ON7WP single axis WIFI rotator controller based on Wemos D1 R1

A very cheap Wireless controller that can be used from anywhere in the world using a Blynk app on your phone.

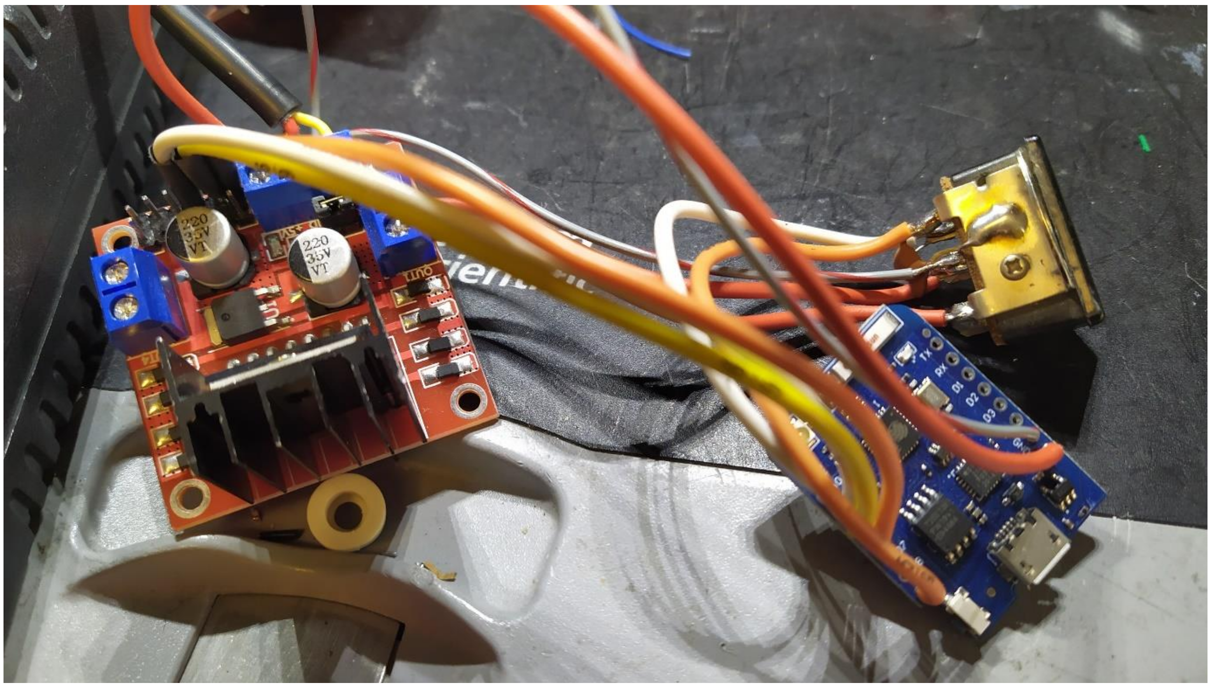
The Hardware are 2 cheap Chinese boards:

- A Wemos Mini D1 ESP8266 based microcontroller
- A L298N Dual motor driver bridge

The software offers remote feedback of the actual rotor position and uses smart motor drive control, slowing down the speed in three steps when the target is almost reached.

The unit can drive any DC motor (Yaesu, Create,….) An AC version is in development.

The Dual motor drives are used in Parallel so the unit can drive up to 3 A motor current.



<u>Motor driver L289N Azimuth and potmeter feedback connections</u>

The Dual motor drives are used in Parallel so the unit can drive up to 3 A motor current.

