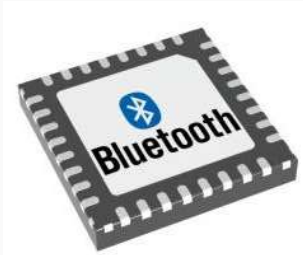
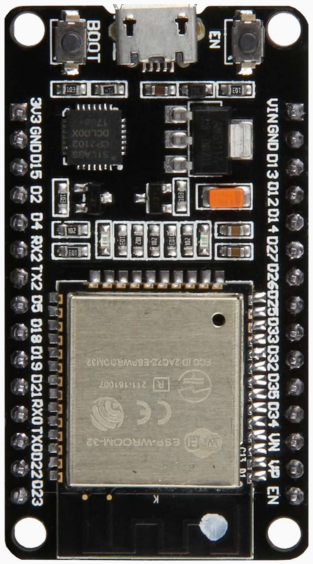
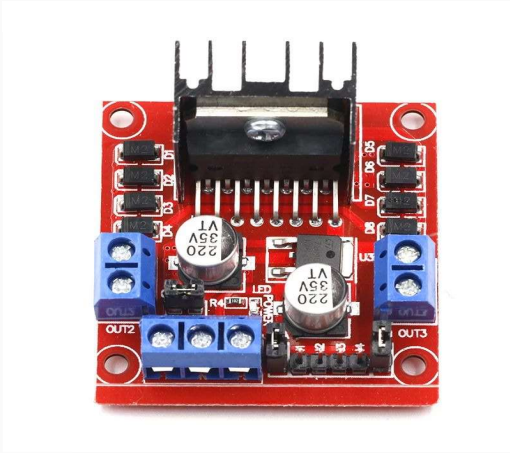
Wireless Car using ESP32

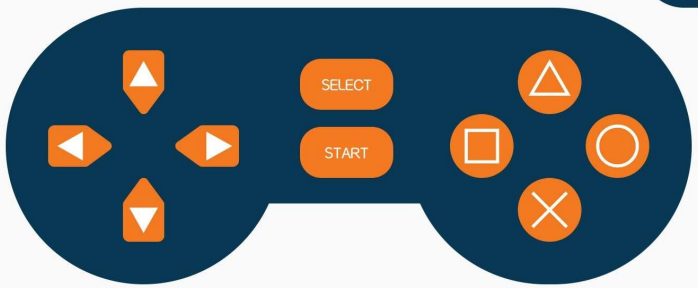
Components

- ESP32 Development Board
- 4xGear Box With Wheels
- L298N Motor Driver
- Bluetooth (esp32 integrated bluetooth)
- Control App (Dabble)





