CNC MILLING MACHINE (ARDUINO OPERATED)

A project submitted in partial fulfillment to the requirements for the Award of Degree of Bachelor of Science in Electrical and Electronic Engineering

By:

Md.Sharifullah Patwary

ID: 162-33-3364

Md.Ismail

ID:162-33-3498

Supervised by:

Tasmia Baten Dina

Sr.Lecturer,

Department of Electrical and Electronic Engineering

Daffodil International University

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
FACULTY OF ENGINEERING
DAFFODIL INTERNATIONAL UNIVERSITY
April,2019

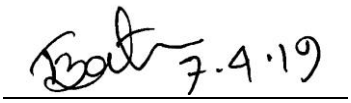
Dedication

To our Parents and Faculties…….

Declaration

This is to certify that the project entitled "CNC Milling Machine (Arduino Operated) has been completed satisfactorily and a partial fulfillment of the requirements of the degree of Bachelor of Science in Electrical and Electronic Engineering.

Signature of the Project Supervisor



Tasmia Baten Dina

Sr. Lecturer

Department of Electrical and Electronic Enginneering

Daffodil International University

Signature of the Candidates

Md.Sharifullah Patwary

ID: 162-33-3364

Md.Ismail

ID: 162-33-3498

Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

We owe our deep gratitude to our project Supervisor Tasmia Baten Dina, Sr. Lecturer, Department of EEE, Daffodil International University for providing us an opportunity to do the project work in "CNC Milling Machine (Arduino Operated) and giving us all support and guidance which made us complete the project duly. We are extremely thankful to her for providing such a nice support and guidance till the completion of our project work by providing all the necessary information for developing a good system, although she had busy schedule managing the corporate affairs.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all our friends, senior, junior brothers and sisters who helped us in successfully completing our project work. Also, we would like to extend our sincere esteems to all staff in laboratory for their timely support.

Abstract

In this paper, Design and implemented of an Aurduino based CNC Milling Machine have been presented and analyzed. CNC Milling Machine (Arduino Operated) is implemented based on the principle of Computer Numeric Control (CNC). A CNC Milling Machine (Arduino operated) basically works with two stepper motors and a Servo motor with the control of Arduino Microcontroller. In our proposed Project we use the Motor Driver (L293D IC) for the moving of the Cutter (pen) in the x axis and y axis direction to the rotation in stepper motors and a servo motor that is used for raising and lowering the Cutter (pen) in the z axis (vertical) directions. In our project we used three different software. Solidworks is for design Inkscape is for cam and processing is for transfer NC code Computer to machine. At last, the result of practical circuit shows the proper function and also verifies the reliable automation within reasonable cost.

Table of Contents

CHAPTER 02: MAJOR COMPONENTS DESCRIPTION (3-14)

MAJOR SOFTWARE DESCRIPTION

LIST of Figures **(3-25)**

CHAPTER 01

INTRODUCTION

1.1 Introduction

Parts manufacturing techniques have greatly evolved during the last many years in response to the highly evolving needs on the various production sectors. Machining, is one on the primary techniques, is finished by removing shavings from raw materials to create high-precision parts.

The elaboration of CNC is usually Computer Numerical Control. The designed part is usually imported into CAM application environment & performs the editing of different machining parameters such as type of current, feed, and spindle speed (RPM), depth of cut & means of cutting etc. Then after verifying the program together with checking the collision detection, NC Code is generated which is sometimes called CNC machine code. Then the CNC program is inputted towards CNC machine & run the program to accomplish the cut.

3-axis machining is among the most preferred techniques to make mechanical parts. For many generations, it has been well known to manufacturers along with players in the industrial sector, as well just as many other domains like architecture, design and skill. It is a relatively simple process, using conventional machining tools such as milling machine, which allows material to be worked tirelessly on 3-axis (X, Y and Z). The machining tool then proceeds to clear out shavings in three basic directions corresponding to the axis of any flat surface. It is very suitable for parts which might be not too deep, but this technique is greatly limited when seeking to handle a deeper part with narrow cavities. The effort can then become very labor-intensive, and renders a finish that is poor.

1.2 Objective

- To reduce the complexity and cost of the CNC Machine
- Simplicity in Machine Operation
- Better Understanding of Machining Operations

1.3 Methodology

- For Collection Information physically work in CNC Milling Machine.
- Collection of information also Form Book and Internet.
- Collection of component from Online Shop.

1.4 Major Components used in the entire project

- ATmega328p (with Arduino Bootloader)*
- USB to Serial adapter
- 2x L293D ICs
- Mini Servo Motor
- Stepper motor
- Prototyping PCB Circuit Board Stripboard
- IC base
- 4x 2pins Screw Terminal Connector (or 2x 4 pins Screw Terminal Connector)

1.5 Major Software Used in the entire project

- Solidworks.
- Inkscape.
- Processing.

1.6 NC CODE

- G-CODE
- M-CODE

1.7 Project Outline

Chapter-02 describes all the hardware and Software with power supply to the project.

Chapter-03 describes the Block Diagram, working procedure, connection diagram and Explanation

Chapter-04 reviews the result found through the project and provides a discussion on the findings.

Chapter-05 describes the limitation of this project and provides the future works that may be approached and conclusion.

1.8 Summary

Firstly we discuss about project, CNC Automation. Then we discuss about problem statement, methodology and objective of this project. Finally we discuss about project outline in this chapter.

CHAPTER 02

MAJOR COMPONENTS DESCRIPTION

2.1 Arduino Nano (ATmega328P V.3)

Inside our project we use Arduino Nano (ATmega328P V. 3). The Arduino Nano ATmega328P V. 3 is a tiny, complete, and breadboard-friendly board. It has about the similar functionality of the Arduino Duemilanove, within the different package. It lacks only a DC strength jack, and works with a Mini-B USB cable as opposed to a standard one.

Nano is automatically sense and switch for the higher potential source of power, there is no dependence on the power select jumper.

Arduino Nano got the breadboard-ability with the arduino and the Mini+USB with smaller footprint as compared to either, so users have more breadboard space. It's got a pin layout that is useful with the Mini or the Basic Stamp (TX, RX, ATN, GND using one top, power and ground on the other). This kind of new version 3. 0 comes with ATmega328P that provide more programming and data memory space. It will be two layers. That makes it easier to hack plus more affordable.

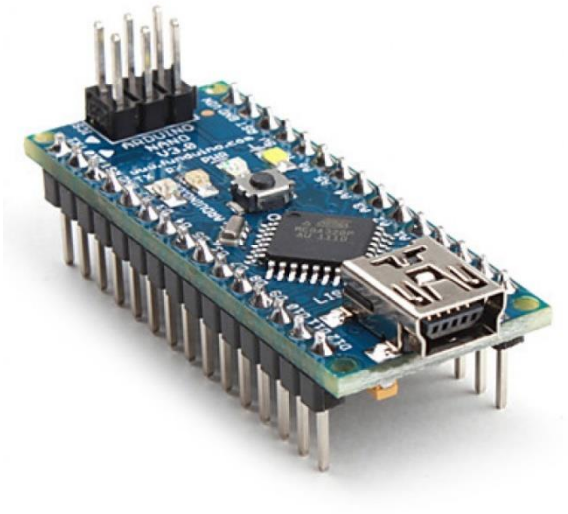


Figure 2.1: Arduino Nano

2.1.1 Specifications

Microcontroller :Atmel ATmega328

Operating Voltage (logic level) :5 V

Input Voltage (recommended) :7-12 V

Input Voltage (limits) :6-20 V

Digital I/O Pins :14 (of which 6 provide PWM output)

Analog Input Pins :8

DC Current per I/O Pin :40 mA

Flash Memory :32 KB (of which 2KB used by bootloader)

SRAM :2 KB

EEPROM :1 KB

Clock Speed : 16 MHz

Dimensions : 0.70” x 1.70”

2.1.2 Features of Arduino Nano

- Green (TX), red (RX) and orange (L) LED
- Power OK blue LED
- Automatic reset during program download
- Auto sensing/switching power input
- Small mini-B USB for programming and serial monitor
- ICSP header for direct program download
- Standard 0.1” spacing DIP (breadboard friendly)
- Manual reset switch

2.1.3 Power

The Arduino Nano is generally powered via the mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power source (pin 27). The power source is automatically selected towards highest voltage source.

2.2 USB to Serial adapter

A USB adapter is many different protocol converters which is useful for converting USB data signals to be able to and from other communications expectations. Commonly, USB adaptors are utilized to convert USB data to standard serial port data and vice versa.

