Using a Stepper Motor Like a Servo Controlled by a Potentiometer

(G Payne – 2017)

Overview:

Stepper Motors can be accurately controlled by digital pulses. They are typically geared. In this demonstration, we'll use a potentiometer to control the direction to which the stepper motor points.

Possible Applications

- Turret control for a water cannon robot
- Directional control for an TV antenna

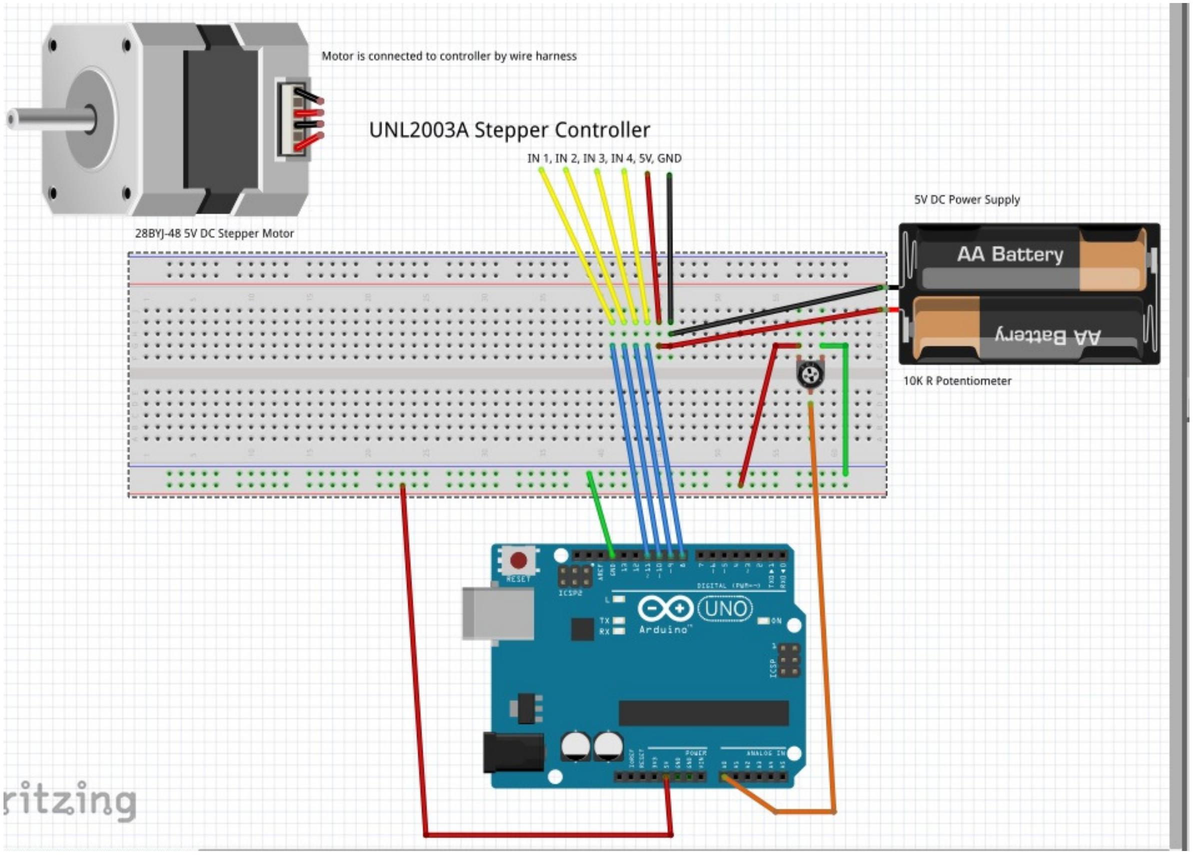
I will not go into detailed discussions of how stepper motors work in this document.
Please go to **Topic 12, "Stepper Motor Basics and Activities"** at
www.HausOfPayne.weebly.com/arduino for great resources on Stepper Motors.

Parts Needed:

-Arduino board and breadboard
-1x 10K potentiometer
- Connecting wires
- 28BYJ-48 stepper motor
- UNL2003A Darlington Array motor controller
- Separate 5V power supply to connect Motor controller to the Arduino for safety
It is NOT recommended to connect the +5V on the controller to the Arduino directly.

Demo:

Connect up the parts on the breadboard and connect it to the Arduino as shown.



Motor is connected to controller by wire harness
UNL2003A Stepper Controller
IN 1, IN 2, IN 3, IN 4, 5V, GND
5V DC Power Supply
28BYJ-48 5V DC Stepper Motor
AA Battery
AA Battery
10K R Potentiometer
UNO
Arduino
ritzing

A larger picture is on www.HausOfPayne.weebly.com/Arduino under Topic 12.

