Brandon · Aug 17 · 5 min read

Controlling a DC Brushed Motor with the L298N Motor Controller

I've recently noticed the same inexpensive gear motor being sold by many different sellers on Amazon and on individual store fronts.

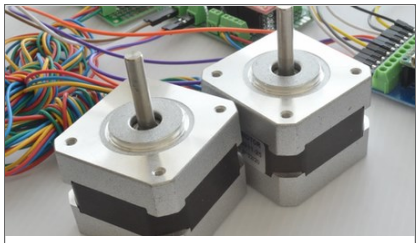
Guides • Introduction to Arduino

In this example, I demonstrate how to control a fairly common 6V gear motor with the L298N Motor Controller.

33 views 0 comments

Recent Posts

See All

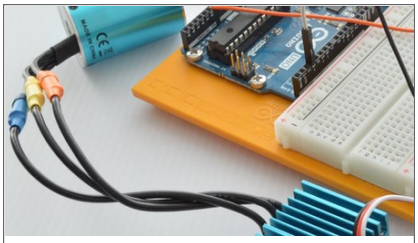


Simultaneously Controlling Stepper Motors ... Arduino

648 0

DIY Arduino Camera Phone Slider

95 0 2



Controlling a Brushless Motor with Arduino

710 0 3

Parts Needed:

In this example, I used the following parts:

Comments

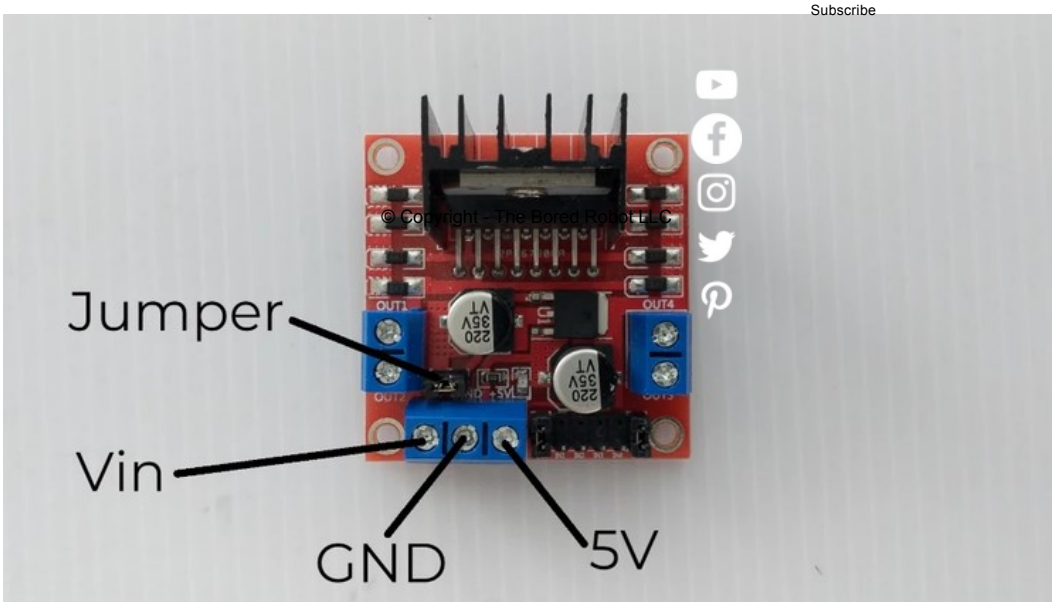
- Arduino Uno

- Breadboard
- Male to male jumper wire
- Brushed DC Motor Kit
  - Male to female jumper wire
  - DC brushed gear motor
  - Dual L298N motor controller
- Potentiometer
- 6V and 12V AA battery holder