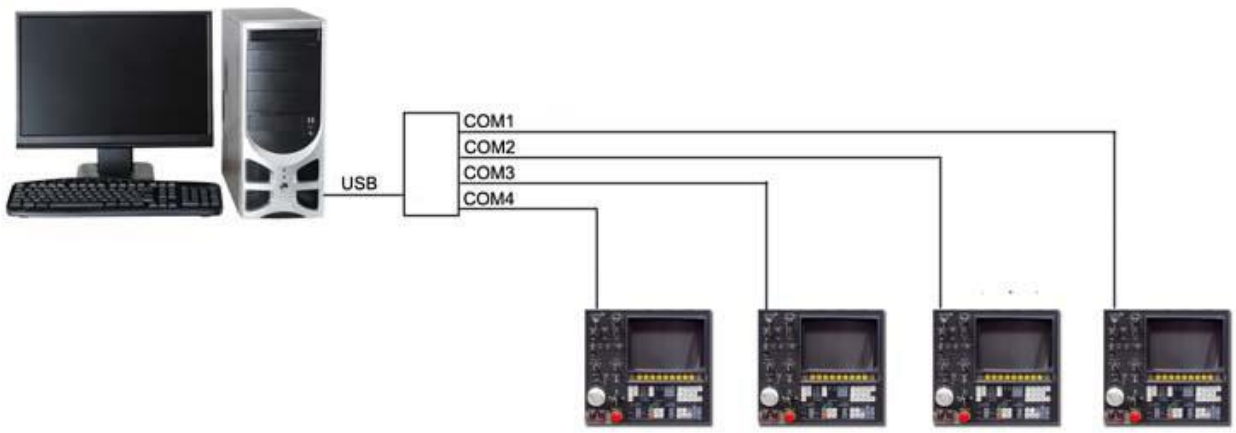


Figure 2.2: Computer Port

2.3 L293D ICs

L293D is actually a Motor Driver IC that permits DC motor to drive relating to either direction. L293D is a 16-pin IC can be control some of two DC motors at the same time in different direction. This means that we can control two DC motor which includes a single L293D IC.

The L293D is a 06 pin IC, with eight pins, on each section, dedicated to the controlling of a motor. There are actually 2 I/O and O/U pins and one Enable pin each motor. L293D consist of two H-bridge. H-bridge is a simplest circuit for controlling a low current valuable motor

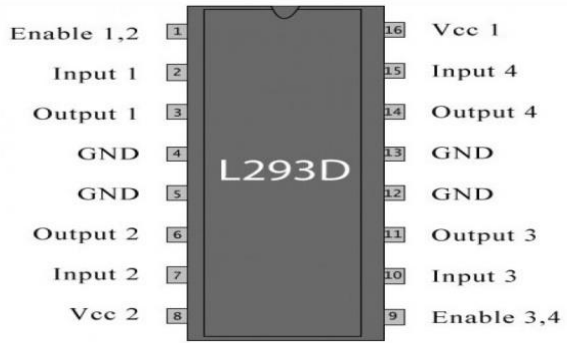


Figure 2.3:L293D IC

2.4 Servo Motor

A servomotor is actually a rotary actuator or linear actuator that provides precise control of linear posture, velocity and acceleration. It consists of a proper motor coupled to a sensor for posture feedback.

A servo motor is an electrical device which may push or rotate an object with superb precision. If we want to rotate plus object at some specific angles or distance, then we use servo motor. It is just consists of simple motor which run through servo apparatus. If motor is used DC powered then it's called DC servo motor, and if its AC powered motor then it is called AC servo motor. We can get quite a high torque servo motor in a small and luxury packages. Doe to these features they think you are used in many applications like toy motor vehicle, RC helicopters and planes, Robotics, Machine etc.

Figure 2.4: Servo Motor

2.4.1 Servo Mechanism

- It consists of three parts:
- Controlled device
- Output sensor
- Feedback System

2.4.2 Controlling servo motor

Servo electronic motor is controlled by PWM (Pulse with Modulation) and that is certainly provided by the control cable connections. There is often a the bare minimum pulse, a maximum pulse including a new repetition rate. Servo motor is capable of turning 90 degree from either track form its basic position. The servo motor expects to check out a new pulse every 20 milliseconds (ms) as well as duration of the pulse will see how far your motor turns. Such as, 1. 5ms pulse could make the motor try the 90° situation, such as if pulse is shorter matched against 1. 5ms shaft moves to 0° and jewel longer than 1. 5 ms than it can certainly turn the servo so that you can 180°.

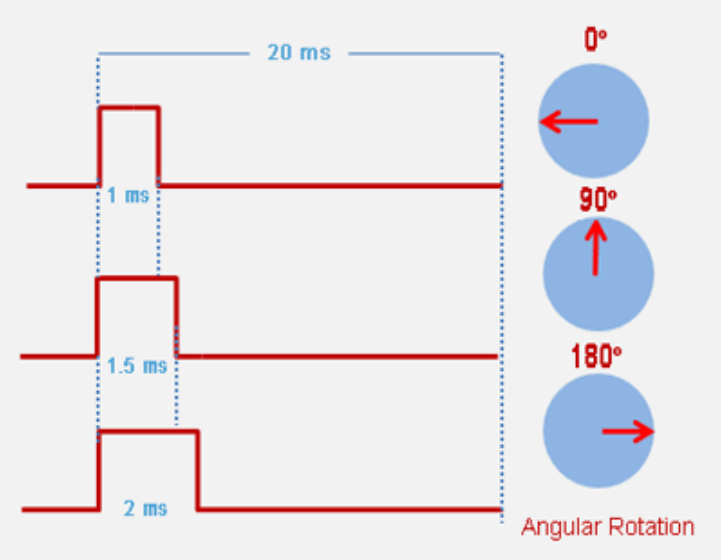


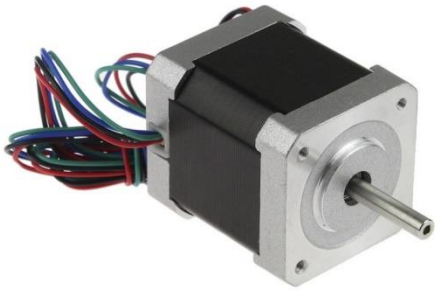
Figure 2.4.2: Controlling Servo Motor

Servo motor may be rotated from 0 to 180 degree, but it can go up to 210 degree, depending on the producing. This degree of rotation can be controlled through the use of the Electrical Pulse of proper width, to be able to its Control pin. Servo checks the pulse in every 20 milliseconds. Pulse of 1 ms (1 millisecond) size can rotate servo to 0 degree, 1. 5ms can easily rotate to 90 degree (neutral position) and also 2 ms pulse can rotate it to be able to 180 degree.

2.5 Stepper motor

The stepper motor is usually an electromagnetic device that converts digital pulses into kinetic shaft rotation. Benefits of steeper motors are cheap, high reliability, high torque at low speeds as well as a simple, rugged construction that operates in almost any environment. The operations of this motor works within the principle in which unlike poles attract each other and like poles repel one another. When the stator windings are excited with some sort of DC supply, it produces magnetic flux and ensures the North and South poles.

The number of input pulses given to the motor decides the step angle and as such the position of motor shaft is controlled by controlling the volume of pulses. This phenomenal feature makes the stepper motor internet marketing well suitable for open-loop control system wherein the actual position of the shaft is maintained with exact amount of pulses without using a feedback sensor.



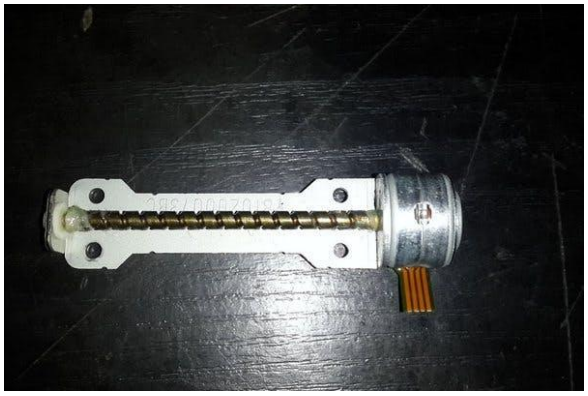


Figure 2.5: Stepper motor

2.6 Prototyping PCB Circuit Board Strip board

Strip board is a generic name for a widely used type of electronics prototyping board seen as a a 0. 1 inches (2. 54 mm) common (rectangular) grid of holes, together with wide parallel strips of copper cladding running in one direction entirely across one side of the board.

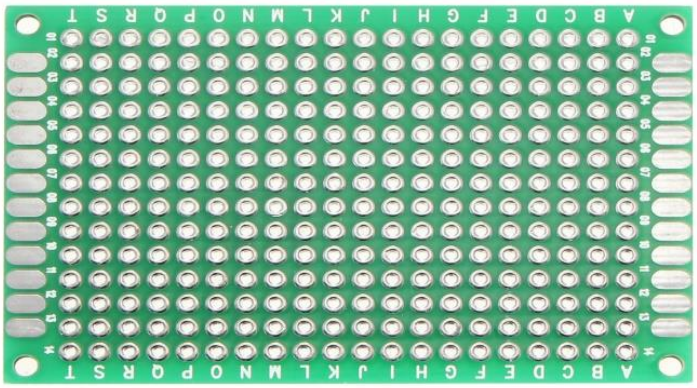


Figure 2.6: PCB Circuit Board

2.7 IC BASE (16 PIN)

An IC socket, or integrated circuit socket, is included in devices that contain an integrated circuit. An IC socket is required as a placeholder for IC chips and is used so that they can allow safe removal and insertion of IC chips because IC chips can become damaged from heat due to soldering.

Figure 2.7: IC Base

MAJOR SOFTWARE DESCRIPTION

2.8 SolidWorks

SolidWorks is a solid modeling computer-aided design (CAD) and computer-aided engineering (CAE) computer program that runs on Microsoft Windows. SolidWorks is published by Dassault Systèmes.

2.8.1 Uses of SolidWorks Software

3D Modeling Software with parametric construction. SolidWorks is computer-aided design (CAD) software owned by Dassault Systèmes. It uses the principle of parametric design and generates three kinds of interconnected files: the part, the assembly, and the drawing.

