Ain Shams University

Faculty of Engineering

Mechatronics Department-Senior 1

MCT333_Mechatronics Systems Design

Final Report on

(Robot Manipulator)

Team (2)

Name	ID
Mohammed Abdulaziz Farrag Ahmed	1805858
Mohamed Emad Mohamed Hussien Omar	1802245
Mohamed Emad Mohamed Moslem	1806770
Alaa Elsayed Metwally Mohamed	1809361
Ali Abdelhakem Mahmod Hussin	1805347

Submitted to:

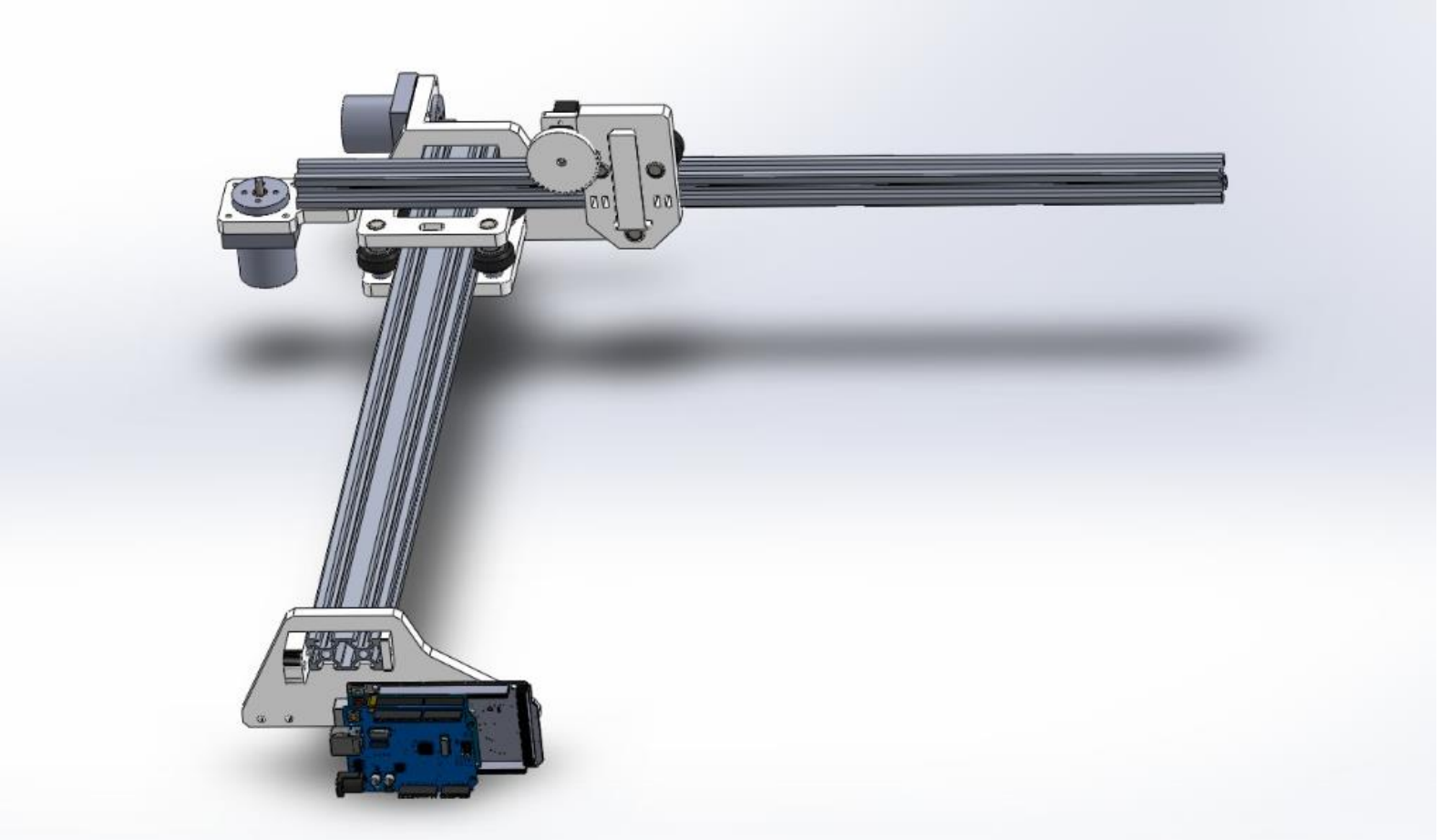
Dr / Mohamed Omar MohamedDr

/Mohamed Ibrahim

Eng / Hossam Moataz Elkeshky

Contents

ROBOT MANIPULATOR



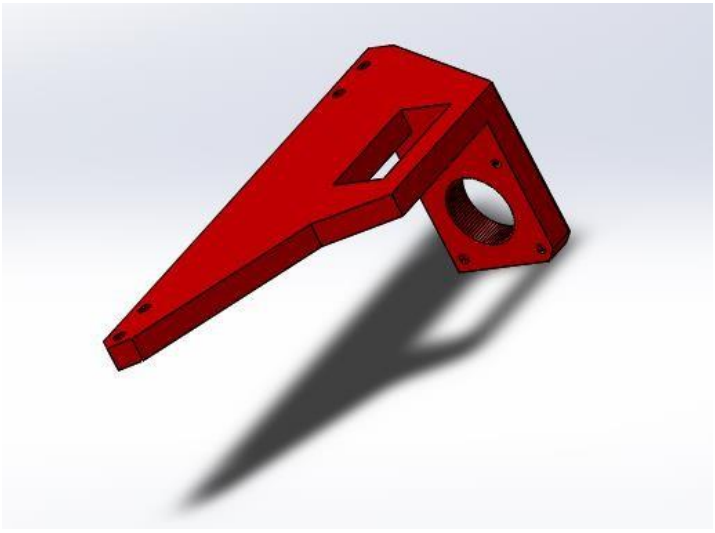
Description of Project:

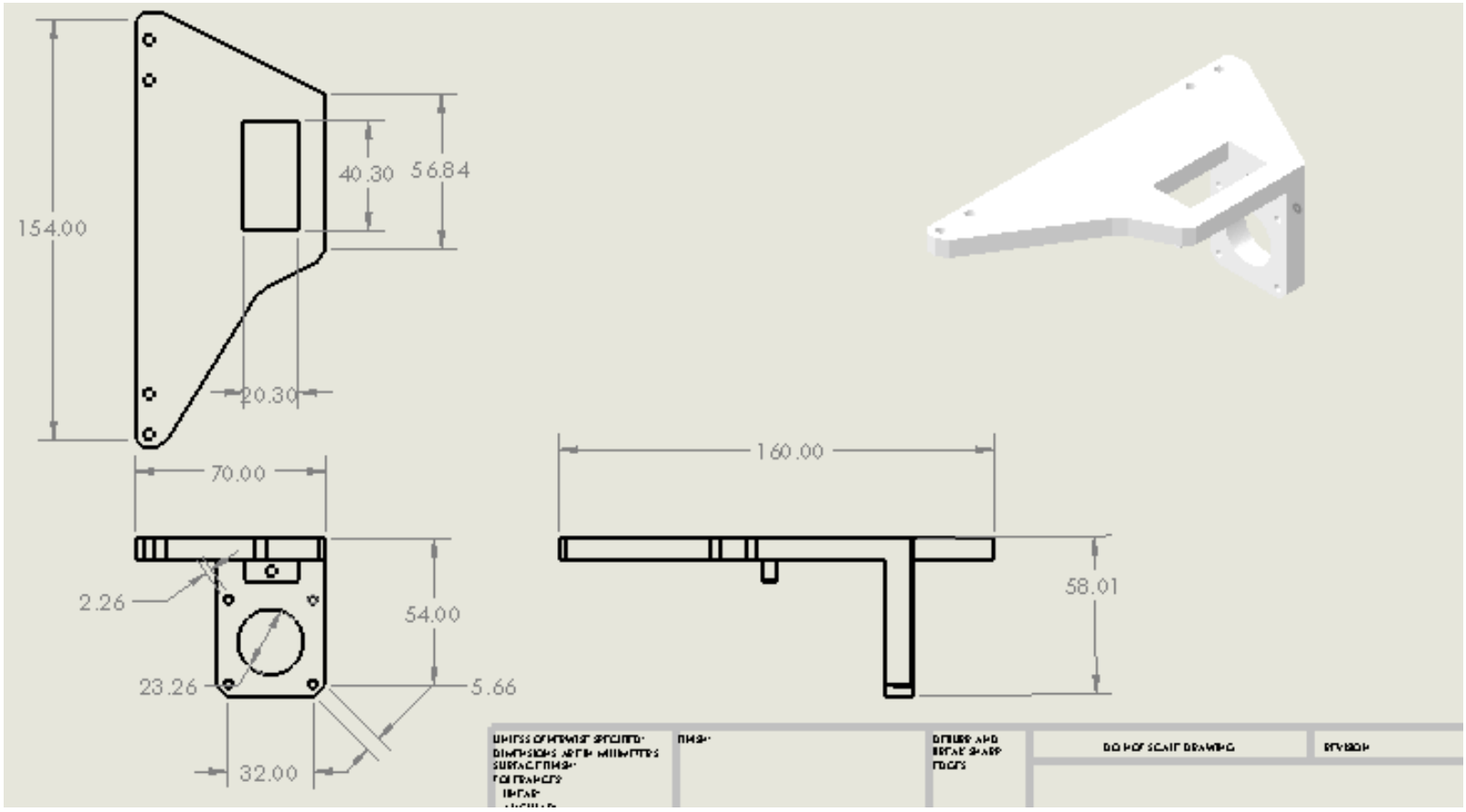
1) Industrial Robot have an objective of writing with 3 degree of freedom actuators, sensors, Control Systems.
2) Writing Arm (2D Printer) that should have high level from accuracy and precision of control.
3) Main objective is writing texts that include capital and small letters with control of font size and type.
4) Scope of 2 phases :
   i. Open loop
   ii. Closed loop
5) Control systems:
   i. Mechanical System
      a. 2 section of Aluminum x and y axes
      b. Carriages or Holders z axis
      c. Fixation and additional parts like Base, gears, Holders,…
   ii. Electrical System
      a) Microcontroller(Arduino)
      b) Actuators (servo motor, stepper motor, Dc motor)
      c) Drivers
      d) 2 Sensors(limit switch)
      e) Cnc shield
      f) Cables and jumpers
   iii. Software System
      a) Ink-scape
      b) G-code
      c) GRBL Plotter
      d) GUI

Mechanical System:

