SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

19ECT213 – IOT SYSTEM ARCHITECTURE

UNIT-3-ACTUATORS AND IOT NETWORKING DEVICES

II YEAR/ IV SEMESTER

ACTUATORS

- An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system.
- An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power.
- When the control signal is received, the actuator responds by converting the energy into mechanical motion.

ACTUATORS

- Following basic actuators are used for signaling and output purpose:
  - LED
  - RGB LED
  - Buzzer
  - Servo Motor
  - DC Motor
  - Relay

HYDRAULIC ACTUATORS

1. Hydraulic Actuators –

A hydraulic actuator uses hydraulic power to perform a mechanical operation. They are actuated by a cylinder or fluid motor. The mechanical motion is converted to rotary, linear, or oscillatory motion, according to the need of the IoT device. Ex- construction equipment uses hydraulic actuators because hydraulic actuators can generate a large amount of force.

Advantages :

- Hydraulic actuators can produce a large magnitude of force and high speed.
- Used in welding, clamping, etc.
- Used for lowering or raising the vehicles in car transport carriers.

Disadvantages :

- Hydraulic fluid leaks can cause efficiency loss and issues of cleaning.
- It is expensive.
- It requires noise reduction equipment, heat exchangers, and high maintenance systems.

PNEUMATIC ACTUATORS

2. Pneumatic Actuators –

A pneumatic actuator uses energy formed by vacuum or compressed air at high pressure to convert into either linear or rotary motion. Example- Used in robotics, use sensors that work like human fingers by using compressed air.

Advantages :

- They are a low-cost option and are used at extreme temperatures where using air is a safer option than chemicals.
- They need low maintenance, are durable, and have a long operational life.
- It is very quick in starting and stopping the motion.

Disadvantages :

- Loss of pressure can make it less efficient.
- The air compressor should be running continuously.
- Air can be polluted, and it needs maintenance.

ELECTRICAL ACTUATORS

3. Electrical Actuators –

An electric actuator uses electrical energy, is usually actuated by a motor that converts electrical energy into mechanical torque. An example of an electric actuator is a solenoid based electric bell.

Advantages :

- It has many applications in various industries as it can automate industrial valves.
- It produces less noise and is safe to use since there are no fluid leakages.
- It can be re-programmed and it provides the highest control precision positioning.

Disadvantages :

- It is expensive.
- It depends a lot on environmental conditions.

THERMAL & MAGNETIC ACTUATORS

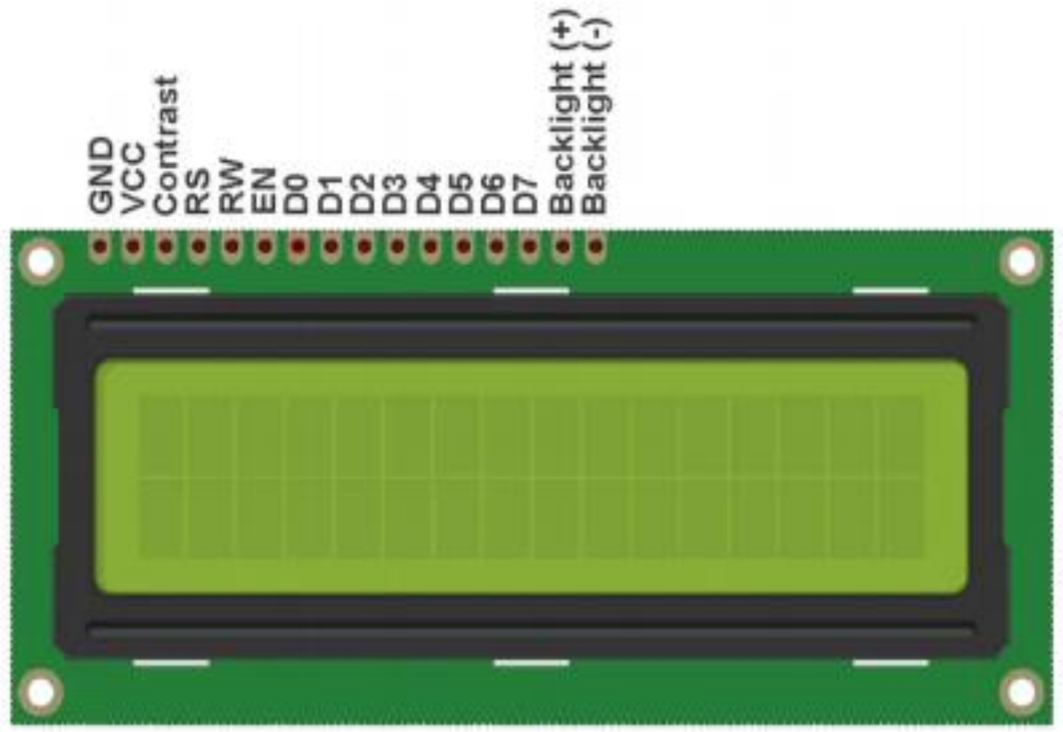
- **Thermal/Magnetic Actuators –**
  These are actuated by thermal or mechanical energy. Shape Memory Alloys (SMAs) or Magnetic Shape-Memory Alloys (MSMAs) are used by these actuators. An example of a thermal/magnetic actuator can be a piezo motor using SMA.
- **Mechanical Actuators –**
  A mechanical actuator executes movement by converting rotary motion into linear motion. It involves pulleys, chains, gears, rails, and other devices to operate. Example – A crankshaft.
- Soft Actuators
- Shape Memory Polymers
- Light Activated Polymers
- With the expanding world of IoT, sensors and actuators will find more usage in commercial and domestic applications along with the pre-existing use in industry.

THERMAL & MAGNETIC ACTUATORS

- **Thermal/Magnetic Actuators –**
  These are actuated by thermal or mechanical energy. Shape Memory Alloys (SMAs) or Magnetic Shape-Memory Alloys (MSMAs) are used by these actuators. An example of a thermal/magnetic actuator can be a piezo motor using SMA.
- **Mechanical Actuators –**
  A mechanical actuator executes movement by converting rotary motion into linear motion. It involves pulleys, chains, gears, rails, and other devices to operate. Example – A crankshaft.
- Soft Actuators
- Shape Memory Polymers
- Light Activated Polymers
- With the expanding world of IoT, sensors and actuators will find more usage in commercial and domestic applications along with the pre-existing use in industry.

LCD INTERFACING

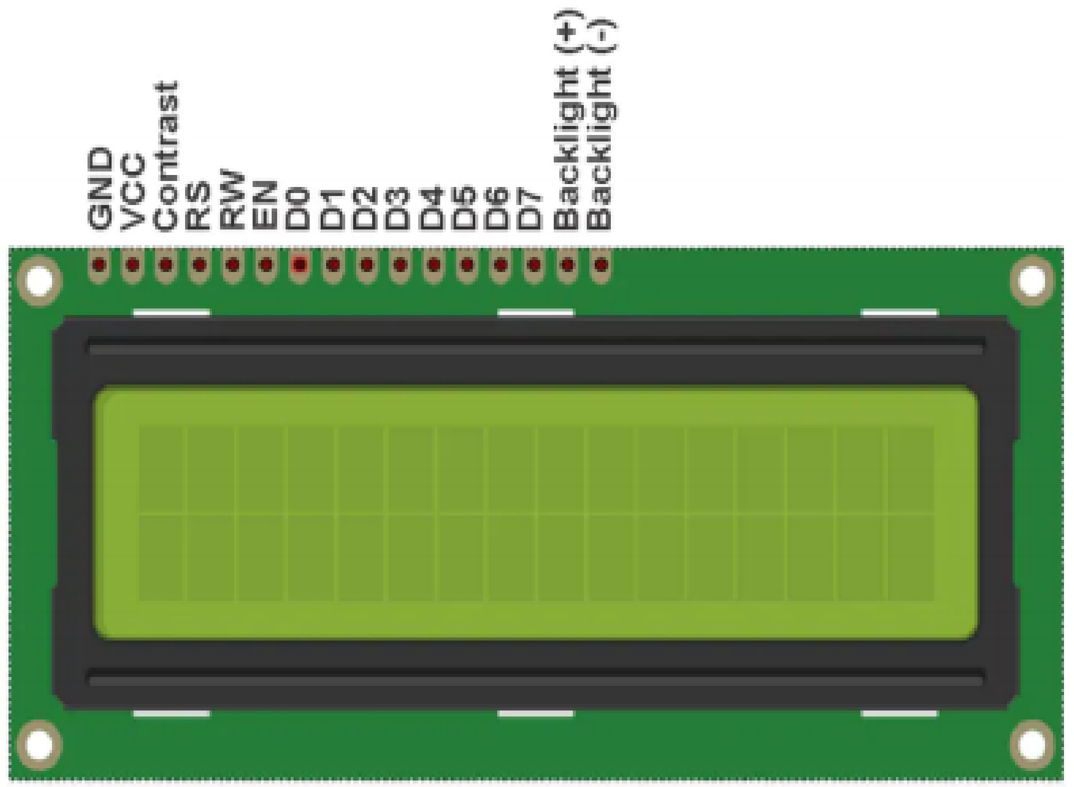
- LCD is a seven segment display that is used to display the output.
- It is available in dimensions **8×1, 8×2, 8×4, 16×1, 16×2 20×1, 20×2, 20×4, 24×1, 24×2, 24×4, 32×1, 32×2, 40×1, 40×2, 40×4 .**
- **Hitachi HD44780** 16X2 is the commonly used lcd for arduino.
- 16 represents **columns (0 - 15)** 2 represents **rows (0 - 1)**.



- pin 1 - **GND** - Ground
- pin 2 - **Vcc** - power supply (5v)
- pin 15 - Backlight (+) - 5V
- pin 16 - Backlight (-) - Gnd
- pin 3 - **Contrast** (adjustable with potentiometer)
- pin 4 - **Register Select**
- pin 5 - **Read or Write**
- pin 6 - **Enable**
- pin 7 to pin 14 (D0 - D7) - **Data pins**

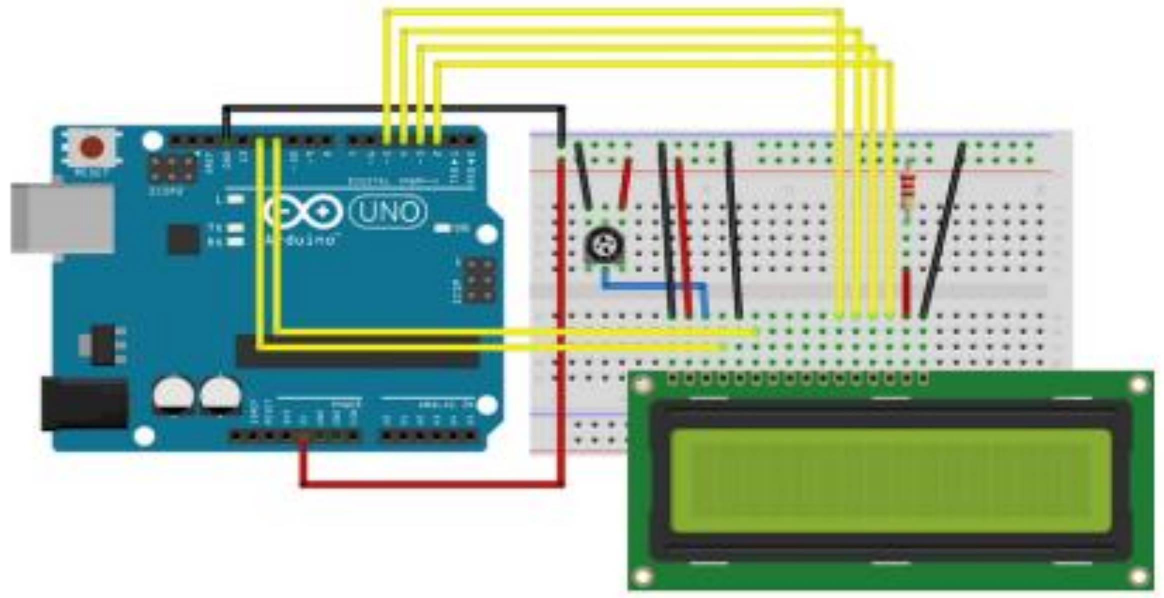
LCD INTERFACING

- **Contrast (pin 3)** - connected to potentiometer to adjust the text contrast . depending upon potentiometer output contrast is adjusted.
- **Register Select (pin 4)** - switch between two registers (data or instruction registers) . Data register holds the display data of the screen . Instruction register holds the next operation to execute.
- **Read / Write (pin 5)** - High - Reads from register . Low - write to register.
  It is permanently grounded to write.
- **Enable (pin 6)** - enables inputing data into the data pins.
- **Data pins (D0 - D7)** - It has two modes 4 bit or 8 bit mode.
  - In **4 bit mode** , last four significant bits (D4 - D7) are used.
  - In **8 bit mode** , all bits are used (D0 - D4).
  - 4 bit mode is commonly used.



LCD INTERFACING

Connection and coding



- pin 1 - **GND** - Ground
- pin 2 - **Vcc** - power supply (5v)
- pin 15 - Backlight (+) - 5V
- pin 16 - Backlight (-) - Gnd
- pin 3 - **Contrast** (10k potentiometer)
- pin 4 - **Register Select - D11**
- pin 5 - **Read or Write - Grounded**
- pin 6 - **Enable - D12**
- pin 11 to pin 14 (D4 - D7) - **D0 , D1 , D2 , D3**

```
//include the library
#include <LiquidCrystal.h>

//initialize the library by associating any needed LCD interface pin
//with the arduino pin number it is connected to
const int rs = 11, en =12, d4 = 3, d5 = 2, d6 = 1, d7 = 0;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
 //set up the LCD's number of columns and rows and begin
 lcd.begin(16, 2);
 lcd.print("hello world!"); // prints hello world
 lcd.clear(); // clears the display
 lcd.write("hello"); //similiar to print
 lcd.setCursor(1,1); //set cursor to 1st row , 1st column
 lcd.scrollLeft();
 lcd.scrollRight(); //scroll disply towards left or right
 lcd.display() ; // turn display on
 lcd.noDisplay(); // turn display off
 lcd.blink(); // sets up blinking cursor
 lcd.cursor(); // sets up a underscore cursor at next writing position
 delay(1000);
}

void loop() {
}
```

Interface L298N DC Motor Driver Module with Arduino

Controlling a DC Motor

In order to have a complete control over DC motor, we have to control its speed and rotation direction. This can be achieved by combining these two techniques.

