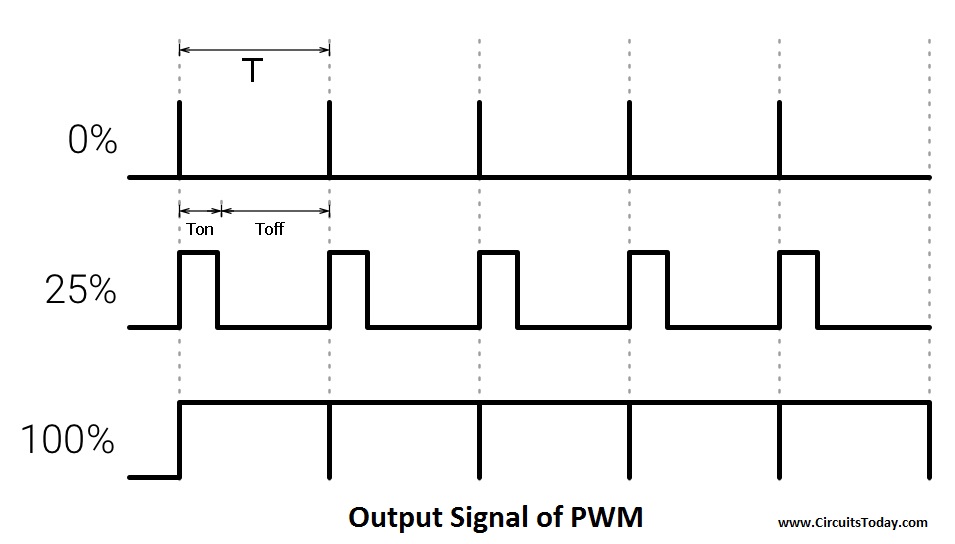
### PWM Speed Control (Pulse Width Modulation)

Microcontroller and Arduino are digital devices; they cannot give the analog output. Microcontroller gives Zero and ONE as output, where ZERO is logical LOW and ONE is logical HIGH. In our case, we are using 5 volt version of the Arduino. So it’s logical ZERO is zero voltage, and logical HIGH is 5 voltage.

Digital output is good for digital devices but sometimes we need the analog output. In such a case the PWM is very useful. In the PWM, output signal switches between zero and one, on high and fixed frequency, as shown in the figure below.

[](http://www.circuitstoday.com/wp-content/uploads/2017/11/Output-Signal-of-PWM.jpg)

**Output Signal of PWM**

As shown in the above figure the ON time is “Ton” and the OFF time is “Toff”. T is the sum of the “Ton” and “Toff” which is called the Time Period. In the concept of PWM “T” is not varying and the “Ton” and the “Toff” can vary, in this way when “Ton” increase “Toff” will decrease and “Toff” increase when “Ton” decrease proportionally.

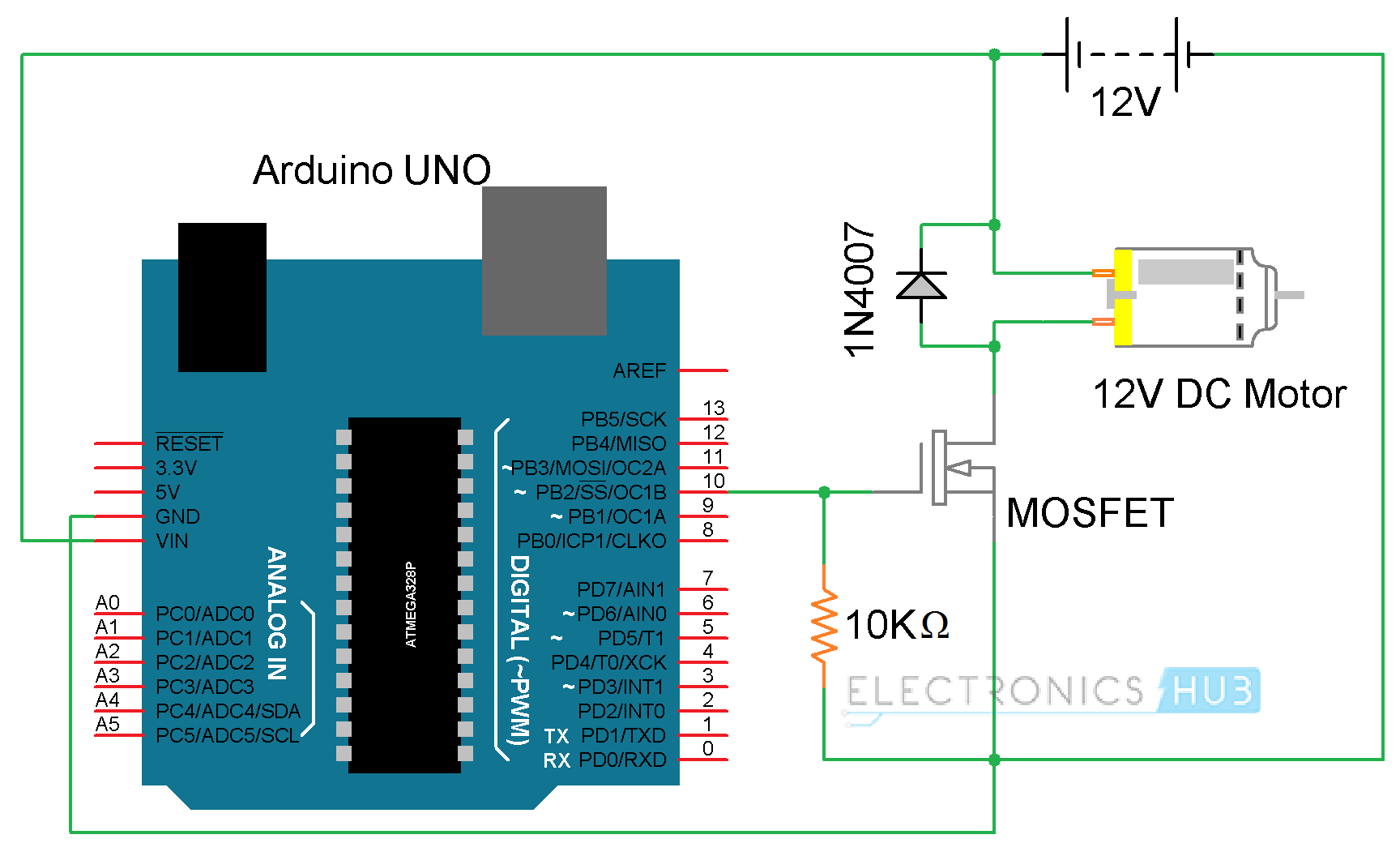
The duty cycle is the fraction of one Time period. Duty cycle is commonly expressed as a percentage or a ratio. A period is the time it takes for a signal to complete an on-and-off cycle. As a formula, a duty cycle may be expressed as:

**DUTY CYCLE = (Ton ÷ T) x100 %**

Now the motor’s speed varies according to duty cycle. Suppose the duty is zero, motor does not run and when duty cycle is 100 % the motor moves on maximum RPM. But this concept is not always right because motor starts running after giving some fixed voltage that is called threshold voltage.

**Transistor**

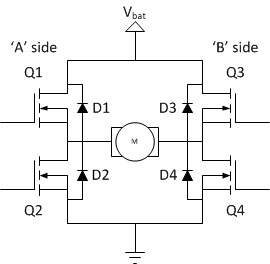
Microcontroller and the Arduino can process signals and consumes almost 20 to 40mA current but motors need high current and voltage, so we are using the transistor for driving the motor. Transistor is connected in series with motor and transistor’s base is connected to Arduino’s PWM pin through a resistance. PWM signal is coming from Arduino and the transistor works as a switch and it short circuit the Emitter (E) and Collector (C) when PWM signal is in High state and normally opens when PWM signal is in LOW state. This process works continuously and the motors runs at desired speed.