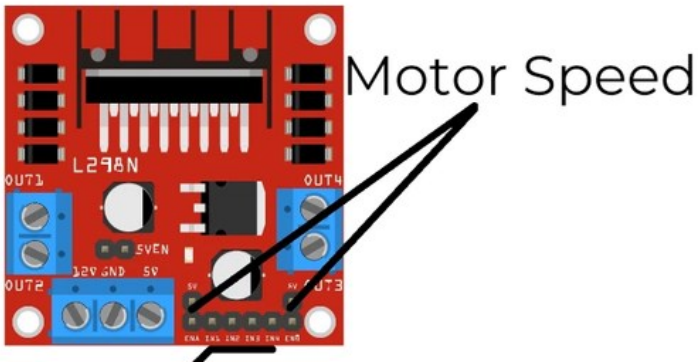
The L298N Motor Controller

L298N actually refers to a specific chip that is ultimately responsible for driving the motors. While it is possible to run a motor with only the chip, it's often mounted to a printed circuit board and heat sink. The board includes header pins and terminal blocks to make the power and communication connections more convenient.

The board used in this example has a combination of screw terminal blocks and male header pins. The two sets of screw terminals are on opposite sides of the board, and connected to the leads of the motors. The screw terminal blocks on the bottom of the board are related to power. The left connection is for the input voltage; this board can be powered by up to 12V. The center screw terminal connection is for ground and the right connection provides a 5V output voltage jumper will disable the 5V supply.



The set of header pins, on the bottom of the board, control the speed and direction of the motors. The ENA, IN1, and IN2 pins are directly related to the speed and direction of the left motor outputs. The ENB, IN3, and IN4 pins control the speed and direction of the right motor outputs. For the left motor, sending a signal between 0-5V to the ENA will control the speed of the motor. A value of 5V will cause the motor to move at full speed and a value of 0V will stop the motor. Some boards come with a jumper connected to the ENA and ENB pins. This connects 5V to these pins so the motors will always move at full speed. Remove these jumpers to control the speeds of the motors. The IN1 and IN2 pins should be set to HIGH or LOW. For example, if IN1 receives a HIGH signal, then IN2 must receive a LOW signal. Reversing this changes the direction of the motor. These methods also apply to the other motor output with the ENB, IN3, and IN4 pins.

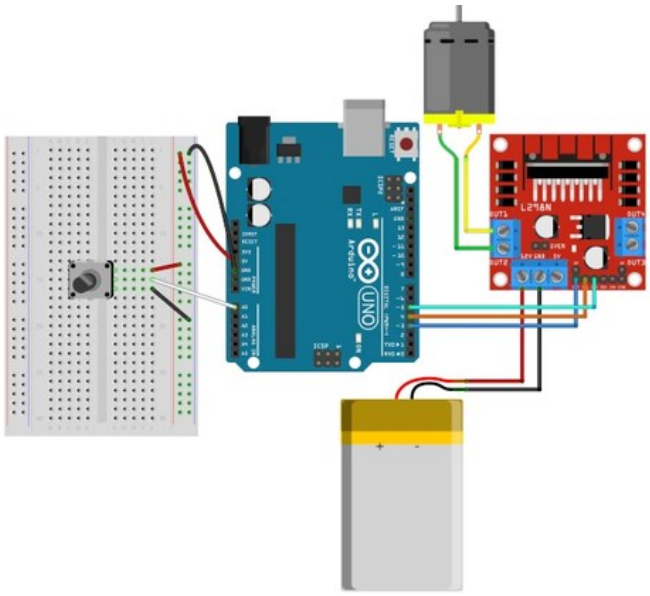


Motor Direction

Building the Circuit

For this example, I used a potentiometer with an Arduino Uno to control the speed of the motor. To build this circuit:

- connect 5V and GND from the Arduino to the breadboard
- place the potentiometer over the center of the breadboard.
- connect the potentiometer to 5V, GND, and A0
- connect pin 3 on the Arduino to ENA
- connect pin 4 to IN1
- connect pin 5 to IN2
- connect the motor leads to the motor output screw terminals on the left side of the board
- Connect the power source to the board with the switch turned off. To start, I will use a 6V AA battery holder (4 batteries). Later in the example, I will use a 12V AA battery holder (8 batteries).