- **PWM** – For controlling speed
- **H-Bridge** – For controlling rotation direction

PWM – For controlling speed

The speed of a DC motor can be controlled by varying its input voltage. A common technique for doing this is to use PWM (Pulse Width Modulation)

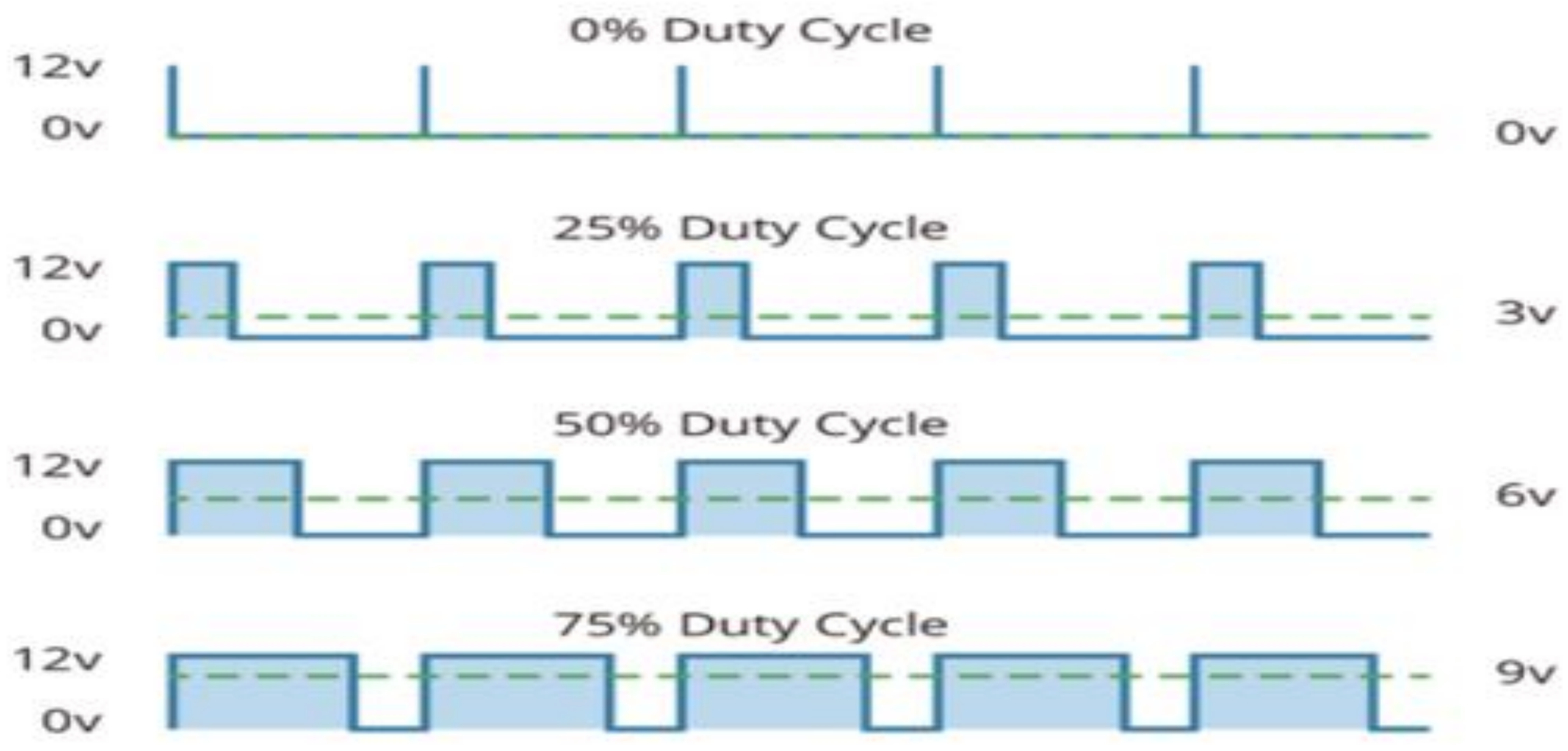
PWM is a technique where average value of the input voltage is adjusted by sending a series of ON-OFF pulses.

Interface L298N DC Motor Driver Module with Arduino

The average voltage is proportional to the width of the pulses known as **Duty Cycle**.

The higher the duty cycle, the greater the average voltage being applied to the dc motor(High Speed) and the lower the duty cycle, the less the average voltage being applied to the dc motor(Low Speed).

Below image illustrates PWM technique with various duty cycles andaverage voltages.

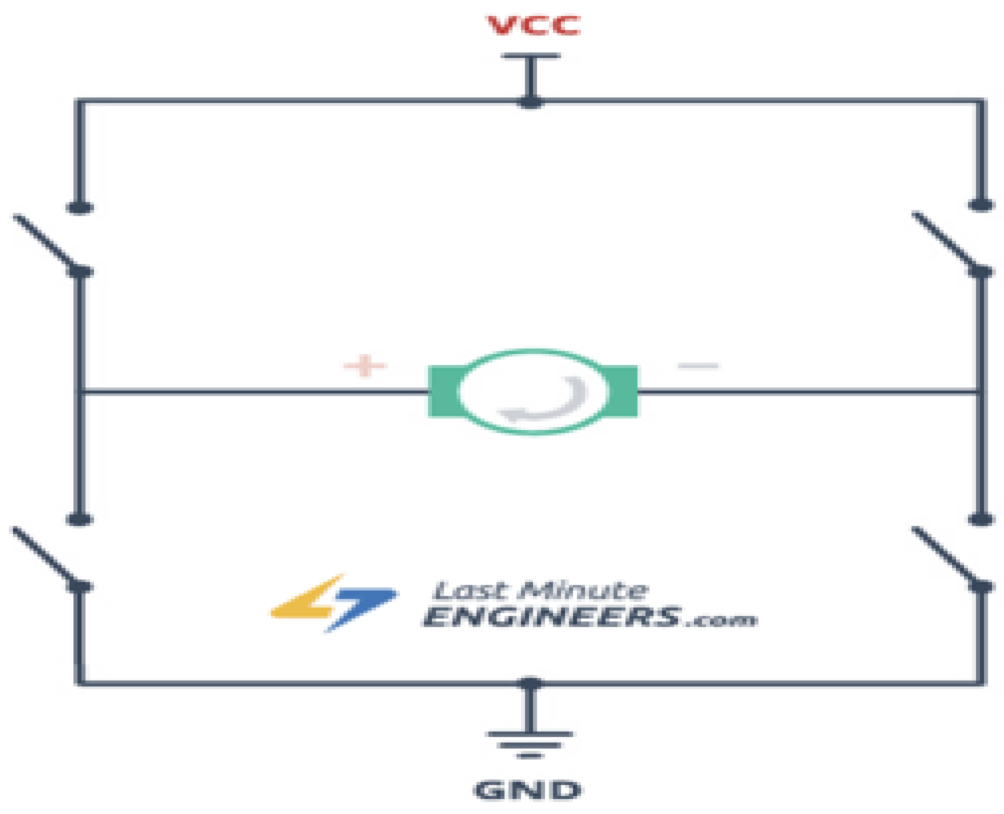


Interface L298N DC Motor Driver Module with Arduino

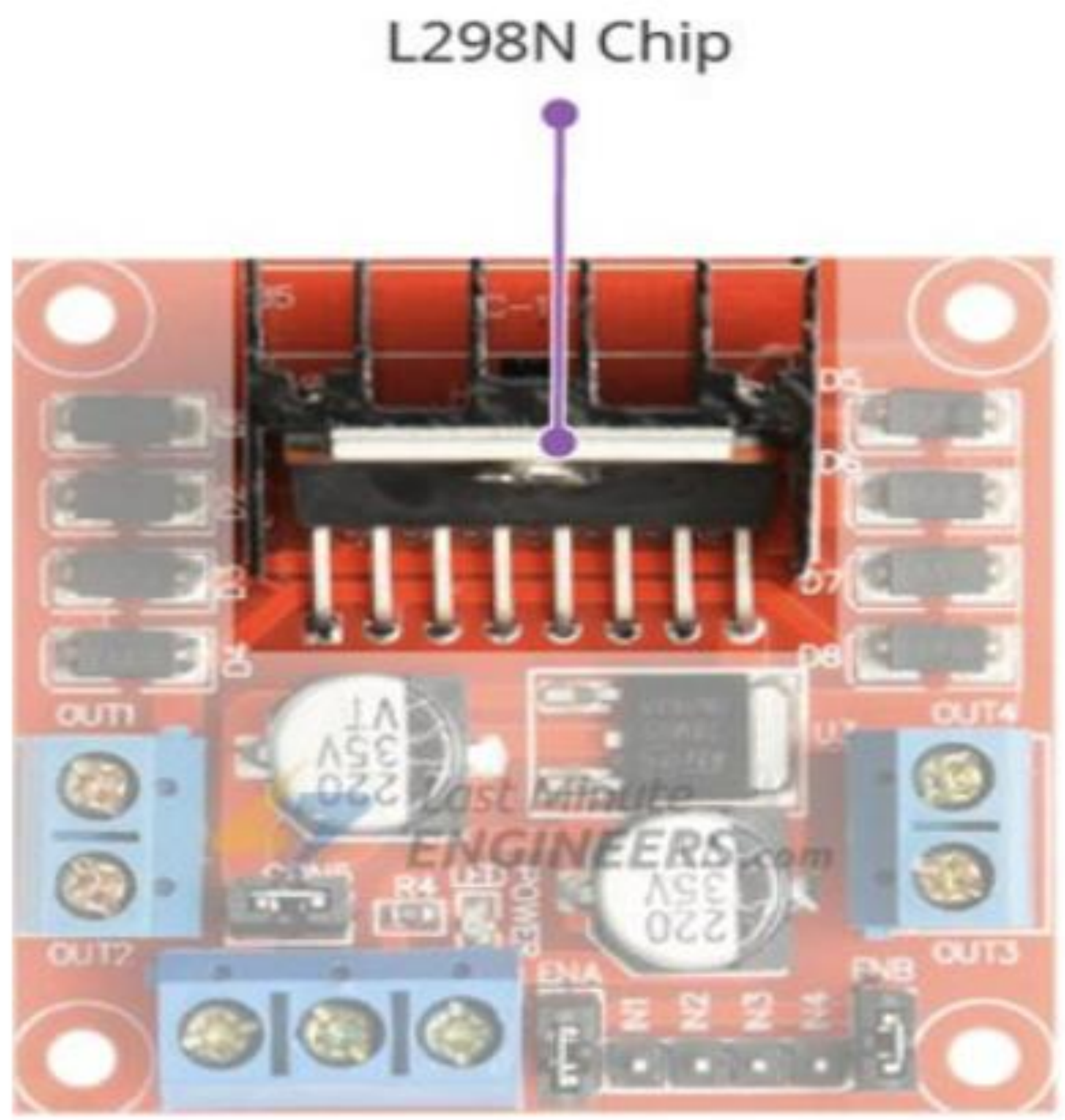
H-Bridge – For controlling rotation direction

The DC motor's spinning direction can be controlled by changing polarity of its input voltage. A common technique for doing this is to use an H-Bridge.

An H-Bridge circuit contains four switches with the motor at the center forming an H-like arrangement.