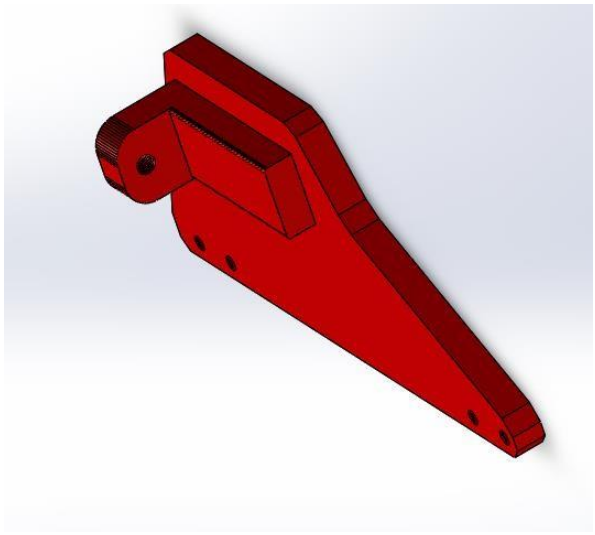
1. Parts Drawing

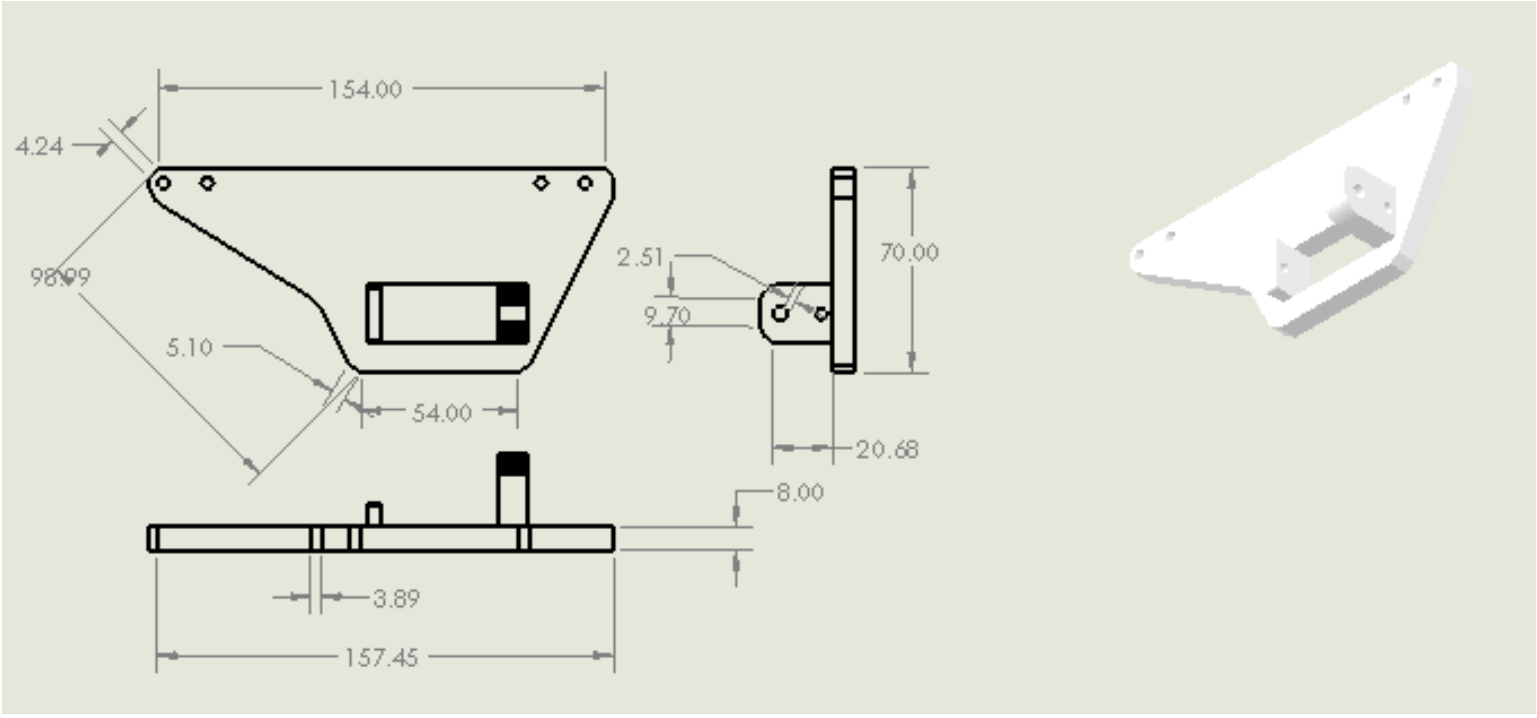
Base Plate Top



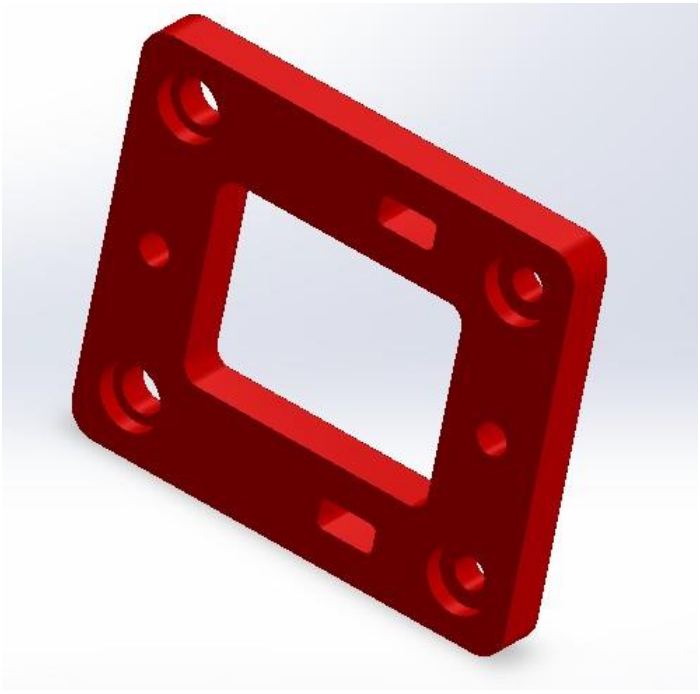


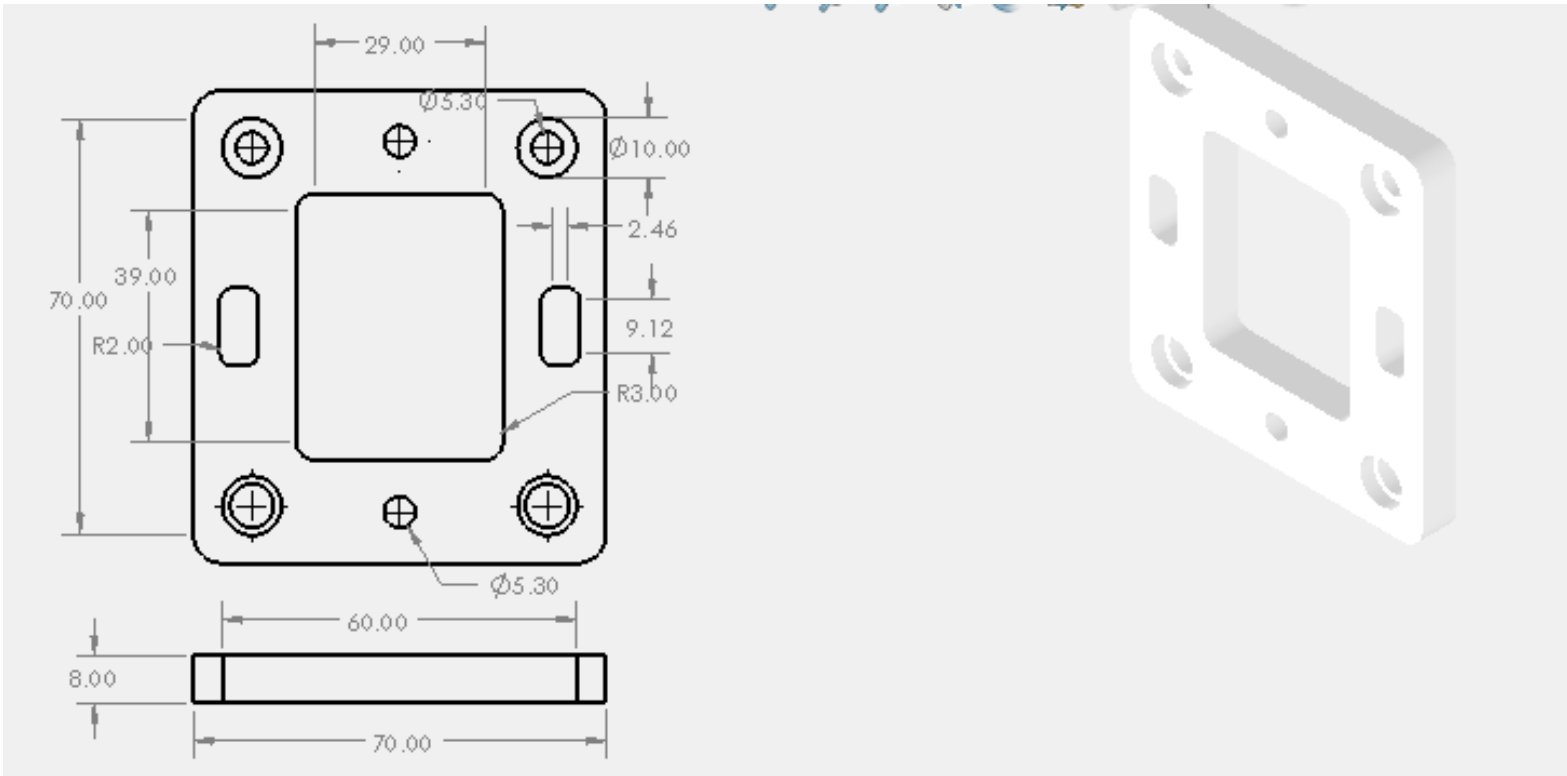
Base Plate Bottom



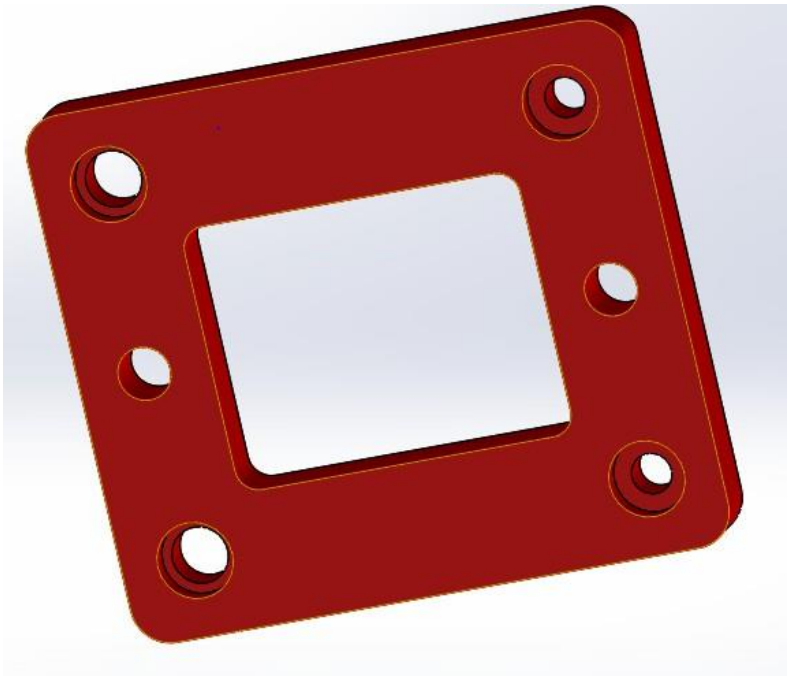


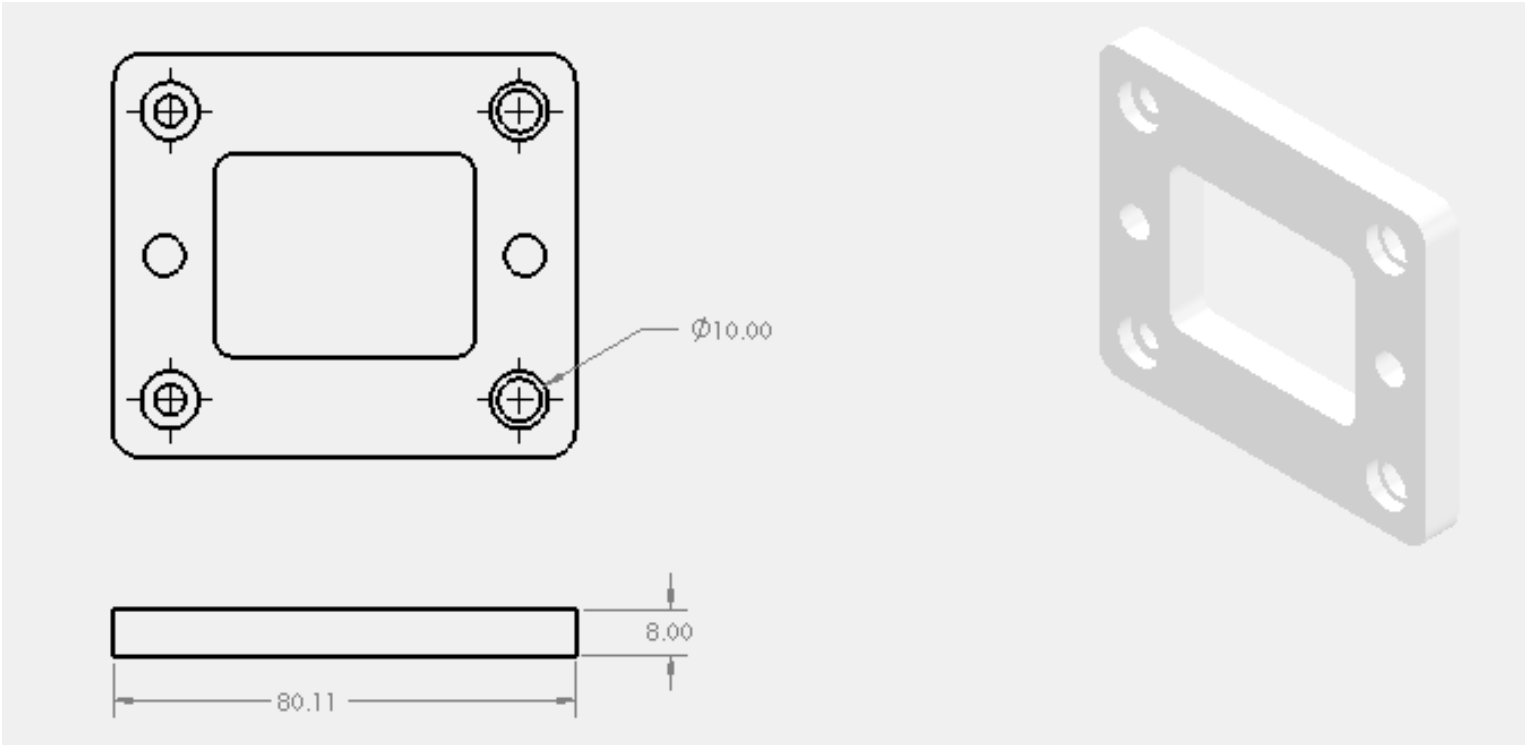
Y Carriage Top



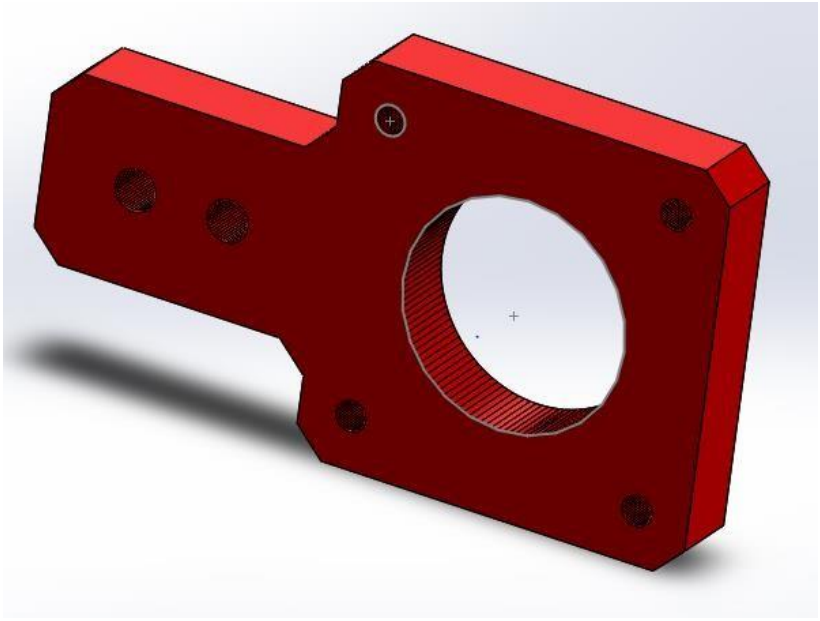


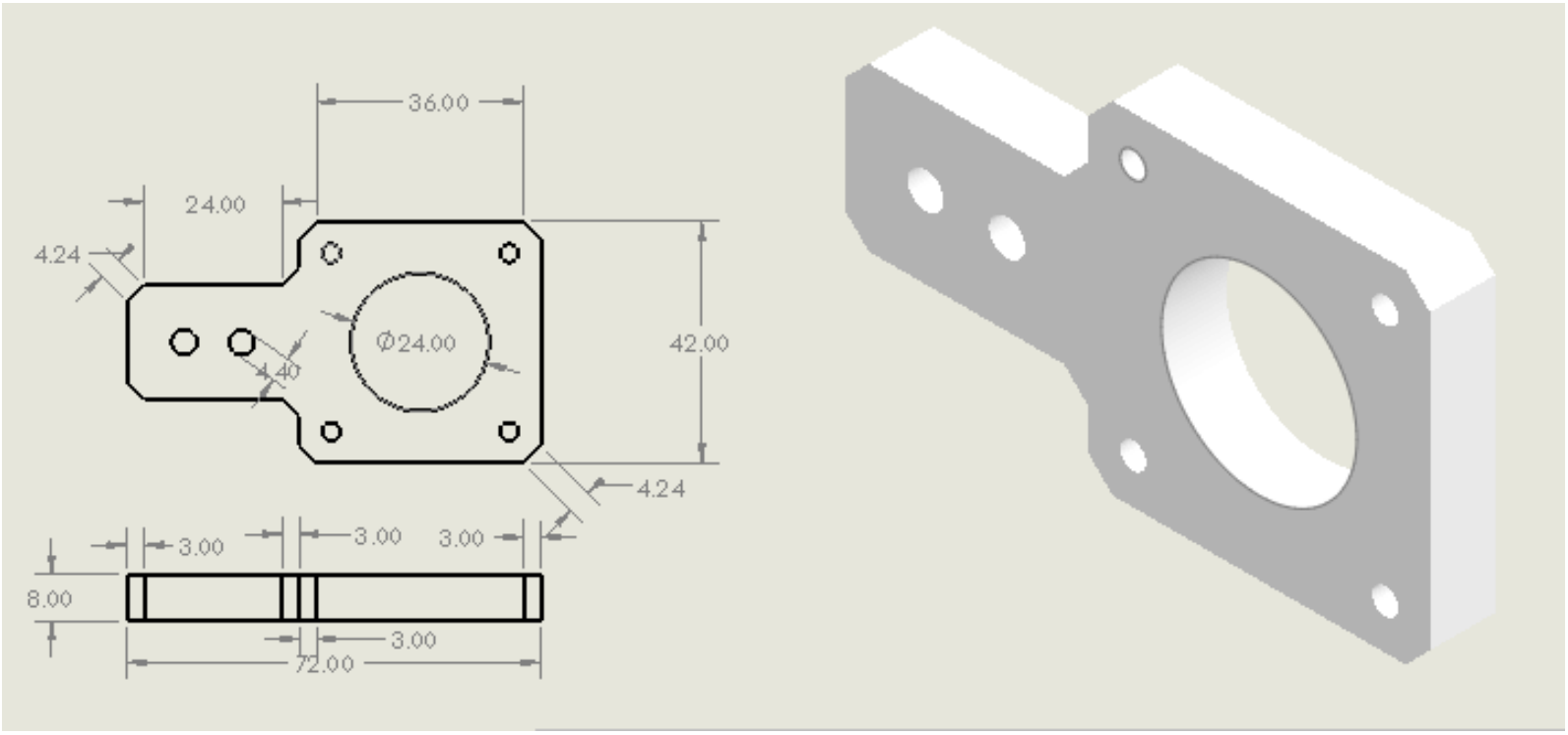
Y Carriage Bottom



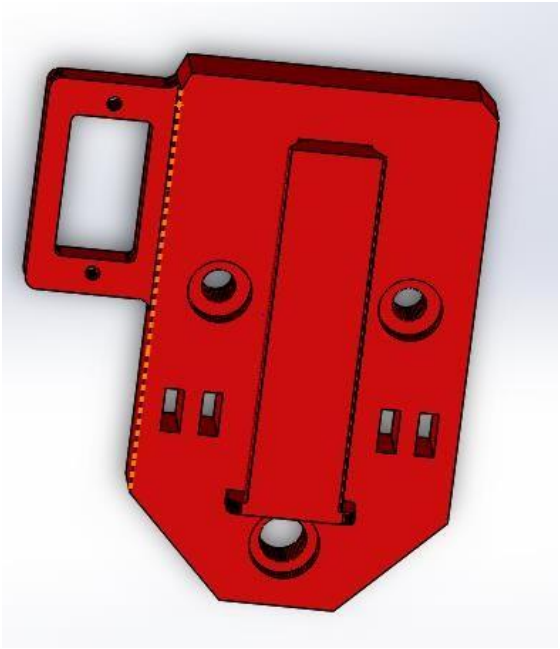


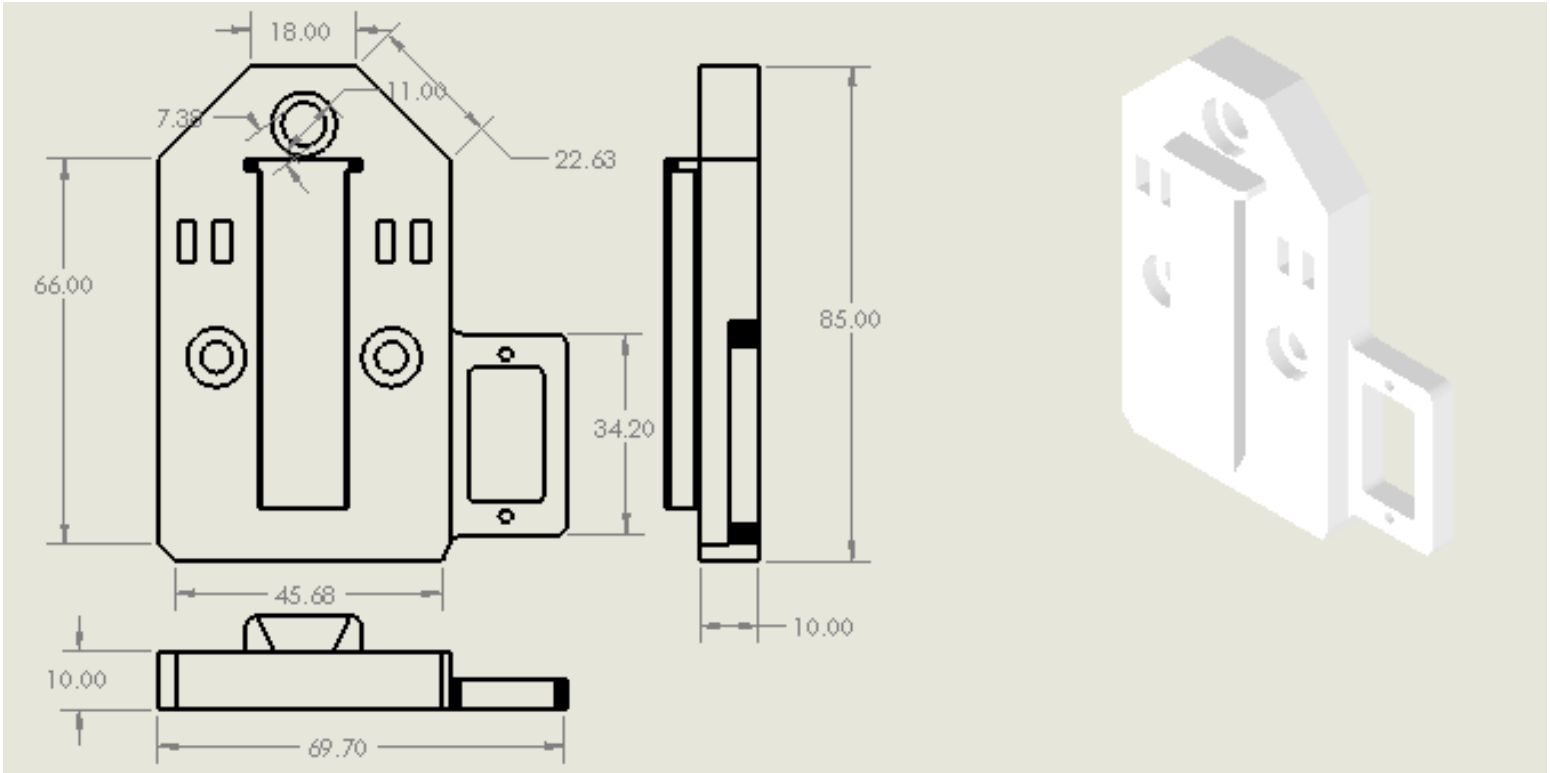
X motor Holder



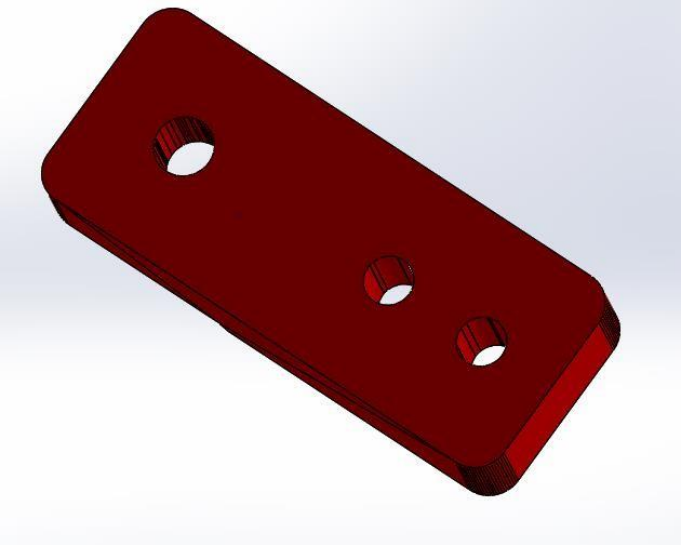


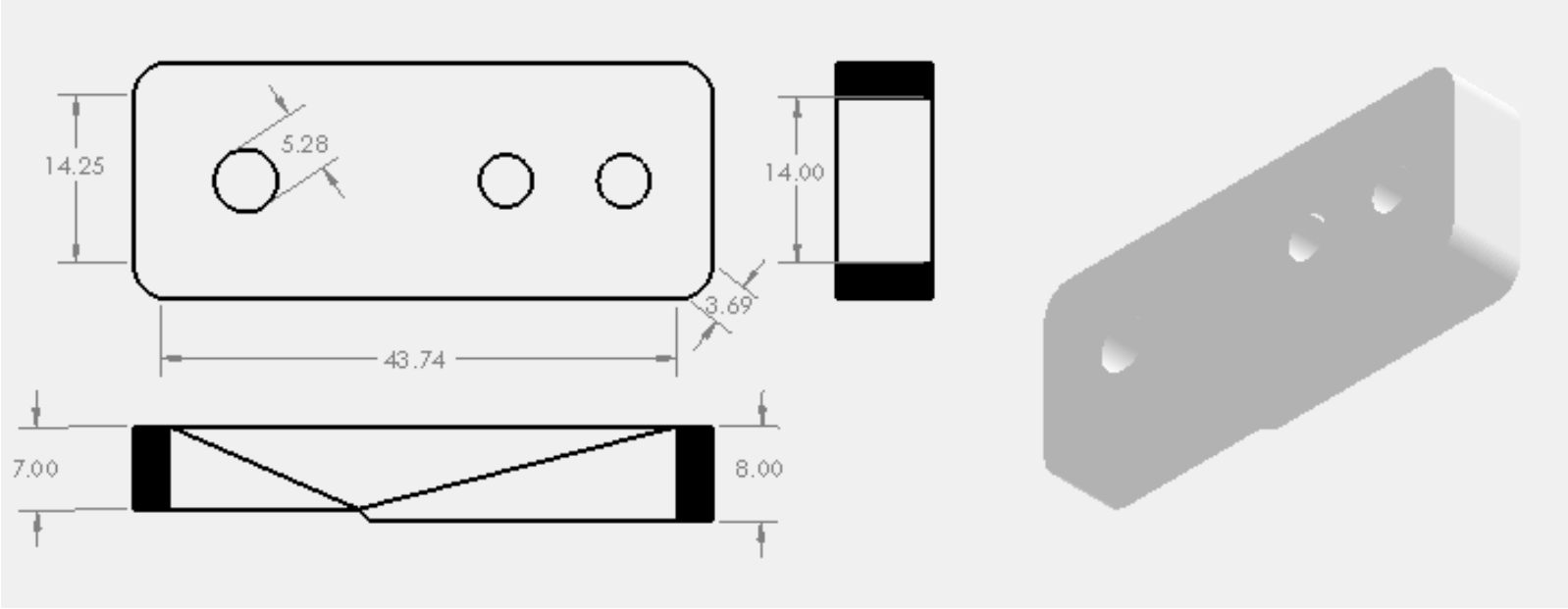
X Carriage



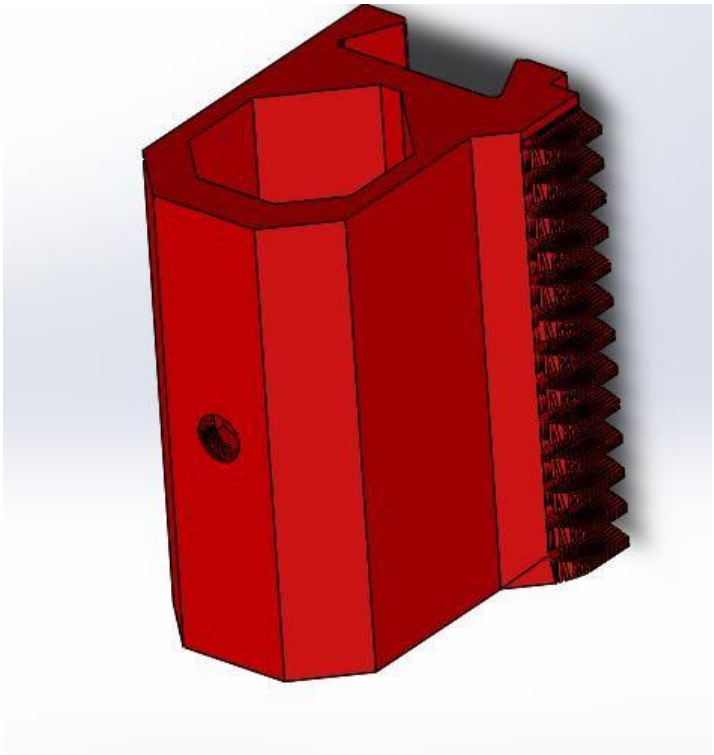


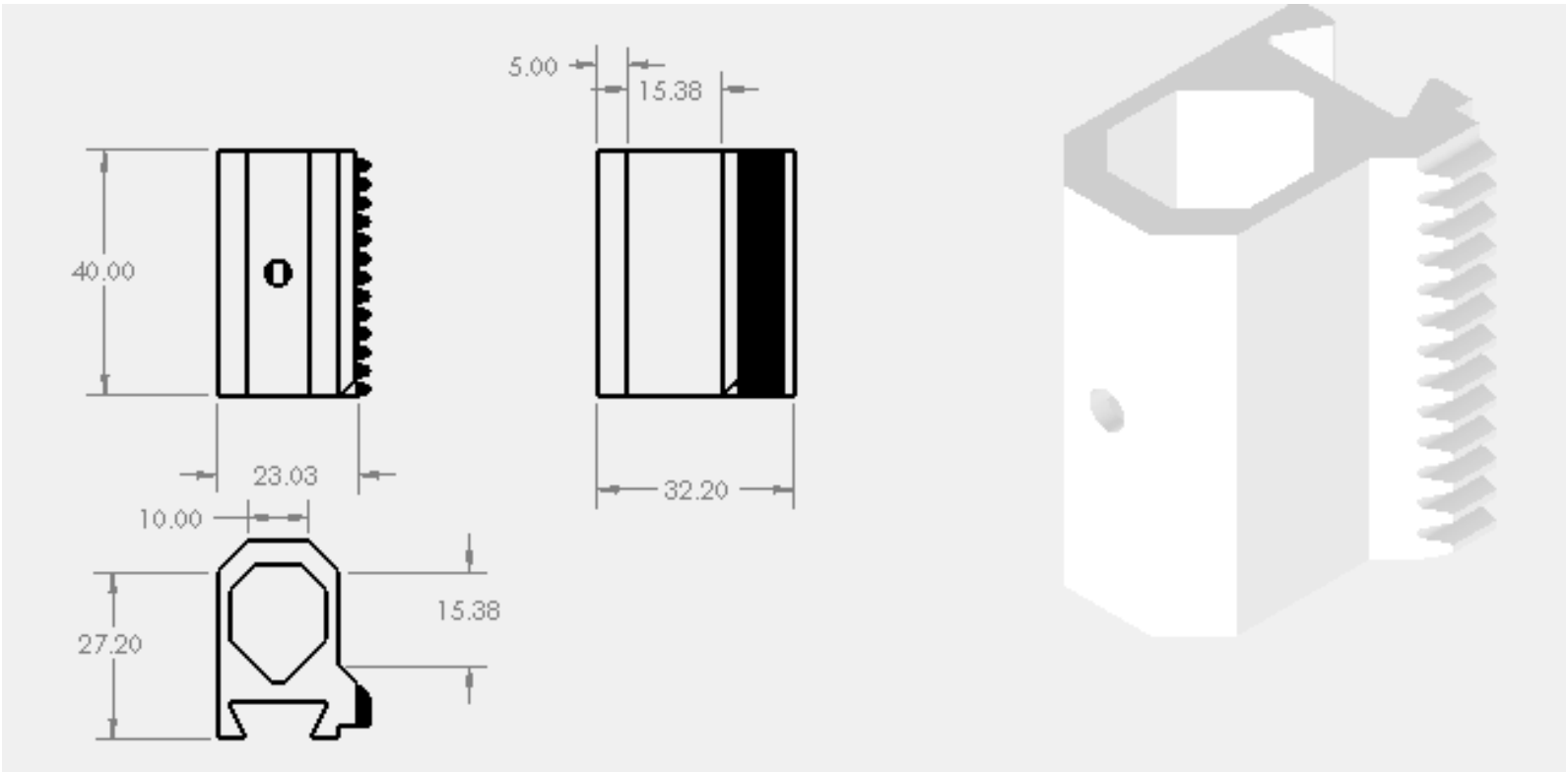
X pulley Holder



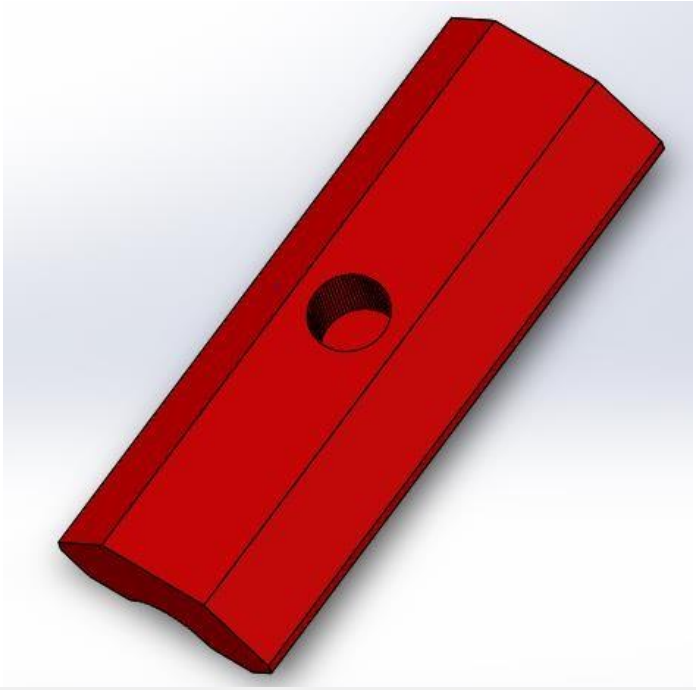


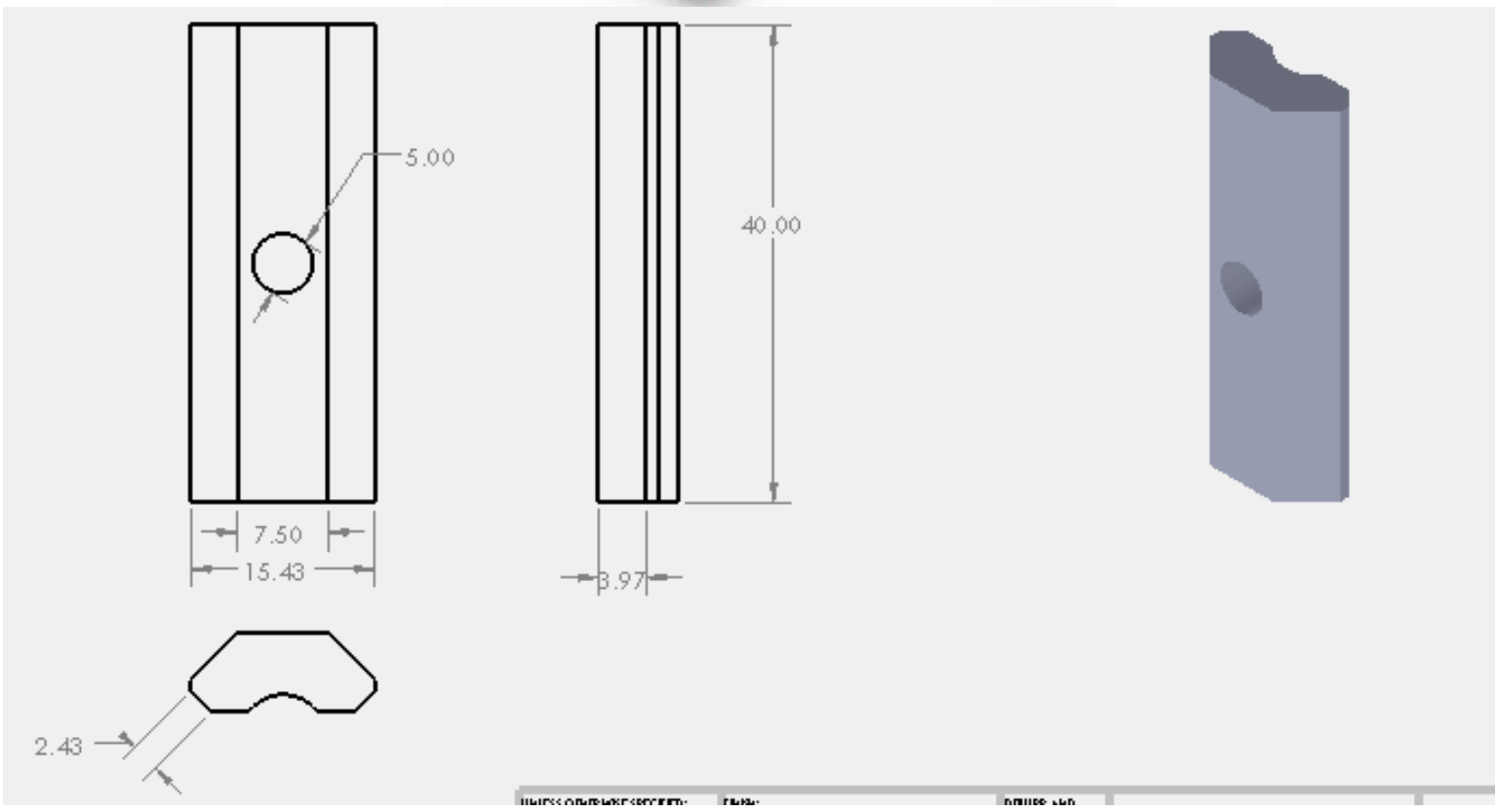
Pen Holder



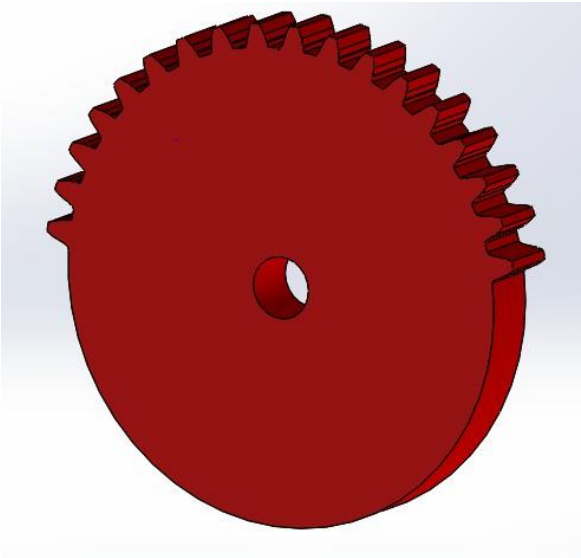


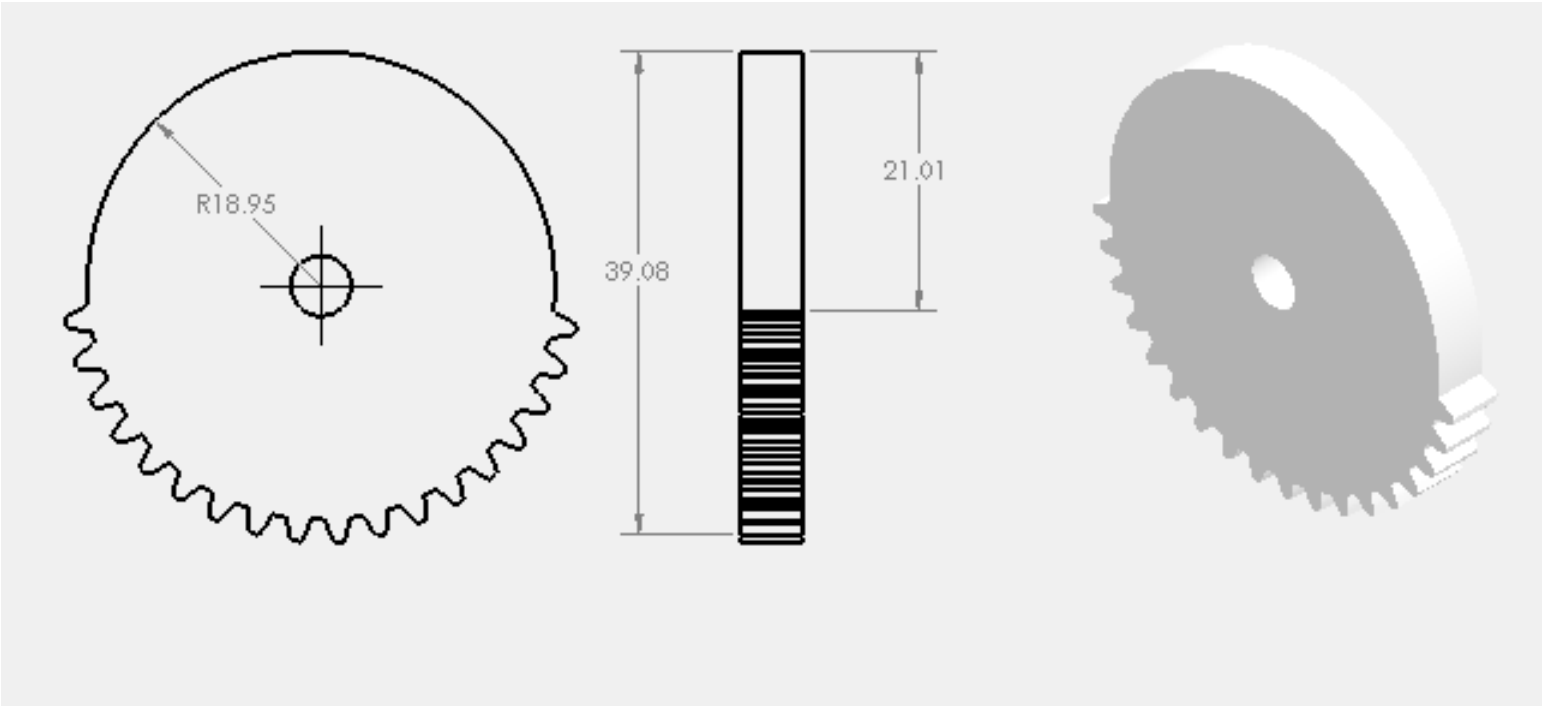
Pen Holder clamp



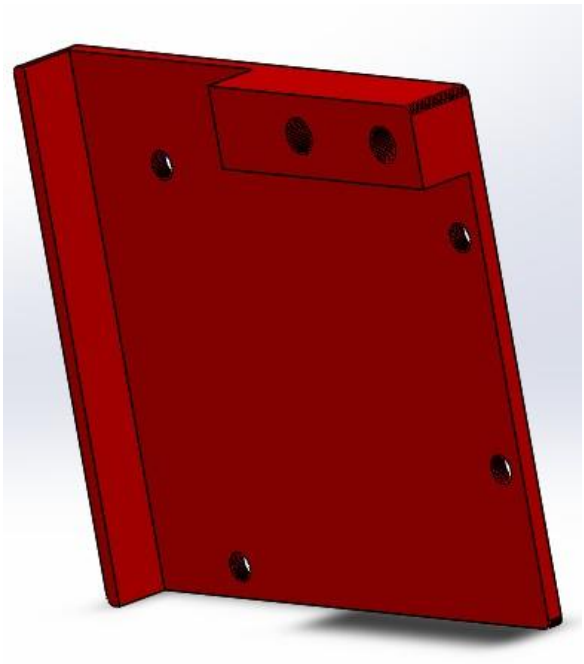


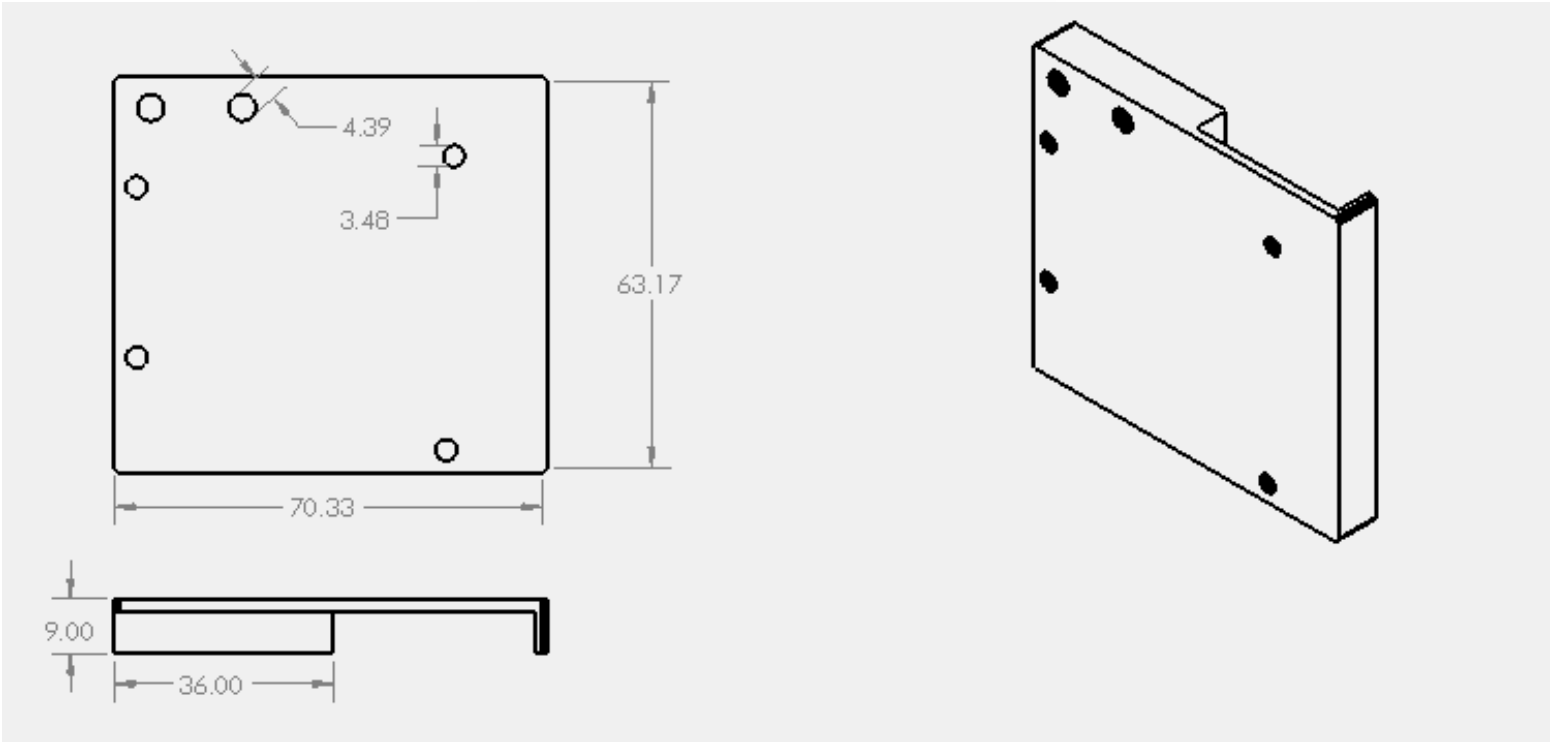
Gear of Z axis





Arduino Holder Plate

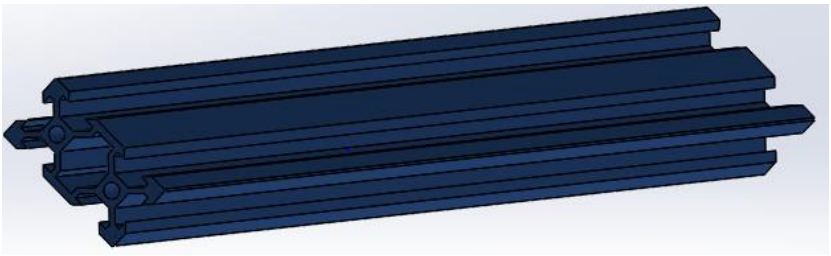




2. Transmission and Connections

2.1. Section of Aluminum

- 2 section 50*4 cm



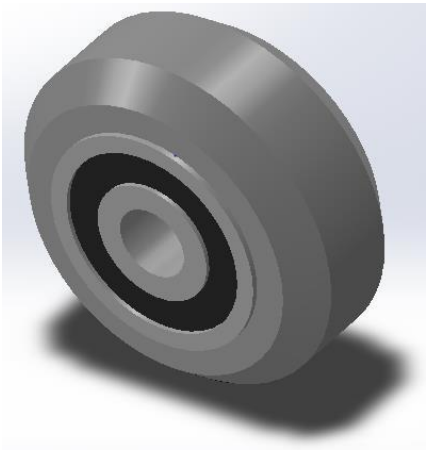
2.2. Timing Pulley GT2

1) 20 teeth
2) Inner diameter 5 mm
3) Width 6 mm

2.3. Timing Belt

1) Width 6 mm

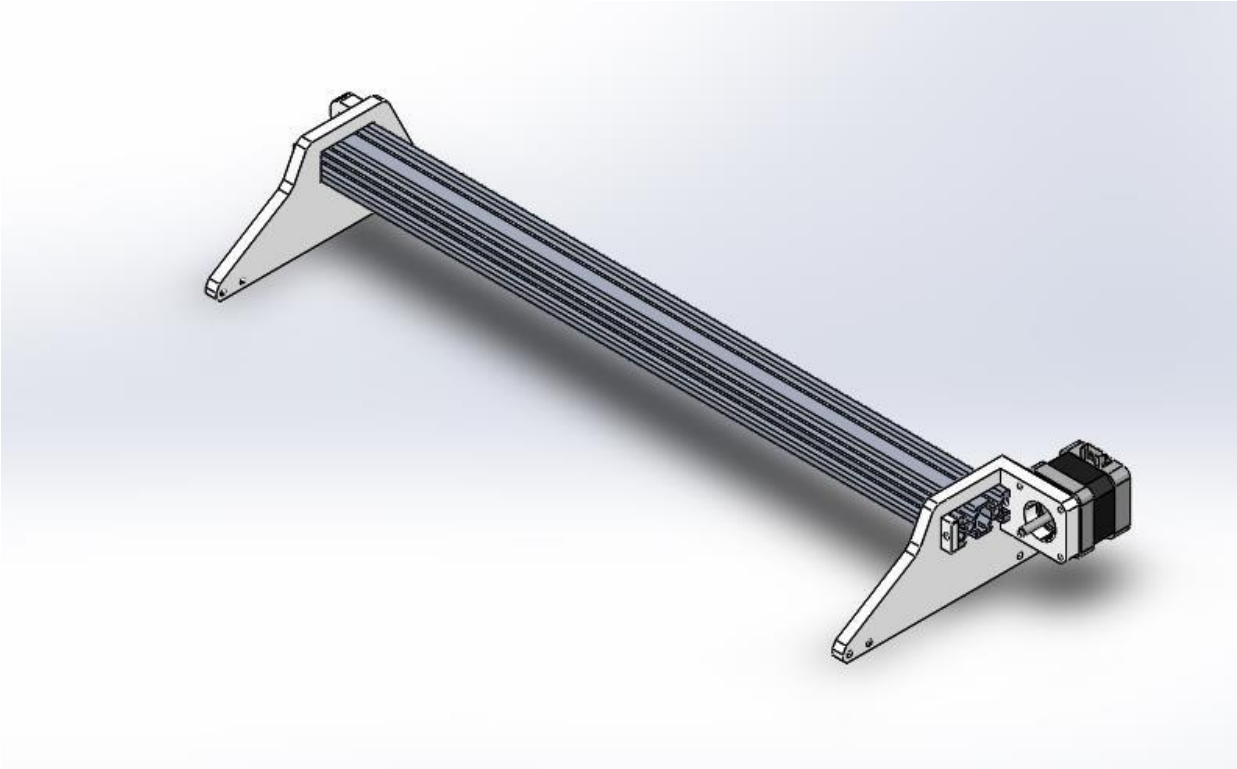
2.4. Wheels with bearing



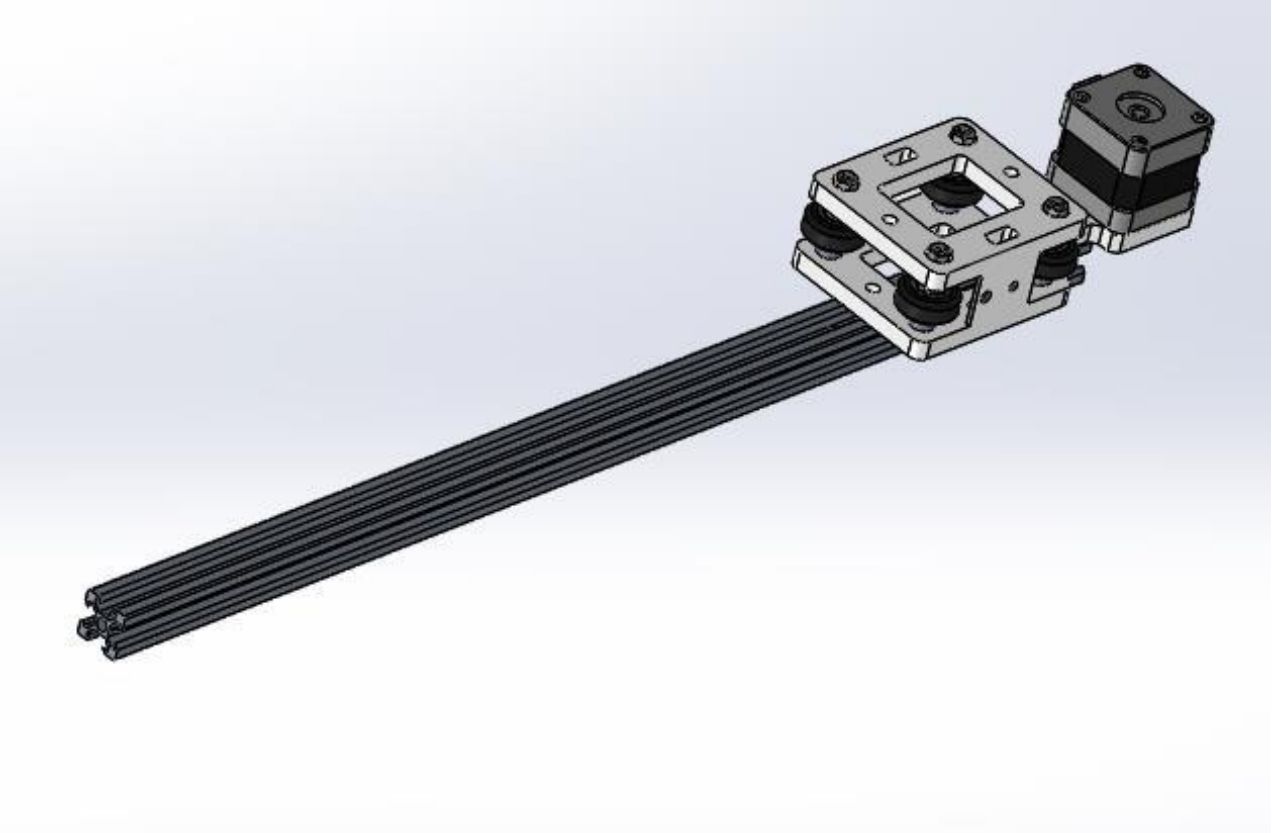
2.5. Spacers , Bolts and Nuts

3. Assembly

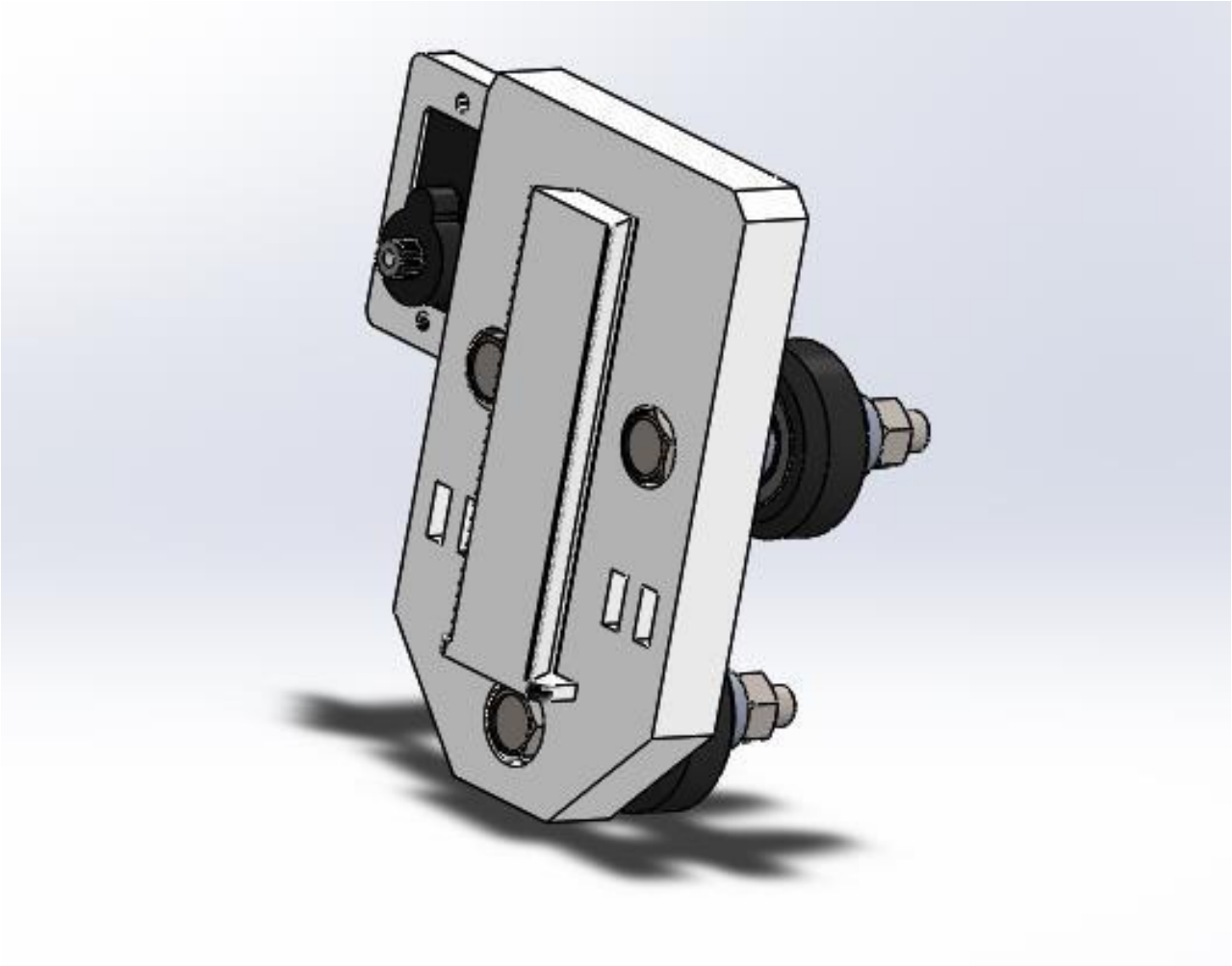
3.1. X-axis



3.2. Y-axis

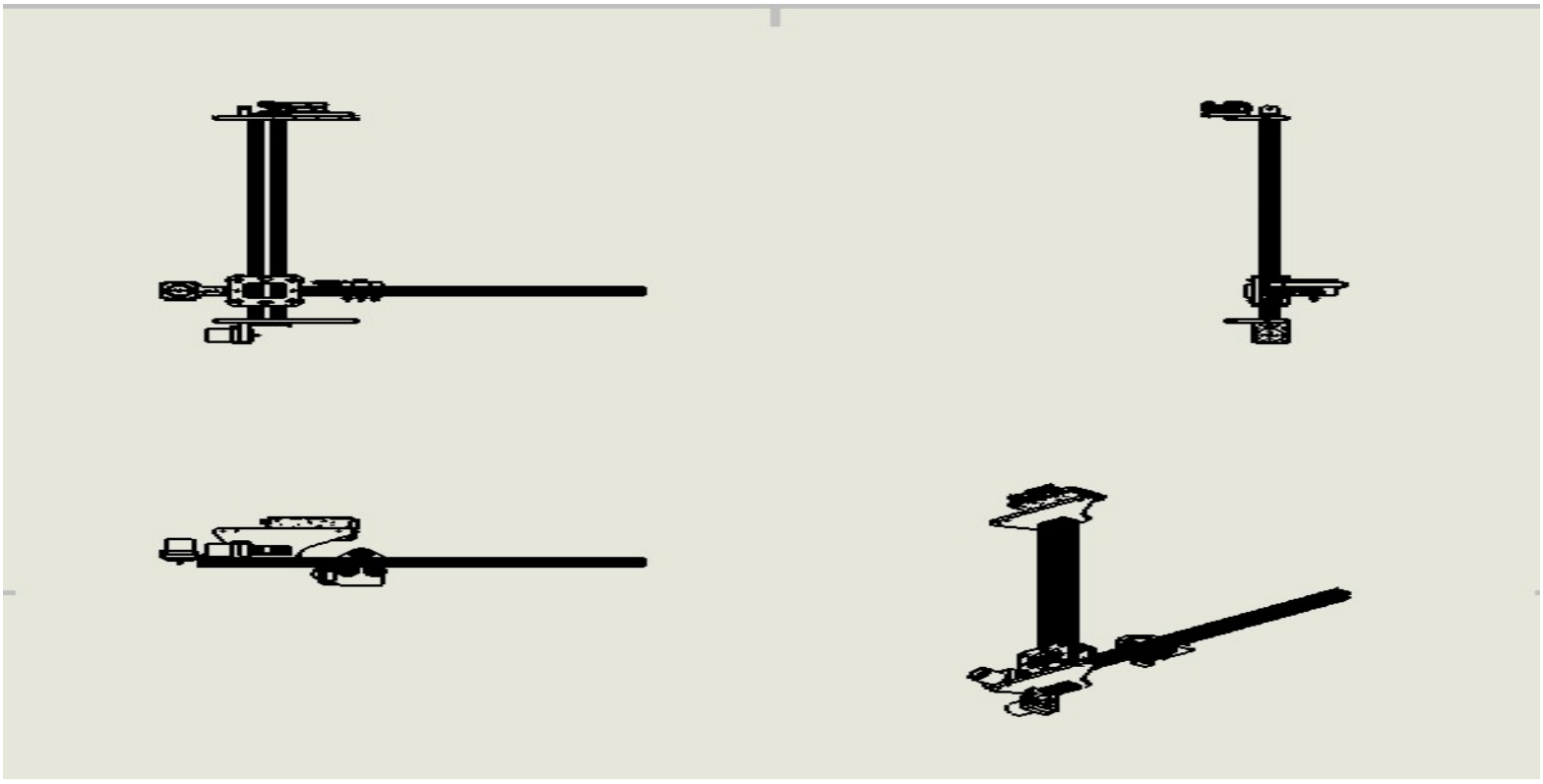


3.3. Z-axis



3.4. Full Mechanism