**Arduino Code:-**

|  |
| --- |
|  |
| int PWMPin = 10; |
|  | int motorSpeed = 0 |
|  |  |
|  | void setup() |
|  | { |
|  |  |
|  | } |
|  |  |
|  | void loop() |
|  | { |
|  |  |
|  | for (motorSpeed = 0 ; motorSpeed <= 255; motorSpeed += 10) |
|  | { |
|  | analogWrite(PWMPin, motorSpeed); |
|  | delay(30); |
|  | } |
|  |  |
|  | for (motorSpeed = 255 ; motorSpeed >= 0; motorSpeed -= 10) |
|  | { |
|  | analogWrite(PWMPin, motorSpeed); |
|  | delay(30); |
|  | } |
|  | } |
|  |  |

**H bridge for Motor Control**

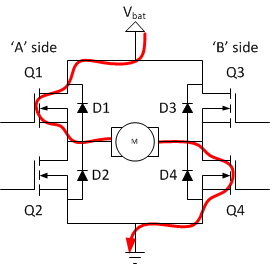


**A TYPICAL H-BRIDGE**

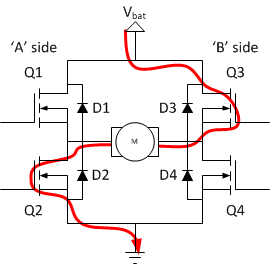
An H – Bridge is a motor driver circuit, consisting of a centrally located motor and switches to drive it in either direction. The switches are generally MOSFETS but all kinds of switches can be used. It usually comes in form of IC but can also be made using various components.

**Brief Working Of A H-Bridge:**

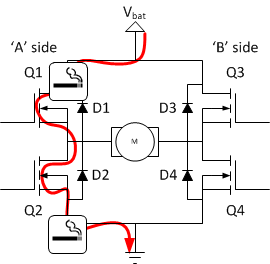
The basic operating mode of an H-bridge is fairly simple: if Q1 and Q4 are turned on, the left lead of the motor will be connected to the power supply, while the right lead is connected to ground. Current starts flowing through the motor which energizes the motor in (let’s say) the forward direction and the motor shaft starts spinning.

[](http://modularcircuits.com/blog/wp-content/uploads/2011/10/image8.png)

If Q2 and Q3 are turned on, the reverse will happen, the motor gets energized in the reverse direction, and the shaft will start spinning backwards.

[](http://modularcircuits.com/blog/wp-content/uploads/2011/10/image9.png)

In a bridge, you should never ever close both Q1 and Q2 (or Q3 and Q4) at the same time. If you did that, you just have created a really low-resistance path between power and GND, effectively short-circuiting your power supply. This condition is called ‘shoot-through’ and is an almost guaranteed way to quickly destroy your bridge, or something else in your circuit.

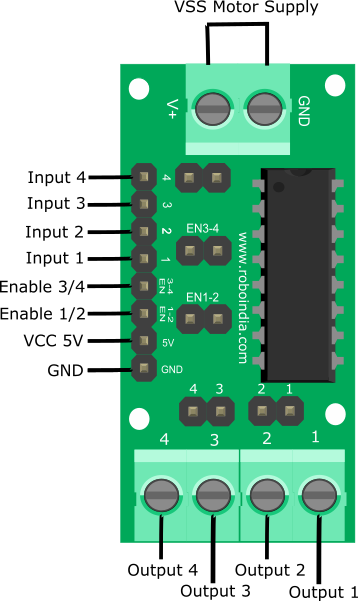
[](http://modularcircuits.com/blog/wp-content/uploads/2011/10/image10.png)

By combining both the features i.e. PWM technique for speed control and H-Bridge connection for direction control, you can have a complete control on a DC Motor.

It is tedious to use transistors for making an effective H-Bridge connection. For this purpose, there are dedicated H-Bridge Motor Driver IC available in the market and the two common IC’s are L293D and L298N.

The Motor Driver is a module for motors that allows you to control the working speed and direction of two motors simultaneously .This Motor Driver is designed and developed based on L293D IC.

L293D is a 16 Pin Motor Driver IC. This is designed to provide bidirectional drive currents at voltages from 5 V to 36 V.

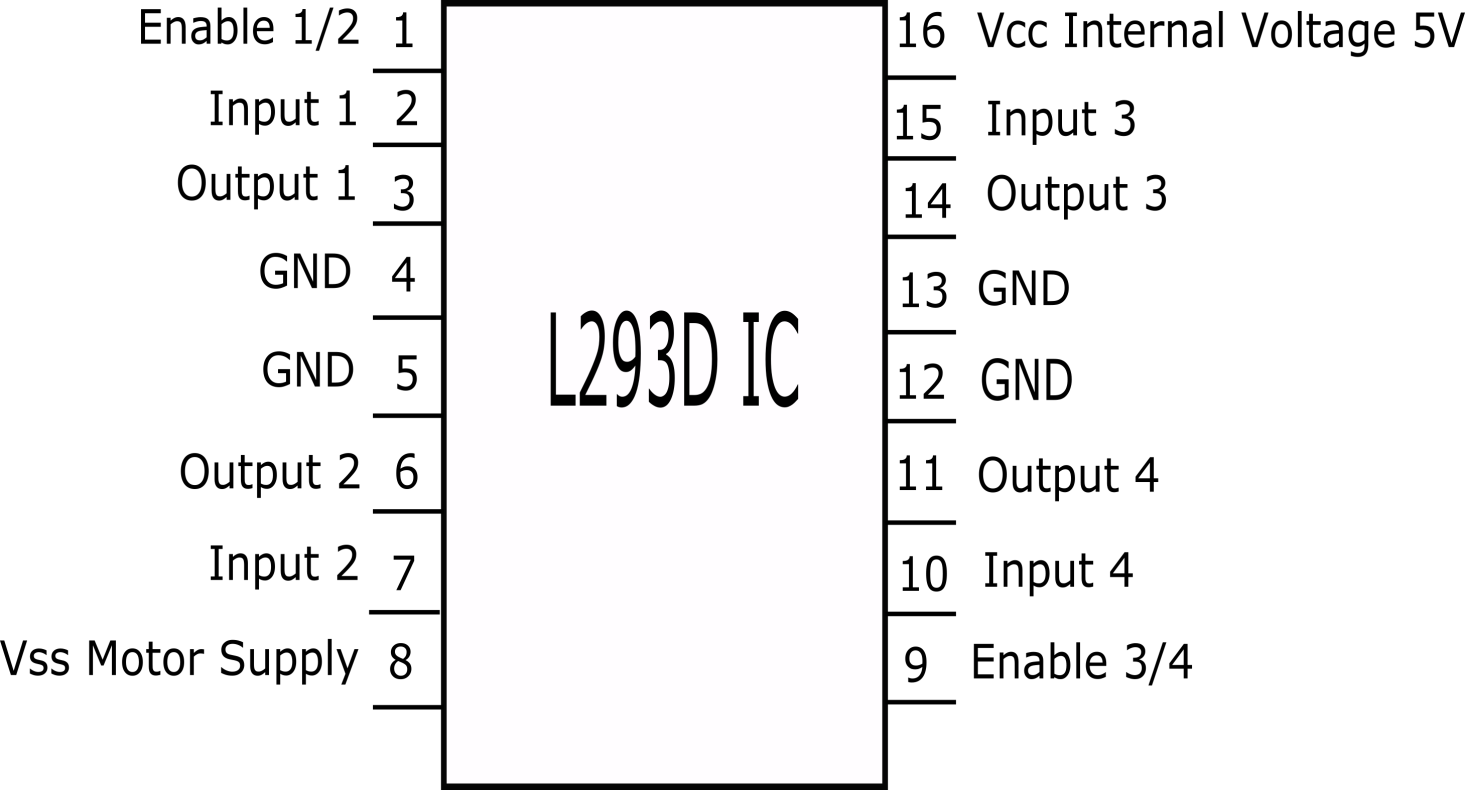


**1.2 Hardware required**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Item** | **Quantity** |
| 1 | [Arduino UNO](https://roboindia.com/store/low-cost-arduino-uno-r3?search=arduino&page=3) | 1 |
| 2 | [Motor Driver](https://roboindia.com/store/Motor-Driver-DC-Geared-Stepper) | 1 |
| 3 | [DC Motor](https://roboindia.com/store/BO-Motors-90-Degrees?search=bo%20motor) | 2 |
| 4 | [Female to Male Jumper wire](https://roboindia.com/store/male-female-connectors-jumper-wire-electrical-electronics-embedded-system-prototype?product_id=253) | 8 |
| 5 | [6xAA Battery](https://roboindia.com/store/6xaa-battery-holder-black) | 1 |

**1.3 L293D IC Pin Out**

The L293D is a 16 pin IC, with eight pins, on each side, to controlling of two DC motor simultaneously. There are 4 INPUT pins, 4 OUTPUT pins and 2 ENABLE pin for each motor.