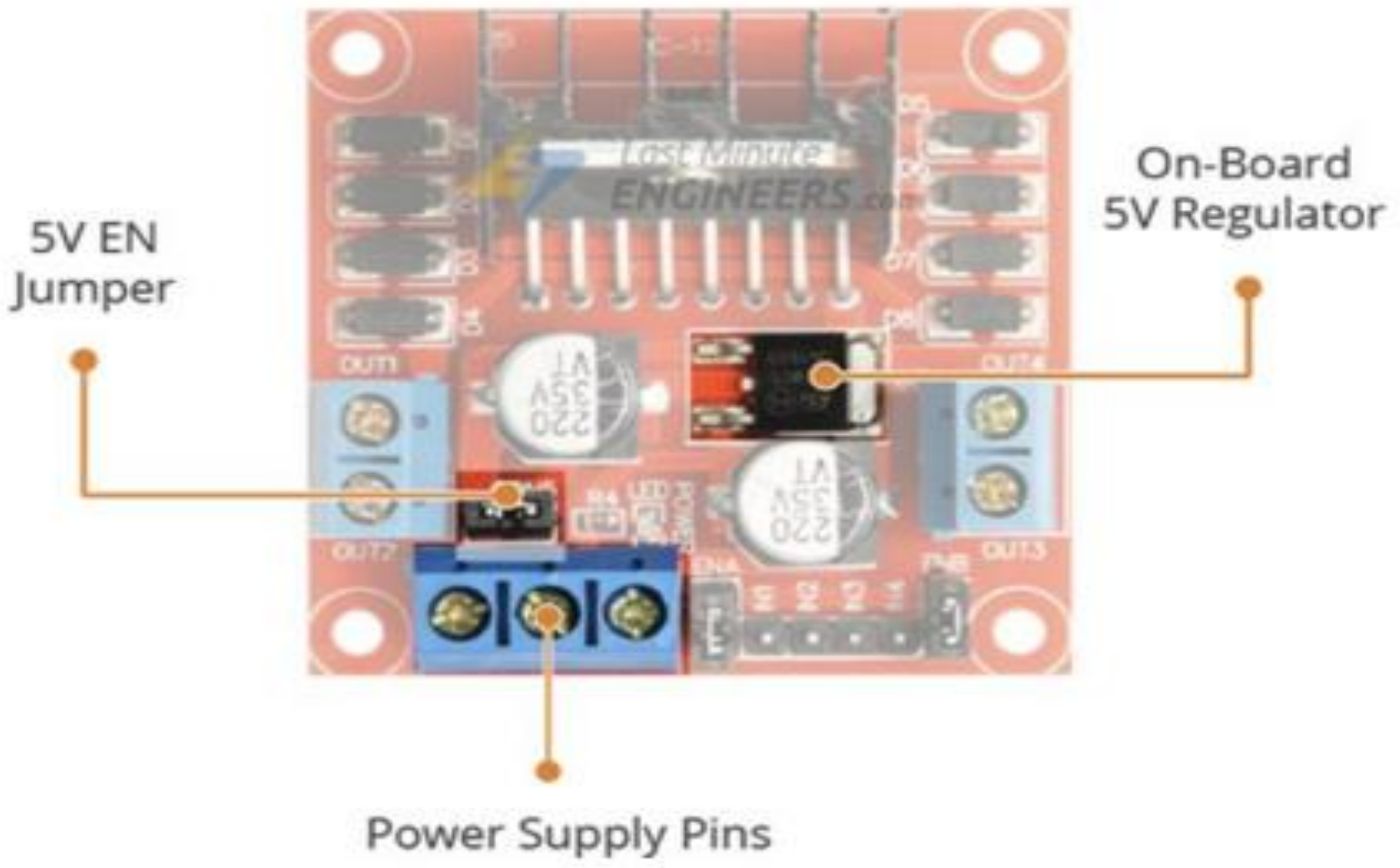


L298N Motor Driver IC



Interface L298N DC Motor Driver Module with Arduino

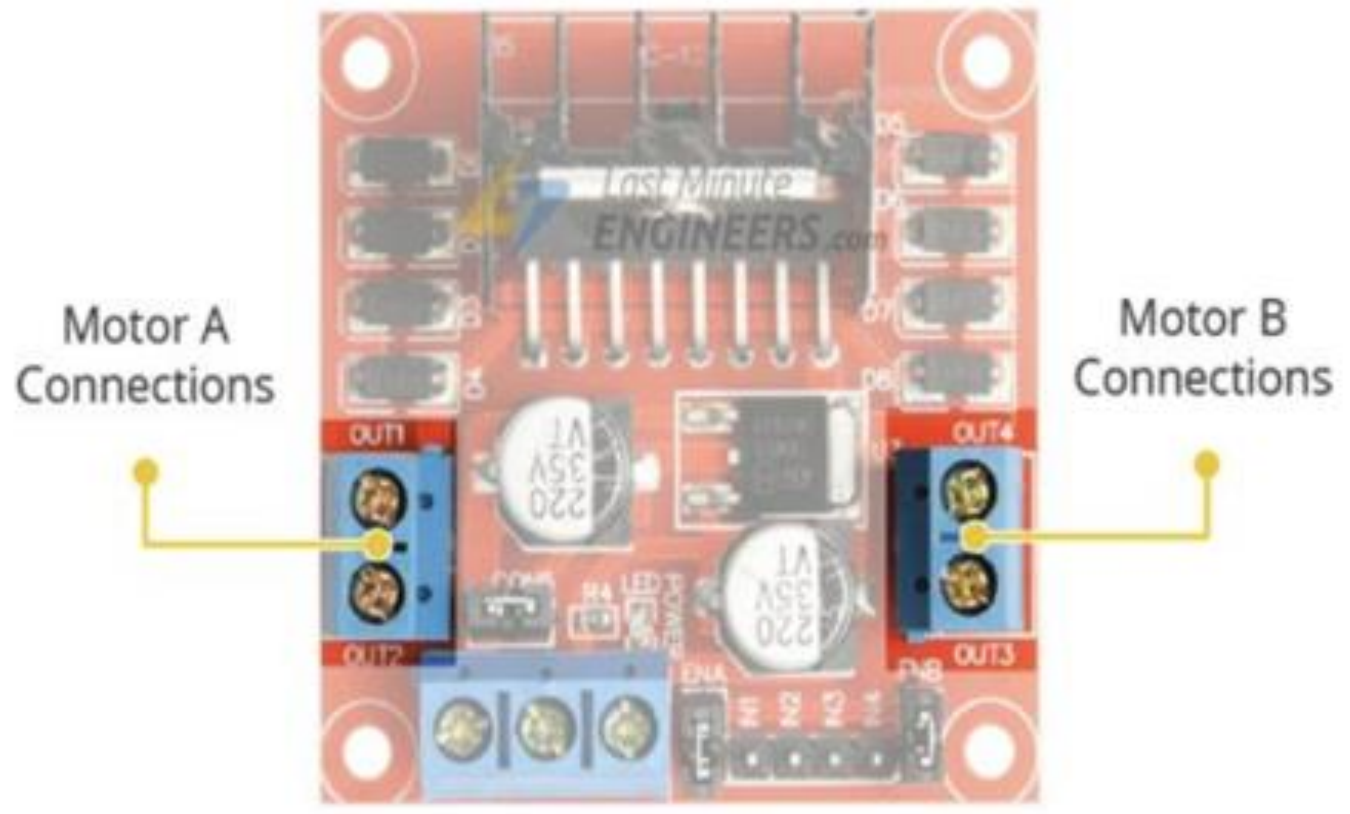
Power Supply



The L298N motor driver module is powered through 3-pin 3.5mm-pitch screw terminals. It consists of pins for motor power supply(Vs), ground and 5V logic power supply(Vss).

Interface L298N DC Motor Driver Module with Arduino

Output Pins

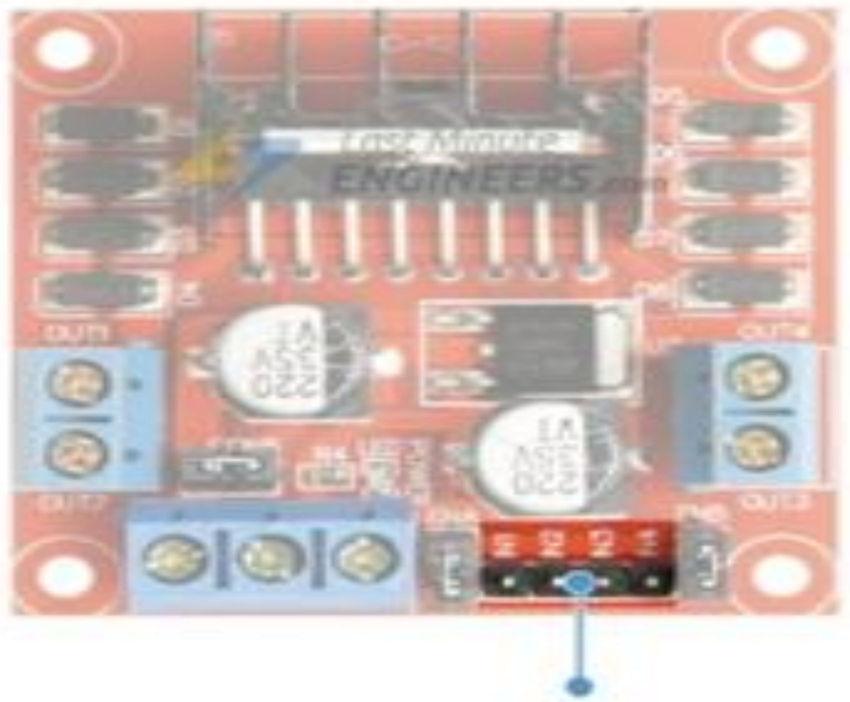


Interface L298N DC Motor Driver Module with Arduino

Control Pins

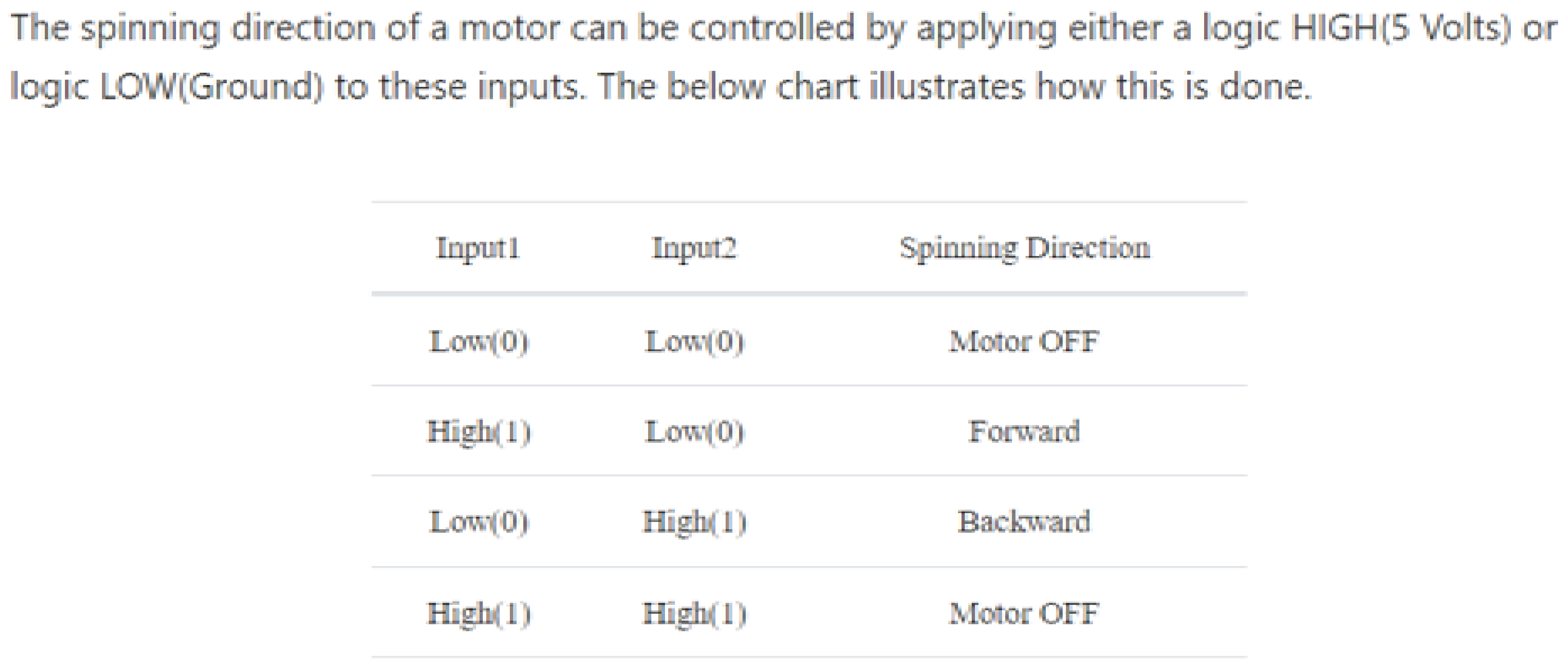
For each of the L298N's channels, there are two types of control pins which allow us to control speed and spinning direction of the DC motors at the same time viz. Direction control pins & Speed control pins.

Direction Control Pins



Direction Control Pins

Interface L298N DC Motor Driver Module with Arduino

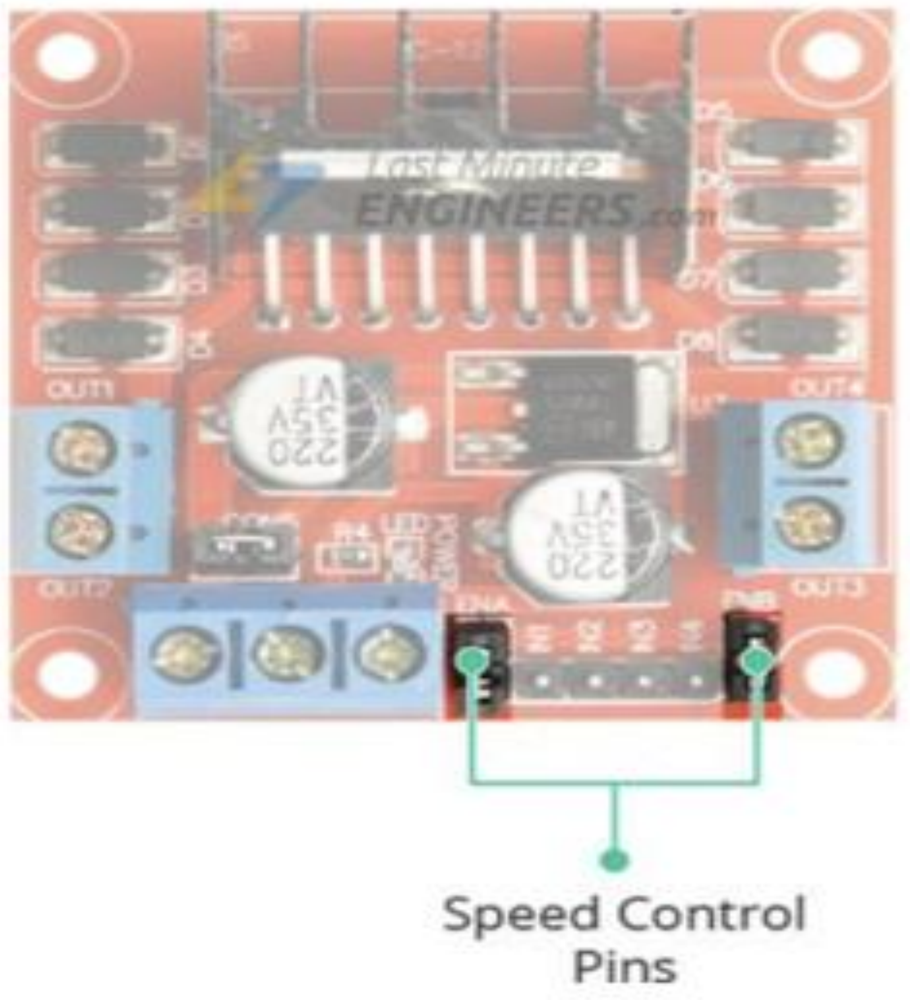


The spinning direction of a motor can be controlled by applying either a logic HIGH(5 Volts) or logic LOW(Ground) to these inputs. The below chart illustrates how this is done.

Input1	Input2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

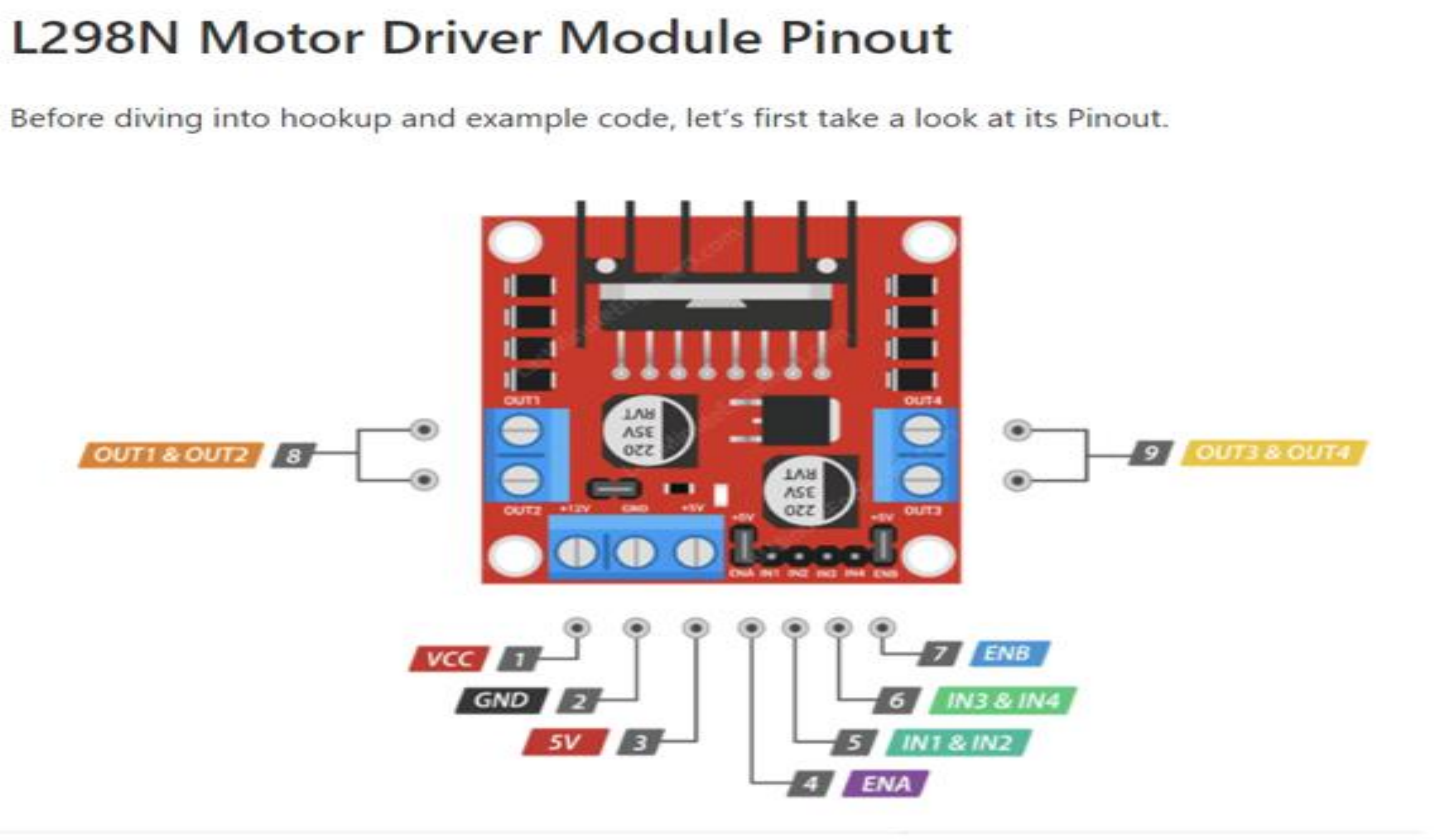
Interface L298N DC Motor Driver Module with Arduino

Speed Control Pins



The speed control pins viz. **ENA** and **ENB** are used to turn the motors ON, OFF and control its speed.

