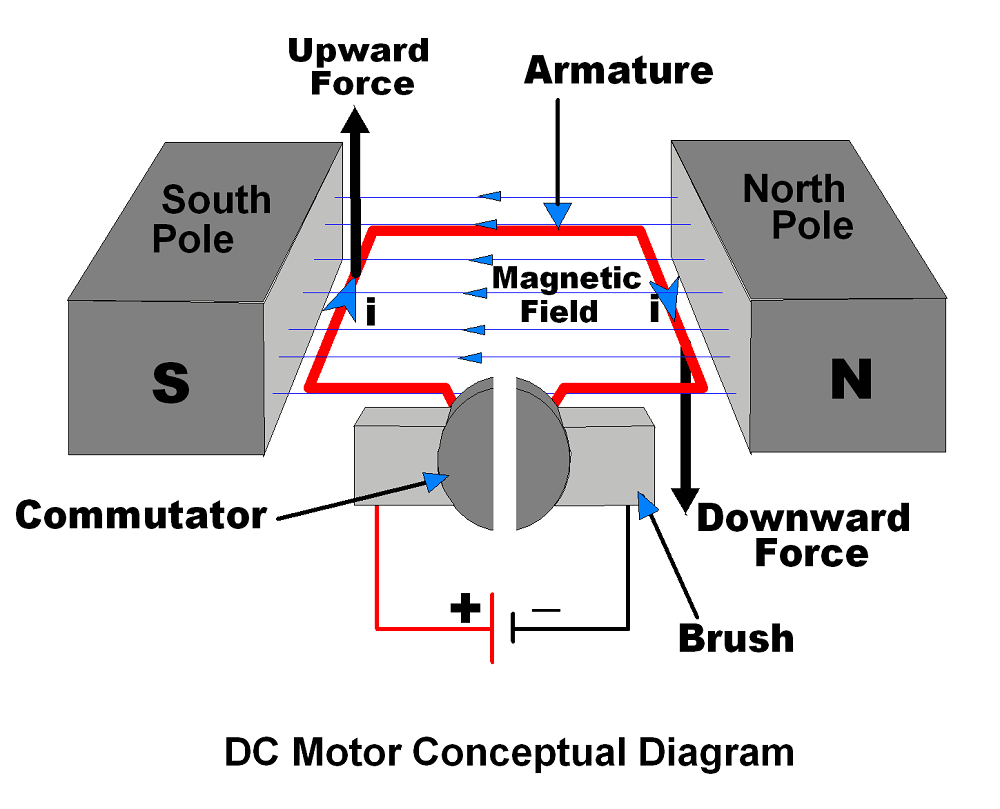
**Interfacing DC Motor with Arduino UNO**

**DC Motor:**



* DC motor uses Direct Current (electrical energy) to produce mechanical movement i.e. rotational movement. When it converts electrical energy into mechanical energy then it is called as DC motor and when it converts mechanical energy into electrical energy then it is called as DC generator.
* The working principle of DC motor is based on the fact that when a current carrying conductor is placed in a magnetic field, it experiences a mechanical force and starts rotating. Its direction of rotation depends upon Fleming’s Left Hand Rule.
* DC motors are used in many applications like robot for movement control, toys, CD/DVD disk drive in PCs/Laptops etc.

**Construction:**



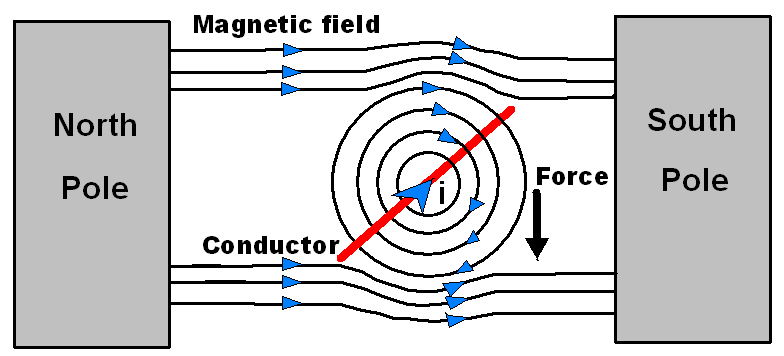
It has mainly two major parts as,

1. **Stator** – Static part of the motor.
2. **Rotor** – Rotating part of the motor.

