**WEB CONTROLLED SERVO MOTOR USING NODEMCU**

***A MINOR PROJECT REPORT***

***OF***

***IOT***

***Submitted by***

M. Sai Kishore (171FA05305)

R.V.L. Karthik (171FA05329)

S. Tushar(171FA05362)

**in**

**Electronics and Communication Engineering**

Under the Esteemed Guidance of

Mr. Vishnu S



(ACCREDITED BY **NAAC** WITH **‘A’**GRADE)

MHRD**NIRF 88**RANK

**DEPARTMENT OF**

**ELECTRONICS & COMMUNICATION ENGINEERING**

**VFSTR, VADLAMUDI**

**GUNTUR-522213, ANDHRA PRADESH, INDIA**

April 2019

**CERTIFICATE**

This is to certify that the minor project report entitled **“WEB CONTROLLED SERVO MOTOR USING NODEMCU”** that is being submitted by **M. Sai Kishore, R.V.L. Karthik, S. Tushar** bearing **Regd. Nos.** **171FA05305, 171FA05329, 171FA05329** in partial fulfilment for the award of II year II semester B.Tech degree in Electronics and Communication Engineering to Vignan’s Foundation for Science Technology and Research , is a record of work carried out by him under the guidance of Mr. S. Vishnu of ECE Department.

Signature of the faculty guide Signature of Head of Department

Mr. S Vishnu Mr.T. Pitchaiah Ph.D., M.E, Asst. Professor  Assoc. Professor

|  |  |  |
| --- | --- | --- |
| S.NO | CONTENTS | PAGE NUMBER |
| 1 | Abstract | 4 |
| 2 | Introduction | 5 |
| 3 | Components required and explanation | 6 - 8 |
| 4 | Code | 8 - 14 |
| 5 | Circuit diagram | 15 |
| 7 | Connections and working | 16 |
| 6 | Advantages, disadvantages and applications | 17 |
| 7 | Conclusion | 18 |
| 8 | references | 19 |

**Abstract:**

In day to day life automation plays an important role in everyone’s life. In some remote areas we require to operate some motors so by using web-controlled system we can control the motor through a webpage. In this project we are controlling the servo motor position through a Web Page within the same WiFi Network.

**INTRODUCTION**

Servo Motors are one of the most commonly used motors in both industries and DIY Projects. The servo motors have high accuracy and they can revolve at accurate wanted angle. In this we are creating a simple Web Page (HTML) with a slider. When this web page is accessed from a laptop or mobile phone that is connected to the same WiFi Network as the ESP8266, we can control the position of the Servo Motor by adjusting the knob.

In this we are using directly NODEMCU by avoiding Arduino and potentiometer. Normally, in order to control a Servo Motor with Arduino all you need is a Servo Motor, Arduino and a Potentiometer. Depending on the position of the POT, we have to control the motor by generating pulse width modulated signal. The PWM value of the Servo Motor input changes and consequently the position of the Servo Motor’s shaft changes. All this process is done manually by a person. In IOT the main target is Automation so we have to avoid the person who is controlling it manually and replace him/her with controlling equipment.

In case of a Web Controlled Servo Motor using ESP8266, ESP8266 drives the Servo Motor but the input isn’t from a POT but a Web Page we created with AJAX.

The slider in the web page sends the angle values and is received by the NodeMCU, which acts as a Web Server. The NodeMCU, upon receiving the value from the slider moved in the webpage. The NodeMCU acts as webserver and it generates transmits corresponding PWM to servo, which then changes the position of the Servo Motor according to the value.

One important thing to remember here is that both the Server and client should be on the same network i.e. the ESP8266, which acts as the Server, and the Laptop (or a Mobile Phone), which is the client, must be connected to the same WiFi network.

**COMPONENTS REQUIRED:**

1. ESP8266 based Board (NodeMCU)
2. Servo Motor
3. Jumper Wires
4. Laptop

**ESP8266 NodeMCU**:

**NodeMCU** is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the **ESP8266** Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.



ESP8266 NODEMCU



NODEMCU PIN DIAGRAM

|  |  |  |
| --- | --- | --- |
| **Pin Category** | **Name** | **Description** |
| Power | Micro-USB, 3.3V, GND, Vin | **Micro-USB:** NodeMCU can be powered through the USB port    **3.3V:** Regulated 3.3V can be supplied to this pin to power the board    **GND:** Ground pins    **Vin:**External Power Supply |
| Control Pins | **EN, RST** | The pin and the button resets the microcontroller |
| Analog Pin | A0 | Used to measure analog voltage in the range of 0-3.3V |
| GPIO Pins | GPIO1 to GPIO16 | NodeMCU has 16 general purpose input-output pins on its board |
| SPI Pins | SD1, CMD, SD0, CLK | NodeMCU has four pins available for SPI communication. |
| UART Pins | TXD0, RXD0, TXD2, RXD2 | NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program. |
| I2C Pins |  | NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C. |

**PINS DESCRIPTION:**

**SERVO MOTOR:**

Stepper motors are DC motors that move in discrete steps. They have multiple coils that are organized in groups called "phases". By energizing each phase in sequence, the motor will rotate, one step at   
a time. With a computer controlled stepping you can achieve very precise positioning and/or speed control. For this reason, stepper motors are the motor of choice for many precision motion control applications. Stepper motors come in many different sizes and styles and electrical characteristics. This guide details what you need to know to pick the right motor for the job.



