ROBOSURGEON

PROJECT REPORT TO BUILD A BLUETHOOTH CONTROLLED CAR

SUBMITTED BY:

Team RoboSergeon

Team Members:

Mahfuz Hasan Reza

Saif Al Shad

Jahrin Binte Zahid

Sarabon Tohura

Sanzida Zaman Mim

SUBMITTED TO:

UIU Mars Rover Team

Submitted on 2th October, 2023

Purpose: Building a bluetooth controlled car.

Components:

ESP32 : Development Board

L298N Motor Driver

4 DC motors & Wheels: drive

1.2 18650 Battery(rechargeable)

Car Chassis

Jumper Wire

If needed: Breadboard(optional) for ease of use.

Components Details:

ESP32: For development puposes, we use this board

L298N motor driver: It controlled the motors

4 DC motors and wheels: It is used for moving the car (driving)

2 18650 Battery(rechargeable): It is used for providing voltage to the circuit

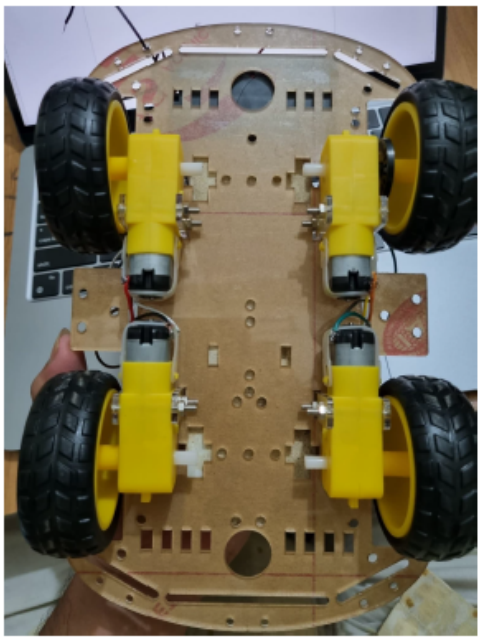
Car chassis (optional): The main body to attach other components (can use any PVC or cardboard as a base)

Jumper Wires: Male-to-male and male-to-female connectors for the circuit (any normal wires can be used, though male-to-female connectors are necessary for connecting with esp)

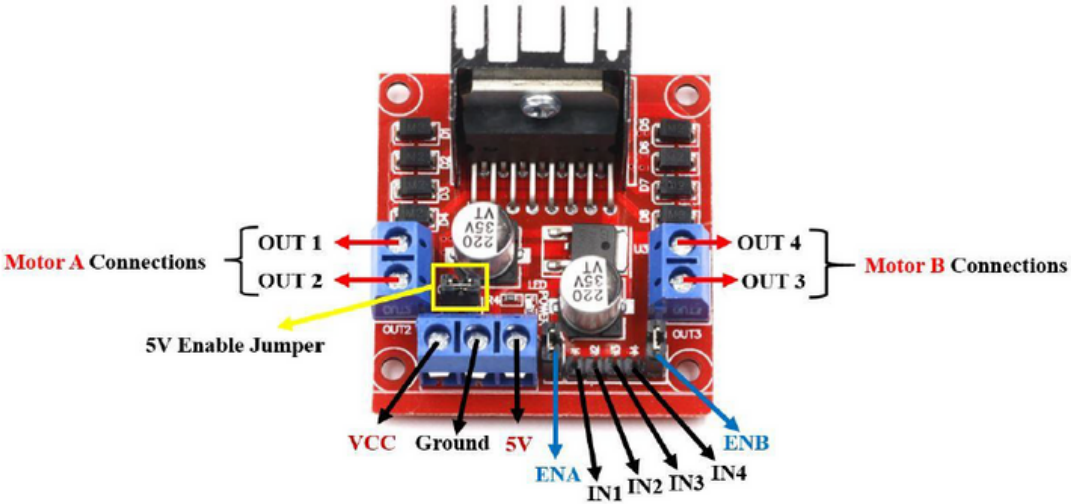
Breadboard(optional): *For connection the wires with others components we used this board*

Following Steps:

1. First we attached the motors to a base. We connected them using the screw provided with the chassis. Glue gun can be used to attach the motors to the base.

