

Arduino and Stepper Motor Configurations

[Learn](#) > Arduino and Stepper Motor Configurations

Hardware Required

Circuit

Unipolar Stepper Circuit and schematic

Bipolar Stepper Circuit and Schematic

Examples

MotorKnob


StepperOneRevolution


StepperOneStepAtATime

StepperSpeedControl

Arduino and Stepper Motor Configurations

Learn how to control a variety of stepper motors using unipolar / bipolar circuits with Arduino.

 AUTHOR: [Arduino](#)

 LAST REVISION: **31/08/2022, 06:54**

Stepper motors, due to their unique design, can be controlled to a high degree of accuracy without any feedback mechanisms. The shaft of a stepper, mounted with a series of magnets, is controlled by a series of electromagnetic coils that are charged positively and negatively in a specific sequence, precisely moving it forward or backward in small "steps".

There are two types of steppers, Unipolars and Bipolars, and it is very important to know which type you are working with. For each of the motors, there is a different circuit. The example code will control both kinds of motors. See the [unipolar](#) and [bipolar](#) motor schematics for information on how to wire up your motor.


The stepper is controlled by with digital pins 8, 9, 10, and 11 for either unipolar or bipolar motors. The Arduino board will connect to a [U2004 Darlington Array](#) if you're using a unipolar stepper or a [SN754410NE H-Bridge](#) if you have a bipolar motor.

Hardware Required

- Arduino Board
- stepper motor
- U2004 Darlington Array (if using a unipolar stepper)
- SN754410ne H-Bridge (if using a bipolar stepper)
- power supply appropriate for your particular stepper
- hook-up wires
- breadboard

Circuit

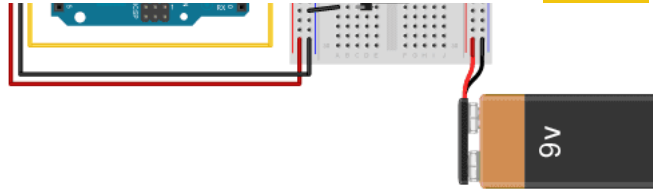
Below you'll find circuits for both unipolar and bipolar steppers. In either case, it is best to power your stepper motors from an external supply, as they draw too much to be powered directly from your Arduino board.

 Note: Both circuits below are four wire configurations. Two wire configurations will not work with the code provided.

Unipolar Stepper Circuit and schematic

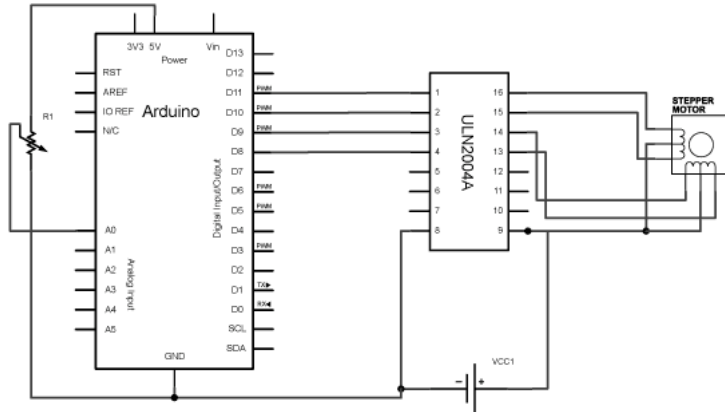


Ayuda



Made with Fritzing.org

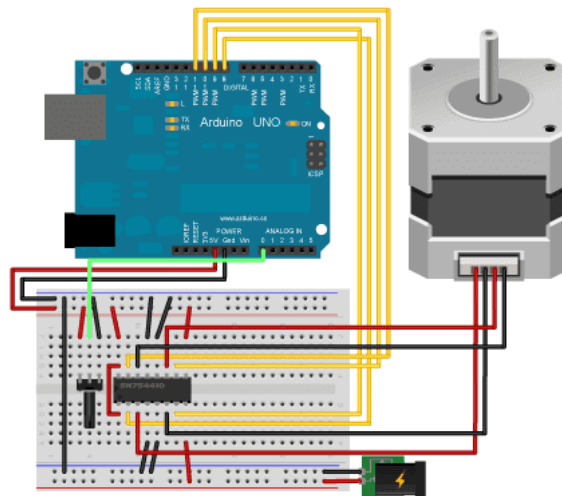
Unipolar Motor Knob Circuit. Image made using Fritzing.



