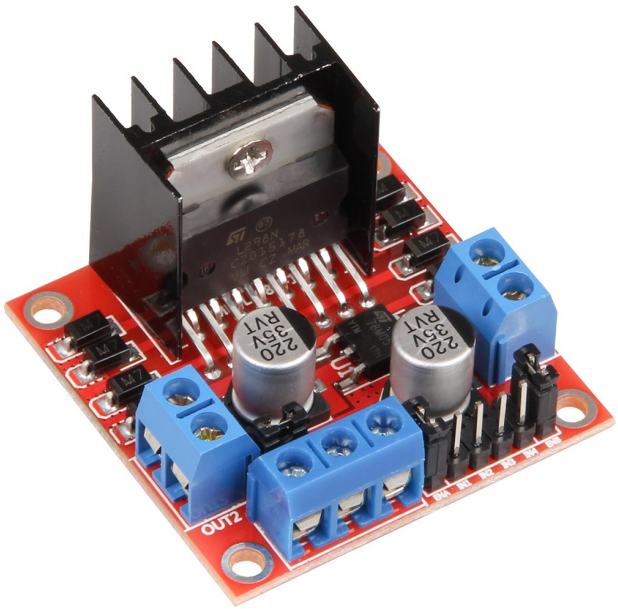


MotoDriver2

INHALTSVERZEICHNIS

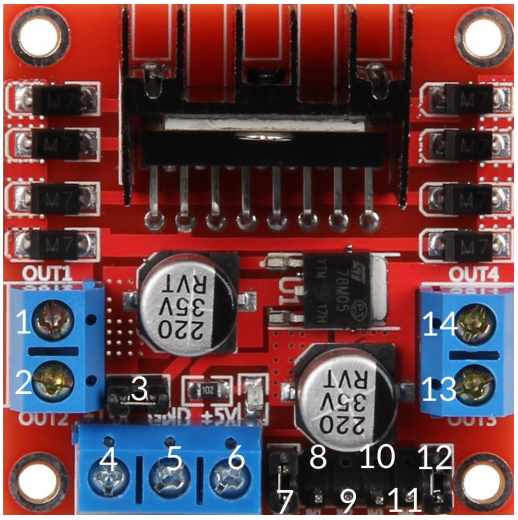
1. EINFÜHRUNG

Sehr geehrter Kunde,
vielen Dank, dass Sie sich für unser Produkt entschieden haben. Im Folgenden haben wir aufgelistet, was bei der Inbetriebnahme zu beachten ist.

2. GERÄTEÜBERSICHT

Der MotoDriver2 ist eine Erweiterungsplatine, die die Ansteuerung und Verwendung von bis zu zwei Gleichstrommotoren ermöglicht. Die Gleichstrommotoren können dabei mit einer konstanten Spannung zwischen 5V und 35V gesteuert werden.

Model	SBC-MotoDriver2
Treiber	L298N
Logische Spannung	5V
Antriebsspannung	5V - 35V
Antriebsstrom	2A
Leistung	Max. 25W
Abmessungen	43 x 43 x 25 mm

