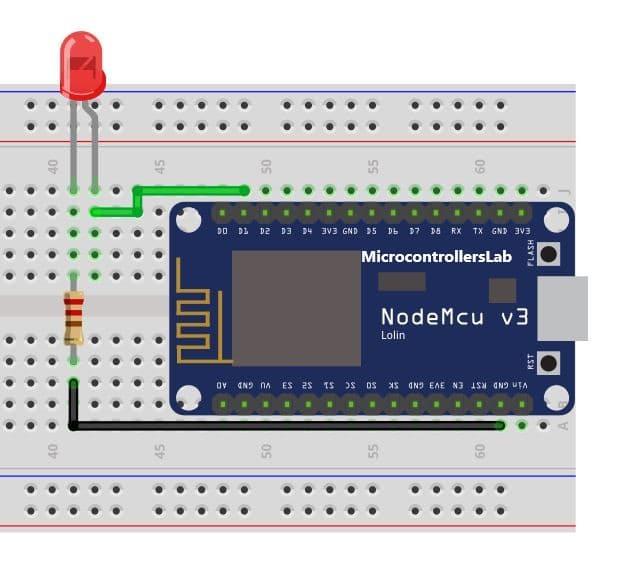
**Aim:**

To blink a LED using Node MCU

**Hardware Requirements:**

* NodeMCU x 1
* LED x 1
* BreadBoard
* 200 ohm – 1K ohm resistor x 1
* Micro USB cable x 1
* PC x 1
* Software Arduino IDE(version 1.6.4+)
* Jumper Wires (Male – Female & Male – Male)

**Circuit:**



**Code:**

int LED = 5; // Assign LED pin i.e: D1 on NodeMCU

void setup() {

// initialize GPIO 5 as an output

pinMode(LED, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

digitalWrite(LED, HIGH); // turn the LED on

delay(1000); // wait for a second

digitalWrite(LED, LOW); // turn the LED off

delay(1000); // wait for a second

}

**Result:**

Blinking of LED has been done successfully using Node MCU and the results are verified.

**Ex: 2 Measuring Temperature using temperature sensor**

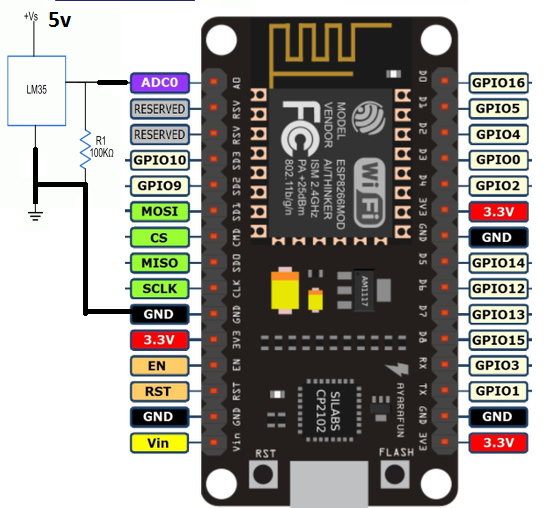
**Aim:**

To measure the temperature of the room / surrounding using temperature sensor & Node MCU

**Hardware Requirements:**

* NodeMCU
* LM35 Temperature Sensor
* Bread Board
* Jumper Wires
* Micro USB Cable
* Arduino IDE
* 200 ohm – 1K ohm resistor x 1
* PC

**Circuit Diagram:**



**Code:**

// initializes or defines the output pin of the LM35 temperature sensor

int outputpin= A0;

//this sets the ground pin to LOW and the input voltage pin to high

void setup() {

Serial.begin(9600);

}

void loop() //main loop

{

int analogValue = analogRead(outputpin);

float millivolts = (analogValue/1024.0) \* 3300; //3300 is the voltage provided by NodeMCU

float celsius = millivolts/10;

Serial.print("in DegreeC= ");

Serial.println(celsius);

//----------Calculation for Fahrenheit ----------//

float fahrenheit = ((celsius \* 9)/5 + 32);

Serial.print(" in Farenheit= ");

Serial.println(fahrenheit);

delay(1000);

}

**Result:**

Temperature has been measured successfully using Temperature sensor and the results are verified.

