MOTOR DRIVER

L293D

Developed by:

Krishna Nand Gupta
Prashant Agrawal
Mayur Agarwal

►DC Motor

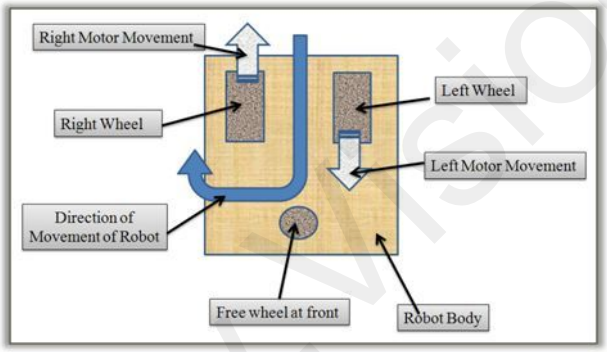
Whenever a robotics hobbyist talk about making a robot, the first thing comes to his mind is making the robot move on the ground. And there are always two options in front of the designer whether to use a DC motor or a stepper motor. When it comes to speed, weight, size, cost... DC motors are always preferred over stepper motors. There are many things which you can do with your DC motor when interfaced with a microcontroller. For example you can control the speed of motor; you can control the direction of rotation.

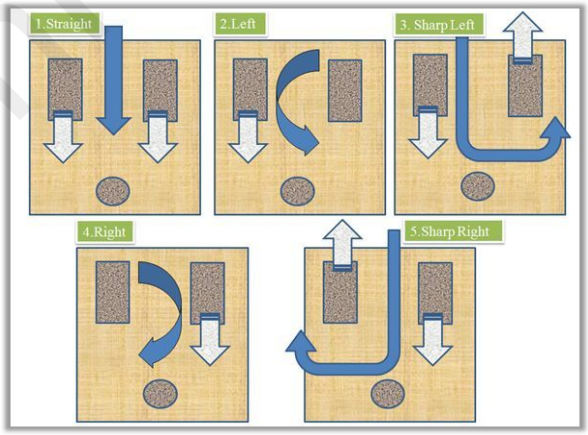
In this part of tutorial we will learn to interface and control of a DC motor with a microcontroller. Usually H-bridge is preferred way of interfacing a DC motor. These days many IC manufacturers have H-bridge motor driver available in the market like L293D is most used H-Bridge driver IC. H-bridge can also be made with the help of transistors and MOSFETs etc. rather of being cheap, they only increase the size of the design board, which is sometimes not required so using a small 16 pin IC is preferred for this purpose.

►Differential drive

By using two motors we can move our robot in any direction. This steering mechanism of robot is called as **differential drive**.

Let's check how it works





Left Motor	Right Motor	Robot Movement
Straight	Straight	Straight
Stop	Straight	Left
Reverse	Straight	Sharp left
Straight	Stop	Right
Straight	Reverse	Sharp Right
Reverse	Reverse	Reverse

►DC Motor with Gear

The DC motors don't have enough torque to drive a robot directly by connecting wheels in it. Gears are used to increase the torque of dc motor on the expense of its speed**.**

Mathematical interpretation:

Rotational power (Pr) is given by:

Pr =Torque (T) * Rotational Speed (ω)

Thus

$$T = \frac{\mathrm{Pr}}{\omega}$$

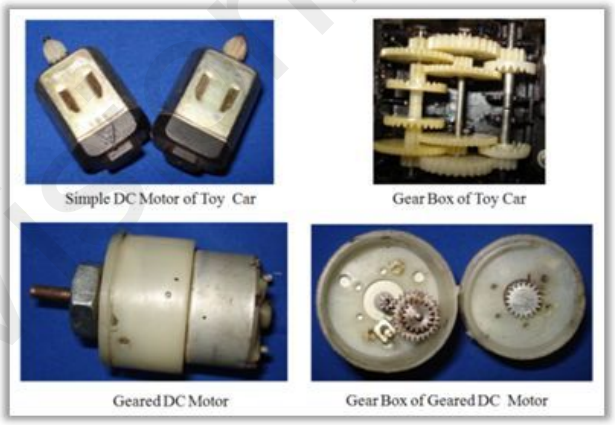
Pr is constant for DC motor for a constant input electrical power. Thus torque (T) is inversely proportional speed (ω).

$$\boxed{T \, \alpha \, \frac{1}{\omega}}$$

Thus to increase the value of torque we have to loose speed.