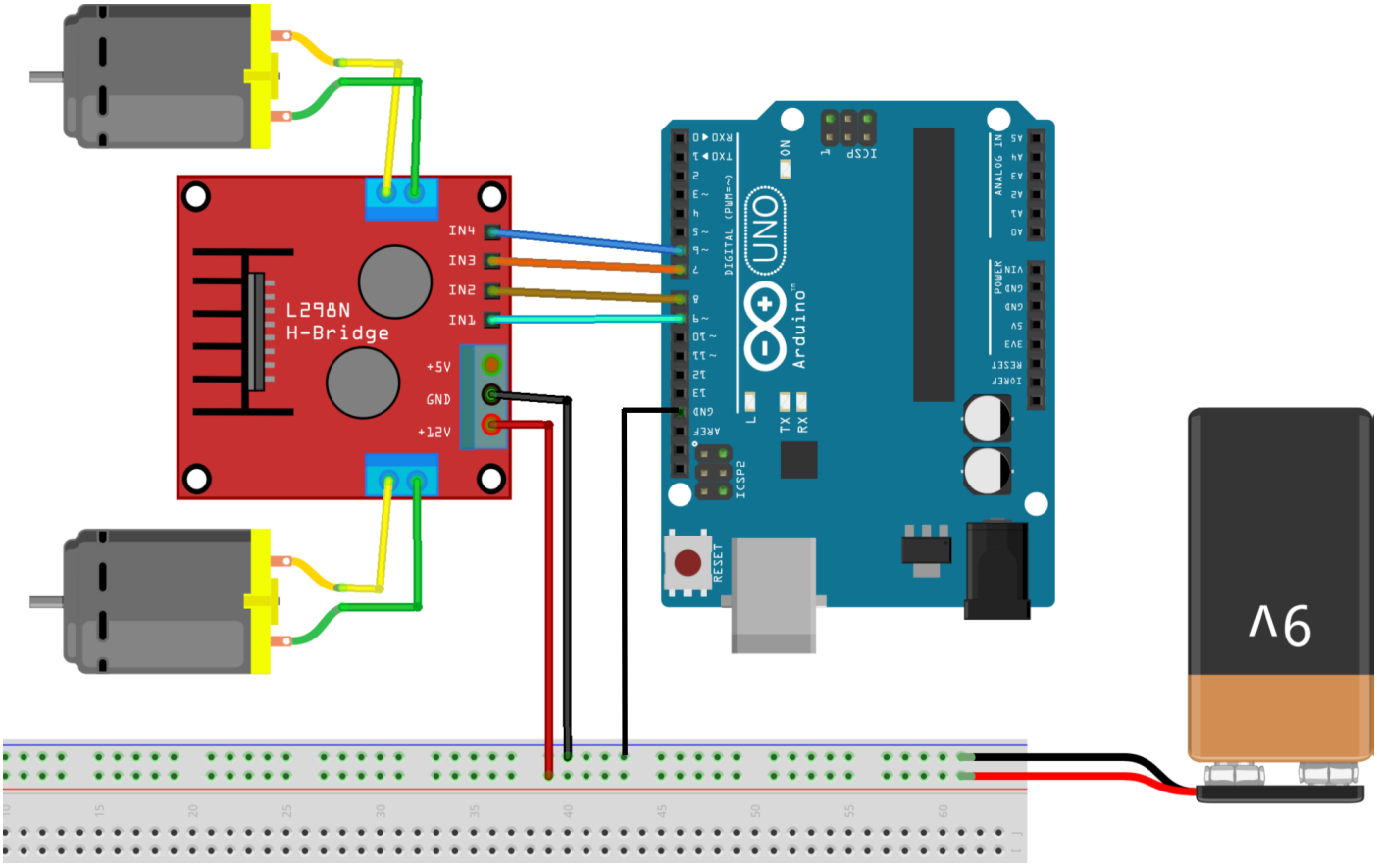


PIN	Belegung
1	DC Motor 1 / Stepper Motor +
2	DC Motor 1 / Stepper Motor GND
3	12V Jumper
4	Stromversorgung +
5	Stromversorgung GND
6	5V Ausgang (wenn Jumper 3 gesetzt)
7	DC Motor 1 Jumper
8	Input 1
9	Input 2
10	Input 3
11	Input 4
12	DC Motor 2 Jumper
13	DC Motor 2 / Stepper Motor +
14	DC Motor 2 / Stepper Motor GND

Hinweis:
Entfernen Sie den Jumper an Steckplatz 3, falls die Stromversorgung über 12V liegt. Dies aktiviert die Stromversorgung zum Onboard 5V Regulator. Der 5V Ausgang ist ideal, um z.B. die Stromversorgung für einen Arduino zu ermöglichen. Dieser ist nur aktiv, wenn der Jumper an Steckplatz 3 gesetzt wurde.

3. VERWENDUNG MIT EINEM ARDUINO

3.1 ANSCHLUSS



MotoDriver 2	Arduino
Input 1	9
Input 2	8
Input 3	7
Input 4	6

Die Stromversorgung für den MotoDriver2 (PIN 4) sollte zwischen 5V und 35V liegen.
Dies ist abhängig von Ihrer Konfiguration und den verwendeten Bauteilen.

3.2 BEISPIEL-CODE

Um die Gleichstrommotoren an dem Modul zu verwenden, verbinden Sie die Motoren, das Modul und Ihren Arduino, wie auf vorherigem Bild zu sehen. Übertragen Sie das nachfolgende Codebeispiel vollständig auf ihren Arduino, um die Funktionalität zu testen.