**Ex: 3 InfraRed Sensor using Node MCU**

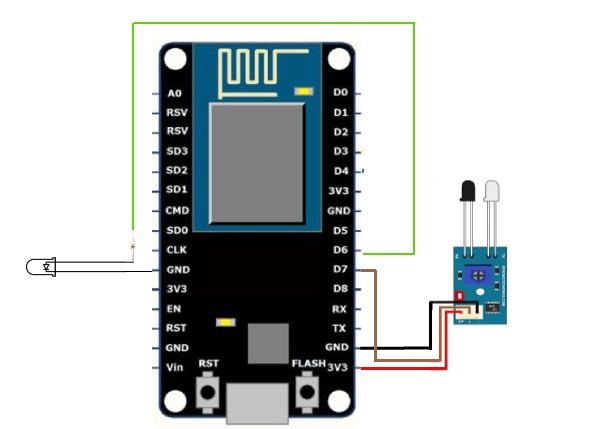
**Aim:**

To detect any motion in the room / surrounding using temperature sensor & Node MCU

**Hardware Requirements:**

* NodeMCU
* IR Sensor
* Bread Board
* Jumper Wires
* Micro USB Cable
* Arduino IDE
* 200 ohm – 1K ohm resistor x 1
* PC
* LED

**Circuit Diagram:**



**Code:**

int ledPin = 12; // choose pin for the LED

int inputPin = 13; // choose input pin (for Infrared sensor)

int val = 0; // variable for reading the pin status

void setup()

{

pinMode(ledPin, OUTPUT); // declare LED as output

pinMode(inputPin, INPUT); // declare Infrared sensor as input

}

void loop()

{

val = digitalRead(inputPin); // read input value

if (val == HIGH)

{ // check if the input is HIGH

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

}

else

{

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

}

}

**Result:**

IR sensor has been used successfully using Node MCU and the results are verified.

**Ex: 4 Light Sensor**

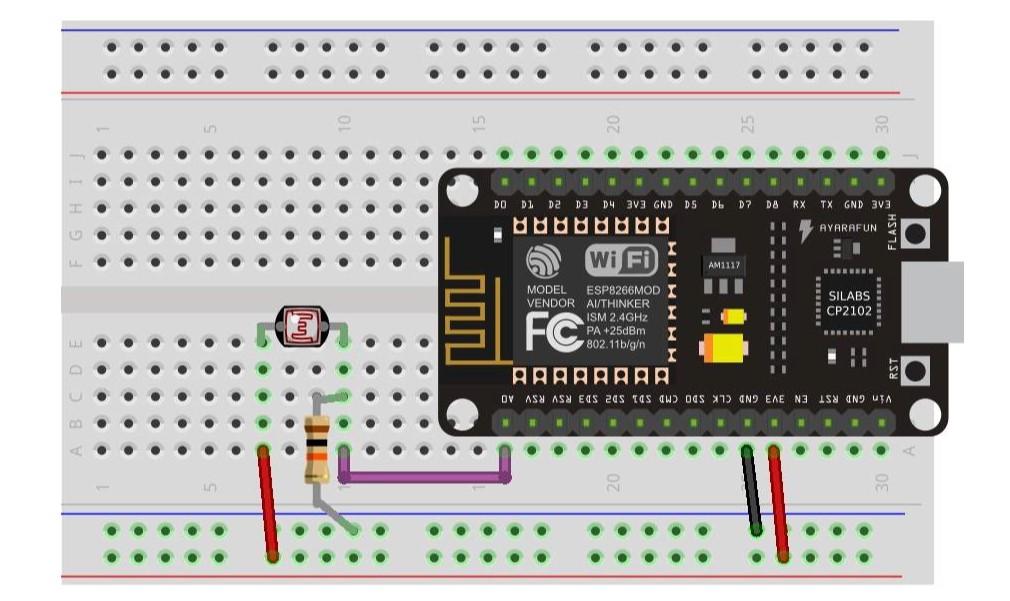
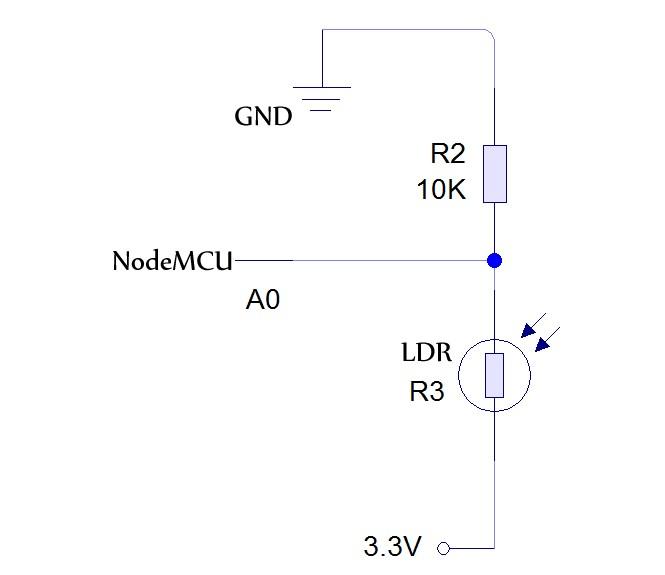
**Aim:**