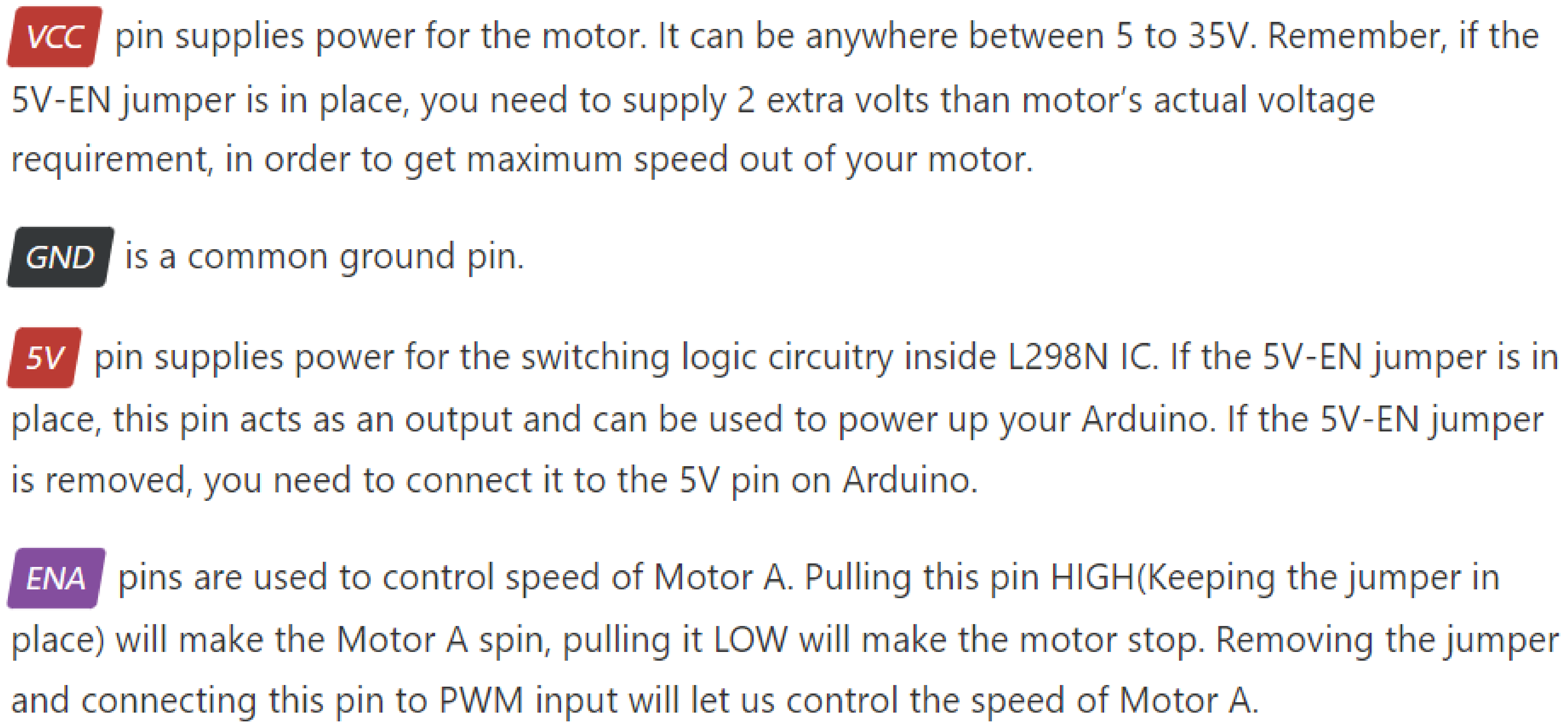
Interface L298N DC Motor Driver Module with Arduino



L298N Motor Driver Module Pinout

Before diving into hookup and example code, let's first take a look at its Pinout.

Interface L298N DC Motor Driver Module with Arduino



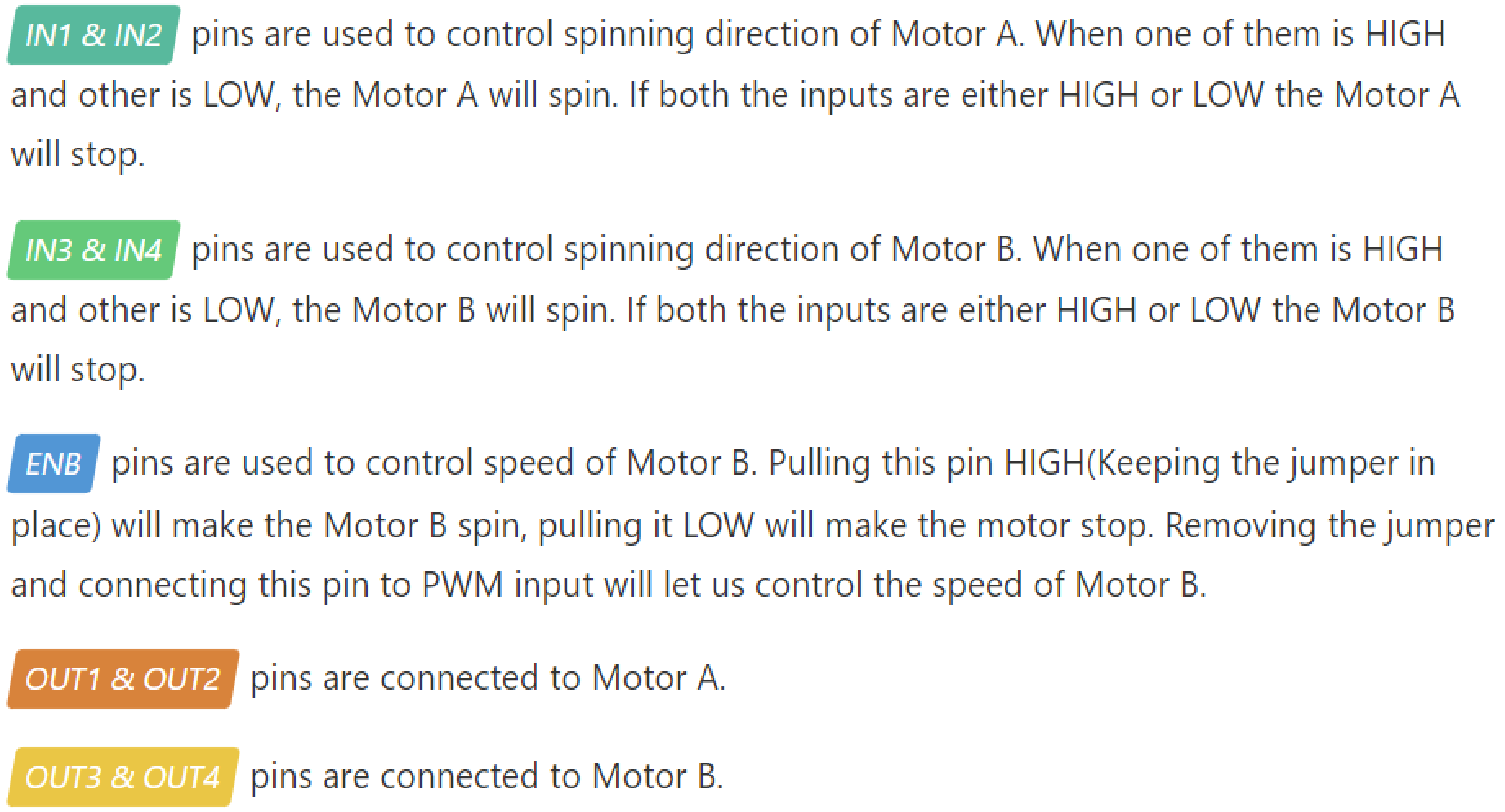
VCC pin supplies power for the motor. It can be anywhere between 5 to 35V. Remember, if the 5V-EN jumper is in place, you need to supply 2 extra volts than motor's actual voltage requirement, in order to get maximum speed out of your motor.

GND is a common ground pin.

5V pin supplies power for the switching logic circuitry inside L298N IC. If the 5V-EN jumper is in place, this pin acts as an output and can be used to power up your Arduino. If the 5V-EN jumper is removed, you need to connect it to the 5V pin on Arduino.

ENA pins are used to control speed of Motor A. Pulling this pin HIGH(Keeping the jumper in place) will make the Motor A spin, pulling it LOW will make the motor stop. Removing the jumper and connecting this pin to PWM input will let us control the speed of Motor A.

Interface L298N DC Motor Driver Module with Arduino



IN1 & IN2 pins are used to control spinning direction of Motor A. When one of them is HIGH and other is LOW, the Motor A will spin. If both the inputs are either HIGH or LOW the Motor A will stop.

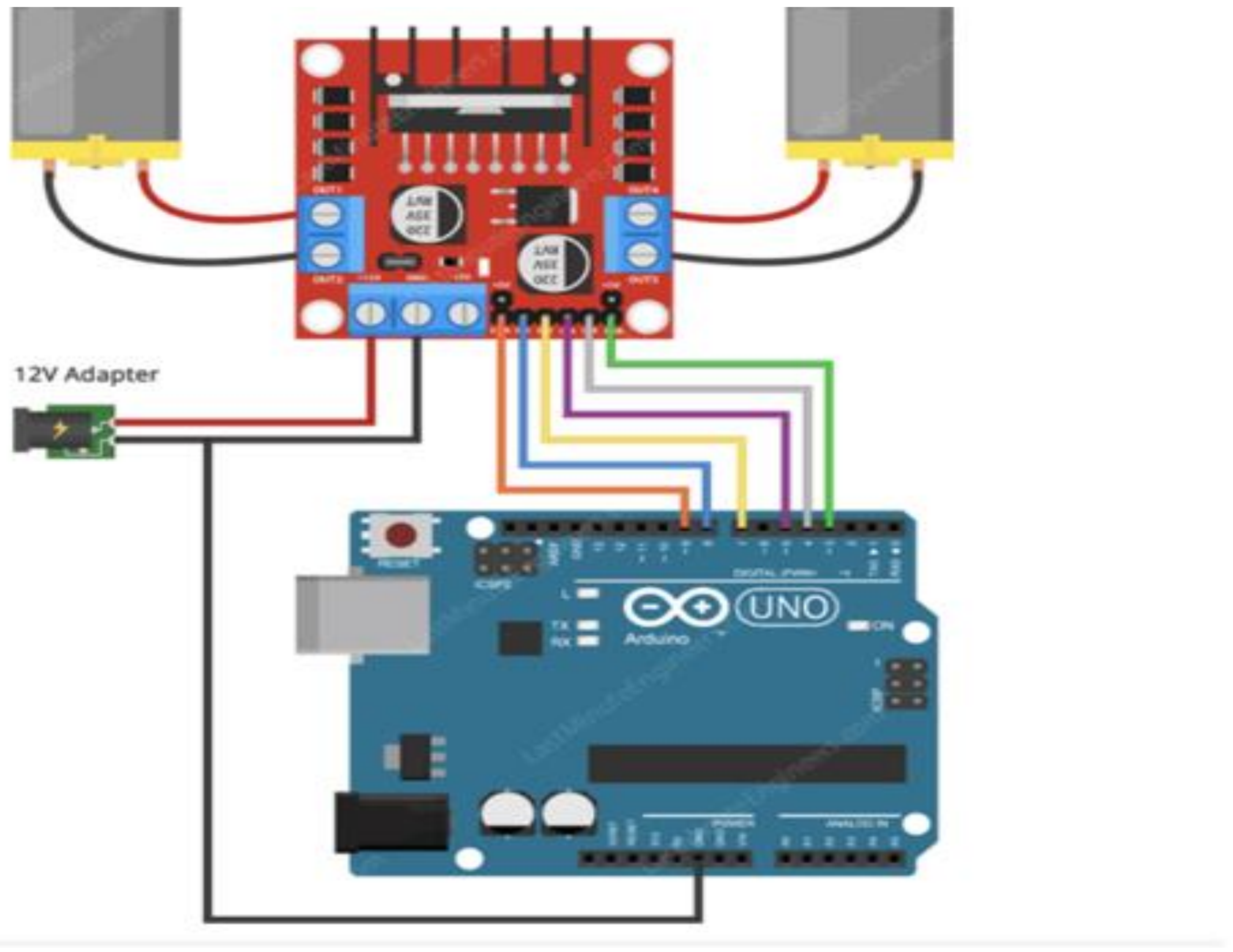
IN3 & IN4 pins are used to control spinning direction of Motor B. When one of them is HIGH and other is LOW, the Motor B will spin. If both the inputs are either HIGH or LOW the Motor B will stop.

ENB pins are used to control speed of Motor B. Pulling this pin HIGH(Keeping the jumper in place) will make the Motor B spin, pulling it LOW will make the motor stop. Removing the jumper and connecting this pin to PWM input will let us control the speed of Motor B.