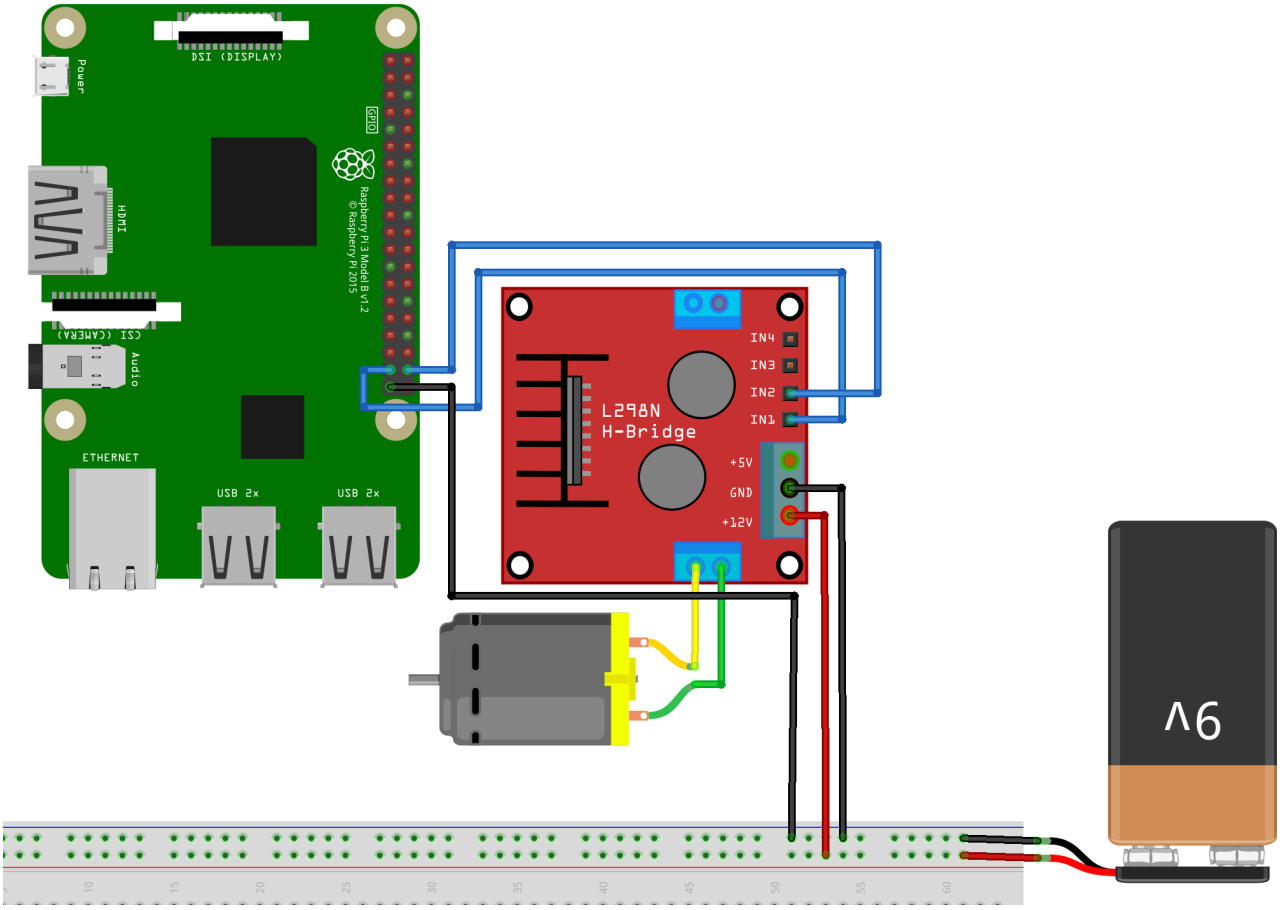
```
//Motor 1
const int motorPin1 = 9;
const int motorPin2 = 8;
//Motor 2
const int motorPin3 = 7;
const int motorPin4 = 6;
int speed = 180;

void setup(){
      //Set pins as outputs
      pinMode(motorPin1, OUTPUT);
      pinMode(motorPin2, OUTPUT);
      pinMode(motorPin3, OUTPUT);
      pinMode(motorPin4, OUTPUT);
      //Motor Control A in both directions
      analogWrite(motorPin1, speed);
      delay(2000);
      analogWrite(motorPin1, 0);
      delay(200);
      analogWrite(motorPin2, speed);
      delay(2000);
      analogWrite(motorPin2, 0);
      //Motor Control B in both directions
      analogWrite(motorPin3, speed);
      delay(2000);
      analogWrite(motorPin3, 0);
      delay(200);
      analogWrite(motorPin4, speed);
      delay(2000);
      analogWrite(motorPin4, 0);
}

void loop(){
}
```


4. VERWENDUNG MIT EINEM RASPBERRY PI

4.1 ANSCHLUSS



MotoDriver 2	Arduino
Input 1	GPIO26
Input 2	GPIO20

Die Stromversorgung für den MotoDriver2 (PIN 4) sollte zwischen 5V und 35V liegen.
Dies ist abhängig von Ihrer Konfiguration und den verwendeten Bauteilen.

4.2 BEISPIEL-CODE

Um Gleichstrommotoren an dem Modul zu verwenden, verbinden Sie einfach die Motoren, das Modul und Ihren Raspberry Pi, wie auf vorherigem Bild zu sehen. Übertragen Sie das nachfolgende Codebeispiel vollständig auf ihren Raspberry Pi, um die Funktionalität zu testen.

```
import sys
import time
import RPi.GPIO as GPIO

mode=GPIO.getmode()

GPIO.cleanup()

Forward=26
Backward=20

GPIO.setmode(GPIO.BCM)
GPIO.setup(Forward, GPIO.OUT)
GPIO.setup(Backward, GPIO.OUT)

def forward(x):
    GPIO.output(Forward, GPIO.HIGH)
    print("Moving Forward")
    time.sleep(x)
    GPIO.output(Forward, GPIO.LOW)

def reverse(x):
    GPIO.output(Backward, GPIO.HIGH)
    print("Moving Backward")
    time.sleep(x)
    GPIO.output(Backward, GPIO.LOW)

while (1):

    forward(5)
    reverse(5)
    GPIO.cleanup()
```