To use Light Sensor using Node MCU

**Hardware Requirements:**

* NodeMCU
* LDR/PhotoResistor
* Bread Board
* Jumper Wires
* Micro USB Cable
* Arduino IDE (with ESP8266 Library installed)
* 200 ohm – 1K ohm resistor x 1
* PC

**Circuit Diagram:**

**Code:**

void setup() {

Serial.begin(9600); // initialize serial communication at 9600 BPS

}

void loop() {

int sensorValue = analogRead(A0); // read the input on analog pin 0

float voltage = sensorValue \* (5.0 / 1023.0); // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V)

Serial.println(voltage); // print out the value you read

}

**Result:**

Light sensor has been used successfully using Node MCU and the results are verified.

**Ex : 5 LDR Sensor**

**Aim:**

To print the luminous intensity level using LDR Sensor

**Code:**

void setup() {

// LDR sensor Light dependent resistor

Serial.begin(9600);

pinMode(A0,INPUT);

}

void loop() {

int x=analogRead(A0);

int y = x \*(3.3/1024);

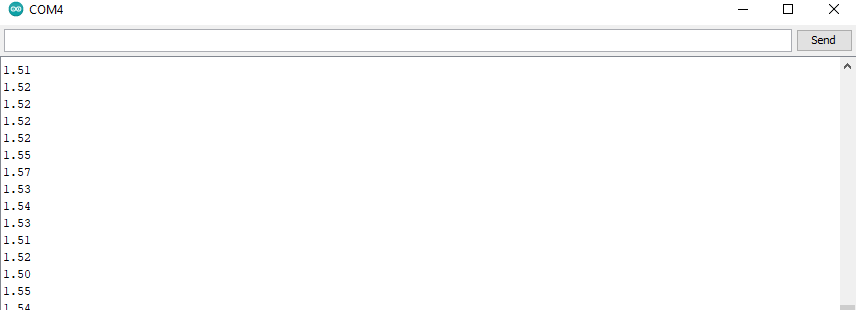
Serial.println(x);

Serial.println(y);

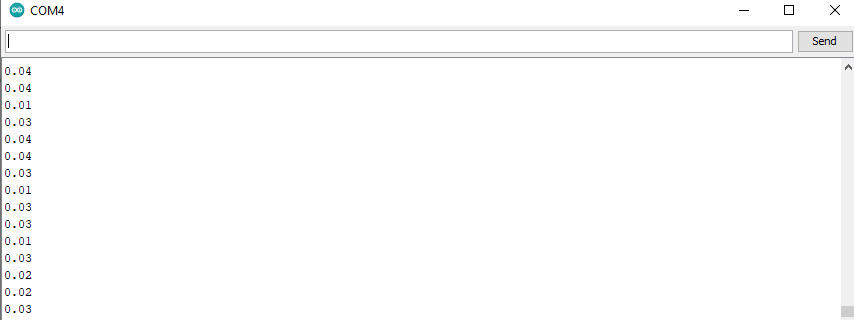
}

**Output:**

with light



without light



**Result:**

LDR Sensor has been used successfully using Node MCU and the results are verified.

**Ex: 6 Proximity Detection**

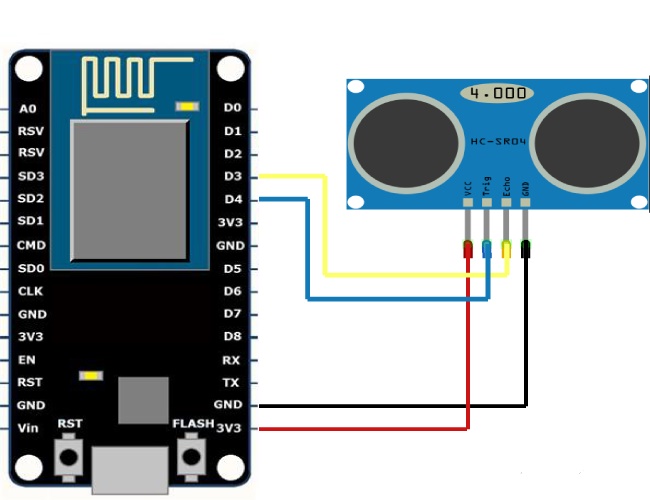
**Aim:**

To detect the location using ultrasonic sensor

**Hardware Required:**

* NodeMCU
* HC-SR04 (Ultra-sonic Sensor)
* Bread Board
* Jumper Wires
* Micro USB Cable