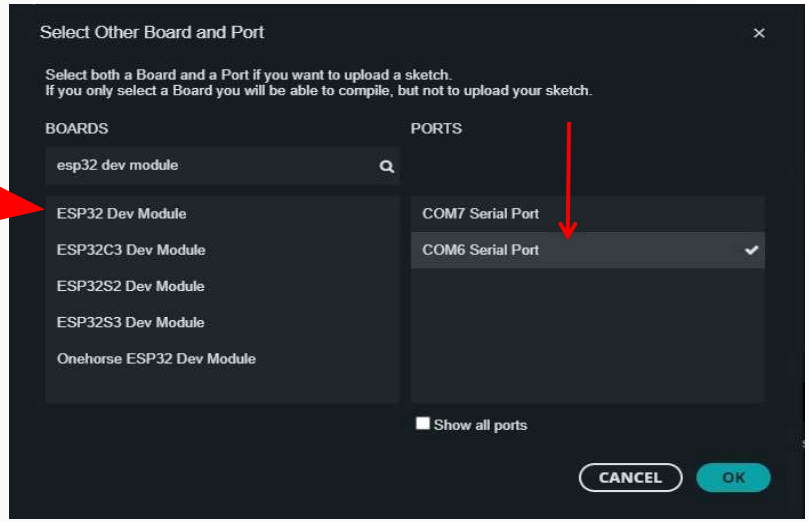


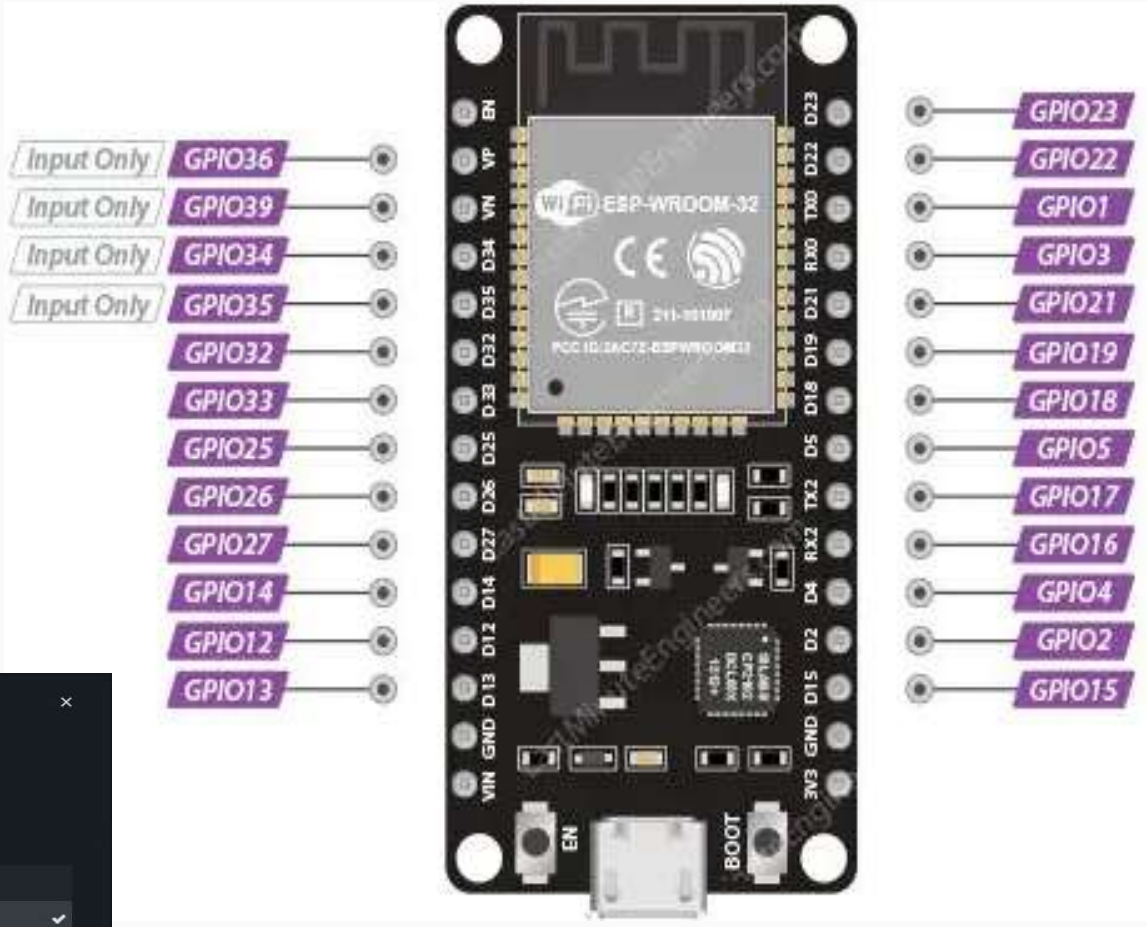


ESP32 Development Board

- After connecting the ESP32 to the desktop click (select board).

