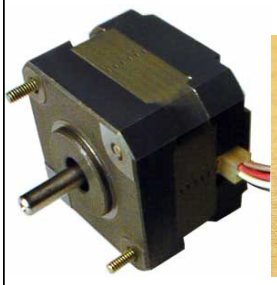
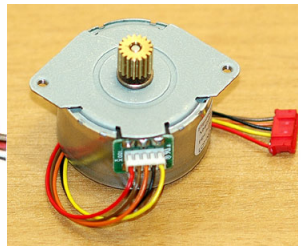
STEPPER MOTORS

Intro to Stepper Motors

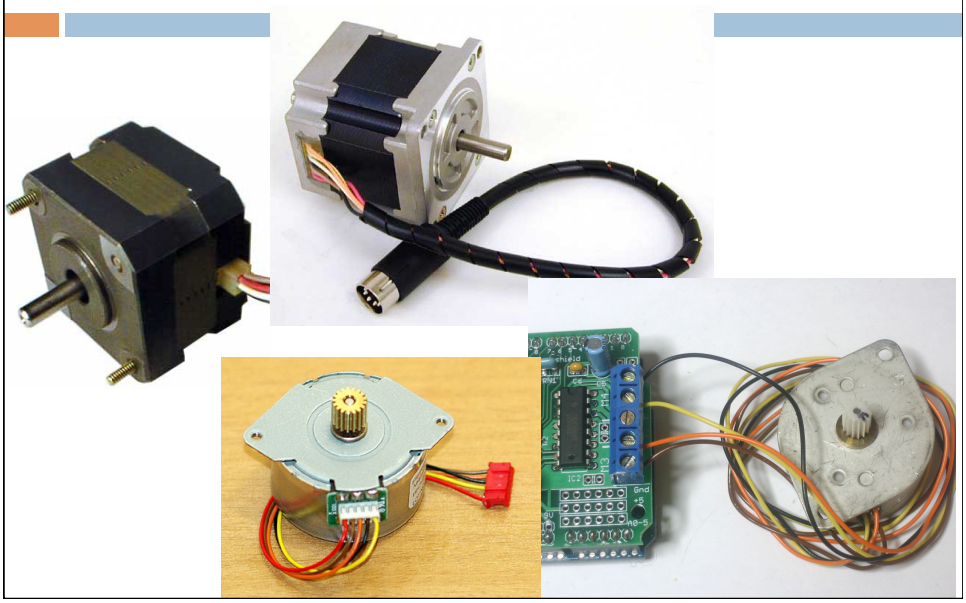
- DC motors with precise control of how far they spin
  - They have a fixed number of "steps" the take to turn one full revolution
  - You can control them one step at a time
  - Makes for very precise and repeatable positioning


Why use steppers?

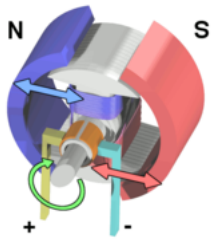
- Very precise
- Much stronger than servos
  - But, they use more current than Arduino can provide
  - So, you need some sort of external power source
- They're a little tricky to drive
  - So, you need some sort of code library, or external driver board
- We'll use both – a library, and an external board

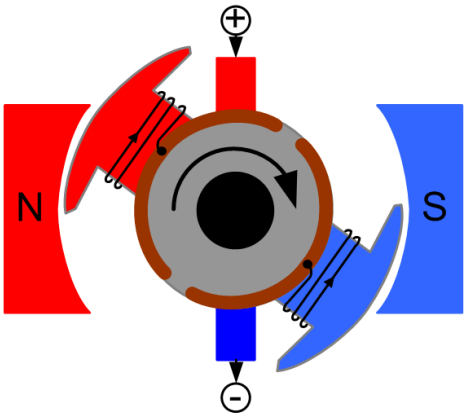
They always have multiple wires



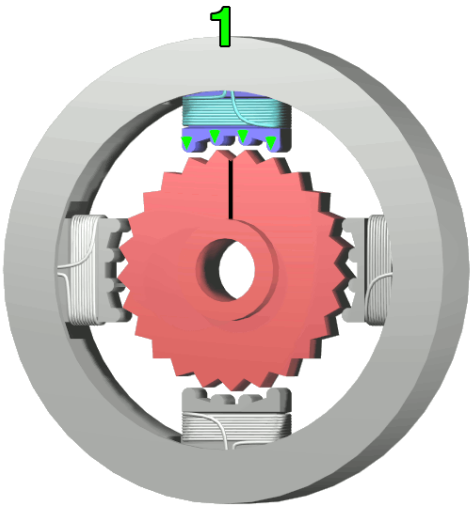
How do they Work?

- Like all motors – electro-magnets get energized and push/pull the rotor around.