Pin 1: When Enable1/2 is HIGH, Left part of IC will work, i.e motor connected with pin 3 and pin 6 will rotate.

Pin 2: Input 1, when this pin is HIGH the current will flow though output 1.

Pin 3: Output 1, this pin is connected with one terminal of motor.

Pin 4/5: GND pins

Pin 6: Output 2, this pin is connected with one terminal of motor.

Pin 7: Input 2, when this pin is HIGH the current will flow though output 2.

Pin 8: VSS, this pin is used to give power supply to connected motors from 5V to 36V maximum depends on Motor connected.

Pin 9: When Enable 3/4 is HIGH, Right part of IC will work, i.e motor connected with pin 11 and pin 14 will rotate.

Pin 10: Input 4, when this pin is HIGH the current will flow though output 4.

Pin 11: Output 4, this pin is connected with one terminal of motor.

Pin 12/13: GND pins

Pin 14: Output 3, this pin is connected with one terminal of motor.

Pin 15: Input 3, when this pin is HIGH the current will flow though output 3.

Pin 16: VCC, for supply power to IC i.e 5V.

**2. Connections with Arduino**

1. Module 5V (VCC) - Arduino 5V.

2. Module GND - Arduino GND.

3. Module 1 - Arduino D8.

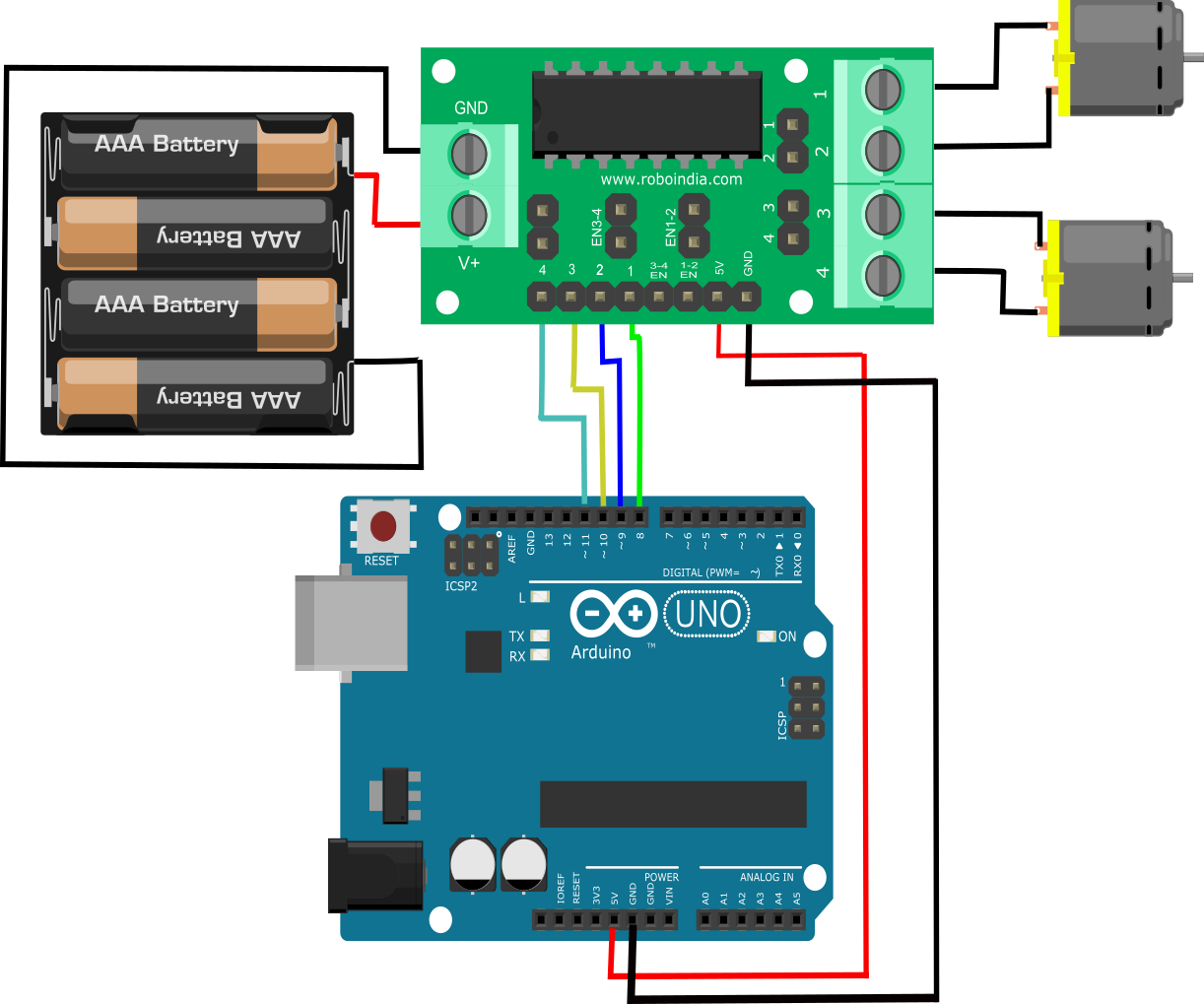
4. Module 2 - Arduino D9.

5. Module 3 - Arduino D10.

6. Module 4 - Arduino D11.

7. Module Motor terminals - DC motors.

8. Module VSS power terminal- External power source of 9V.



Make the connection as shown above. 

Make sure that the Jumpers are preset on the Enable 1-2 and Enable 3-4 pins of module, so that motor will be enabled and work at maximum speed.

**2.1. Working Mechanism**

Rotation of motor depends on Enable Pins. When Enable 1/2 is HIGH , motor connected to left part of IC will rotate according to following manner:

|  |  |  |
| --- | --- | --- |
| **Input 1** | **Input 2** | **Result** |
| 0 | 0 | Stop |
| 0 | 1 | Anti Clockwise |
| 1 | 0 | Clockwise |
| 1 | 1 | Stop |

**3. Programming:**

Here is the code to run this circuit.

[*You may download this code (Arduino Sketch) from here.*](https://roboindia.com/tutorials/admin/source32145898/shruti/Motor_Driver_arduino/L293D_IC_Arduino_.zip)

//Tutorial by RoboIndia on Motor Control

//Hardware Require: Motor Driver(By RoboIndia) & Arduino

//Motor A

const int inputPin1  = 10;    // Pin 15 of L293D IC

const int inputPin2  = 11;    // Pin 10 of L293D IC

//Motor B

const int inputPin3  = 9;   // Pin  7 of L293D IC

const int inputPin4  = 8;   // Pin  2 of L293D IC

void setup()

{

   pinMode(inputPin1, OUTPUT);

   pinMode(inputPin2, OUTPUT);

   pinMode(inputPin3, OUTPUT);

   pinMode(inputPin4, OUTPUT);

}

void loop()

{

   digitalWrite(inputPin1, HIGH);

   digitalWrite(inputPin2, LOW);

   digitalWrite(inputPin3, HIGH);

   digitalWrite(inputPin4, LOW);