Made with Fritzing.org

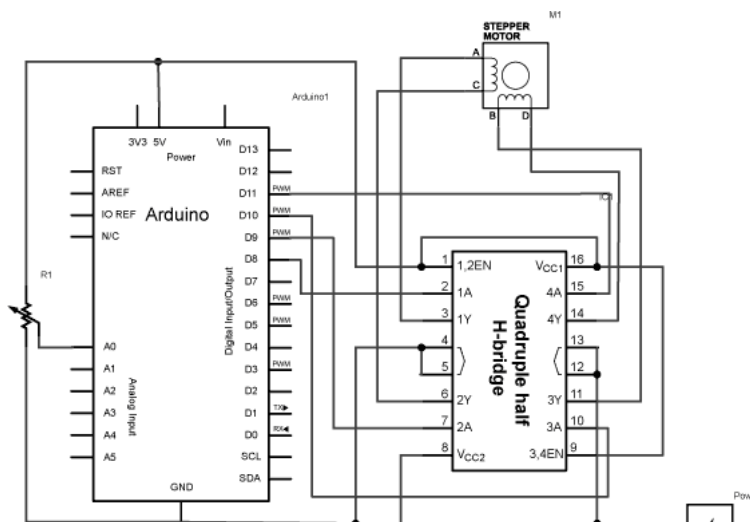
Unipolar Motor Knob Schematic. Image made using Fritzing.

Bipolar Stepper Circuit and Schematic



Made with Fritzing.org

Bipolar Motor Knob Circuit. Image made using Fritzing.



Ayuda

Examples

MotorKnob

A stepper motor follows the turns of a potentiometer (or other sensor) on analog input 0.

[COPY](#)

```

1 #include <Stepper.h>
2
3 // change this to the number of steps on your motor
4 #define STEPS 100
5
6 // create an instance of the stepper class, specifying
7 // the number of steps of the motor and the pins it's
8 // attached to
9 Stepper stepper(STEPS, 8, 9, 10, 11);
10
11 // the previous reading from the analog input
12 int previous = 0;
13
14 void setup() {
15     // set the speed of the motor to 30 RPMs
16     stepper.setSpeed(30);
17 }
18
19 void loop() {
20     // get the sensor value
21     int val = analogRead(0);
22
23     // move a number of steps equal to the change in the
24     // sensor reading
25     stepper.step(val - previous);
26
27     // remember the previous value of the sensor
28     previous = val;
29 }

```

StepperOneRevolution

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

[COPY](#)

```

1 #include <Stepper.h>
2
3 const int stepsPerRevolution = 200; // change this to fit the number of steps
4 // for your motor
5
6 // initialize the stepper library on pins 8 through 11:
7 Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);
8
9 void setup() {
10     // set the speed at 60 rpm:
11     myStepper.setSpeed(60);
12     // initialize the serial port:
13     Serial.begin(9600);
14 }
15
16 void loop() {
17     // step one revolution in one direction:
18     Serial.println("clockwise");
19     myStepper.step(stepsPerRevolution);
20     delay(500);
21
22     // step one revolution in the other direction:
23     Serial.println("counterclockwise");
24     myStepper.step(-stepsPerRevolution);
25     delay(500);
26 }

```

StepperOneStepAtATime

[Ayuda](#)

COPY

```
1 #include <Stepper.h>
2
3 const int stepsPerRevolution = 200; // change this to fit the number of steps
4 // for your motor
5
6 // initialize the stepper library on pins 8 through 11:
7 Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);
8
9 int stepCount = 0; // number of steps the motor has taken
10
11 void setup() {
12   // initialize the serial port:
13   Serial.begin(9600);
14 }
15
16 void loop() {
17   // step one step:
18   myStepper.step(1);
19   Serial.print("steps:");
20   Serial.println(stepCount);
21   stepCount++;
22   delay(500);
23 }
```

StepperSpeedControl

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.

COPY

```
1 #include <Stepper.h>
2
3 const int stepsPerRevolution = 200; // change this to fit the number of steps
4 // for your motor
5
6
7 // initialize the stepper library on pins 8 through 11:
8 Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);
9
10 int stepCount = 0; // number of steps the motor has taken
11
12 void setup() {
13   // nothing to do inside the setup
14 }
15
16 void loop() {
17   // read the sensor value:
18   int sensorReading = analogRead(A0);
19   // map it to a range from 0 to 100:
20   int motorSpeed = map(sensorReading, 0, 1023, 0, 100);
21   // set the motor speed:
22   if (motorSpeed > 0) {
23     myStepper.setSpeed(motorSpeed);
24     // step 1/100 of a revolution:
25     myStepper.step(stepsPerRevolution / 100);
26   }
27 }
```

[Back to top](#)[Trademark](#) [Help Center](#)

NEWSLETTER

FOLLOW US

[Contact Us](#)

Enter your email to sign up

SUBSCRIBE

[Distributors](#)[Careers](#)