Figure 1 Full mechanism



4. Electrical System:

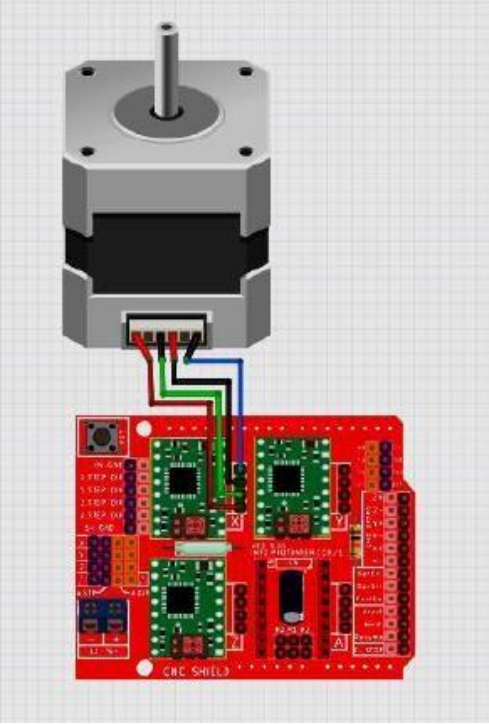
4.1. Open Loop

Components and Connections:

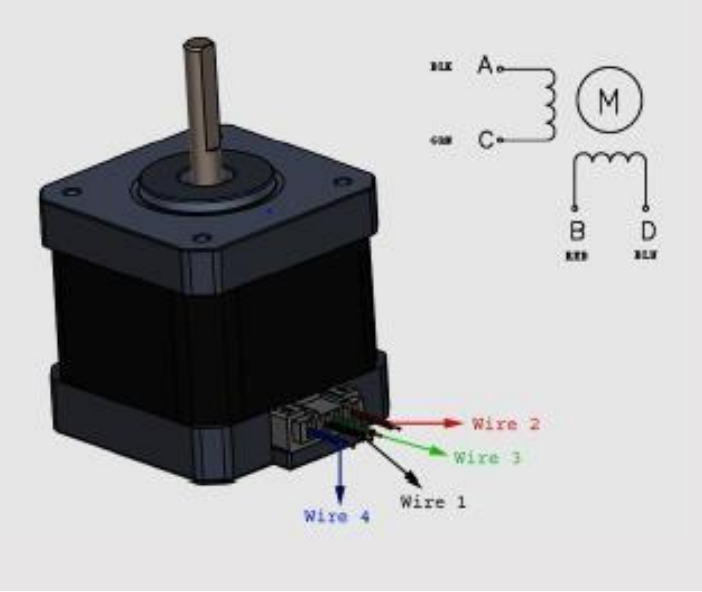
1) Stepper Motor (Nema 17)

- Stepper motors are DC motors that move in discrete steps.
- Used for accurate position control.

Connection

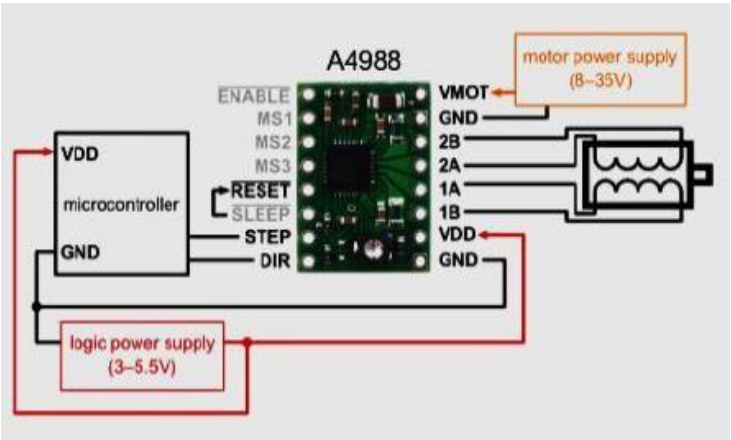


Configuration



2) Stepper Motor Drive

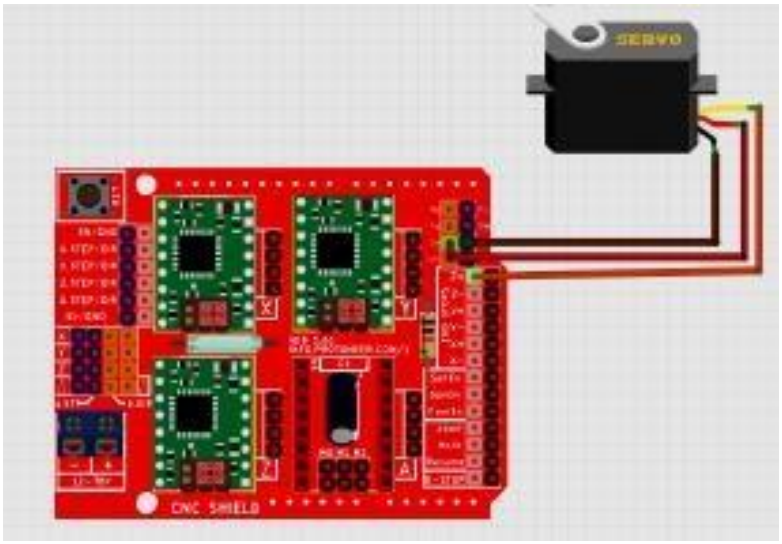
1. It can acceptable with cnc shield.
2. Main features is current limiting and overcurrent Protection.
3. To protect high current using heat sink.
4. It has different modes but to limit current put in full step mode to measure the current running through a single motor coil without clocking the STEP input. The measured current will be 0.7 times the current limit.



3) Servo Motor

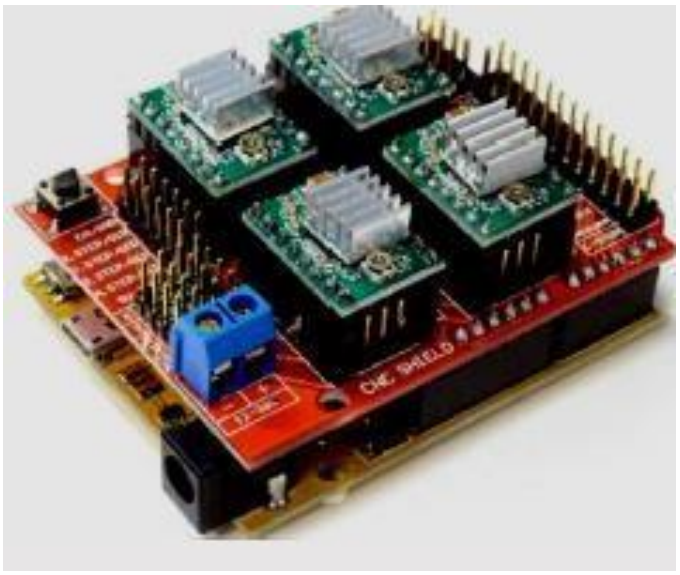
- It is a motor provides feedback control on shaft to make an accurate position and precision rotation.

Connection

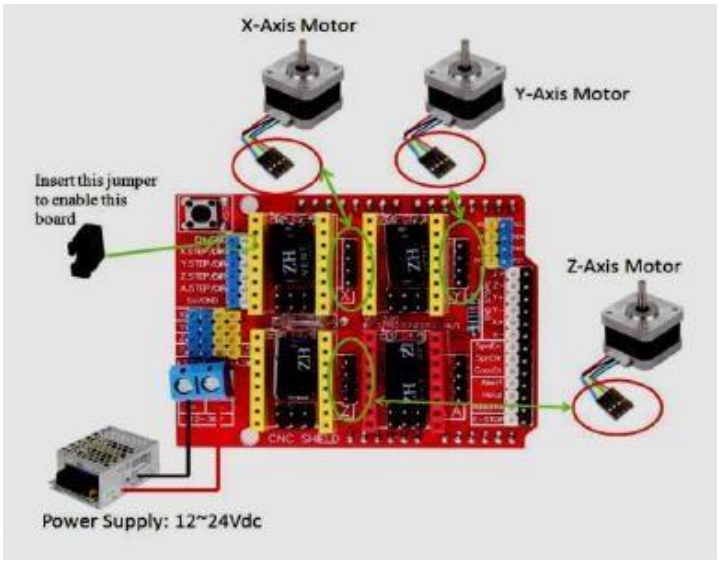


4) CNC Shield

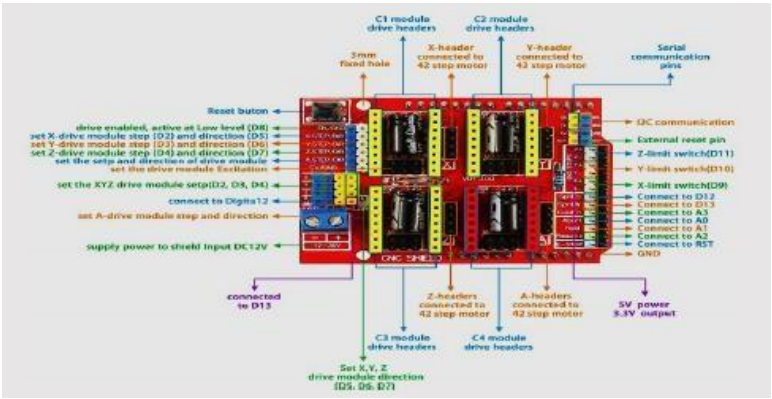
- It Can compatible 4 stepper motors and 2 stepper motors + 2 Servo motors
- It can compatible stepper driver (Nema 17).



Connection



Configuration



5) **Battery 24v**

6) **Limit Switches**

1. 2-limit switch to make Homing in x, y axes while process of writing begins it should start by active them.

7) **Microcontroller**

1. Arduino Uno

b. Schematic of Motors without cnc shield

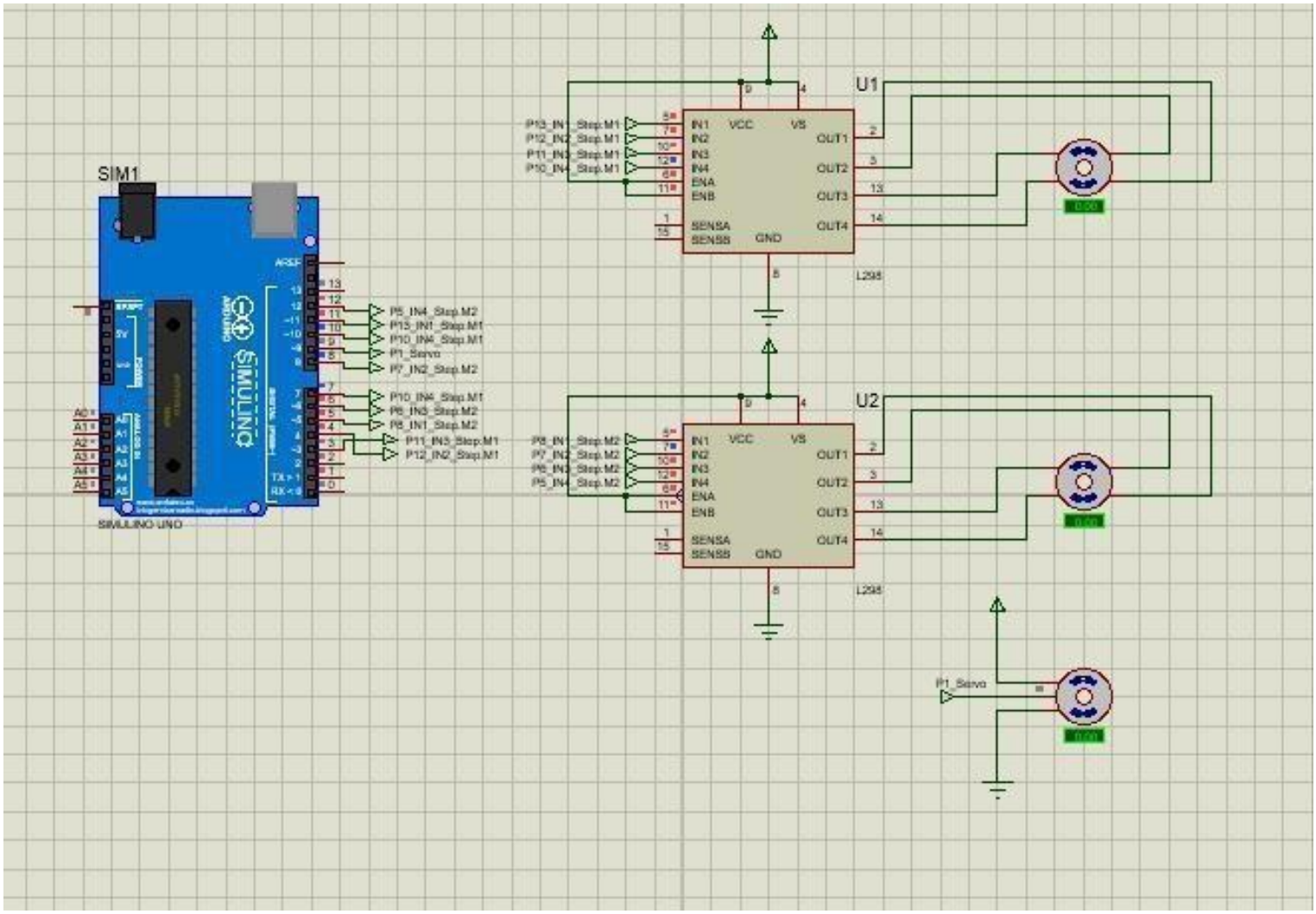
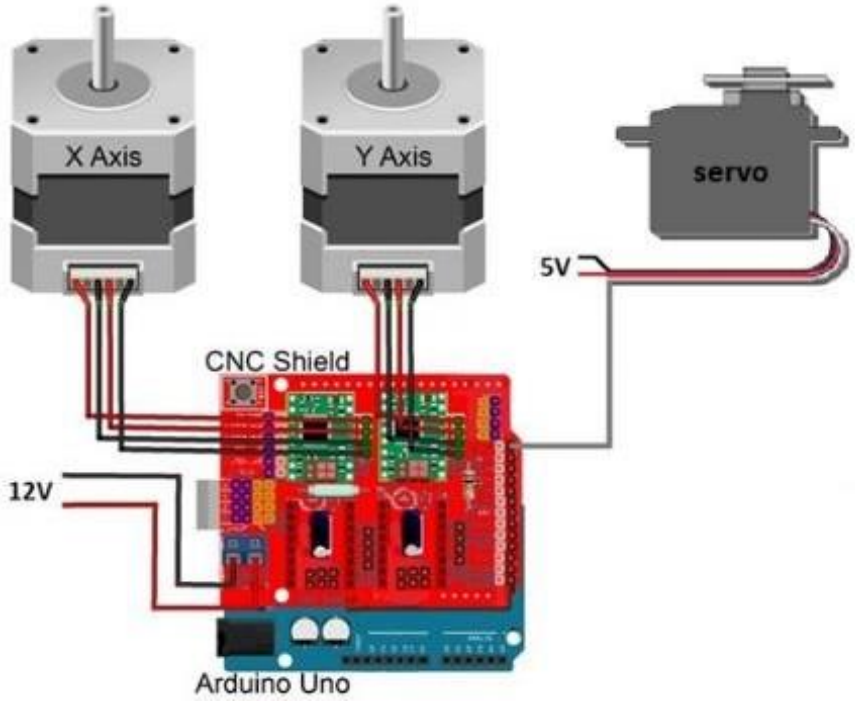


Figure 2 Schematic open loop

c. The Circuit of Open Loop