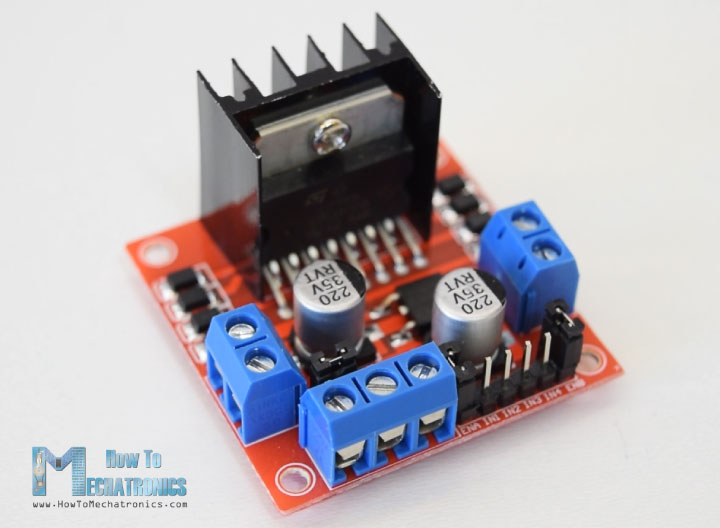
}

**4. Output**

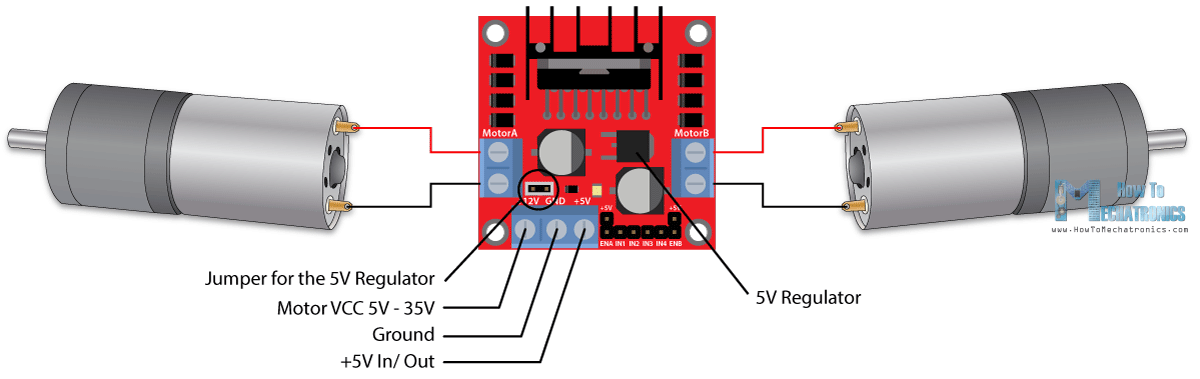
After the connection you will copy and paste this code in Arduino IDE than upload the code. Both motors will start rotating.

***L298N Driver***

The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A.

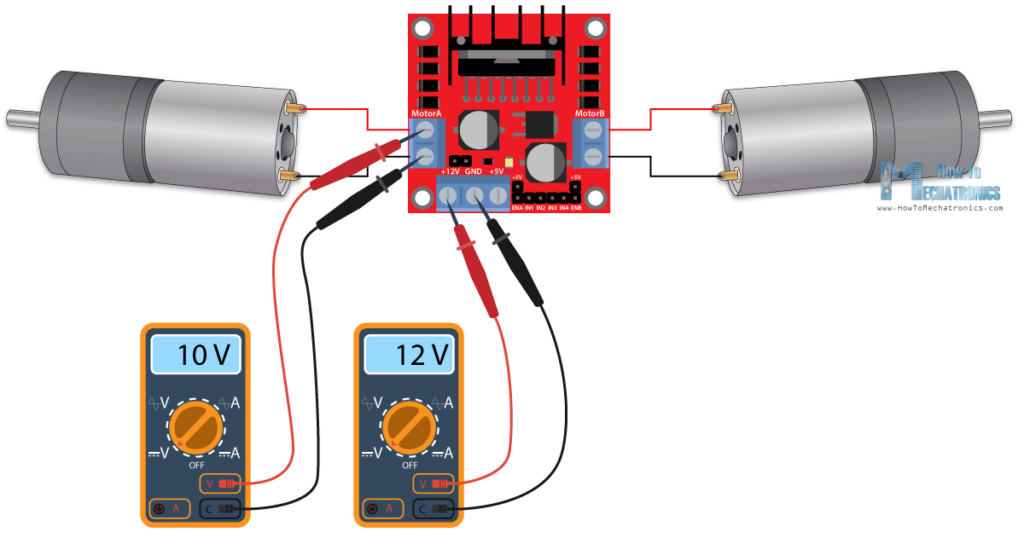


Let’s take a closer look at the pinout of L298N module and explain how it works. The module has two screw terminal blocks for the motor A and B, and another screw terminal block for the Ground pin, the VCC for motor and a 5V pin which can either be an input or output.



This depends on the voltage used at the motors VCC. The module have an onboard 5V regulator which is either enabled or disabled using a jumper. If the motor supply voltage is up to 12V we can enable the 5V regulator and the 5V pin can be used as output, for example for powering our Arduino board. But if the motor voltage is greater than 12V we must disconnect the jumper because those voltages will cause damage to the onboard 5V regulator. In this case the 5V pin will be used as input as we need connect it to a 5V power supply in order the IC to work properly.

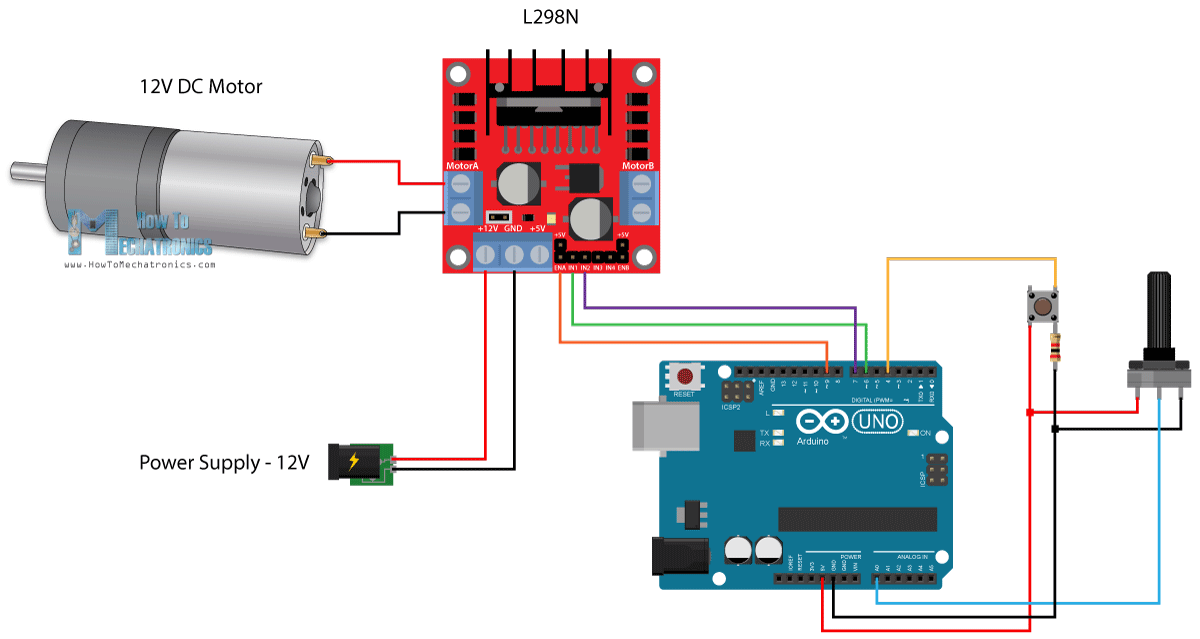
We can note here that this IC makes a voltage drop of about 2V. So for example, if we use a 12V power supply, the voltage at motors terminals will be about 10V, which means that we won’t be able to get the maximum speed out of our 12V DC motor.



Next are the logic control inputs. The Enable A and Enable B 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 and Input 2 pins are used for controlling the rotation direction of the motor A, and the inputs 3 and 4 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.

**Arduino and L298N**

[](https://howtomechatronics.com/wp-content/uploads/2017/08/Arduino-and-L298N-Circuit-Diagram-DC-Motor-Control.png?x57244)

So we need an L298N driver, a DC motor, a potentiometer, a push button and an Arduino board.

