CLIL di Automazione Industriale con ARDUINO, TINKERCAD e WOKWI

PROF. DELBARBA LUCA

01/02/2023

INTRODUZIONE

Questo testo contiene una serie di esercitazioni che possono essere realizzate con il simulatore ThinkerCad disponibile sul sito https://www.tinkercad.com/ previa registrazione e col simulatore Wokwi disponibile sul sito https://wokwi.com/ previa registrazione.

I simulatori in oggetto permettono di programmare una scheda Arduino UNO e di risolvere semplici problemi di automazione industriale utilizzando i più comuni componenti elettronici di base ed una serie nutrita di sensori ed attuatori.

La maggior parte delle esercitazioni proposte contiene una breve descrizione dei componenti utilizzati. Per ulteriori dettagli è necessario fare riferimento a testi specifici di elettronica ed automazione.

Una conoscenza di base dell'elettronica e dell'elettrotecnica è necessaria per capire gli schemi proposti.

Quasi tutti gli esercizi presentano una possibile soluzione software da caricare su una scheda Arduino.

Il vantaggio offerto dall'utilizzo di ThinkerCAD, rispetto ad altri software di simulazione, è la possibilità di replicare in modo identico il circuito su una breadboard e di utilizzare lo stesso programma simulato per Arduino.

NOTA BENE:
Alcuni esempi e immagini sono stati reperiti sul web cercando materiale che non fosse coperto da copyright. Si ringraziano tutti coloro che hanno messo reso disponibile il materiale.
Se per errore fosse stato inserito materiale protetto cortesemente segnalatelo alla mail sopra citata.

SOMMARIO

CLIL PARTE 1 STEPPER MOTOR

There are several types of electric motors, each with their own advantages and disadvantages.
If you're needing a motor to provide accurate positioning, your options are somewhat limited.
You may consider servo motors, which typically only have 180° of rotation.

Another option may be some sort of DC motor with feedback via an encoder or even a Hall effect sensor.
However, today we'll evaluate the pros and cons of the almighty hybrid stepper motor and see how it works.

A stepper motor literally takes small steps to advance its rotation.

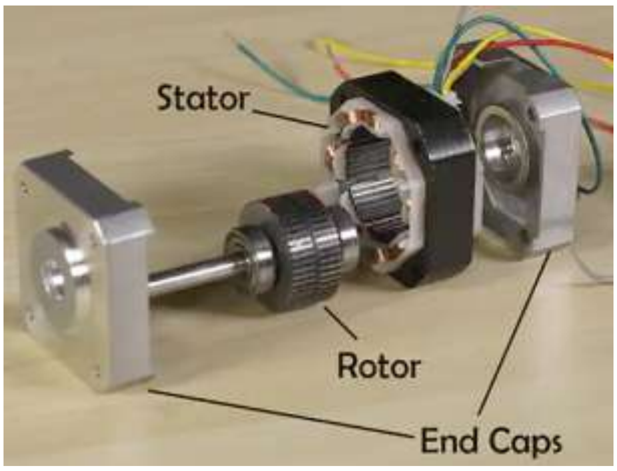
A common configuration for stepper motors is 200 steps per revolution, but you'll likely see motors with 400 or even more steps per revolution.

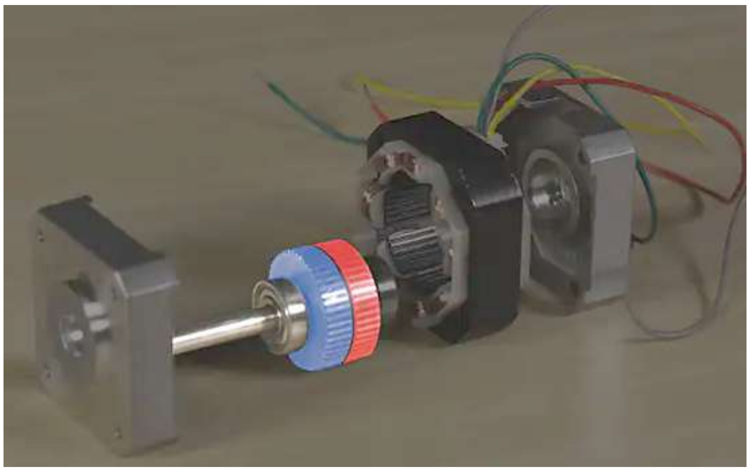
If you divide 360° by the number of steps per revolution, you'll find the number of degrees each step will rotate.
For example, a 200 step motor will rotate 1.8° per step. Stepper motors may be listed for sale by this figure rather than the number of steps/revolution.

Stepper motors are also able to prevent rotation by holding their position.
If you've seen stepper motors in action, you may have noticed they can get warm to the touch even while not rotating. This is because the motor is likely being held in position which does consume some power.

Being able to hold position and rotate precisely are the two main advantages of stepper motors, which is why they're found in devices such as CNC machines and 3D printers.

Next, let's take a look inside a stepper motor.



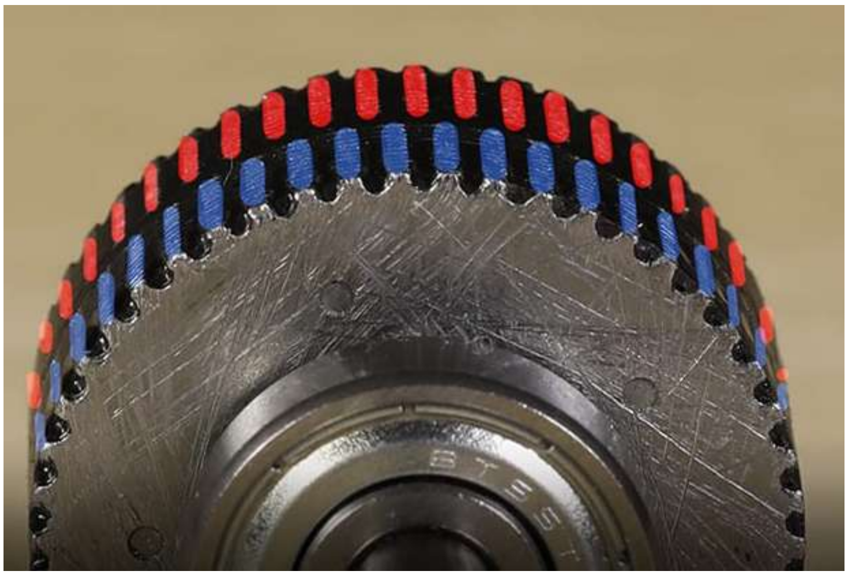


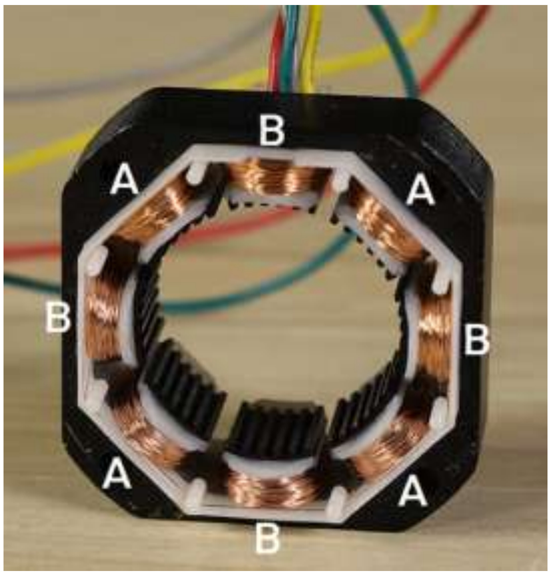
Inside the motor, we'll see two main components: the rotor and the stator.

As you may have guessed, the rotor is the part of the motor that rotates, while the stator is stationary and attached to the motor housing.

The rotor assembly consists of a permanent magnet rotor and the motor shaft.
Like all magnets, the rotor has a north and south pole.

Between these poles the rotor is divided into two halves called rotor cups; on each side of this division there are teeth that alternate. North teeth line up with the troughs between the South teeth.





There are teeth in the motor's stator as well. Looking at the stator more closely, we'll see that it consists of eight electromagnetic coils.
These eight coils are divided between two electrical phases, typically named phase A and B.

The image shows how the phases are divided within the stator.

Normally each phase will have two wires exiting the motor, some motors will have an additional center tap wire on each phase.

Energizing a phase of the motor will cause the rotor to turn to the next "step".

The teeth of the energized stator poles will attract rotor teeth of opposing polarity.
The rotor will be held in this position until power is removed.

To rotate the motor continuously, the phases need to be energized in a particular order: typically +A, +B, -A, -B.

It's impractical to make these changes manually, which is why stepper motors are usually accompanied by a microcontroller and a stepper motor driver IC (integrated circuit) or driver board.

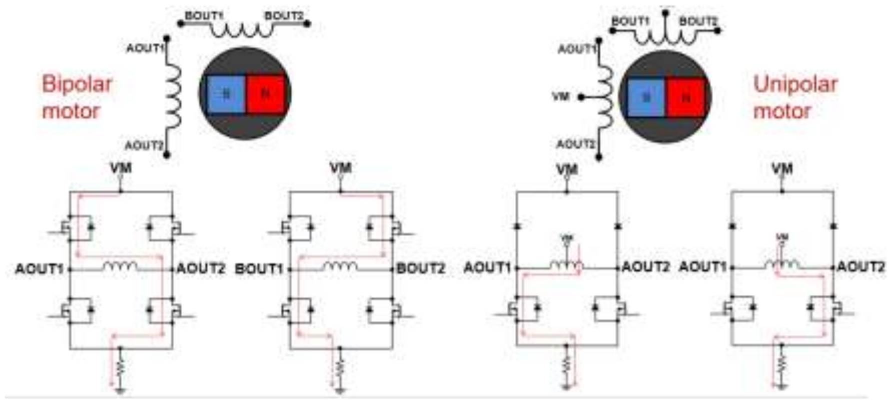
Driving a motor in the way described above is called "wave stepping".

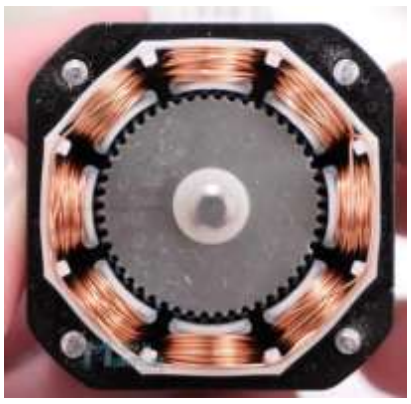
There are several different methods of driving stepper motors. Most commonly found are full stepping, which delivers the highest torque, and many degrees of microstepping, which allow for smoother operation.

Stepper motors have revolutionized automation, enabling precision operations even without sensors for feedback.

TYPES OF STEPPER MOTOR

Broadly speaking there are two types of stepper motor - unipolar and bipolar.

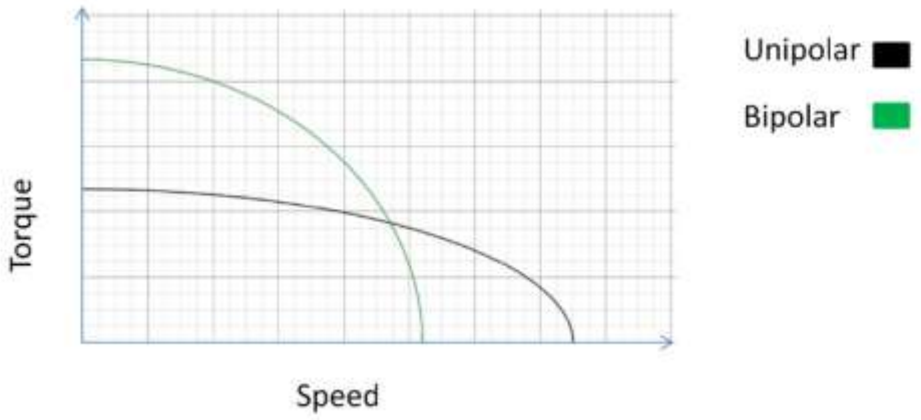




The main difference between "unipolar" and "bipolar" stepper motors is the availability of the center-tap wire, which splits the full coils of the winding in half.
This splitting can be done with one connection wire for the pair or two wires (one for the adjacent ends of each coil). By deleting the center tap, the unipolar connection becomes a bipolar-series connection.

Motor-lead colors are somewhat standardized in the industry and are fully consistent within the product line of an individual vendor, so many wiring diagrams show color rather than numbering the leads.

However, there is more to the situation than just choosing unipolar or bipolar configuration. It also means a change in the electrical characteristics of the windings inside the motor and thus affects voltage, resistance, and inductance, velocity, acceleration, and torque characteristics (see Figure).