- 2 Stepper motor Nema 17 to drive 2 axis (x axis and y axis)
- 1 servo motor to drive pen (z axis)
- A4988 Drivers to control Stepper motors
- 2 limit switches to make Homing in (x axis, y axis)
- Battery 24v to power on Arduino in operation
- All of this components compatible in CNC SHIELD

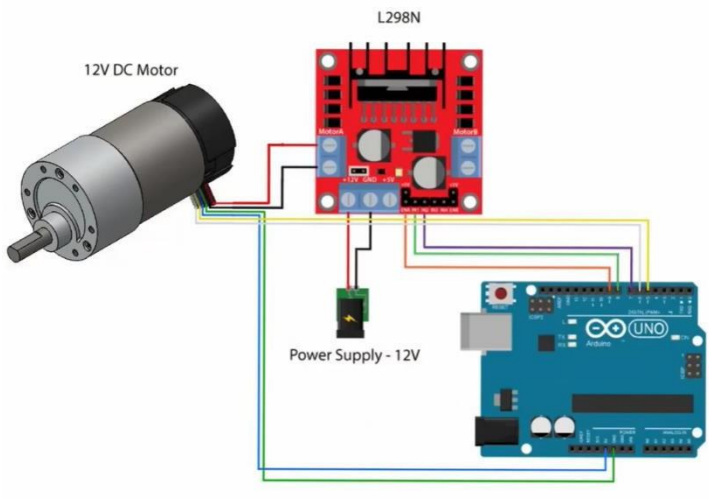
4.2. Closed Loop

Components and Connections

1) Dc Motors with Encoder

- It have a feedback control on speed and position so it can control and minimize error by itself to get accurate position with certain speed.

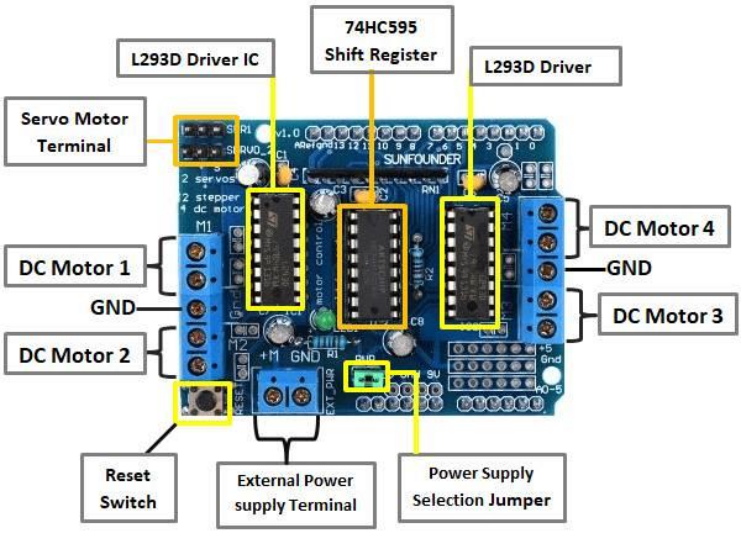
Connection

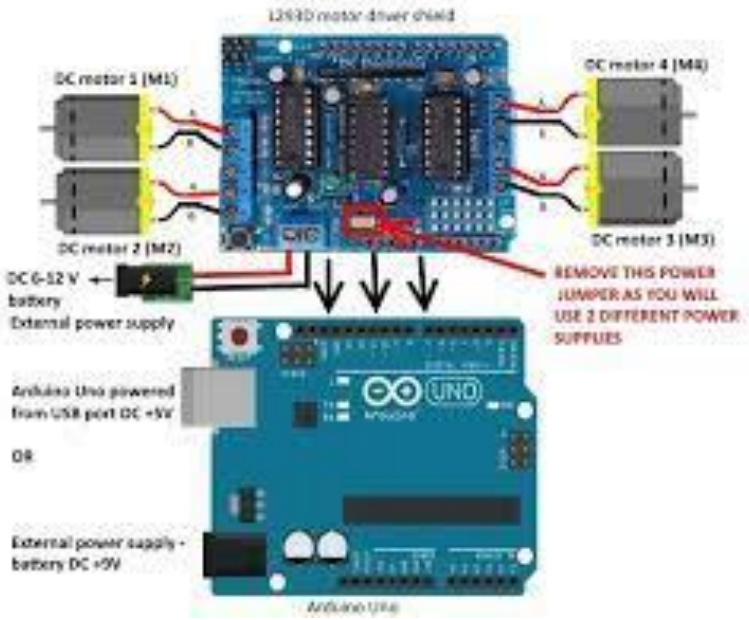


Configuration

2) L293D Motor shield Driver

- It can drive :
  1) 4 bi-directional DC motors with 8-bit speed selection(0-255)
  2) 2 stepper motors (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping.
  3) 2 servo motors
- The shield offers total 4 H-Bridges and each H-bridge can deliver up to 0.6A to the motor.
- There exists three scenarios when it comes to supplying power for the motors through shield.
  1) Single DC power supply for both Arduino and motors
  2) (Recommended) Arduino powered through USB and motors through a DC power supply
  3) Two separate DC power supplies for the Arduino and motors:





3) Battery 24 v

4) Arduino Uno Microcontroller

5) Limit Switches

- In case to make Homing in x-axisand y-axis in starting of operation

6) Servo Motors

- Previously describe before in page 25

We could replace DC motor in Z-axis with servomotor andlimit switch to feedback of its movement.(will happen in hardware but incircuit will have 3 dc motor)