2.9 Inkscape

Inkscape is CAM software. Computer-aided manufacturing (CAM) is an application technology that uses computer software and machinery to facilitate and automate manufacturing processes. Basically its uses convert 2D or 3D Image to Machine language. That is called G-Code and M-code.

2.10 Processing

Processing is a flexible software sketchbook and a language for learning how to code within the context of the visual arts. Its uses transfer NC Code Computer to Machine and also transfer different type Instruction using Key Board for machine .The Processing software is free and open source, and runs on Windows platforms.

Instruction List:

P: Select serial port

1: Set speed to 0.001 inches (1 mil) per jog

2: Set speed to 0.010 inches (10 mil) per jog

3: Set speed to 0.100 inches (100 mil) per jog

Arrow keys: Jog in x-y plane

h: Go home

0: Set home to the correct location

g: Stream a NC Code file

x: Stop streaming NC Code

2.11 NC Code

G-code which has many variants is the common name for the most widely used numerical control (NC) programming language. It is used mainly in computer-aided manufacturing to control automated machine tools.

Normally Two types of NC Cod

- G-Code
- M-Code

2.11.1 G-Code

G-code is a programming language that instructs machines how to move.

2.11.1.1 List of G-Code

G-Code	Description
G00	Rapid Travers Motion
G01	Linear Interpolation Motion
G02	Circular Interpolation Motion (CW)
G03	Circular Interpolation Motion (CCW)
G28	Machine Home
G40	Cutter Compensation Cancel
G41	Cutter Compensation Left To Program Path
G42	Cutter Compensation Right To Program Path
G43	Tool Length Compensation
G54 - G59	Work Coordinate #1 Thru #6
G80	Fixed Cycle Cancel
G81	Drill Canned Cycle
G82	Spot Drill Canned Cycle

G83	Peck Drill Canned Cycle
G84	Tapping Canned Cycle
G85	Fixed Cycle (Boring)
G86	Fixed Cycle (Boring)
G87	Fixed Cycle (Back Boring)
G88	Fixed Cycle (Boring)
G89	Fixed Cycle (Boring)
G90	Absolute Value Command
G91	Incremental Value Command
G92	Work Offset Set
G98	Canned Cycle Initial Point
G99	Canned Cycle Initial Point Return

2.11.2 M-Code

M codes are MACHINE codes; these operate most of the basic electrical control functions such as Coolant, Tool changers, safety circuits etc.

2.11.1.1 List of M-Code

M-Code	Description
M00	Program Stop
M01	Optional Stop: Operator Selected to Enable
M03	Spindle Clockwise
M04	Spindle Counter Clockwise
M05	Spindle Stop
M06	Tool Change Commend
M07/M08	Cooler ON
M09	Cooler OFF
M30	Program Stop
M97	Local Subprogram Call
M98	Subprogram Call
M99	Subprogram Return

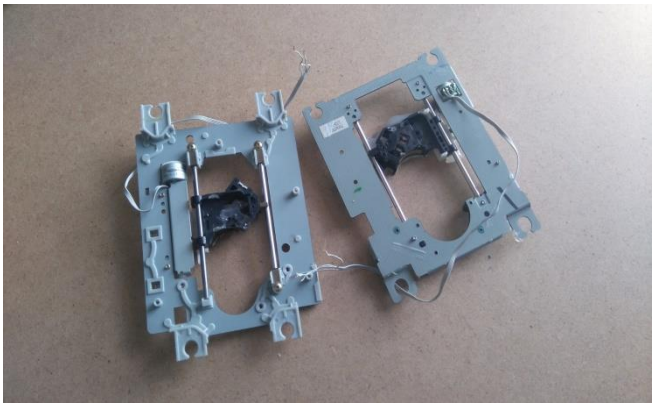
CHAPTER 03

SYSTEM DESIGN & DEVELOPMENT

3.1 Mechanical Part Design

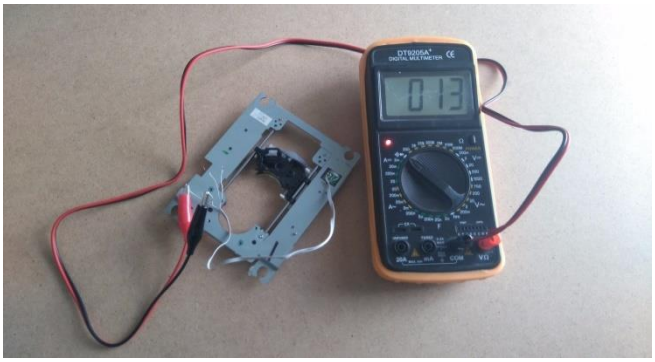
First step to get started on building this CNC machine is to disassemble the DVD drives and lift off them the stepper motors. Use the screwdriver to open and lift off them the rails.

Now we have the two stepper motors we should instead solder some cables on them. Proceed using caution, like as 2nd image above. Now we should instead find the correct combination to drive and have tried them correctly, so take a multi meter with alligator clips (4th image) and don it "short-circuit" function (4th image). Usually (5th image), the first and second cables are concluding the circuit - the led is started up and a beep sounds - consequently we have found the first phase-motor involving stepper motor. The other two cables, 3rd and 4th, uses the second phase-motor involving stepper motor.

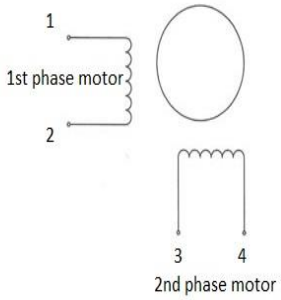


(1)

(2)



(3)



(4)

Figure 3.1: Steeper Motor Setup

3.1.1 Constriction of X axis

Place the opposite stepper motor on two Particle board pieces and mark them with a pen so as to open the (4) holes for the anchoring screws. Again, make sure that the motor can be perfectly aligning (use a triangle ruler).

Place both pieces of Particle board on the X axis (big Particle board piece) and mark them with a pen to be able to open the (4) holes required to fit about the mounting angles.

The particular stepper motor is installed and fixed inside the columns; the couplings and the bearing are situated on the support opposite side, observing that it is correctly adjusted and leveled in order to avoid deviations in the milling. The structure is fixed alongside the base of the X axis.

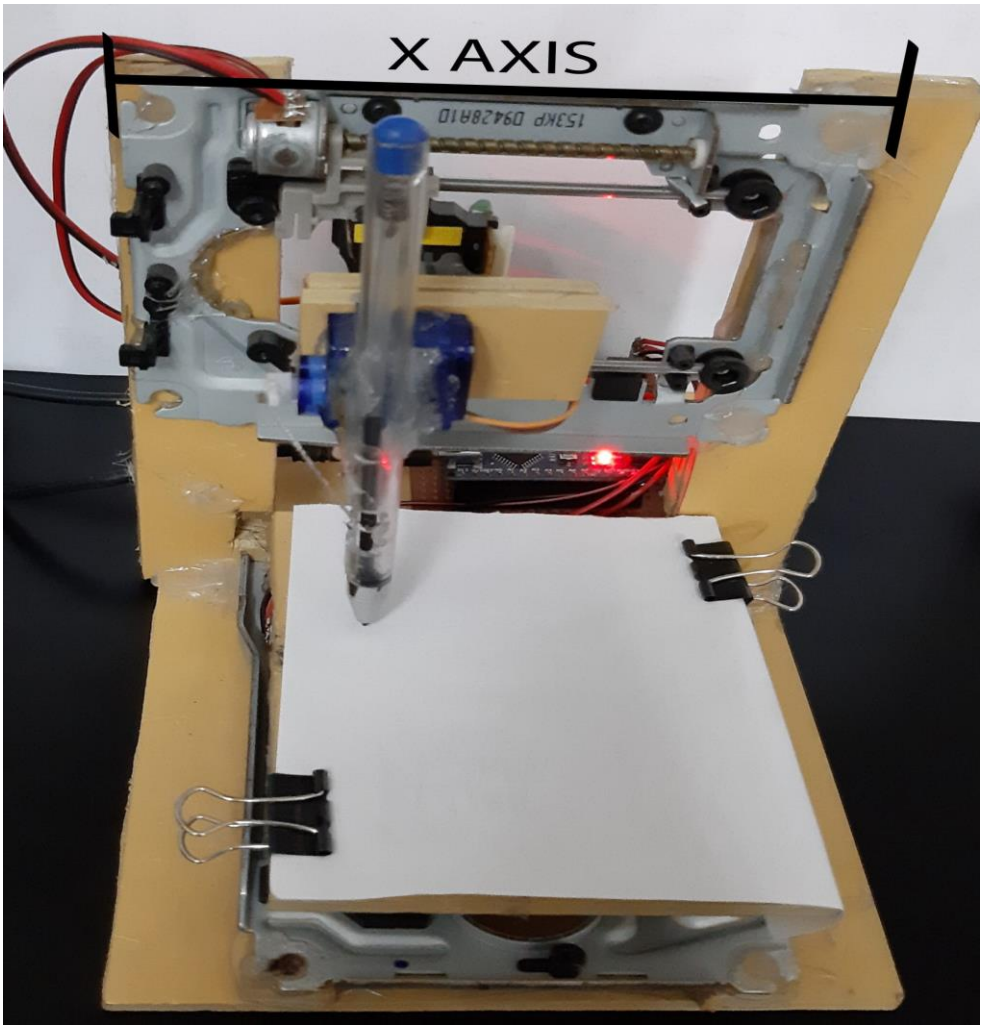


Figure: 3.1.1: X AXIS

3.1.3 Constriction of Y axis

Place and the second stepper motor on two Particle board pieces and mark them with a pen in an effort to open the (4) holes for the screws. Again, make sure that the motor is certainly perfectly aligning (use a triangle ruler).