5. SONSTIGE INFORMATIONEN

Unsere Informations- und Rücknahmepflichten nach dem Elektrogesetz (ElektroG)

Symbol auf Elektro- und Elektronikgeräten:

Diese durchgestrichene Mülltonne bedeutet, dass Elektro- und Elektronikgeräte **nicht** in den Hausmüll gehören. Sie müssen die Altgeräte an einer Erfassungsstelle abgeben.
Vor der Abgabe haben Sie Altbatterien und Altakkumulatoren, die nicht vom Altgerät umschlossen sind, von diesem zu trennen.

Rückgabemöglichkeiten:
Als Endnutzer können Sie beim Kauf eines neuen Gerätes, Ihr Altgerät (das im Wesentlichen die gleiche Funktion wie das bei uns erworbene neue erfüllt) kostenlos zur Entsorgung abgeben.
Kleingeräte bei denen keine äußere Abmessungen größer als 25 cm sind können unabhängig vom Kauf eines Neugerätes in Haushaltsüblichen Mengen abgeben werden.

Möglichkeit Rückgabe an unserem Firmenstandort während der Öffnungszeiten:
Simac GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn

Möglichkeit Rückgabe in Ihrer Nähe:
Wir senden Ihnen eine Paketmarke zu mit der Sie das Gerät kostenlos an uns zurücksenden können.
Hierzu wenden Sie sich bitte per E-Mail an Service@joy-it.net oder per Telefon an uns.

Informationen zur Verpackung:
Verpacken Sie Ihr Altgerät bitte transportsicher, sollten Sie kein geeignetes Verpackungsmaterial haben oder kein eigenes nutzen möchten kontaktieren Sie uns, wir lassen Ihnen dann eine geeignete Verpackung zukommen.

6. SUPPORT

Wir sind auch nach dem Kauf für Sie da. Sollten noch Fragen offen bleiben oder Probleme auftauchen stehen wir Ihnen auch per E-Mail, Telefon und Ticket-Supportsystem zur Seite.

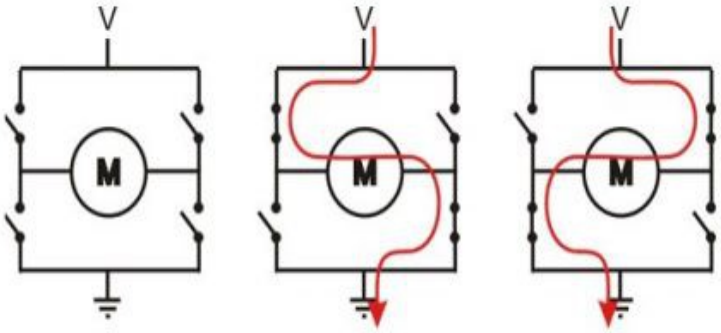
E-Mail: service@joy-it.net

Ticket-System: http://support.joy-it.net

Telefon: +49 (0)2845 98469 – 66 (9:30 - 17:00 Uhr)

Für weitere Informationen besuchen Sie unsere Website:

www.joy-it.net



Arduino + Stepper (L298N)

Description

Bipolar stepper motors always have only 4 wires. Bipolar stepper motors always have 2 coils. By driving the current in seperate directions through each of the coils, we can have a total of 4 different states:

- Coil A current flowing ‘left to right’.
- Coil A current flowing ‘right to left’.
- Coil B current flowing ‘left to right’.
- Coil B current flowing ‘right to left’.