Arduino Code

I want to set up the Arduino code so that rotating the potentiometer directly controls the speed and direction of a single motor. I also want it so that the midpoint of the potentiometer will stop the motor, moving the potentiometer. Let's first start the code by defining the pins. I will use pin A0 to read the potentiometer value, and pins 3-5 for the ENA, IN1, and IN2 pins respectively. I also define integer variables for the value that is being read from the potentiometer and for the motor speed.

```
#define PotPin A0
#define ENA1Pin 3
#define IN1Pin 4
#define IN2Pin 5

int PotVal;
int MotorSpeed;
```

In the setup, pins 3-5 should be set to outputs and the baud rate can be set to 9600.

```
void setup() {
  pinMode(ENA1Pin, OUTPUT);
  pinMode(IN1Pin, OUTPUT);
  pinMode(IN2Pin, OUTPUT);
  Serial.begin(9600);
}
```

In the loop, use the "analogRead" command to read the value from the potentiometer and save it to a

variable. This value should be in between 0 and 1023, so the midpoint could be rounded to 512.

```
PotVal = analogRead(PotPin);
```

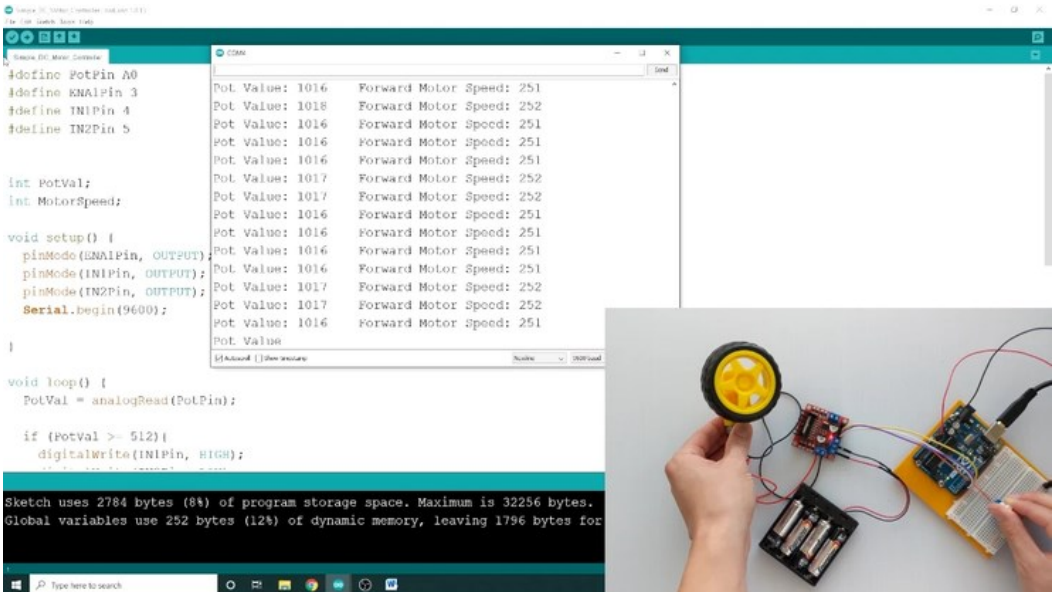
I can use an "if/else" statement to control the speed and direction of the motor, based on what's being read from the potentiometer. If the potentiometer value is greater than or equal to 512, then I'll used the "digitalWrite" command to set IN1 to HIGH and IN2 to LOW. Since I plan on using the "analogWrite" command to control the speed of the motor, I need to map the potentiometer value. This value should be mapped from 512-1023 to 0-255. I also use print statements to help with debugging in the future.

```
if (PotVal >= 512){
  digitalWrite(IN1Pin, HIGH);
  digitalWrite(IN2Pin, LOW);
  MotorSpeed = map(PotVal, 512, 1023, 0, 255);
  analogWrite(ENA1Pin, MotorSpeed);

  Serial.print("Pot Value: ");
  Serial.print(PotVal);
  Serial.print("    ");
  Serial.print("Forward Motor Speed: ");
  Serial.println(MotorSpeed);
}
```

To make the motor go in the other direction, I'll need to reverse the signals going to the IN1 and IN2 pins. I'll also need to map the potentiometer values from 0-512 to 255-0.