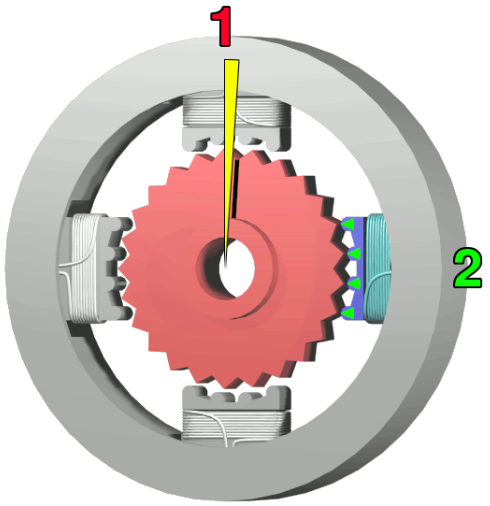




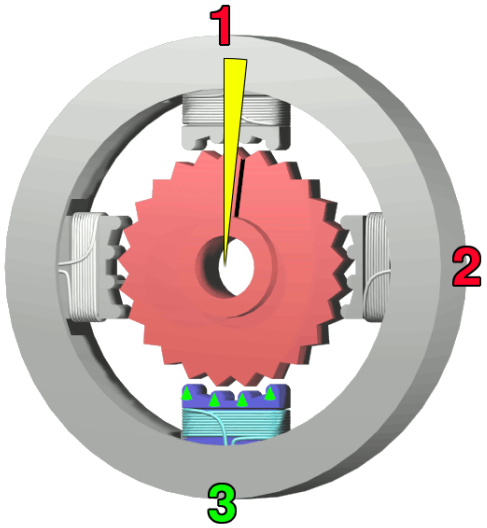
Steppers have precise internals



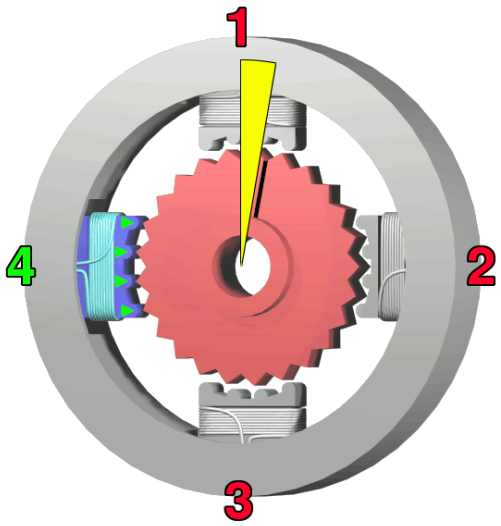
Steppers have precise internals



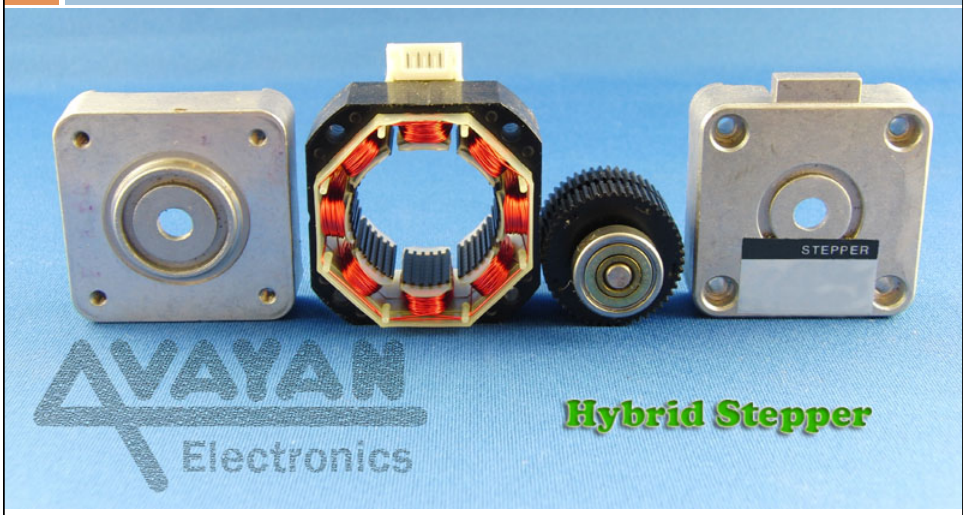
Steppers have precise internals



Steppers have precise internals

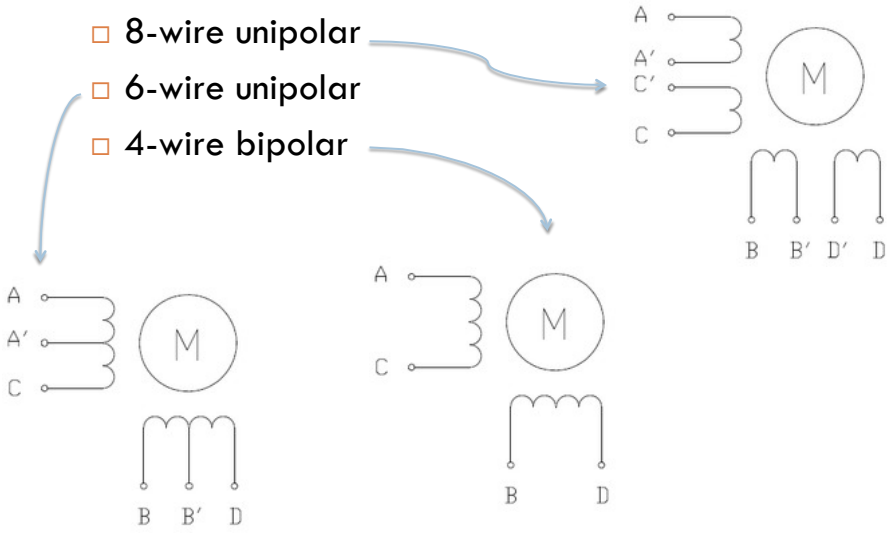


Stepper Internals



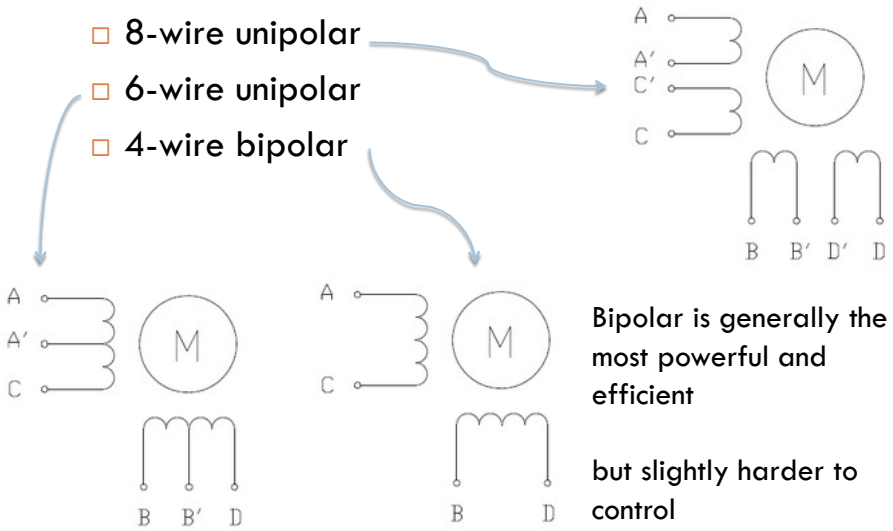