* The South and North poles of permanent magnet or Electromagnet are the stator part of the DC motor and armature connected with commutator is rotating part of the DC motor.
* South and North poles are used to create a magnetic field as shown in figure.
* The Armature is a conducting material which is placed in between magnetic field produced by North & South pole.
* The current (i) shown in figure is flowing through Armature.
* Brushes are used to attach DC supply to the Armature via commutator.
* Commutators have segments which are attached with each end of conducting Armature. Hence, commutator also rotates with Armature. Brushes are stator part which always keep in contact with commutator.

**Principle of working:**

As shown in above figure left hand side of armature lifting upward and right hand side of armature going downward. This is because of force, which depends upon direction of magnetic field and direction of current flowing through armature. [Fleming’s Left Hand Rule](https://en.wikipedia.org/wiki/Fleming%27s_left-hand_rule_for_motors) is used to determine direction of force (upward/downward).



Let’s see how force act on armature conductor.

* As shown in above figure, the current (i) direction flowing through conductor is inward, hence magnetic field generated around conductor is having direction as per [Right Hand Grip Rule](https://en.wikipedia.org/wiki/Right-hand_rule#/media/File:Manoderecha.svg) shown in figure.
* The magnetic field in between North and South poles having direction from North to South as shown in figure.
* Magnetic lines of forces generated by current carrying conductor and by two poles are shown in figure. These both force lines look in same direction in the above half part of conductor whereas in the below half part of conductor look in opposite direction.
* Hence concentration of magnetic lines of force is more in above part of the conductor which result in the force that moves conductor downward.

This is for one side of armature conductor whereas on other side of armature conductor current direction will be opposite as shown in constructional diagram and hence force will move the conductor of armature in opposite direction. Hence two side of armature move in upward and downward direction which results in rotation of armature.

Also we can change rotation direction (Clockwise/Anticlockwise) of DC motor by simply changing polarity of applied voltage at motor terminals.

**Rotation speed of DC Motor/**

**Speed Controlling of DC Motor**

Speed (N) of DC motor is measured in RPM (Rotation Per Minute) and it is given by,

N = 60AE / PZ Φ

Where,

                        E = Back EMF

                        A = Parallel paths

                        Z = No. of armature conductors

                        P = No. of poles

                        Φ = Flux

Device Constant          K = 60A/PZ

Back EMF        E = V – IARA

Hence, speed   N = K \* (V – IARA) / Φ

Here, we can see that speed of DC motor can be controlled through,

* Terminal voltage of armature i.e. V
* External resistance with armature i.e. RA
* Field flux i.e. Φ

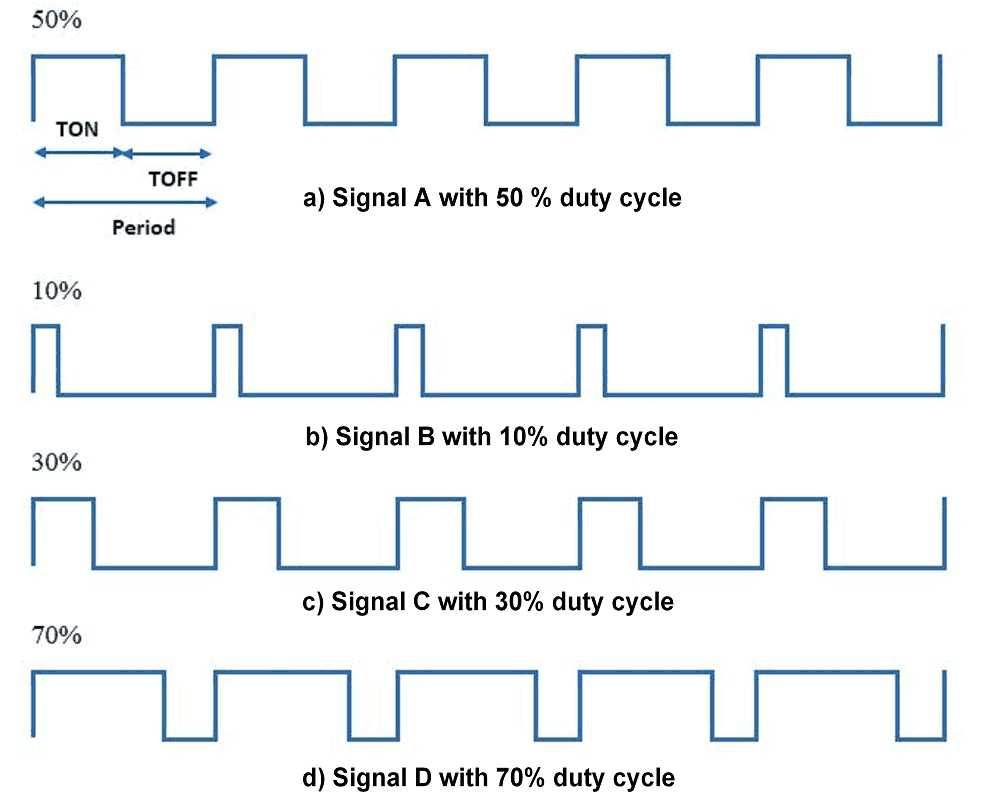
From above speed control parameters, we can find that V and RA are related with armature circuit and Φ related with magnetic field, hence they are classified as,