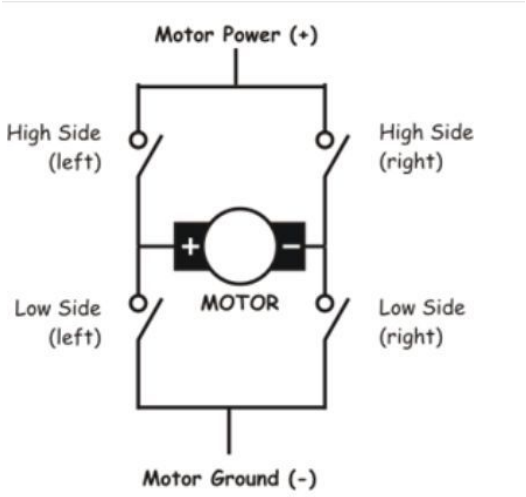
Note:

- In toy car, there is a gear box that contains several combinations of gears.
- Geared motor has gears box at its front.

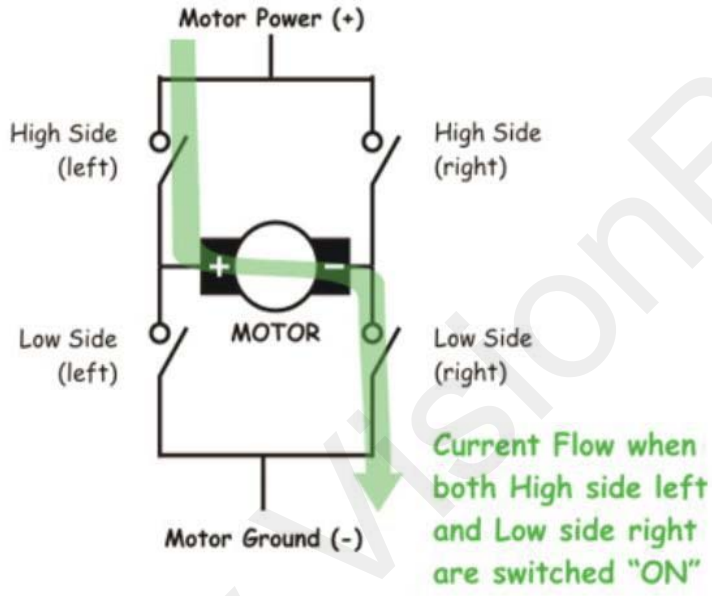


►Working Theory of H-Bridge

The name "H-Bridge" is derived from the actual shape of the switching circuit which controls the motion of the motor. It is also known as "Full Bridge". Basically there are four switching elements in the H-Bridge as shown in the figure below.



As you can see in the figure above there are four switching elements named as "High side left", "High side right", "Low side right", "Low side left". When these switches are turned on in pairs motor changes its direction accordingly. Like, if we switch on High side left and Low side right then motor rotate in forward direction, as current flows from P\power supply through the motor coil goes to ground via switch low side right. This is shown in the figure below.



Similarly, when you switch on low side left and high side right, the current flows in opposite direction and motor rotates in backward direction. This is the basic working of H-Bridge. We can also make a small truth table according to the switching of H-Bridge explained above.

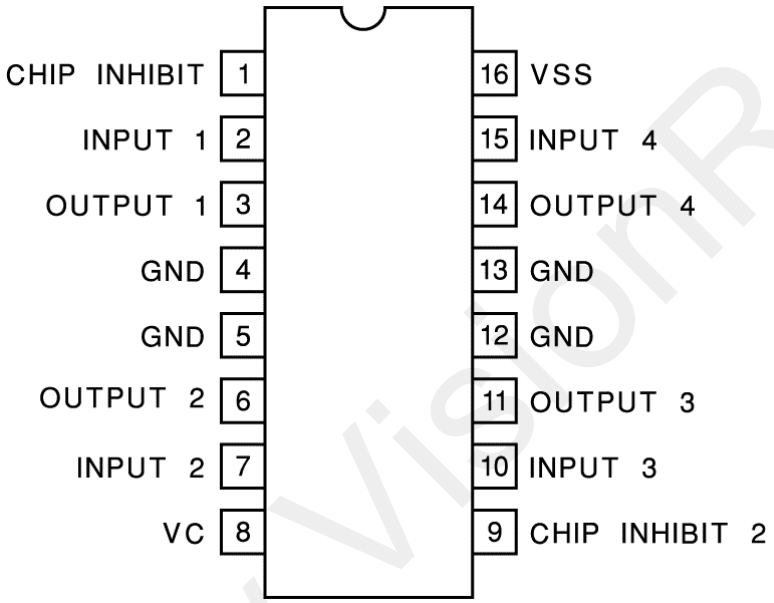
Truth Table				
High Left	High Right	Low Left	Low Right	Description
On	Off	Off	On	Motor runs clockwise
Off	On	On	Off	Motor runs anti-clockwise
On	On	Off	Off	Motor stops or decelerates
Off	Off	On	On	Motor stops or decelerates