Different Flavors of Steppers

- 8-wire unipolar
- 6-wire unipolar
- 4-wire bipolar



Different Flavors of Steppers

- 8-wire unipolar
- 6-wire unipolar
- 4-wire bipolar

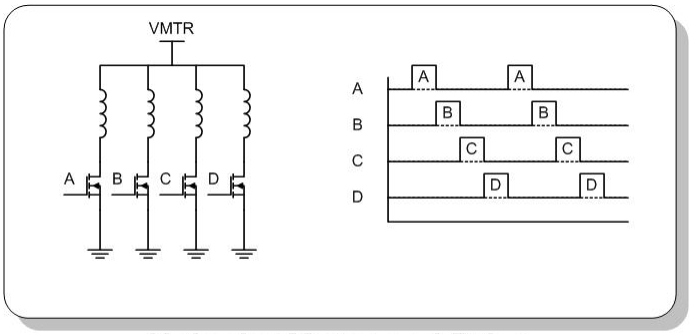


Bipolar is generally the most powerful and efficient

but slightly harder to control

Make it Turn

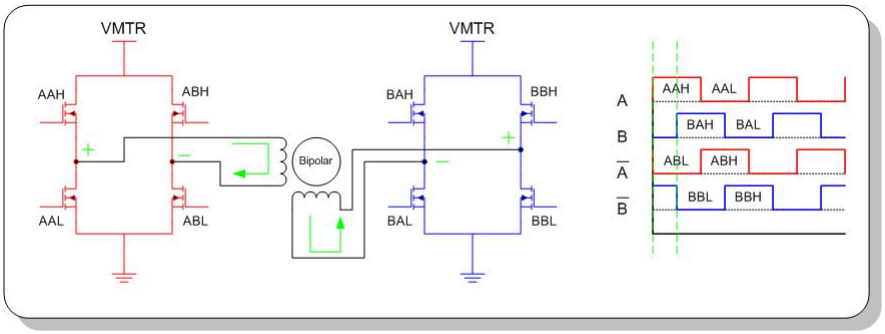
- Energize the coils in a very specific sequence