Place the two main pieces of Particle board on the X axis (big Particle board piece) and mark them with a pen in an effort to open the (4) holes required to fit over the mounting angles.

These dimensions permit the movement of the flexible coupling, at the other end there's an easy bearing that serves as a guide. The bearing ought to be correctly adjusted, since it will support the screw within the Y-axis, we must observe that the screw must turn without problems, as well when the nut of the bushing that moves any bed.

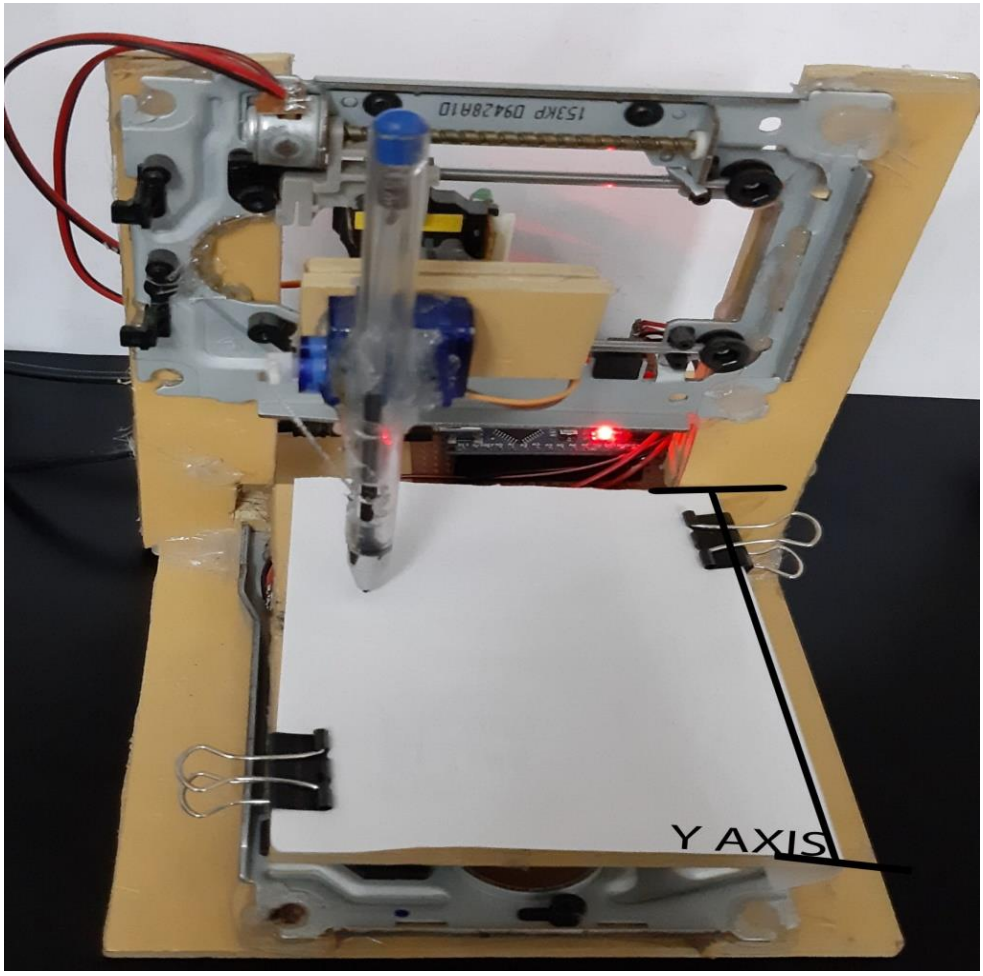


Figure: 3.1.2: Y AXIS

3.1.3 For Z axis

We will need something to attach it on Y axis, a flat surface. On that surface we will attach the servo motor (Z axis) and the pen base. Pen must be able to move up and down with the help of servo motor.

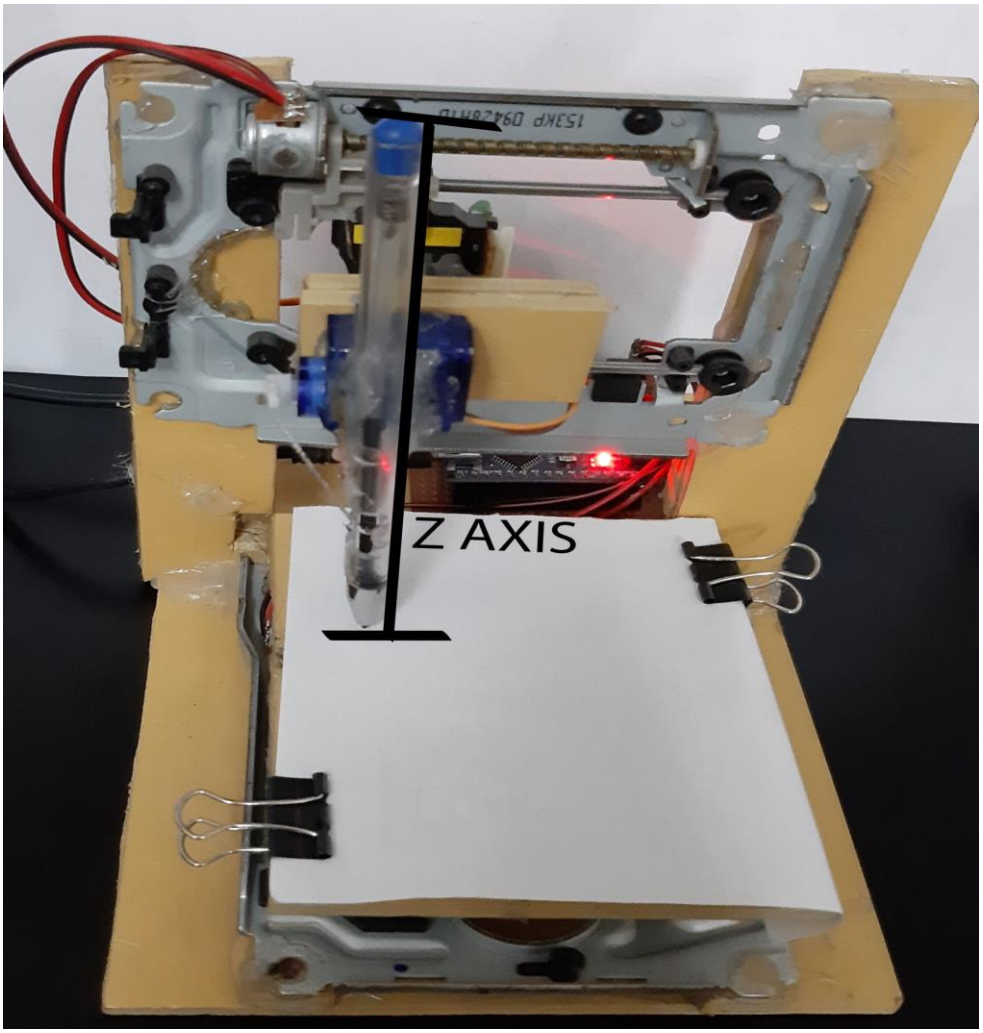


Figure: 3.1.3: Z AXIS

3.2 Electrical Part Design

The guts of our machine is an Arduino Nano (ATmega328P Versus. 3) board that will control the movement of each actuator according to the instruction received from the computer, in order to manipulate these stepper motors we need a stepper motor driver to manipulate the speed and direction of each actuator.

In this case we will use an L293D H bridge motor driver. This will receive the motor command directed from Arduino through its inputs and control the stepper motors featuring an output.