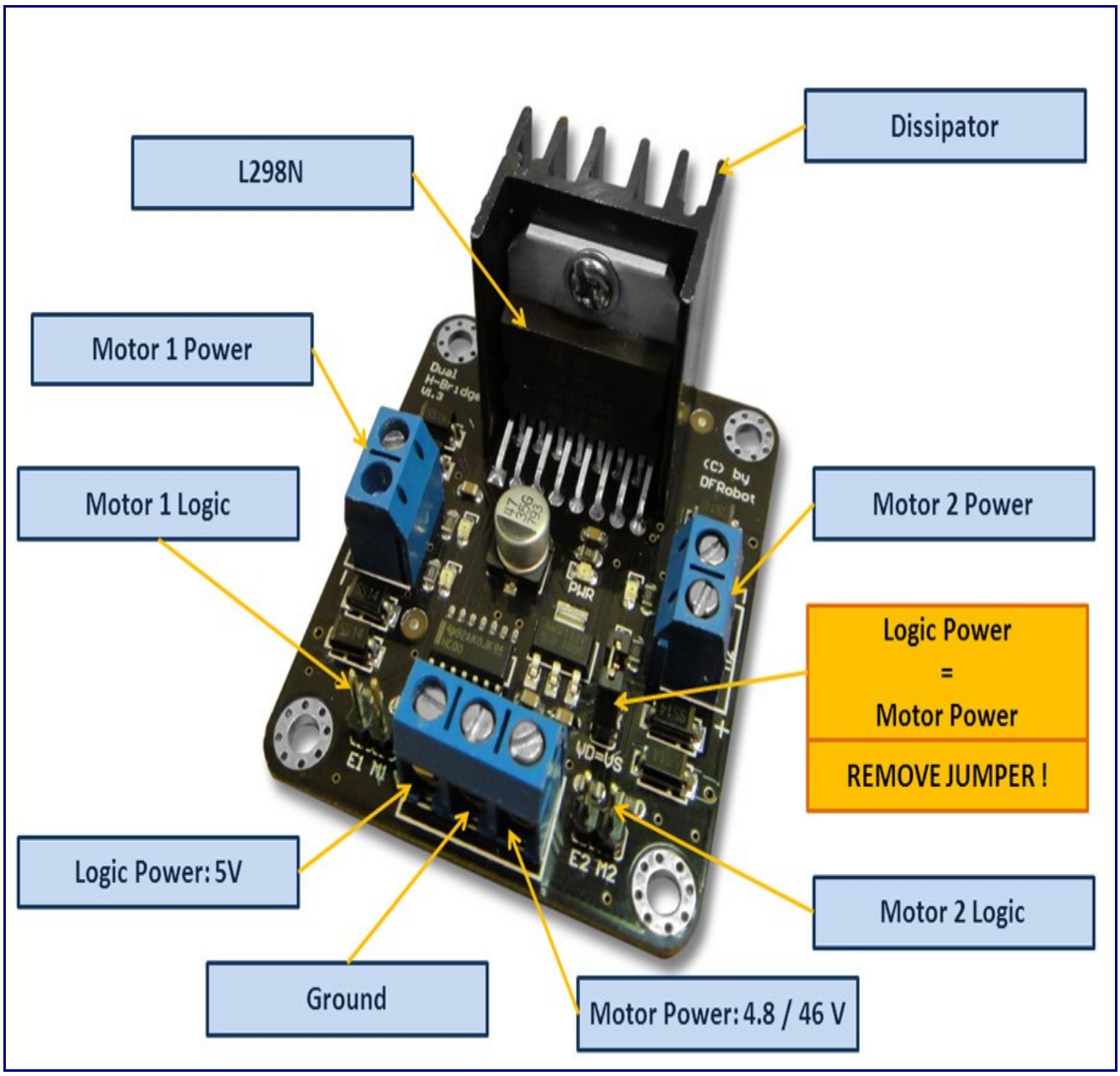
Note that the number of poles *inside* a stepper motor is often greater than just 2; individual physical poles inside the stepper motor are wired in series to create 2 coils / 4 wires you see in schematics.