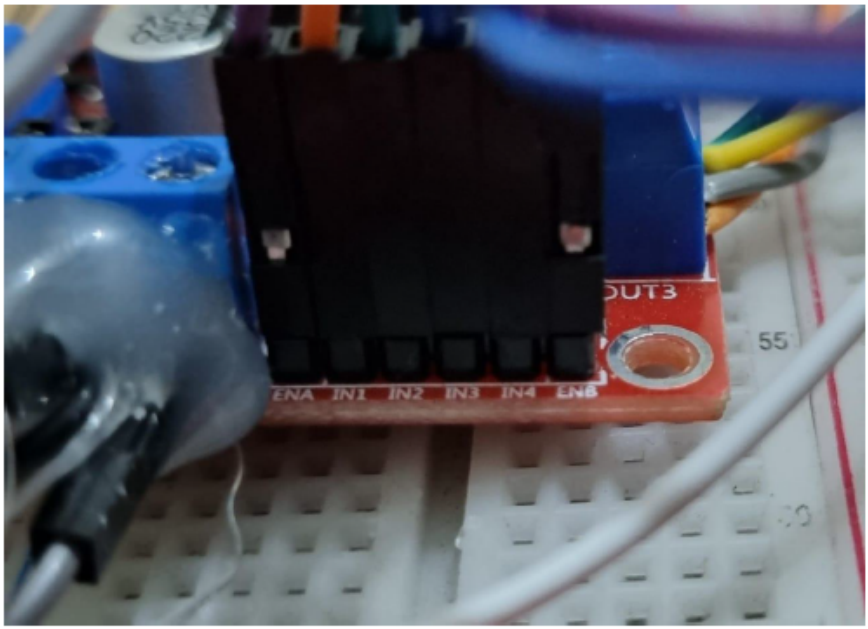


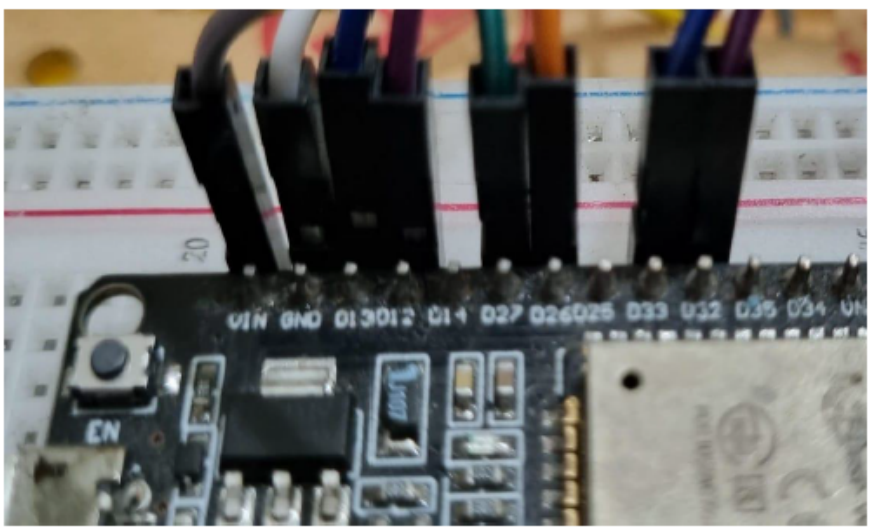
2. Then connected both left motor's positive terminals to "out1" and negative terminals to "out2" of the L298N driver.
3. And repeated the process for the right motors and connected them to "out3" and "out4" of the motor driver.



4. After that we connected the battery with the motor driver. Positive terminal to 12V marked port. We marged the ground of the battery with the ESP32 ground connector to and connected it with the GND marked port. Lastly we connected the 5v from our ESP32 with the 5Vmaked port beside the GND of the driver so that the ESP32 board can take power from the motor driver.