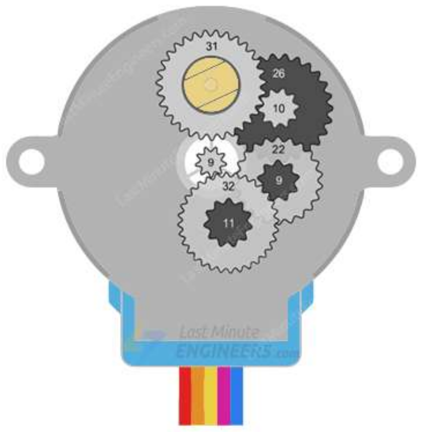


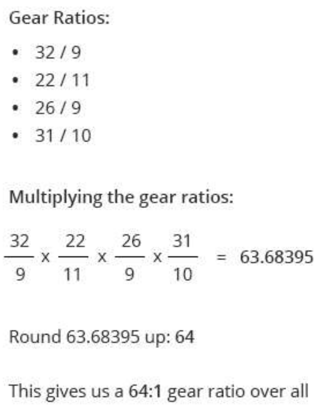
Unipolar and bipolar pole arrangements have different speed versus torque rolloff curves

Bipolar motors have 4 wires connecting to the two separate coils inside the motor - one pair for each coil.
There are also two types of unipolar motor, those with 5 wires and those with 6 wires.

The 6-wire motors can also be referred to as hybrid motors.
They are similar to the 4-wire bipolar motors and just have an extra wire connected to the centre of each of the coils. If you want to use a 6-wire motor in bipolar mode just ignore the wires that connect to the centres of the coils.

The 5-wire motors cannot be driven by a driver designed for a bipolar motor.
An example of a 5-wire motor is the small **28BYJ-48** motor which can be seen in many Arduino projects and usually uses a **ULN2003** chip as its driver.




Gear Ratios:

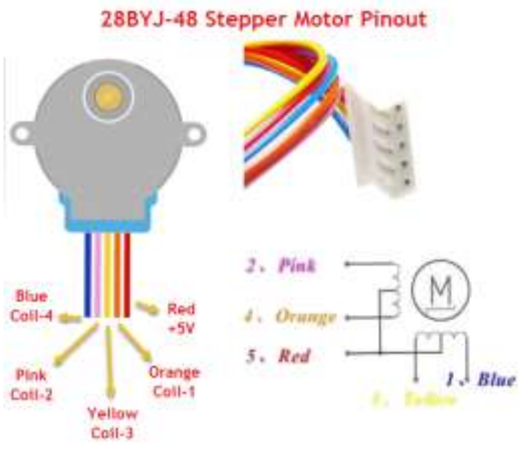
- 32 / 9
- 22 / 11
- 26 / 9
- 31 / 10

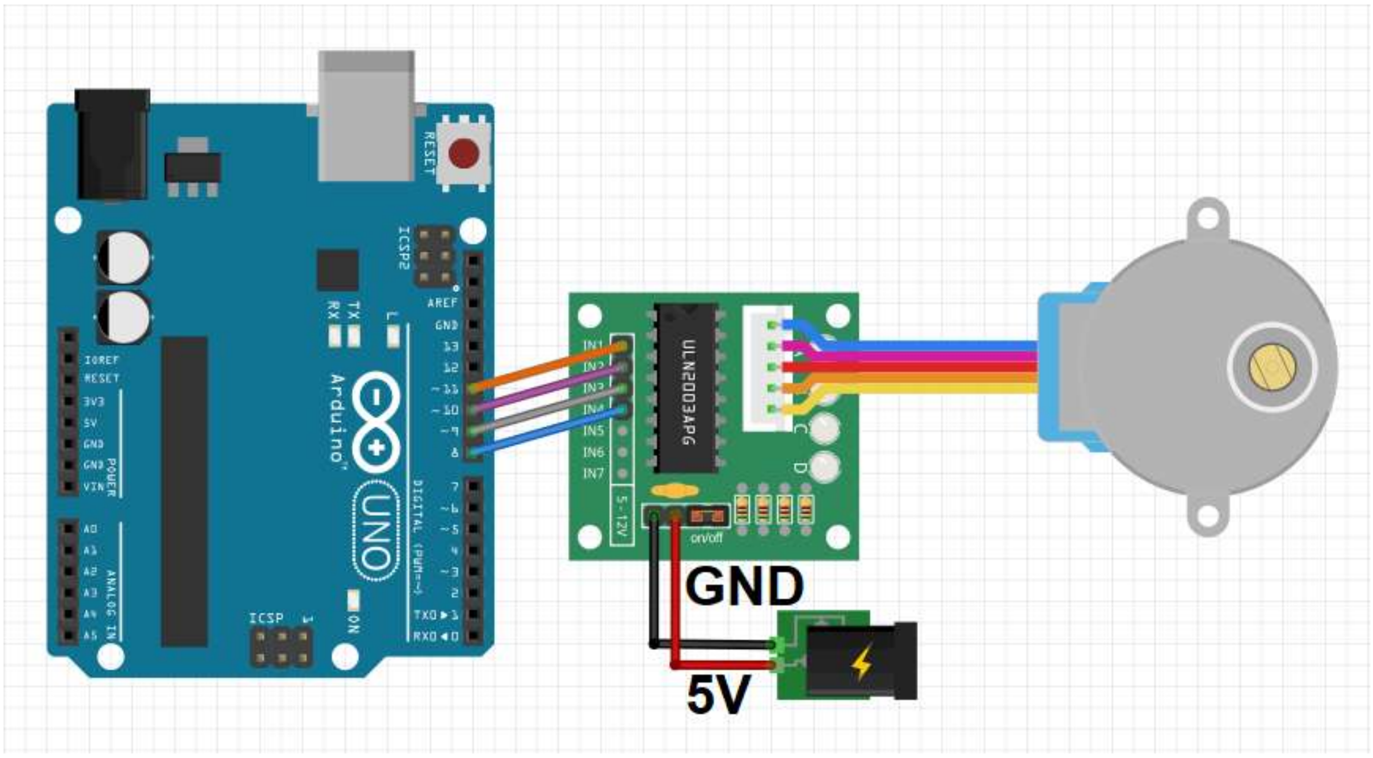
Multiplying the gear ratios:

$$\frac{32}{9} \times \frac{22}{11} \times \frac{26}{9} \times \frac{31}{10} = 63.68395$$

Round 63.68395 up: 64

This gives us a **64:1** gear ratio over all





MOVE FROM A TO B BY A SPECIFIC NUMBER OF STEPS

We will define the speed and acceleration at which we want the stepper motor to rotate and then make it rotate by a specified number of steps, stop, then rotate in the opposite direction in a specified number of steps.

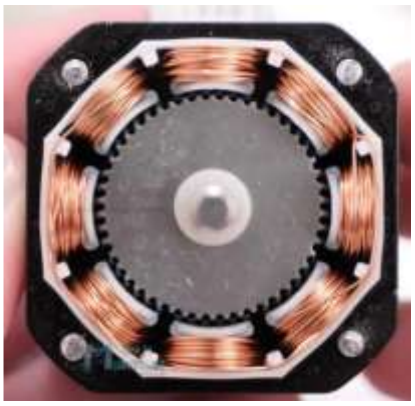
Specifically, we will do 3 rotations in one direction (meaning total steps will be the steps per revolution multiplied by 3 revolutions), stop and complete 3 rotations in the opposite direction.

CODE:

```
//Arduino Code - Move from A to B by a specific number of steps with set speed and acceleration

#include <AccelStepper.h> //Include the AccelStepper library

// Define the motor pins:
#define MP1  8 // IN1 on the ULN2003
#define MP2  9 // IN2 on the ULN2003
#define MP3  10 // IN3 on the ULN2003
#define MP4  11 // IN4 on the ULN2003

#define MotorInterfaceType 8 // Define the interface type as 8 = 4 wires * step factor (2 for half step)

//Define the pin sequence (IN1-IN3-IN2-IN4)
AccelStepper stepper = AccelStepper(MotorInterfaceType, MP1, MP3, MP2, MP4);

const int SPR = 2048;//Steps per revolution

void setup() {
  stepper.setMaxSpeed(1000);   //Set the maximum motor speed in steps per second
  stepper.setAcceleration(200); //Set the maximum acceleration in steps per second^2
}

void loop() {
  stepper.moveTo(3*SPR); //Set the target motor position (i.e. turn motor for 3 full revolutions)
  stepper.runToPosition(); // Run the motor to the target position

  delay(1000);

  stepper.moveTo(-3*SPR);//Same as above: Set the target motor position (i.e. turn motor for 3 full revolutions)
  stepper.runToPosition(); // Run the motor to the target position

  delay(1000);
}
```

MOTOR SPECIFICATIONS

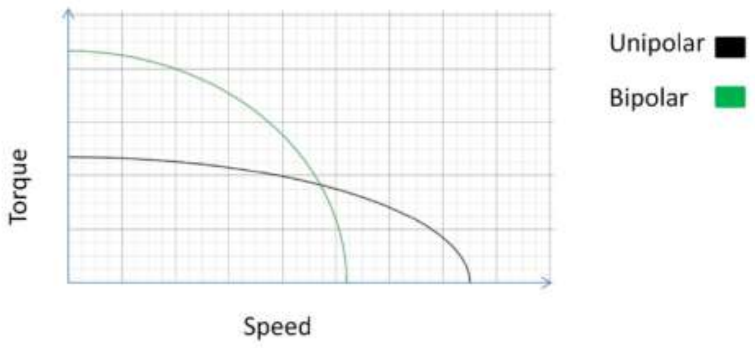
Datasheets normally quote the coil current, coil resistance, nominal voltage and holding torque and steps per revolution. For example, for this motor the values are 1 Amp, 2.7 Ohms, 2.7volts, 1.4Kg-cm and 200 steps/rev.



The nominal voltage is irrelevant for all practical purposes. The important figure is the rated current.

The rated current is normally the current per-coil and when currents are quoted for stepper motor driver boards that is normally also a per-coil figure.

The holding torque is the torque available to resist rotation while the motor is stationary.
The available torque will decline as speed increases.
Some manufactures provide graphs showing how the torque varies with speed.



OPERATING VOLTAGE

Stepper motors are very different from regular DC motors.

With a DC motor you control the current in order to control the speed of the motor.
The usual way to control the current is to vary the voltage - perhaps using the Arduino analogWrite() function to control a ***Pulse Width Modulated*** power supply to the motor.

Stepper motors pretty much draw their full current all the time, even when they are stationary - that is how they resist being moved from their present position. This means they are very inefficient.

For all practical purposes the nominal voltage of a stepper motor is irrelevant.

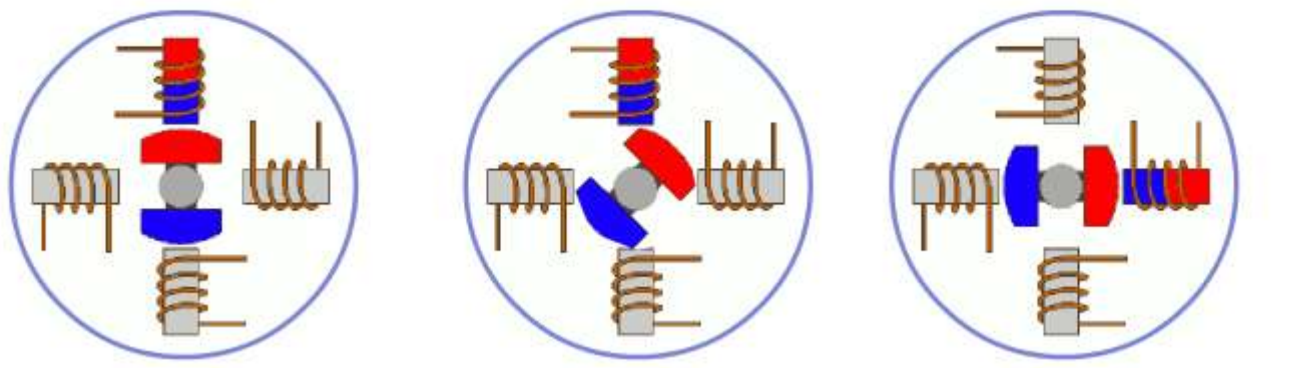
It is the voltage which would drive the rated current through the coil when the motor is stationary based on Ohms law e.g. 2.7v = 1A * 2.7 Ohms.

However, as soon as the motor starts moving the combination of the inductance of the coils and the back-emf generated by the movement will prevent the nominal voltage from producing the rated current.

For this reason stepper motors are normally driven with a much higher voltage.
This, in turn, means that a specialized stepper motor driver board is needed which can limit the current to whatever the motor can take. If the current is not limited the high voltage would quickly destroy the motor.

STEPPER MOTOR STEPS

Each of those rotations is called a "step", with an integer number of steps making a full rotation. In that way, the motor can be turned by a precise angle.