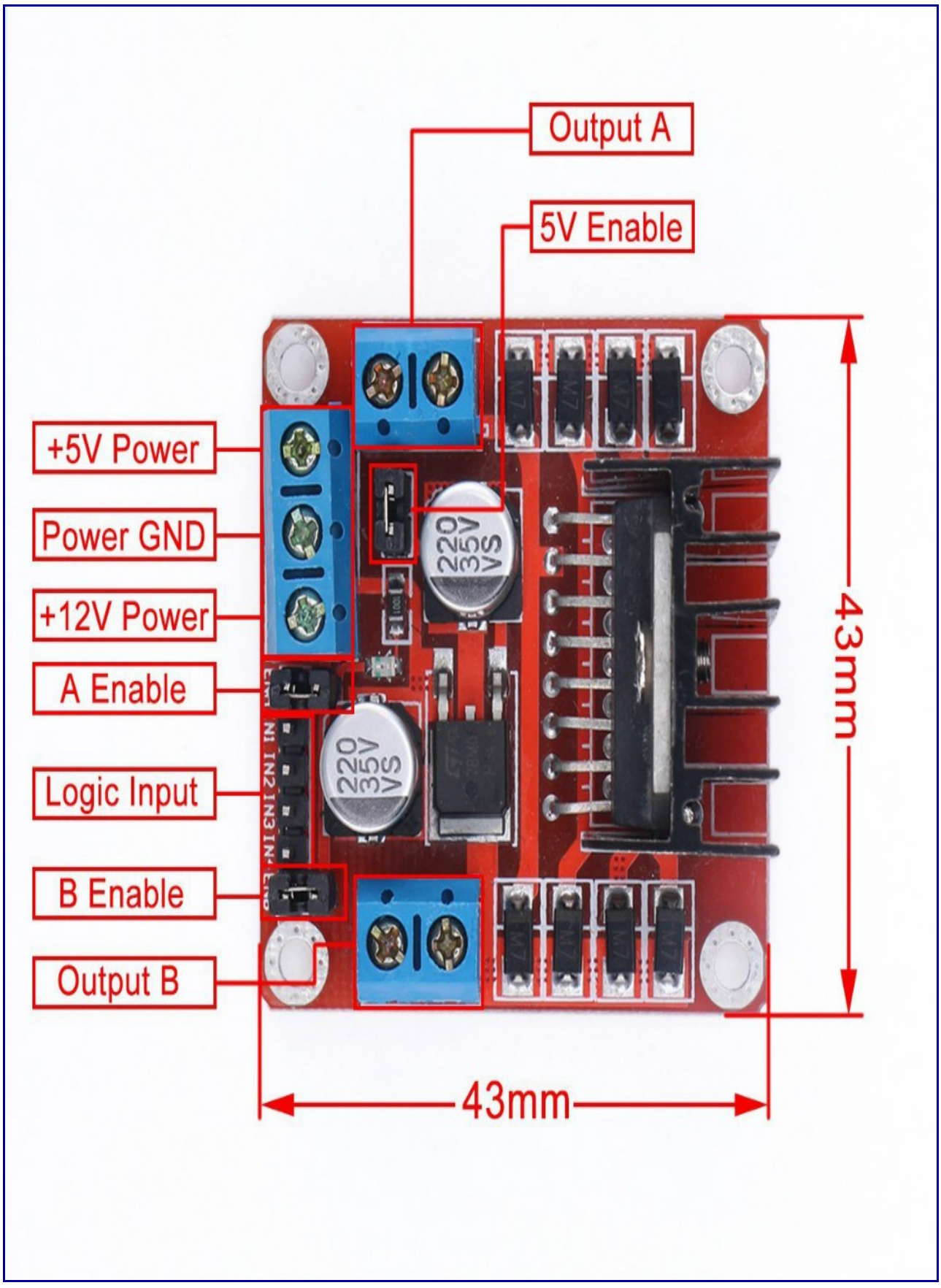
Bipolar stepper motors require a dual H-bridge to drive them; one H-bridge for each coil. Bipolar motors offer increased torque compared to unipolar motors. Flyback diodes are required to prevent voltage spikes when the power to the coil is turned off and the stepper motor acts like a generator briefly (back-emf).