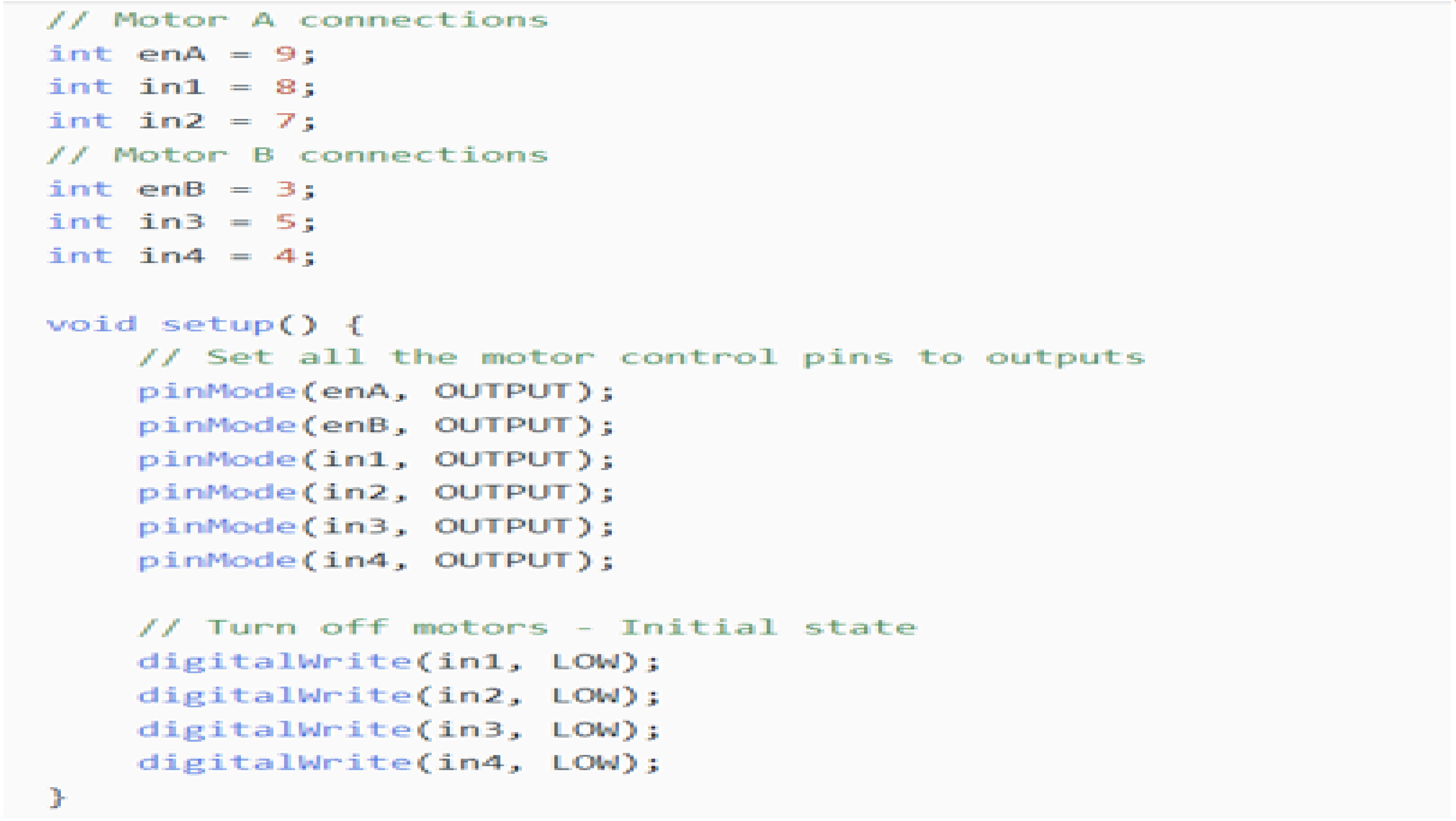
OUT1 & OUT2 pins are connected to Motor A.

OUT3 & OUT4 pins are connected to Motor B.

Interface L298N DC Motor Driver Module with Arduino



Interface L298N DC Motor Driver Module with Arduino

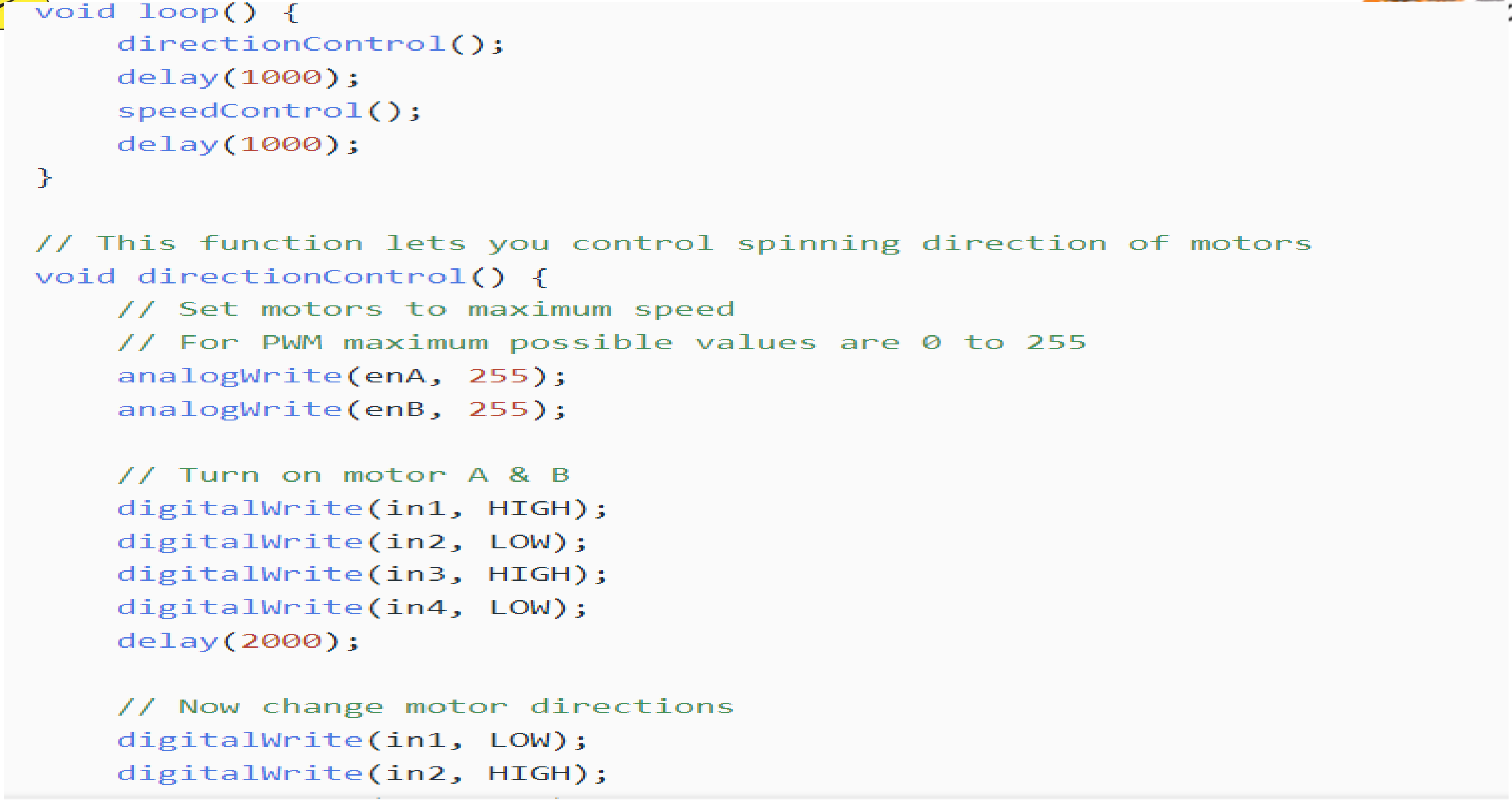


```
// Motor A connections
int enA = 9;
int in1 = 8;
int in2 = 7;
// Motor B connections
int enB = 3;
int in3 = 5;
int in4 = 4;

void setup() {
    // Set all the motor control pins to outputs
    pinMode(enA, OUTPUT);
    pinMode(enB, OUTPUT);
    pinMode(in1, OUTPUT);
    pinMode(in2, OUTPUT);
    pinMode(in3, OUTPUT);
    pinMode(in4, OUTPUT);

    // Turn off motors - Initial state
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
}
```

Interface L298N DC Motor Driver Module with Arduino



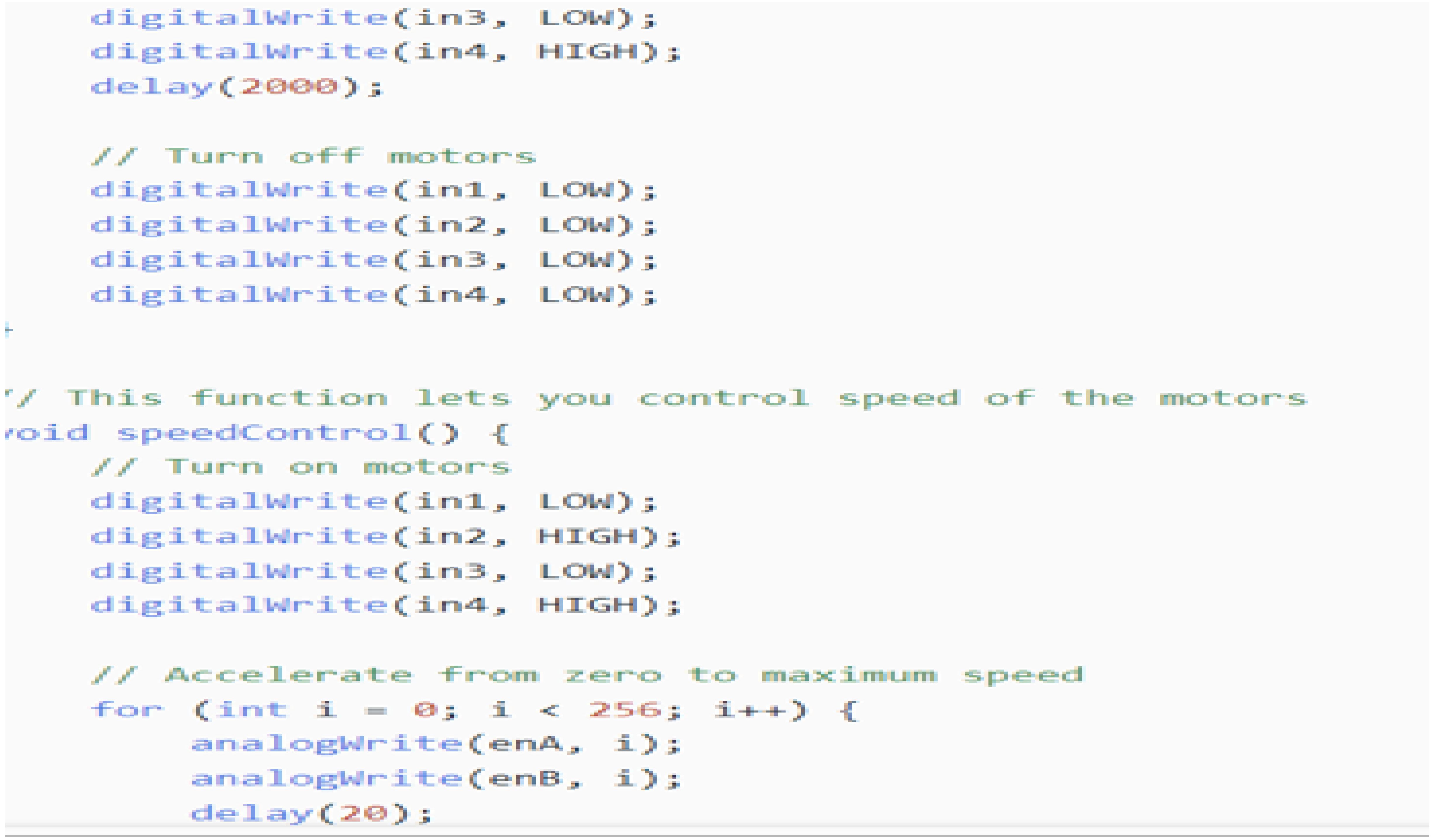
```
void loop() {
	directionControl();
	delay(1000);
	speedControl();
	delay(1000);
}

// This function lets you control spinning direction of motors
void directionControl() {
	// Set motors to maximum speed
	// For PWM maximum possible values are 0 to 255
	analogWrite(enA, 255);
	analogWrite(enB, 255);

	// Turn on motor A & B
	digitalWrite(in1, HIGH);
	digitalWrite(in2, LOW);
	digitalWrite(in3, HIGH);
	digitalWrite(in4, LOW);
	delay(2000);

	// Now change motor directions
	digitalWrite(in1, LOW);
	digitalWrite(in2, HIGH);
```