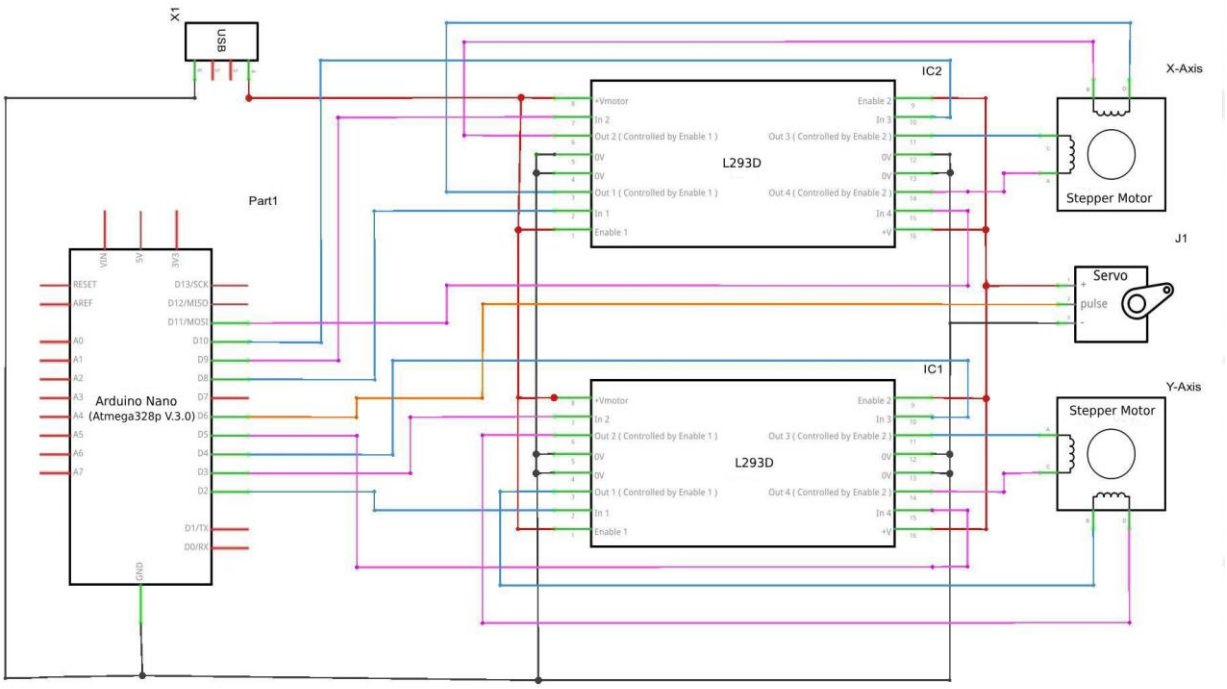


Figure 3.2: Schematic Diagram

3.3 Power

We requires more current than one USB port can grant, so we must connect one more USB cable, Connect only power cables (usually red and black) while using the primary one. The voltage remains 5V, but it is advisable to double the current! (From wiki: max. current of USB 2.0: 0.5 A and of USB 3.0 & 3.1: 0.9 A).

3.4 Making G-Code & M-Code

We placed a metal surface on X axis to fit a paper sheet on it. We need to make a paper sheet that dimensions 75mm x 75mm, but working area is only 40mm x 40mm.So our total working area is 40mm x 40mm.

At First we need to create an Image By using Solidworks.

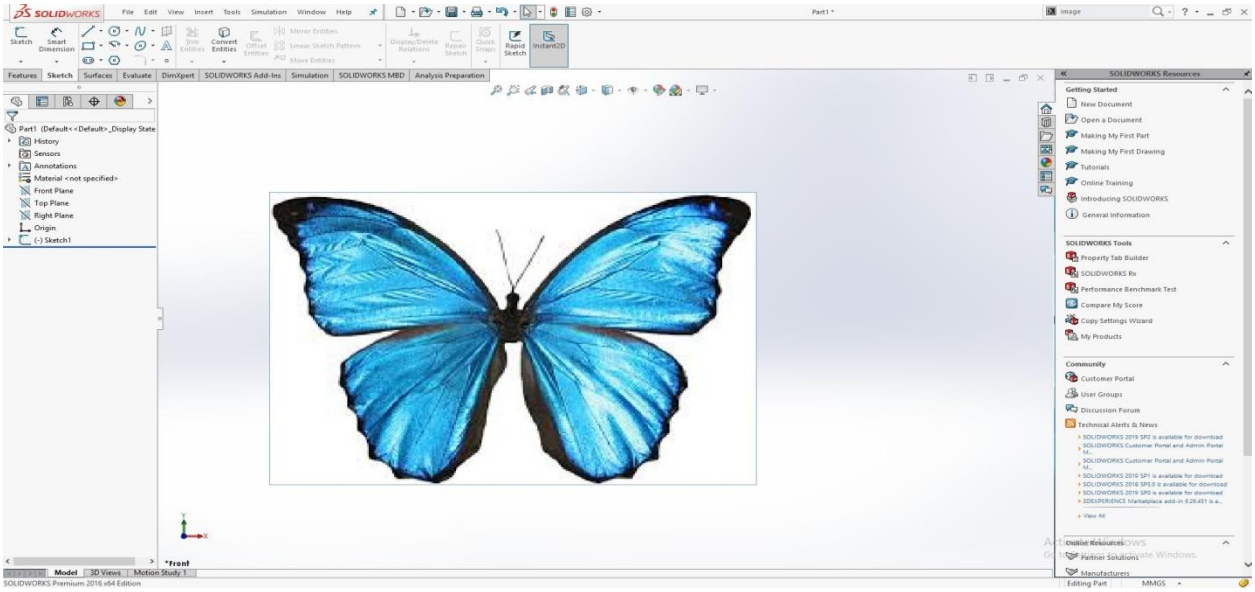


Figure 3.4(a): Create 3D Image

Now we need to open our CAM software (Inkscape) to making G-Code and need to adjust some parameters. Then set the working area to Click File menu →Document Properties. Then set 75mm x 75mm

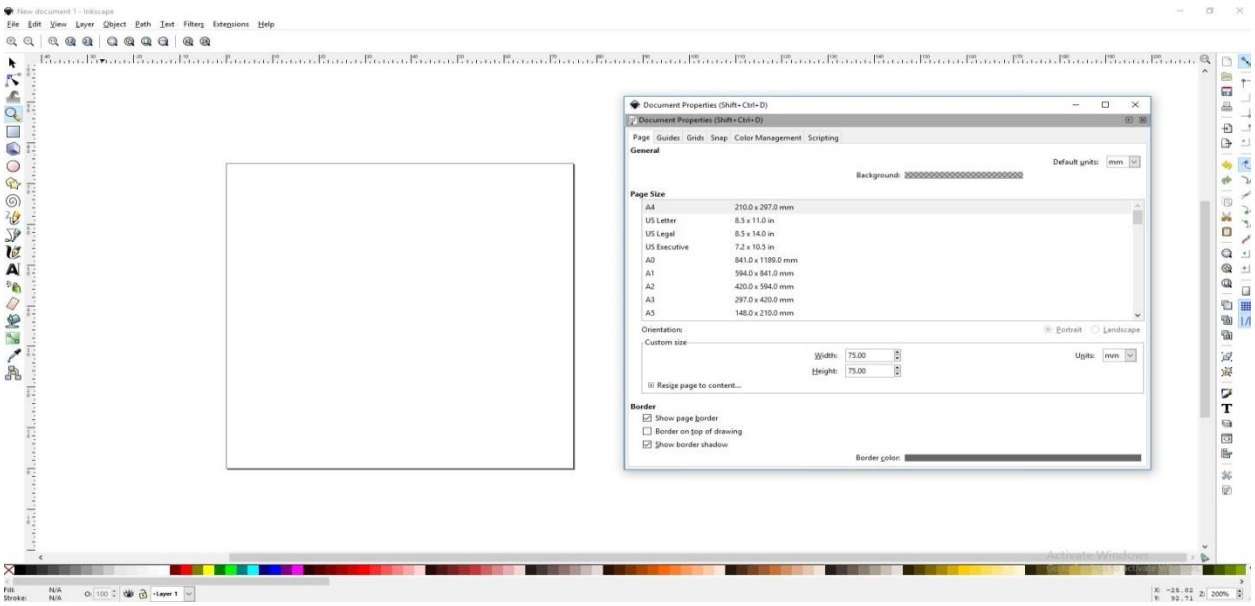


Figure 3.4(b): Create Working Area

2nd Process any 2D Sketch or wright down anything by using Front Tool or Pen Tool then making a Tool Path to click Path Manu →Object to Path (Creating Path Automatically)

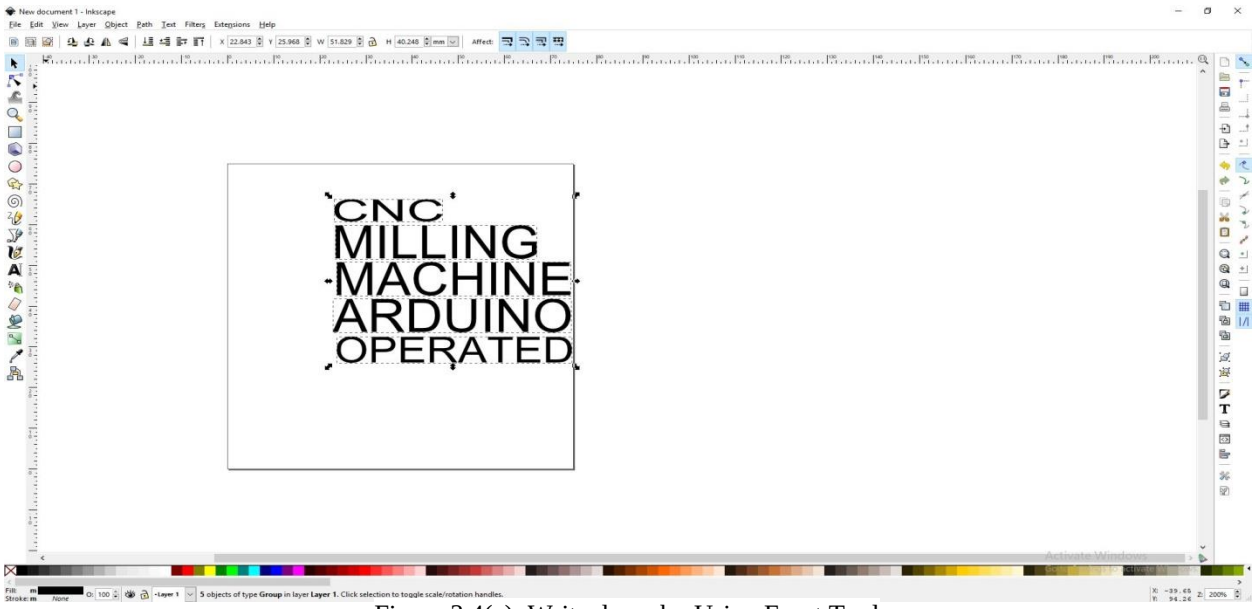


Figure 3.4(c): Write down by Using Front Tool

If we want to create Tool Path for 2D/3D Image we need to use File Manu →Open →Set in the workplace. Then Click Path Manu →Trace Bitmap (Set the parameter) →ok. For Create Tool Path click Path Manu →Object to Path (Creating Path Automatically)