Unipolar Motor and Drive

Make it Turn

- Energize the coils in a very specific sequence



Bipolar Motor and Drive

Make it Turn

- Energize the coils in a very specific sequence

Clockwise Rotation ↓

Index	1a	1b	2a	2b
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

Clockwise Rotation ↓

Index	1a	1b	2a	2b
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1
5	1	0	0	1
6	1	1	0	0
7	0	1	1	0
8	0	0	1	1

Alternate Full Step Sequence
(Provides more torque)

Clockwise Rotation ↓

Index	1a	1b	2a	2b
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1
9	1	0	0	0
10	1	1	0	0
11	0	1	0	0
12	0	1	1	0
13	0	0	1	0
14	0	0	1	1
15	0	0	0	1
16	1	0	0	1

Half Step Sequence

Use a Library

Functions

- Stepper(steps, pin1, pin2)
- Stepper(steps, pin1, pin2, pin3, pin4)
- setSpeed(rpm)
- step(steps)

Example

- Motor Knob

Knob Example

```
#include <Stepper.h>

// change this to the number of steps on your motor
#define STEPS 100

// create an instance of the stepper class, specifying
// the number of steps of the motor and the pins it's
// attached to
Stepper stepper(STEPS, 8, 9, 10, 11);

// the previous reading from the analog input
int previous = 0;

void setup()
{
  // set the speed of the motor to 30 RPMs
  stepper.setSpeed(30);
}

void loop()
{
  // get the sensor value
  int val = analogRead(0);

  // move a number of steps equal to the change in the
  // sensor reading
  stepper.step(val - previous);

  // remember the previous value of the sensor
  previous = val;
}
```

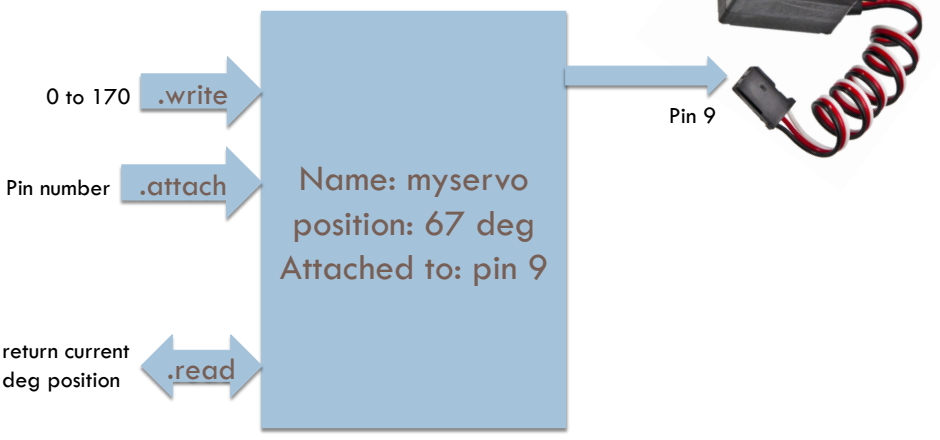
Back up: what's a Class?

- A way of packaging things up into a little code bundle
  - An instance of a class can remember things about itself
  - An instance of a class can do things when you tell it to
  - Each instance of the class can behave differently
- Let's back up and look carefully at the servo code

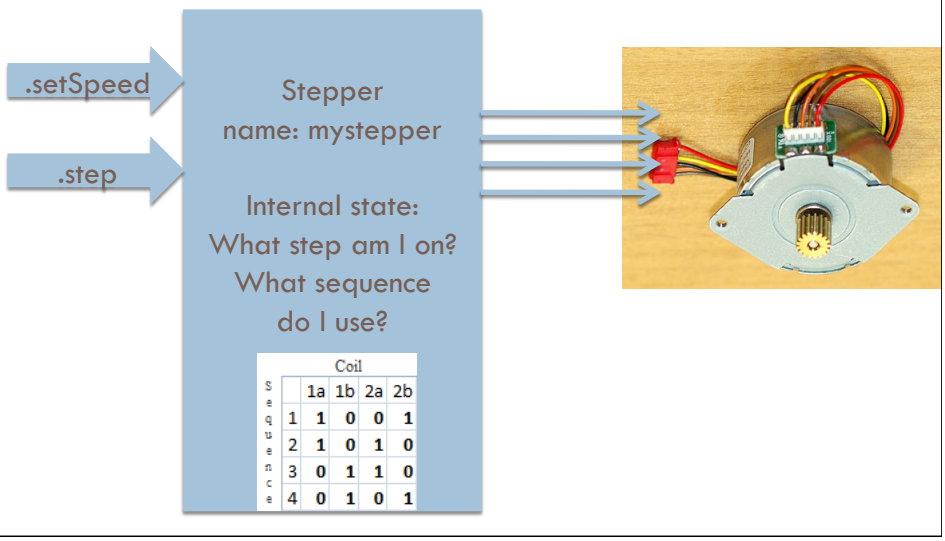
Servo + light sensor

```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop()
{
  val = analogRead(potpin);            // reads the value of the potentic

  val = map(val, 0, 1023, 0, 179);     // scale it to use it with the ser
  myservo.write(val);                  // sets the servo position accordi
  delay(15);                           // waits for the servo to get ther
}
```

