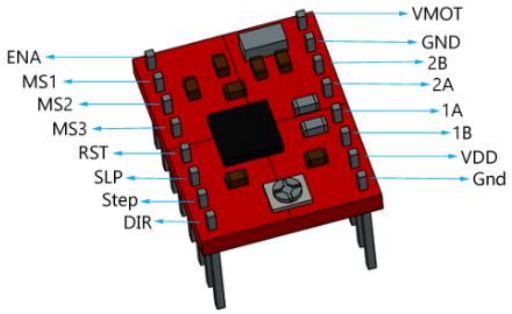
Controlling Linear Actuator by Entering Length

A4988 Stepper Motor Pins:



Vdd and GND: Should be connected to the 5v and GND parts of the Arduino.
Vmot and GND: Should be connected to 12 volt and GND to provide the 12 volt needed by the stepper motor.
1A,1B,2A,2B: Pins to which the stepper motor is connected.
Dır: Controls the direction of the motor.
Step: Controls the steps.
MS1, MS2, MS3: Microstep Selection Pins.
Sleep and Reset: When they are connected to each other, the controller becomes active.
En: When the Enable pin is active, the motor is grounded. We can limit the power usage by making this pin active and passive.

Micro step Resolution	MS1	MS2	MS3
Full Step	low	low	low
Half Step	high	low	low
Quarter Step	low	high	low
Eighth Step	high	high	low
Sixteenth Step	high	high	high

For one revolution needed steps are calculated as;

Full Step mode

$$\frac{360}{1.8^\circ} = 200$$

Half step;

$$\frac{360}{0.9°} = 400$$

Quarter step;

$$\frac{360}{0.45°} = 800$$

Eighth step;

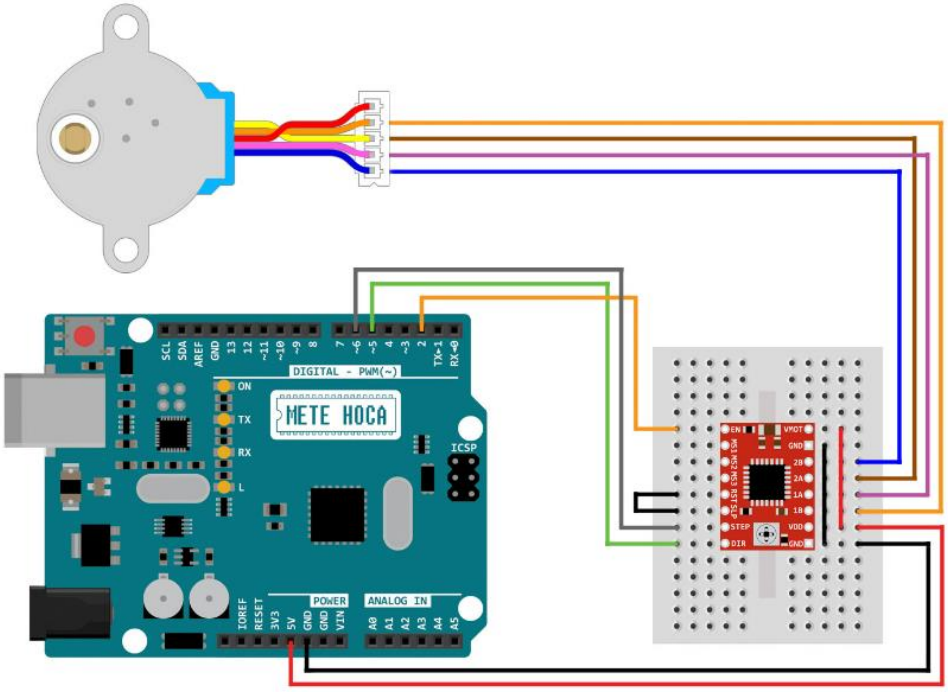
$$\frac{360}{0.225°} = 1600$$

Sixteenth step;

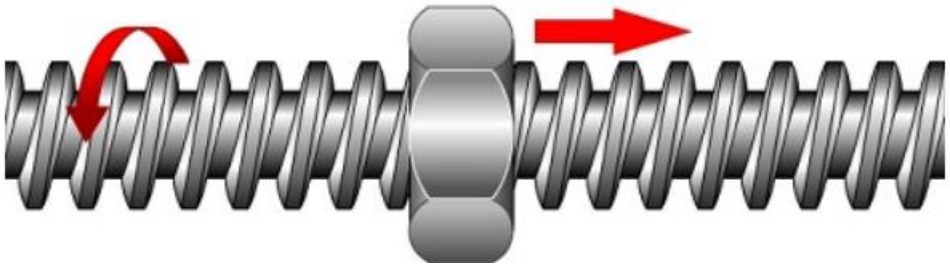
$$\frac{360}{0.1125°} = 3200$$

Stepper Motor Controller Connections:

Stepper motor model is: 17HS4401S



Trapezoidal lead screw motion principle:



The screw rotates, the nut does not rotate, the nut moves along the screw.

Linear Actuator:

Linear actuators are created by properly combining the stepper motor and trapezoidal lead screw. With each step of the stepper motor, the part attached to the screw shaft moves by the length of the Lead. It is used to move a load back and forth.

$L = p \times n_s$

L = Lead of thread.

P = Thread pitch.

n_s = Number of thread starts.