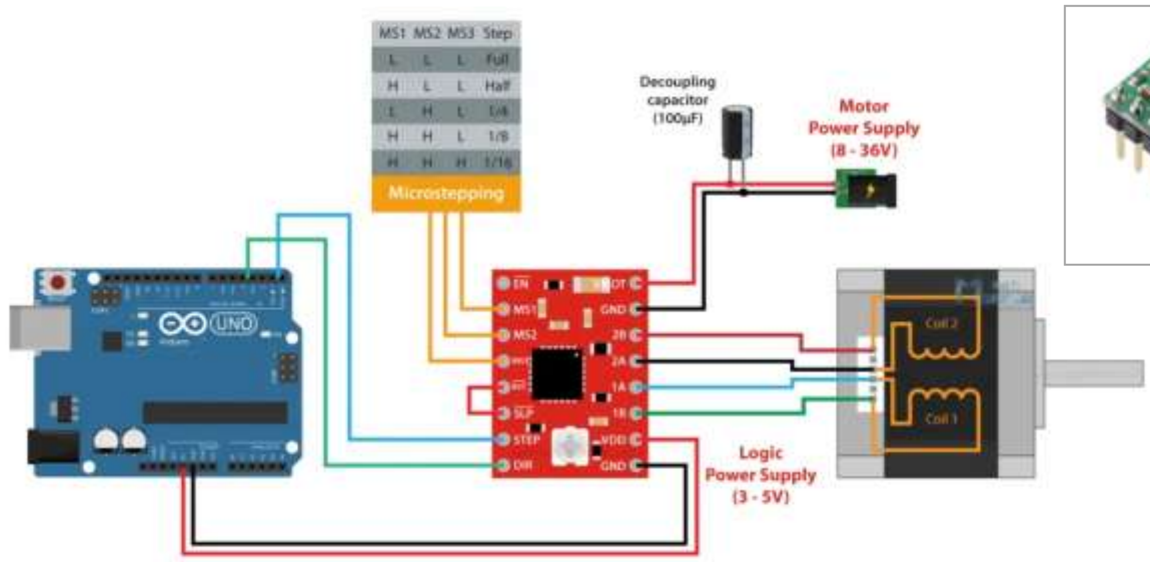
To make the motor shaft turn, first, one electromagnet is given power, which magnetically attracts the gear's teeth.

When the gear's teeth are aligned to the first electromagnet, they are slightly offset from the next electromagnet.

So when the next electromagnet is turned on and the first is turned off, the gear rotates slightly to align with the next one, and from there the process is repeated.

STEPPER MOTOR DRIVER BOARDS

These are specialized components designed to control stepper motors conveniently and efficiently.
The Pololu A4988 is a typical example that is often used with Arduino.



MS1	MS2	MS3	Step
L	L	L	Full
H	L	L	Half
L	H	L	1/4
H	H	L	1/8
H	H	H	1/16

Microstepping
Decoupling capacitor (100µF)
Motor Power Supply (8 - 36V)
Logic Power Supply (3 - 5V)
EN
MS1
MS2
SLP
STEP
DIR
GND
2B
2A
1A
1B
VDD
GND
Coil 2
Coil 1
UNO

Generally speaking specialized stepper motor driver boards only require two connections (plus GND) to the Arduino for step and direction signals.
Normally specialized stepper motor driver boards have the ability to ***limit the current*** in the motor which allows them to drive the motor with a high voltage (up to 35v for the Pololu A4988) for better high speed performance.

Each stepper motor has a maximum current (supplied by the manufacturer) which must not be exceeded. Otherwise, the engine will heat up and be damaged!

All drivers, usually, have the ability to do microstepping: *microstepping is a method of controlling stepper motors, typically used to achieve higher resolution or smoother motion at low speeds.*

The Pololu A4988 can do 1/2, 1/4, 1/8 and 1/16 microsteps. It defaults to full steps: 200 full steps per revolution.

TUNING MOTOR CURRENT WITH A4888

For the A4988 the calculation for the maximum trip current is:

→ I_max= Vref/(8*Rs)

with

- ***Vref****= reference voltage we set on driver*
- ***Rs****= sensing resistors on driver*

With official "Pololus A4988 driver", the sensing resistors are Rs=0.05 ohm, so a Vref of 0.4 should produce a maximum current

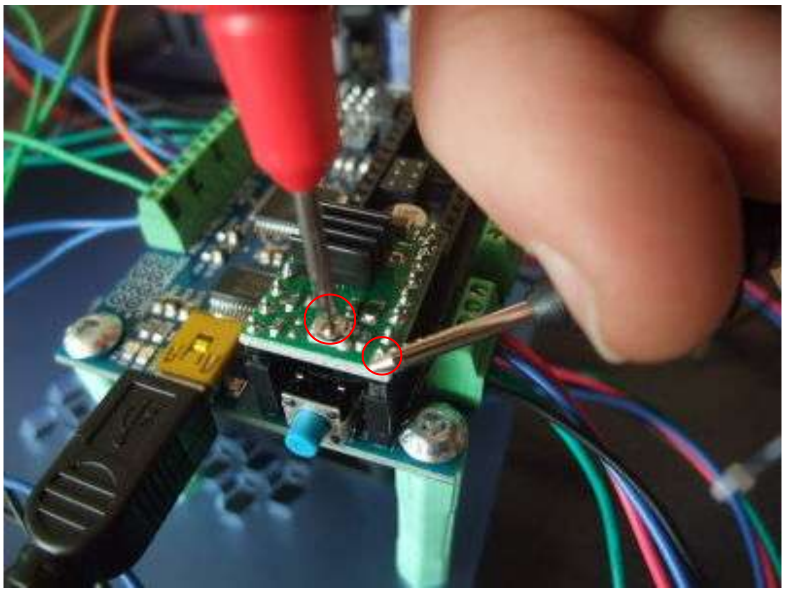
I_max= 0.4/(8*0.05)=1A.

Clockwise increases the current which will make the motor run hotter and counterclockwise reduces it which will cool it down.

As another example, aiming for 50% temperature rise on 1A rated steppers by using max 0.7A, so rearrange it as:

Vref = I_max * 8 * Rs → Vref = 0.7A * 8 * 0.05 = 0.280V

The Vref signal is accessible as the "VREF" pin on the carriers with voltage regulators, as the through-hole via on the carriers without, and also as the wiper on the trim pot itself on both carriers.

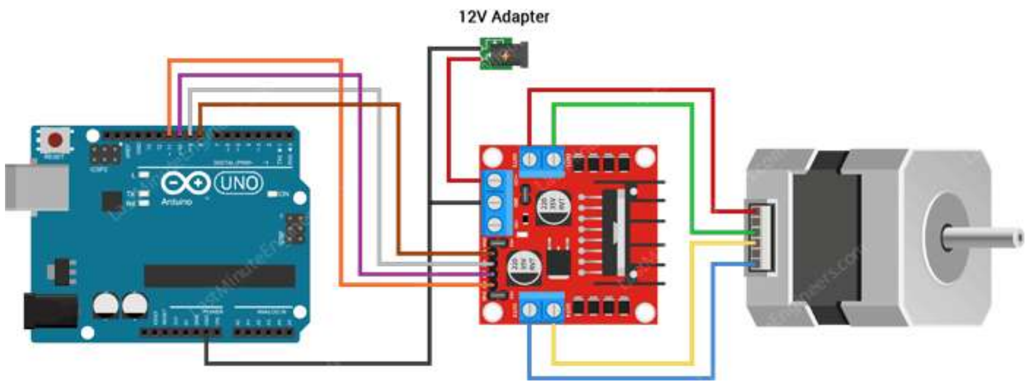


example of a v-ref checking, + probe on the turnpot and - on a ground pin

Note: most "Made in China" A4988 Pololu driver " knock-offs have Rs=0.1 ohm.

H-BRIDGE DRIVER - E.G. LN298

These can be made to control a stepper motor but they are a very poor choice - mainly because they have no method for limiting the current and therefore cannot use high voltages.
They are also more trouble to connect to an Arduino (they require more pins) and more trouble to control with an Arduino (more calculations for the Arduino to do).



The L298N Motor Driver Board is built around the L298 dual full-bridge driver, made by STMicroelectronics.

With this motor driver you can control DC motors, stepper motors, relays, and solenoids. It comes with two separate channels, called A and B, that you can use to drive 2 DC motors, or 1 stepper motor when combined.

The L298N is usually mounted on a (red) breakout board, which makes wiring a lot easier. The breakout board also includes a 78M05 5 V power regulator.

Why is my stepper motor getting HOT?

One thing that is very important to remember is that the L298 does not have an easy way to set a current limit unlike other stepper motor drivers like the A4988. This means that the current draw depends on the relationship between the inductance and resistance (L/R) of the stepper motor that you connect to it.

When the motor draws too much current, you can damage the driver and the motor will get hot!

What this means for you, is that you need to be careful when selecting the stepper motor and power supply to use with this motor driver. Not all stepper motors will work! The L298N operating voltage is between 4.8 and 46 volts (max 35 V when mounted on the breakout board).
Since the driver can supply a maximum of 2 amperes per channel, you need to find a stepper motor that can be used in this voltage range and doesn't exceed the maximum current rating.

Check the datasheet of your stepper motor and look for the voltage/current draw of the motor.
If you can't find the datasheet, you can measure the resistance of one of the windings and use the following formula to get an estimation of the current draw:

$I = U \div R$ or Current draw (A) = Supply voltage (V) ÷ Winding resistance (Ω)

I would try to find a motor that draws less than 2 A at the voltage that you want to use.

CHOOSING A MOTOR AND MOTOR DRIVER

First choose the motor

The important specification is the torque of the motor.
For example for a small NEMA 17 it is 0.59 Nm.
The available useful torque will decline as the speed increases and at no-load maximum speed it will be zero.
Some (probably the more expensive) motor manufactures provide graphs showing how the torque varies with speed.

To figure out what motor you need you will have to measure or estimate the torque required.
It would be a good idea to choose a motor with a good margin of surplus torque.

It is not too difficult to make a rough measurement of the torque required but it is beyond the scope of this note.

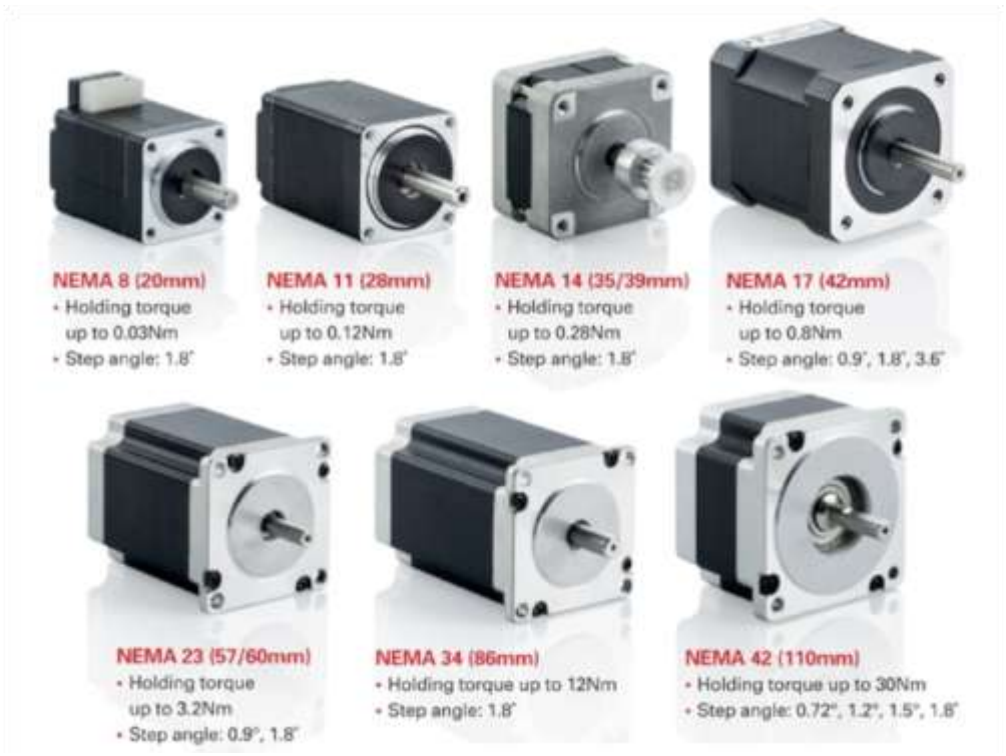
Then choose the stepper motor driver

When you have selected a motor and know what current it requires you can choose a stepper motor driver that can comfortably supply the required current.

You should be aware that the economical single-chip stepper drivers (such as the A4988 and the DRV8825) can only supply about 2 amps. If your motor requires more than that, you will need to get one of the more expensive commercial stepper drivers. However the working principle will be practically identical to the A4988.

NEMA 17 – 23 – 34 – 42

These standards only define the size of the front face of the motor and the location and size of the mounting screw holes. They say nothing about the power of the motor. The 17 is an abbreviation of 1.7 inches.


NEMA 8 (20mm)
• Holding torque up to 0.03Nm
• Step angle: 1.8°
NEMA 11 (28mm)
• Holding torque up to 0.12Nm
• Step angle: 1.8°
NEMA 14 (35/39mm)
• Holding torque up to 0.28Nm
• Step angle: 1.8°
NEMA 17 (42mm)
• Holding torque up to 0.8Nm
• Step angle: 0.9°, 1.8°, 3.6°
NEMA 23 (57/60mm)
• Holding torque up to 3.2Nm
• Step angle: 0.9°, 1.8°
NEMA 34 (86mm)
• Holding torque up to 12Nm
• Step angle: 1.8°
NEMA 42 (110mm)
• Holding torque up to 30Nm
• Step angle: 0.72°, 1.2°, 1.5°, 1.8°

MICROSTEPS

Most (but certainly not all) stepper motors do 200 full steps per revolution.
By appropriately managing the current in the coils it is possible to make the motor move in smaller steps.
The Pololu A4988 can make the motor move in 1/16th steps per revolution.

It defaults to full steps: 200 full steps per revolution.

The main advantage of microstepping is to reduce the roughness of the motion.
The only fully accurate positions are the full-step positions.
The motor will not be able to hold a stationary position at one of the intermediate positions with the same position accuracy or with the same holding torque as at the full step positions.

Generally speaking when high speeds are required full steps should be used.