Servo Object (class instance)

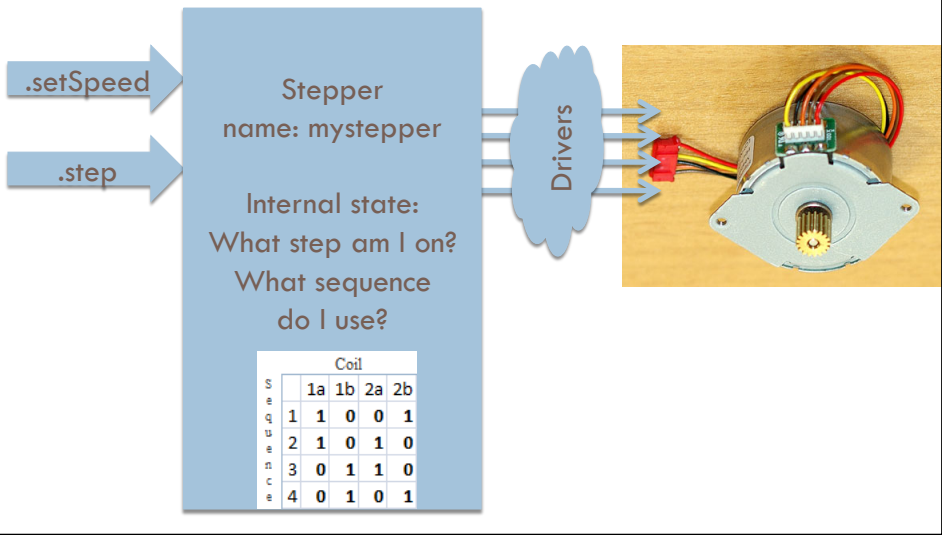


Stepper Object



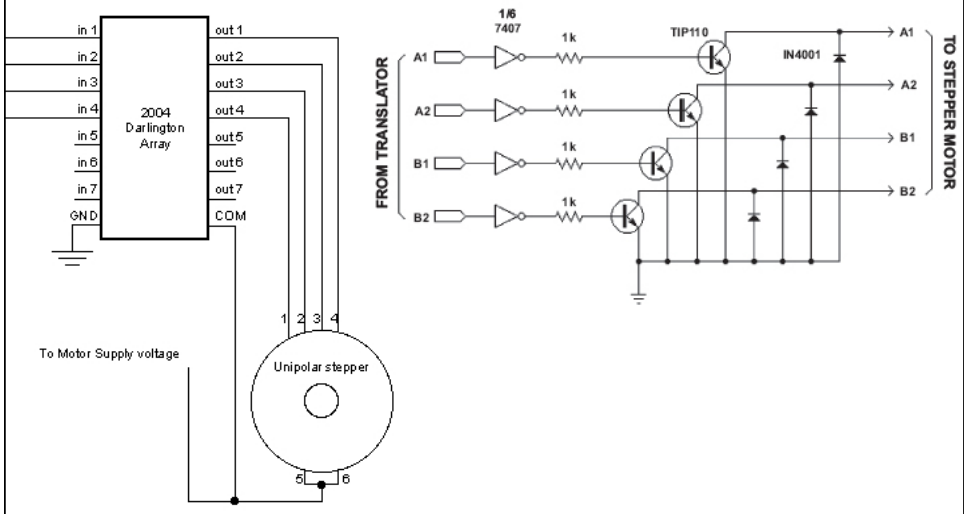
Sequence	Coil 1a	1b	2a	2b
1	1	0	0	1
2	1	0	1	0
3	0	1	1	0
4	0	1	0	1