Watch out as the wires need to be crossed !!!! (see picture)

Azimuth motor connections to L298 Motor driver board:

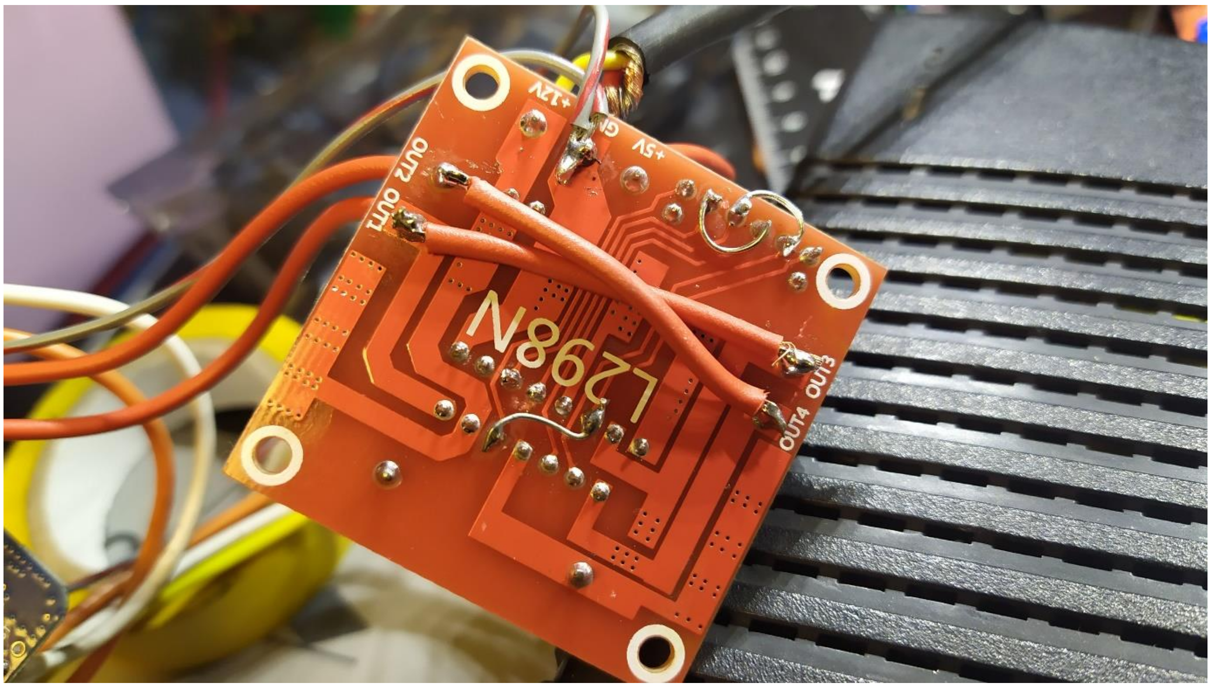
int enA = 12 = D6

int in1 = 14 = D5

int in2 = 16 = D0

ON BOARD ADC INPUT

A0 is the analog in value coming from the rotor potentiometer: **(max 3,3 V !!!)**



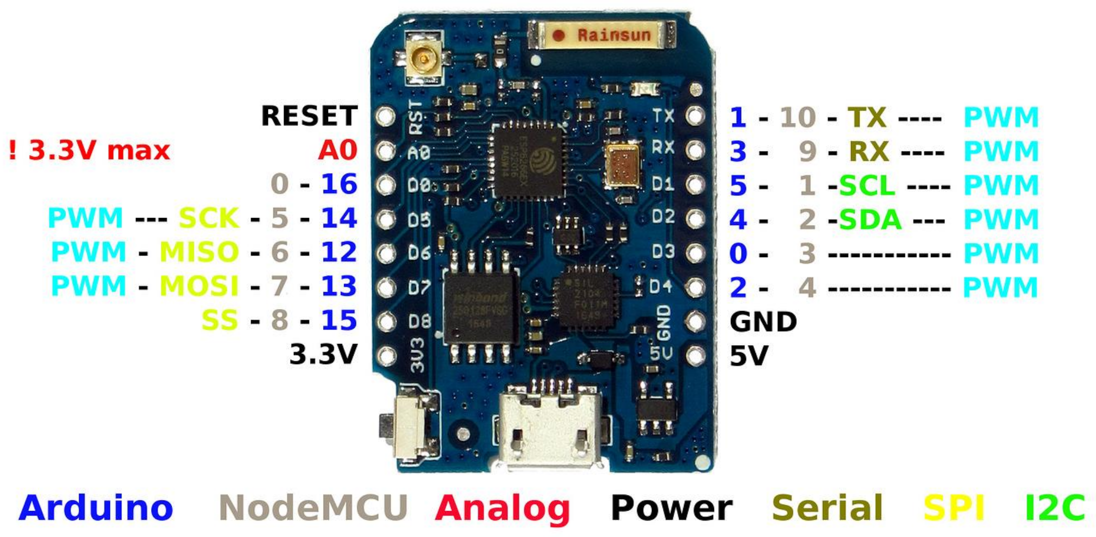
Becareful with the pinning used in the Arduino program, they differ from the naming on the Wemos D1 board !!