Stepper motor

**ESP8266 Servo Arduino Code**

The Arduino IDE Code for the Web Controlled Servo using ESP8266 tutorial is given below. Program is divided in two parts Web Page (**index.h**) and Main Hardware Code **ESPservo.ino**

Before uploading code make changes in SSID and Password as per your wifi

**Arduino code to configure NodeMCU**

#include <ESP8266WiFi.h>

#include <ESP8266WebServer.h>

#include <Servo.h>

#include "index.h";

#define LED 2

#define ServoPin 14 //D5 is GPIO14

//WiFi Connection configuration

const char \*ssid = "WIFI123";

const char \*password = "123456789";

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

// twelve servo objects can be created on most boards

ESP8266WebServer server(80);

void handleServo(){

String POS = server.arg("servoPOS");

int pos = POS.toInt();

myservo.write(pos); // tell servo to go to position

delay(15);

Serial.print("Servo Angle:");

Serial.println(pos);

digitalWrite(LED,!(digitalRead(LED))); //Toggle LED

server.send(200, "text/plane","");

}

void handleRoot() {

String s = MAIN\_page; //Read HTML contents

server.send(200, "text/html", s); //Send web page

}

//================================================// Setup

//================================================

void setup() {

delay(1000);

Serial.begin(115200);

Serial.println();

pinMode(LED,OUTPUT);

myservo.attach(ServoPin); // attaches the servo on GIO2 to the servo object

//Connect to wifi Network

WiFi.begin(ssid, password); //Connect to your WiFi router

Serial.println("");

// Wait for connection

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

//If connection successful show IP address in serial monitor

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("192.168.43.225");

Serial.println(WiFi.localIP()); //IP address assigned to your ESP

//Initialize Webserver

server.on("/",handleRoot);

server.on("/setPOS",handleServo); //Sets servo position from Web request

server.begin();

}

//================================================

// LOOP

//================================================

void loop() {

server.handleClient();

}

//================================================

##### index.h

Press**Ctrl+Shift+N**give name**index.h**and copy paste below code**.**

**HTML CODE TO CREATE WEBPAGE:**

const char MAIN\_page[] PROGMEM = R"=====(

<!DOCTYPE html>

<html>

<head>

<title>ESP8266 Servo | Circuits4you.com</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<style>

.angle{

  width: 79px;

  height: 50px;

  position: absolute;

  vertical-align: middle;

  margin-top: 50px;

  margin-left: -114px;

  border: 0px none;

  background: rgba(0, 0, 0, 0) none repeat scroll 0% 0%;

  font: normal normal bold normal 20px Arial;

  text-align: center;

  color: rgb(34, 34, 34);

  padding: 0px;

}

.spd{

  width: 79px;

  height: 50px;

 position: absolute;

  vertical-align: middle;

  margin-top: 50px;

  margin-left: -114px;

  border: 0px none;

  background: rgba(0, 0, 0, 0) none repeat scroll 0% 0%;

  font: normal normal bold normal 50px Arial;

  text-align: center;

  color: rgb(34, 34, 34);

  padding: 0px;

}

.imageDiv{

    padding: 5%;

}

.flx{

  display: flex;

}

</style>

<body>

<div style="width:100%;">

<div style="width:50%;  margin: 0 auto;">

  <h3>Circuits4you.com</h3>

  <h4>ESP8266 Servo Motor Control Demo</h4>

</div>

</div>

<div style="width: 50%; margin: 0 auto;" class="flx">

  <svg viewBox="0 0 500 500" width="250" height="250" id="mySVG" style="background:#fff; border: 1px solid black">

  <path fill="none" stroke="#30D8D9"  stroke-width="50" d="M 376.79805300444093 404.6682053761632 A 200 200 0 1 0 121.44247806269212 403.2088886237956"></path>

  <path id="arc1" fill="none" stroke="#00A8A9" stroke-width="50" style="stroke-linecap: round;"/>

  <text x="230" y="260" fill="#777" id="angle" class="spd">0</text>

  <text x="200" y="300" fill="#777" class="angle">Servo Angle</text>

  </svg>

</div>

<script>

function sendData(pos) {

  var xhttp = new XMLHttpRequest();

  xhttp.onreadystatechange = function() {

    if (this.readyState == 4 && this.status == 200) {

    console.log(this.responseText);