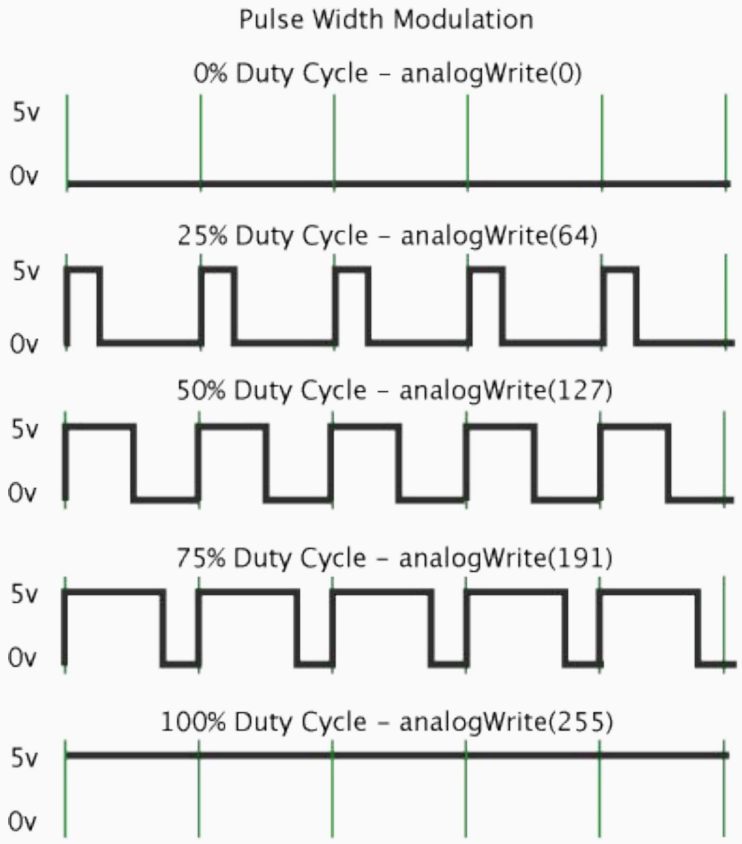
- a window will appear, type 'esp32 dev module' in the search bar and select the port.





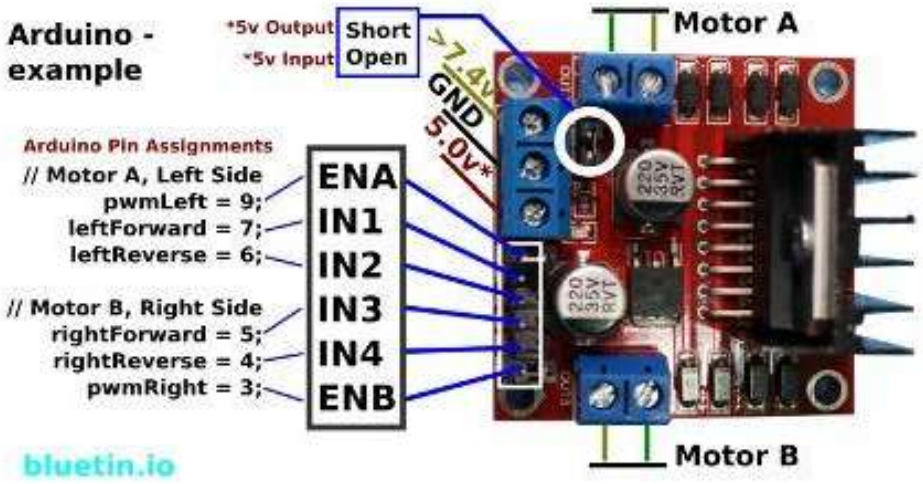
PWM (Pulse Width Modulation)

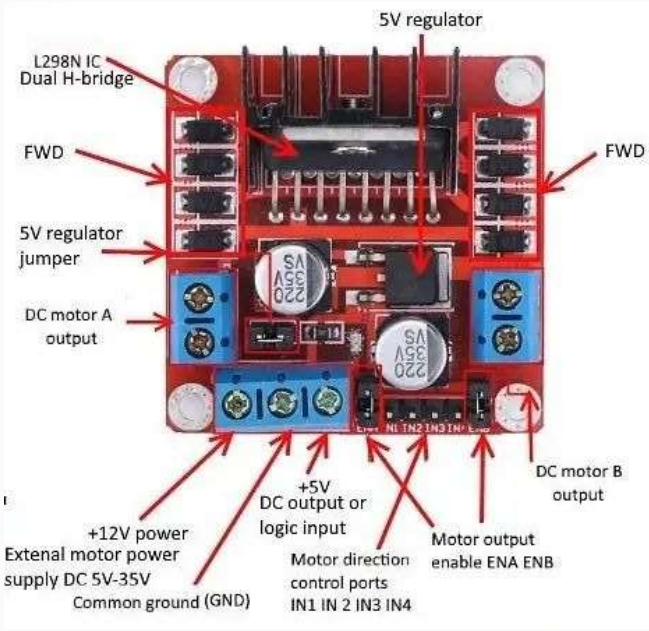
- Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off.
- This on-off pattern can simulate voltages in between the full Vcc of the board (5 V on ESP32) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off.
- The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width.
- If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and Vcc controlling the brightness of the LED.