As already said, H-bridge can be made with the help of transistors as well as MOSFETs; the only thing is the power handling capacity of the circuit. If motors are needed to run with high current then lot of dissipation is there. So head sinks are needed to cool the circuit.

Now you might be thinking why I did not discuss the cases like High side left on and Low side left on or high side right on and low side right on. Clearly seen in the diagram, you don't want to burn your power supply by shorting them. So that is why those combinations are not discussed in the truth table.

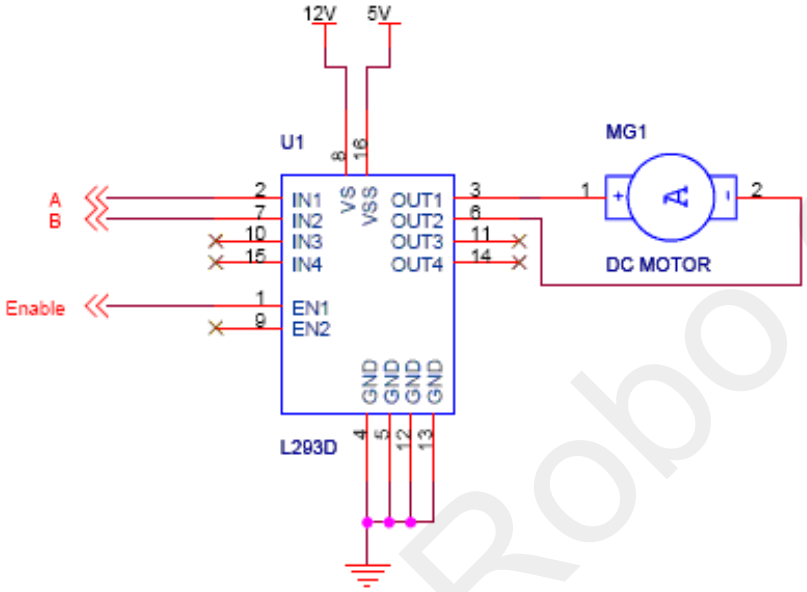
So we have seen that using simple switching elements we can make our own H-Bridge, or other option we have is using an IC based H-bridge driver.

► L293D Dual H-Bridge Motor Driver



L293D is a dual H-Bridge motor driver, so with one IC we can interface two DC motors which can be controlled in both clockwise and counter clockwise direction and if you have motor with fix direction of motion. You can make use of all the four I/Os to connect up to four DC motors. L293D has output current of 600mA and peak output current of 1.2A per channel. Moreover for protection of circuit from back EMF output diodes are included within the IC. The output supply (VCC2) has a wide range from 4.5V to 36V, which has made L293D a best choice for DC motor driver.

A simple schematic for interfacing a DC motor using L293D is shown below.



Truth Table

A	B	Description
0	0	Motor stops or Breaks
0	1	Motor Runs Anti-Cloclwise
1	0	Motor Runs Clockwise
1	1	Motor Stops or Breaks