4.3. Schematic Closed loop system

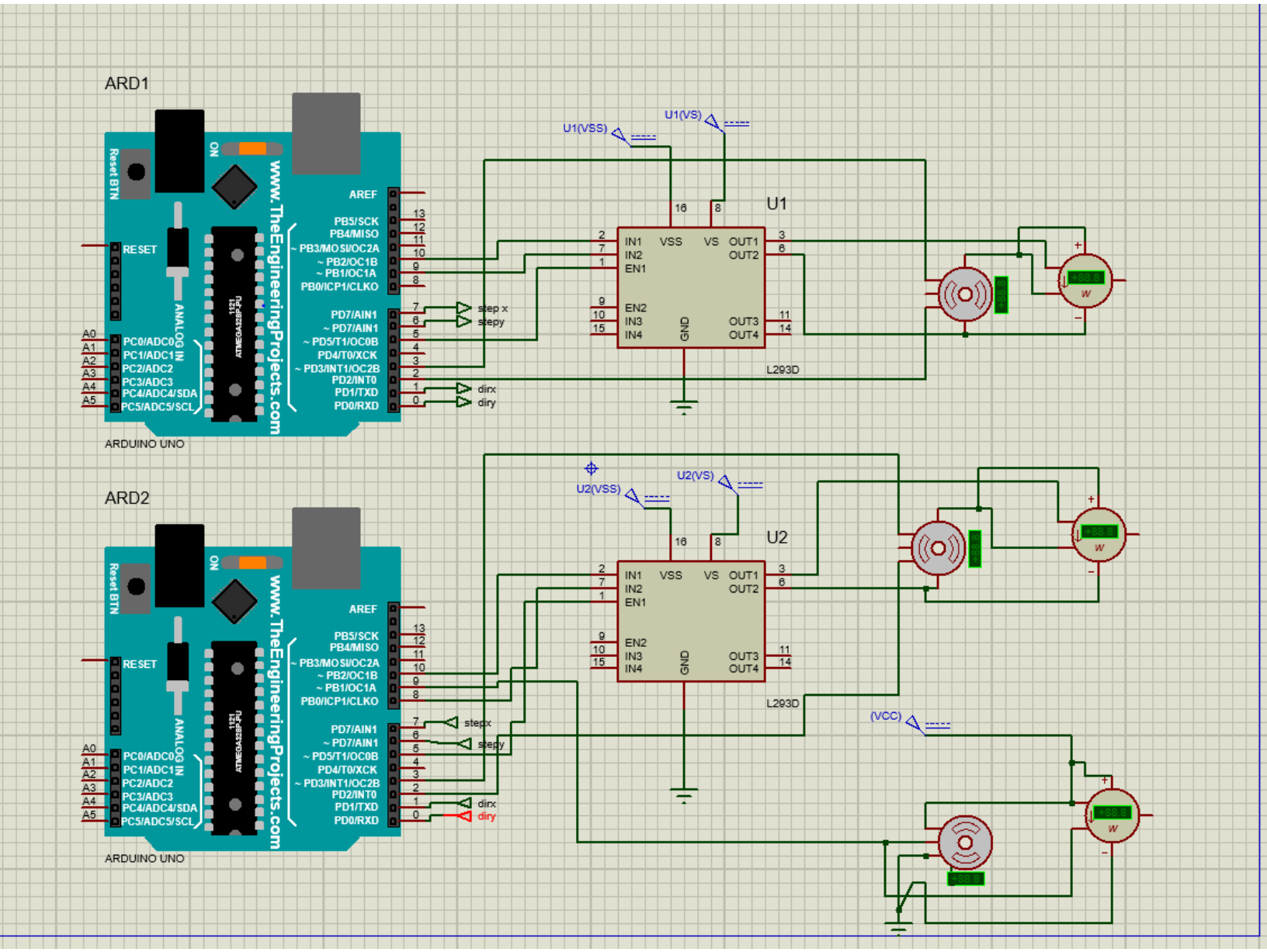
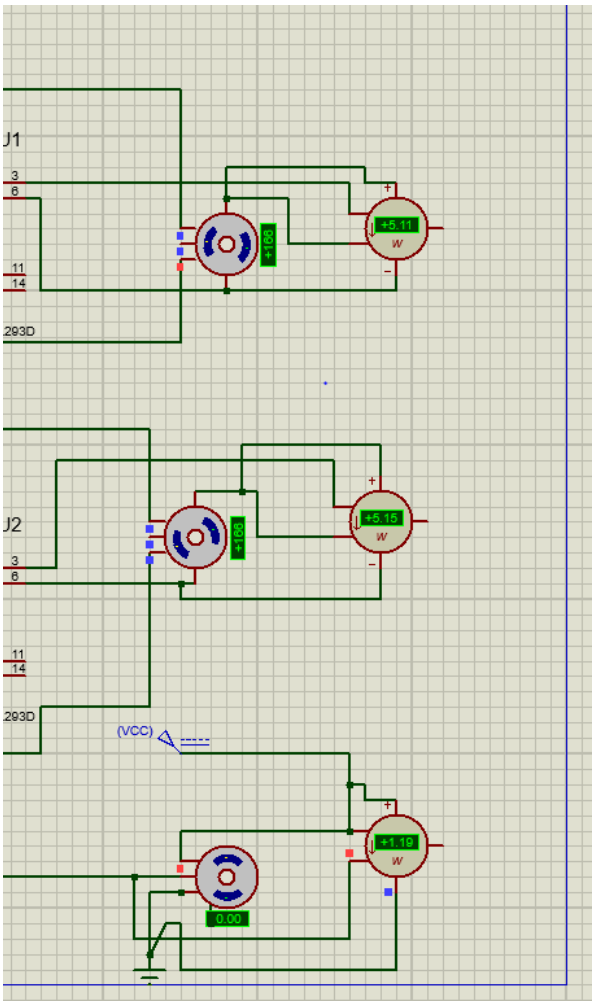
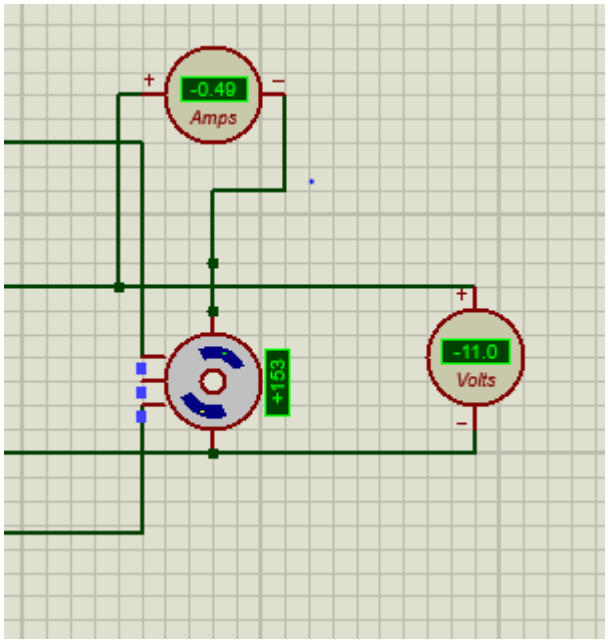


Figure 3 schematic closed loop

4.4. The power consumption in closed loop





- Volt in Dc motors:11v
- Current in Dc motor:0.47A
- Power = 0.47*11=5.12 watt
- power in servo = 1.19watt

Modeling and Simulation

1. Open loop system

1.1. Model on Matlab

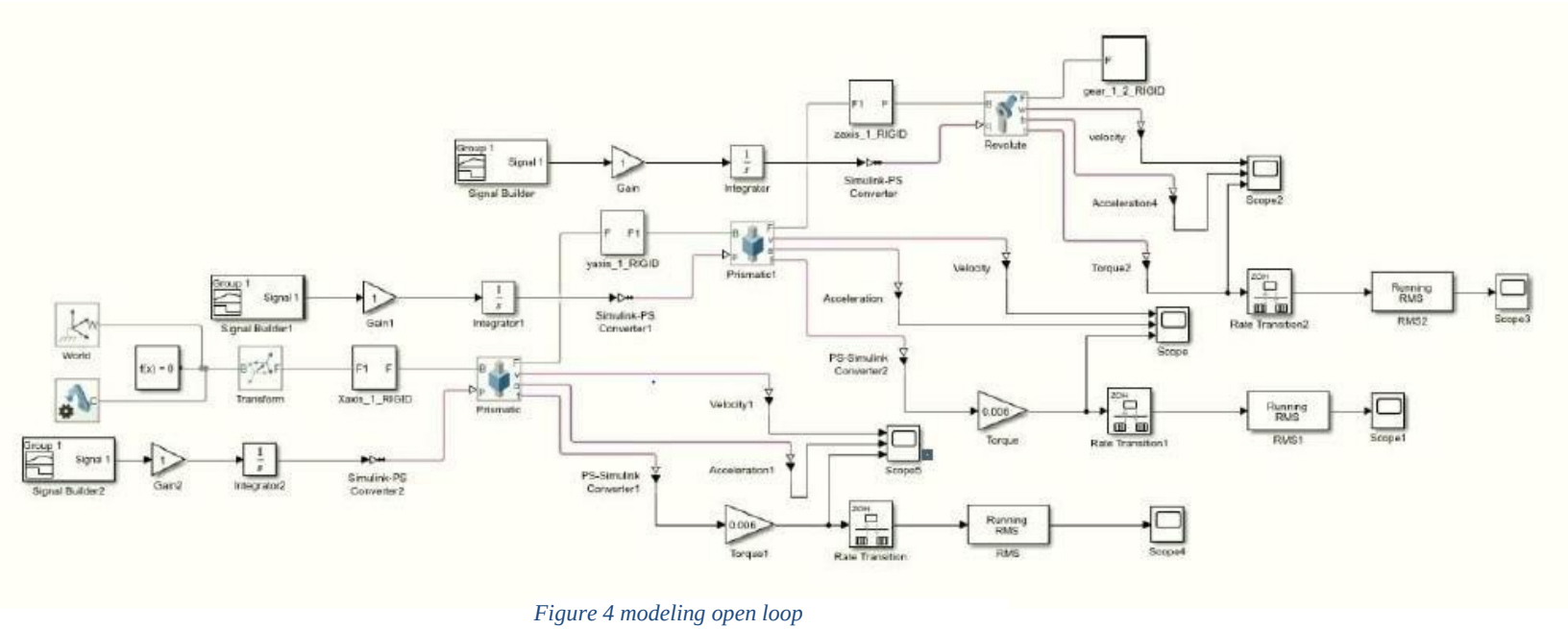
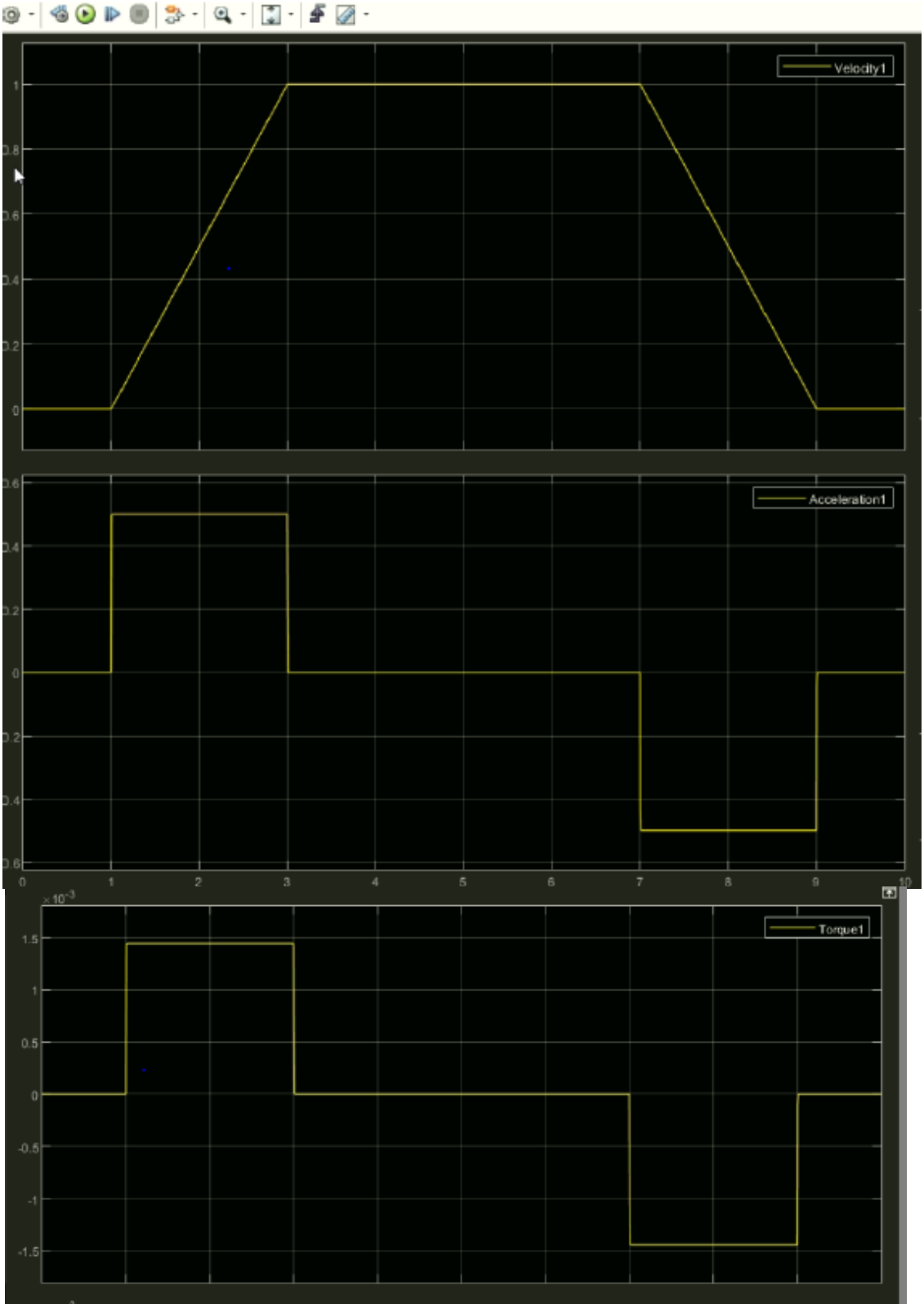