You can get the components needed for this Arduino Tutorial from the links below:

* L298N Driver
* 12V High Torque DC Motor
* Arduino Board …
* Potentiometer **Arduino Code**

Here’s the Arduino code:

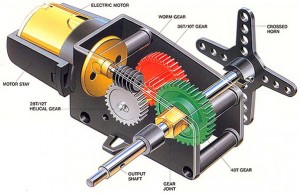
1. /\* Arduino DC Motor Control - PWM | H-Bridge | L298N - Example 01
2. by Dejan Nedelkovski, www.HowToMechatronics.com
3. \*/
4. #define enA 9
5. #define in1 6
6. #define in2 7
7. #define button 4
8. **int** rotDirection = 0;
9. **int** pressed = **false**;
10. **void** setup() {
11. pinMode(enA, OUTPUT);
12. pinMode(in1, OUTPUT);
13. pinMode(in2, OUTPUT);
14. pinMode(button, INPUT);
15. // Set initial rotation direction
16. digitalWrite(in1, LOW);
17. digitalWrite(in2, HIGH);
18. }
19. **void** loop() {
20. **int** potValue = analogRead(A0); // Read potentiometer value
21. **int** pwmOutput = map(potValue, 0, 1023, 0 , 255); // Map the potentiometer value from 0 to 255
22. analogWrite(enA, pwmOutput); // Send PWM signal to L298N Enable pin
23. // Read button - Debounce
24. if (digitalRead(button) == **true**) {
25. pressed = !pressed;
26. }
27. **while** (digitalRead(button) == **true**);
28. delay(20);
29. // If button is pressed - change rotation direction
30. **if** (pressed == **true** & rotDirection == 0) {
31. digitalWrite(in1, HIGH);
32. digitalWrite(in2, LOW);
33. rotDirection = 1;
34. delay(20);
35. }
36. // If button is pressed - change rotation direction
37. if (pressed == **false** & rotDirection == 1) {
38. digitalWrite(in1, LOW);
39. digitalWrite(in2, HIGH);
40. rotDirection = 0;
41. delay(20);
42. }
43. }

**Description:** So first we need to define the pins and some variables needed for the program. In the setup section we need to set the pin modes and the initial rotation direction of the motor. In the loop section we start by reading the potentiometer value and then map the value that we get from it which is from 0 to 1023, to a value from 0 to 255 for the PWM signal, or that’s 0 to 100% duty cycle of the PWM signal. Then using the analogWrite() function we send the PWM signal to the Enable pin of the L298N board, which actually drives the motor.

Next, we check whether we have pressed the button, and if that’s true, we will change the rotation direction of the motor by setting the Input 1 and Input 2 states inversely. The push button will work as toggle button and each time we press it, it will change the rotation direction of the motor.

**OPERATION OF SERVO AND STEPPER MOTORS**

SERVO MOTORS

The servo motor is most commonly used for high technology devices in the industrial application like automation technology. It is a self contained electrical device, that rotate parts of a machine with high efficiency and great precision. The output shaft of this motor can be moved to a particular angle. Servo motors are mainly used in home electronics, toys, cars, airplanes, etc. 

DC Servo Motor

The motor which is used as a DC servo motor generally have a separate DC source in the field of winding & armature winding. The control can be archived either by controlling the armature current or field current. Field control includes some particular advantages over armature control. In the same way armature control includes some advantages over field control. Based on the applications the control should be applied to the DC servo motor. DC servo motor provides very accurate and also fast respond to start or stop command signals due to the low armature inductive reactance. DC servo motors are used in similar equipments and computerized numerically controlled machines.



AC Servo Motor

AC servo motor is an AC motor that includes encoder is used with controllers for giving closed loop control and feedback. This motor can be placed to high accuracy and also controlled precisely as compulsory for the applications. Frequently these motors have higher designs of tolerance or better bearings and some simple designs also use higher voltages in order to accomplish greater torque. Applications of an AC motormainly involve in automation, robotics, CNC machinery, and other applications a high level of precision and needful versatility.



**Positional Rotation Servo Motor**

Positional rotation servo motor is a most common type of servo motor. The shaft’s o/p rotates in about 180o. It includes physical stops located in the gear mechanism to stop turning outside these limits to guard the rotation sensor. These common servos involve in radio controlled water, radio controlled cars, aircraft, robots, toys and many other applications.

**Continuous Rotation Servo Motor**

Continuous rotation servo motor is quite related to the common positional rotation servo motor, but it can go in any direction indefinitely. The control signal, rather than set the static position of the servo, is understood as the speed and direction of rotation. The range of potential commands sources the servo to rotate clockwise or anticlockwise as preferred, at changing speed, depending on the command signal. This type of motor is used in a radar dish if you are riding one on a robot or you can use one as a drive motor on a mobile robot.



**Linear Servo Motor**

Linear servo motor is also similar the positional rotation servo motor is discussed above, but with an extra gears to alter the o/p from circular to back-and-forth. These servo motors are not simple to find, but sometimes you can find them at hobby stores where they are used as actuators in higher model airplanes.



WORKING

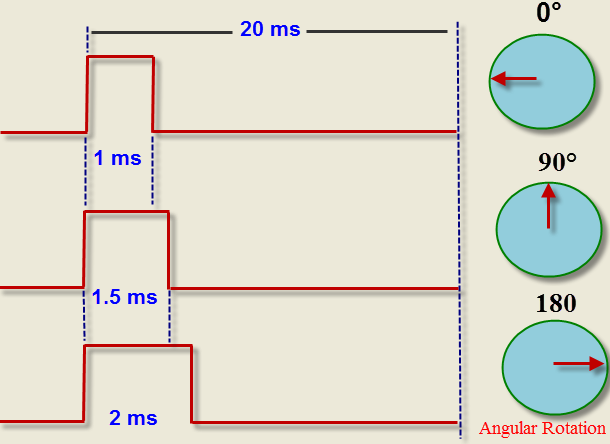
A unique design for servo motors are proposed in controlling the robotics and for control applications. They are basically used to adjust the speed control at high torques and accurate positioning. Parts required are motor position sensor and a highly developed controller. These motors can be categorized according the servo motor controlled by servomechanism. If DC motor is controlled using this mechanism, then it is named as a DC servo motor. Servo motors are available in power ratings from fraction of a watt to 100 watts.The rotor of a servo motor is designed longer in length and smaller in diameter so that it has low inertia. To know more about this, please follow the link: Servo motor working principle and interfacing with 8051 microcontroller

Applications

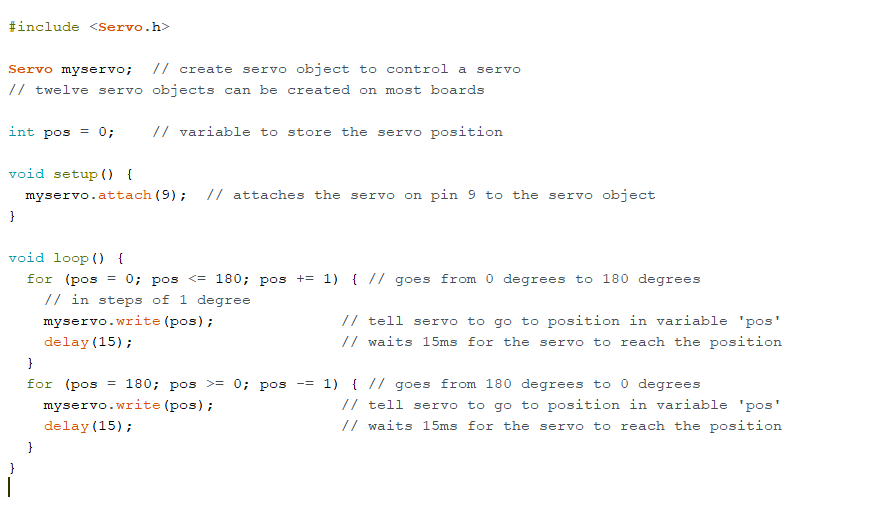
* The servo motor is used in robotics to activate movements, giving the arm to its precise angle.
* The Servo motor is used to start, move and stop conveyor belts carrying the product along with many stages. For instance, product labeling, bottling and packaging
* The servo motor is built into the camera to correct a lens of the camera to improve out of focus images.