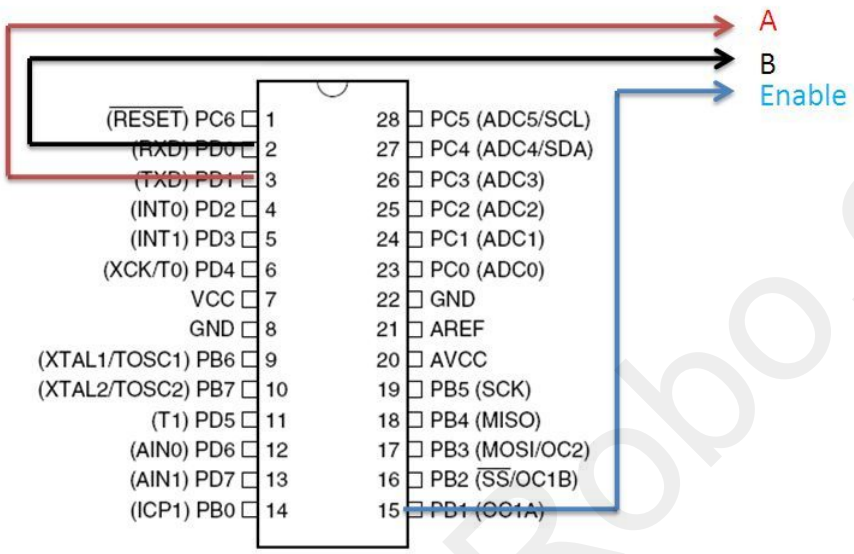
For above truth table, the Enable has to be Set (1). Motor Power is mentioned 12V, but you can connect power according to your motors.

As you can see in the circuit, three pins are needed for interfacing a DC motor (A, B, Enable). If you want the o/p to be enabled completely then you can connect Enable to VCC and only 2 pins needed from controller to make the motor work.

As per the truth mentioned in the image above its fairly simple to program the microcontroller. Its also clear from the truth table of BJT circuit and L293D the programming will be same for both of them, just keeping in mind the allowed combinations of A and B.

►Interface L293D with AVR

From Here I talk specifically with respect to our robot to control a motor using AVR we need to control 3 pins as shown in above A, B and corresponding enable. Connections of motor to ATmega8 are as shown below.



So if you want to run a motor in forward direction your code will be

```
#include <avr/io.h>
int main()
{
        Step 1   Set pin2 (PD0) and pin3 (PD1) as output pin;
        Step 1   Set pin15 (PB1) as output pin;
        Step 2   Send 0 at PD0 and 1 at PD1;
        Step 3   Send 1 at PB1;
While(1)
        {
              //do any job here
        }
        Return 0;
}
```

All 4 steps here related to input output pin configuration of AVR so I will write code directly for detail check I/O tutorial

```
#include <avr/io.h>
int main()
{       DDRD|= ((1<<PORTD0)|(1<<PORTD1));
        DDRB|= (1<<PORTB1);
        PORTD|= 1<<PORTD1;
        PORTD&= ~(1<<PORTD0);
        PORTB|= (1<<PORTB1);
While(1)
        {
                //do any job here
        }
        Return 0;
}
```

What if you want to oscillatory motion 2 second forward then to 2 second reverse then 2 second break

```
#include <avr/io.h>
#include<avr/delay.h>
int main()
{       DDRD|= ((1<<PORTD0)|(1<<PORTD1));
        DDRB|= (1<<PORTB1);
        PORTB|= (1<<PORTB1);
        uint8_t  i;
While(1)
        { //Forward
                PORTD|= 1<<PORTD1;
                PORTD&= ~(1<<PORTD0);
                _delay_ms(100);
        //reverse
                PORTD|= 1<<PORTD0;
                PORTD&= ~(1<<PORTD1);
                _delay_ms(100);
        //stop
                PORTD&= ~((1<<PORTD0)|(1<<PORTD1));
                _delay_ms(100);
        }
        Return 0;
        }
```

►Motor speed control using PWM

If and only if you have gone though PWM tutorial proceed further. Suppose you want to run motor by half of the its rating speed then send 50% duty cycle square wave at enable pin effectively you will get 50% on tine but due to high frequency and inertia motor seems to run continuously so code will be

```
#include <avr/io.h>

int main()

{
        DDRD|= ((1<<PORTD0)|(1<<PORTD1));
        DDRB|= (1<<PORTB1);
        PORTD|= ((0<<PORTD0)|(1<<PORTD1));
        // ######################### PWM code to get 50% duty cycle square wave at
enable (PB1)
        TCCR1A=0xA1;
        OCR1A=128;
        TCCR1B=0x04;
        //#########################
While(1)
        {
                //do any job here
        }
        Return 0;
}
```


HAPPY MOTOR CONTROL ☺