* Armature control method
* Field control method

The DC motor speed can be controlled by applying varying DC voltage; whereas the direction of rotation of the motor can be changed by reversing the direction of current through it.

**PULSE WIDTH MODULATION TECHNIQUE:**

Pulse Width Modulation is popular technique to control speed of DC motors. It controls average voltage (V) applied to the DC motor terminals by means of pulse width as shown in below figure.



TON is the time for which signal is HIGH and TOFF is the time for which it is LOW. So terminal voltage applies to DC motor is only for TON (ON) time of Period.

E.g.

If PWM with 50% duty cycle as shown in above figure it will provide average ≈50% voltage to the motor terminal.

So in this way we get simple DC motor speed control using PWM method.

Higher duty cycle gives higher speed and lower duty cycle gives lower speed.

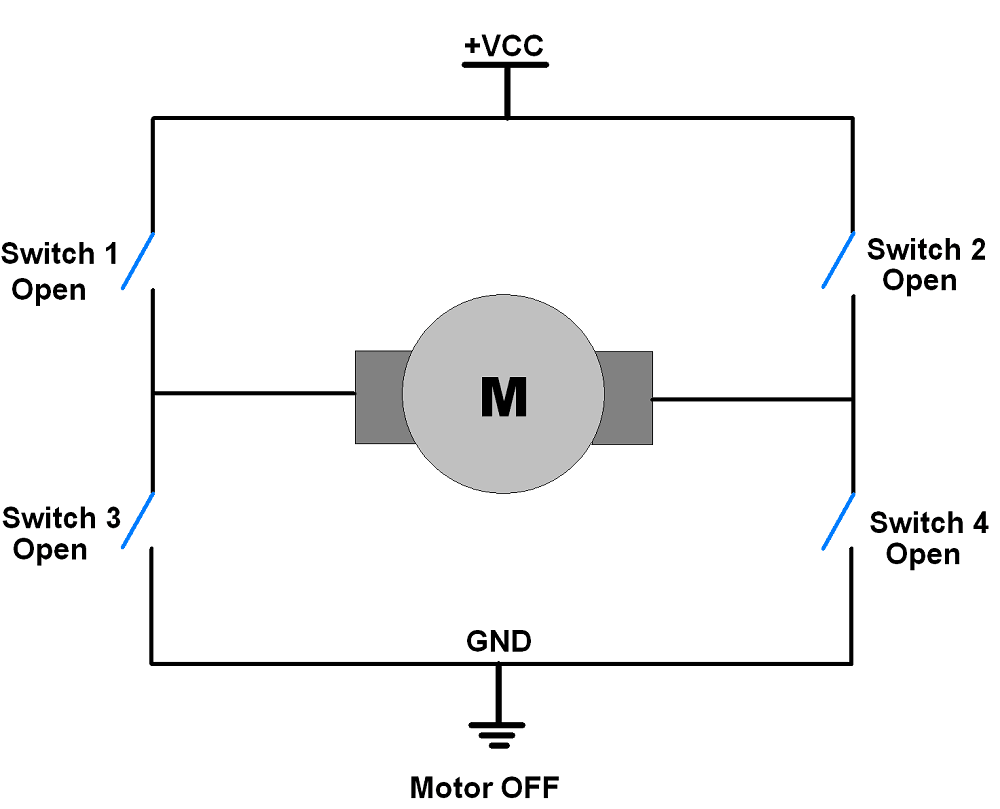
We can vary pulse width precisely using Microcontroller to get fine control over DC motor.

Now, we will see how to change rotational directions of DC motors.