It is possible with most drivers including the Pololu A4988 to use the Arduino program to change the microstep setting.

STEPPER MOTOR SPEED

By comparison with regular DC motors stepper motors are very slow devices.

Typical speeds might be 1000 to 4000 steps per second and for a 200 step motor that would represent 5 to 20 rps (300 to 1200 rpm).

Generally speaking the motors with low coil resistance and high currents (and low nominal voltages) will be most suitable for higher speeds. A high voltage will also be needed for high speed.

ACCELERATION

If the stepper motor is required to move a heavy load it will normally be necessary to start the movement slowly (as with any motor) and accelerate to the desired speed and, equally, to decelerate when it is necessary to stop.

This is quite different from a DC motor which will accelerate and decelerate automatically.

If you try to start or stop a stepper motor too quickly it will simply skip steps with no damage to motor.
However The Arduino has no means to know whether or how many steps have been missed and all of the position control will be lost.

For this reason, in particular, it is essential to choose a motor with sufficient torque for the job and to use acceleration and deceleration when necessary.

POSITION FEEDBACK

Stepper motors do not have the ability to tell the Arduino what position they are at, nor do they have the ability (like a servo) to go to a particular position. All they can do is move N steps from where they are now.

If it is essential to have position feedback a rotary encoder can be attached to the motor shaft - but that is beyond the scope of this notes.

INITIAL POSITION

When it starts up the Arduino has no means of knowing where the stepper motor is positioned - for example somebody might have moved it manually when the power was off.

The usual way to establish a datum for counting steps is with a limit switch.
At startup the Arduino will move the motor until it triggers the switch.
The Arduino will then regard that step position as step zero for the purpose of future position keeping.

STEPPER MOTOR APPLICATIONS

Step motors are used every day in both industrial and commercial applications because of their low cost, high reliability, high torque at low speeds. For making more advanced technologies we use stepper motors rather than simple motors.

Computer-controlled stepper motors are a type of motion-control positioning system.

Step Motors are typically digitally controlled motors, for use in holding or precise positioning applications.

These motors are used at

- -Industrial Machines : CNC Machines, Milling Machines, Laser Cutters etc,
- -Computer Technology: CD-Roms, DVD Players, Floppy Disk Drives, Scanners etc.
- -Printing : Printers, Plotters, 3D Printers etc.
- -Intelligent Lighting Systems : Lasers, Optical Devices, Mirror Mounts

ARDUINO LIBRARIES

When using an Arduino with a specialized stepper motor driver board such as the Pololu A4988 there is little to be gained from using an Arduino library unless you need the acceleration feature of the AccelStepper library.

The AccelStepper library provides an object-oriented interface for 2, 3 or 4 pin stepper motors and motor drivers.

The standard Arduino IDE includes the Stepper library (http://arduino.cc/en/Reference/Stepper) for stepper motors. It is perfectly adequate for simple, single motor applications.

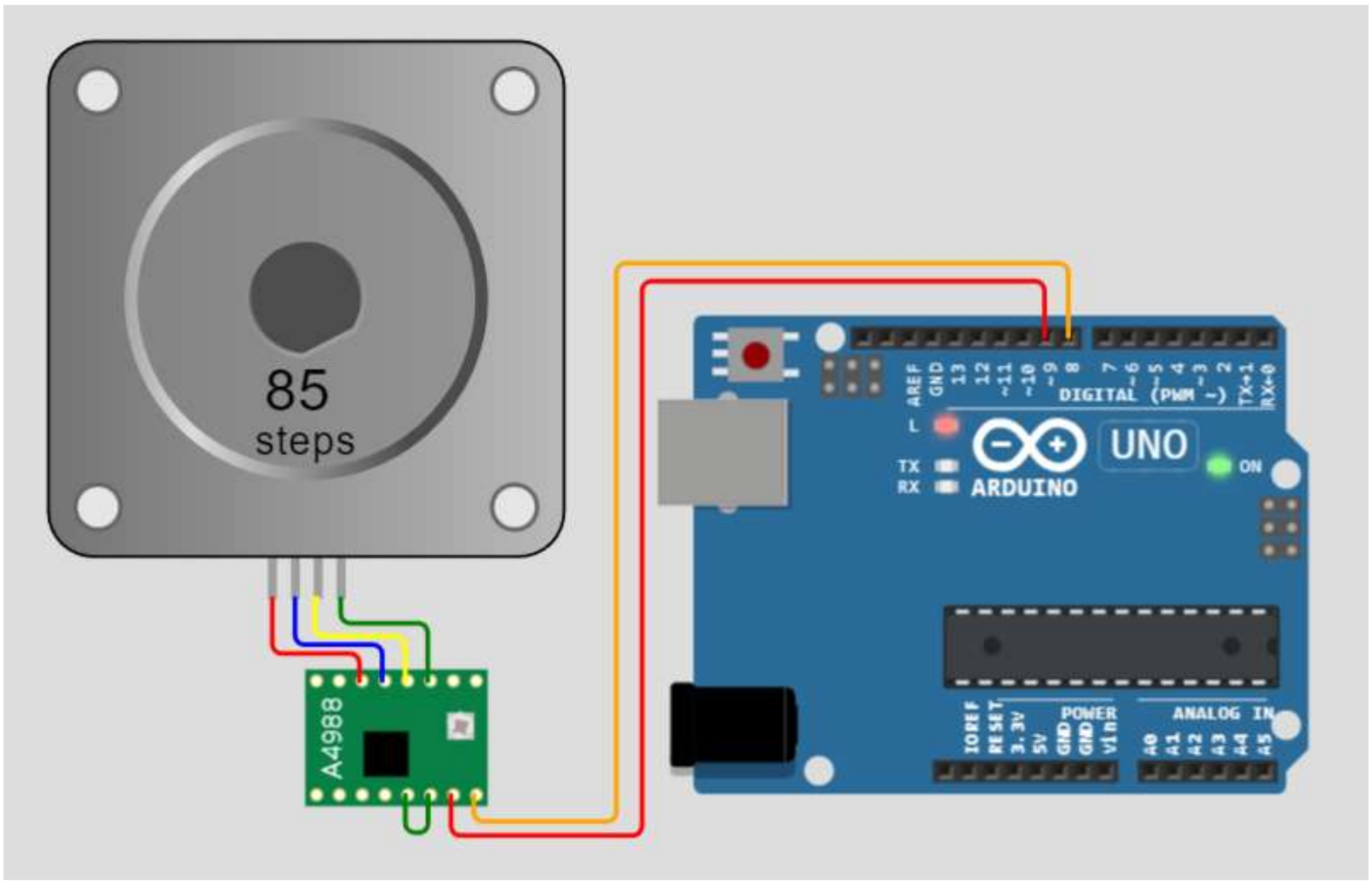
AccelStepper significantly improves on the standard Arduino Stepper library in several ways:

- Supports acceleration and deceleration
- Supports multiple simultaneous steppers, with independent concurrent stepping on each stepper
- Most API functions never delay() or block (unless otherwise stated)
- Supports 2, 3 and 4 wire steppers, plus 3 and 4 wire half steppers.
- Supports stepper drivers such as the Sparkfun EasyDriver (based on 3967 driver chip)
- Very slow speeds are supported

ARDUINO CODE SAMPLE FOR STEPPER MOTOR + A4988 DRIVER

The code is intended as a first step to getting your motor working. It shows how easy it is to control a motor without a library when a specialized stepper motor driver such as the Pololu A4988 is used.

Simulate the circuit assigned in wokwi.



```
// test a stepper motor with a Pololu A4988 driver board, onboard led will flash at each step
// this version uses delay() to manage timing

byte directionPin = 8;
byte stepPin = 9;
int numberOfSteps = 200;
byte ledPin = 13;
int pulseWidthMicros = 20;  // microsecondo
int millisbetweenSteps = 20; // milliseconds - or try 100 for slower steps

void setup() {

  Serial.begin(9600);

  Serial.println("Starting StepperTest");

  digitalWrite(ledPin, LOW);

  delay(2000);

  pinMode(directionPin, OUTPUT);

  pinMode(stepPin, OUTPUT);

  pinMode(ledPin, OUTPUT);

  digitalWrite(directionPin, HIGH);

  for(int n = 0; n < numberOfSteps; n++) {

    digitalWrite(stepPin, HIGH);

    delayMicroseconds(pulseWidthMicros);

    digitalWrite(stepPin, LOW);

    delay(millisbetweenSteps);

    digitalWrite(ledPin, !digitalRead(ledPin));

  }


  delay(2000);

  digitalWrite(directionPin, LOW);

  for(int n = 0; n < numberOfSteps; n++) {

    digitalWrite(stepPin, HIGH);

    // delayMicroseconds(pulseWidthMicros); // probably not needed

    digitalWrite(stepPin, LOW);

    delay(millisbetweenSteps);

    digitalWrite(ledPin, !digitalRead(ledPin));

  }

}


void loop() { }
```

Try adding a button that starts motor movement only if it's pressed. Modify the code accordingly.

Add new wokwi scheme below.

Arduino code.

CLIL PARTE 2 STEPPER MOTOR

Read the attached code and then create the circuit capable of realizing it in the wokwi simulator.

```
// test a stepper motor with a Pololu A4988 driver board or equivalent
// this version uses millis() to manage timing rather than delay()
// and the movement is determined by a pair of momentary push switches (push button)
// press one and it turns CW, press the other and it turns CCW

byte directionPin = 9;
byte stepPin = 8;
byte buttonCWpin = 10;
byte buttonCCWpin = 11;
boolean buttonCWpressed = false;
boolean buttonCCWpressed = false;
byte ledPin = 13;

unsigned long curMillis;
unsigned long prevStepMillis = 0;
unsigned long millisBetweenSteps = 25; // milliseconds

void setup() {
     Serial.begin(9600);
     Serial.println("Starting Stepper Demo with millis()");
     pinMode(directionPin, OUTPUT);
     pinMode(stepPin, OUTPUT);
     pinMode(ledPin, OUTPUT);
     pinMode(buttonCWpin, INPUT_PULLUP);
     pinMode(buttonCCWpin, INPUT_PULLUP);
}

void loop() {
    curMillis = millis();
    readButtons();
    actOnButtons();
}

void readButtons() {
    buttonCCWpressed = false;
    buttonCWpressed = false;

    if (digitalRead(buttonCWpin) == LOW) {
        buttonCWpressed = true;
    }
    if (digitalRead(buttonCCWpin) == LOW) {
        buttonCCWpressed = true;
    }
}

void actOnButtons() {
    if (buttonCWpressed == true) {
        digitalWrite(directionPin, LOW);
        singleStep();
    }
    if (buttonCCWpressed == true) {
        digitalWrite(directionPin, HIGH);
        singleStep();
    }
}

void singleStep() {
    if (curMillis - prevStepMillis >= millisBetweenSteps) {
            // next 2 lines changed 28 Nov 2018
        //prevStepMillis += millisBetweenSteps;
        prevStepMillis = curMillis;
        digitalWrite(stepPin, HIGH);
        digitalWrite(stepPin, LOW);
    }
}
```

Add the WOKWY schema function in the box below and any explanations.

BRIEFLY ANSWER THE FOLLOWING QUESTIONS

1. What are the main parts of a stepper motor?
2. Why are powered stepper motors hot even if the shaft is not rotating?
3. What are the main differences between bipolar and unipolar steppers?
4. What type of stepper motor provides the most torque at the same speed?
5. What type of engine is the 28BYJ-48?
6. How much current does a small stepper motor typically consume?
7. What is the typical working voltage of stepper motors?
8. What is a "driver"?
9. What is meant by "microstepping"?
10. How do you choose a stepper motor?
11. What does the abbreviation NEMA 23 mean?
12. With a 1/16 microsteps what angular precision is obtained?
13. What is the maximum speed of a stepper motor?
14. What torque does a small NEMA 17 motor typically have?
15. What does the Arduino code in “sample 1” do?

CLIL PART 3
MOTORS AND SELECTING THE RIGHT ONE

INTRODUCTION

At any given moment, you are near at least one or two types of motors.
From the vibration motor in your cell phone, to the fans and CD drive in your favorite gaming system, motors are all around us. Motors provide a way for our devices to interact with us and the environment.

With a myriad of applications for motors, the design and operation of them can vary.

WHAT YOU WILL LEARN

In this tutorial we'll cover some of these basic motor types and uses:

- DC Brush Motors
- Brushless Motors
- Stepper Motors
- Linear Motors

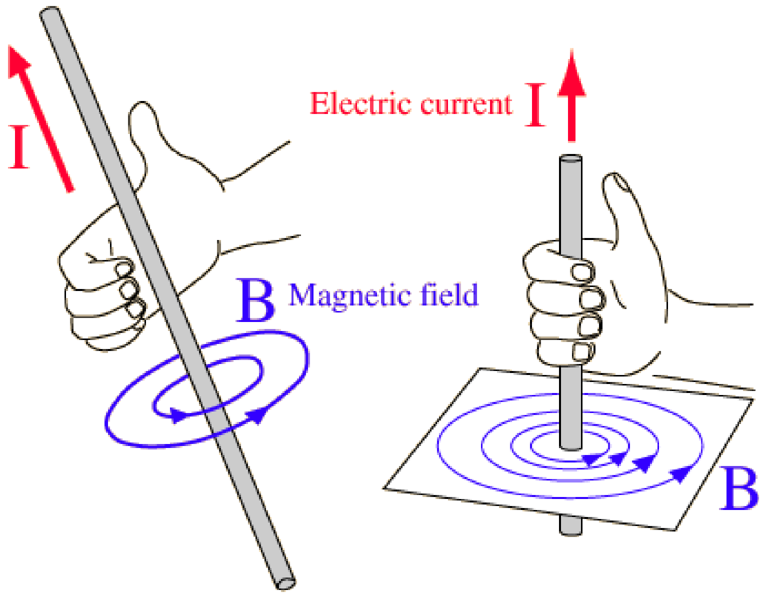
WHAT MAKES A MOTOR MOVE?