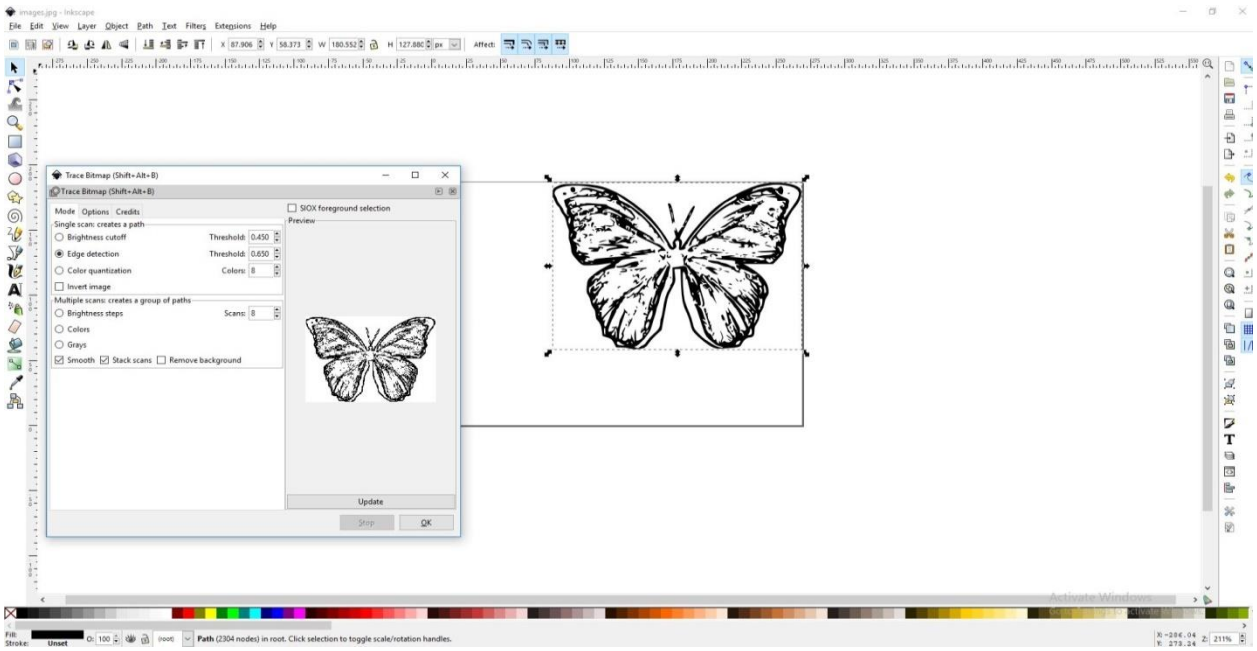


Figure 3.4(d): Creating Tool Path

3rd Process click File Manu →Save as →Set MakerBot Unicom G-Code (*.gcode) →Save. Output File.

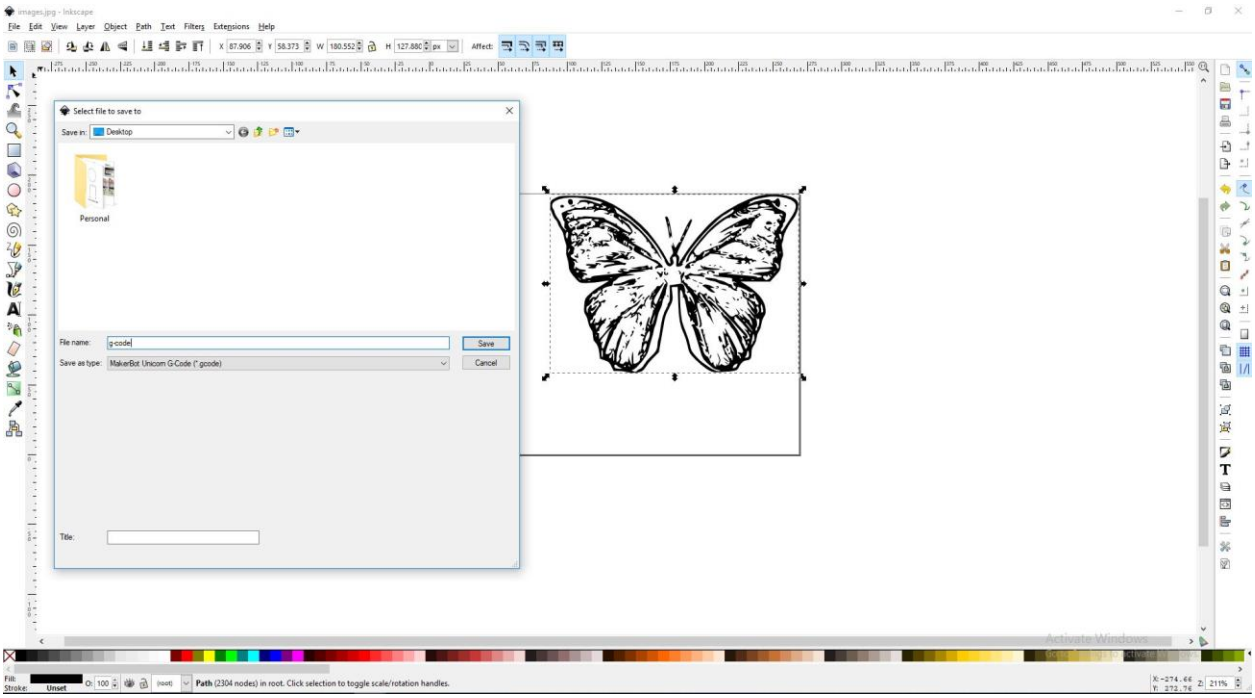


Figure 3.4(e): Making G-Code

Our G-code and M-Code are created.

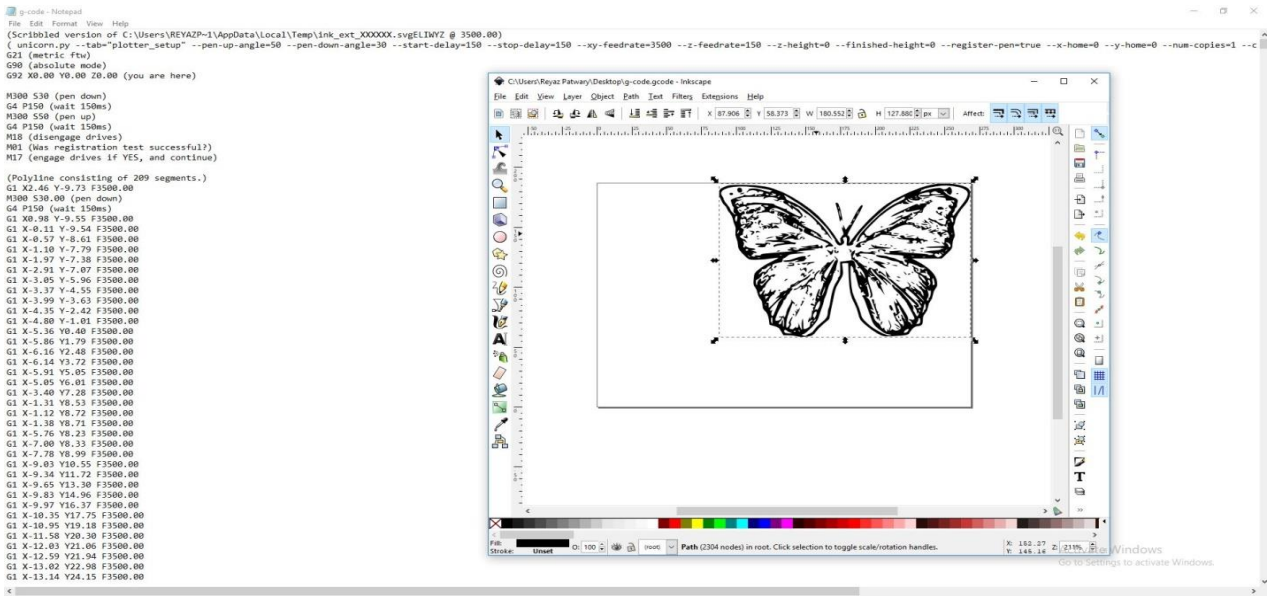


Figure 4.3(f): G-Code

3.5 Working Procedure

We need to open Processing Software and select the Computer Port.Then transfer the G-Code to machine this way. Click Run →Click P (From Keyboard)→Select The COM Port →Click g (From Keyboard)→Select the Desired G-Code file (This file will be uploaded in Arduino Microcontroller)→Click Ok.