**Bidirectional DC Motor Using H-Bridge Configuration**

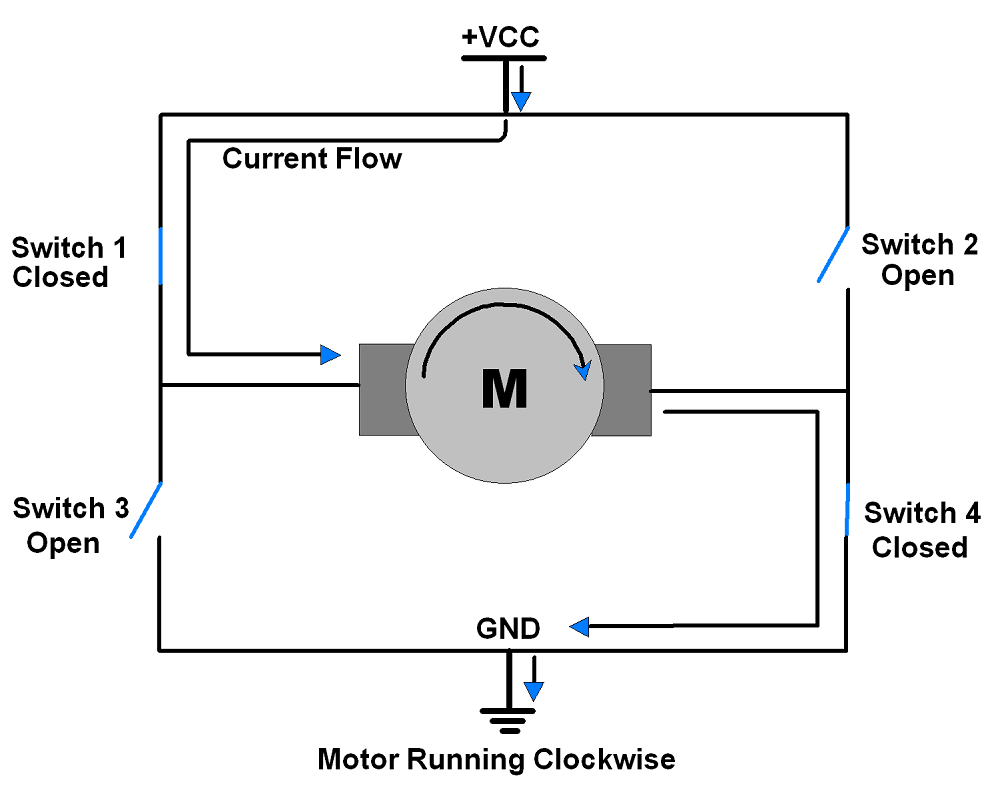
DC motors are Bidirectional i.e. we can rotate DC motor in either direction (Clockwise/Anticlockwise) by just altering their terminal polarity. We can control DC motor direction by using H-Bridge concept as shown in below figure.

-  As all switches are OFF, Motor remains disconnected i.e. OFF.



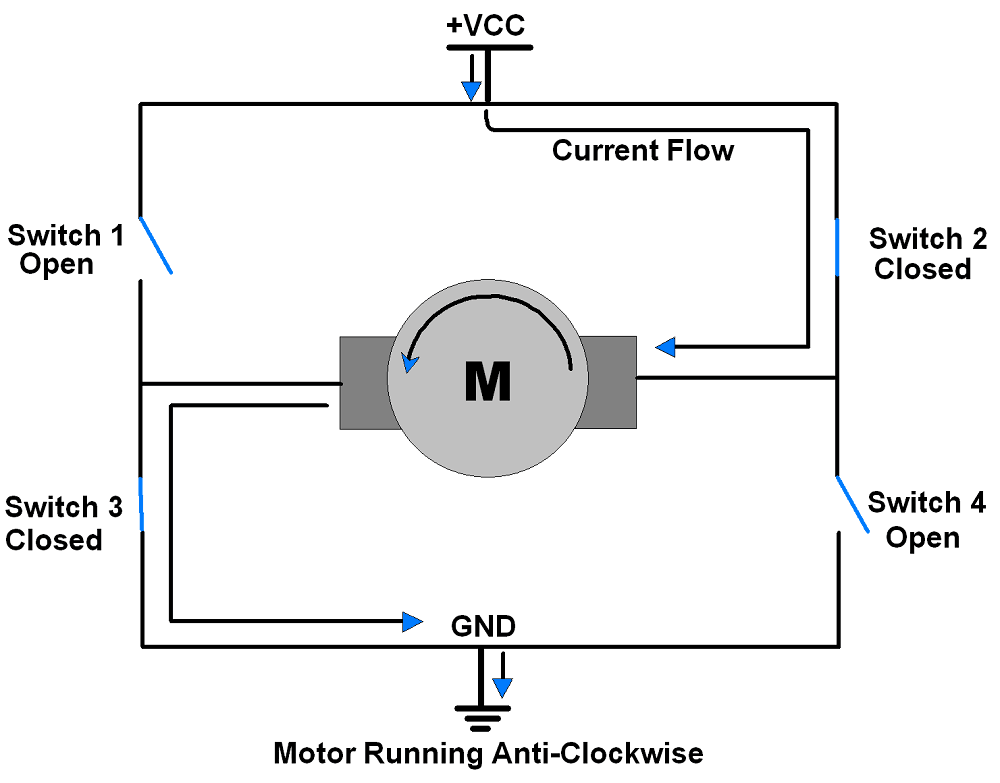
Condition: Switch 1and Switch 4 are closed