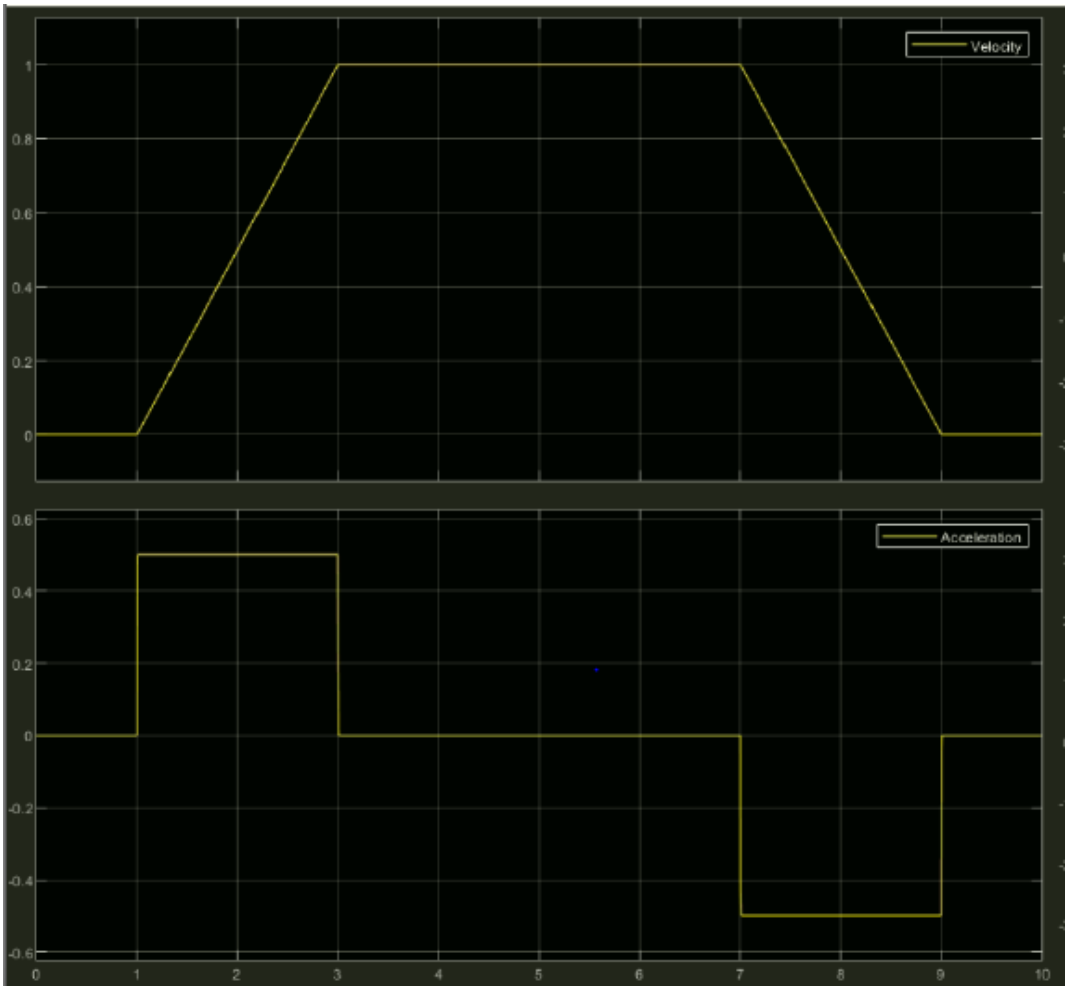
Figure 4 modeling open loop

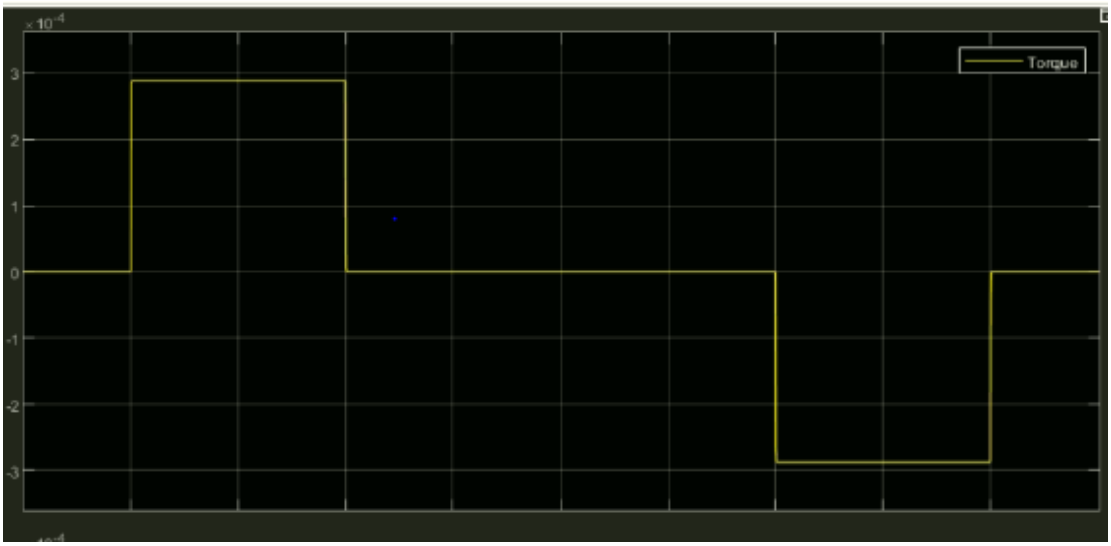
1.2. Actuator Sizing

X-axis

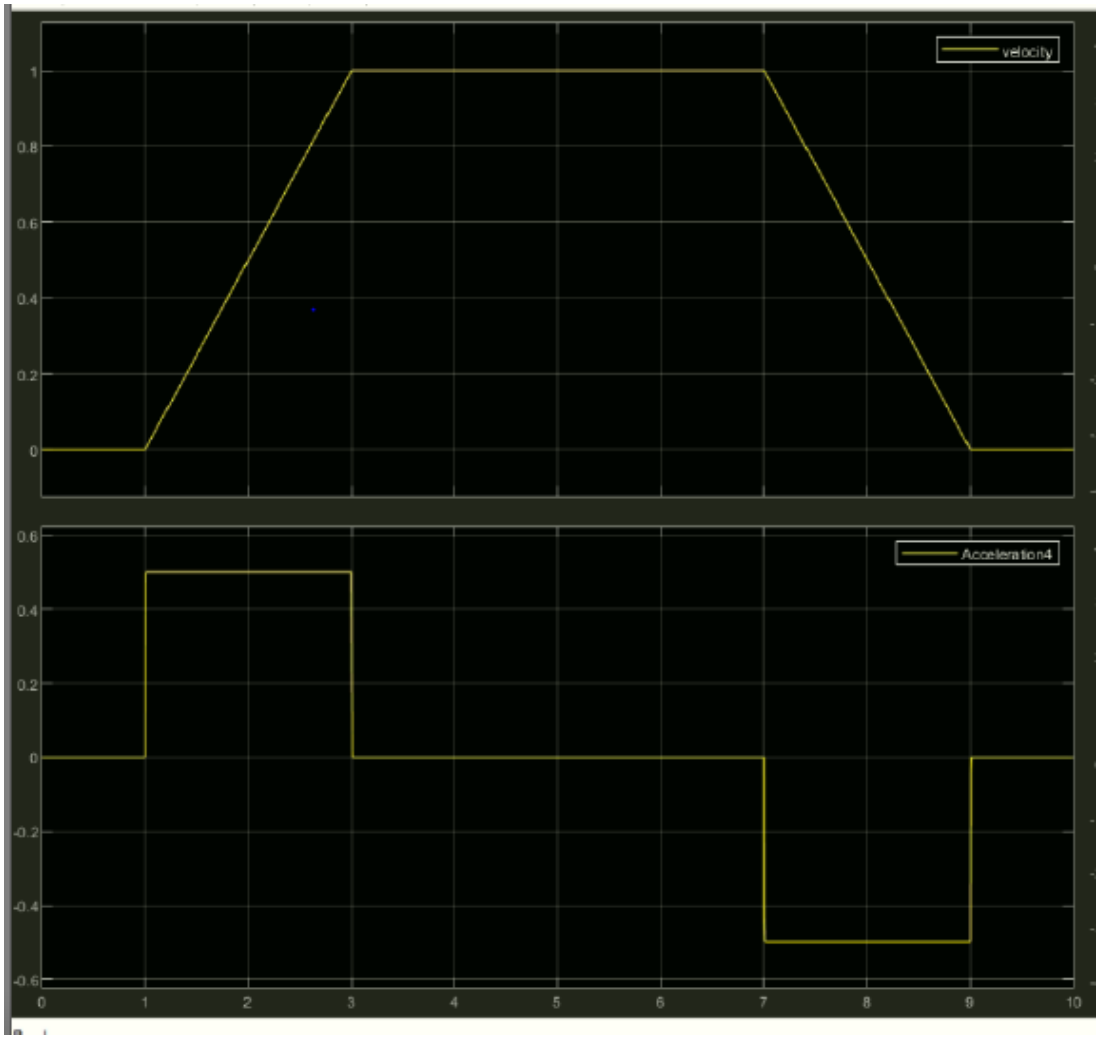


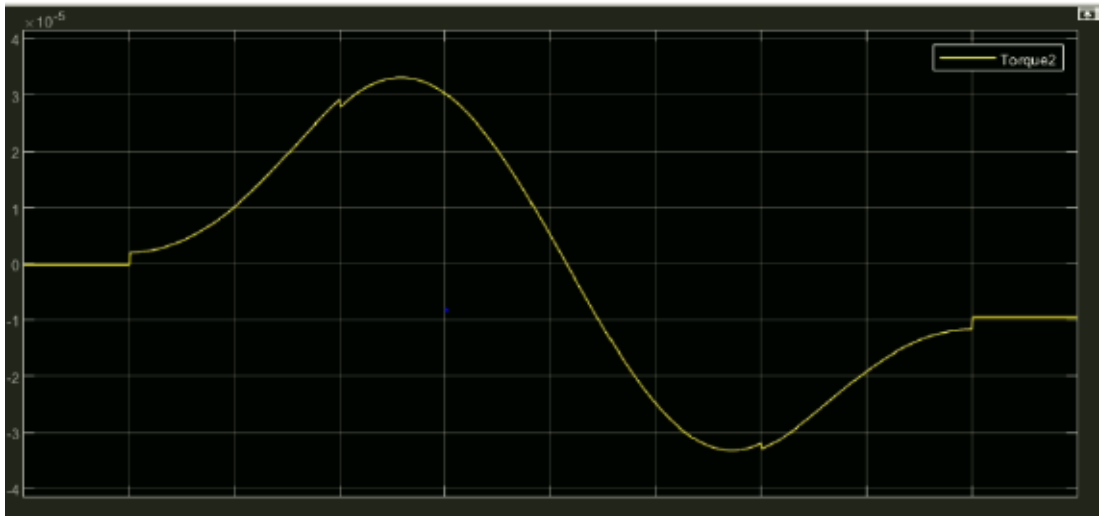
Y-axis





Z-axis





- **Due to Calculations on Matlab the suitable Motor is Nema 17**

1.3. Simulation

Scan this

2. Closed loop system

2.1. Model in Matlab

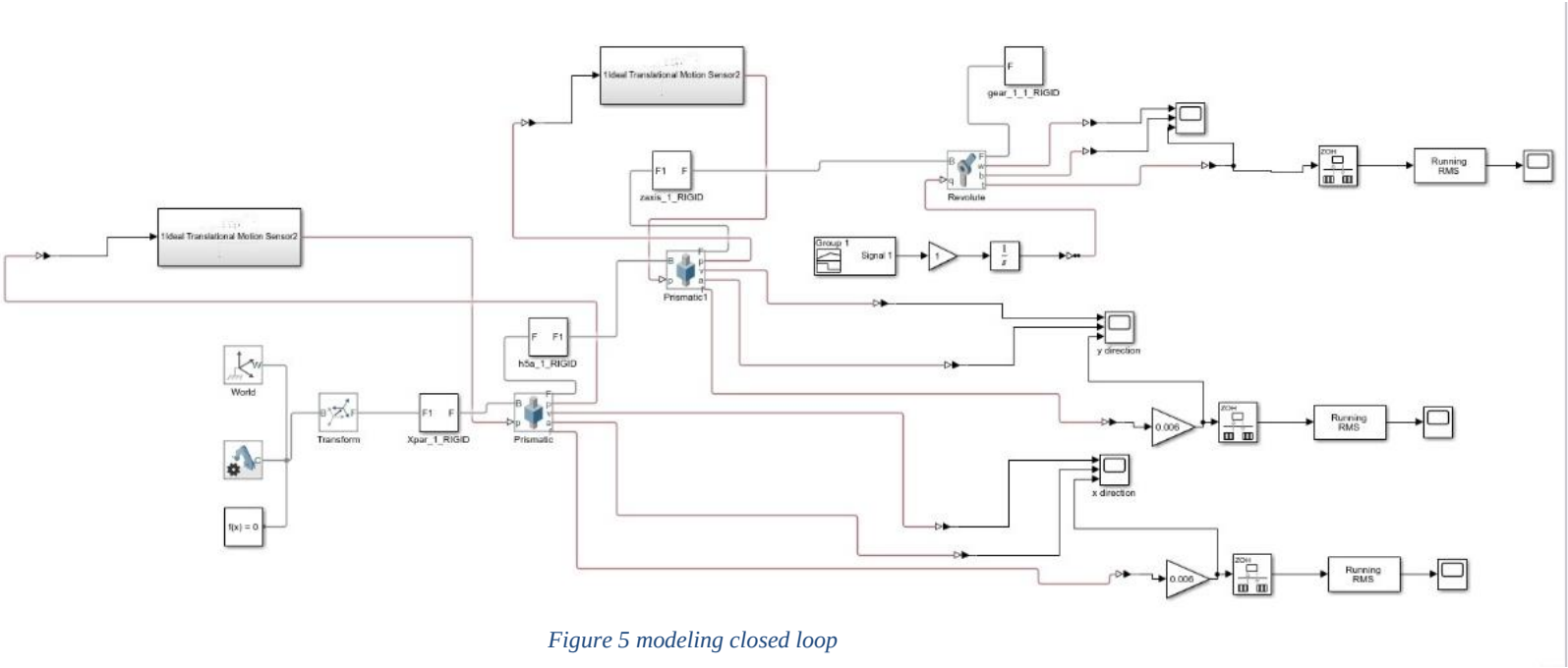
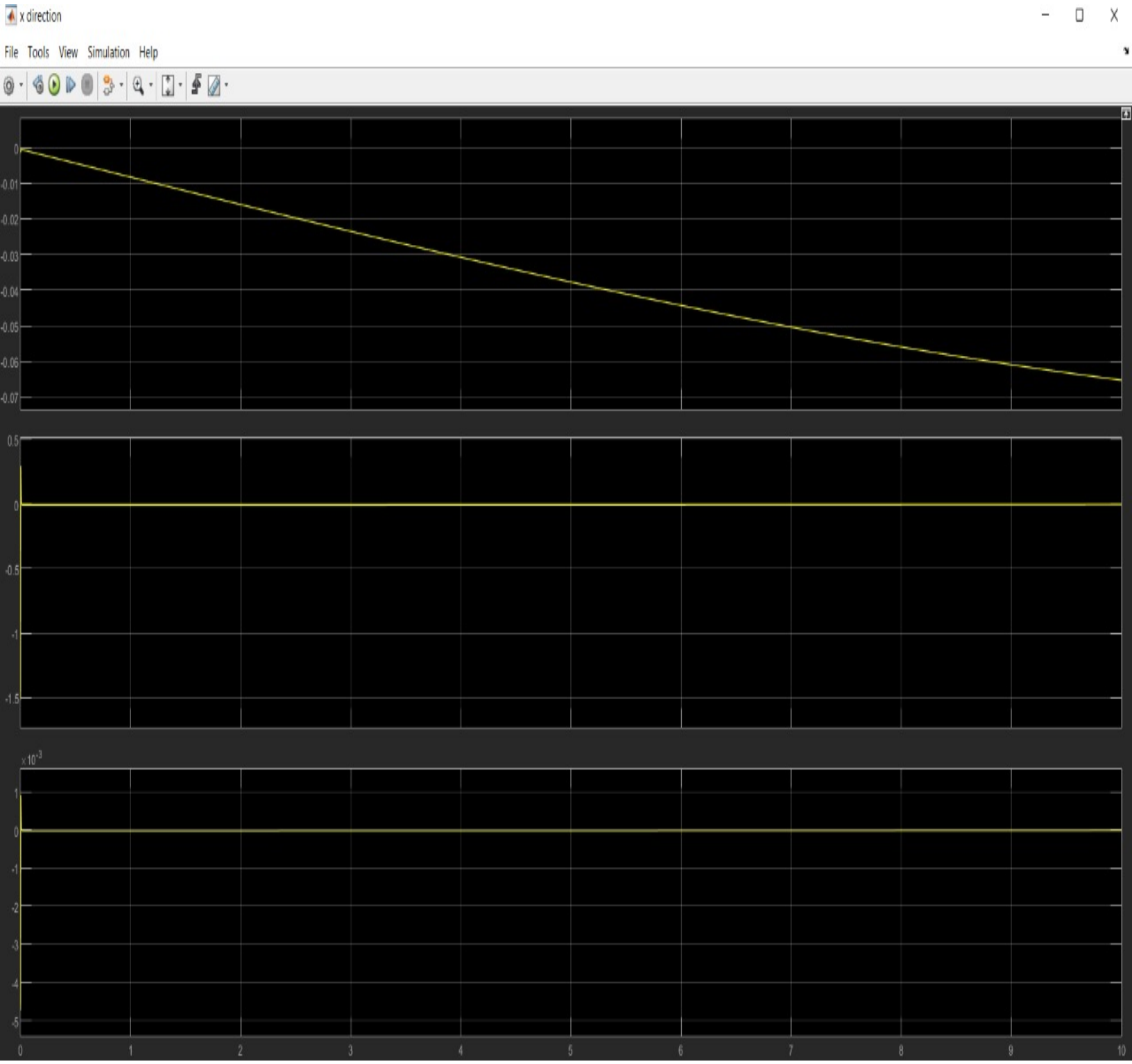


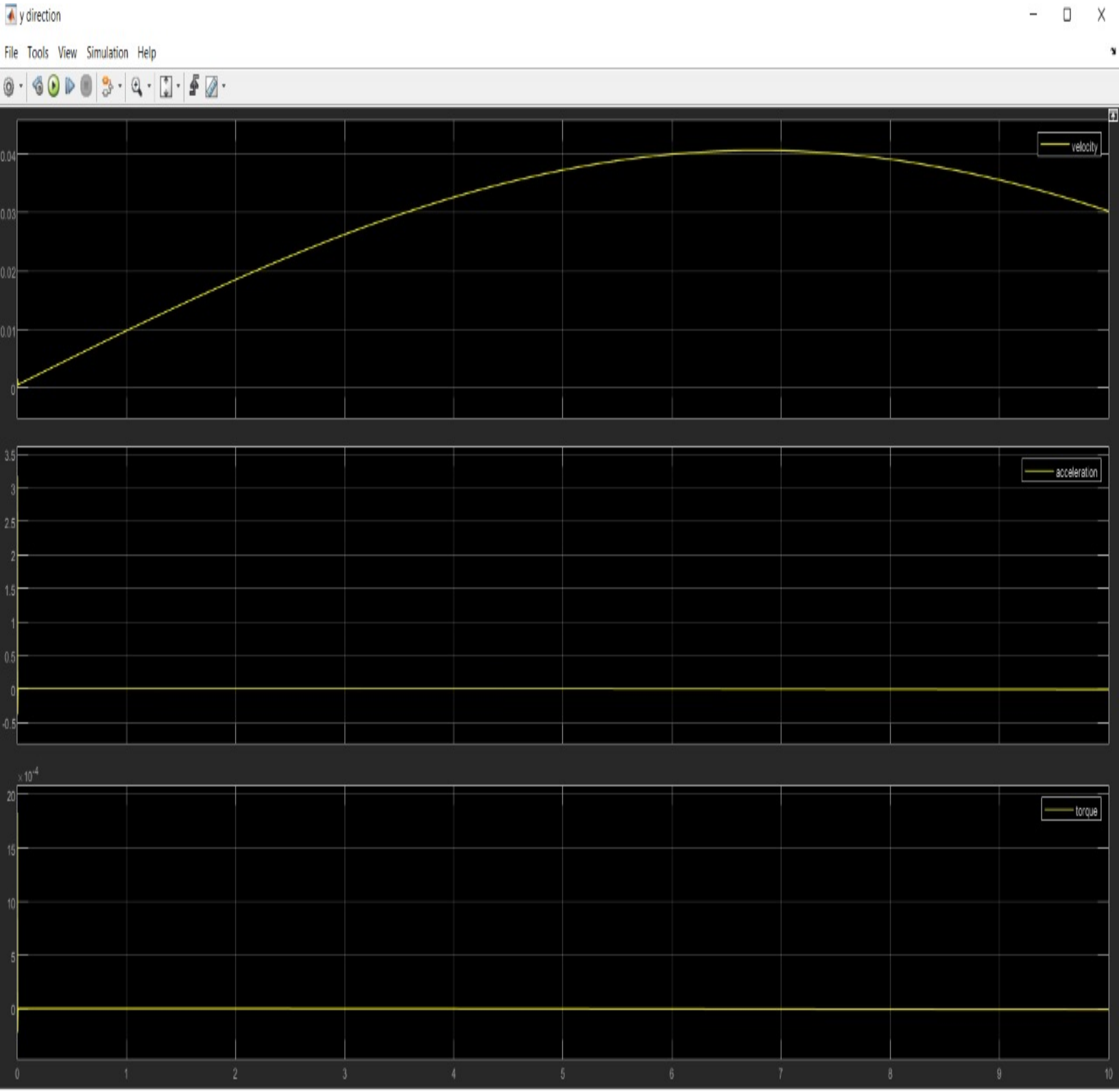
Figure 5 modeling closed loop

2.2. Actuator sizing

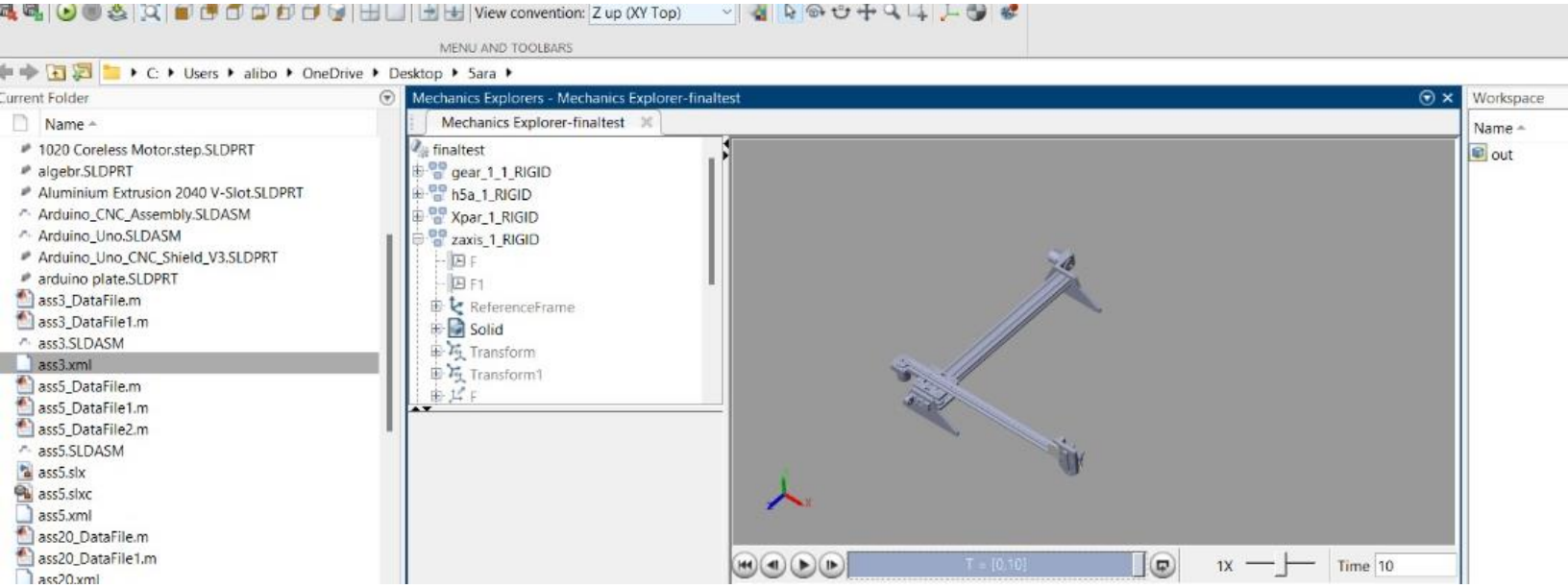
X –Axis



Y axis


y direction
File Tools View Simulation Help
velocity
acceleration
torque
0.04
0.03
0.02
0.01
0
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
20
15
10
5
0
0 1 2 3 4 5 6 7 8 9 10

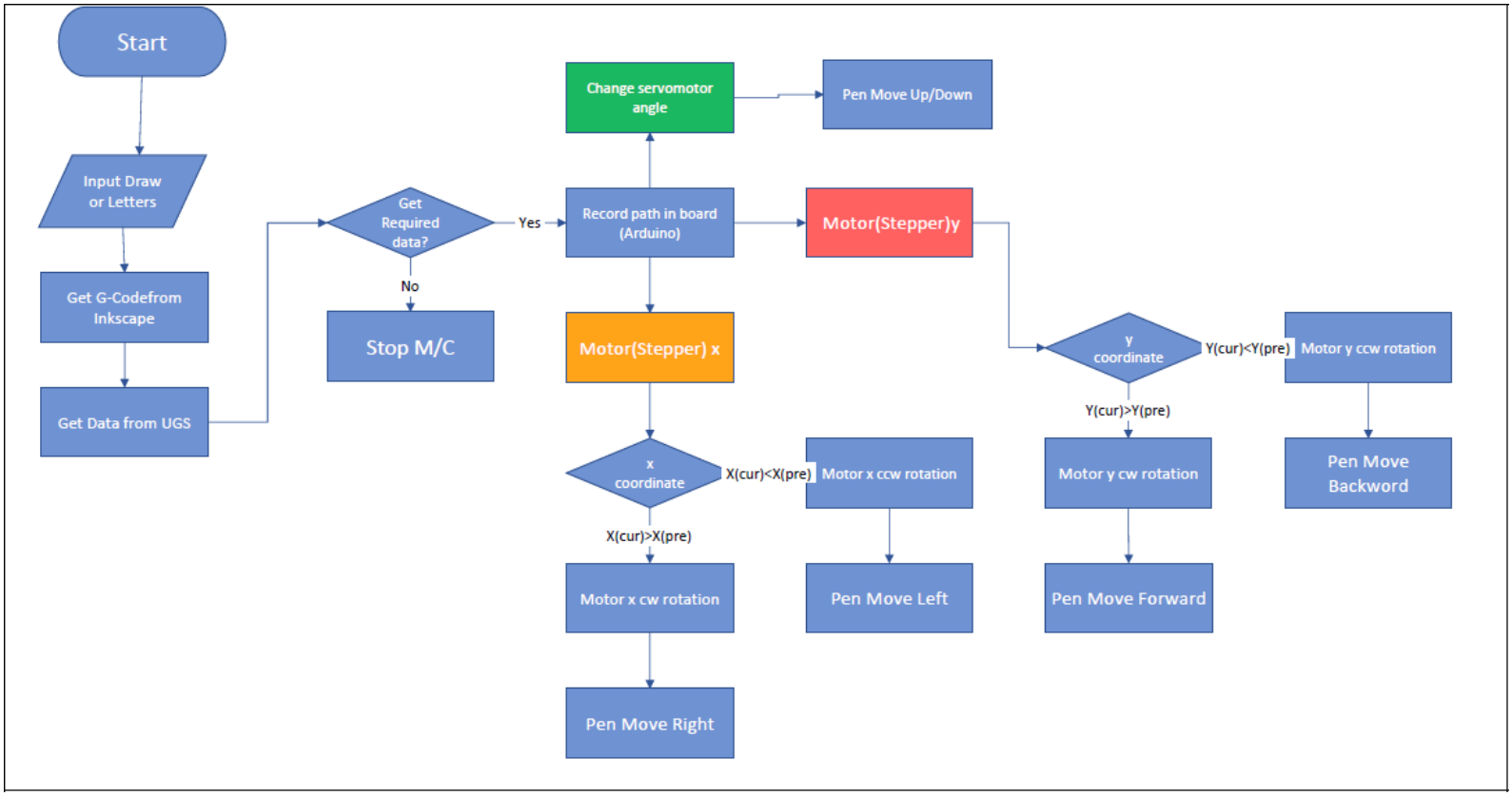
2.3. Simulation



Software System

1. Flow Chart

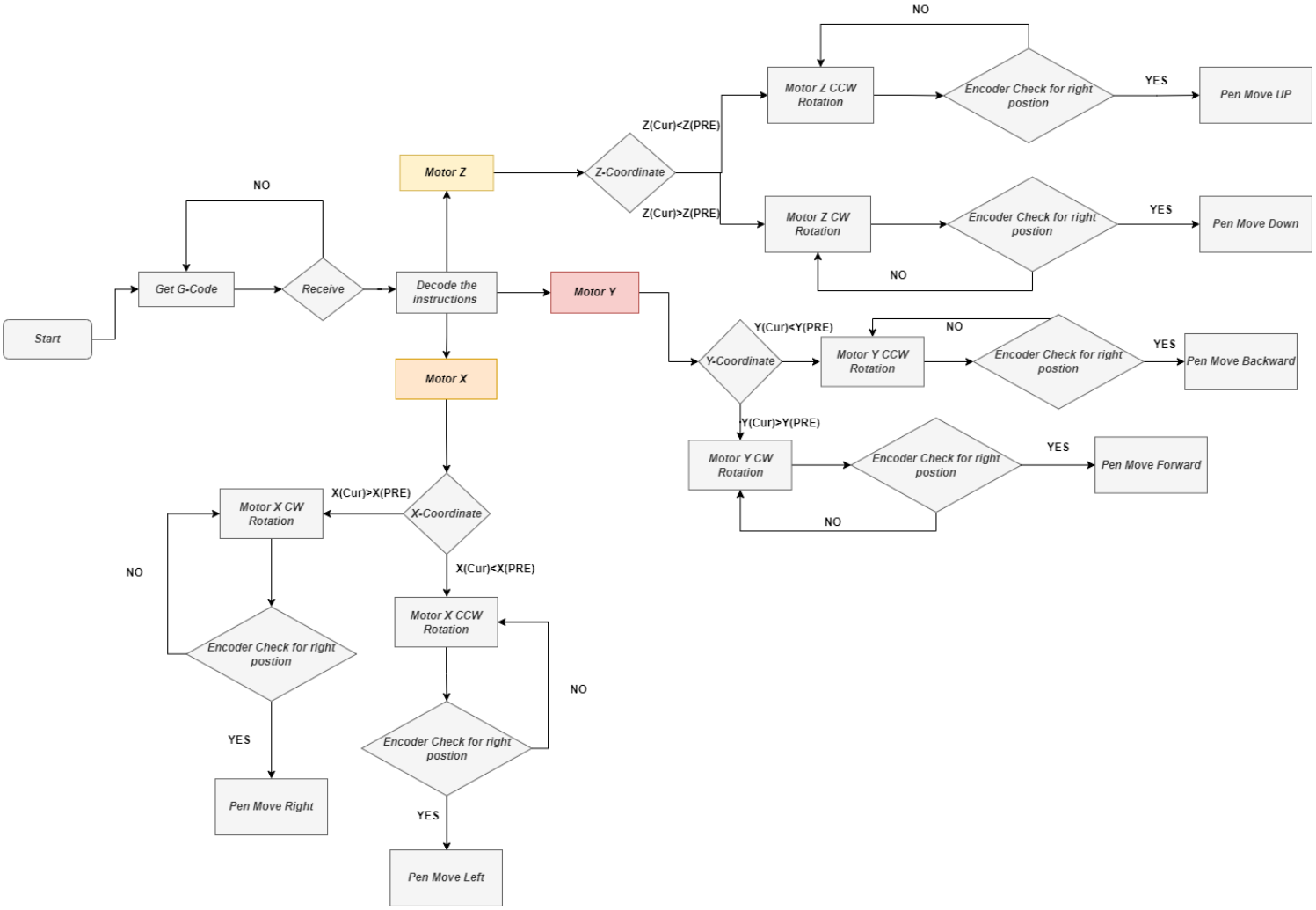
1.1. Open Loop Phase



- Open loop system is simple but not accurate in control of position and easier to affect with external effects and noises.

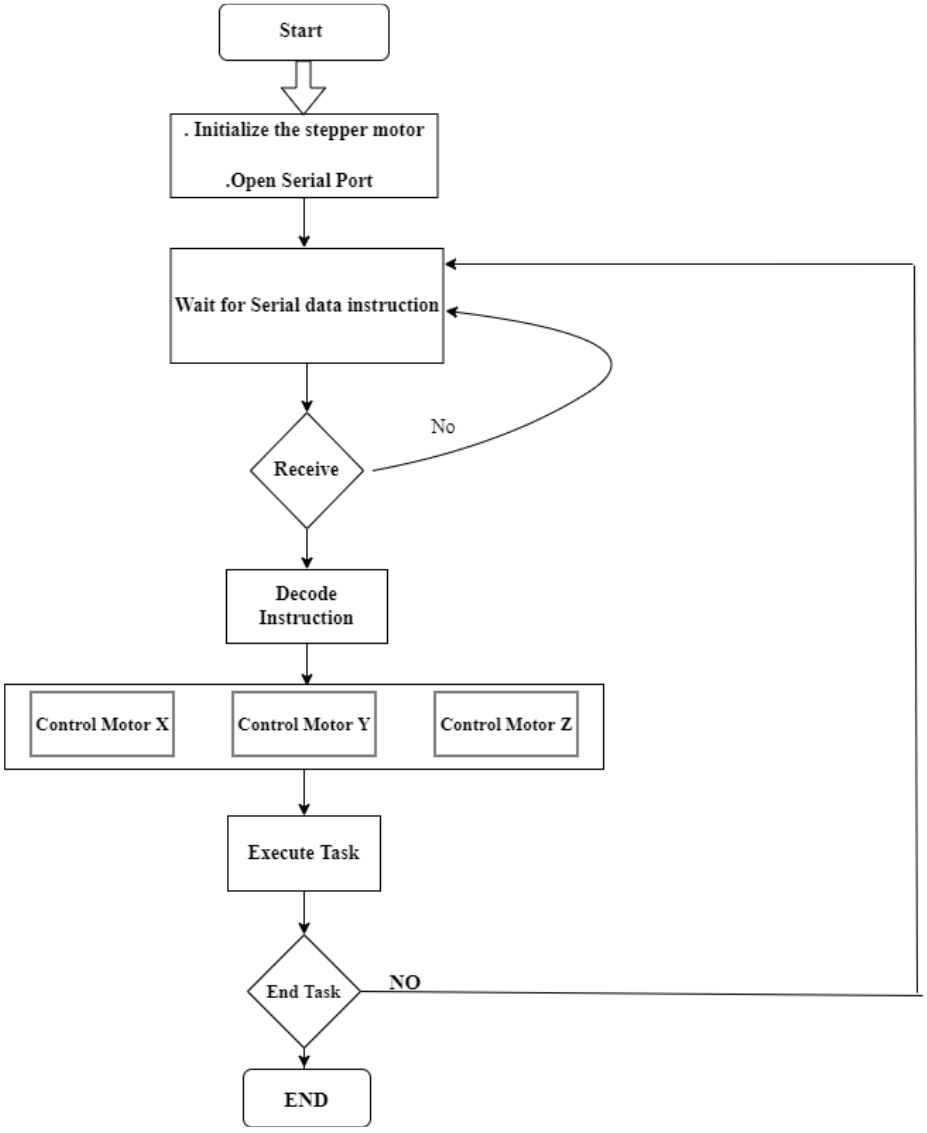
- Open loop system have not feedback control on it so it have not a method to 100% that it get the certain output.