Interface L298N DC Motor Driver Module with Arduino



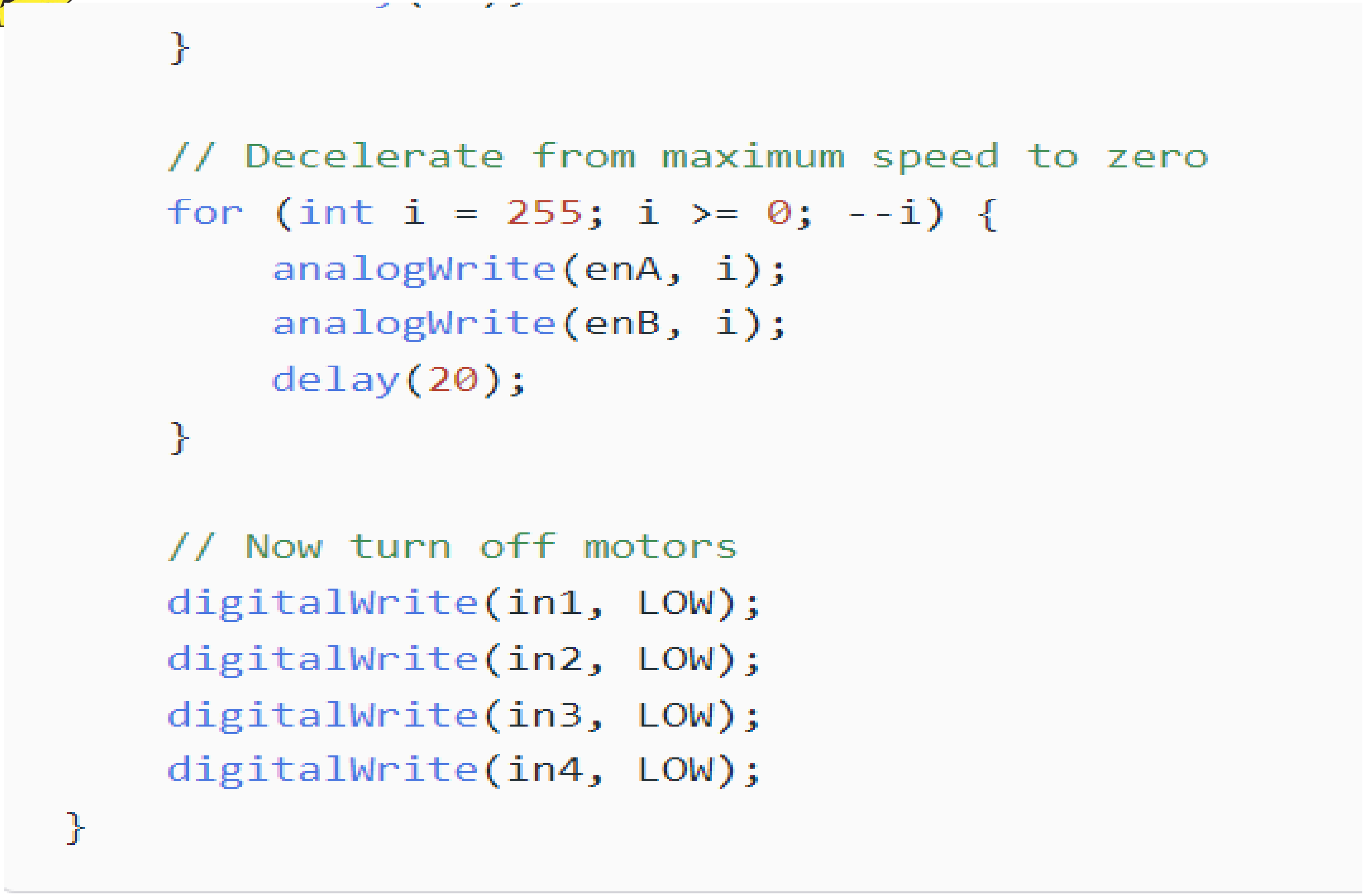
```
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    delay(2000);

    // Turn off motors
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
}

// This function lets you control speed of the motors
void speedControl() {
    // Turn on motors
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);

    // Accelerate from zero to maximum speed
    for (int i = 0; i < 256; i++) {
        analogWrite(enA, i);
        analogWrite(enB, i);
        delay(20);
```

Interface L298N DC Motor Driver Module with Arduino



```
	}

	// Decelerate from maximum speed to zero
	for (int i = 255; i >= 0; --i) {
		analogWrite(enA, i);
		analogWrite(enB, i);
		delay(20);
	}

	// Now turn off motors
	digitalWrite(in1, LOW);
	digitalWrite(in2, LOW);
	digitalWrite(in3, LOW);
	digitalWrite(in4, LOW);
}
```

HC-05 Bluetooth Module

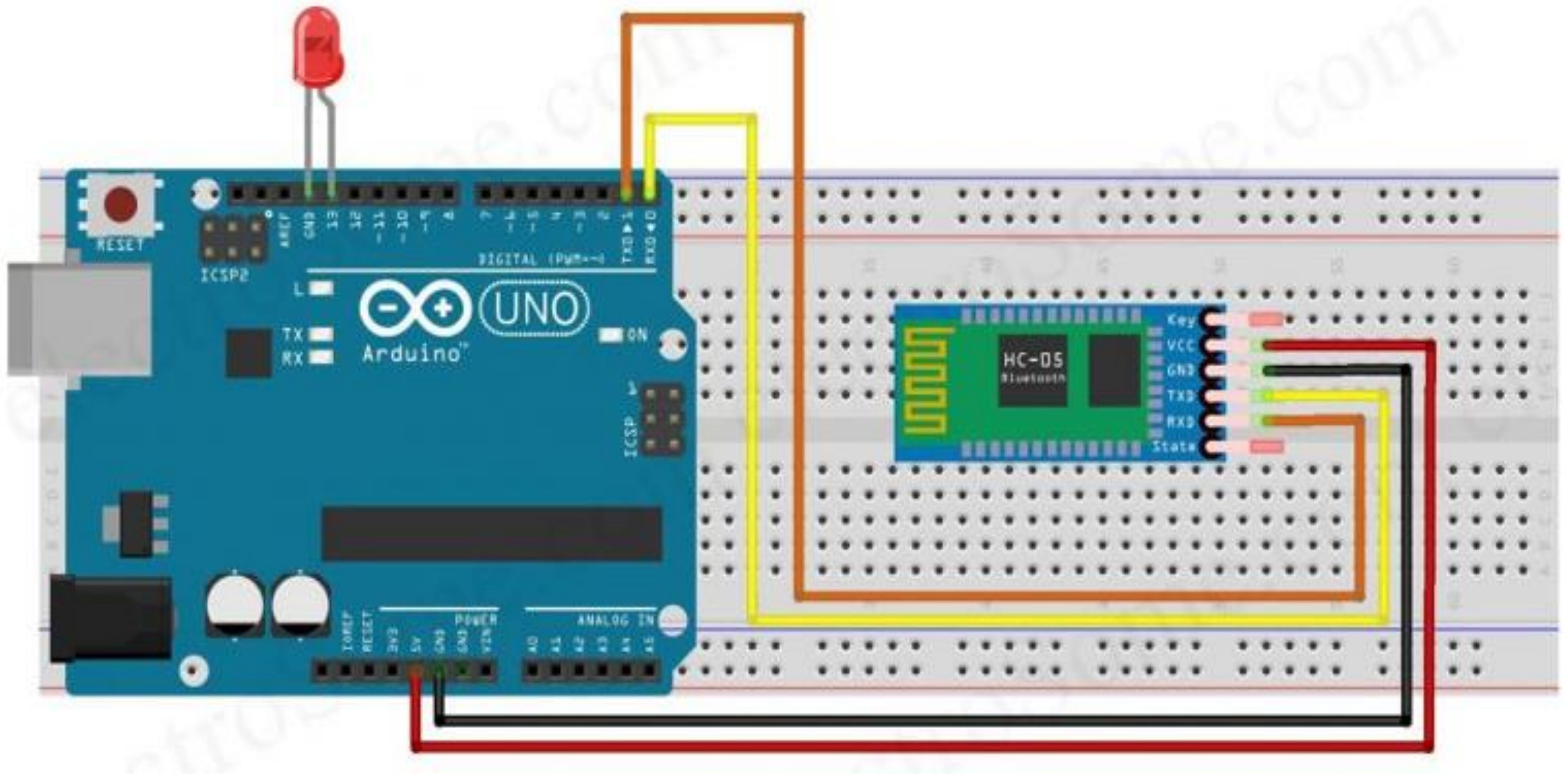
HC-05 Bluetooth Module

- Operating Voltage : 4 V to 6V (have internal 3.3V regulator).
- Operating Current : 30mA
- Integrated antenna and an edge connector.
- Range about 10 meters.
- Configurable in both master and slave modes.
- Pins : STATE, RXD, TXD, GND, VCC, KEY/ENABLE

Pin Out

- **STATE :** State pin indicates whether the module is connected or paired with a device. When the module is not connected, this pin will be in LOW state and the on-board LED will be flashing fast. But when the module is paired or connected to a device, the state pin will be in HIGH state and the on-board LED will be flashing with a delay.
- **RXD :** This is UART RX pin. This pin is used to send AT command when the module is in command mode. And it is used to send data to the connected device when the module is in data mode.
- **TXD :** This is UART TX pin. This pin is used push out responses to AT command when the module is in command mode. And it is used push out data send by the connected device when the module is in data mode.
- **GND :** Power supply -ive.
- **VCC :** Power supply +ive.
- **EN/KEY :** This input is used to switch between command and data mode. If this pin is set HIGH, the module will be in command mode. Similarly if this pin is set LOW, the module will be in data mode.

Circuit Diagram



Description

- RXD pin of HC-05 Bluetooth – TXD pin of Arduino Uno
- TXD pin of HC-05 Bluetooth – RXD pin of Arduino Uno
- GND pin of HC-05 Bluetooth – GND pin of Arduino Uno
- VCC pin of HC-05 Bluetooth – 5V output pin of Arduino Uno
- Positive pin of LED – Pin 13 of Arduino Uno
- Negative pin of LED – GND pin of Arduino Uno

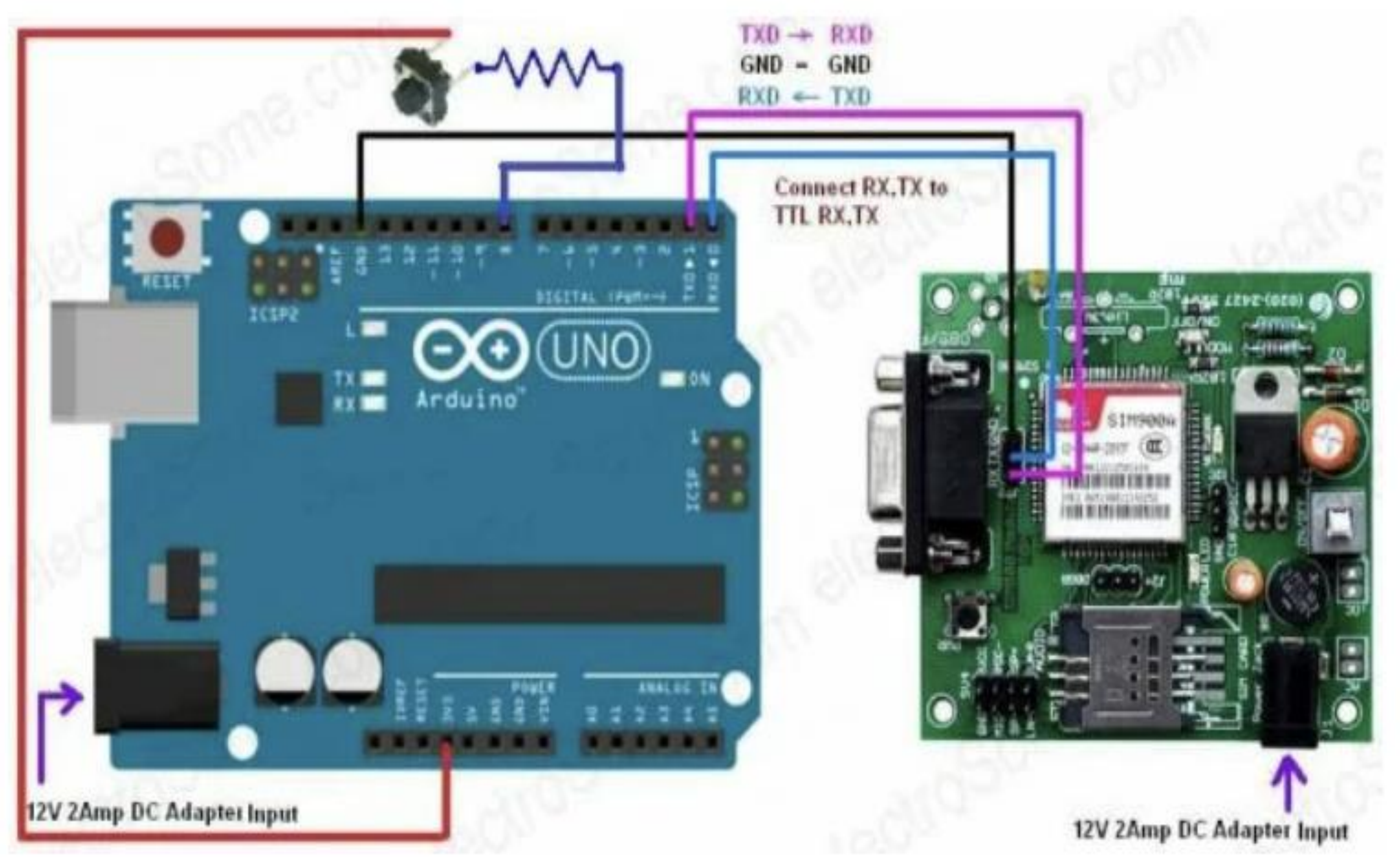
```
char data = 0;            //Variable for storing received data

void setup()
{
    Serial.begin(9600);   //Sets the data rate in bits per second (baud) for serial
data transmission
    pinMode(13, OUTPUT); //Sets digital pin 13 as output pin
}

void loop()
{
    if(Serial.available() > 0)      // Send data only when you receive data:
    {
        data = Serial.read();       //Read the incoming data and store it into
variable data
        Serial.print(data);         //Print Value inside data in Serial monitor
        Serial.print("\n");         //New line
        if(data == '1')             //Checks whether value of data is equal to 1
            digitalWrite(13, HIGH); //If value is 1 then LED turns ON
        else if(data == '0')        //Checks whether value of data is equal to 0
            digitalWrite(13, LOW);  //If value is 0 then LED turns OFF
    }
}
```

Interfacing GSM module to Arduino

Circuit Diagram



TXD → RXD
GND - GND
RXD ← TXD
Connect RX,TX to TTL RX,TX
12V 2Amp DC Adapter Input
12V 2Amp DC Adapter Input

Interfacing GSM Modem with Arduino – Circuit Diagram

Connections

- Connect **TX pin of GSM Module to RX pin** of Arduino Uno.
- Connect **RX pin of GSM Module to TX pin** of Arduino Uno.
- Connect GND pin of GSM Module to GND pin of Arduino Uno.