    }

  };

  xhttp.open("GET", "setPOS?servoPOS="+pos, true);

  xhttp.send();

}

function polarToCartesian(centerX, centerY, radius, angleInDegrees) {

  var angleInRadians = (angleInDegrees-90) \* Math.PI / 180.0;

  return {

    x: centerX + (radius \* Math.cos(angleInRadians)),

    y: centerY + (radius \* Math.sin(angleInRadians))

  };

}

function describeArc(x, y, radius, startAngle, endAngle){

    var start = polarToCartesian(x, y, radius, endAngle);

    var end = polarToCartesian(x, y, radius, startAngle);

    var largeArcFlag = endAngle - startAngle <= 180 ? "0" : "1";

    var d = [

        "M", start.x, start.y,

        "A", radius, radius, 0, largeArcFlag, 0, end.x, end.y

    ].join(" ");

    return d;

}

window.onload = function() {

  document.getElementById("arc1").setAttribute("d", describeArc(250, 250, 200, 220, 210));

};

var svg  = document.getElementById("mySVG");

    pt   = svg.createSVGPoint(),

svg.addEventListener('mousedown',function(evt){

  var loc = cursorPoint(evt);

  var degrees = Math.atan2(loc.x-250,loc.y-250)\*180/Math.PI + 90;

  var offset = 220;

  degrees = (degrees + 90)

  degrees = degrees + offset;

  if(degrees > 360)

  {

    degrees = degrees - 360;

  }

  degrees = 360 - degrees;

  angle = degrees + offset;

  console.log(degrees, angle);

   if(degrees<281

{

    document.getElementById("arc1").setAttribute("d", describeArc(250, 250, 200, offset, angle));

    var servoAng = Math.round(((angle - 220)/280) \* 100);

    document.getElementById("angle").innerHTML=servoAng;

    sendData(servoAng);

  }

});

// Get point in global SVG space

function cursorPoint(evt){

  pt.x = evt.clientX; pt.y = evt.clientY;

  return pt.matrixTransform(svg.getScreenCTM().inverse());

}

</script>

</body>

</html>

)=====";

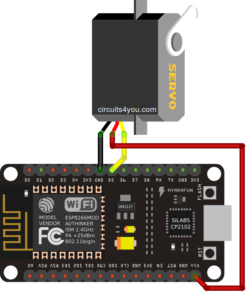
To write code we are using arduino software and selecting ESP8266 board and write it. After completion of code we are compiling it and rectifying the code if any errors are detected and execute it.

After completion of execution we are going to dump the code in NodeMcu.

Again, we are writing a code with (.h) extension i.e., writing HTML code. This code is used for creation of web page with slider.

**Circuit Diagram:**

The circuit diagram for Web Controlled Servo using ESP8266 WiFi Module is given in the image below



**CONNECTIONS:**

Pin D5 of NodeMCU is connected to the orange wire of the servo motor and this pin provide the necessary PWM signal to the Servo motor depending on the pwm signal generated by NodeMCU the servo motor position is determined. The red wire of servo is connected to the 5v (Vin pin). And the brown wire is connected to ground.

**WORKING:**

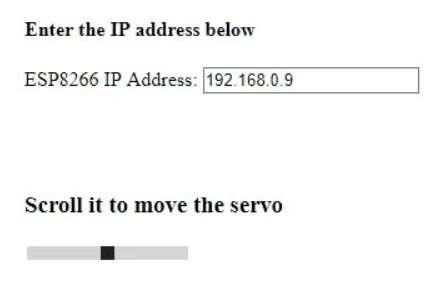
The program is responsible for the operation. In this we are dumping the code into NodeMCU and connected it to a wifi network. The stepper motor is connected to the NodeMCU. Ensure that laptop also connected to the same wifi network. After the code is uploaded, if you open the serial monitor, you can see the status of the ESP8266 WiFi Module. After that you have to follow the below steps:

Set mode to station mode

Connect ESP8266 to WiFi and start web server.

Now we can proceed for web control.

Now Get IP address of ESP8266 from serial monitor and open it using any web browser. If everything is right, when you change the position of the slider, the position of the servo Motor will be changed.

****

**ADVANTAGES:**

* Stepper motor gives accurate angle.
* Circuit complexity is low.
* We can control it through internet.

**DISADVANTAGES:**

* The devices are connected to same WiFi network.
* Limited range application.

**APPLICATIONS:**

* To control the motor through internet.
* We can operate servo motor present in remote location.
* If it is extended, we can control the robot through internet.

**CONCLUSION:**

By this project we can control the servo motor angle connected with NodeMCU through the internet within the same network. This project can be extended to control servo motors present in robots through internet.

**REFERENCES:**

* <https://circuits4you.com/2019/01/12/esp8266-servo-motor-control/>
* “The Internet of Things” by Samuel Greengard.
* “The Fourth Industrial Revolution” by Klaus Schwab.