# City, University of London

# Department of Electrical and Electronic Engineering

# LAB - MANUAL

# EE3600 / EE3700 Design - III

## DC Motor Control with L298N and Arduino

## T1 LAB - 4



**Version: 2020/2021**

# 1.0 Introduction

In this design lab, we are going to learn about Arduino L298N motor driver module interfacing. The L298N module has a famous L298 motor driver IC which is the main part of this module. This module uses the PWM method to control the speed of DC motors. The inexpensive L298N H-Bridge module is a simple way to achieve this. Coupling the L298N H-Bridge to a microcontroller platform like an Arduino will give you the ability to control both the speed and rotation direction of DC motors.

# 2.0 Required Components

- 1 x Arduino UNO
- 1 × Breadboard
- 1 × L 298N
- 2 × 3V DC Motors
- Connecting wires

# 3.0 Component Description

## 3.1 DC Motors

Today DC motors range from huge models used in industrial equipment to tiny devices that can fit in the palm of your hand. They are inexpensive and are ideal for use in your Robotics, Quadcopter, and Internet of Things (IoT) projects.

In a simple DC motor, there are two main components, the “stator” and the “armature”. The *stator* is a permanent magnet and provides a constant magnetic field. The *armature*, which is the rotating part, is a simple coil.

The armature is connected to a DC power source using a 2-piece ring installed around the motor shaft, these ring sections are called “commutator rings”. The two pieces of the commutator rings are connected to each end of the armature coil. Direct Current (DC) of a suitable voltage is applied to the commutator rings via two “brushes” that rub against the rings.

When DC is applied to the commutator rings it flows through the armature coil, producing a magnetic field. This field is attracted to the stator magnet (remember, opposite magnetic polarities attract, same polarities repel) and the motor shaft begins to spin.

The motor shaft rotates until it arrives at the junction between the two halves of the commutator. At that point, the brushes meet the other half of the commutator rings, reversing the polarity of the armature coil (or coils, most modern DC motors have several). This is great because at this point the motor shaft has rotated 180 degrees and the magnetic field polarities need to be reversed for the motor to continue rotating. This process repeats itself indefinitely until the current is removed from the armature coils.

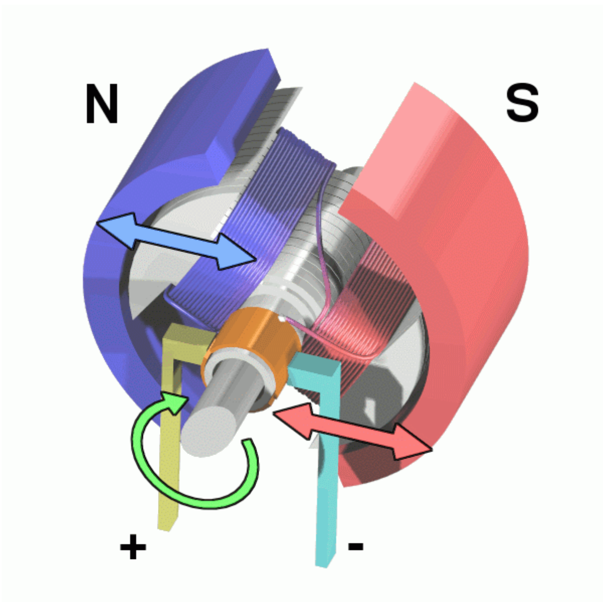


**Figure 1: DC motor**

The motor just described is referred to as a brushed DC motor because (obviously) it has brushes. Brushes, however, create many problems – they can start to wear over time, they rub against the motor shaft and they can even cause sparks as the motor gets older.

Better quality DC motors are the brushless variety. Brushless motors use a more complex arrangement of coils and do not require a commutator. The moving part of the motor is connected to the permanent magnet. Because they do not contain brushes these brushless motors will last longer and are also much quieter than brushed DC motors. Most quadcopter motors are brushless motors.

DC motors are specified by the voltage level at which they operate. Common hobbyist motors run at 6 Volts or 12 volts DC.