The most vague and simple answer is magnetism! Ok, now let's take this simple force and turn it into a super car!

To keep things simple, we will need to look at some concepts through the lens of the thought experiment.
Some liberties will be taken, but if you want to get down and dirty with the details, you can consult Dr. Griffiths.
For our thought experiment, we are going to state that a magnetic field is produced by a moving electron *i.e. current*.
While this creates a classical model for us to use, things break down when we reach the atomic level.
To understand the atomic level of magnetism more, Griffiths explains that in another book...

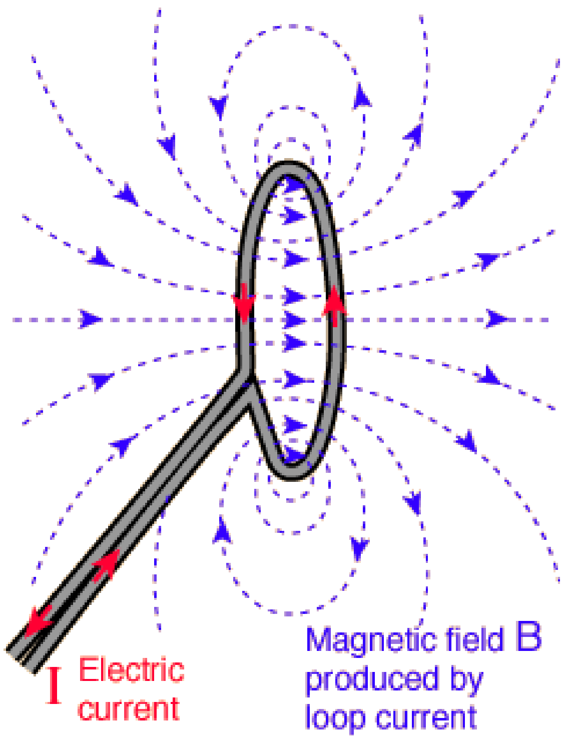
ELECTROMAGNETISM

To create a magnet or magnetic field, we are going to have to look at how they are generated.
The relationship between current and magnetics field behave according to the right-hand rule. As current passes through a wire, a magnetic field forms around the wire in the direction of your fingers as they wrap around it.
This is a simplification of Ampère's force law as it acts on a current carrying wire.
Now, if you place that same wire in a pre-existing magnetic field, you can generate a force.
This force is referred to as the Lorentz force.



The right-hand rule shows the direction of the magnetic field in relation to the current path.

If the current is increased, the strength of the magnetic field is strengthened. Though, to do something useful with the field, it would take incredible amounts of current. Furthermore, the wire delivering the current would be carrying the same magnetic strength, thus creating uncontrolled fields.
By bending the wire into a loop, a directed and concentrated field can be created.

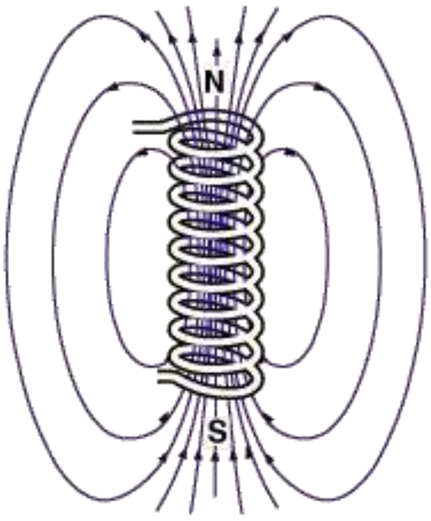


The field has not changed. By bending the wire into a loop, field directions are simply aligned.

ELECTROMAGNETS

By looping wire and passing a current, an electromagnet is created. If one loop of wire can concentrate the field, what can you do with more? How about a few **hundred** more!
The more loops you add to the circuit, the stronger the field becomes for a given current. If that's the case, why don't we see **thousands **, if not **millions**, of windings in motors and electromagnets?
Well, the longer the wire the higher resistance it has. Ohm's law (V = I*R) says to maintain the same current as resistance increases, voltage must increase. In some cases it makes sense to use higher voltages; in other cases some use larger wire with less resistance.

Using larger wire is more costly and is generally more difficult to work with. These are factors that have to be weighed when designing a motor.



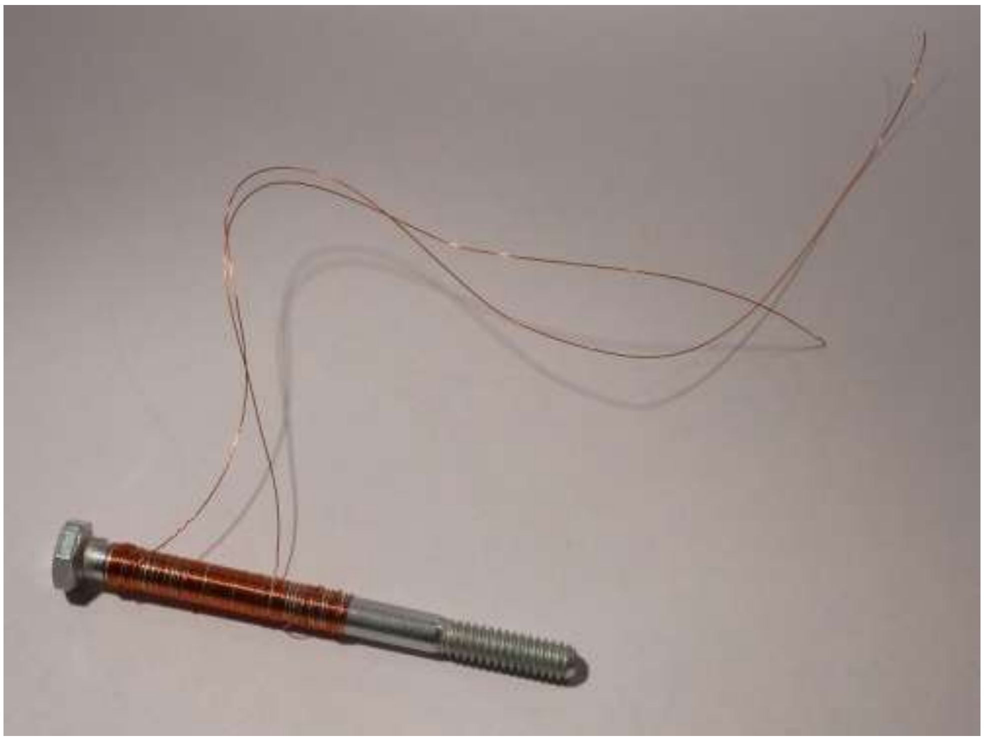
An energized electromagnet producing a magnetic field.

EXPERIMENT TIME

To create your own electromagnet, simply find a bolt (or other round steel object), some magnet wire (30-22 gauge works fine), and a battery.

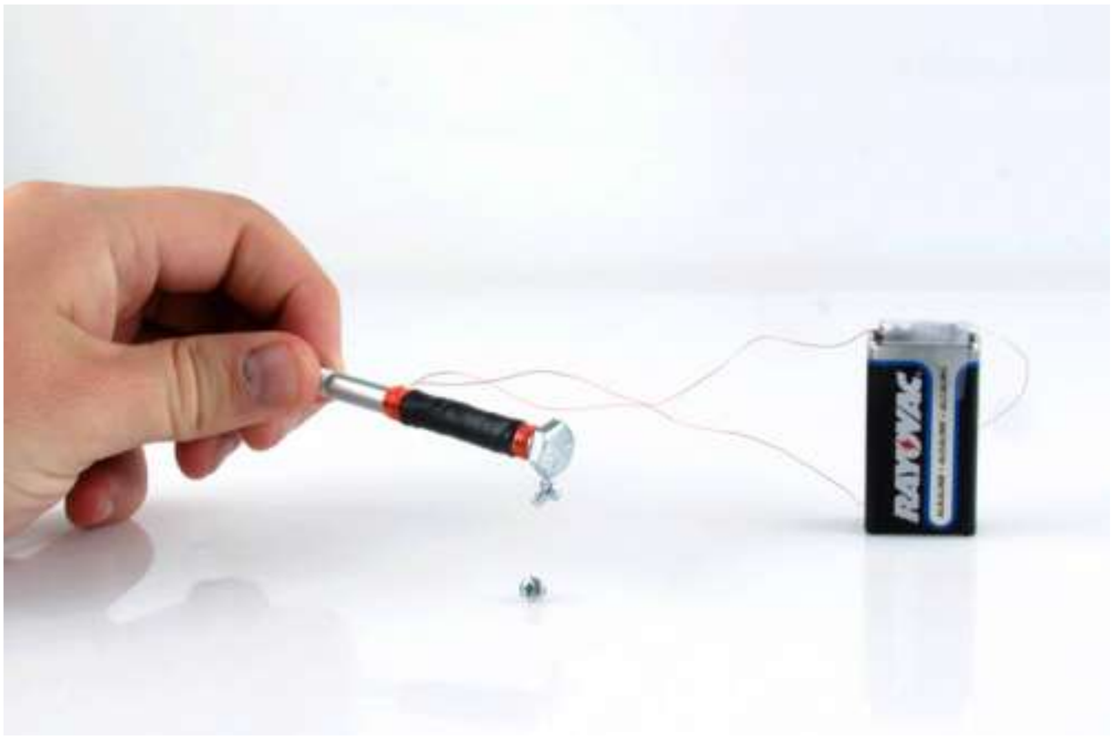
*Note: Lithium Batteries are **NOT** recomended for this experiment.*

Wrap between 75-100 turns of wire around the steel. Using a steel center further concentrates the magnetic field, increasing its effective strength. We will go over why this is happens in the next section.



A bit of heat shrink or tape can help keep the coils on the steel center.

Now, using sand paper, remove the insulation from the ends of the wires, and connect each wire to each terminal of the battery. Congratulations! You have built the first component of a motor!
To test the strength of your electromagnet, try to pick up paper clips or other small steel objects.

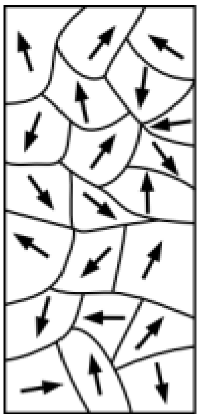


It's not magic, it's SCIENCE!!!

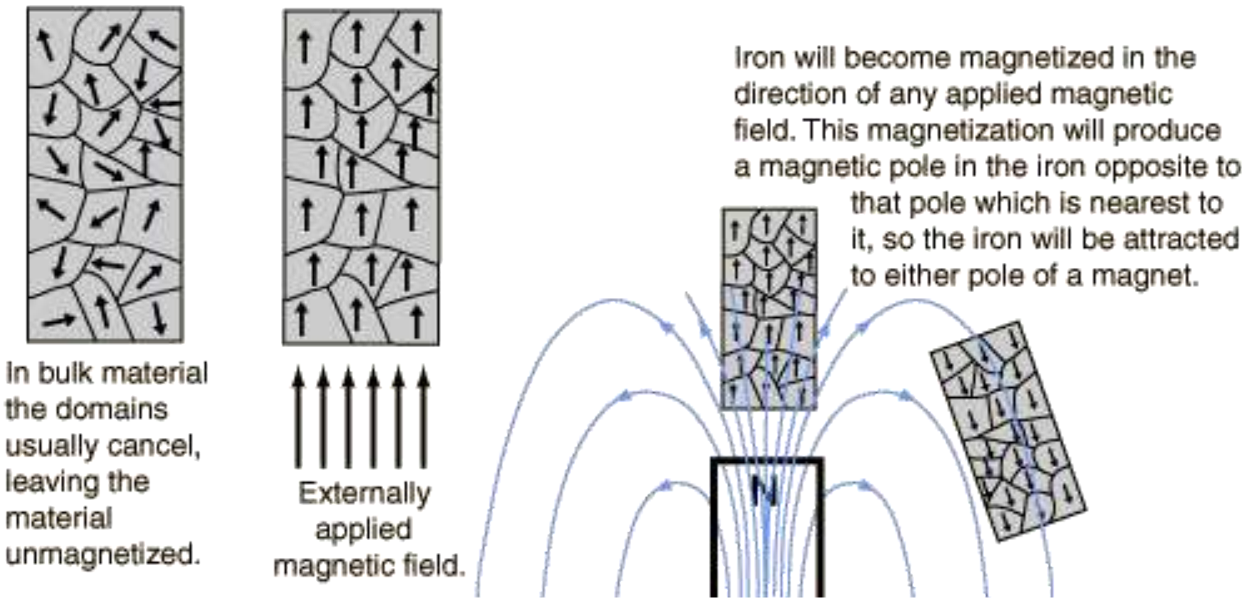
FERROMAGNETISM

Looking back to the beginning of our thought experiment, magnetic fields may only be produced by a current. Taking the definition of current as a flow of electrons, electrons orbiting an atom should create a current and thus a magnetic field! If every atom has electrons is everything magnetic?
YES! All matter, including frogs, can express magnetic properties when given enough energy. But not all magnetism is created equally. The reason I can pick up screws with a refriderator magnent and not a frog is the difference between ferromagnetism and paramagnetism.
The way to differentiate the two (and a few more types) is through the study of quantum mechanics.