Switch 2, Switch 3 is open. In this condition Motor will start rotating in **Clockwise direction**.

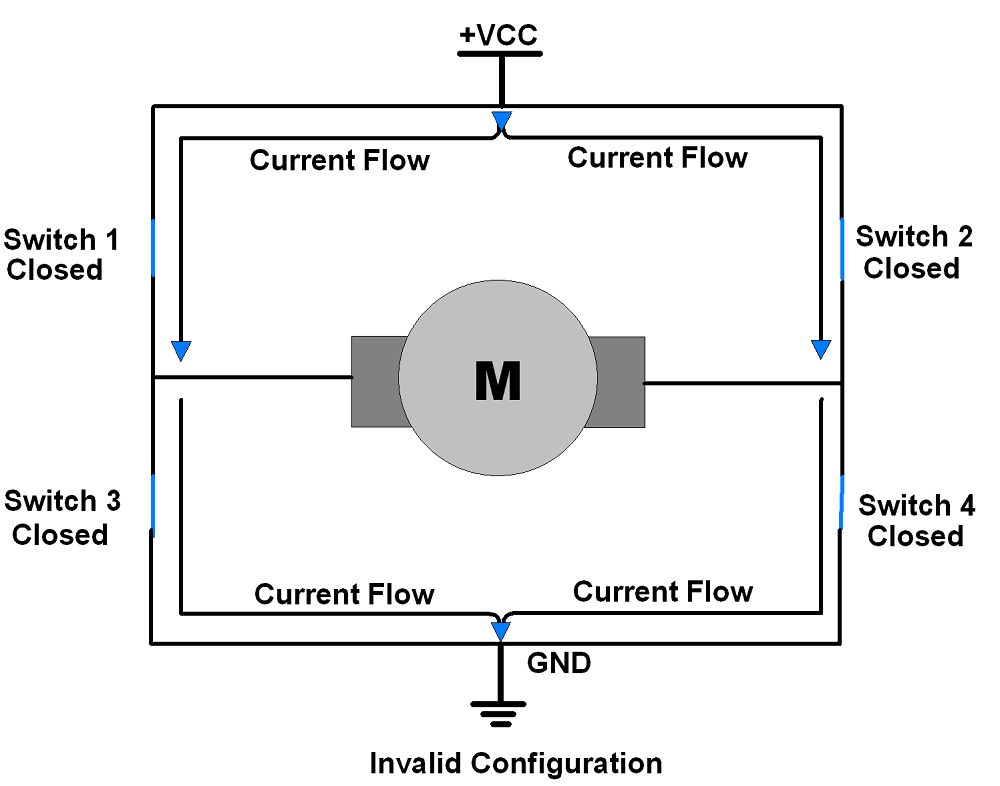


Condition: Switch 3and Switch 2 are closed.

Switch 1, Switch 4 are open. In this condition Motor will start rotating in **Anticlockwise direction**.