1.2. Closed Loop Phase



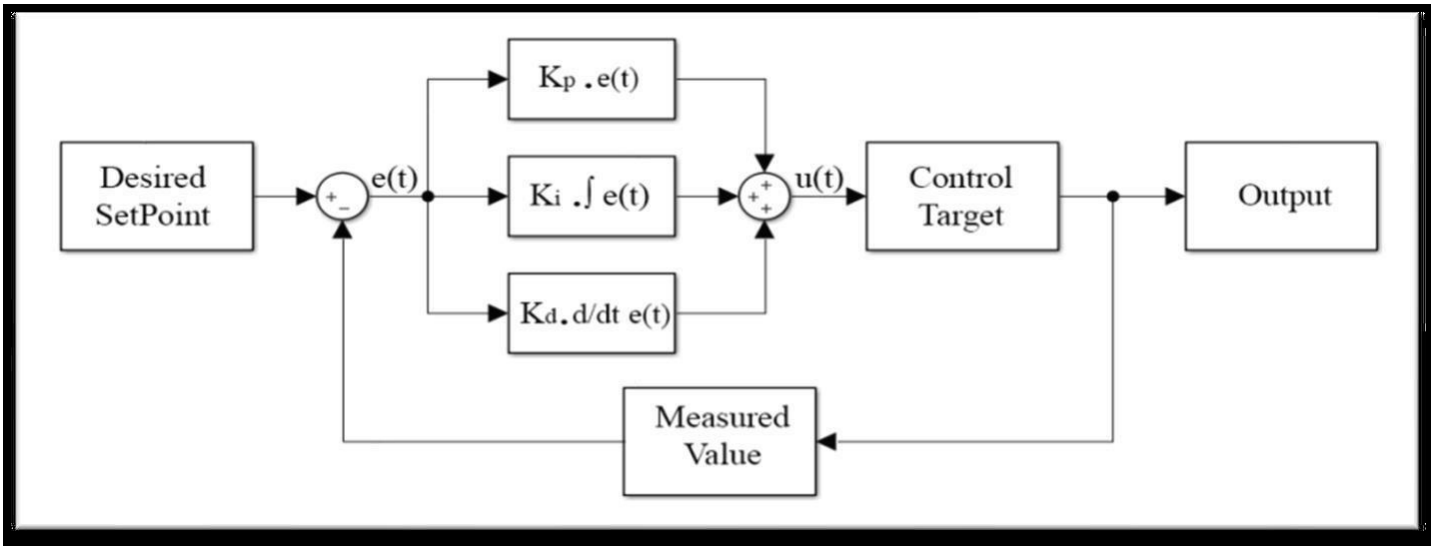
- When G-code is applied the motors will move to the desired instructions if not the motors will stop.
- The moving of motor will be direct CW while the point of coordinate less than the target point in otherwise the moving will be opposite CCW while the point of coordinate more than the target point……

a) The flow chart of application execution. At start, power supply and computer are turned on.
b) After that all motors are initialized to its zero position.
c) These zero positions are given through software. The circuit board is ready to accept instructions from computer.
d) These instructions are in the form of G-codes. It will wait still instructions to be received.
e) After instructions are received, it starts to decode it into its own language that is in the voltage and current form. As per instruction, when task is completed, it is the end of the flow of execution.

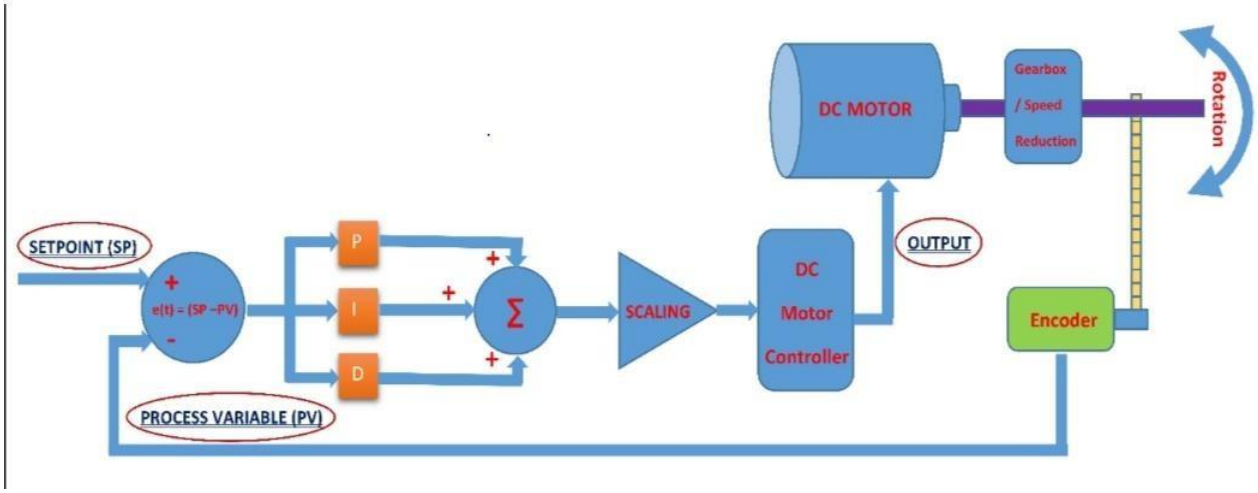


2. PID Controller

- It Can be configured the values and using it on Matlab also on Arduino by certain Library(PID_vl.h)
  - Kp: Proportional Constant
    Function: Decrease Rising time, used for present values
  - Ki: Integral Constant
    Function: Decrease steady state error, used for past values
  - Kd : Derivative Constant
    Function: Decrease Oscillations, used for future values



2.1. PID Configuration in Closed Loop Phase



- We have one of two Microcontroller Arduino Uno will act as a DC servo Controller **in case of servo motor and Dc with magnetic encoder** used to perform PID to X,Y axes.
- Motor will be driven by speed or position but with this PID controller, the setpoint are (**step**) + (**direction**) signals from **Arduino Uno** which has GRBL firmware pre-installed.

- SETPOINT - SP: They are (**(step) + (direction) signals in x-axis)** + **((step) + (direction) signals in y-axis)** signals that are sent from Arduino Uno R3 to Arduino Mega 2560, and the Arduino Uno has GRBL firmware pre-installed.
- PROCESS VARIABLE - PV: The measured feedback value from quadrature magnetic encoders to Arduino Mega 2560.
- OUTPUT: The PWM signals from Arduino L293D Motor Shield (controlled by Arduino Mega 2560) to printer DC motors.

2.2. PID Tuning

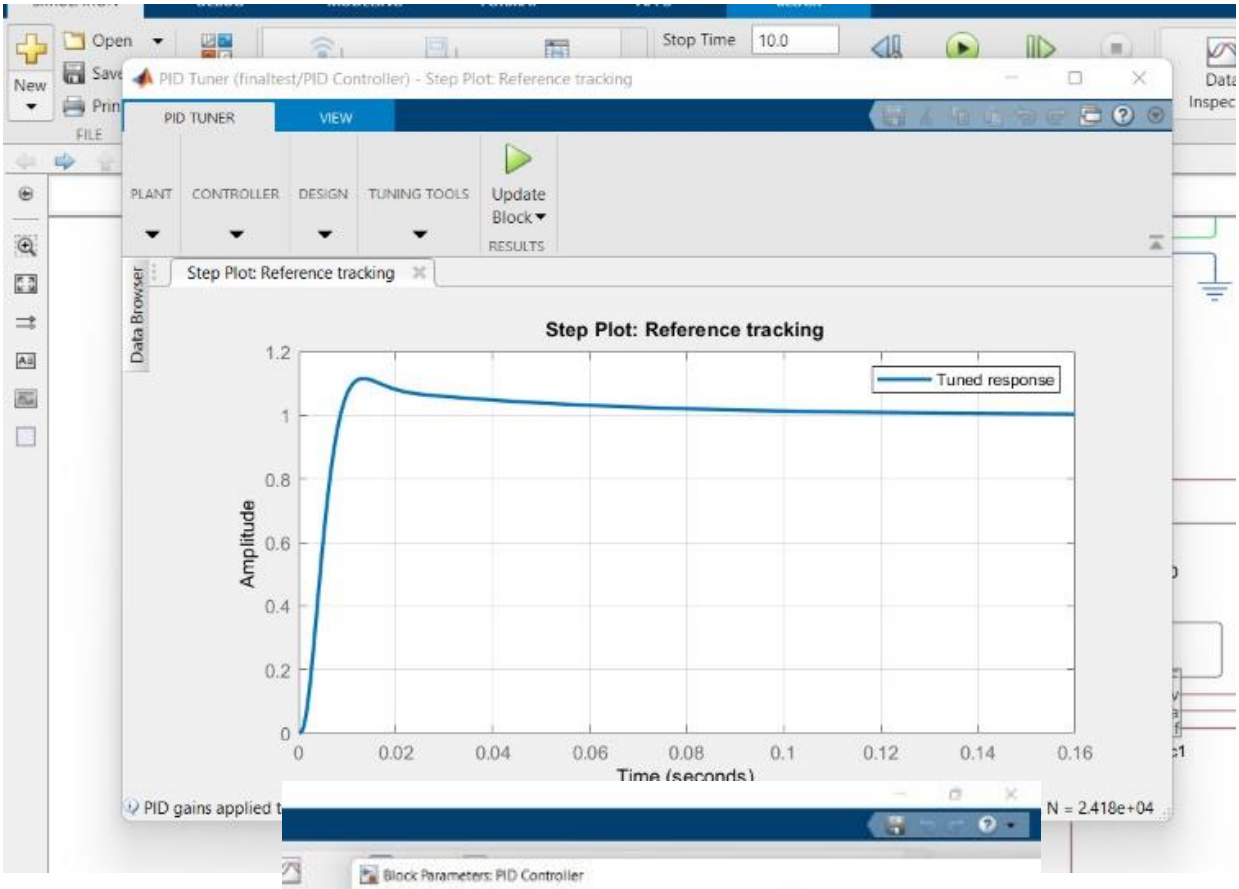


Figure 6 Graph of tuning

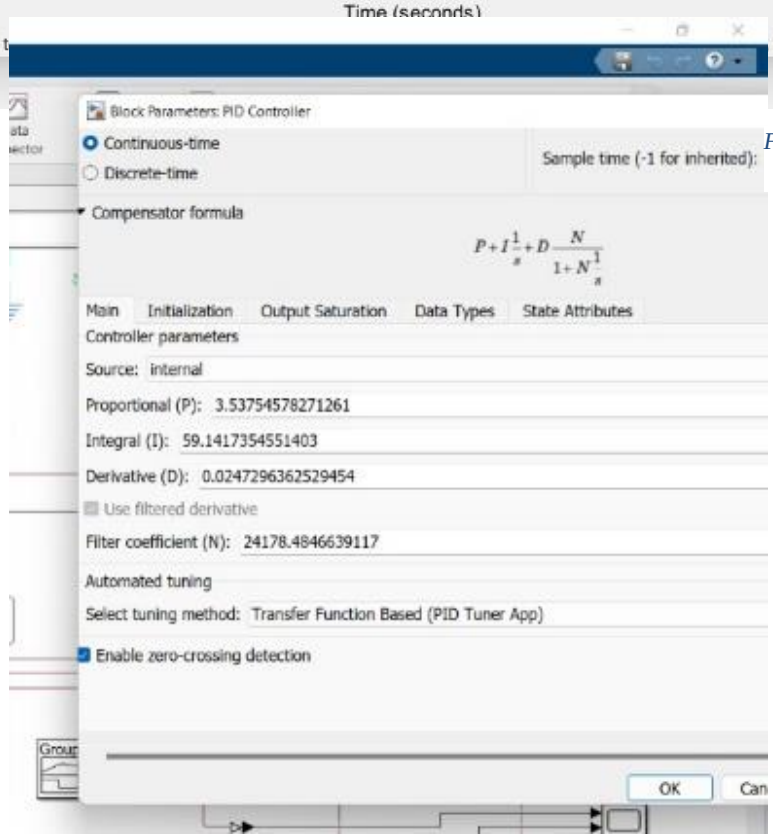


Figure 7 Parameters of Tuning

3. Code

```
#include "FlexiTimer2.h"

// PID library
#include <PID_v1.h>

// AFMotor library
#include <AFMotor.h>

// Quadrature Encoder Library
#include "Encoder.h"

// Create the motor driver instances
AF_DCMotor motorX(1, MOTOR12_8KHZ);
AF_DCMotor motorY(2, MOTOR12_8KHZ);

// Set up pins for the quadrature encoders - Arduino MEGA2560 has 6 interrupt pins.
#define EncoderX_ChannelA   18  // Interrupt 5
#define EncoderX_ChannelB   22
#define EncoderY_ChannelA   20  // Interrupt 3
#define EncoderY_ChannelB   24

// Set up STEP & DIRECTION pins for X and Y axis
#define STEP_XPIN           19  // Interrupt 4
```

```
#define STEP_YPIN           21  // Interrupt 2
#define DIR_XPIN            23
#define DIR_YPIN            25

// For calculating the actual movements
#define STEPSPERMM_X      39    // STEP/mm is used in the GRBL for DC motor X axis.
#define DEADBW_X          4.5     // Deadband width = 4.5 --> Acceptable error for positioning in mm: 0.18mm.

#define STEPSPERMM_Y      41   // STEP/mm is used in the GRBL for DC motor Y axis.
#define DEADBW_Y          19.2    // Deadband width = 19.2 --> Acceptable error for positioning in mm: 0.1mm.


// Set up Input
double INPUT_X;
double INPUT_Y;

double OLD_INPUT_X = 0;
double OLD_INPUT_Y = 0;

// Set up Actual value
double ACTUAL_X_MM;
double ACTUAL_Y_MM;
```

```
double OLD_ACTUAL_X_MM;
double OLD_ACTUAL_Y_MM;


// PID controller constants
double KP_X = 20.0;      // P for X motor
double KI_X = 0.03;      // I for X motor
double KD_X = 0.01;      // D for X motor

double KP_Y = 9.0;       // P for Y motor
double KI_Y = 0.02;      // I for Y motor
double KD_Y = 0.01;      // D for Y motor

// The Output variable motor speed to the motor driver
double OUTPUT_X;
double OUTPUT_Y;

double OLD_OUTPUT_X = 0;
double OLD_OUTPUT_Y = 0;

// Setpoint
double SETPOINT_X = 0;
double SETPOINT_Y = 0;
```

```
double OLD_SETPOINT_X = 0;
double OLD_SETPOINT_Y = 0;

double ERROR_X = 0;
double ERROR_Y = 0;

// PID controller
PID myPID_X(&INPUT_X, &OUTPUT_X, &SETPOINT_X, KP_X, KI_X, KD_X, DIRECT);
PID myPID_Y(&INPUT_Y, &OUTPUT_Y, &SETPOINT_Y, KP_Y, KI_Y, KD_Y, DIRECT);

// Setup optical encoders
Encoder XEncoder(EncoderX_ChannelA, EncoderX_ChannelB);
Encoder YEncoder(EncoderY_ChannelA, EncoderY_ChannelB);

void setup()
{
  // For debugging
  Serial.begin(115200);

  pinMode(STEP_XPIN, INPUT);
  pinMode(STEP_YPIN, INPUT);
  pinMode(DIR_XPIN, INPUT);
  pinMode(DIR_YPIN, INPUT);
```

```
    // The stepper simulator
    attachInterrupt(4, doXstep, RISING);  // PIN 19 (Interrupt 4) - Interrupt X step at rising edge pulses
    attachInterrupt(2, doYstep, RISING);  // PIN 21 (Interrupt 2) - Interrupt Y step at rising edge pulses

    // Outpout PWM limits
    myPID_X.SetOutputLimits(-255,255);
    myPID_Y.SetOutputLimits(-255,255);

    // Compute output every 1ms
    myPID_X.SetSampleTime(1);
    myPID_Y.SetSampleTime(1);

    // Setup PID mode
    myPID_X.SetMode(AUTOMATIC);
    myPID_Y.SetMode(AUTOMATIC);

    // Apply PID every 1ms by FlexiTimer2
    FlexiTimer2::set(1, 1.0/1000, doPID);
    FlexiTimer2::start();
  }

  void loop()
  {
```

```
void loop()
{
  // Read X and Y axis optical encoders
  INPUT_X = XEncoder.read();
  INPUT_Y = YEncoder.read();

  // Calculating the error
  ERROR_X = (INPUT_X - SETPOINT_X);
  ERROR_Y = (INPUT_Y - SETPOINT_Y);


}

void doXstep()
{
  if ( digitalRead(DIR_XPIN) == HIGH ) SETPOINT_X--;
  else SETPOINT_X++;
}

void doYstep()
{
  if ( digitalRead(DIR_YPIN) == HIGH ) SETPOINT_Y--;
```

```
void doPID()
{
  interrupts();
  myPID_X.Compute();
  myPID_Y.Compute();
  if (abs(ERROR_X) < DEADBW_X) // If the motor X is in position within the deadband width (acceptable error)
    {
      motorX.setSpeed(0); // Turn off the X motor
    }
  else
    {
      motorX.setSpeed(abs(int(OUTPUT_X)));  // X Motor is regulated by PID controller ouput
    }
  if (abs(ERROR_Y) < DEADBW_Y) // If the motor Y is in position within the deadband width (acceptable error)
    {
      motorY.setSpeed(0); // Turn off the Y motor
    }
  else
    {
      motorY.setSpeed(abs(int(OUTPUT_Y)));  // Y Motor is regulated by PID controller ouput
    }
```

```
    }
  int directionX;
  int directionY;

  if(OUTPUT_X > 0)
    {
      directionX = FORWARD;
    }
  if(OUTPUT_X < 0)
    {
      directionX = BACKWARD;
    }

  if(OUTPUT_Y > 0)
    {
      directionY = FORWARD;
    }
  if(OUTPUT_Y < 0)
    {
      directionY = BACKWARD;
    }
```

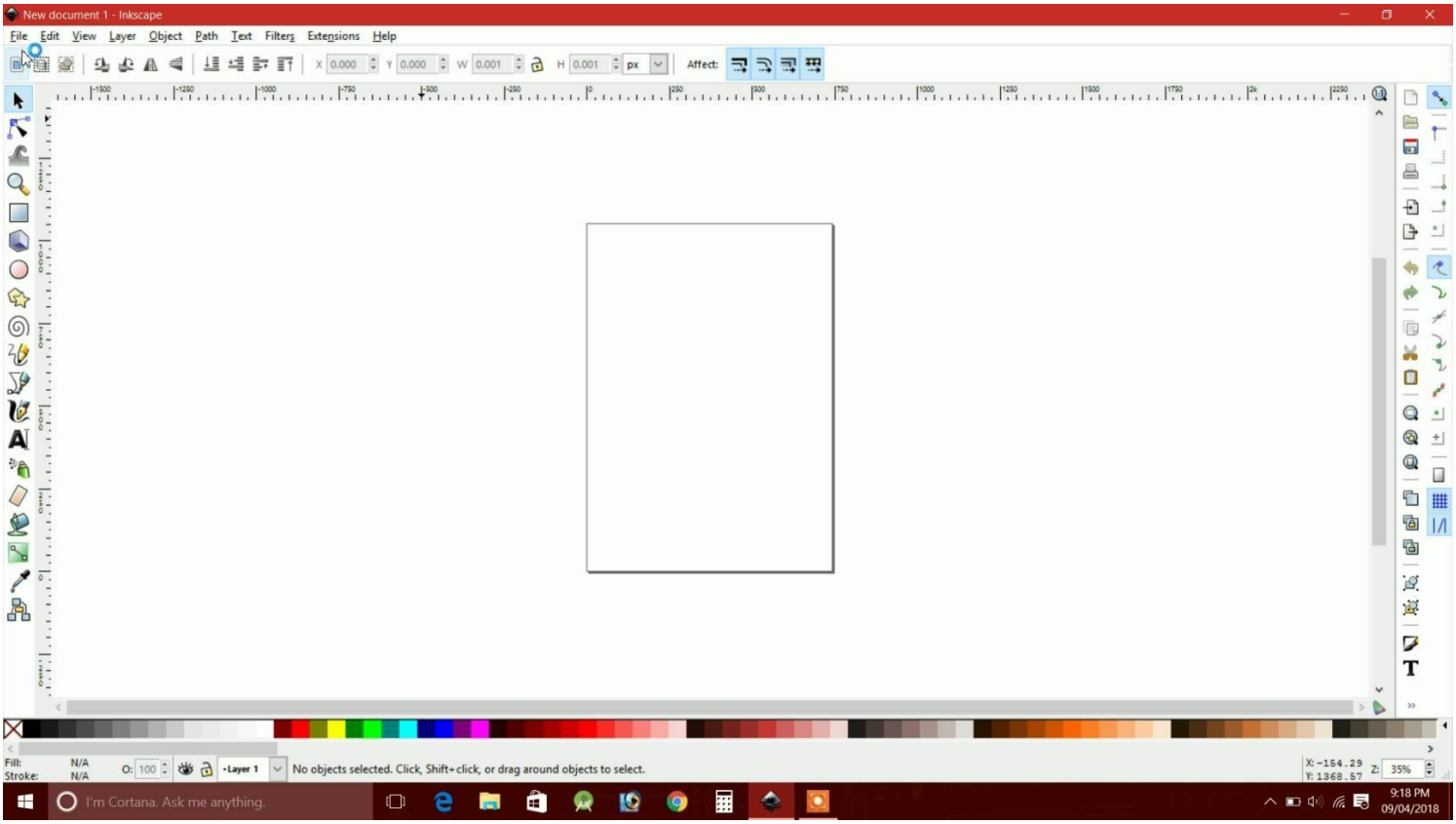
4. Software Programs

4.1. INKSCAPE

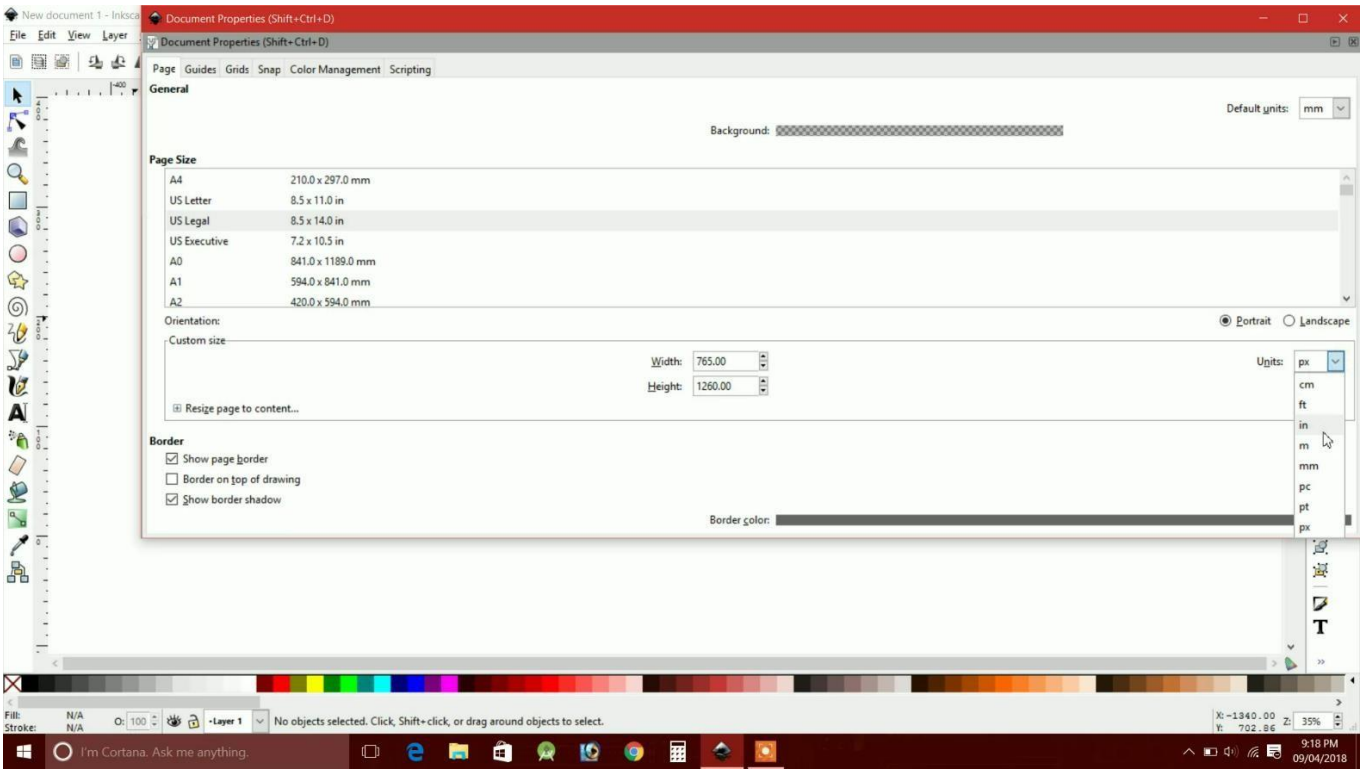
- A source control software have graphics converter that convert images or text to G-code.