Length Calculation in Code:

```
toplamAdim = (mmFinal / 8) * stepsPerRevolution;
```

The specified 'stepsPerRevolution' is the number of steps required for the stepper motor to complete one revolution. The specified 'stepsPerRevolution' is the number of steps required for the stepper motor to complete one revolution.

$L = p \times n_s$

The number of 'starts' of the used shaft: $n_s = 4$

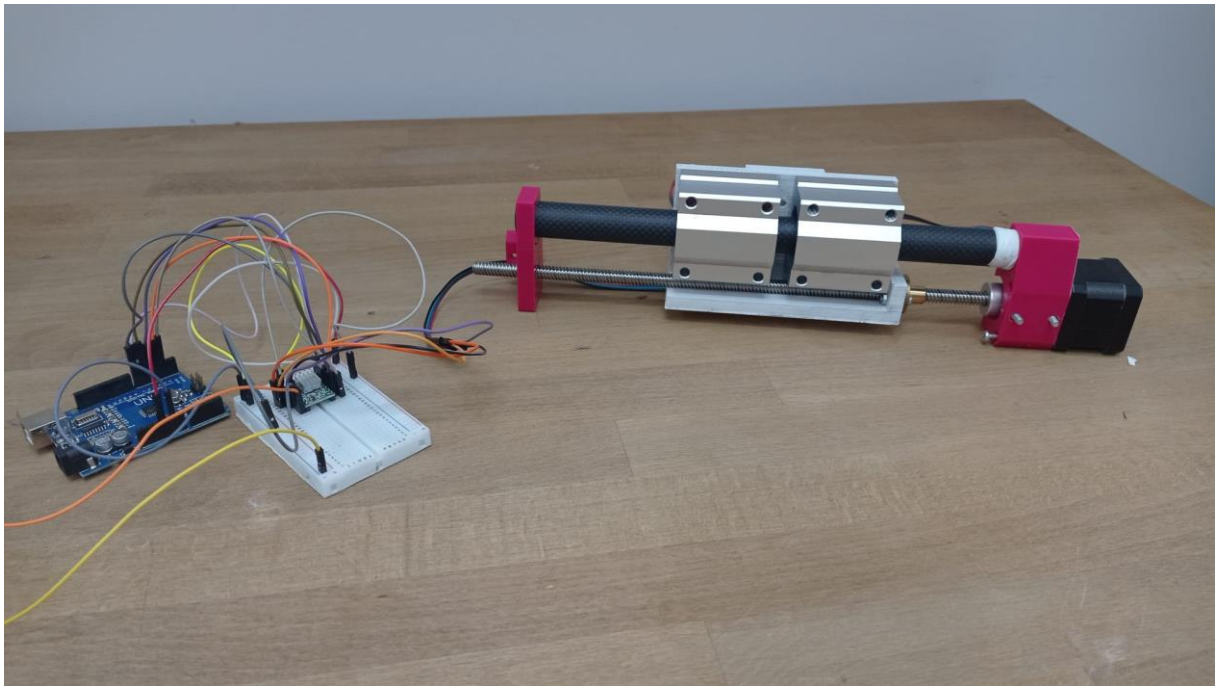
Pitch value of shaft: p = 2

Lead of thread is : L = 8

So when the 'stepsPerRevolution' is completed, the shaft moves 8 mm. By multiplying the formula by 1/8, we ensure that the shaft moves 1 mm when the 'stepsPerRevolution' is completed. In this way, we can determine the 'mmFinal' value as the length we want our part to move and ensure that it moves at the desired length. When we equate this equation to the toplamAdim command, we obtain the number of steps the stepper motor must take to cover the specified distance.

T8 Trapezoidal lead screw:

Linear Actuator:



Controlling Linear Actuator by Entering Length in Arduino Uno

```
#define dirPin 6
#define stepPin 7
#define controlPin 2
#define MS1 3
#define MS2 4
#define MS3 5
#define stepsPerRevolution 200
#define joyX A0
#define joyY A1
int xValue;
int yValue;
int toplamAdim;
#define mmFinal 50
int i;
int son;

void setup(){
  pinMode(stepPin, OUTPUT);
 pinMode(dirPin, OUTPUT);
  pinMode(MS1, OUTPUT);
 pinMode(MS2, OUTPUT);
  pinMode(MS3, OUTPUT);
}

void loop() {

  // Microstep control settings:
 digitalWrite(MS1, LOW);
 digitalWrite(MS2, HIGH);
 digitalWrite(MS3, LOW);
 toplamAdim = (mmFinal / 8) * stepsPerRevolution;


 // Set the spinning direction counterclockwise:

 if(mmFinal > 0){
  digitalWrite(dirPin, HIGH);
  if(son != 1){
   for( i = 0; i < toplamAdim; i++){

   digitalWrite(stepPin, HIGH);
   delayMicroseconds(2000);
   digitalWrite(stepPin, LOW);
   delayMicroseconds(2000);
   son = 1;
    }
   }
```

```
  }
  if(mmFinal < 0){
   // Set the spinning direction counterclockwise:
  digitalWrite(dirPin, LOW);
   if(son != 1){
    for( i = 0; i < -toplamAdim; i++){

    digitalWrite(stepPin, HIGH);
    delayMicroseconds(2000);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(2000);
   son = 1;
    }
   }
  }
 }
```

Prepared by Mert Sürekci

02.08.2024