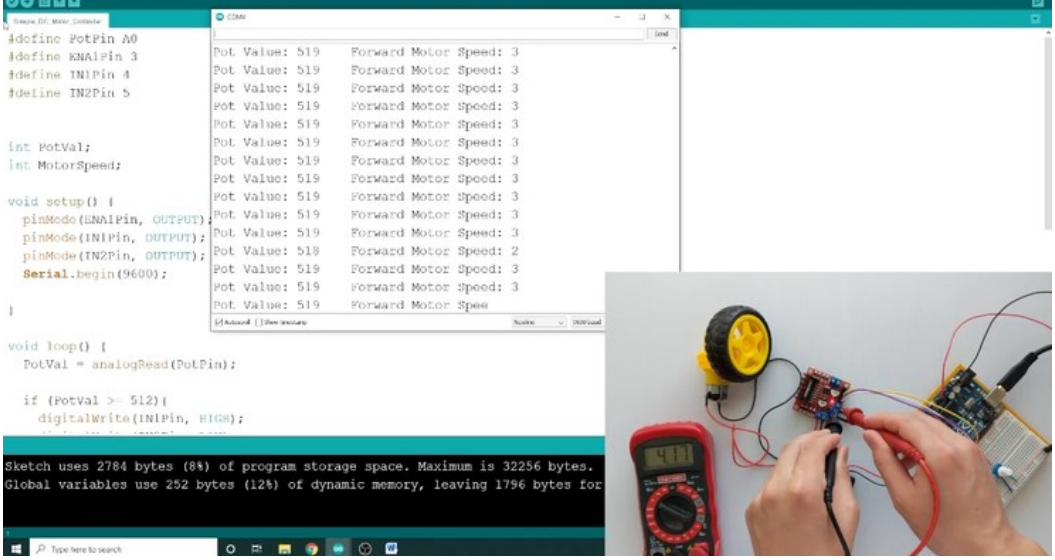
```
else{
  digitalWrite(IN1Pin, LOW);
  digitalWrite(IN2Pin, HIGH);
  MotorSpeed = map(PotVal, 0, 512, 118, 0);
  analogWrite(ENA1Pin, MotorSpeed);

  Serial.print("Pot Value: ");
  Serial.print(PotVal);
  Serial.print("    ");
  Serial.print("Reverse Motor Speed: ");
  Serial.println(MotorSpeed);
}
```

The full Arduino sketch can be found here. After uploading the code, rotating the potentiometer should control the speed of the motor. The motor should also spin in reverse if the potentiometer passes its midpoint.