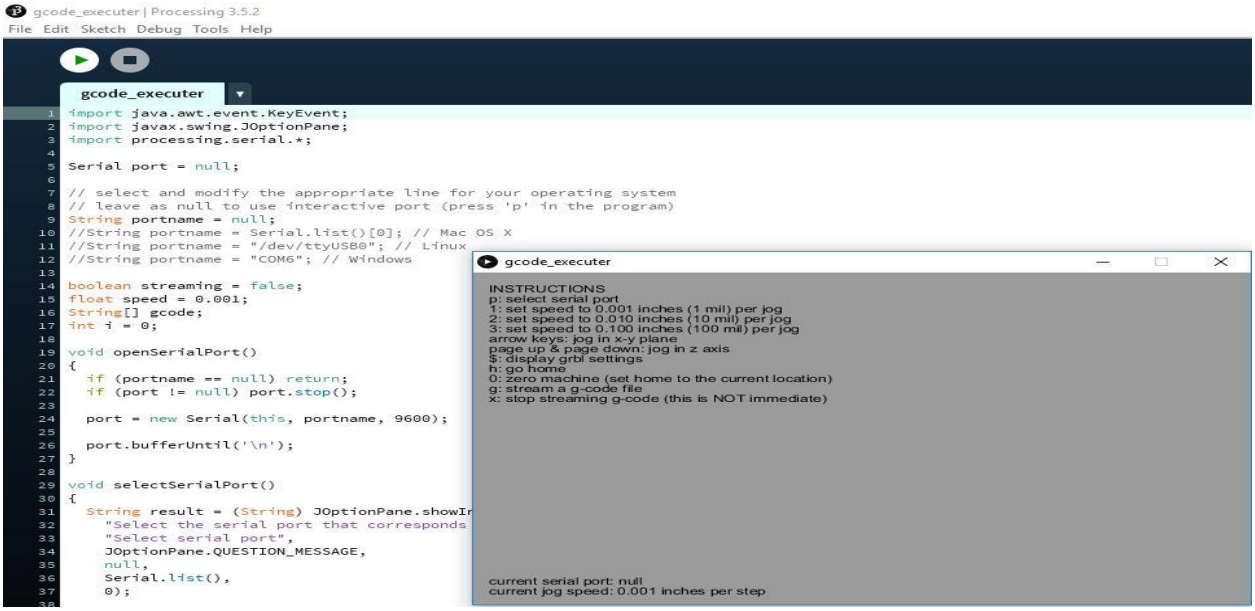


Figure 3.5: Transfer G-Code PC to Arduino Microcontroller

Once we select the G-Code file the machine will automatically start drawing to move X, Y & Z by Following G-Code.

Figure 3.5: Working Procedure

3.6 Summary

In this part we discuss about Mechanical Part, Electrical Part & Software Part connection Diagram with Detail explanation and finally discuss about working procedure in this project.

CHAPTER 04

RESULT AND DISCUSSION

4.1 Introduction

Result is the output of a project. Result presents the success of a project. We find out the successful result of our project by different experiment.

4.2 Project Before setup

Active Arduino Nano: In the figure 4.1; we can see, the system is not running. Now we need to send G-code to start the System.

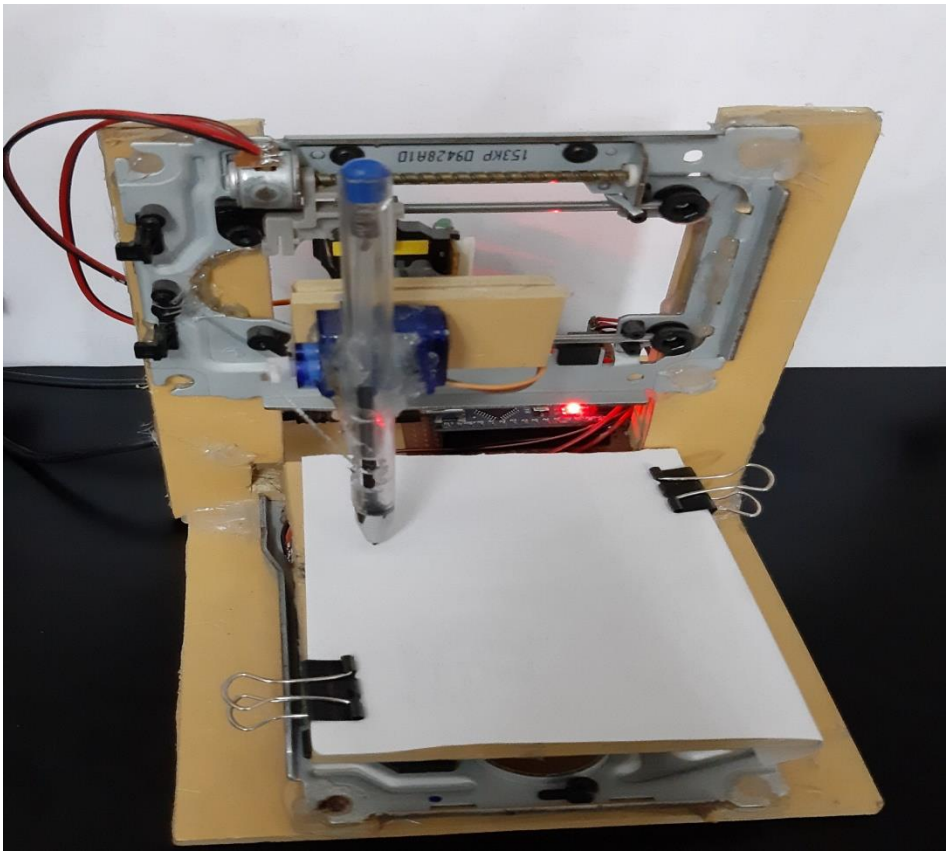


Figure: 4.2: Before Send G-Code

4.2.1 Project After setup

Send G-Code to the Arduino: In the figure 4.2.1, we can see Moving X, Y and Z Axis by following G-Code and making the final output.

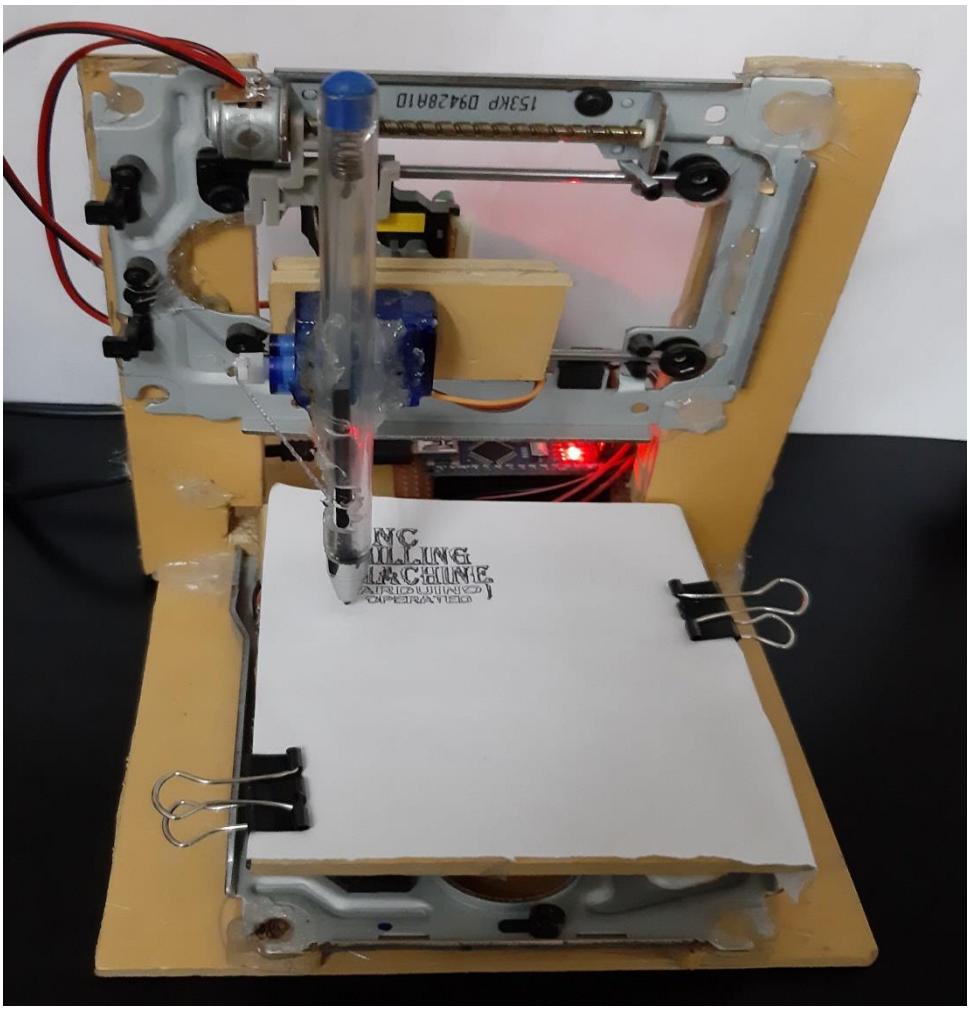


Figure: 4.2.1: After Send G-Code

4.3 Discussion

Finally the constructed circuit and Body are nicely working. At first we done a project, then we used this concept in this project by physically working CNC Workshop, searching internet, studying Books, discussing with my teacher.

Finally we overcome all type problems and complete our project.

4.5 Summary

Firstly we discuss about control the system, then software system to making Design and G-code and transfer that G-code to machine. Then we discuss some output lastly we discuss about discussion.