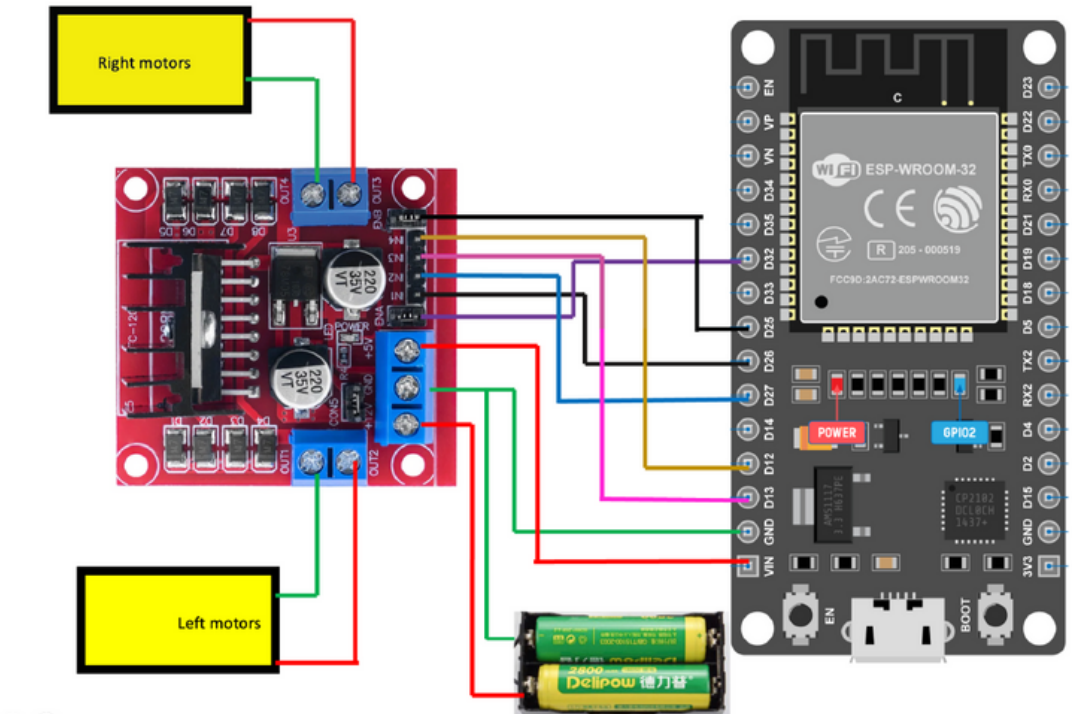
5. We connected the ENA, IN1,IN2,IN3,IN4, ENB ports with the ESP32's D32, D26,D27,D13,D12 & D33 no pin respectively.





6. We removed both of the ENA and ENB jumpers so that we can control the speed of the motors using PWM. If anyone doesn't want to control the speed of the motors then they should leave the jumper on the board.
7. Then we downloaded the "bluetooth RC Controller" app from the play store and started writing our code according to the input values the app passes via bluetooth.

Circuit Diagram:



Internal Control:

- The bluetooth car app sends different characters for different functions and the ESP receives them and controls the motors using the L298N motor driver module.
  Forward → 'F' . Back→ 'B'. Left→ 'L'. Right→ 'R'

- To move forward the In1 and In3 gets HIGH and In2 and In4 gets LOW. and we send the PWM value to both ENA and ENB pins. So, all motors run forward according to the value from PWM.

- To move Backward the In1 and In3 gets LOW and In2 and In4 gets HIGH. We send the PWM value to both ENA and ENB pins. So, all motors run Backward according to the value from PWM.

- To move Left the In1 and In4 gets LOW and In2 and In3 gets HIGH. As a result the left runs backwards and the right motors run forward and the car turns left.

- To move Right the In1 and In4 gets HIGH and In2 and In3 gets LOW. As a result the right motors run backwards and the left motors run forward and the car turns right.

- From the speed setter function of the app we can send values from 1 to 9 and 'q'. So we set different speed values for these inputs in the ESP32. So the car can run at different speeds.

Code Implementation:

```
#include <BluetoothSerial.h>
BluetoothSerial ESP_BT;

#define r_pwm 33
#define l_pwm 32
#define RF 13
#define LF 26
#define RB 12
#define LB 27
#define trigP 5
#define echoP 18
#define SOUND_SPEED 0.034

int incoming;
int speed = 150;
long duration;
float distanceCm;
float calDistance(){
  digitalWrite(trigP, LOW);
  delayMicroseconds(2);
  digitalWrite(trigP, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigP, LOW);

  duration = pulseIn(echoP, HIGH);

  distanceCm = duration * SOUND_SPEED/2;

  delay(10);
  return distanceCm;
}
```

