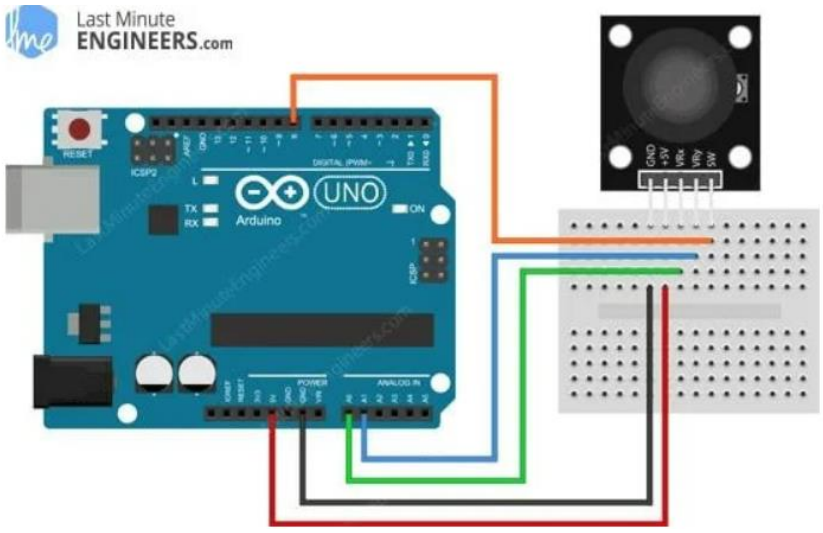
Moving the Timing Belt Linear Actuator with Joystick

Joystick:

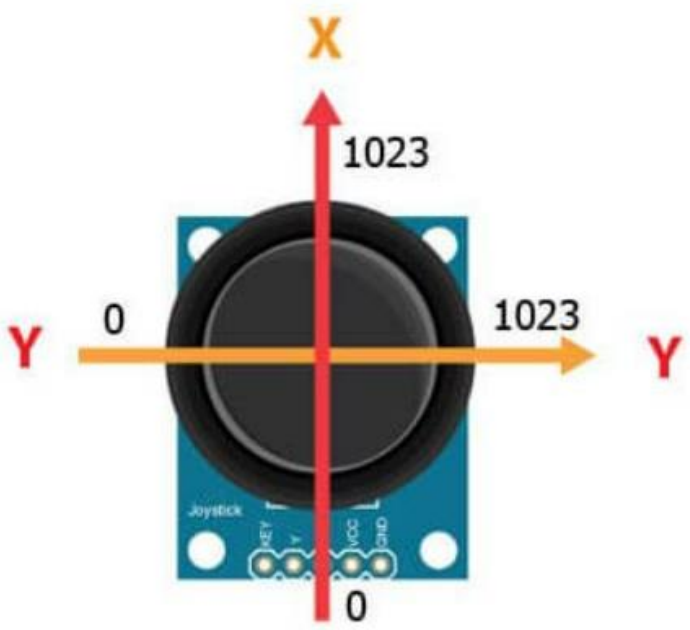
Joystick Connections:

Arduino Uno is used in the project:

Vcc pin is the pin where 5 volts are supplied. GND pin is the ground connection. VRx ; Allows the joystick to be read in the horizontal direction (X coordinate); that is, it measures how far the joystick is pushed left and right. VRy ; Allows the joystick to be read in vertical direction (Y coordinate).

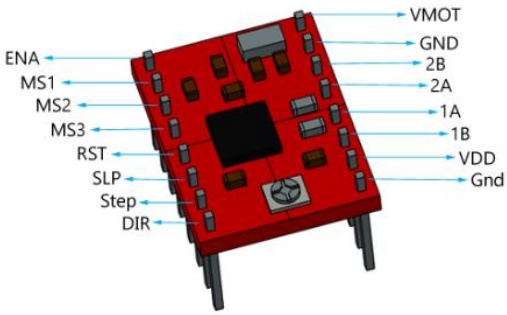


Reading Analog Values from Joystick:



The change in the joystick control lever causes a value change in the potentiometer inside the joystick. Arduino obtains digital data according to the value changes in the potentiometer. Since the Arduino board has a 10-bit ADC resolution, the values in each analog channel (axis) can vary between 0 and 1023. The value that the X and Y axes will take according to the movement of the control lever will be between 0 and 1023. When it is in the exact middle position, we will obtain approximately 512. These values we obtain are the data that we need to use in determining the direction and adjusting the speed while controlling the movement of the stepper motor.