To reverse the direction in which the DC motor rotates we simply reverse the polarity of the DC current that we apply to it. Changing the speed, however, is a different story.

One method of changing the speed of a DC motor is to simply reduce its supply voltage. While this will work to some degree it is not a particularly good method of controlling motor speed as lowering the voltage will also lower the torque that the motor can produce. Also, once the voltage drops below a certain point the motor will not rotate at all.

### 3.1.1 Controlling a DC Motor

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

- Pulse Width Modulation (PWM) – For controlling speed
- H-Bridge – For controlling rotation direction

#### 3.1.1.1 Pulse Width Modulation (PWM)

A far better method of controlling DC motors is to use pulse width modulation. With PWM the motor is sent a series of pulses. Each pulse is of the full voltage that the motor can handle so a 6-volt motor will be sent 6-volt pulses while a 12-volt motor will be sent 12-volt pulses. The width of the pulses is varied to control the motor speed, pulses with a narrow width will cause the motor to spin quite slowly. Increasing the pulse width will increase the speed of the motor, as illustrated below.

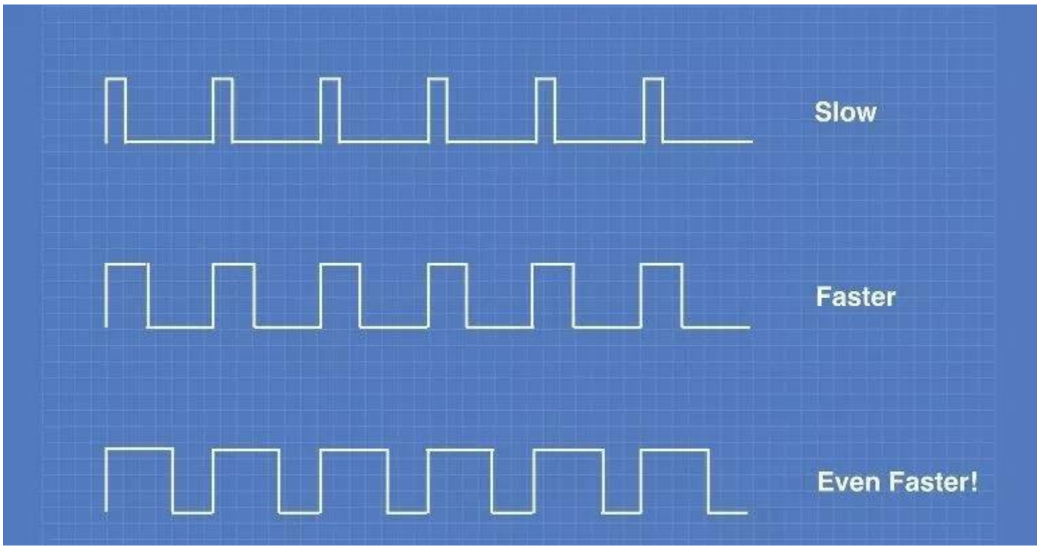


**Figure 2: DC motor pulse width 1**

To stop the motor completely you just stop pulsing it, essentially sending it zero volts. To run it at full speed you supply the full voltage, again without pulsing it.