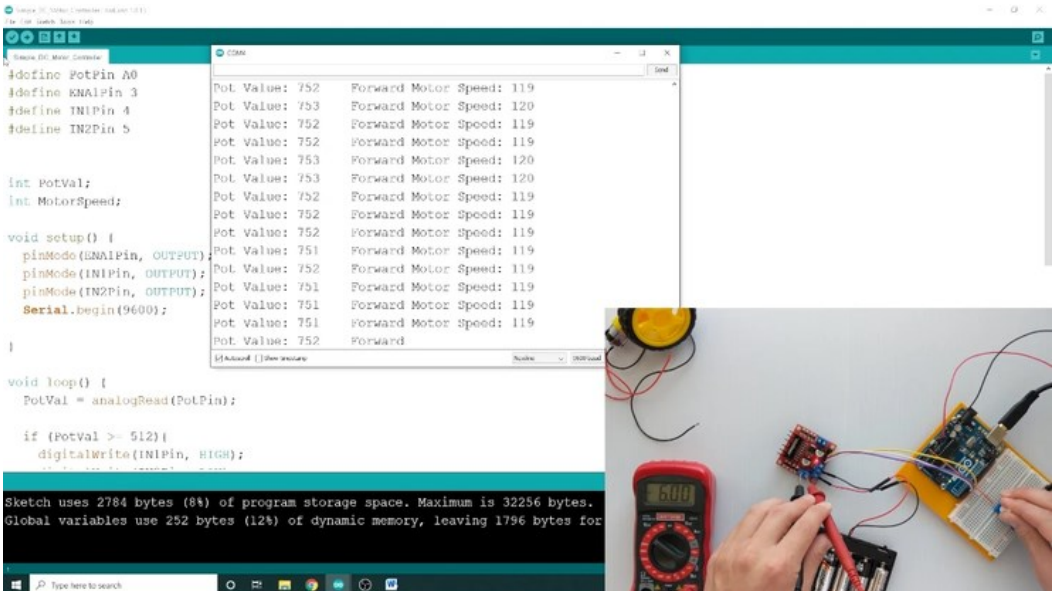


Power Issues

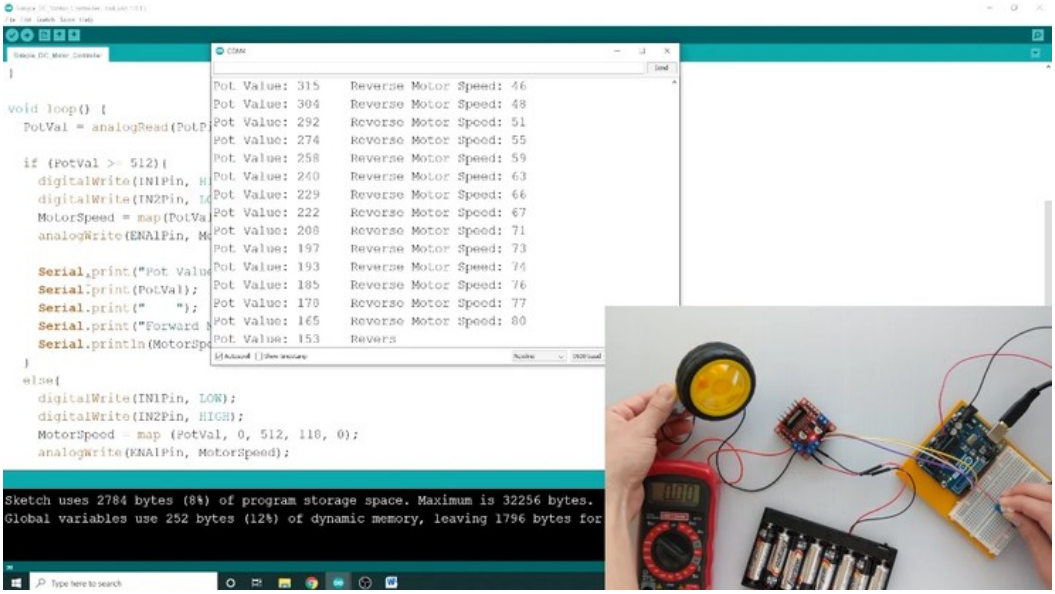
If you're using a 6V power source to power this motor controller, you may encounter an issue with the motor not changing directions. You may also find that the 5V output from the motor controller doesn't actually reach 5V when the power source is at 6V.



This issue of the motor not going in reverse is simply solved by switching out the power source. In my case, I used a 12V power source. However, it's important to be careful when doing this. The particular motors that I'm using in this example are only rated to go up to 6V. I need to find the appropriate value to send to the "analogWrite" command so that the motor outputs are sending anything higher than 6V. To do this, start by disconnecting the motor leads from the motor controller. Connect the leads of a multimeter to the motor controller output to measure the voltage. Then adjust the potentiometer until the multimeter reads 6V in each direction. At this point look at the serial monitor and take note of what the motor speed signal is. This value will now be the end points for your "map" functions in your Arduino sketch.



Update you map commands within your "if" statements with the new maximum speed value. After doing this, the motor should now go in reverse and the 5V output actually reaches 5V.



The Bored Robot LLC is a participant in the Amazon Services LLC Associates Program, an affiliate advertising program designed to provide a means for sites to earn advertising fees by advertising and linking to Amazon.com.