Ferromagnetism will be our focus, since it is the strongest phenomenon and is what we have the most experience with. Further, to relieve us from having to understanding this at the quantum level, we are going to accept that atoms of ferromagnetic materials *tend* to align their magnetic fields with their neighbors. Though they tend to align, inconsistencies in material and other factors like crystaline structure create magnetic domains.



When magnetic domains are aligned in a random order, neighboring fields cancel each other out resulting in a non-magnetized material. Once in the presence of an strong external field it is possible to re-align these domains. By aligning these domains, the overall field strengthens, creating a magnet!

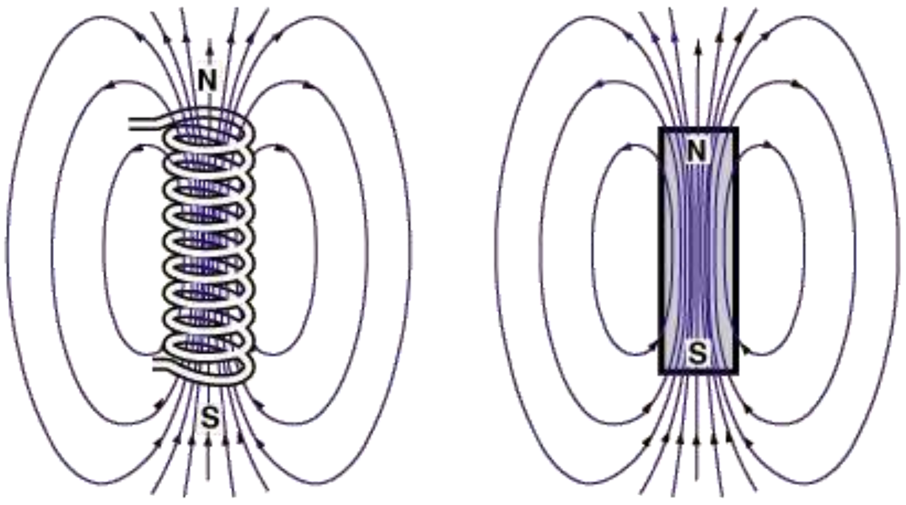

Iron will become magnetized in the direction of any applied magnetic field. This magnetization will produce a magnetic pole in the iron opposite to that pole which is nearest to it, so the iron will be attracted to either pole of a magnet.
N
In bulk material the domains usually cancel, leaving the material unmagnetized.
Externally applied magnetic field.

(Credit: HyperPhysics)

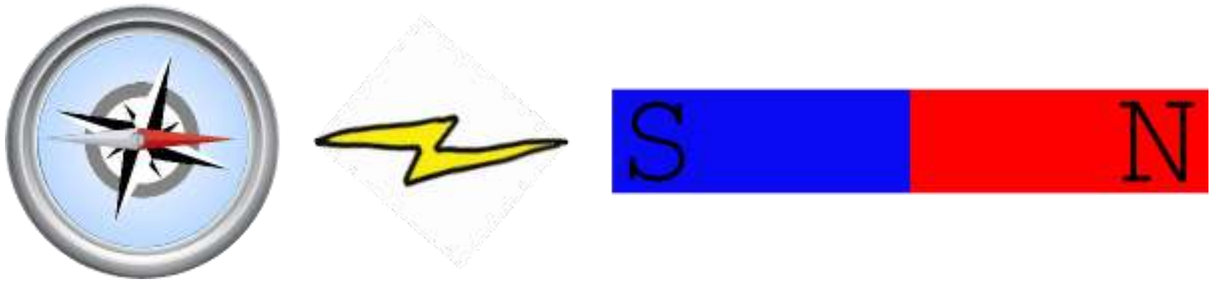
This re-alignment can be permanent depending on the strength of the field. This is great because we'll need these in the next section.

PERMANENT MAGNETS

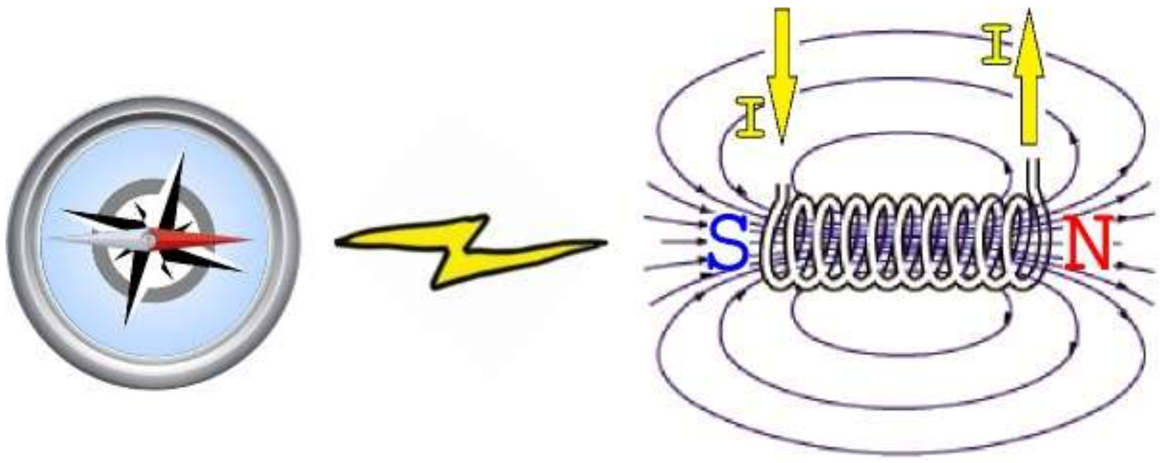
Permanent magnets behave in the same way as electromagnets. The only difference is, well, they are permanent.



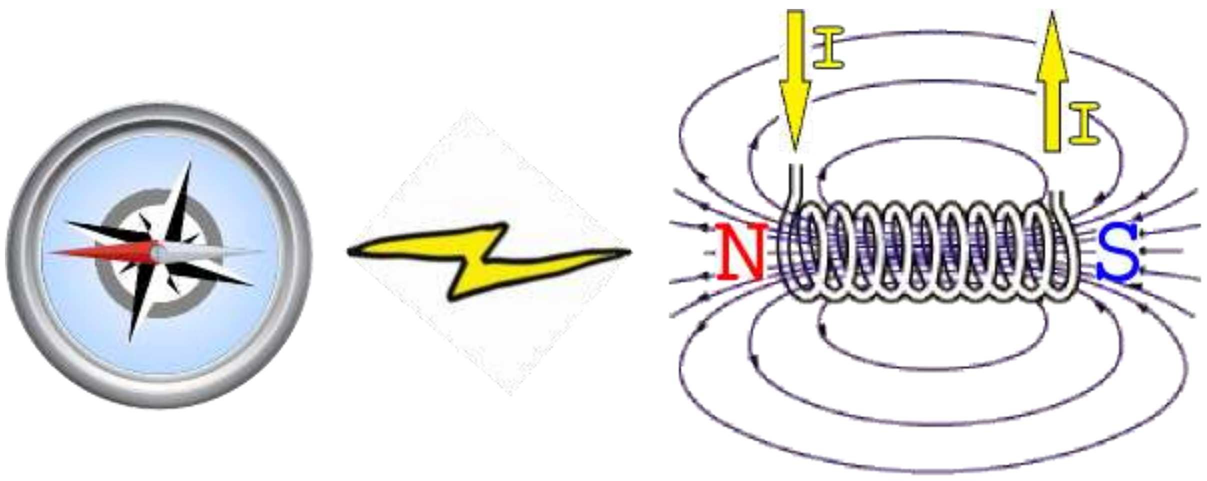
In all drawings, arrows will be pointing away from the north pole and towards the south pole. Another convention is to use the color red to represent north and blue to represent south. To identify a magnets polarity, you can use a compass. Since opposites attract, the needle will point north to the south pole of the magnet.



You can perform the same experiment with an electromagnet to determine polarity.



If you reverse the flow of current, you can see how an electromagnet can reverse its poles.



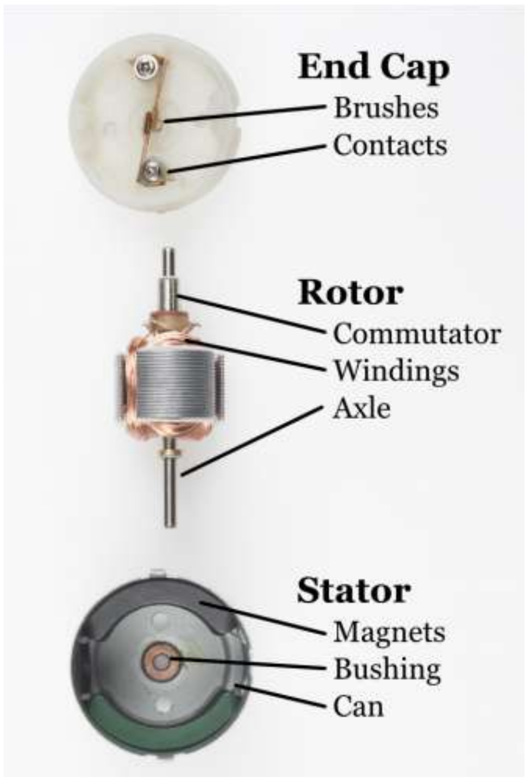
This is a key principle for building motors! Now, let's look at some different motors and how they use magnets and electromagnets.

DC BRUSH MOTORS - THE CLASSIC

The DC brush motor is one of the simplest motors in use today.
You can find these motors just about anywhere. They are in household appliances, toys, and automobiles.
Being simple to construct and control, these motors are the go-to solution for professionals and hobbyists alike.

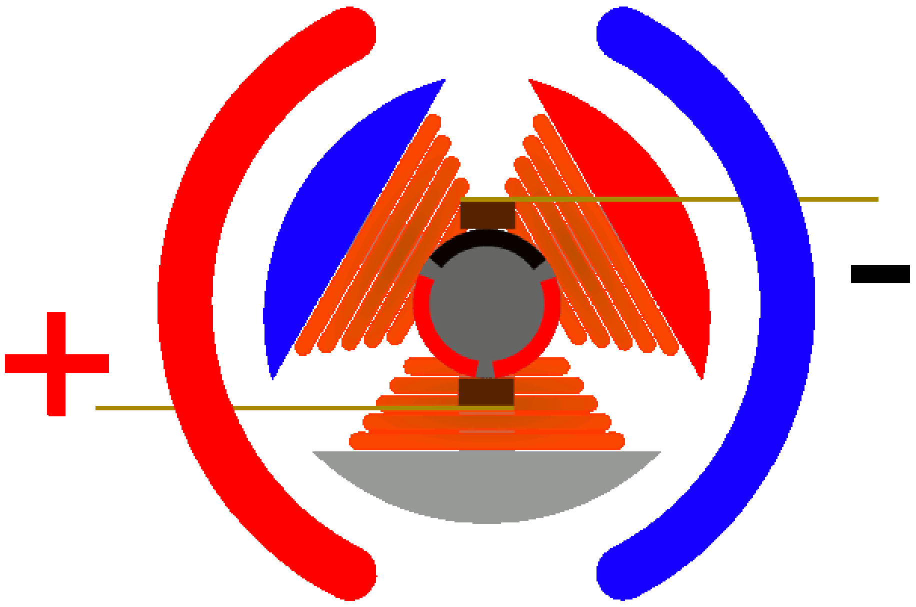
THE ANATOMY OF A BRUSH MOTOR

To better understand how one works, let's start by tearing down a simple hobby motor.
As you can see, they are simple in construction, comprising of a few key components.