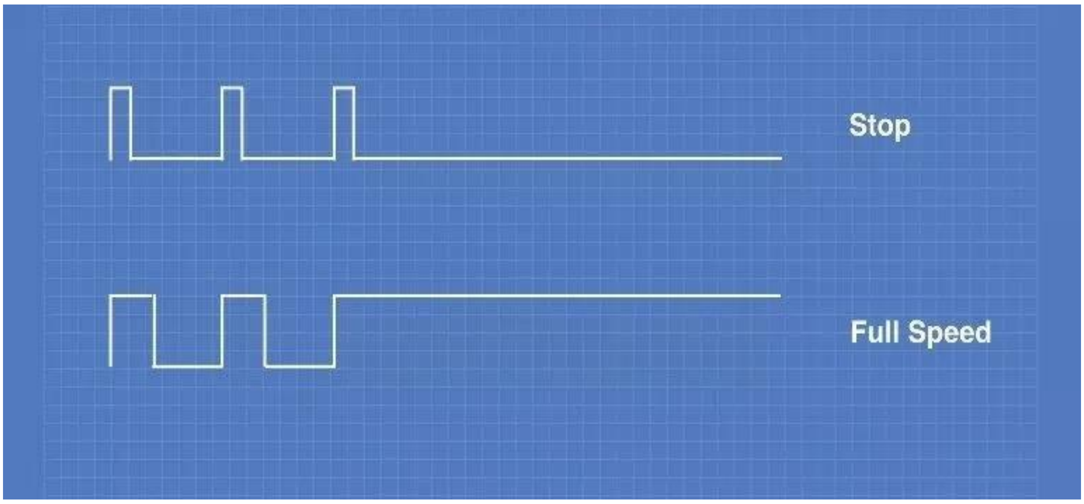


**Figure 3: DC motor pulse width 2**

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 and average voltages.

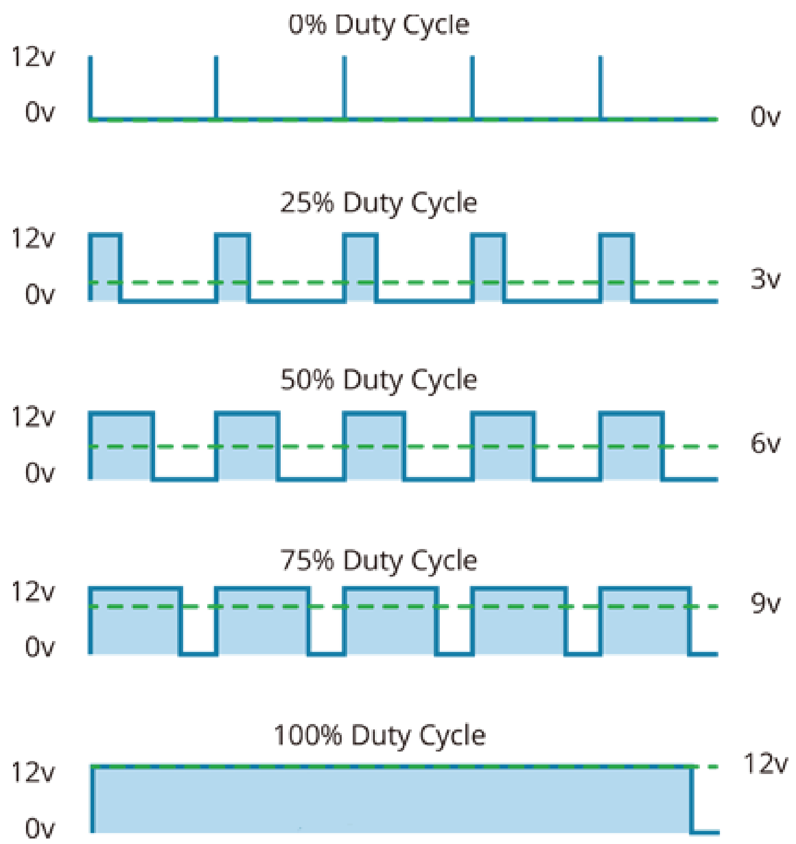


**Figure 4: DC motor duty cycle**

### 3.1.1.2 H-Bridge

Now that you know how DC motors work, we will learn how you can reverse their direction by changing the polarity and how you can change their speed using pulse width modulation. Let us study an easy way to achieve this using a common circuit configuration called an “**H-Bridge**”. An “H-Bridge” is simply an arrangement for switching the polarity of the voltage applied to a DC motor, thus controlling its direction of rotation.

To visualize how this all works together we will use some switches, although in real life an H-Bridge is usually built using transistors. Using transistors also allows you to control the motor speed with PWM, as described above.