Stepper Object

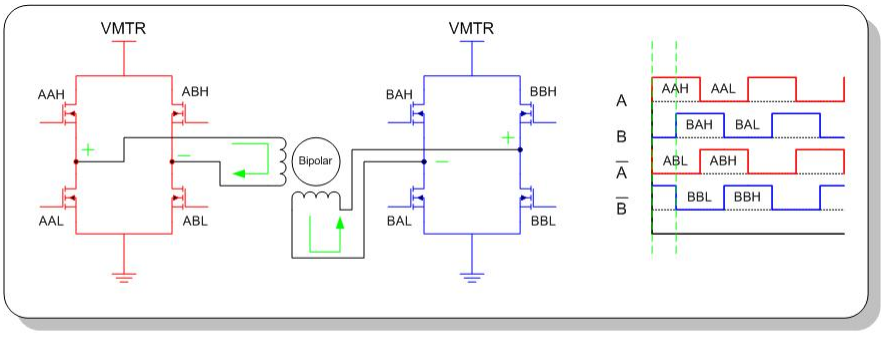


Sequence	Coil 1a	1b	2a	2b
1	1	0	0	1
2	1	0	1	0
3	0	1	1	0
4	0	1	0	1

Driver Circuits… (unipolar)

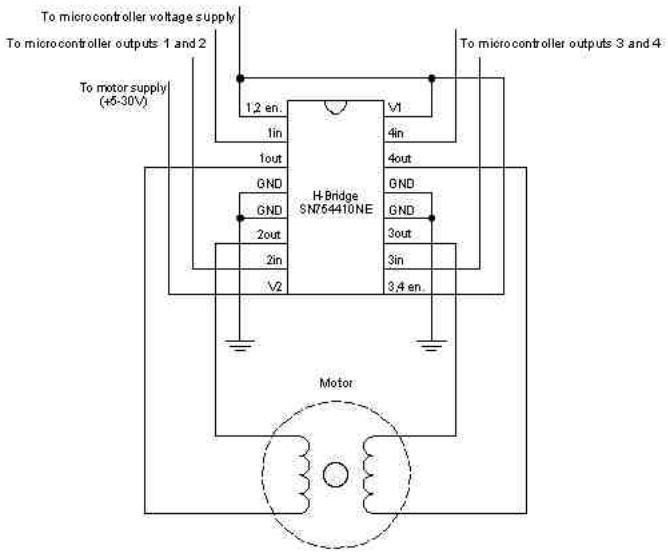


Driver Circuits… (bipolar)



Bipolar Motor and Drive

Driver Circuits… (bipolar)



Stepper Specs

- Degrees/Step
  - Common values: 15, 7.5, 3.6, 1.8 deg/step
  - This is the same as 24, 48, 100, and 200 steps/full-rev
- Coil Resistance
  - Measured resistance of motor coils
- Volts/Amps
  - Rated values for running the motor
  - Amps is the important one!
  - Remember V=IR, so V/R = I
  - Example: 6VDC, 7.9 Ω = .76A

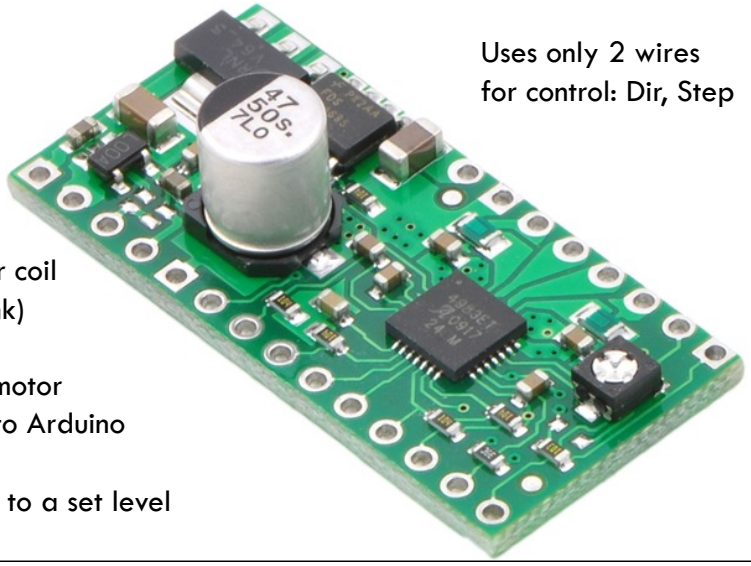
So far…

- Steppers move very precisely and are relatively powerful
- But are a bit of a pain to drive
  - Four wires from the Arduino
  - External driver circuits
  - Extra power supply to worry about
  - Use stepper library to make stepper "objects" for each one that you use
  - Your program needs to keep track of how far you've turned

Easier Motor Driving…

- There are chips specifically designed for driving steppers
  - They manage the sequence of signals
  - They manage the higher voltages of the motors
  - They have "chopper drives" to limit current
  - They can even do "microstepping"
    - This lets you do ½, ¼, 1/8, or 1/16 step
    - Increases resolution and smoothness, but might reduce power