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

$$\frac{360}{0.9°} = 400$$

Half step;

$$\frac{360}{0.45°} = 800$$

Quarter step;

$$\frac{360}{0.225°} = 1600$$

Eighth step;

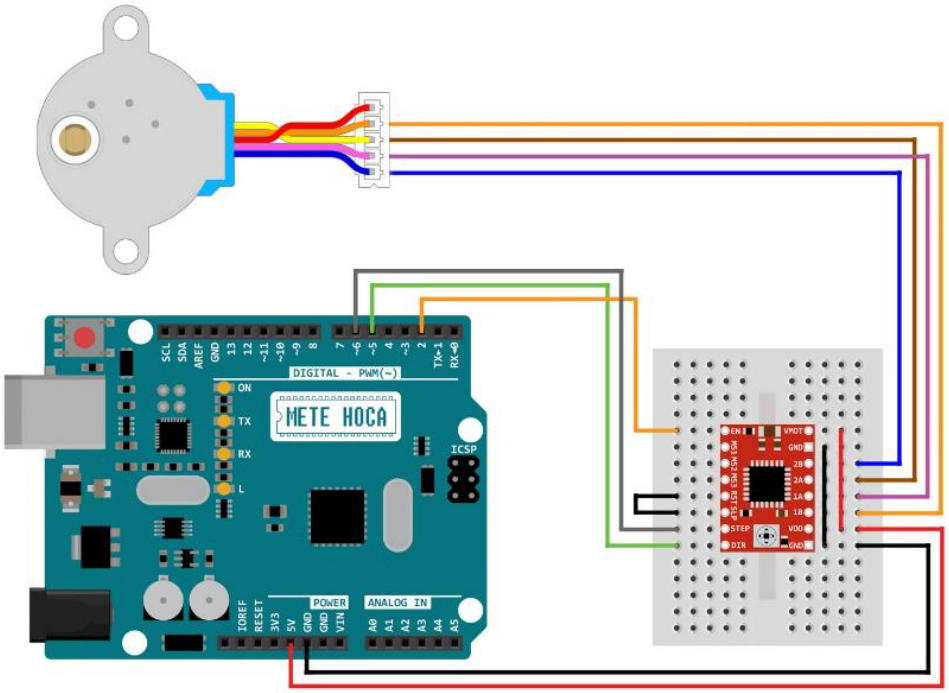
$$\frac{360}{0.1125°} = 3200$$

Sixteenth step;

$$\frac{360}{0.05625°} = 6400$$

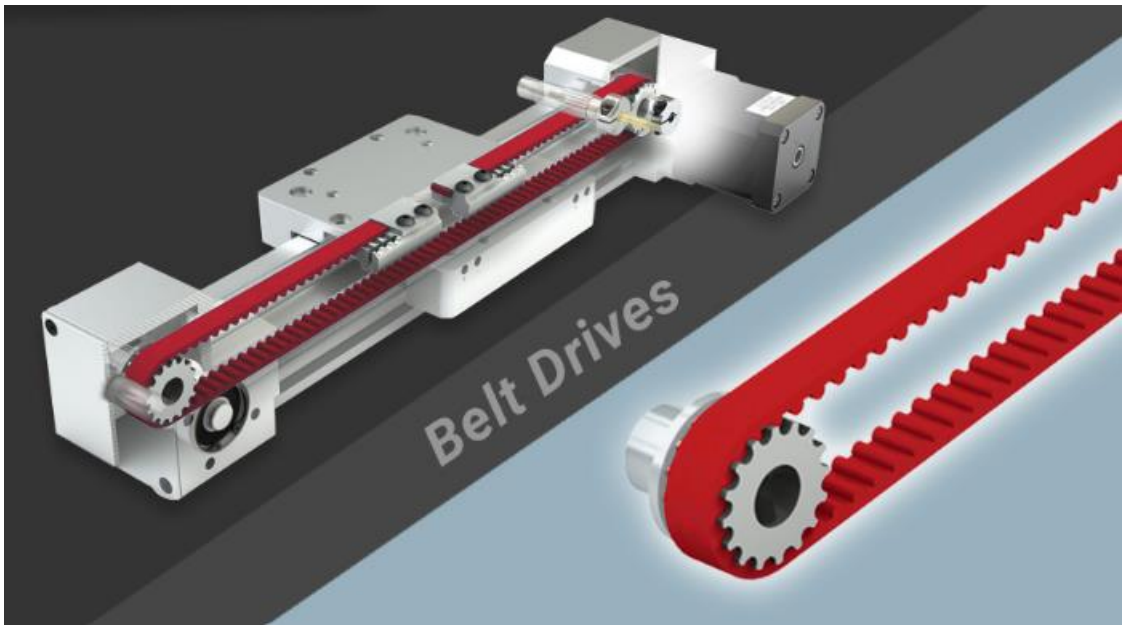
Stepper Motor Controller Connections:

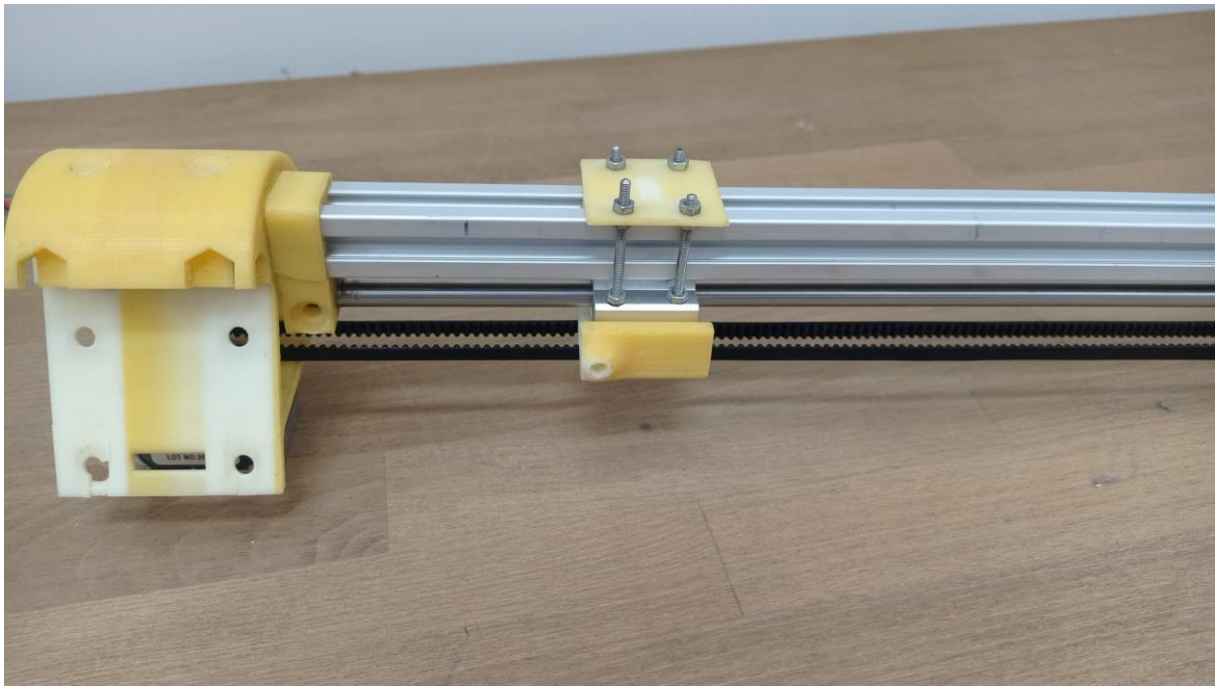
Stepper motor model is: JK42HM34-1334

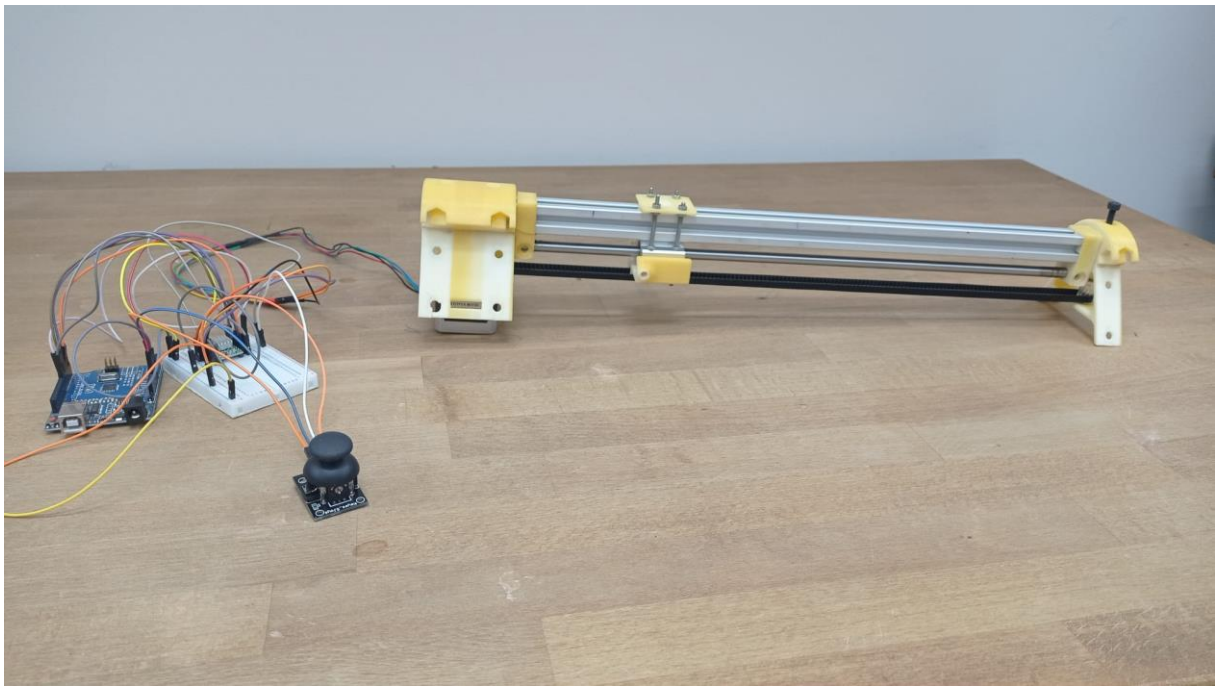


Timing Belt Moving Principle:

As the gear wheel rotates, the gear of the gear wheel drive the gear of the timing belt.The timing belt starts to move in this way. As the timing belt moves, the part connected to the timing belt also starts to move.







Moving the Timing Belt Linear Actuator with Joystick in Arduino UNO

```
#define dirPin 6
#define stepPin 7
#define controlPin 2
#define MS1 3
#define MS2 4
#define MS3 5
#define stepsPerRevolution 400
#define joyX A0
#define joyY A1
int xValue;
int yValue;


void setup() {
  pinMode(stepPin, OUTPUT);
 pinMode(dirPin, OUTPUT);
  pinMode(joyX, INPUT);
 pinMode(joyY, INPUT);
  pinMode(MS1, OUTPUT);
 pinMode(MS2, OUTPUT);
  pinMode(MS3, OUTPUT);
}
```

```
void loop() {
  // Microstep control settings:
 digitalWrite(MS1, LOW);
 digitalWrite(MS2, LOW);
 digitalWrite(MS3, LOW);
  xValue = analogRead(joyX);
  yValue = analogRead(joyY);
 // Set the spinning direction counterclockwise:
 digitalWrite(dirPin, HIGH);
 // Spin the stepper motor:
 if(yValue < 480){
 // Enable pin is grounded:
 digitalWrite(controlPin, LOW);
 // four lines result in 1 step:
 digitalWrite(stepPin, HIGH);
 delayMicroseconds(400 + yValue);
 digitalWrite(stepPin, LOW);
 delayMicroseconds(400 + yValue);
 }

 // Set the spinning direction clockwise:
 digitalWrite(dirPin, LOW);
 // Spin the stepper motor:
 if(yValue > 520){
 // Enable pin is grounded:
 digitalWrite(controlPin, LOW);
 // These four lines result in 1 step:
 digitalWrite(stepPin, HIGH);
 delayMicroseconds(1423 - yValue);
 digitalWrite(stepPin, LOW);
 delayMicroseconds(1423 - yValue);

 }
 else{
  // Enable pin is grounded:
 digitalWrite(controlPin, HIGH);
  }
}
```