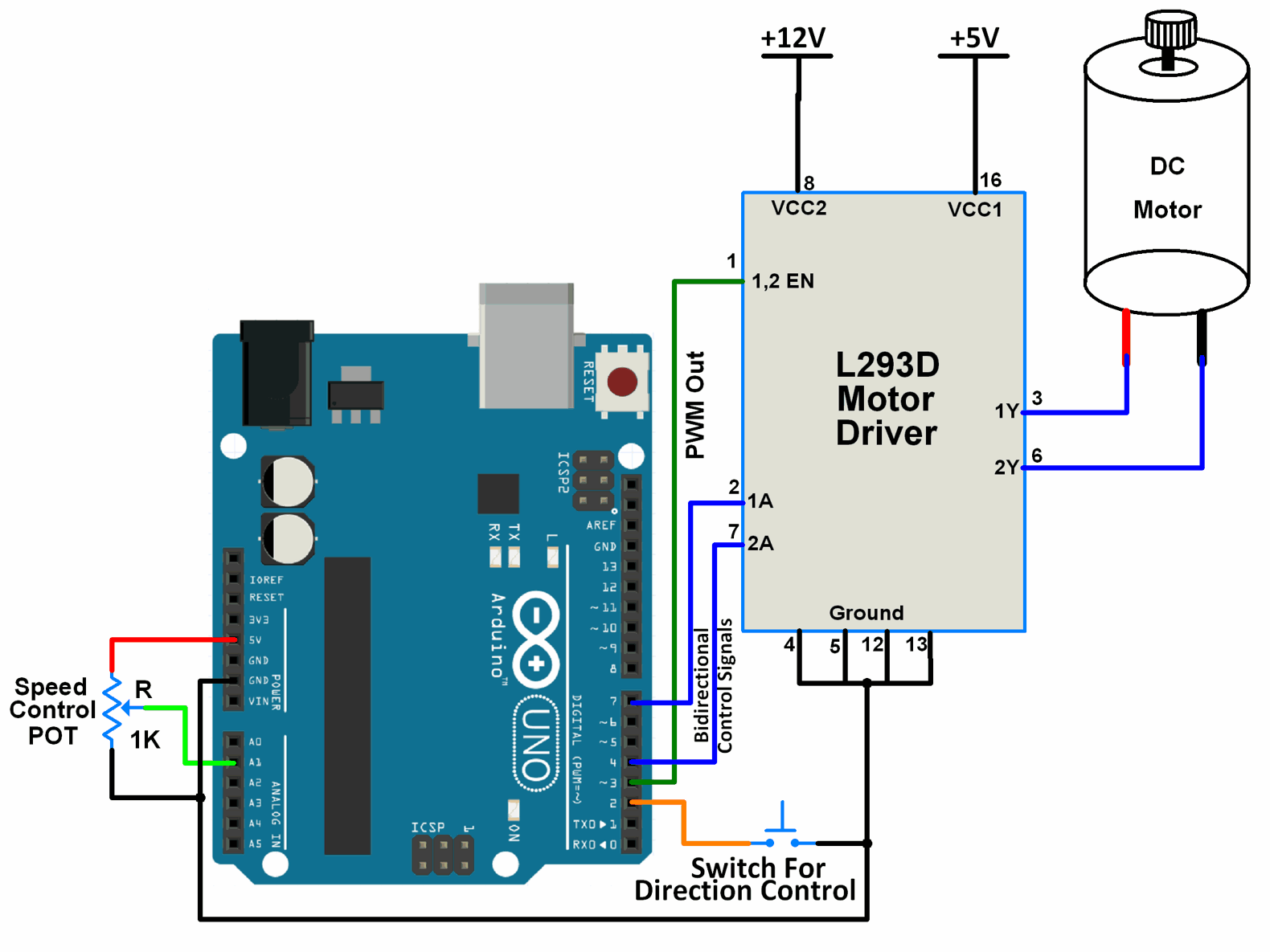


Another configuration when all switches are closed which is known as invalid due to its short circuit type connection as shown in below figure.



In this way we can control DC motor in both of its directions. L293D motor driver IC can be used control DC motor movement in both directions. It has in-built H-bridge motor drive.

**Circuit Diagram:**

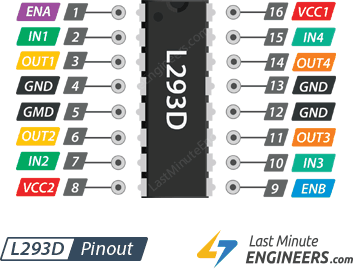


**Objectives:**

To control the speed and rotational direction of DC motor using Arduino Uno.

Here, a potentiometer is used as a means for speed control and an input from a tactile switch is used to change the direction of the motor.

L293D motor driver IC is used for controlling the direction of the motor.



PWM wave generated on the Arduino UNO is used to provide a variable voltage to the motor through L293D. In Arduino, analogWrite function is used to generate PWM wave.