As example if you would drive PIN5 in the Arduino software…

> Strangely this activates pin D1 instead of D5.

This might be a clue 😉 … pin #5 in Arduino format is referring to D1 on the board … which is what the board actually responds to, except in limited cases where the Arduino IDE is set for that actual board.

In other words, ignore the silk screened references and follow the Arduino pinouts for that board.



PS, this “issue” of mistakenly using the silkscreened labeling instead of Arduino referenced numbers **IS** mentioned repeatedly in this forum 😜

Don not forget to resolder the jumper next to the external antenna connector if you want to use the external antenna input !

Arduino IDE Program as a text file:

```
#define BLYNK_PRINT Serial

#include <WiFiClient.h>

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>


char auth[] = "Put your Blynk token here";

char ssid[] = "Put your wifi ssid here";

char pass[] = "Put your wpassword here";

int temp;



// Motor Azimuth connections


int enA = 12;


int in1 = 14;


int in2 = 16;




int rotorfeedback; // variable to store the value coming from the analog pin


double act;// variable to store the position value coming from the conversion formula


double error;// variable to store the pointing error


int heading; // variable to store the position value coming from the remote control
```

```
double set;// variable to store the position value coming from the conversion formula

void setup() {

// Set all the motor control pins to outputs

pinMode(enA, OUTPUT); // PWM speed control ESP8266 uses 10 BIT PWM !!

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

// Turn off motors - Initial state

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

// Display wifi connect status on serial monitotr
 Serial.begin(115200);

  delay(10);
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);

  int wifi_ctr = 0;

  while (WiFi.status() != WL_CONNECTED) {

    delay(500);

    Serial.print(".");

  }


  Serial.println("WiFi connected");


Blynk.begin(auth, ssid, pass, "Put your server name here", Port Number);

}


#define HEADING V0

#define ACT V1



BLYNK_WRITE(V0) //writing the wanted direction from input with 20 degrees offset to protect potmeter in rotor

{

temp = param.asInt();

}



// the loop function runs over and over again forever

void loop() {

Blynk.run();

int real=act-20;

Blynk.virtualWrite(ACT, real);


Blynk.virtualWrite(HEADING, temp);

set=temp+20;
```

```
rotorfeedback= analogRead(0); // read the analog in value:

act = (double) rotorfeedback * (400/1024.0); // position conversion formula

if(set < act)

{

TurnCCW();

}

else {

TurnCW();

}

//speedcontrol while reaching target

if ((abs(set-act))<20)

{

analogWrite(enA, 900);

}

if ((abs(set-act))<5)
```

```
{

analogWrite(enA, 800);

}

if ((abs(set-act))<1)

{

analogWrite(enA, 0);
Stop();

}

}

// This function lets you control the motor to turn clockwise

void TurnCW(){

analogWrite(enA, 1023);

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

}

void TurnCCW(){
```

```
analogWrite(enA, 1023);

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

}

void Stop(){

// Turn off motors

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

}
```