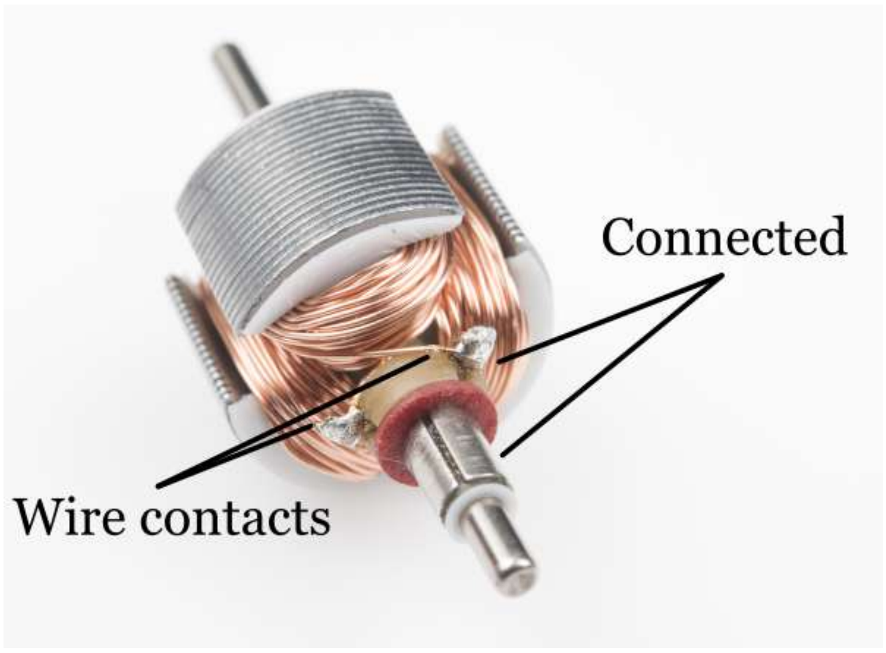
- Brushes - Delivers power from the contacts to the armature through the commutator
- Contacts - Brings power from the controller to the brushes
- Commutator - Delivers power to the appropriate set of windings as the armature rotates
- Windings - Converts electricity to a magnetic field that drives the axle
- Axle - Transfers the mechanical power of the motor to the user application
- Magnets - Provide a magnetic field for the windings to attract and repel
- Bushing - Minimizes friction for the axle
- Can - Provides a mechanical casing for the motor

THEORY OF OPERATION



As the windings are energized, they attract to the magnets located around the motor.
This rotates the motor until the brushes make contact with a new set of commutator contacts.
This new contact energizes a new set of windings and starts the process again.
To reverse the direction of the motor, simply reverse the polarity on the motor contacts.

Sparks inside a brush motor are produced by the brush jumping to the next contact.
Each wire of a coil is connected to the two closest commutator contacts.



An odd number of windings is always used to prevent the motor from getting locked into a steady state. Larger motors also use more sets of windings to help eliminate "cogging," thus providing smooth control at low revolutions per minute (RPMs).
Cogging can be demonstrated by rotating the motor axle by hand. You will feel "bumps" in the motion where the magnets are closest to the exposed stator. Cogging can be eliminated with a few tricks in design, but the most prevalent is removing the stator all together.
These types of motors are referred to as ironless or coreless motors.

PROS

- Simple to control
- Excellent torque at low RPM
- Inexpensive and mass produced

CONS

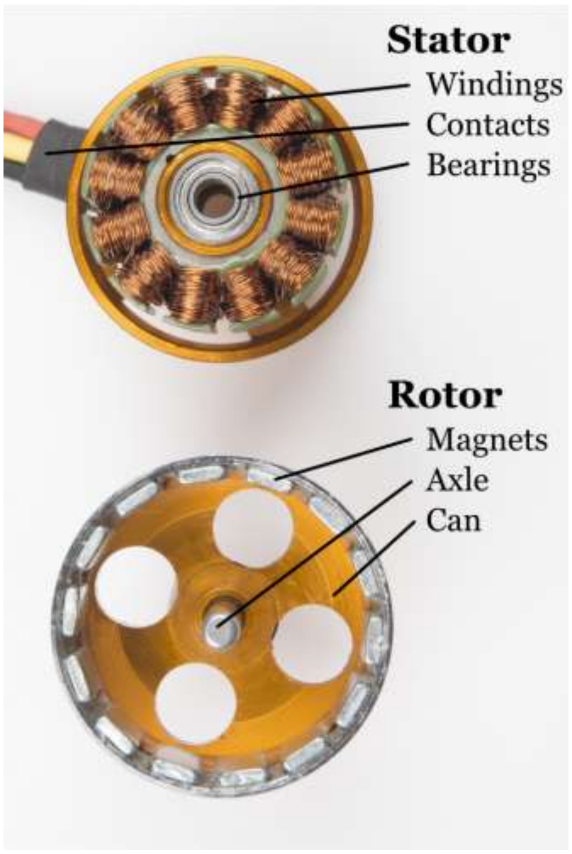
- Brushes can wear out over time
- Brush arcing can generate electromagnetic noise
- Usually limited in speed due to brush heating

BRUSHLESS MOTORS - MORE POWER!

Brushless motors are taking over! Ok, maybe that was an overstatement. However, brushless motors have begun to dominate the hobby markets between aircraft and ground vehicles.
Controlling these motors had been a hurdle up until microcontrollers became cheap and powerful enough to handle the task.
There is still work being done to develop faster and more efficient controllers to unlock their amazing potential. Without brushes to fail, these motors deliver more power and can do so silently.
Most high-end appliances and vehicles are moving to brushless systems. One notable example is the Tesla Model S.

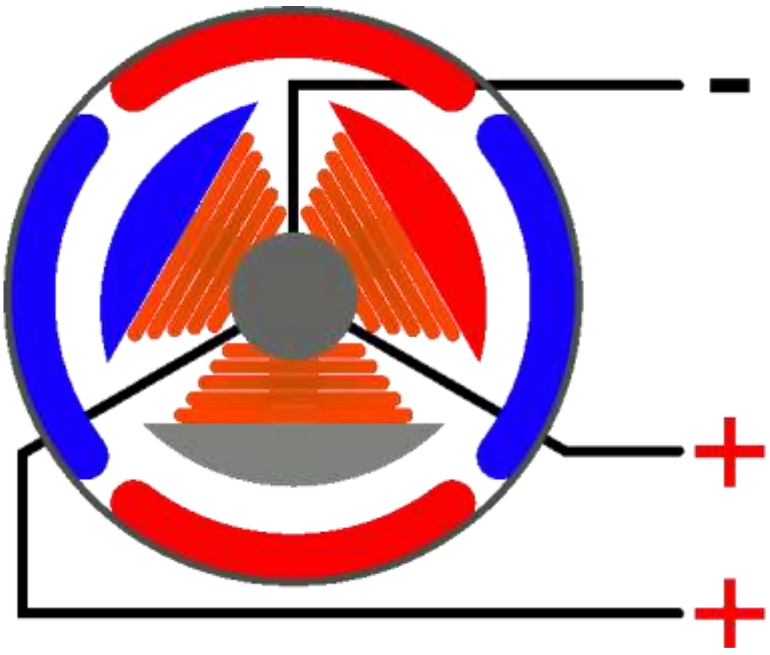
THE ANATOMY OF A BRUSHLESS MOTOR

To better understand how one works, let's start by tearing down a simple brushless motor.
These are commonly found on remote control airplanes and helicopters.



- Windings - Converts electricity to a magnetic field that drives the rotor
- Contacts - Brings power from the controller to the windings
- Bearings - Minimizes friction for the axle
- Magnets - Provide a magnetic field for the windings to attract and repel
- Axle - Transfers the mechanical power of the motor to the user application

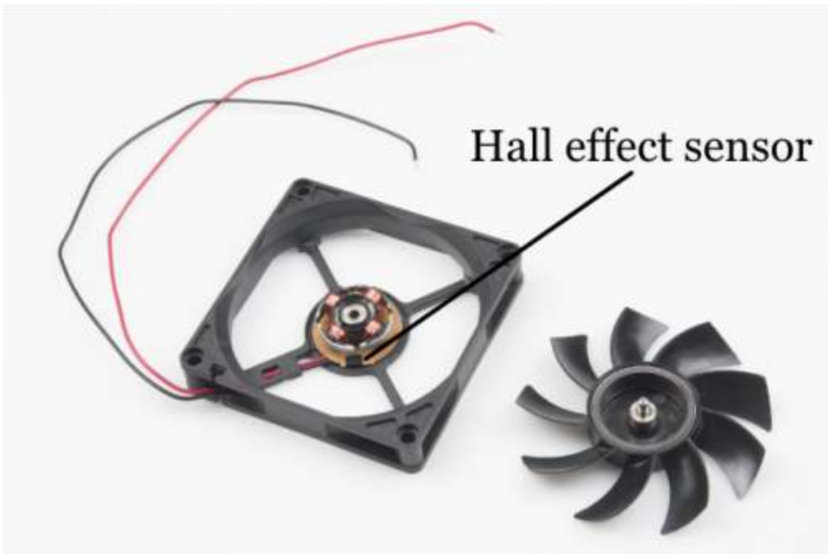
THEORY OF OPERATION



The mechanics of a brushless motor are incredibly simple.
The only moving part is the the rotor, which contains the magnets. Where things become complicated is orchestrating the sequence of energizing windings.
The polarity of each winding is controlled by the direction of current flow. The animation demonstrates a simple pattern that controllers would follow. Alternating current changes the polarity, giving each winding a "push/pull" effect.
The trick is keeping this pattern in sync with the speed of the rotor.

There are two (widely used) ways this can be accomplished. Most hobby controllers measure the voltage produced (back EMI) on the un-energized winding.
This method is very reliable in high velocity operation. As the motor rotates slower, the voltage produced becomes more difficult to measure and more errors are induced.

Newer hobby controllers and many industrial controllers utilize Hall effect sensors to measure the magnets position directly. This is the primary method for controlling computer fans.



PROS

- Reliable
- High speed
- Efficient
- Mass produced and easy to find

CONS

- Difficult to control without specialized controller
- Requires low starting loads
- Typically require specialized gearboxes in drive applications

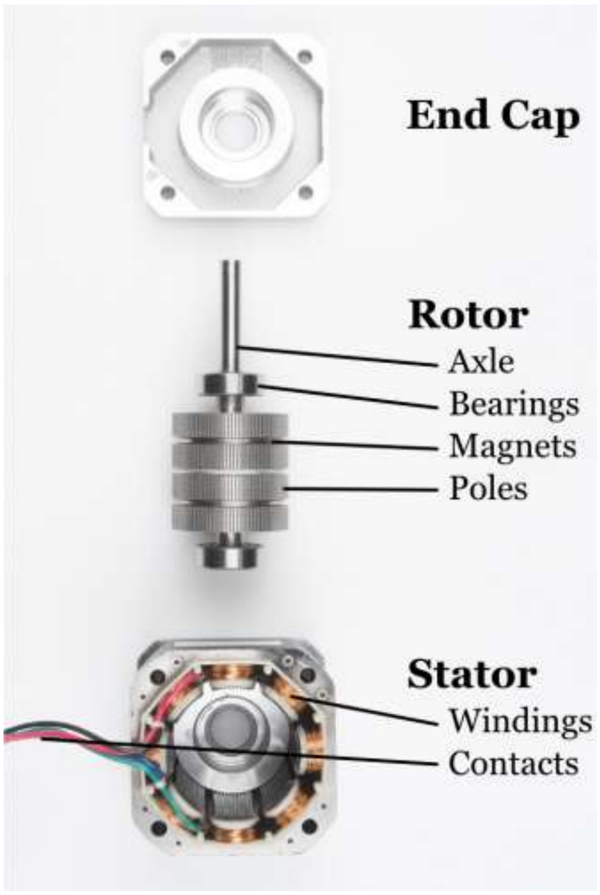
STEPPER MOTORS - SIMPLY PRECISE

Stepper motors are great motors for position control.
They can be found in desktop printers, plotters, 3d printers, CNC milling machines, and anything else requiring precise position control.
Steppers are a special segment of brushless motors. They are purposely built for high-holding torque. This high-holding torque gives the user the ability to incrementally "step" to the next position.

This results in a simple positioning system that doesn't require an encoder.
This makes stepper motor controllers very simple to build and use.

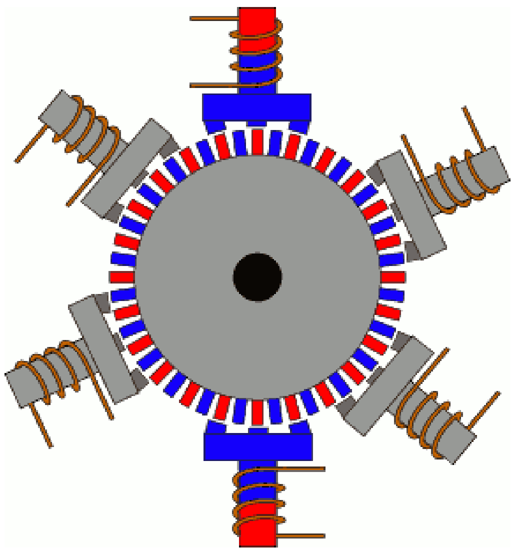
THE ANATOMY OF A STEPPER MOTOR

To better understand how one works, let's start by tearing down a simple stepper motor.
As you can see, these motors are built for direct drive loads containing a few key components.