Steps to boot GSM module

1. Power ON the GSM module by providing 5V and GND.
2. Insert the SIM card to GSM module and lock it.
3. Initially blinking rate of network LED will be high. After sometime observe the blinking rate of 'network LED' (GSM module will take some time to establish connection with mobile network)
4. Once the connection is established successfully, the network LED will blink continuously for every 3 seconds.
5. Even we can check the connection establishment of GSM module with mobile by making a call to the number of the SIM. If we hear a ring back, the GSM module has successfully established network connection.

```
void setup()
{
    Serial.begin(9600);   //Initialise serial to communicate with GSM
Modem
}

void loop()
{
     delay(10000); //Give enough time for GSM to register on Network
     SendSMS();    //Send one SMS
     while(1);     //Wait forever
}

void SendSMS()
{
  Serial.println("AT+CMGF=1");    //To send SMS in Text Mode
  delay(1000);
  Serial.println("AT+CMGS=\"+9198xxxxxxxx\"\r"); //Change to destination
phone number
  delay(1000);
  Serial.println("Hello from GSM Modem!");//the content of the message
  delay(200);
  Serial.println((char)26); //the stopping character Ctrl+Z
  delay(1000);
}
```

ESP8266 WIFI MODULE

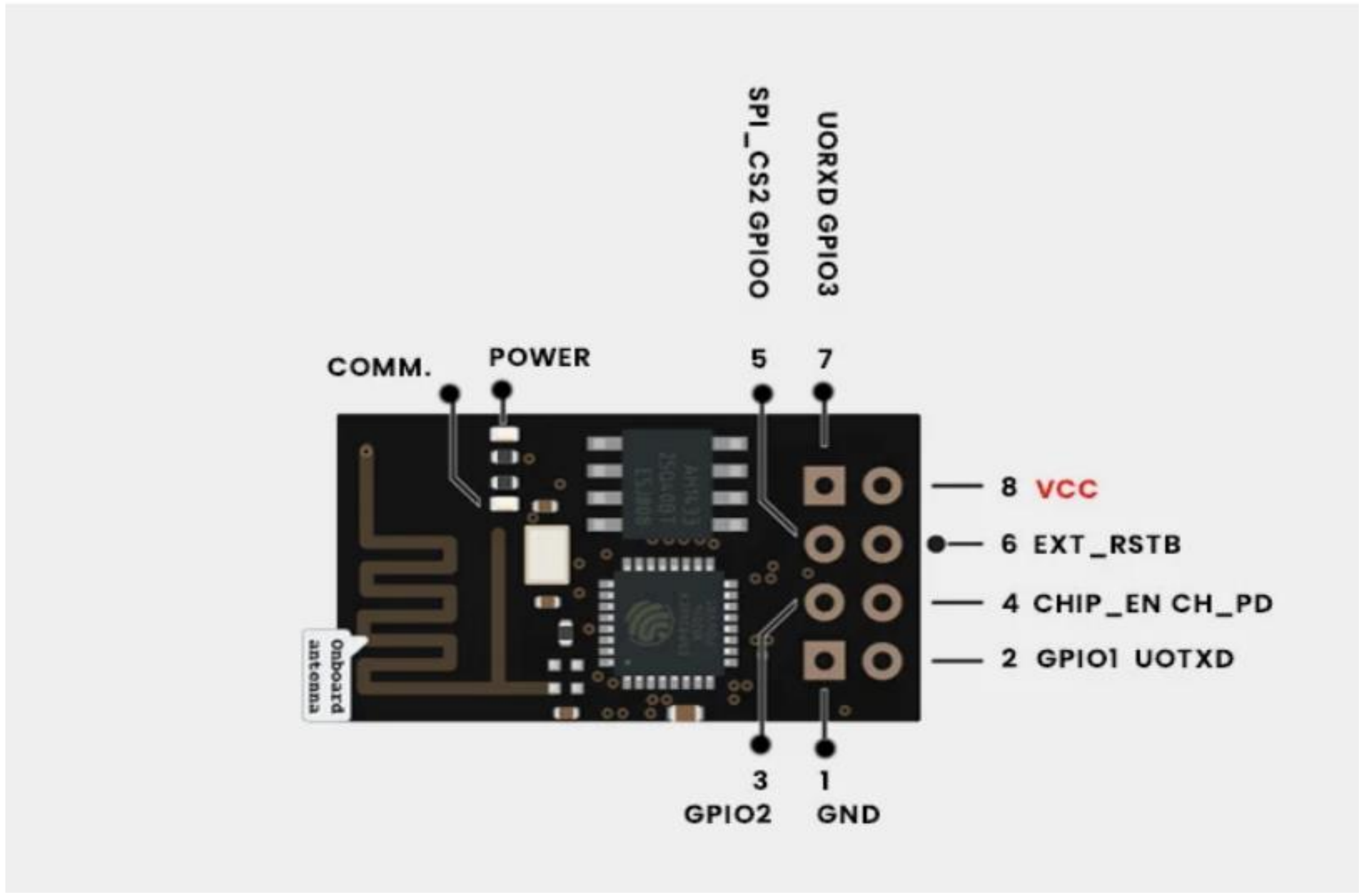
What Is ESP8266?

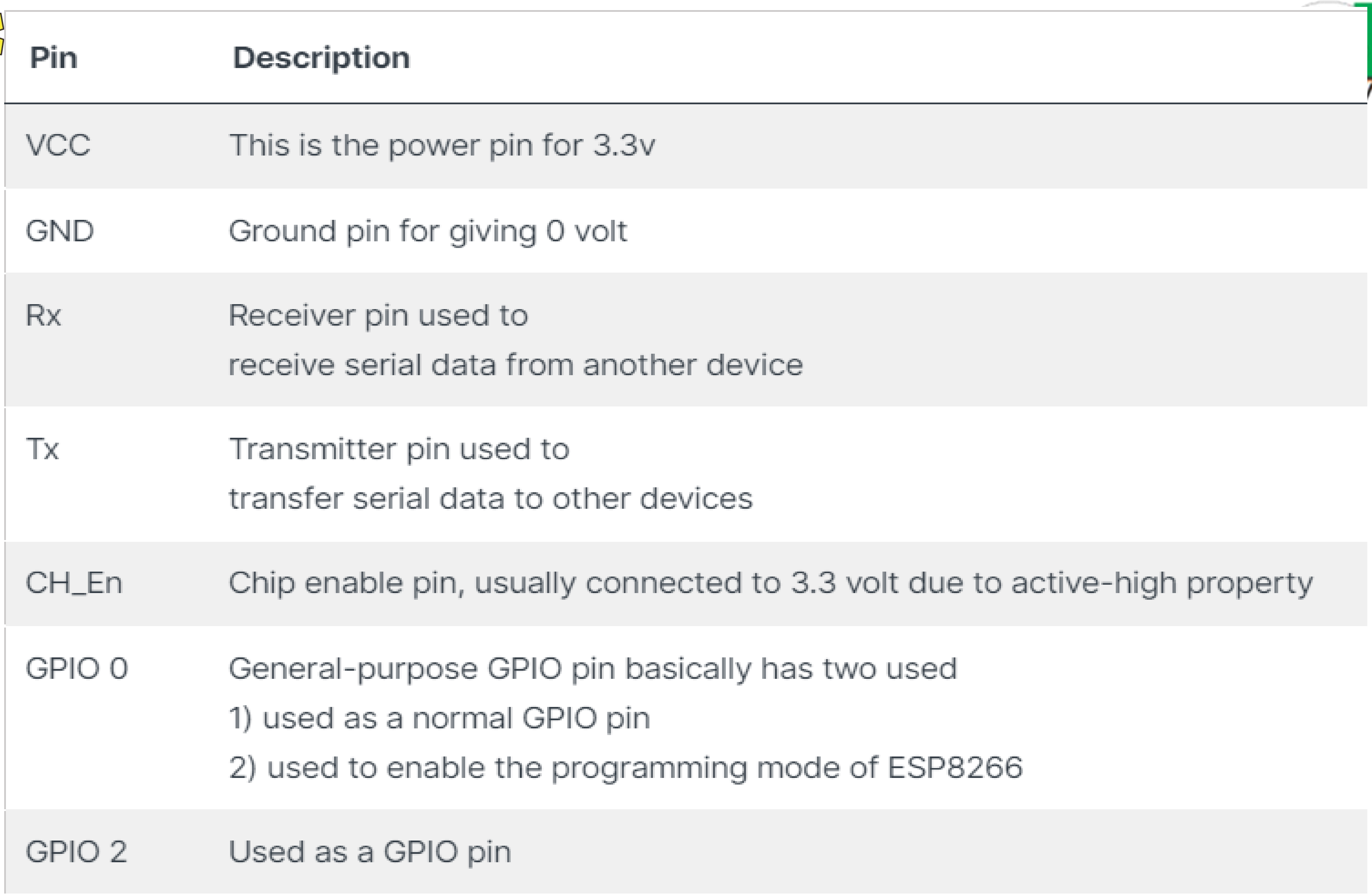
ESP8266 is a Low-cost wifi module that can provide internet connectivity to your small-scale embedded system/projects. This module comes with a single-chip CPU, GPIO pins, analog pins, I2C and SPI pins. The processor used in this module is the L106 32 bit RISC microprocessor, which runs on 80 MHz

Features of ESP8266:

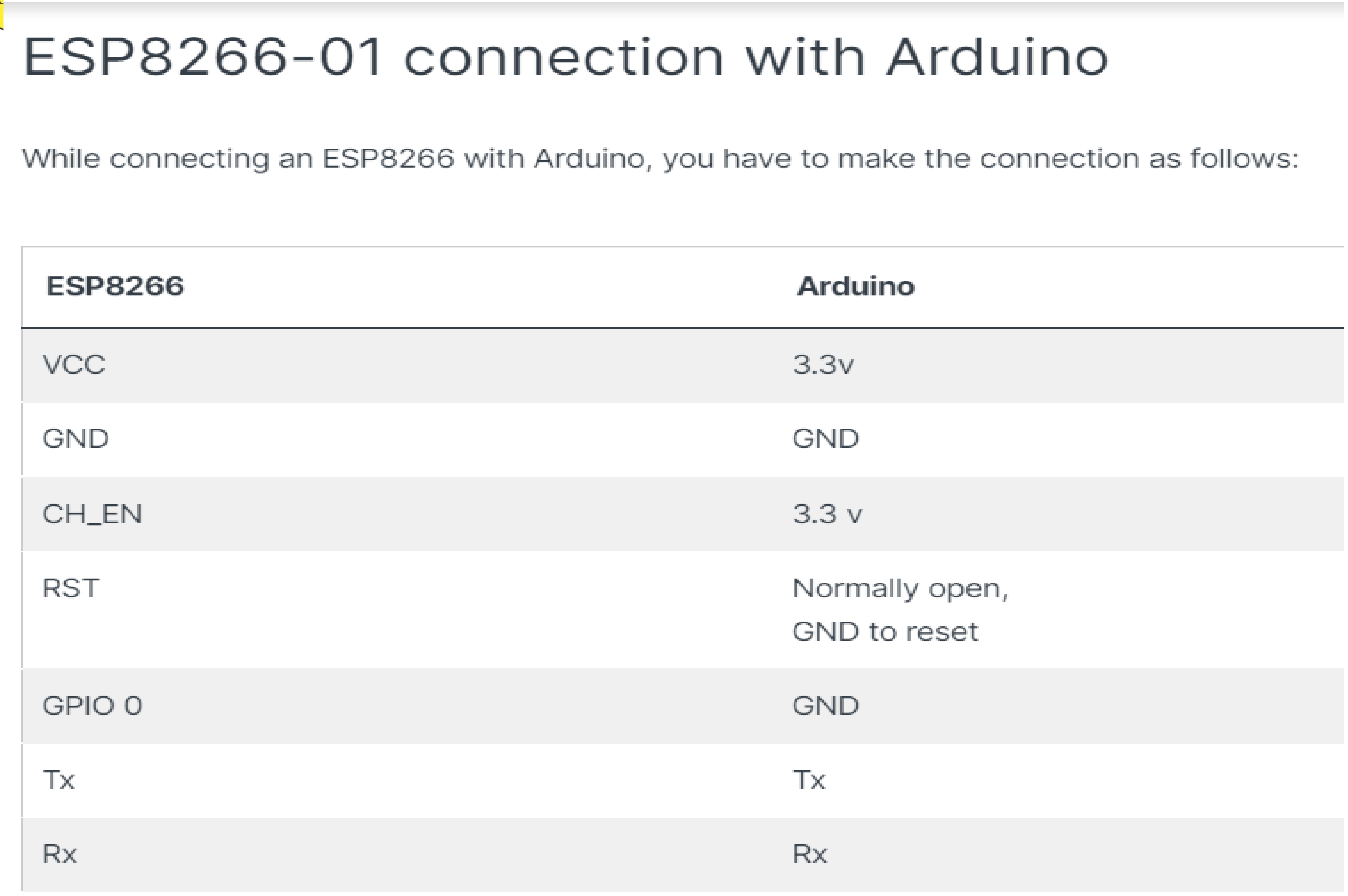
- On-chip Wi-Fi modules
- It has 2 GPIO pins
- Has inbuild 10 bit ADC (Analog to digital converter)
- 32 KB instruction RAM
- 16 KB system data RAM
- 32-bit microcontroller

Pinout of ESP8266





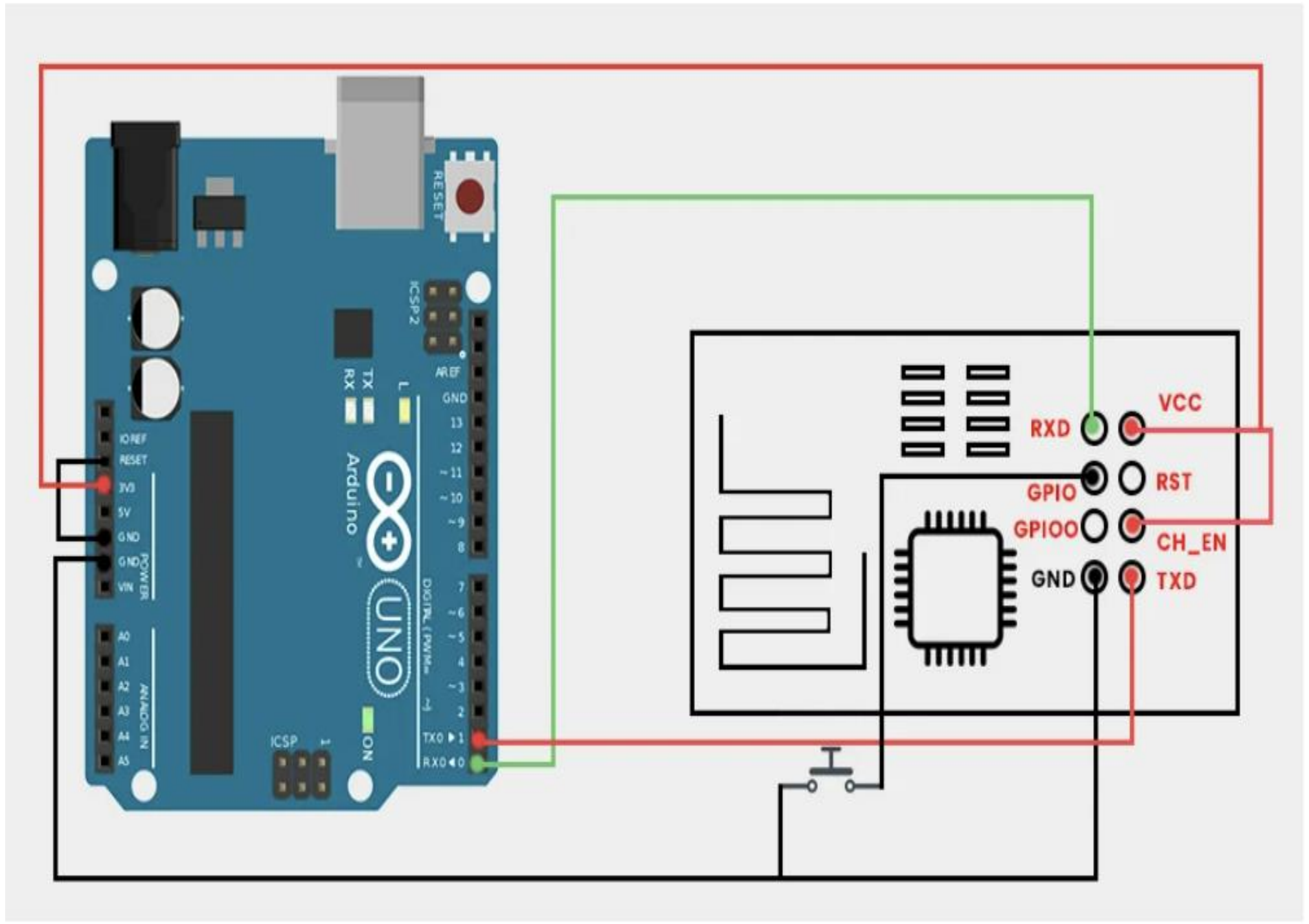
Pin	Description
VCC	This is the power pin for 3.3v
GND	Ground pin for giving 0 volt
Rx	Receiver pin used to receive serial data from another device
Tx	Transmitter pin used to transfer serial data to other devices
CH_En	Chip enable pin, usually connected to 3.3 volt due to active-high property
GPIO 0	General-purpose GPIO pin basically has two used 1) used as a normal GPIO pin 2) used to enable the programming mode of ESP8266
GPIO 2	Used as a GPIO pin