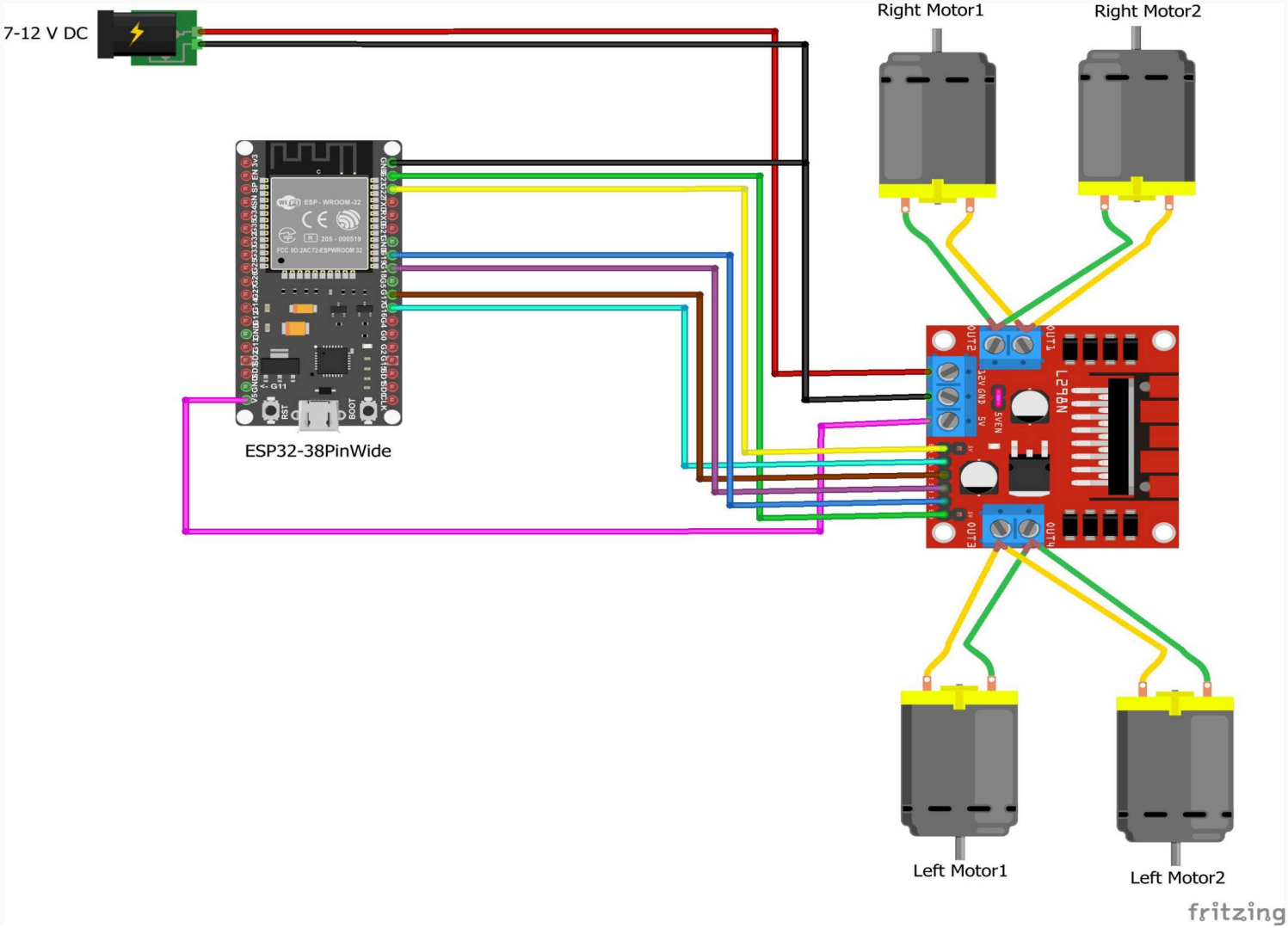
L298N Motor Driver

- The Enable A (ena) and Enable B (enb) pins are used for enabling and controlling the speed of the motor. If a jumper is present on this pin, the motor will be enabled and work at maximum speed, and if we remove the jumper we can connect a PWM input to this pin and in that way control the speed of the motor. If we connect this pin to a Ground the motor will be disabled.
- Next, the **Input 1 (in1)** and **Input 2 (in2)** pins are used for controlling the rotation direction of the motor A, and the **input 3 (in3)** and **input 4 (in4)** for the motor B. Using these pins we actually control the switches of the H-Bridge inside the L298N IC. If input 1 is LOW and input 2 is HIGH the motor will move forward, and vice versa, if input 1 is HIGH and input 2 is LOW the motor will move backward. In case both inputs are same, either LOW or HIGH the motor will stop. The same applies for the inputs 3 and 4 and the motor B.





Circuit Diagram



Code Explanation

```
#define CUSTOM_SETTINGS
#define INCLUDE_GAMEPAD_MODULE
#include <DabbleESP32.h>

#define in1 25
#define in2 26
#define in3 27
#define in4 14

#define ena 32
#define enb 33

int speed = 255;  /*max speed*/

int pwm = 1000; /* 1 KHz */
int pwm_resolution = 8;
int ena_pwm_channel = 4;
int enb_pwm_channel = 5;
```

```
void Move(int right_wheels_speed, int left_wheels_speed)
{
  if (right_wheels_speed > 0)
  {
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
  }
  else if (right_wheels_speed < 0)
  {
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
  }

  if (left_wheels_speed > 0)
  {
    digitalWrite(in3,LOW);
    digitalWrite(in4,HIGH);
  }
  else if (left_wheels_speed < 0)
  {
    digitalWrite(in3,HIGH);