**Controlling a Servo Motor with Angle rotations**

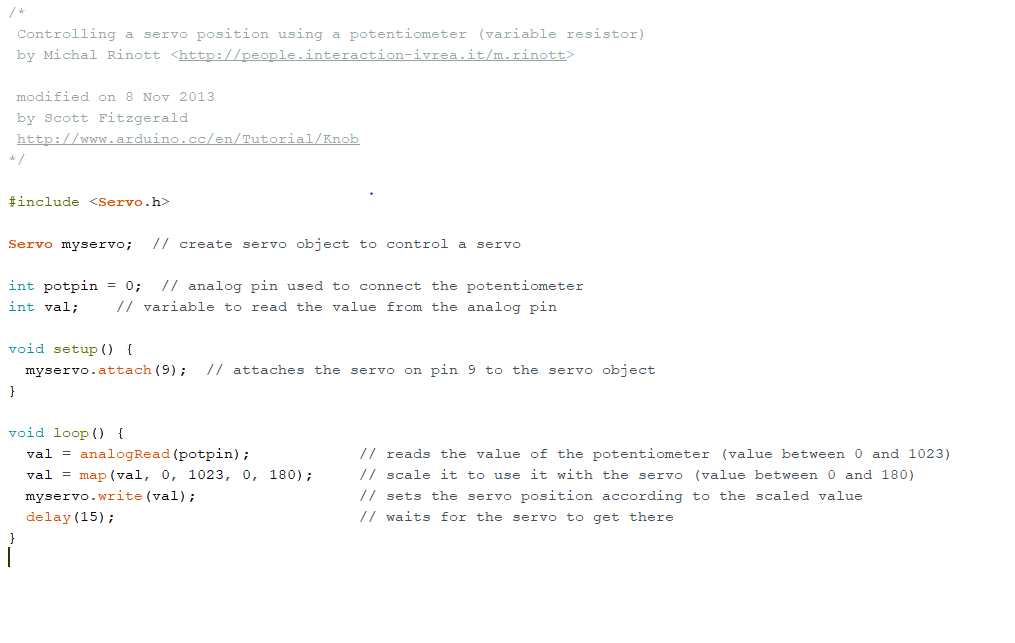
Servo motor working principle mainly depends upon duty cycles. It uses Pulse Width Modulated (PWM) waves as control signals. The angle of rotation is resolute by the pulse width of  the control pin. Here the servo motor used for angle of rotation from 0 to 180 degrees. We can control the precise angular position by varying the pulse among 1ms to 2ms.

* [](https://www.elprocus.com/wp-content/uploads/2014/11/rotation.png)

Servo Motor Programming With angular rotations

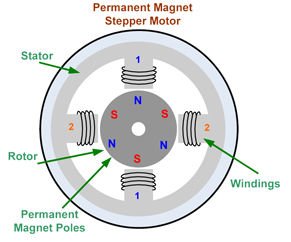


Controlling Servo Using A Potentiometer



**STEPPER MOTOR**

[](https://cdn.instructables.com/FUK/54L4/IB22QT6A/FUK54L4IB22QT6A.LARGE.jpg)

[](https://cdn.instructables.com/FH2/O7RU/IB22QR8X/FH2O7RUIB22QR8X.LARGE.jpg)

A Stepper Motor or a step motor is a brushless, synchronous motor, which divides a full rotation into a number of steps. Unlike a brushless DC motor, which rotates continuously when a fixed DC voltage is applied to it, a step motor rotates in discrete step angles.

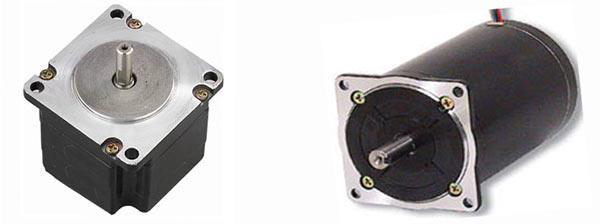
The Stepper Motors therefore are manufactured with steps per revolution of 12, 24, 72, 144, 180, and 200, resulting in stepping angles of 30, 15, 5, 2.5, 2, and 1.8 degrees per step. The stepper motor can be controlled with or without feedback.

A stepper motor consists of two main parts, a rotor and a stator. The rotor is the part of the motor that actually spins and provides work. The stator is the stationary part of the motor that houses the rotor. In a stepper motor, the rotor is a permanent magnet. The stator consists of multiple coils that act as electromagnets when an electrical current is passed through them. The electromagnetic coil will cause the rotor to align with it when charged. The rotor is propelled by alternating which coil has a current running through it.

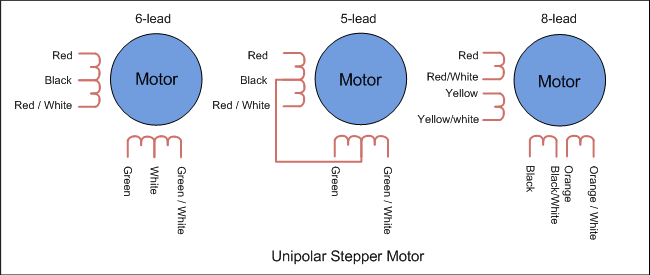
Imagine a motor on an RC airplane. The motor spins very fast in one direction or another. You can vary the speed with the amount of power given to the motor, but you cannot tell the propeller to stop at a specific position.

Now imagine a printer. There are lots of moving parts inside a printer, including motors. One such motor acts as the paper feed, spinning rollers that move the piece of paper as ink is being printed on it. This motor needs to be able to move the paper an exact distance to be able to print the next line of text or the next line of an image.

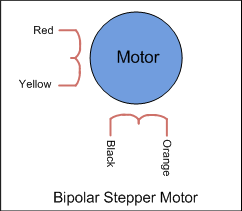
There is another motor attached to a threaded rod that moves the print head back and forth. Again, that threaded rod needs to be moved an exact amount to print one letter after another. This is where the stepper motors come in handy.



**Unipolar Stepper Motors**  
The unipolar stepper motor operates with one winding with a center tap per phase. Each section of the winding is switched on for each direction of the magnetic field. Each winding is made relatively simple with the commutation circuit, this is done since the arrangement has a magnetic pole which can be reversed without switching the direction of the current.



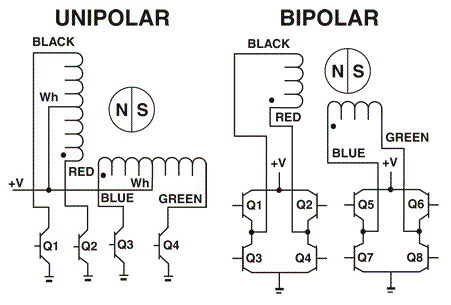
In most cases, given a phase, the common center tap for each winding is the following; three leads per phase and six leads for a regular two phase stepper motor. You will usually see that both these phases are often joined internally, this makes the stepper motor only have five leads. Often a stepper motor controller will be used to activate the drive transistors in the proper order

**Bipolar Stepper Motors**  
With bipolar stepper motors there is only a single winding per phase. The driving circuit needs to be more complicated to reverse the magnetic pole, this is done to reverse the 

current in the winding. This is done with a H-bridge arrangement, however there are several driver chips that can be purchased to make this a more simple task.

Unlike the unipolar stepper motor, the bipolar stepper motor has two leads per phase, neither of which are common. Static friction effects do happen with a H-bridge with certain drive topologies, however this can be reduced with dithering the stepper motor signal at a higher frequency.

Bipolar stepper motors can be a bit more difficult to operate, and the unipolar motor does feature twice the amount of wire in the same space. Different projects will require different types and settings of stepper motors.

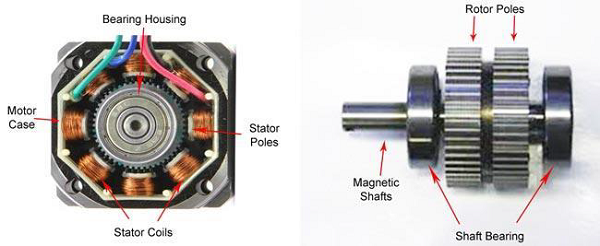


How a Stepper Motor Works?

A regular DC motor spins in only direction whereas a Stepper motor can spin in precise increments.

Stepper motors can turn an exact amount of degrees (or steps) as desired. This gives you total control over the motor, allowing you to move it to an exact location and hold that position. It does so by powering the coils inside the motor for very short periods of time. The disadvantage is that you have to power the motor all the time to keep it in the position that you desire.

All you need to know for now is that, to move a stepper motor, you tell it to move a certain number of steps in one direction or the other, and tell it the speed at which to step in that direction. There are numerous varieties of stepper motors. The methods described here can be used to infer how to use other motors and drivers which are not mentioned in this tutorial. However, it is always recommended that you consult the datasheets and guides of the motors and drivers specific to the models you have.



Components Required

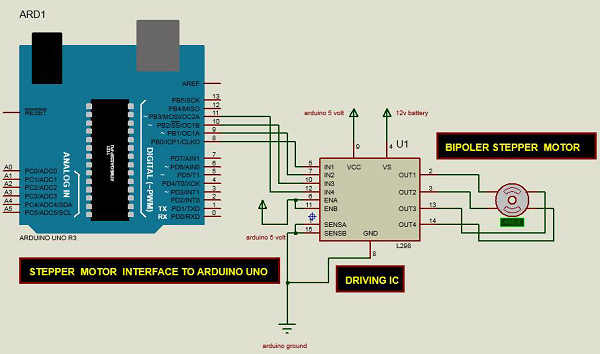
You will need the following components −

* 1 × Arduino UNO board
* 1 × small bipolar stepper Motor as shown in the image given below
* 1 × LM298 driving IC



Procedure

Follow the circuit diagram and make the connections as shown in the image given below.