**Circuit Diagram:**



**Code: (1)**

void setup() {

// ultrosonic HCSR -04 - proximity detection

pinMode(D6,OUTPUT); //trigger pin

pinMode(D7,INPUT);//Echopin //rcr

}

void loop() {

// put your main code here, to run repeatedly:

//trigger sonic waves

digitalWrite(D6,LOW);

delayMicroseconds(2);

digitalWrite(D6,HIGH);

delayMicroseconds(10);

// receive ECHO and find DISTANCE

long duration= pulseIn(D7,HIGH);

float dist= duration \* 0.034/2;

Serial.println("Distance is...");

Serial.println(dist);

delay(2000);

}

**Code:(2)**

// defines pins numbers

const int trigPin = 2; //D4

const int echoPin = 0; //D3

// 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

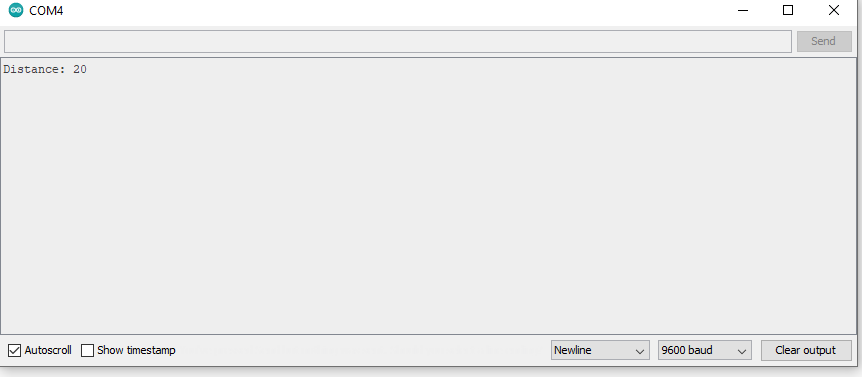
Serial.print("Distance: ");

Serial.println(distance);

delay(2000);

}

**Output:**



**Result:**

Proximity Detection using Ultra Sonic Sensor has been used successfully using Node MCU and the results are verified.

**Ex : 7 Servo Motor**

**Aim:**

To push or rotate an object with great precision at specific angles or distance using servo motor

**Code:**

#include <Servo.h>

Servo newservo1;//define a name for servo

void setup() {

// put your setup code here, to run once:

newservo1.attach(D6);

}

void loop() {

// put your main code here, to run repeatedly:

newservo1.write(5);

delay(1000);

newservo1.write(10);

delay(1000);

}

**Result:**

Servo motor has been used successfully rotated using Node MCU and the results are verified

**Ex : 8 Potentiometer with servo motor**

**Aim:**

To use a potentiometer with servo motor

**Code:**

# include<Servo.h>

Servo mew 1;

Setup ()

New1.attach(D6);

Serial.begin(9600);

pinMode(A0, INPUT);

loop()

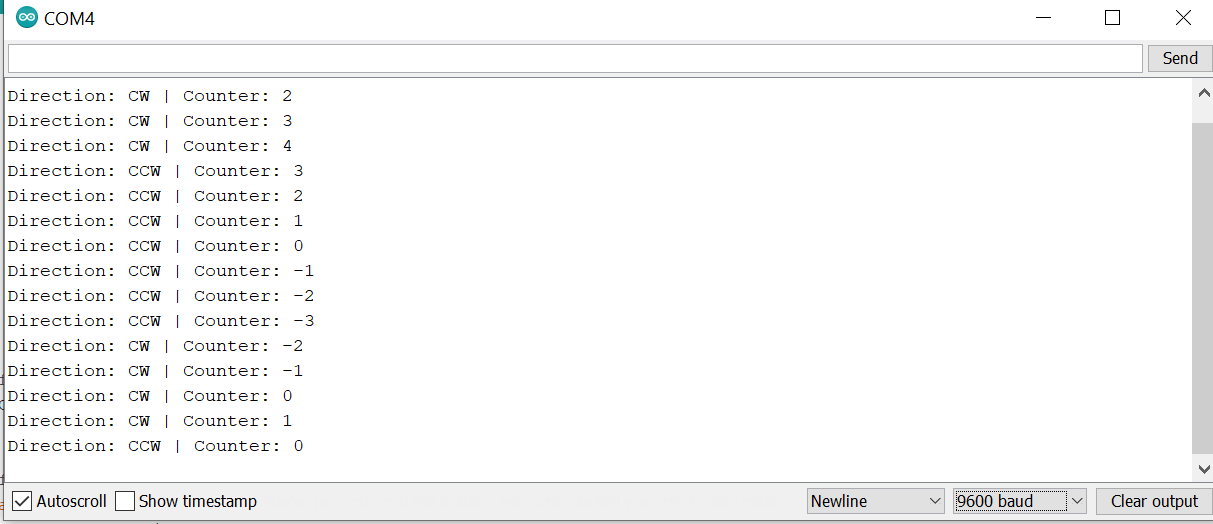
float x=analogRead(A0);