**Code for direction and speed control:**

const int POT\_input = A1; /\* assign ADC Channel \*/

bool d1 = HIGH;

bool d2 = LOW;

void setup() {

pinMode(4, OUTPUT); /\* Motor control pin 1 \*/

pinMode(7, OUTPUT); /\* Motor control pin 2 \*/

pinMode(3, OUTPUT); /\* PWM pin for Speed Control \*/

pinMode(2, INPUT\_PULLUP); /\* Interrupt pin for direction control \*/

attachInterrupt(digitalPinToInterrupt(2), motor, FALLING); /\* Interrupt on falling edge on pin 2 \*/

}

void loop() {

int pwm\_adc;

pwm\_adc = analogRead(POT\_input); /\* Input from Potentiometer for speed control \*/

digitalWrite(4,d1);

digitalWrite(7,d2);

analogWrite(3, pwm\_adc / 4);

}

void motor(){

d1 = !d1;

d2 = !d2;

\_delay\_ms(200);

}

**Functions Used:**

**1.  digitalPinToInterrupt(pin)**

* This function is used to declare the digital pin as an interrupt pin.
* Example,digitalPinToInterrupt(2) is used to declare digital pin 2 as an interrupt pin.
* On UNO board, only pins 2 and 3 can be configured as interrupt pins. Hence, argument to this function can only be pin 2 or pin 3.

**2.  attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)**

* This function is used to configure the mode of interrupt event and declare the ISR for that interrupt. The interrupt event and ISR is for the interrupt pin declared by the function digitalPinToInterrupt(pin).
* ISR in this function is the name of the ISR that will be used for this interrupt.
* mode defines when the interrupt will be triggered. There are four modes available to choose from :  
  - LOW : trigger the interrupt whenever the pin is low.  
  - CHANGE : trigger the interrupt whenever the pin changes value.  
  - RISING : trigger when the pin goes from low to high.  
  - FALLING : trigger when the pin goes from high to low.
* Example, attachInterrupt(digitalPinToInterrupt(2), motor, FALLING) configures digital pin 2 as an interrupt pin with ISR named motor and which generates interrupt for every falling edge event on pin 2.

**3.  analogWrite(pin,value)**

* This function is used for generating PWM on PWM digital pins(pins 3,5,6,9,10,11 for Arduino UNO).
* value can be any number between 0 to 255. 0 being 0% duty cycle and 255 being 100% duty cycle.

**DC motor control with L293D Motor Driver IC and Arduino**

 One of the easiest and inexpensive way to control DC motors is to interface **L293D Motor Driver IC** with Arduino. It can control both speed and spinning direction of two DC motors.

**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**

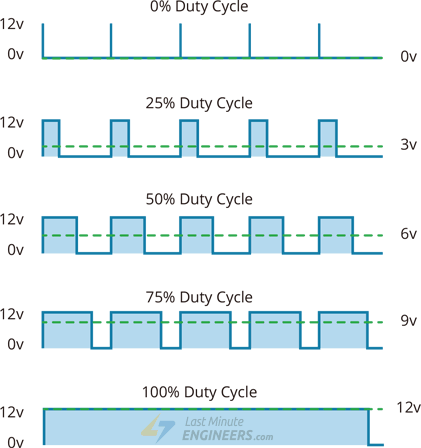
The speed of a DC motor can be controlled by varying its input voltage. A common technique for doing this is to use PWM (Pulse Width Modulation)

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

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.



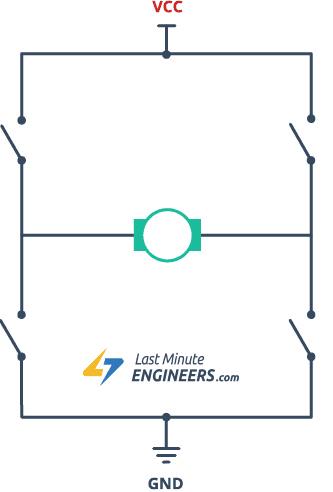
**H-Bridge – For controlling rotation direction**

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.

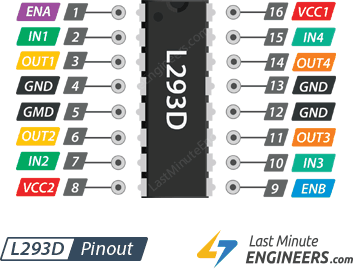
An H-Bridge circuit contains four switches with the motor at the center forming an H-like arrangement.

Closing two particular switches at the same time reverses the polarity of the voltage applied to the motor. This causes change in spinning direction of the motor.

Below figure illustrates H-Bridge circuit working.



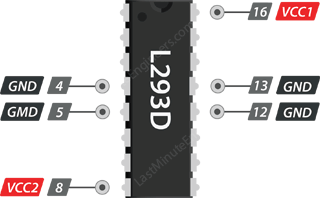
**L293D Motor Driver IC**



The L293D is a dual-channel H-Bridge motor driver capable of driving a pair of DC motors or one stepper motor.