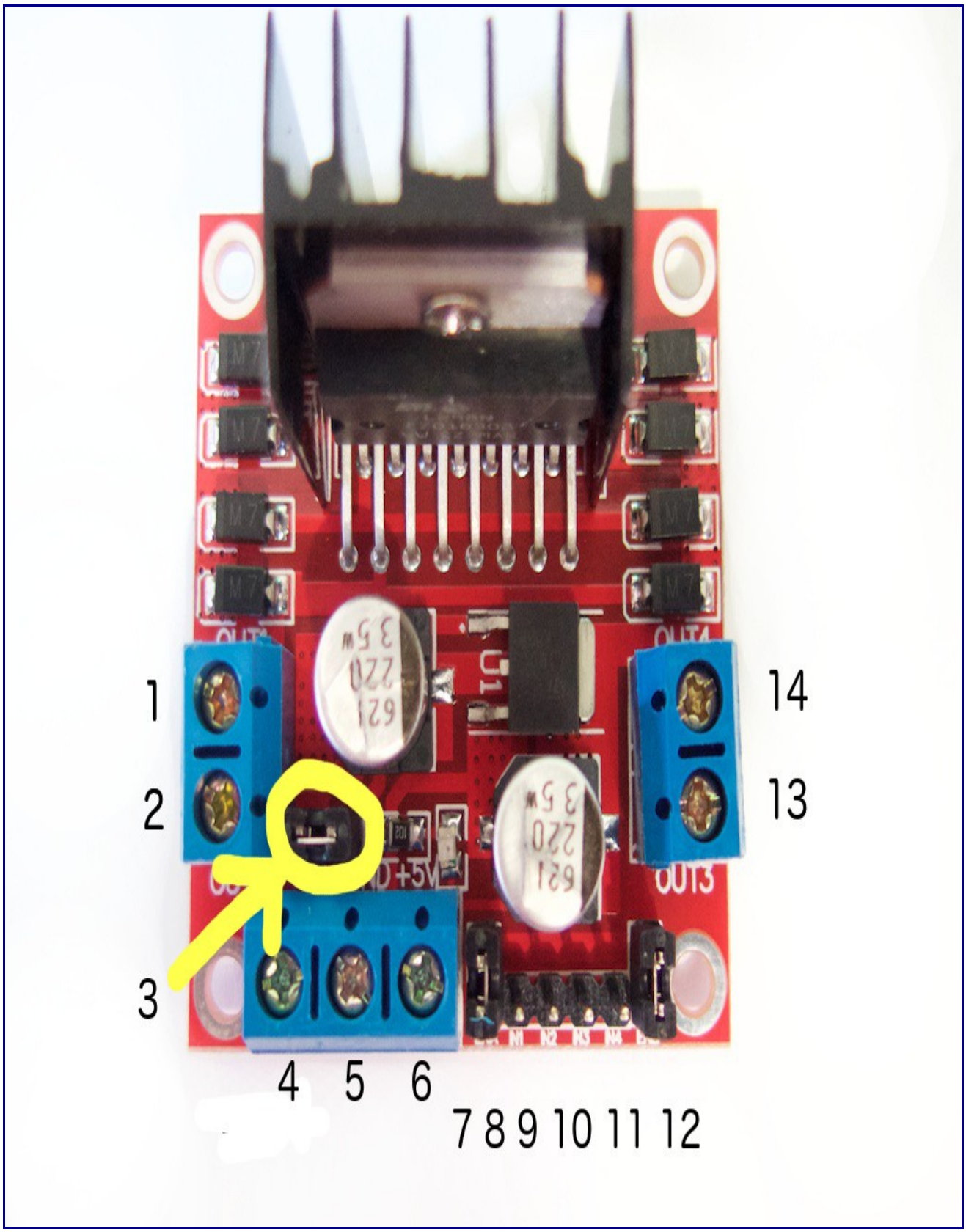
Note: You can also connect 5,6 or 8 wire unipolar motors and connect them as bipolar motors by not connecting the common lead(s). They will not have as much torque as bipolar motors due to thinner wire with a higher electrical resistance used in the coils (bifilar windings).