Pololu A4988 driver



Uses only 2 wires
for control: Dir, Step

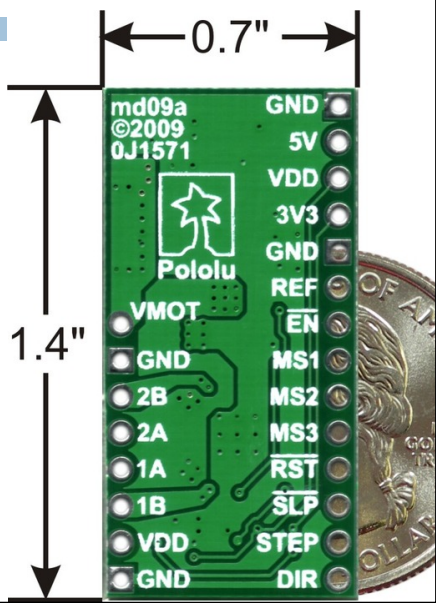
Up to 2A per coil
(with heat sink)

8 – 35V on motor
Provides 5v to Arduino

Limits current to a set level

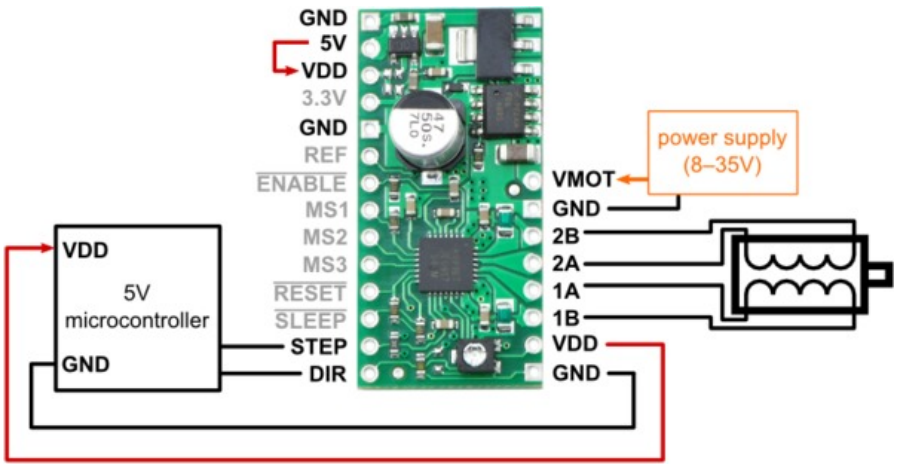
Pololu A4988 driver

MS1	MS2	MS3	Microstep Resolution
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	Quarter step
High	High	Low	Eighth step
High	High	High	Sixteenth step