That means it can individually drive up to two motors making it ideal for building two-wheel robot platforms.

**Power Supply**

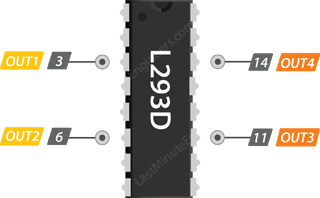


The L293D motor driver IC actually has two power input pins viz. ‘Vcc1’ and ‘Vcc2’.

Vcc1 is used for driving the internal logic circuitry which should be 5V.

From Vcc2 pin the H-Bridge gets its power for driving the motors which can be 4.5V to 36V. And they both sink to a common ground named GND.

**Output Terminals**



The L293D motor driver’s output channels for the motor A and B are brought out to pins OUT1,OUT2 and OUT3,OUT4 respectively.

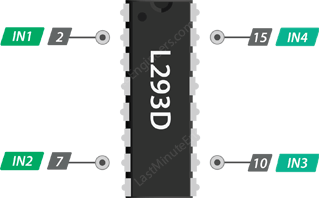
You can connect two DC motors having voltages between 4.5 to 36V to these terminals.

Each channel on the IC can deliver up to 600mA to the DC motor. However, the amount of current supplied to the motor depends on system’s power supply.

**Control Pins**

For each of the L293D’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.

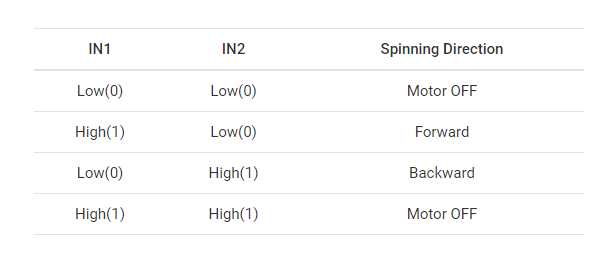
**Direction Control Pins**



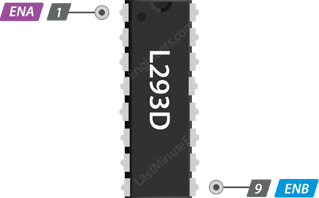
Using the direction control pins, we can control whether the motor spins forward or backward. These pins actually control the switches of the H-Bridge circuit inside L293D IC.

The IC has two direction control pins for each channel. The IN1,IN2 pins control the spinning direction of the motor A while IN3,IN4 control motor B.

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

****

**Speed Control Pins**



The speed control pins viz. ENA and ENB are used to turn ON, OFF and control speed of motor A and motor B respectively.

Pulling these pins HIGH will make the motors spin, pulling it LOW will make them stop. But, with Pulse Width Modulation (PWM), we can actually control the speed of the motors.

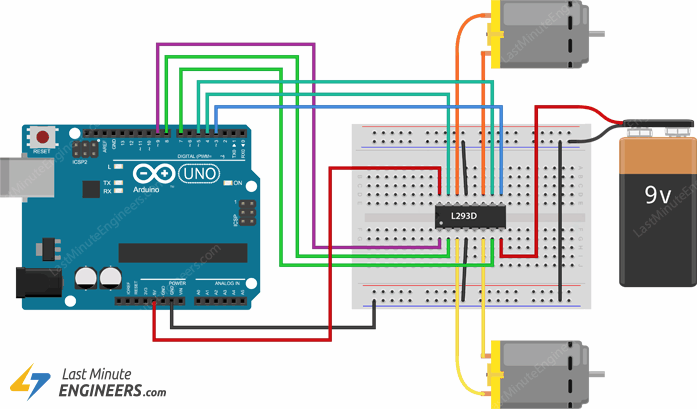
**Wiring L293D motor driver IC with Arduino UNO**

Connect 9V power supply to Vcc2 pin. Next, we need to supply 5 Volts for the L293D’s logic circuitry. Connect Vcc1 pin to 5V output on Arduino. Make sure you common all the grounds in the circuit.

Now, the input and enable pins(ENA, IN1, IN2, IN3, IN4 and ENB) of the L293D IC 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 across OUT1 & OUT2 and the other motor across OUT3 & OUT4. You can interchange your motor’s connections, technically, there is no right or wrong way.

Circuit diagram:



**Code:**

// 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);

}

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);

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);

}

// 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);

}