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

new1.write (x);

delay(15);}

**Output:**



**Result:**

Servo motor with potentiometer has been used successfully rotated using Node MCU and the results are verified

**Ex : 9 IoT with cloud**

**Aim:**

To connect with Internet through WIfi Acces point using Node MCU

**Code:**

#include<ESP8266WIFI>

Char\* SSid =”rajesh”;

Char\* Password =”Koushan”

Void setup()

WiFi.begin(SSid, Password);

Serial.begin(115200);

Serial.print(“Connecting: ”);

While (WiFi.status()!= WL\_CONNECTED)

{

Serial.print(“Waiting to connect”)

While(WiFi.Status()!=WL\_CONNECTED)

{

Serial. Print(“Waiting to connect”);

delay(1000);

}

Serial.println(‘\n’);

Serial.println (“Connection established”);

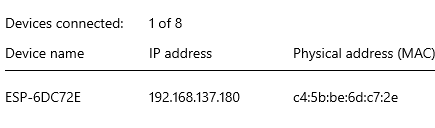
Serial.println (“IP Address\t”);

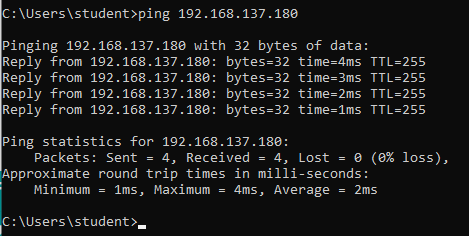
Serial.println(WiFi.LocalIP());

}

**Output:**

\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**Result:**

Connecting with internet has been done successfully using Node MCU and the results are verified

**Ex : 10 Making a node as Server**

**Aim:**

To make a node as a server using Node MCU

**Code:**

#include<ESP8266WiFi>

#include<ESP8266Webserver.h>

ESP8266Webserver server(80)

Char\* ssid, Char pass;

Void setup()

WiFi.begin(SSid, Password);

Serial.begin(115200);

Serial.print(“Connecting: ”);

While (WiFi.status()!= WL\_CONNECTED)

{

Serial.print(“Waiting to connect”)

While(WiFi.Status()!=WL\_CONNECTED)

{

Serial. Print(“Waiting to connect”);

delay(1000);

}

Serial.println(‘\n’);

Serial.println (“Connection established”);

Serial.println (“IP Address\t”);

Serial.println(WiFi.LocalIP());

}

Server.on(%[]() { Server.send()

}

**Result:**

Making a node has been done successfully using Node MCU and the results are verified