ESP8266-01 connection with Arduino

While connecting an ESP8266 with Arduino, you have to make the connection as follows:

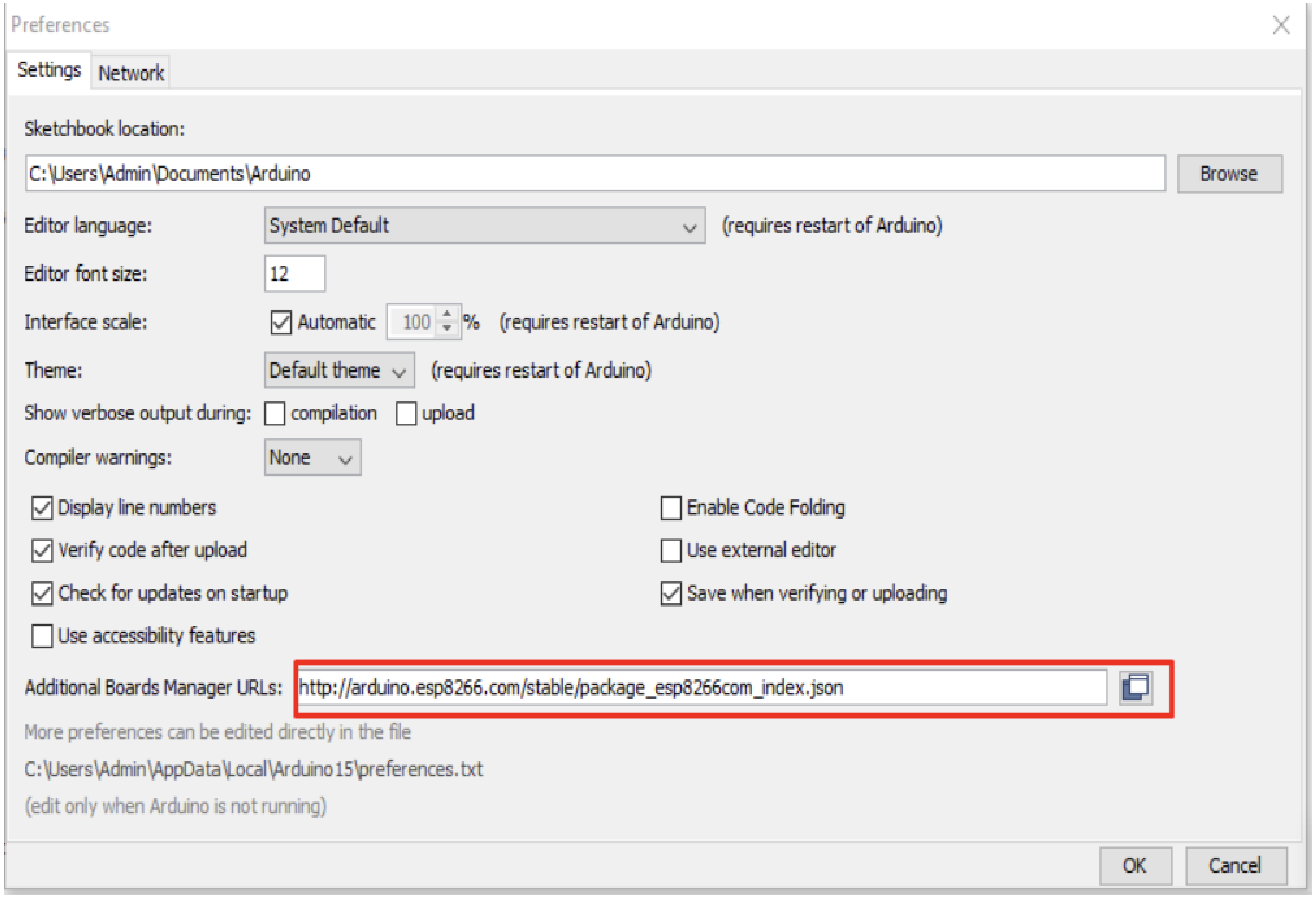
ESP8266	Arduino
VCC	3.3v
GND	GND
CH_EN	3.3 v
RST	Normally open, GND to reset
GPIO 0	GND
Tx	Tx
Rx	Rx


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX0 ▶1
RX0 ◀0
DIGITAL (PWM~)
TX
RX
L
Arduino
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
VCC
RXD
RST
GPIO
GPIO0
CH_EN
GND
TXD

Software, Boards Installation:

You have to follow few simple steps to install ESP8266 in the Arduino IDE:

- First, you need to download the **Arduino IDE**.
- After that, we need to install the ESP8266 Board in Arduino IDE.
- Copy the following link to add ESP8266 or ESP8266 integrated board in Arduino IDE.
- http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Go to Arduino IDE, then follow the path File/preferences and open the preference tab.
- And paste the above link in the additional board manager URL box as shown in the image.



Preferences

Settings | Network

Sketchbook location:

C:\Users\Admin\Documents\Arduino [Browse]

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: [✓] Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: [ ] compilation [ ] upload

Compiler warnings: None

- [✓] Display line numbers
- [✓] Verify code after upload
- [✓] Check for updates on startup
- [ ] Use accessibility features
- [ ] Enable Code Folding
- [ ] Use external editor
- [✓] Save when verifying or uploading

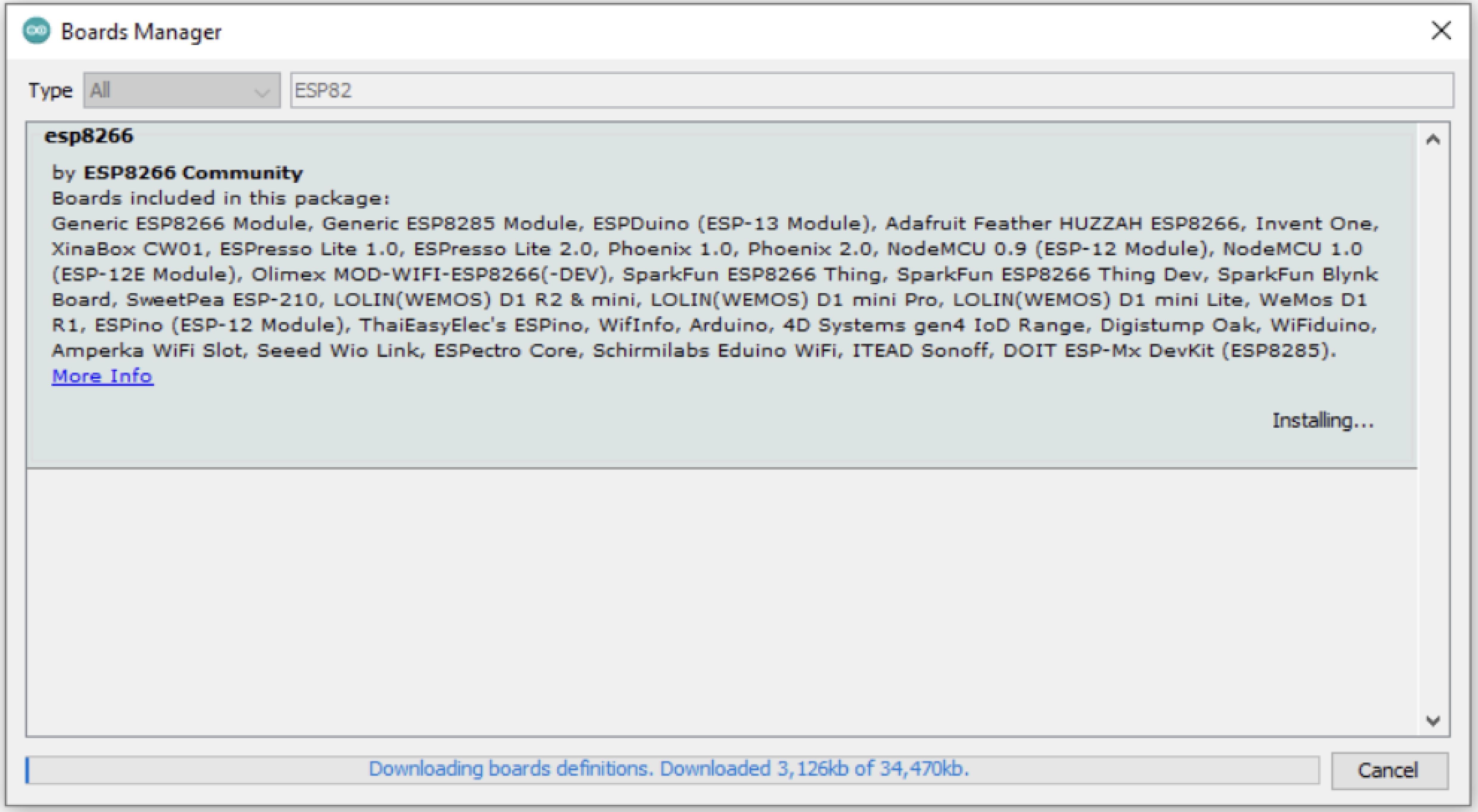
Additional Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

C:\Users\Admin\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

- After this, go to **Tool/ Board Tools/board/board manager** and type **ESP8266**. You will find a board of ESP8266 click on the install option.



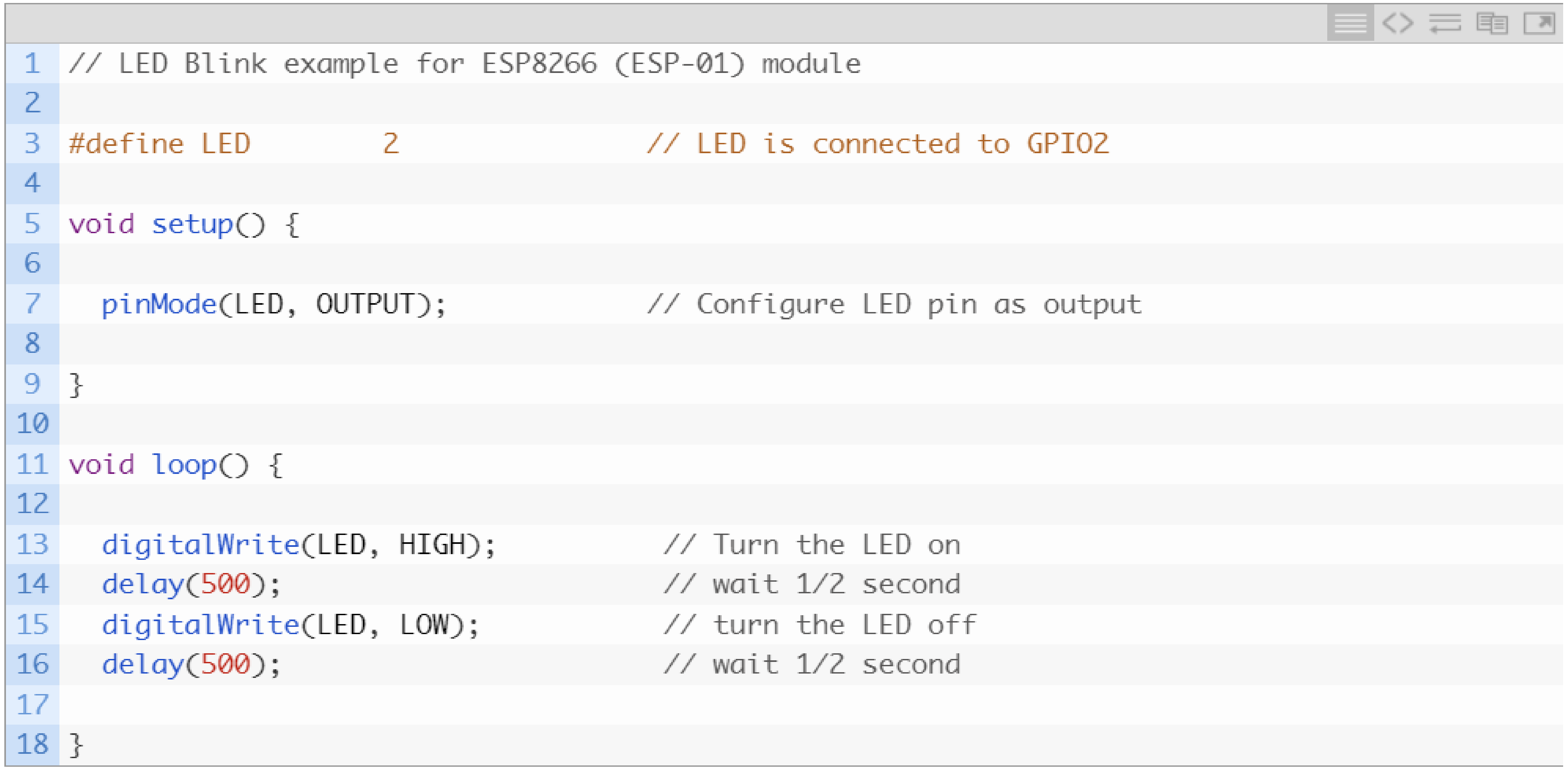
esp8266
by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).
More Info

Installing...

Downloading boards definitions. Downloaded 3,126kb of 34,470kb.

- This is how your ESP8266 board get installed.

Arduino code for ESP8266 module:



```
// LED Blink example for ESP8266 (ESP-01) module

#define LED        2               // LED is connected to GPIO2

void setup() {

  pinMode(LED, OUTPUT);            // Configure LED pin as output

}

void loop() {

  digitalWrite(LED, HIGH);          // Turn the LED on
  delay(500);                       // wait 1/2 second
  digitalWrite(LED, LOW);           // turn the LED off
  delay(500);                       // wait 1/2 second

}
```