    digitalWrite(in4,LOW);
  }

  else
  {
    digitalWrite(in1,LOW);
    digitalWrite(in2,LOW);
    digitalWrite(in3,LOW);
    digitalWrite(in4,LOW);
  }
```

```
void setUpPinModes()
{
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);

  pinMode(ena,OUTPUT);
  pinMode(enb,OUTPUT);

  //Set up PWM for speed
  ledcSetup(ena_pwm_channel, pwm, pwm_resolution);
  ledcSetup(enb_pwm_channel, pwm, pwm_resolution);
  ledcAttachPin(ena, ena_pwm_channel);
  ledcAttachPin(enb, enb_pwm_channel);
}

void setup()
{
  setUpPinModes();
  Dabble.begin("MyBluetoothCar");
}
```

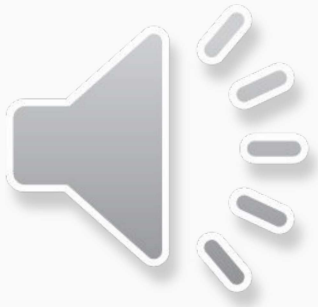
```
void loop()
{
  int right_wheels_speed = 0;
  int left_wheels_speed = 0;
  Dabble.processInput();
  if (GamePad.isUpPressed())
  {
    right_wheels_speed = speed;
    left_wheels_speed = speed;
  }

  if (GamePad.isDownPressed())
  {
    right_wheels_speed = -speed;
    left_wheels_speed = -speed;
  }

  if (GamePad.isLeftPressed())
  {
    right_wheels_speed = speed;
    left_wheels_speed = -speed;
  }

  if (GamePad.isRightPressed())
  {
    right_wheels_speed = -speed;
    left_wheels_speed = speed;
  }

  Move(right_wheels_speed, left_wheels_speed);
}
```



Flashing the Code on ESP32

Upload the code on ESP board

upload Completed, Now the code become in esp board and you can disconnect the board

```
Dabble.begin("MyBluetoothCar");
}

Output
Writing at 0x000f2995... (84 %)
Writing at 0x000f802d... (86 %)
Writing at 0x00100ea0... (88 %)
Writing at 0x00109142... (90 %)
Writing at 0x0010f6e7... (93 %)
Writing at 0x001154d8... (95 %)
Writing at 0x0011abdc... (97 %)
Writing at 0x00120690... (100 %)
Wrote 1120704 bytes (707866 compressed) at 0x00010000 in 10.1 seconds (effective 885.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```