**Ex : 11 Interfacing a GPS**

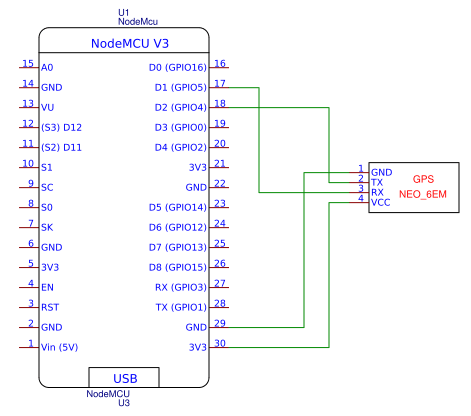
**Aim:**

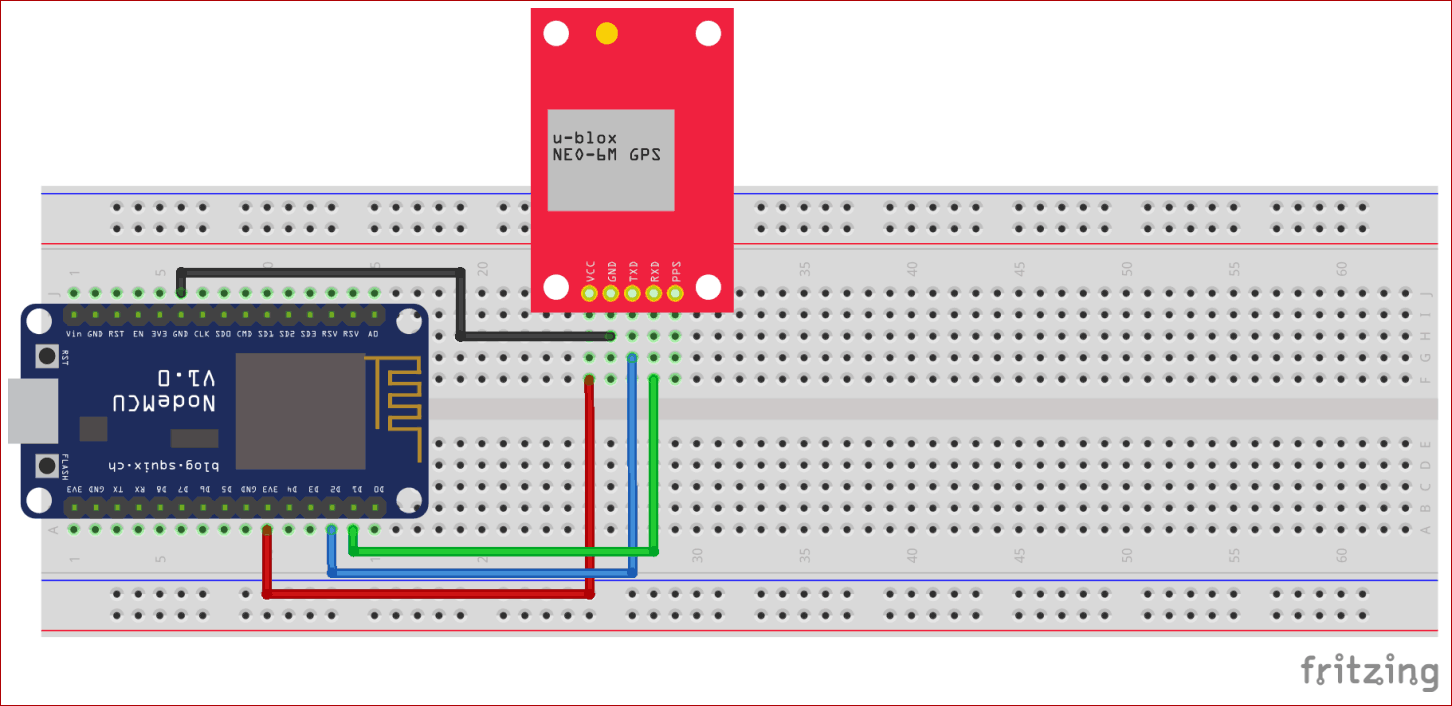
To find the present location using GPS module

**Hardware Required:**

* NodeMCU ESP8266
* GPS module
* Bread Board
* Jumper wires

**Circuit Diagram:**





**Code:**

#include <TinyGPS++.h> // library for GPS module

#include <SoftwareSerial.h>

#include <ESP8266WiFi.h>

TinyGPSPlus gps; // The TinyGPS++ object

SoftwareSerial ss(4, 5); // The serial connection to the GPS device

const char\* ssid = "Onlilo\_SP"; //ssid of your wifi

const char\* password = "ArduinoUno"; //password of your wifi

float latitude , longitude;

int year , month , date, hour , minute , second;

String date\_str , time\_str , lat\_str , lng\_str;

int pm;

WiFiServer server(80);

void setup()

{

Serial.begin(115200);

ss.begin(9600);

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password); //connecting to wifi

while (WiFi.status() != WL\_CONNECTED)// while wifi not connected

{

delay(500);

Serial.print("."); //print "...."