Arduino Code

/\* Stepper Motor Control \*/

#include <Stepper.h>

const int stepsPerRevolution = 90;

// change this to fit the number of steps per revolution

// for your motor

// initialize the stepper library on pins 8 through 11:

Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {

// set the speed at 60 rpm:

myStepper.setSpeed(5);

// initialize the serial port:

Serial.begin(9600);

}

void loop() {

// step one revolution in one direction:

Serial.println("clockwise");

myStepper.step(stepsPerRevolution);

delay(500);

// step one revolution in the other direction:

Serial.println("counterclockwise");

myStepper.step(-stepsPerRevolution);

delay(500);

}

Code to Note

This program drives a unipolar or bipolar stepper motor. The motor is attached to digital pins 8 - 11 of Arduino.

Result

The motor will take one revolution in one direction, then one revolution in the other direction.

**ENCODERS**

An encoder is a sensor of mechanical motion that generates digital signals in response to motion. As an electro-mechanical device, an encoder is able to position, velocity and direction. There are two different types of encoders: linear and rotary.

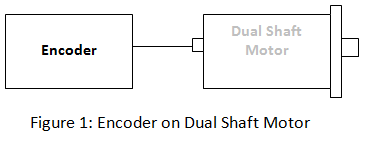
A linear encoder responds to motion along a path, while a rotary encoder responds to rotational motion.

An encoder is generally categorized by the means of its output.

An incremental encoder generates a train of pulses which can be used to determine position and speed.

An absolute encoder generates unique bit configurations to track positions directly.

**Block Diagram for Encoders**



**Basic Types of Encoders**

Linear and rotary encoders are broken down into two main types: the absolute encoder and the incremental encoder. The construction of these two types of encoders is quite similar; however they differ in physical properties and the interpretation of movement.

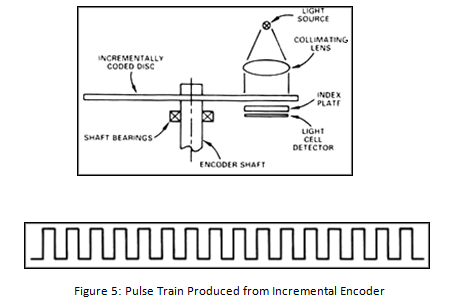
**Incremental Encoder**



Single-Ended Encoder

An Incremental rotary encoder is also referred to as a quadrature encoder. This type of encoder utilizes sensors that use optical, mechanical or magnetic index counting for angular measurement.

**How do Incremental Encoders Work?**   
  
Incremental rotary encoders utilize a transparent disk which contains opaque sections that are equally spaced to determine movement. A light emitting diode is used to pass through the glass disk and is detected by a photo detector. This causes the encoder to generate a train of equally spaced pulses as it rotates. The output of incremental rotary encoders is measured in pulses per revolution which is used to keep track of position or determine speed.

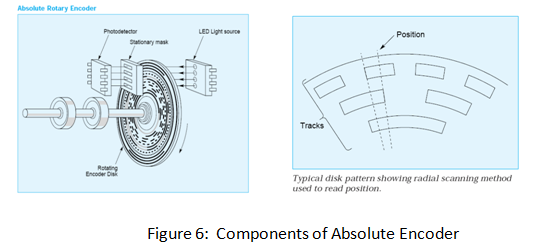


A single-channel output is commonly implemented in applications in which direction of movement is not significant. Instances in which direction sensing is important, a 2-channel, quadrature, output is used. The two channels, A and B, are commonly 90 electrical degrees out of phase and the electronic components determine the direction based off the phase relationship between the two channels. The position of an incremental encoder is done by adding up all the pulses by a counter.   
  
A setback of the incremental encoder is count loss which occurs during power loss. When restarting, the equipment must be referenced to a home position to reinitialize the counter. However, there are some incremental encoders, like those sold at Anaheim Automation, which come equipped with a third channel called the index channel. The index channel produces a single signal pulse per revolution of the encoder shaft and is often used as a reference marker. The reference marker is then denoted as a starting position which can resume counting or position tracking.   
  
**NOTE:** Incremental rotary encoders are not as accurate as absolute rotary encoders due to the possibility of interference or a misread.

**Absolute Encoder**

An absolute encoder contains components also found in incremental encoders. They implement a photodetector and LED light source but instead of a disk with evenly spaced lines on a disc, an absolute encoder uses a disk with concentric circle patterns.

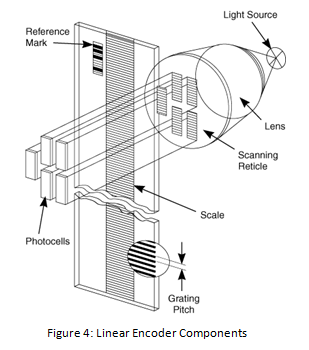
**How do Absolute Encoders Work?**   
  
Absolute encoders utilize stationary mask in between the photodetector and the encoder disk as shown below. The output signal generated from an absolute encoder is in digital bits which correspond to a unique position. The bit configuration is produced by the light which is received by the photodetector when the disk rotates. The light configuration received is translated into gray code. As a result, each position has its own unique bit configuration.



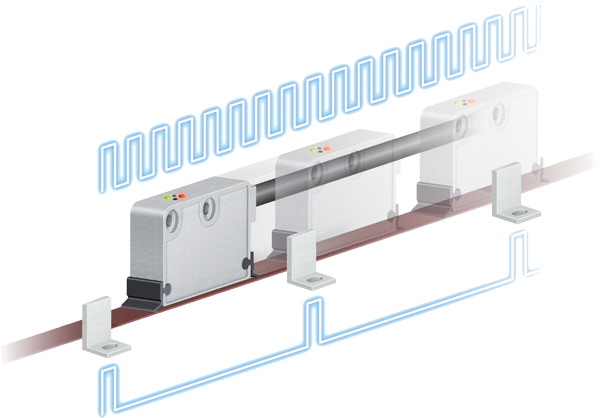
**Linear Encoder**

A linear encoder is a sensor, transducer or reading-head linked to a scale that encodes position. The sensor reads the scale and converts position into an analog or digital signal that is transformed into a digital readout. Movement is determined from changes in position with time. Both optical and magnetic linear encoder types function using this type of method. However, it is their physical properties which make them different.

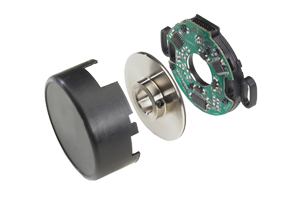
**How do Optical Linear Encoders Work?**   
  
The light source and lens produce a parallel beam of light which pass through four windows of the scanning reticle. The four scanning windows are shifted 90 degrees apart. The light then passes through the glass scale and is detected by photosensors. The scale then transforms the detected light beam when the scanning unit moves. The detection of the light by the photosensor produces sinusoidal wave outputs. The linear encoder system then combines the shifted signals to create two sinusoidal outputs which are symmetrical but 90 degrees out of phase from each other. A reference signal is created when a fifth pattern on the scanning reticle becomes aligned with an identical pattern on the scale.



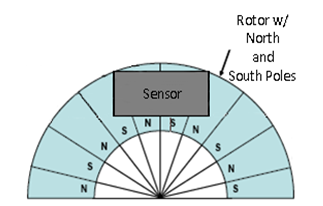
**How does a Linear Encoder Work?**   
  
A Linear Encoder system uses a magnetic sensor readhead and a magnetic scale to produce TTL or analog output for Channel A and B. As the magnetic sensor passes along the magnetic scale, the sensor detects the change in magnetic field and outputs a signal. This output signal frequency is proportional to the measuring speed and the displacement of the sensor. Since a linear encoder detects change in the magnetic field, the interference of light, oil, dust, and debris have no effect on this type of system; therefore they offer high reliability in harsh environments.