In the Figure 5, we can see four switches which are all in the open or “OFF” position. In the centre of the circuit is a DC motor. If you look at the circuit as it is drawn here you can distinctly see a letter “H”, with the motor attached in the centre or “bridge” section thus the term “H-Bridge”.

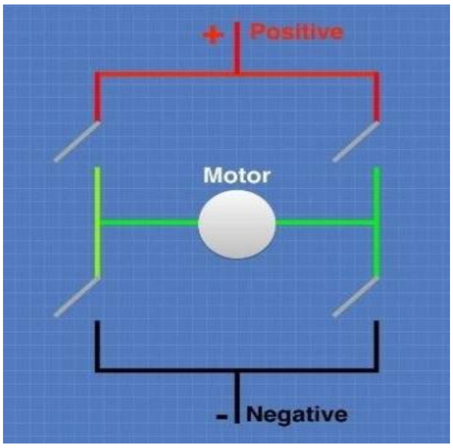


**Figure 5: H bridge control -1**

If we close (i.e. turn on) two of the switches you can see how the voltage is applied to the motor, causing it to turn clockwise.

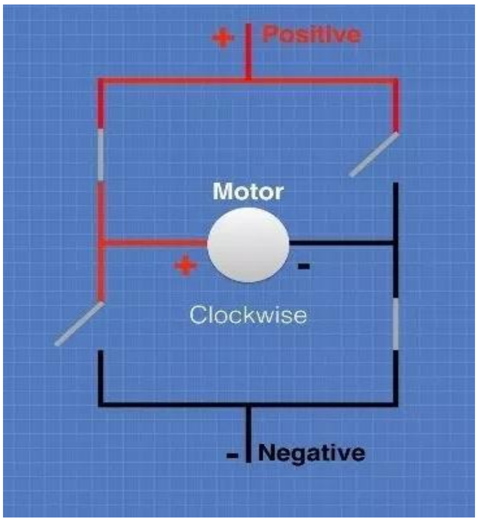


**Figure 6: H bridge control -2**

Now we will open those switches and close the other two. As you can see this causes the polarity of the voltage applied to the motor to be reversed, resulting in our motor spinning counter clockwise.

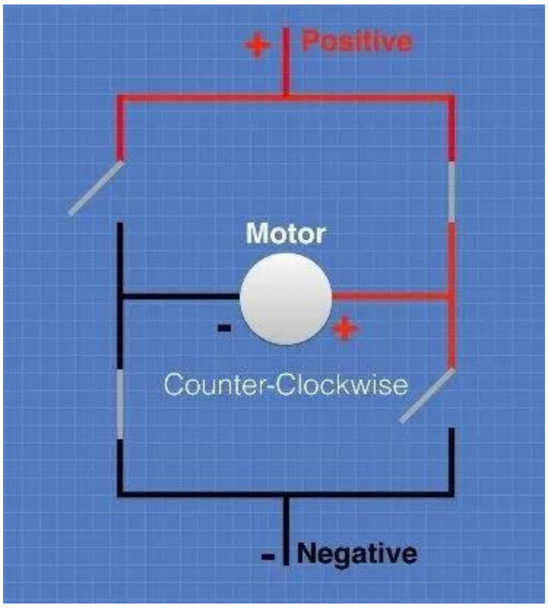


**Figure 7: H bridge control -3**

This is simple but effective. In fact, all you need to do is design a circuit to drive the motor full-speed in either direction you could actually build this as shown here, using a 4PDT (4 Pole Double-Throw) centre-off switch. But of course, we want to control the motor using an Arduino, so an electronic circuit where the switches are replaced by transistors is what we need.

### 3.2 The L 298N H-Bridge

While you can use discrete transistors to build an H-Bridge there are several advantages in using an integrated circuit. Number of H-Bridge motor driver IC's are available and all of them work in pretty much the same fashion. One of the most popular is the L298N.

The L298N is a member of a family of IC's that all have the designation "L298". The difference between the family members is in the amount of current they can handle.

The L298N can handle up to 3 amperes at 35 Volts DC, which is suitable for most hobby motors.