header file

object create

define pin/value name

global variable

function for calculate the distance between front object and ultrasonic sensor

return the distance value in cm

```
void stopC(){
  digitalWrite(RF, LOW);
  digitalWrite(LB, LOW);
  digitalWrite(LF, LOW);
  digitalWrite(RB, LOW);
}
void setup(){
  Serial.begin(9600);
  ESP_BT.begin("roboS");

  pinMode(RF, OUTPUT);
  pinMode(LF, OUTPUT);
  pinMode(LB, OUTPUT);
  pinMode(RB, OUTPUT);
  digitalWrite(RF, LOW);
  digitalWrite(LB, LOW);
  digitalWrite(LF, LOW);
  digitalWrite(RB, LOW);
  Serial.begin(115200);
  pinMode(trigP, OUTPUT);
  pinMode(echoP, INPUT);
}
void loop(){
  if(ESP_BT.available()){
    incoming = ESP_BT.read();
      if(incoming == 'S'){   // stop all
        digitalWrite(RF,LOW);
        digitalWrite(LB,LOW);
        digitalWrite(LF,LOW);
        digitalWrite(RB,LOW);
      }
      if(incoming == 'B'){ // back
        digitalWrite(RF,LOW);
        digitalWrite(LB,HIGH);
        digitalWrite(LF,LOW);
        digitalWrite(RB,HIGH);
        analogWrite(l_pwm,speed);
        analogWrite(r_pwm,speed);
```

function for stop the car

for ultrasonic sensor

for stop the car

for backward the car

```
if(incoming == 'F'){
   if(calDistance()>30){
      digitalWrite(RF, HIGH);
      digitalWrite(LB, LOW);
      digitalWrite(LF, HIGH);
      digitalWrite(RB, LOW);
      analogWrite(l_pwm, speed);
      analogWrite(r_pwm, speed);
   }
   else{
      stopC();
   }
}
if (incoming == 'H'){ // back left
   digitalWrite(RF, HIGH);
   digitalWrite(LB, HIGH);
   digitalWrite(LF, LOW);
   digitalWrite(RB, LOW);
   analogWrite(l_pwm, speed);
   analogWrite(r_pwm, speed);
}
if(incoming == 'L'){ // left
  digitalWrite(RF, HIGH);
  digitalWrite(LB, LOW);
  digitalWrite(LF, HIGH);
  digitalWrite(RB, LOW);
  analogWrite(l_pwm, 100);
  analogWrite(r_pwm, 255);
}
if(incoming == 'G'){  //forward left
  digitalWrite(RF,HIGH);
  digitalWrite(LB,LOW);
  digitalWrite(LF,HIGH);
  digitalWrite(RB,LOW);
  analogWrite(l_pwm,150);
  analogWrite(r_pwm,255);
 }
```

for forward the car

forward distance check

for back left the car

for left move the car

for forward left the car

```
if(incoming == 'J'){ //backright
  digitalWrite(RF,LOW);
  digitalWrite(LB,LOW);
  digitalWrite(LF,HIGH);
  digitalWrite(RB,HIGH);
  analogWrite(l_pwm,speed);
  analogWrite(r_pwm,speed);
 }
if(incoming == 'R'){ // right
  digitalWrite(RF,HIGH);
  digitalWrite(LB,LOW);
  digitalWrite(LF,HIGH);
  digitalWrite(RB,LOW);
  analogWrite(l_pwm,255);
  analogWrite(r_pwm,100);
}

if (incoming == 'I'){ // forward right
  digitalWrite(RF,HIGH);
  digitalWrite(LB,LOW);
  digitalWrite(LF,HIGH);
  digitalWrite(RB,LOW);
  analogWrite(l_pwm,255);
  analogWrite(r_pwm,150);
}
if(incoming == '1'){
  speed=150;
}
if(incoming == '2'){
  speed=160;
}
if(incoming == '3'){
  speed=170;
}
if(incoming == '4'){
  speed=180;
}
if(incoming == '5'){
  speed=190;
}
```

for back right the car

for right move the car

for forward right the car

for manual speed balance the car

```
      if(incoming == '6'){
        speed=200;
      }
      if(incoming == '7'){
        speed=210;
      }
      if(incoming == '8'){
        speed=220;
      }
      if(incoming == '9'){
        speed=235;
      }
      if(incoming == 'q'){
        speed=255;
      }
  }
  delay(20);
}
```

for manual speed balance the car