}

Serial.println("");

Serial.println("WiFi connected");

server.begin();

Serial.println("Server started");

Serial.println(WiFi.localIP()); // Print the IP address

}

void loop()

{

while (ss.available() > 0) //while data is available

if (gps.encode(ss.read())) //read gps data

{

if (gps.location.isValid()) //check whether gps location is valid

{

latitude = gps.location.lat();

lat\_str = String(latitude , 6); // latitude location is stored in a string

longitude = gps.location.lng();

lng\_str = String(longitude , 6); //longitude location is stored in a string

}

if (gps.date.isValid()) //check whether gps date is valid

{

date\_str = "";

date = gps.date.day();

month = gps.date.month();

year = gps.date.year();

if (date < 10)

date\_str = '0';

date\_str += String(date);// values of date,month and year are stored in a string

date\_str += " / ";

if (month < 10)

date\_str += '0';

date\_str += String(month); // values of date,month and year are stored in a string

date\_str += " / ";

if (year < 10)

date\_str += '0';

date\_str += String(year); // values of date,month and year are stored in a string

}

if (gps.time.isValid()) //check whether gps time is valid

{

time\_str = "";

hour = gps.time.hour();

minute = gps.time.minute();

second = gps.time.second();

minute = (minute + 30); // converting to IST

if (minute > 59)

{

minute = minute - 60;

hour = hour + 1;

}

hour = (hour + 5) ;

if (hour > 23)

hour = hour - 24; // converting to IST

if (hour >= 12) // checking whether AM or PM

pm = 1;

else

pm = 0;

hour = hour % 12;

if (hour < 10)

time\_str = '0';

time\_str += String(hour); //values of hour,minute and time are stored in a string

time\_str += " : ";

if (minute < 10)

time\_str += '0';

time\_str += String(minute); //values of hour,minute and time are stored in a string

time\_str += " : ";

if (second < 10)

time\_str += '0';

time\_str += String(second); //values of hour,minute and time are stored in a string

if (pm == 1)

time\_str += " PM ";

else

time\_str += " AM ";

}

}

WiFiClient client = server.available(); // Check if a client has connected

if (!client)

{

return;

}

// Prepare the response

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n <!DOCTYPE html> <html> <head> <title>GPS DATA</title> <style>";

s += "a:link {background-color: YELLOW;text-decoration: none;}";

s += "table, th, td </style> </head> <body> <h1 style=";

s += "font-size:300%;";

s += " ALIGN=CENTER> GPS DATA</h1>";

s += "<p ALIGN=CENTER style=""font-size:150%;""";

s += "> <b>Location Details</b></p> <table ALIGN=CENTER style=";

s += "width:50%";

s += "> <tr> <th>Latitude</th>";

s += "<td ALIGN=CENTER >";

s += lat\_str;

s += "</td> </tr> <tr> <th>Longitude</th> <td ALIGN=CENTER >";

s += lng\_str;

s += "</td> </tr> <tr> <th>Date</th> <td ALIGN=CENTER >";

s += date\_str;

s += "</td></tr> <tr> <th>Time</th> <td ALIGN=CENTER >";

s += time\_str;

s += "</td> </tr> </table> ";

s += "</body> </html>"

client.print(s); // all the values are send to the webpage

delay(100);

}

**Result:**

Location has been successfully identified using GPS and Node MCU and the results are verified.

**Ex : 12 Interfacing Bluetooth**

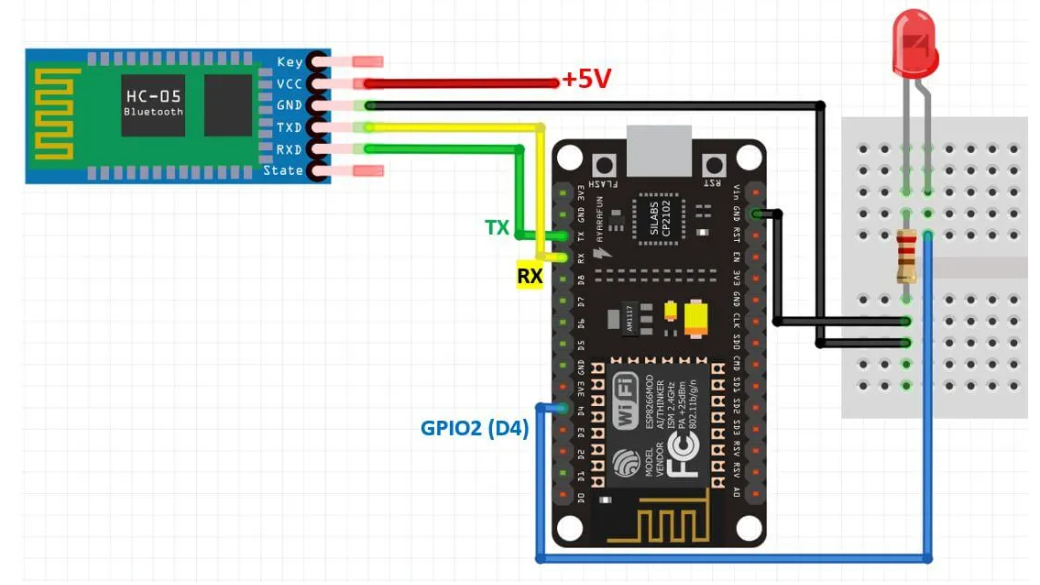
**Aim:**

To interface Bluetooth module using Node MCU

**Hardware Required:**

* ESP8266 Node MCU
* Bluetooth Module
* LED
* 1K-ohm resistor
* Connecting Wires
* Breadboard