The L298N contains two complete H-Bridge circuits, so it can drive a pair of DC motors. This makes it ideal for robotics projects, as most robots have either two or four powered wheels.

Here is a diagram of the pinouts of an L298N integrated circuit:

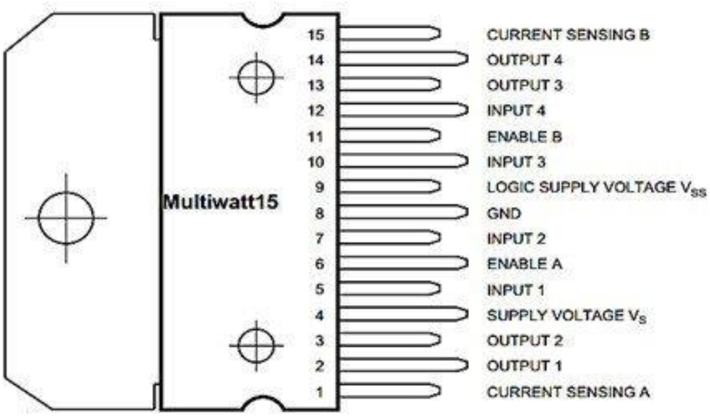


**Figure 6: L 298N IC**

Although you can certainly purchase an L298N integrated circuit and wire it up yourself it is far easier to just buy a complete L298N circuit board, which is wired up and complete with connectors for motors, power supplies and input logic. These boards also have a 5-volt voltage regulator which can be used to supply the logic circuits. L298N driver boards are available from several sources like eBay or your local electronics shop at very reasonable prices.

### 3.2.1 L298N Module Pinouts

You will find a few different styles of L298N boards, but they all operate in the same way. The board contains an L298N mounted on a heatsink, a 5-volt voltage regulator to (optionally) provide power for logic circuits, supporting diodes and capacitors and connectors as follows:

- Logic inputs for each H-Bridge circuit
- Power supply inputs for the motor power supply

- A
  n optional 5 Volt power input for the logic circuits.
- Outputs for each DC motor