Hardware Required

- Arduino Board
- L298N stepper driver board
- Bipolar stepper motor (i.e. NEMA17)


L298N
Dissipator
Motor 1 Power
Motor 1 Logic
Motor 2 Power
Logic Power
=
Motor Power
REMOVE JUMPER !
Logic Power: 5V
Motor 2 Logic
Ground
Motor Power: 4.8 / 46 V
Dual H-Bridge
(C) by DFRobot
VD=VS
E1 M1
E2 M2


Output A
5V Enable
+5V Power
Power GND
+12V Power
A Enable
Logic Input
B Enable
Output B
220
35V
VS
43mm
43mm

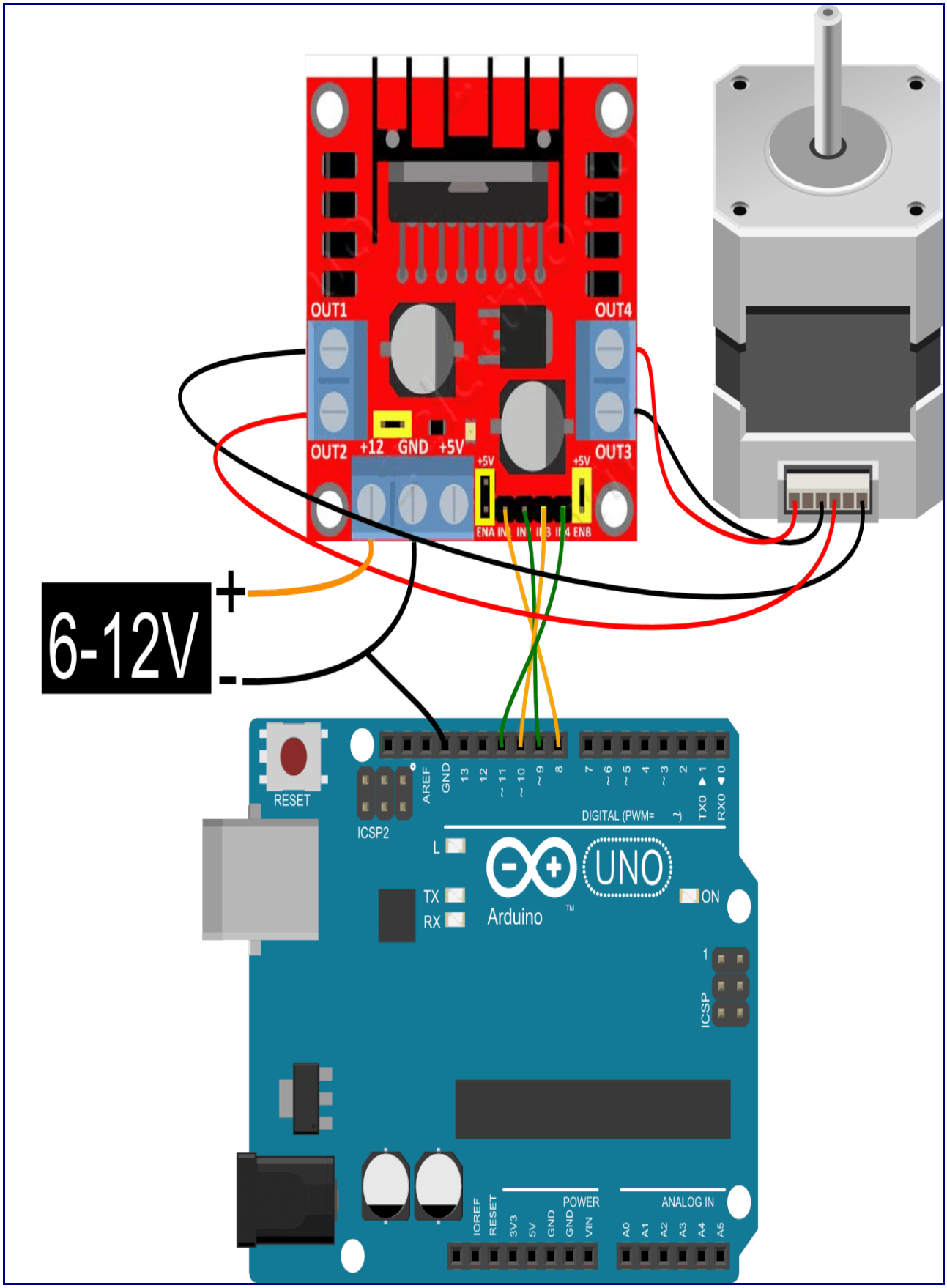

1
2
3
4
5
6
7
8
9
10
11
12
13
14
OUT4
OUT3
+5V

Pinout

1. DC motor 1 “+” or stepper motor A+
2. DC motor 1 “-” or stepper motor A-
3. 12V jumper – remove this if using a supply voltage greater than 12V DC. When the jumper is in place, the onboard voltage regulator is active (12V max to 5V).
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if the 12V jumper at #3 is in place. This is ideal for powering your Arduino.
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
13. DC motor 2 “+” or stepper motor B+
14. DC motor 2 “-” or stepper motor B-

Connect the L298N stepper driver board to a 9V…12V power supply using pin #4 (+12V) and #5 (GND). Leave the jumper in #3 in place. You can now use the +5V pin at #6 (and the GND pin at #5) to power your Arduino. If you remove the jumper, the onboard voltage regulator is disabled and the +5V pin at #6 is no longer active.

Circuit


OUT1
OUT4
OUT2
+12
GND
+5V
OUT3
+5V
+5V
ENA
ENB
6-12V
+
-
RESET
ICSP2
AREF
GND
DIGITAL (PWM=
TX0
RX0
L
TX
RX
UNO
Arduino
ON
ICSP
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

Code

Example 1

```
/*
Stepper Motor Control - one revolution

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

The motor should revolve one revolution in one direction, then
one revolution in the other direction.


Created 11 Mar. 2007
Modified 30 Nov. 2009
by Tom Igoe

*/

#include <Stepper.h>

const int stepsPerRevolution = 200; // 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(60);
 // 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);
}
```

Example 2

```
/*
Stepper Motor Control - one step at a time

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

The motor will step one step at a time, very slowly. You can use this to
test that you've got the four wires of your stepper wired to the correct
pins. If wired correctly, all steps should be in the same direction.
```

```
Use this also to count the number of steps per revolution of your motor,
if you don't know it. Then plug that number into the oneRevolution
example to see if you got it right.

Created 30 Nov. 2009
by Tom Igoe

*/

#include <Stepper.h>

const int stepsPerRevolution = 200; // 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);

int stepCount = 0; // number of steps the motor has taken

void setup() {
 // initialize the serial port:
 Serial.begin(9600);
}

void loop() {
 // step one step:
 myStepper.step(1);
 Serial.print("steps:");
 Serial.println(stepCount);
 stepCount++;
 delay(500);
}
```

Example 3

```
/*
Stepper Motor Control - speed control

This program drives a unipolar or bipolar stepper motor.
The motor is attached to digital pins 8 - 11 of the Arduino.
A potentiometer is connected to analog input 0.

The motor will rotate in a clockwise direction. The higher the potentiometer
value,
the faster the motor speed. Because setSpeed() sets the delay between steps,
you may notice the motor is less responsive to changes in the sensor value at
low speeds.

Created 30 Nov. 2009
Modified 28 Oct 2010
by Tom Igoe

*/

#include <Stepper.h>

const int stepsPerRevolution = 200; // 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);

int stepCount = 0; // number of steps the motor has taken

void setup() {
 // nothing to do inside the setup
}

void loop() {
 // read the sensor value:
 int sensorReading = analogRead(A0);
 // map it to a range from 0 to 100:
 int motorSpeed = map(sensorReading, 0, 1023, 0, 100);
 // set the motor speed:
 if (motorSpeed > 0) {
 myStepper.setSpeed(motorSpeed);
 // step 1/100 of a revolution:
 myStepper.step(stepsPerRevolution / 100);
 }
}
```