The standard ‘Stepper.h’ library that comes with Arduino is rather basic. For this demo, you will need to download and install the ‘**AccelStepper**’ library into your Arduino/libraries folder. It can be downloaded from

http://www.airspayce.com/mikem/arduino/AccelStepper/

In the Arduino IDE, start a new sketch and enter the code below. Save it as ‘potStepperDemo’.
Upload it to the Arduino board.

```
// Use a stepper motor like a servo when controlled by a potentiometer

// Based on Maker Show Episode 8 by Bret Statem
// https://channel9.msdn.com/Shows/themakershow/8

// You need the AccelStepper library for this sketch to run.  You can get it from
// here: http://aka.ms/AccelStepper
// The AccelStepper constructor expects the "pins" specified to be the ends of each
// coil respectively.
// First the ends of the Blue/Yellow coil, then the ends of the Pink/Orange coil //
// (Blue,Yellow,Pink,Orange)
// However, 28BYJ connector, ULN2003 board, and our current configuration is that pins
// are arranged in the proper FIRING order,
// Blue, Pink, Yellow, Orange.
// No biggie, that just means that we need to pay attention to what pins on our
// Arduino,
// map to which ends of the coils, and pass the pin numbers in in the proper sequence.
// To help with that, I will specify my pin variables based on their color.

#include <AccelStepper.h>
#define HALFSTEP 8
#define FULLSTEP 4


#define blue 8
#define pink 9
#define yellow 10
#define orange 11


// How many steps to go before reversing
 int targetPosition = 2048;  //2049 steps per rotation when wave or full stepping

// Initialize with pin sequence IN1-IN3-IN2-IN4 for using the
// AccelStepper with 28BYJ-48
// Notice, I'm passing them as Blue, Yellow, Pink, Orange (coil ends order) not
// Blue, Pink, Yellow, Orange (firing order).

AccelStepper stepper1(FULLSTEP, blue, yellow, pink, orange);

// -----------------------------------------------------------------------------
// Gord Payne 2017 - Control the Stepper with a 10K Potentiometer like a servo.
// I'm using FULLSTEP because it's less twitchy and doesn't introduce as much
// interference into the potentiometer as HALFSTEP.
```

```
int curPot, lastPot; // current potentiometer value, last potentiometer value
int potPin = A0; // potentiometer is connected to analog port A0
int curStep; // the current step that the motor is on (between 0 and 2048)
int curDegree; // will store the degree equivalent of the potentiometer reading

void setup() {
  Serial.begin(9600);
  //Set the initial speed (read the AccelStepper docs on what "speed" means
  stepper1.setSpeed(400.0);
  //Tell it how fast to accelerate
  stepper1.setAcceleration(500.0);
  //Set a maximum speed it should exceed
  stepper1.setMaxSpeed(8000.0);
  //Tell it to move to the target position
  stepper1.moveTo(targetPosition);
  pinMode(potPin, INPUT);
  curPot = analogRead(potPin); // read the starting value for the potentiometer
  curDegree = map(curPot,0,1023,0,360); //scale the value to a degree value between 0
                                        // and 360
  lastPot = curPot;
  Serial.print(curDegree);
  Serial.print("\t");
  curStep = map(curDegree, 0, 360, 0, 2048);//scale the degree to the corresponding
                                            //step value
  Serial.println(curStep);
  stepper1.moveTo(curStep); // set the stepper motor to the current step
  stepper1.run(); // execute the motor step change
}

void loop() {
  curPot = analogRead(potPin); // read the potentiometer

  if (abs(curPot - lastPot) > 5) { // if the potentiometer hasn't moved more than 5
                                   // values
    curDegree = map(curPot,13,1005,0,360); // update values and set the stepper motor
                                           // to the new direction
    curStep = map(curDegree, 0, 360, 0, 2048);
    Serial.print(curPot);
    Serial.print("\t");
    Serial.print(curDegree);
    Serial.print("\t");
    Serial.println(curStep);
    stepper1.moveTo(curStep);// set the stepper motor to the current step
    lastPot = curPot; // reset the last pot value to the current pot value
  }

  stepper1.run();// execute the motor step change
  delay(5); // slight delay for sketch (Don't go lower than 2. Motor may hang)
}
```

Load the sketch into the Arduino, open the Status Monitor and watch the motor turn as you turn the knob on the potentiometer.

Now go out and MAKE SOMETHING AMAZING!!!!