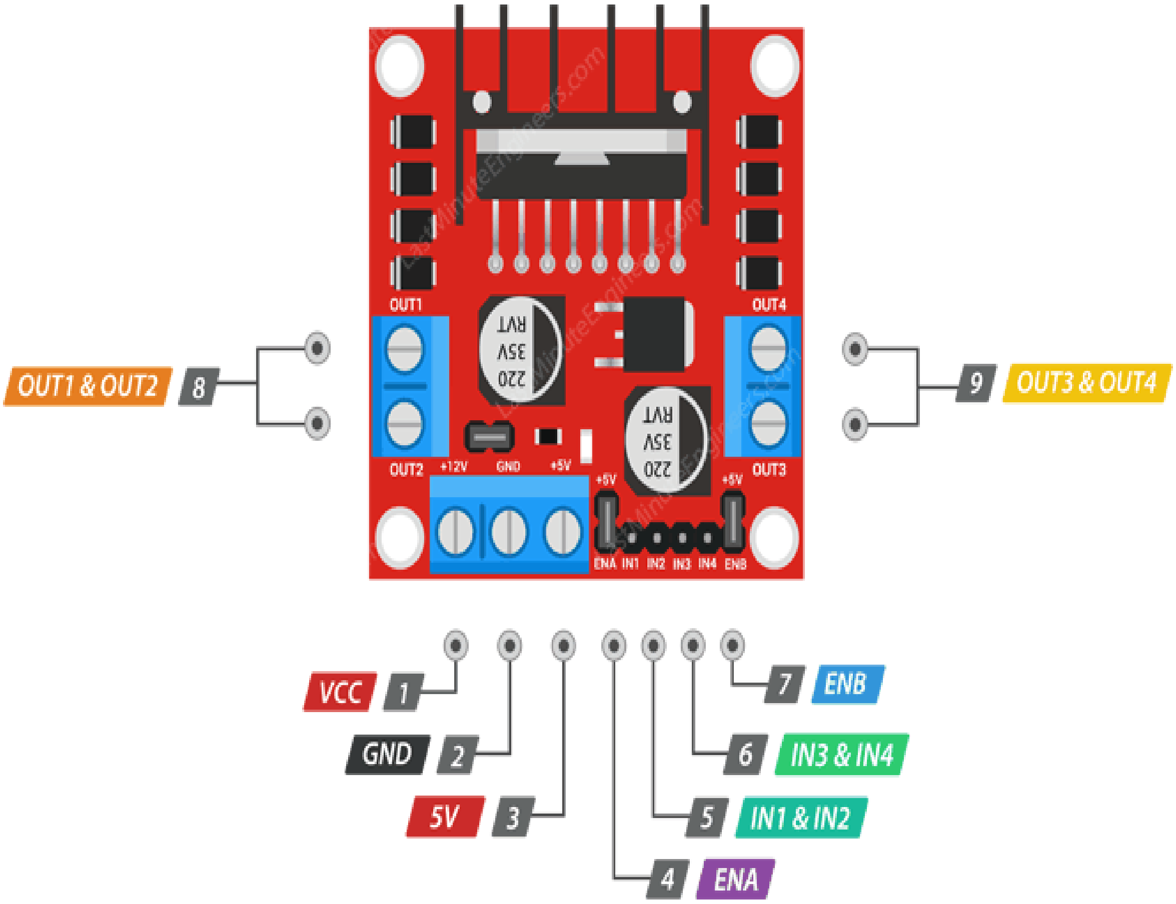


**Figure 7: L298N pin layout**

You will notice that the board also has several jumpers. Most of the time you will leave them in place, except for one. They are as follows:

**VCC** pin supplies power for the motor. It can be anywhere between 5 to 35V.

**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.

**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.

Speaking of the motor power supply it needs to be a bit higher voltage than the actual motor requirements. This is due to the internal voltage drop in the transistors that form the H-Bridge circuit. The combined voltage drop is 1.4 volts, so if you are using 6 Volt motors, you'll need to give the board 7.4 volts, if you have 12-volt motors then your motor supply voltage will need to be 13.4 volts.

To simplify things a bit we shall just discuss the inputs and enable for Motor A, Motor B functions identically. The two Input lines control the direction that the motor rotates. We can call one direction "forward" and the other one "reverse" if it makes more sense to you just substitute "clockwise" and "counter-clockwise".

We control the motor direction by applying either a Logic 1 (5 Volts) or Logic 0 (Ground) to the inputs. The Table 1 illustrates how this is achieved:

| INPUT 1 | INPUT 2 | DIRECTION |
|---|---|---|
| Ground (0) | Ground (0) | Motor Off |
| 5 Volts (1) | Ground (0) | Forward |
| Ground (0) | 5 Volts (1) | Reverse |
| 5 Volts (1) | 5 Volts (1) | Not Used |

**Table 1: Motor input - output combination**

As you can see only two combinations are used to control the direction of the motor's rotation.

The Enable line can be used to turn the motor on, turn it off and to control its speed. When the Enable line is at 5 Volts (1) the motor will be ON. Grounding the Enable line (0) will turn the motor OFF. To control the speed of the motor we apply a Pulse Width Modulation (PWM) signal to the Enable line. The shorter the pulse width, the slower the motor will spin.

**Warning:** DO NOT supply power to both the motor power supply (VCC) input and 5V power supply input when EN (ENA and ENB) jumpers are in place.

## 4.0 Working of the Project

Now we know everything about this design, we can begin connecting it up to our Arduino board. Start by connecting power supply to the motors. In our experiment we are

using DC Motors. They are rated for 1 volt – 6 volts. So, we will connect onboard (Arduino board) 5V power supply to the VCC terminal (**please see page 11 paragraph with bold letters**). Considering internal voltage drop of L298N IC, the motors will receive 3 volts. Motors might spin with slightly lower rpm, but this would not affect our overall design.