[**Magnetic Rotary Encoder**](http://www.anaheimautomation.com/products/encoder/optical-incremental-rotary.php?tID=1063&pt=t&cID=422)



A magnetic encoder consists of two parts: a rotor and a sensor. The rotor turns with the shaft and contains alternating evenly spaced north and south poles around its circumference. The sensor detects these small shifts in the position N>>S and S>>N. There many methods of detecting magnetic field changes, but the two primary types used in encoders are: Hall Effect and Magneto resistive. Hall Effect sensors work by detecting a change in voltage by magnetic deflection of electrons. Magneto resistive sensors detect a change in resistance caused by a magnetic field.



**Commutation Encoders**

A commutation encoder contains the same fundamental components as incremental encoders but with the addition of commutation tracks alongside the outer edge of the disk for U/V/W output.

**How do Commutation Encoders Work?**

**Commutation encoders** utilize a transparent disk which includes opaque sections that are equally spaced to determine movement. A light emitting diode is used to pass through the glass disk and is detected by a photo detector. This causes the encoder to generate a train of equally spaced pulses as it rotates. The output of incremental rotary encoders is measured in pulses per revolution which is used to keep track of position or determine speed.   
  
The outer part of the encoder disk includes commutation tracks which provide a controller with information on the exact position of the motor poles, so that the proper controller input can be supplied to the motor. The commutation tracks of the encoder read the motor position and instruct the controller as to how to provide efficient and proper current to the motor to cause rotation. Commutation output for U/V/W can be in the form of differential output or open-collector (manufacturer dependent).

**How are Encoders Controlled?**

Encoders are controlled through the rotation the shaft it is mounted to. The shaft comes into contact with a hub which is in internal to the encoder. As the shaft rotates, it causes the disc, with both transparent and solid lines, to rotate across the circuitry of the encoder. The circuitry of the encoder contains an LED which is captured by a photoelectric diode and outputs pulses to the user. The speed at which the disc rotates will be dependent on the speed of the shaft the encoder is connected to.

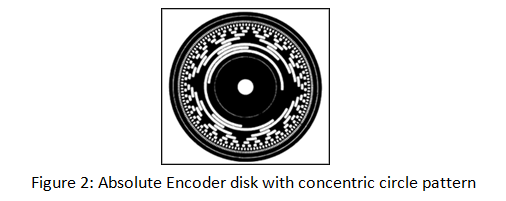
**Physical Properties**

**Linear Encoders**

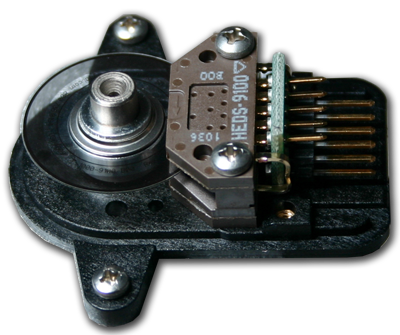
The key components of a linear encoder are a scanning unit, sensor, transducer or readhead, paired with a transmissive or reflective scale, which encodes position. The scale of a linear encoder is generally made of glass and mounted to a support and the scanning unit contains a light source, photocells, and a second glass piece called the scanning reticle. Collectively, the linear encoder is able to convert motion into digital or analog signals to determine the change in position over time.

[**Rotary Encoders**](http://www.anaheimautomation.com/products/encoder/rotary-encoders-list.php?cID=422)

The key components of a **rotary encoder** are the disk, light sources and detectors, and electronics. The disk contains a unique pattern of concentric etched circles and alternates between opaque and transparent segments. This pattern provides unique bit configurations and is used to assign specific positions. For every concentric ring in a rotary encoder, there is a light source and light detector which identify lines etched on the disk. The electronics consist of an output device which takes the signal obtained from the sensor (light/detector source) to provide feedback of position and/or velocity. All of these components are enclosed in a single housing unit.

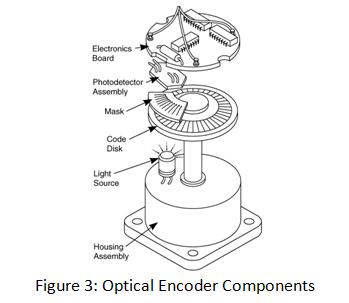


[**Incremental Encoders**](http://www.anaheimautomation.com/products/encoder/incremental-encoders-list.php?cID=422&cdID=363)



Differential-Type Encoder

The key components of an **incremental encoder** are a glass disk, LED (light emitting diode), and a photo detector. The transparent disk contains opaque sections which are equally spaced to deflect light while the transparent sections allow light to be passed through shown in Figure 2 below. An optical encoder utilizes a light emitting diode which shines light through the transparent portions of the disk. The light that shines through is received by the photo detector which produces an electrical signal output.



**Where are Encoders Used?**

Encoders have become a vital source for many applications requiring feedback information. Whether an application is concerned with speed, direction or distance, an encoders vast capability allow users to utilize this information for precise control. With the emergence of higher resolutions, ruggedness, and lower costs, encoders have become the preferred technology in more and more areas. Today, encoder applications are all around us. They are utilized in printers, automation, medical scanners, and scientific equipment. 

**How to Install an Encoder**

After selecting the appropriate motor, it is important to know how to properly install it. The installation of each encoder is dependent upon its mounting or base option. If an encoder is to be mounted to a motor shaft, then a centering tool can be used to align the hole of the encoder to the shaft. The different mounting options have varying functionalities. An R-option allows for a +/- 15 degree play of motion in which the encoder can rotate back and forth. A T-Option however, uses adhesive to stick to the back of a motor. 

**Advantages of an Encoder**

- Highly reliable and accurate  
- Low-cost feedback  
- High resolution  
- Integrated electronics  
- Fuses optical and digital technology  
- Can be incorporated into existing applications  
- Compact size

**Disadvantages of an Encoder**

- Subject to magnetic or radio interference (Magnetic Encoders)  
- Direct light source interference (Optical Encoders)  
- Susceptible to dirt, oil and dust contaminates

[**Troubleshooting**](http://www.anaheimautomation.com/support/links/troubleshooting-index.php)  
**Problem:** No output  
**Solution:** No output may be a result of various factors. Steps can be taken to ensure the proper functionality of the encoder. No mechanical movement results in any signal being output from the encoder. To correct this issue, observe if the encoder is rotating. Verify all wring between the encoder and the driver/controller is correct and the appropriate voltage supply is used. Having loose connections or improper voltage supply may not allow the encoder to function properly. Finally, ensure the correct signal type (e.g. open collector, pull-up, line driver or push-pull) is being used for your application. If the problem persists, swap encoders, if possible, to determine if the encoder is the issue.   
  
**Problem:** Unable to find index pulse  
**Solution:** The index pulse, or reference marker, is a once per revolution output of an encoder and is best found using an oscilloscope. Verify all the wiring between the encoder and the driver/controller is correct and the appropriate voltage supply is used. If that does not solve the issue, try lowering the RPM of the motor, as the driver/controller may not be able to identify the index pulse at very high RPM values.   
  
**Problem:** Count output indicates incorrect direction  
**Solution:** If the count output indicates an incorrect direction then check for the wire configuration. See if any wires are reversed. If they are reversed, simply swap wires.   
  
**NOTE:** If your application is using index, reversing the wire configuration causes the reference alignment to also change. If so, please make the appropriate changes to your application.   
  
**Problem:** Encoder is not rotating  
**Solution:** When encoders are exposed in open environments, dust and debris particles may accumulate around the shaft. Simply clean the exposed area and ensure that there are not objects obstructing the encoder from rotating.   
  
**Problem:** Noise Interference  
**Solution:** To improve the noise immunity of encoders it is strongly advised that no other electrical equipment be nearby or be kept at a fair distance. Encoder cables should also be shielded and proper wires should be grounded to minimize electrical noise.   
  
**Problem:** Distorted or incorrect output  
**Solution:** Distorted or incorrect output can be any combination of loose wiring connections, encoder output not compatible with driver/controller, electrical noise or improper alignment. Check for wire connections, compatibility issues with the encoder and the driver/controller, alignment of the encoder and the shaft to solve this issue.

[**Formulas**](http://www.anaheimautomation.com/support/links/formulas-index.php)

The relationship between the encoder CPR frequency and the speed of the motor (RPM) is given by the following equation:   
  
f = (cycles/rev)\*(rev/sec)/1000 = kHz   
  
RPM = Revolution per Minute  
CPR = Cycles per Revolution   
  
Distance Conversion:   
  
(PPR) / (2\*pi\*radius of shaft) = pulses per inch  
(Pulses per inch)^-1 = inch per pulse