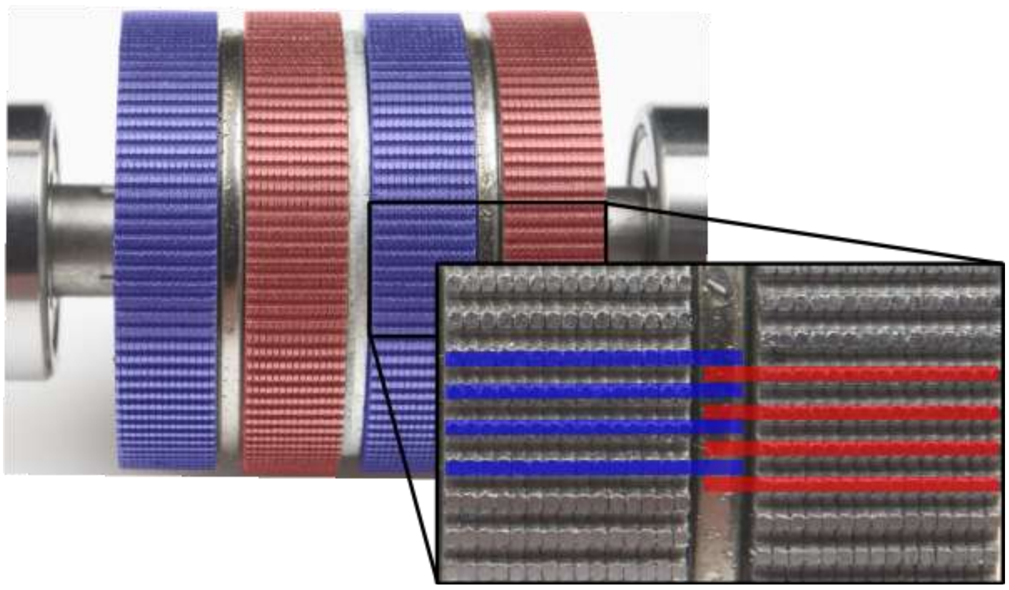
- Axle - Transfers the mechanical power of the motor to the user application
- Bearings - Minimizes friction for the axle
- Magnets - Provide a magnetic field for the windings to attract and repel
- Poles - Increases the resolution of the step distance by focusing the magnetic field
- Windings - Converts electricity to a magnetic field that drives the axle
- Contacts - Brings power from the controller to the windings

THEORY OF OPERATION



(Credit: PCB heaven)

Stepper motors behave exactly the same as a brushless motor, only the step size is much smaller.
The only moving part is the the rotor, which contains the magnets. Where things become complicated is orchestrating the sequence of energizing windings. The polarity of each winding is controlled by the direction of current flow. The animation demonstrates a simple pattern that controllers would follow. Alternating current changes the polarity, giving each winding a "push/pull" effect. A notable difference is how the magnet structure of a stepper is different. It is difficult to get an array of magnets to behave nicely on a small scale. It's also very expensive. To get around this, most stepper motors utilize a stacked plate method to direct the magnetic poles into "teeth".



In a brushless motor, back EMF is used to measure velocity.
A stepper relies on the short throw of each winding to "guarantee" it reaches the desired point in time.
In highspeed travel, this can lead to stalling where the rotor can't keep up with the sequence. There are ways around this, but they rely on a higher understanding of the relationship between motor windings and inductance.

PROS

- Excellent position accuracy
- High holding torque
- High reliability
- Most steppers come in standard sizes

CONS

- Small step distance limits top speed
- It's possible to "skip" steps with high loads
- Draws maximum current constantly

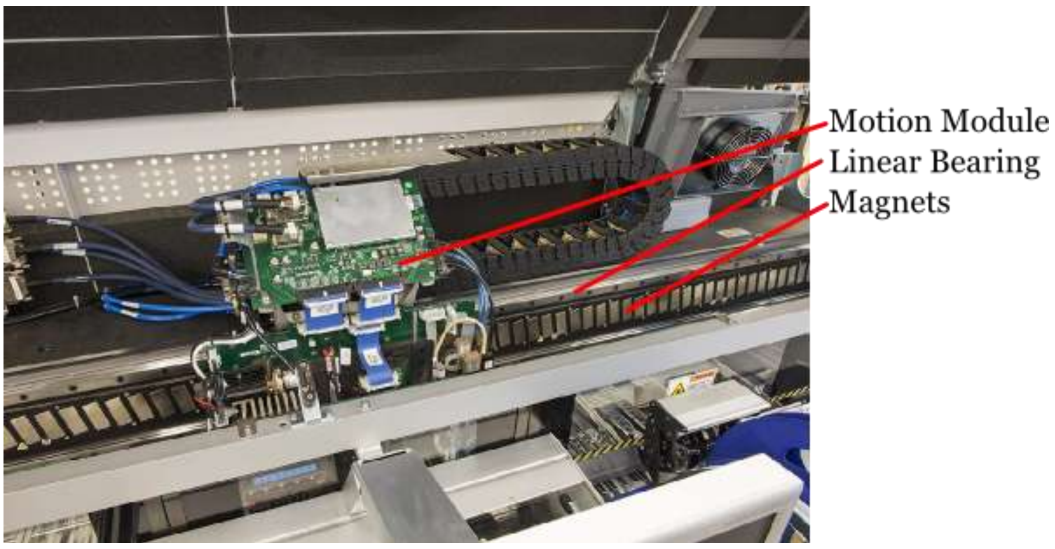
LINEAR MOTORS - THE FUTURE!!!

The future is linear! In high-speed pick and place machines speed is everything. With speed comes friction, with friction comes maintanence, with maintanance comes downtime, with downtime comes lost productivity.
By removing the components needed to transfer rotary to linear motion, the system becomes much lighter and more efficient. Linear motors are simple to maintain, and, with only one moving part, are incredibly reliable.
Did I mention they are incredibly fast?! This is the pick and place machine we are using in production, and it is incredibly fast! This machine also packs such a punch, there is a warning for pacemakers on it. There is an entire row of high-power, rare-earth magnets.

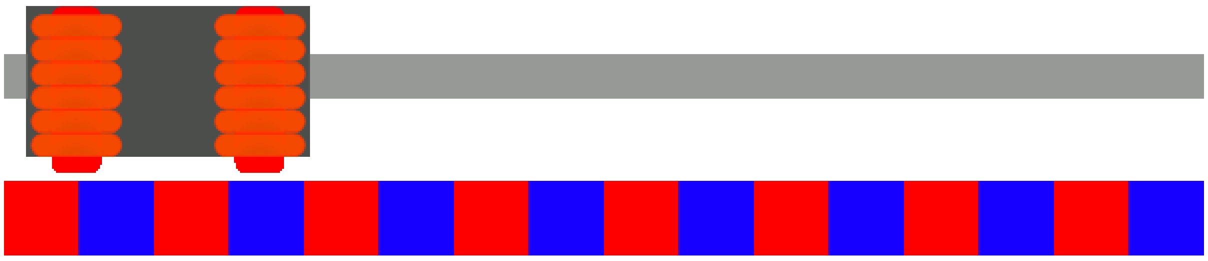
THE ANATOMY OF A LINEAR MOTOR

To better understand how one works, let's look inside our pick and place machine downstairs.

- Motion Module - Contains electromagnets and controller.
- Magnets - Provide a magnetic field for the coils to attract and repel
- Linear Beaning - Keeps the motor in alignment with magnets and is the only moving part.



THEORY OF OPERATION



The mechanics of a linear motor is nearly identical to a brushless motor. The only difference is if you were to take a brushless motor and unfold it into a straight line you'd have a linear motor.

The Motion Module is the only moving part.

Where things become complicated is orchestrating the sequence of energizing coils. The polarity of each coil is controlled by the direction of current flow.
Alternating current changes the polarity giving each coil a "push/pull" effect. In a linear motor, there is typically an encoder or some advanced positioning system to keep track of the location of the Motion Module.

To reach a high position accuracy, the controllers are much more complicated than anything found on a conventional system.

Microstepping is a method to "throttle" the magnets to provide smooth and precise motion.
To achieve this though, linear motors require a highly specialized controller tuned for each motor.

As controller technology improves, we are likely to see these motors decrease in price.

Maybe someday our 3D printers will print in seconds and not hours!

PROS

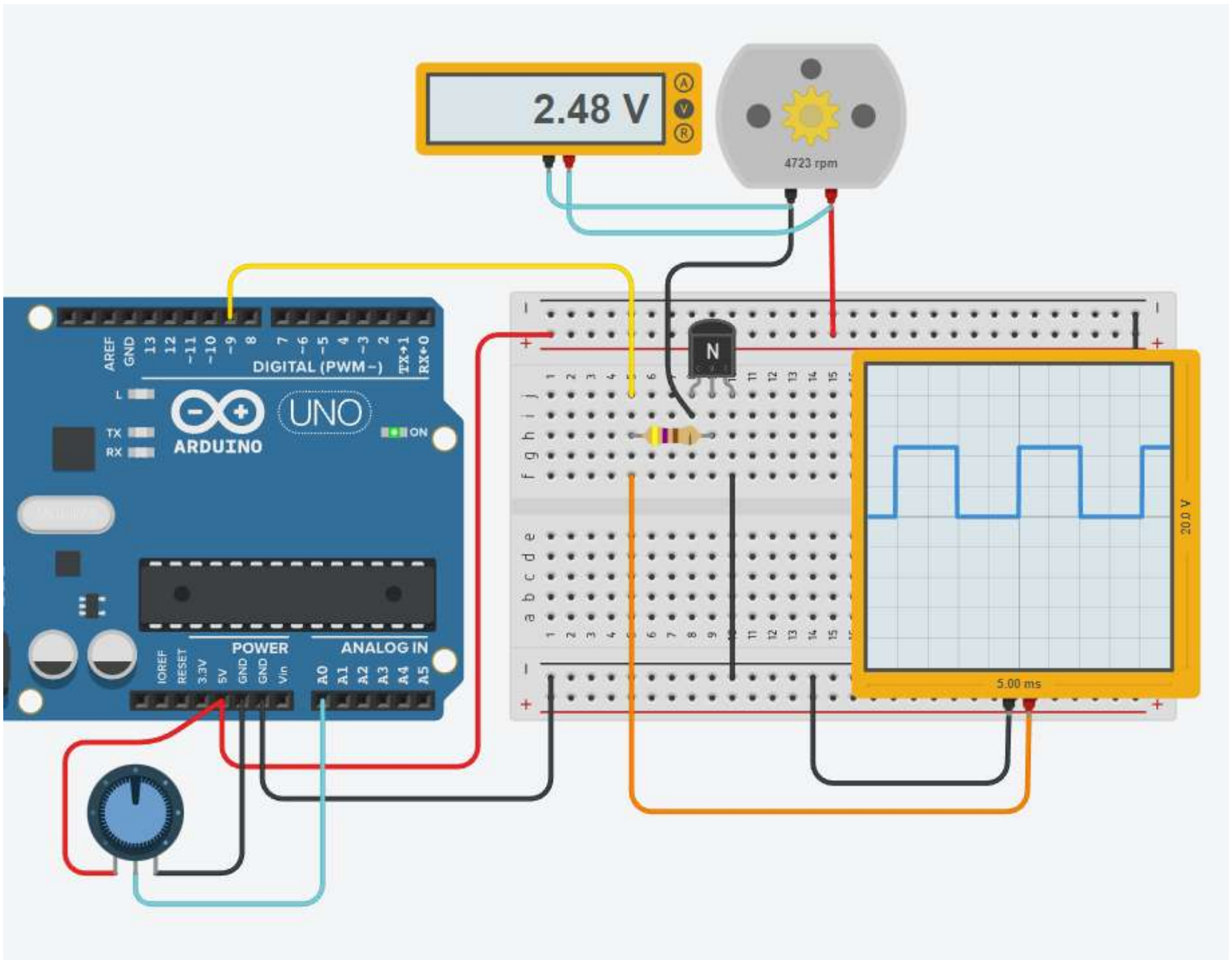
- Reliable
- High speed
- Efficient
- No rotary to linear conversion required

CONS

- Expensive
- Require custom controllers
- Purpose built for each system
- Did I mention expensive?

SPEED REGULATION OF A DC MOTOR

Adjust the rotation speed of the DC motor by a potentiometer and PWM technique.



The DC motor cannot be powered directly from an Arduino PIN since the current required by the motor is higher than that supplied by a PIN.

If the current required by the motor is a few hundred mA, the Arduino 5V output can be used.

If a dedicated power supply is used, it is necessary to put the ground in common with that of the Arduino to guarantee correct operation (the same reference is needed for the ground).

The speed regulation of the DC motor is carried out in 2 ways:

- regulating the voltage across the motor (for example with a potentiometer)

- regulating the time (PWM) in which the maximum power supply voltage of the motor is applied to its ends

The 2nd method allows you to adjust the speed while also keeping the driving torque always high, while in the 1st way the torque drops proportionally to the applied voltage.

To supply sufficient current to the motor it is necessary to use an amplifier (transistor) controlled in PWM by an Arduino PIIN.

The oscilloscope allows you to view the PWM regulation signal (0-5V) generated by an Arduino PIN.

EXERCISE

Complete the assigned code that allows you to adjust the motor speed and show the (0-100) % of the adjustment on the screen via the serial line

CODE:

```
int motorPin = 9;  // PWM
int potPin = A0;   // POTENTIOMETER
int potValue = 0;

void setup()
{
  pinMode(potPin, INPUT);
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  // read the value from the sensor
  if (analogRead(potPin) != potValue)
  {
      __________________
      __________________
      __________________
      __________________
  }
  delay(20); // Wait for 20ms
}
```

BRIEFLY ANSWER THE FOLLOWING QUESTIONS

1. What is the relationship between the electric current and the magnetic field in a conductor?
2. Why is copper wire used in the form of coils in motors?
3. What is the difference between a magnet and an electromagnet?
4. What happens in an electromagnet if we reverse the flow of current?
5. What are the main parts of a brush DC motor?
6. Come si elimina il "cogging" nei motori CC?
7. What is the torque of a DC motor like at low rpm?
8. What are the benefits of a brushless DC motor?
9. What are the disadvantages of a brushless DC motor?
10. Why are linear motors lighter and more efficient than DC motors for achieving linear motion?
11. What are linear bearings used for in linear motors?
12. Why is "microstepping" used in linear motors?