Next, we need to supply 5 V for the L298N's logic circuitry. We will make use of the Arduino's on-board 5V to power the logic circuit. So, remove the jumpers from ENA and ENB pins from L298N module.

Now, the input and enable pins (ENA, IN1, IN2, IN3, IN4 and ENB) of the L298N module are connected to six Arduino digital output pins (9, 8, 7, 5, 4 and 3). Note that the Arduino output pins 9 and 3 are both PWM-enabled.

Finally, connect one motor to terminals OUT1 & OUT2 (Motor A) and the other motor to terminals OUT3 & OUT4 (Motor B). You can interchange your motor's connections, technically, there is no right or wrong way.

The input and enable lines in the L298N are driven from six Arduino digital output pins, as follows:

| Arduino | L298N |
|---|---|
| 9 | ENA |
| 8 | IN1 |
| 7 | IN2 |
| 5 | IN3 |
| 4 | IN4 |
| 3 | ENB |

**Table 2: Arduino – L298N pin configuration**

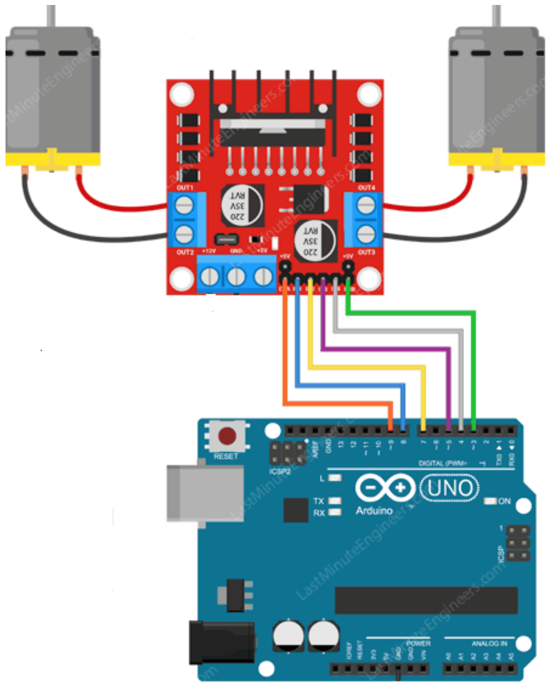


**Figure 8: Arduino, L298N and motor circuit**

## 5.0 Code

Following the hardware setup, we need to write a program that will take care of everything for our design. The program will control the motors according to our required design specification.

## 6.0 Tasks

1. Understand the overall concept of this design,
2. Familiarise yourself with hardware setup,
3. Understand the given sketch for this design,
4. Investigate how to control the motors speed by modifying the given sketch,
5. Change the delay time of the motors and see how motors are responding. Specially vary the delay time between left and right motors.

6. Compare the motor operations with the **Table1** (Page12) data,
7. Connect the ENA and ENB to different pins apart from pins:3,5,6,9,10,11 in the Arduino board and observe the motor operation (don't forget to change the pin numbers in the sketch as well),
8. Suggestions for future developments and implementation ideas.

## 7.0 Marking Scheme

| Description | Marks |
|---|---|
| Total marks for this project | **10** |
| Hardware Design | **04** |
| Oral (each member of the group will be accessed) | **06** |

## 8.0 Sample MCQs for Quiz

8.1) Which function does not return any value when it is called by any other function?

1) AnalogRead()
2) DigitalRead()
3) Serial.println()
4) Serial.Read()

8.2) Arduino (Atmega) pins can source / sink current up to

1) 30 mA
2) 45 mA
3) 40 mA
4) 25 mA

8.3) The use of function pinMode() is not required to set pin as an output pin before calling ___________?

1) DigitalWrite()
2) AnalogRead()
3) Analogreference()
4) AnalogWrite()

8.4) What is the correct syntax to initialize serial communication?

1) Serial.begun(9600)
2) Serial.begin()
3) Serial.begin(9600)
4) Serial.initialize(9600)

# End of Lab - 4