**Circuit Diagram:**



**Code:**

int led\_pin = 2;

void setup() {

pinMode(led\_pin, OUTPUT);

Serial.begin(9600);

}

void loop() {

if (Serial.available())

{

char data\_received;

data\_received = Serial.read();

if (data\_received == 'O')

{

digitalWrite(led\_pin, HIGH);

Serial.write("LED is now ON!\n");

}

else if (data\_received == 'X')

{

digitalWrite(led\_pin, LOW);

Serial.write("LED is now OFF!\n");

}

else

{

Serial.write("Specify correct option\n");

}

}

}

**Result:**

Bluetooth module has been successfully interfaced with Node MCU and the results are verified.

**Ex : 13 Interfacing GSM**

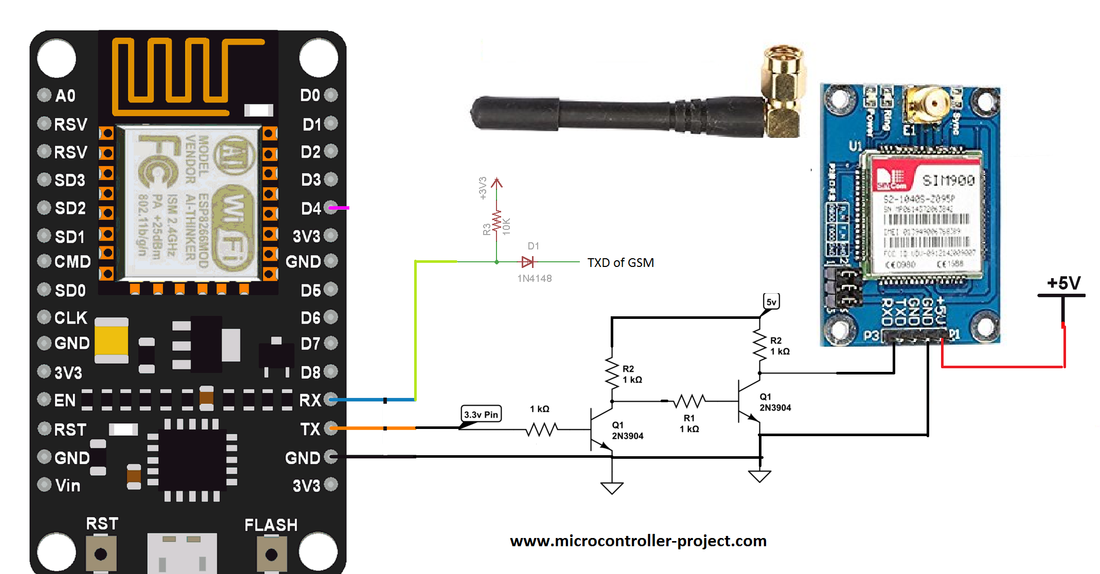
**Aim:**

To interface with GSM and send message using Node MCU

**Hardware Required:**

* GSM SIM900A
* Node MCU
* Jumper Wire
* Power adapter 5V
* SIM card
* Breadboard

**Circuit Diagram :**



**Code:**

void setup()

{

//Begin nodemcu serial-0 channel

Serial.begin(9600);

}

void loop()

{

Serial.print("AT"); //Start Configuring GSM Module

delay(1000); //One second delay

Serial.println();

Serial.println("AT+CMGF=1"); // Set GSM in text mode

delay(1000); // One second delay

Serial.println();

Serial.print("AT+CMGS="); // Enter the receiver number

Serial.print("\"+91XXXXXXXXXX\"");

Serial.println();

delay(1000);

Serial.print("IOT LAB"); // SMS body - Sms Text

delay(1000);

Serial.println();

Serial.write(26); //CTRL+Z Command to send text and end session

while(1); //Just send the text ones and halt

}

**Result:**

Interface with GSM and sending a SMS using Node MCU has been done successfully and the results are verified.