- Steps of Converting graphics or text to G-code

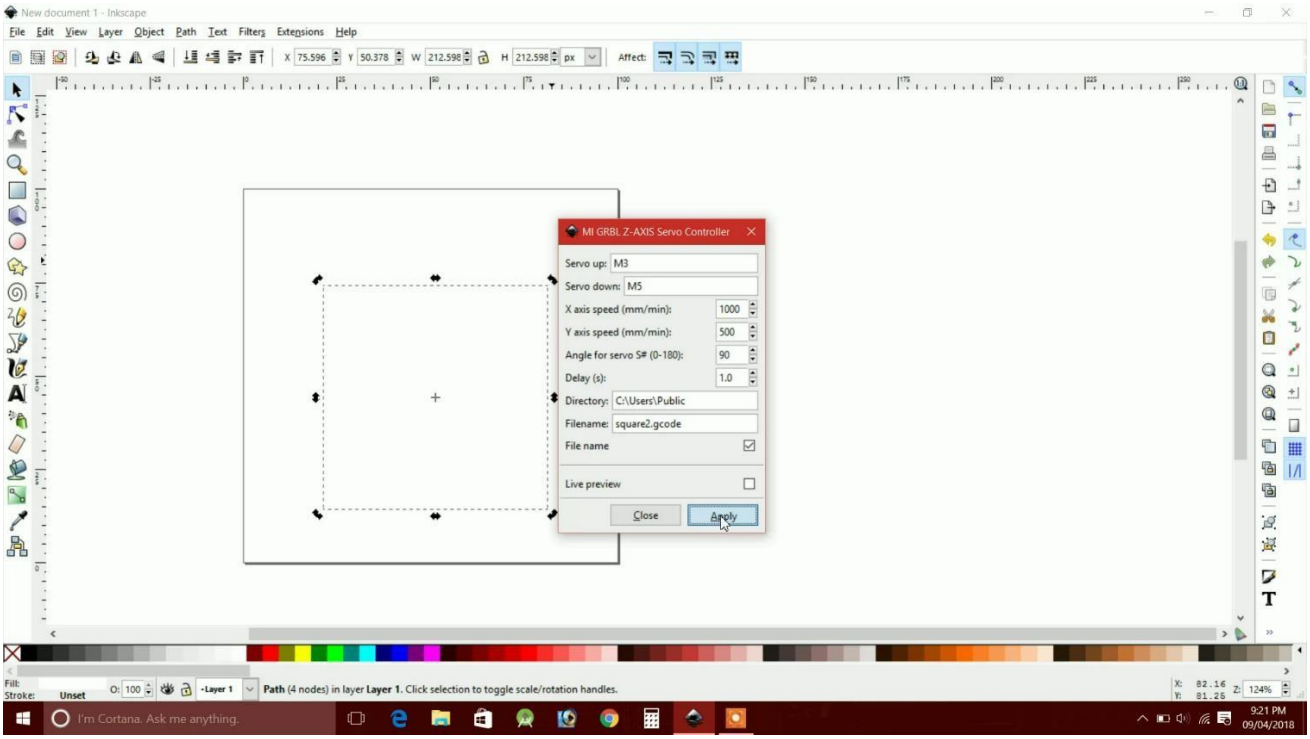
1) Select Position and select Text tool to write your text



1) Shift+ctrl+D or Properties to select the Font Size



2) Mark your Text after configure font size and press Shift+Ctrl+c or choose Object to path to convert text to G-code

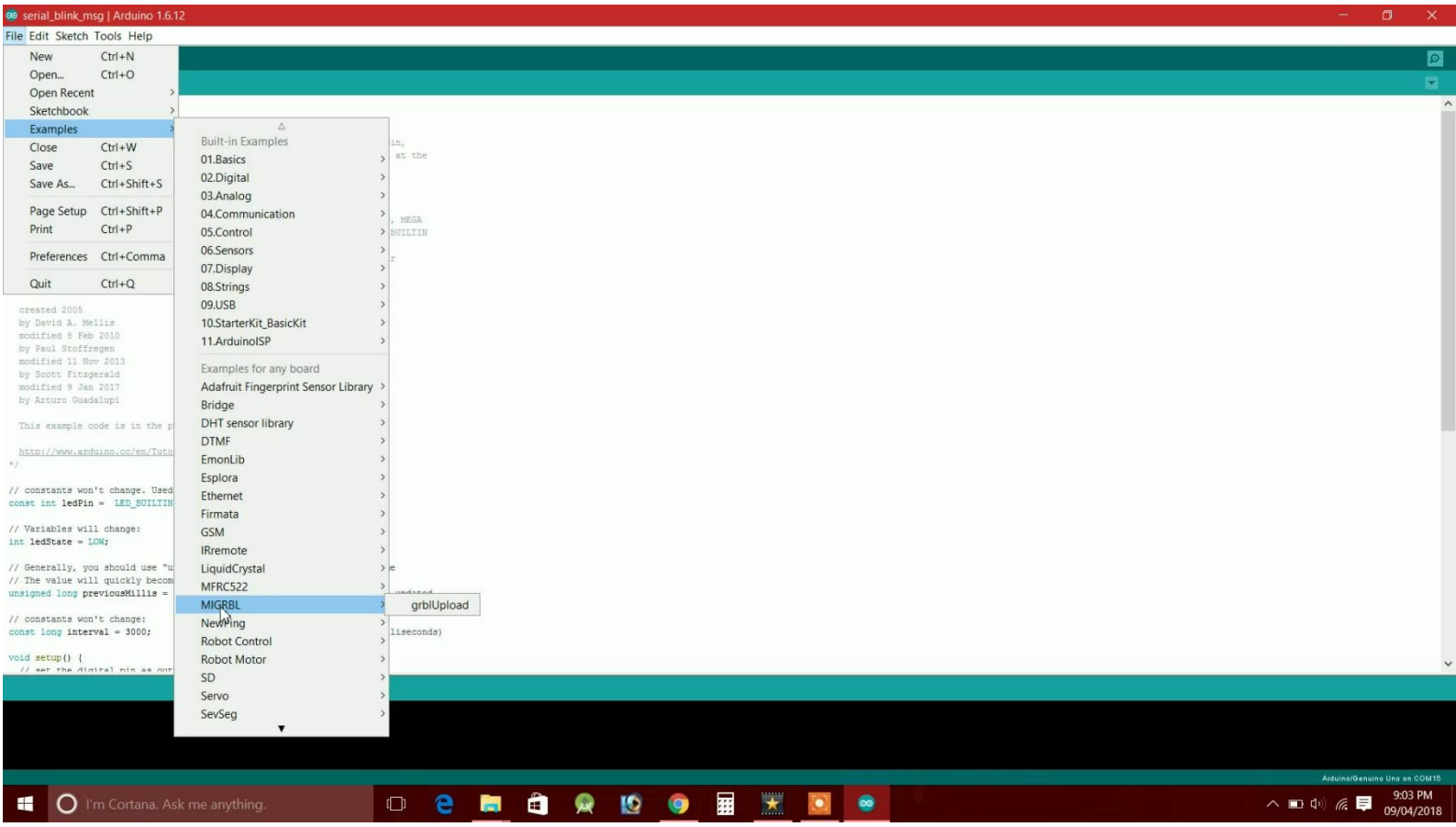


4.2. GRBL FIRMWARE

- It’s a source used to control servo motor(Z-axis in our case of printer)

- Generate PWM frequency of the spindle control pin to a suitable frequency for controlling the servomotor.

- Configure parameters By setting them on Serial monitor Arduino:
  - ➢ Parameter that tells where limit switches are positioned.
  - ➢ Parameter make stepper motors on mode
  - ➢ Travel resolution (steps/mm)
  - ➢ Maximum feed rate, acceleration, maximum travel speed.

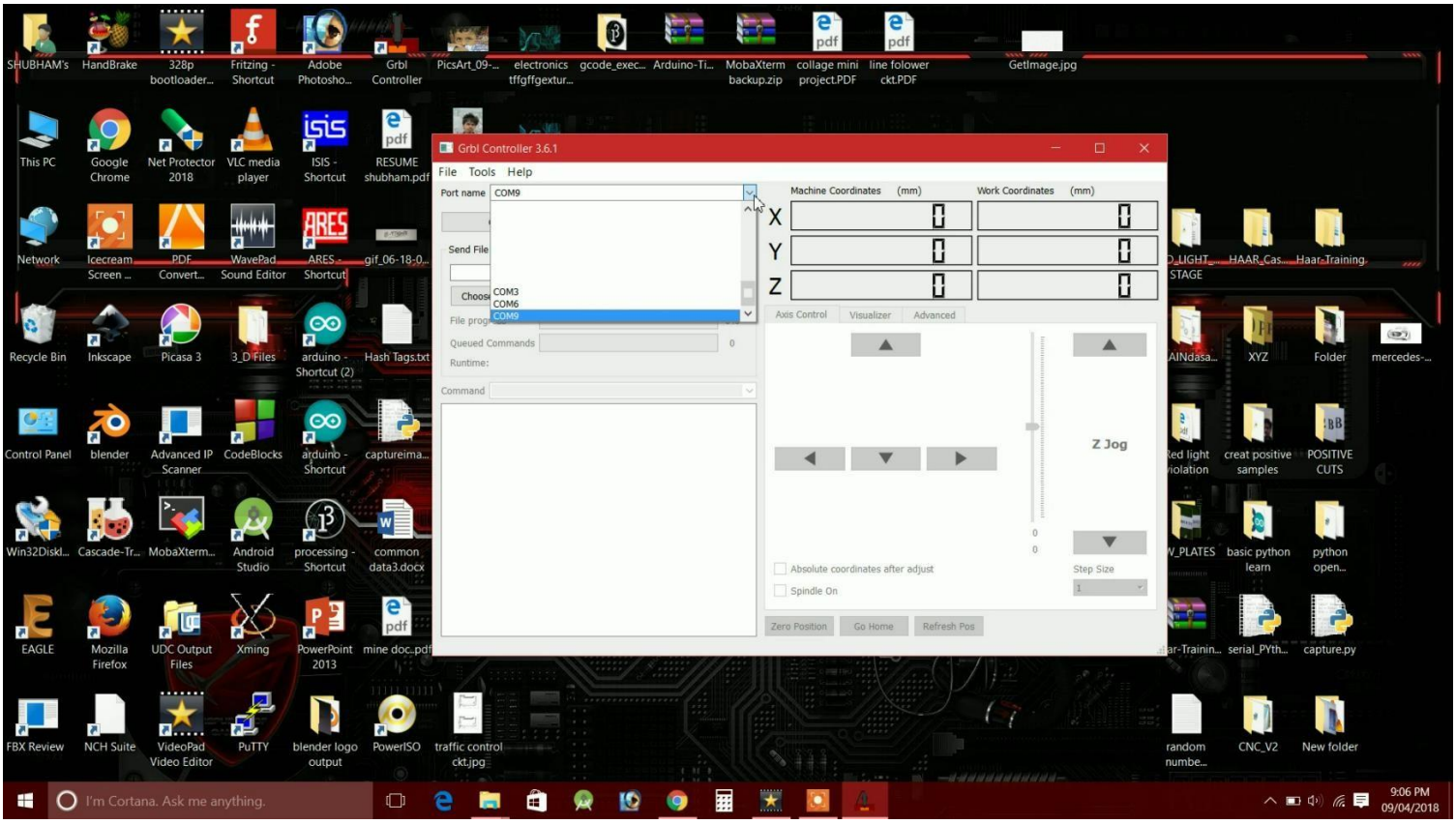


4.3. GRBL Plotter

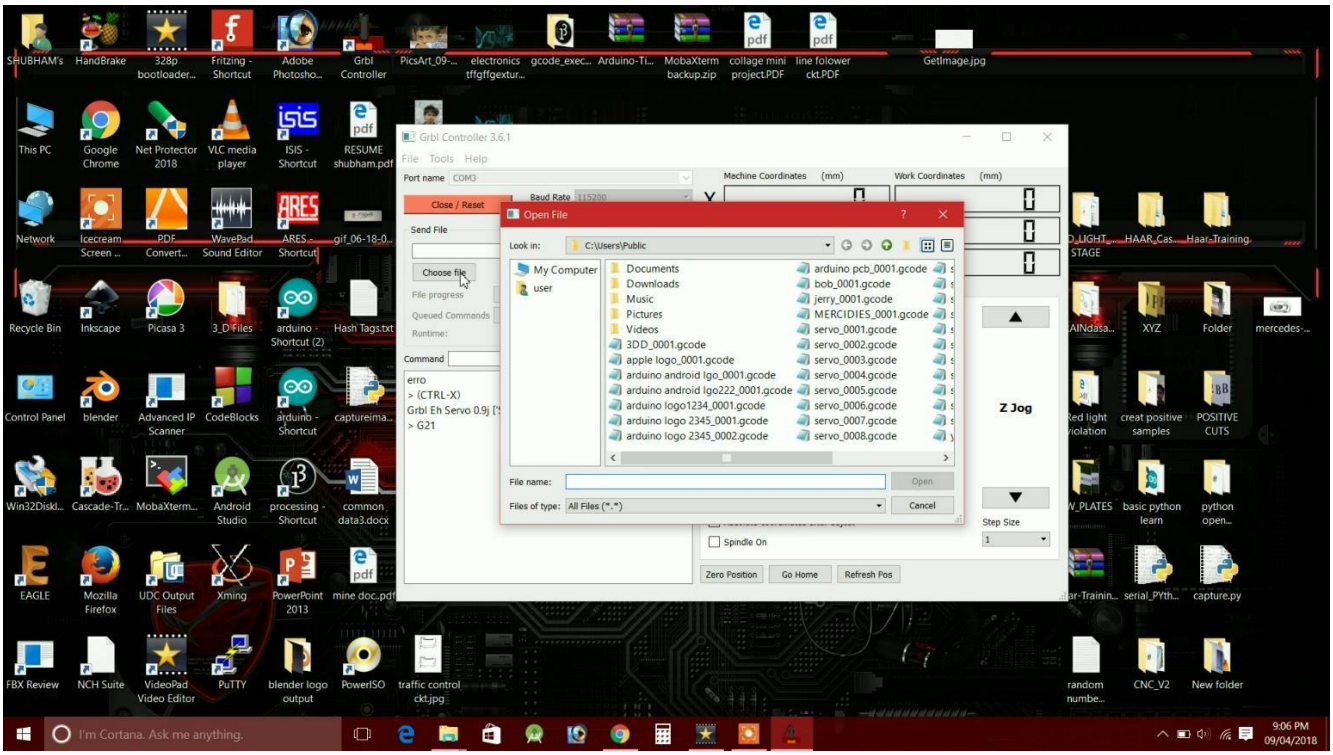
- It’s used to monitor or can be comparing between the actual length and the drawn length of text

- Steps to used it:

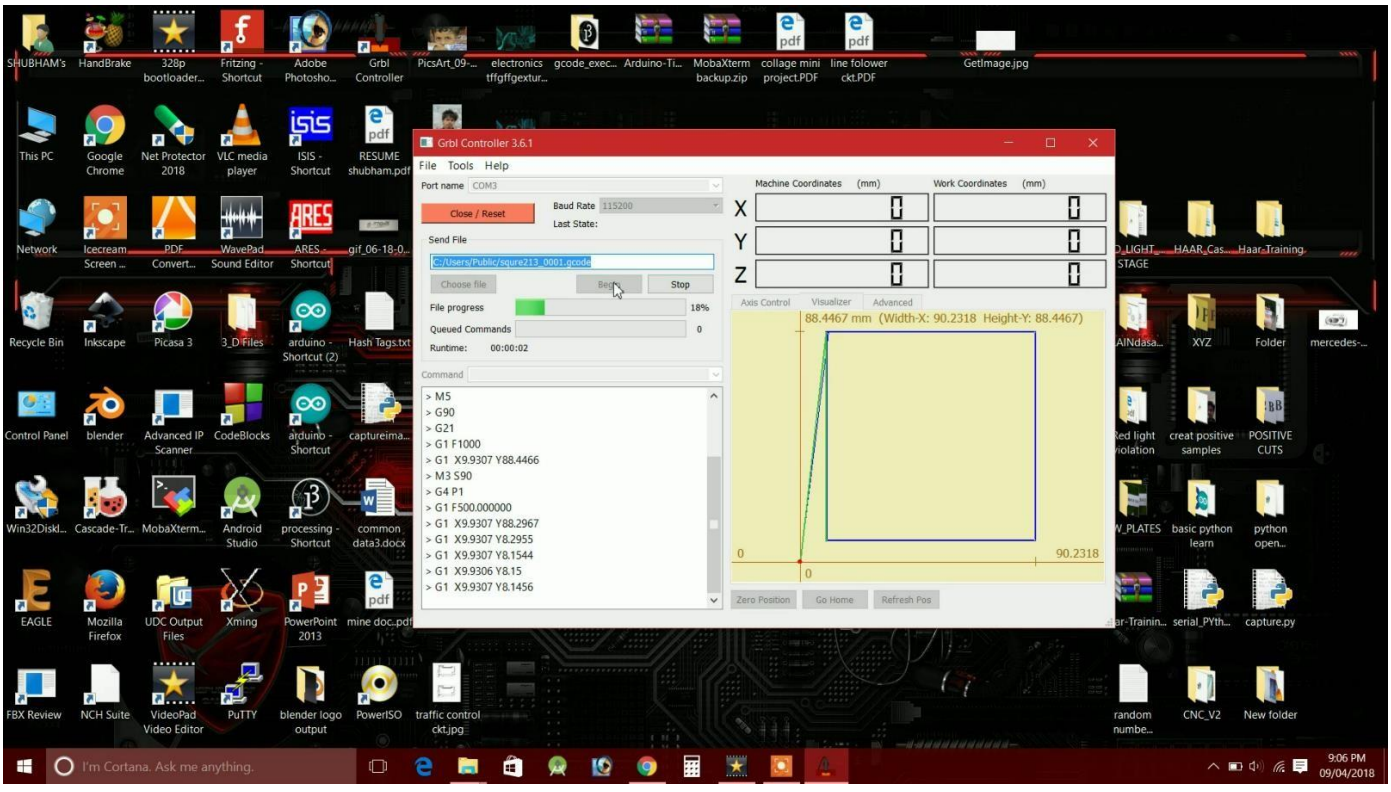
1) Select port channel that open in Arduino



2) Choose the file of G-code that have the text or graphics

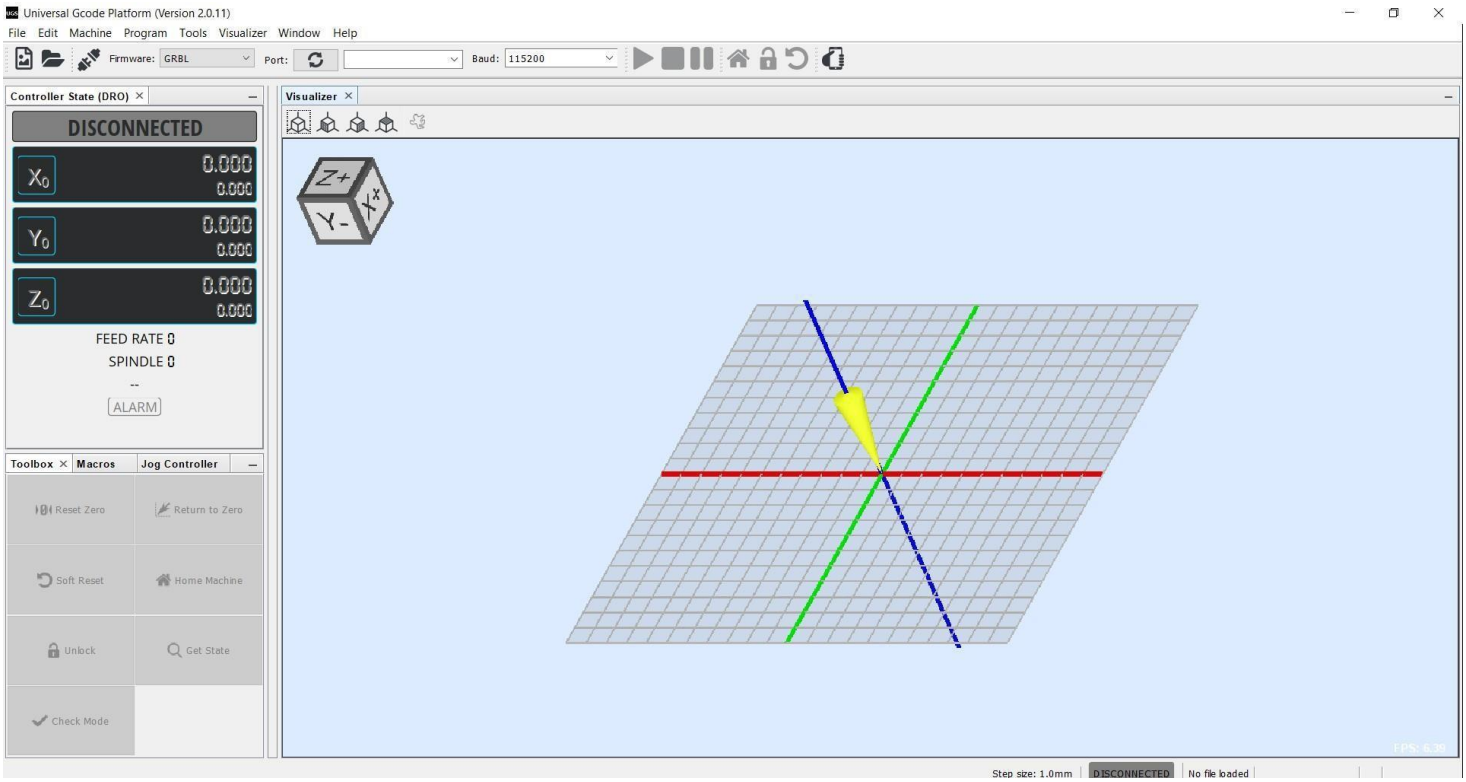


3) Start operation by press on Begin and enjoy!!!



4.4. UGS (Universal G-Code Sender)

- Source monitoring software that monitor the axis when Write text or draw graphics



4.5. GUI (Graphical User Interface)

- Framework enable to communicate with printer through symbols
- Can used it to send some instruction like decrease or increase font size, stop and start,…
- Can used some application to send instruction by Wi-Fi or Bluetooth to Arduino and make the certain action.