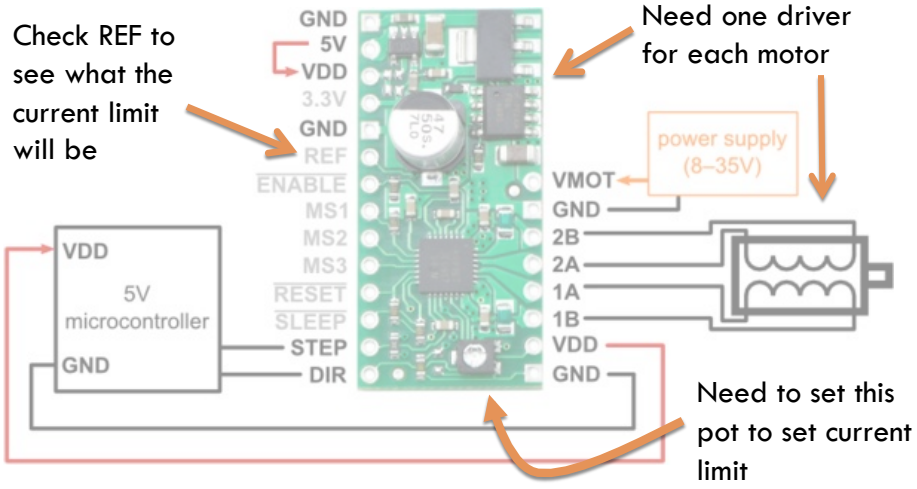
Pololu A4988 driver

Minimal Connection



Pololu A4988 driver

Minimal Connection



Current Limit

- Turn pot (use a tiny screwdriver) and check REF

$$I_{TripMAX} = V_{REF} / (8 \times R_S)$$

  - Rs = 0.05 Ω

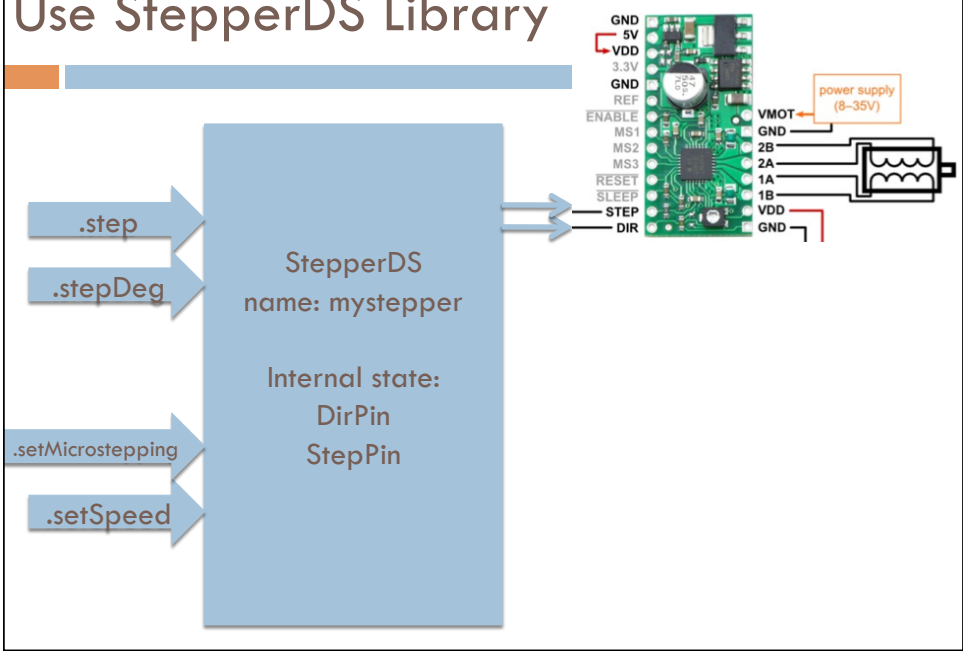
V REF	Current Limit
.1v	.250A
.15v	.375A
.2v	.500A
.25v	.625A
.3v	.750A
.35v	.875A
.4v	1.000A
.45v	1.125A

Using a Dir/Step driver

- Set the Dir pin – 0 is one dir, 1 is the other
- Toggle the Step pin up and down
  - You get one step per rising edge

```
for (int i=0; i < steps; i++) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(usDelay);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(usDelay);
}
```

Use StepperDS Library



Use the StepperDS Library

```
#include <StepperDS.h>

// 1.8 degrees per step = 200 steps per revolution
#define STEPS 200

// define the dir and step pins
#define DIR_PIN 8
#define STEP_PIN 9

// create an instance of the StepperDS class, specifying
// the number of steps of the motor and the pins it's attached to
StepperDS stepper(STEPS, DIR_PIN, STEP_PIN);

// the previous reading from the analog input
int previous = 0;

void setup() {
  // set the speed of the motor to 100 RPMs
  stepper.setSpeed(100);
}

void loop() {
  int val = analogRead(0);  // get the sensor value
  // move a number of steps equal to the change in sensor reading
  stepper.step(val - previous);
  previous = val;     // remember the previous value of the sensor
}
```

Use the StepperDS Library

```
void loop()
{
  //rotate a specific number of degrees
  Serial.println("rotateDeg(353) - 100 RPM i.e. clockwise fast");
  sepper.setSpeed(100)
  stepper.stepDeg(353);
  delay(1000);

  Serial.println("rotateDeg(-37.5) - at speed 10 i.e. CCW slow");
  stepper.setSpeed(10);
  stepper.stepDeg(-37.5);  //reverse
  delay(1000);

  //rotate a specific number of microsteps (default is full-step)
  // 1.8 degree motors have 200 steps per revolution
  Serial.println("rotate(400) CW at half speed two times around");
  stepper.setSpeed(50);
  stepper.step(400);
  delay(1000);

  Serial.println("rotate(-210) CCW at quarter speed");
  stepper.setSpeed(25);
  stepper.step(-210); //reverse
  delay(1000);

}
```

Finale

- Use one Pololu driver for each stepper
  - Measure REF, turn pot, to set current limit
  - Set microstepping if desired
  - Use stepperDS library (on web site)
    - Unzip the stepperDS.zip file in your Arduino/libraries folder
- Big motors are around ~1A, smaller are ~.5
  - Both are 1.8 deg/step (200 steps/rev)

Next Assignment

- Use a pair of steppers to make a suspended-style drawing machine
  - Use same pairs as for museum assignment
  - Due 2/28 – 3/1
- Grab a couple motors, and a couple drivers
  - Use timing belt if you like, or design your own mechanism…
  - Make a simple suspended drawing machine, or interpret this in your own way…
  - Draw randomly, or with intent…