CHAPTER 05

CONCLUTIONS

5.1 Conclusions

The existing project of the CNC contains much strength, especially the control system that is what makes it more important. With regard to any time it takes to generate a model, accomplishing this of cutting by a CNC is much shorter than executing it by hand, besides not requiring labor, one operator for the machine. The control system incorporates a very efficient communication with the computer application that sends the code. The mechanical system has functionality with the movements in the programmed cutting axis. The mechanical control system along with the software work together, Resulting in a prototype great for slabs in soft materials and high perfection. The control system also comprises the power stage with the motors. In the same way the system includes the software program that contains the G code and will send it towards microcontroller of the control system. In the elaboration of this project we understood how to calibrate each one of the bases of each axis, since this influences the elaboration on the tracks. We understood how to make the interface on the machine with the software, manipulate this system seriously isn't of such complexity. The set of these systems ends up with the roughing of materials such as acrylic and wood. It allows us to record the tracks on the circuits previously made in the program. It truly is proposed as a posterior investigation the adaptation of any plastic injection system that allows the acknowledgement of 3D figures.

5.2 Future Scope

- The pen of the machine can be replaced by a laser so it can be work CNC Laser cutting machine
- CNC Milling Machine (Arduino Operated) can be used on wood.
- This pen can be used both Milling and drilling purpose.
- The servo can be replaced with a stepper motor and the pen with a 3D pento make a 3D printer.

REFERENCE

Ibrahim Zeid. “Mastering SolidWorks : the Design Approach”, Second Edition.

M. Cifuentes & J. S. Jaramillo. “Diseño de un sistema de manufactura automático para circuitos impresos”. Bachelor’s thesis. Facultad de Ingeniería y Tecnología. Universidad Tecnológica de Pereira, Colombia, 2015.

INC, A. E. M. (2008). Sm series stepper motors [Manual de software informático].

Groover, M. P. (2007). Fundamentos de manufactura moderna. México: McGRAW HILL/INTERAMERICANA EDITORES.

Salas, R., Pérez, J., y Ramírez, J. (2007). Técnicas de diseño, desarrollo y montaje de circuitos impresos. Universidad de los Andes. Venezuela.

J. J, Montes & M. Martín. “Control manual para CNC”. Bachelor’s thesis, Universidad de Valladolid, España, 2013.

Groover, M. P. (2007). Fundamentos de manufactura moderna. México: McGRAW HILL/INTERAMERICANA EDITORES.

DMOS Microstepping Driver with Translator And Overcurrent Protection “Driver A4988”, Allegro Micro Systems.

C. Guillen, A. Duque, D. Buelvas, K. Grau & C. Ochoa, “Revisión de sistemas de fresado CNC para la elaboración de placas de circuitos impresos PCB”, Investigación y Desarrollo en TIC, vol. 7, no. 2, pp. 61- 66, 2016.

Benhabib, Beno. Manufacturing: Design, Production, Automation, and Integration New York: Marcel Dekker, 2003

Alonso Rodriguez, Jose Antonio. Sistemas de prototipo rápido, Universidad Politecnica de Valencia. 2002

Antonín Max et al. “Enhancement of teaching design of CNC milling machines”, 2015

APPENDIX

```
#include <Servo.h>
#include <Stepper.h>

#define line_safeguard_distance  512

const int pnZUp = 80;
const int pnZDown = 40;

const int pnServoPin = 6;

const int stepsPerRevolution = 20;

Servo pnServo;

Stepper myStepperY(stepsPerRevolution, 2,3,4,5);
Stepper myStepperX(stepsPerRevolution, 8,9,10,11);

struct point {
  float x;
  float y;
  float z;
};

struct point actuatorPos;
```

```
float SInc = 1;
int SDe = 0;
int LDe = 50;
int pnDe = 50;

float StepsPerMillimeterX = 6.0;
float StepsPerMillimeterY = 6.0;

float Xmi = 0;
float Xma = 40;
float Ymi = 0;
float Yma = 40;
float Zmi = 0;
float Zma = 1;

float Xpos = Xmi;
float Ypos = Ymi;
float Zpos = Zma;

boolean verbose = false;

void setup() {
  Serial.begin( 9600 );

  pnServo.attach(pnServoPin);
  pnServo.write(pnZUp);
```

```
  delay(200);

  myStepperX.setSpeed(250);
  myStepperY.setSpeed(250);

  Serial.println("Mini CNC conspirator alive and dropkick!");
  Serial.print("X area  is from ");
  Serial.print(Xmi);
  Serial.print(" to ");
  Serial.print(Xma);
  Serial.println(" mm.");
  Serial.print("Y area is from ");
  Serial.print(Ymi);
  Serial.print(" to ");
  Serial.print(Yma);
  Serial.println(" mm.");
}

void loop()
{
  delay(200);
  char line[ line_safeguard_distance ];
  char c;
  int lineSignal;
  bool lineIsComment, lineSemiColon;

  lineSignal = 0;
```

```
lineSemiColon = false;
lineIsComment = false;

while (1) {

  while ( Serial.available()>0 ) {
    c = Serial.read();
    if (( c == '\n') || (c == '\r') ) {
      if ( lineSignal > 0 ) {
        line[ lineSignal ] = '\0';
        if (verbose) {
          Serial.print( "Received singlal : ");
          Serial.println( line );
        }
        processIncomingLine( line, lineSignal );
       lineSignal = 0;
      }
      else {

      }
      lineIsComment = false;
      lineSemiColon = false;
      Serial.println("ok");
    }
    else {
      if ( (lineIsComment) || (lineSemiColon) ) {
        if ( c == ')' )  lineIsComment = false;
```

```
      }
      else {
       if ( c <= ' ' ) {
       }
       else if ( c == '/' ) {
       }
       else if ( c == '(' ) {
         lineIsComment = true;
       }
       else if ( c == ';' ) {
         lineSemiColon = true;
       }
       else if ( lineSignal >= line_safeguard_distance-1 ) {
         Serial.println( "ERROR -linesafeguard overflow" );
         lineIsComment = false;
         lineSemiColon = false;
       }
       else if ( c >= 'a' && c <= 'z' ) {
         line[ lineSignal++ ] = c-'a'+'A';
       }
       else {
         line[ lineSignal++ ] = c;
       }
      }
    }
  }
}
```

```
}

void processcomingLine( char* line, int charNB ) {
  int currentsignal = 0;
  char buffer[ 64 ];
  struct point newPos;

  newPos.x = 0.0;
  newPos.y = 0.0;



  while(currentsignal < charNB ) {
    switch ( line[ currentsignal++ ] ) {
    case 'U':
      pnUp();
      break;
    case 'D':
      pnDown();
      break;
    case 'G':
      buffer[0] = line[ currentsignal++ ];
      buffer[1] = '\0';

      switch ( atoi( buffer ) ){
      case 0:
      case 1:
```

```
    char* indexX = strchr( line+currentsignal, 'X' );
    char* indexY = strchr( line+currentsignal, 'Y' );
    if ( indexY <= 0 ) {
      newPos.x = atof( indexX + 1);
      newPos.y = actuatorPos.y;
    }
    else if ( indexX <= 0 ) {
      newPos.y = atof( indexY + 1);
      newPos.x = actuatorPos.x;
    }
    else {
      newPos.y = atof( indexY + 1);
      indexY = '\0';
      newPos.x = atof( indexX + 1);
    }
    draLine(newPos.x, newPos.y );
    //        Serial.println("ok");
    actuatorPos.x = newPos.x;
    actuatorPos.y = newPos.y;
    break;
  }
  break;
case 'M':
  buffer[0] = line[ currentsignal++ ];
  buffer[1] = line[ currentsignal++ ];
  buffer[2] = line[ currentsignal++ ];
```

```
    buffer[3] = '\0';
    switch ( atoi( buffer ) ){
    case 300:
      {
        char* indexS = strchr( line+currentsignal, 'S' );
        float Spos = atof( indexS + 1);
        //          Serial.println("ok");
        if (Spos == 30) {
          pnDown();
        }
        if (Spos == 50) {
          pnUp();
        }
        break;
      }
    case 114:
      Serial.print( "Absolute place  : X = " );
      Serial.print( actuatrPos.x );
      Serial.print( "  -  Y = " );
      Serial.println( actuatrPos.y );
      break;
    default:
      Serial.print( "Command not recognized : M");
      Serial.println( buffer );
    }
  }
}
```

```
}
void draLine(float x1, float y1) {
if (verbose)
  {
    Serial.print("fx1, fy1: ");
    Serial.print(x1);
    Serial.print(",");
    Serial.print(y1);
    Serial.println("");
  }
 if (x1 >= Xma) {
    x1 = Xma;
  }
  if (x1 <= Xmi) {
    x1 = Xmi;
  }
  if (y1 >= Yma) {
    y1 = Yma;
  }
  if (y1 <= Ymi) {
    y1 = Ymi;
  }

  if (verbose)
  {
    Serial.print("Xpos, Ypos: ");
    Serial.print(Xpos);
```

```
  Serial.print(",");
  Serial.print(Ypos);
  Serial.println("");
}

if (verbose)
{
  Serial.print("x1, y1: ");
  Serial.print(x1);
  Serial.print(",");
  Serial.print(y1);
  Serial.println("");
}


x1 = (int)(x1*StepsPerMillimeterX);
y1 = (int)(y1*StepsPerMillimeterY);
float x0 = Xpos;
float y0 = Ypos;

//  Let's find out the change for the coordinates
long dx = abs(x1-x0);
long dy = abs(y1-y0);
int sx = x0<x1 ? StepInc : -StepInc;
int sy = y0<y1 ? StepInc : -StepInc;

long i;
```

```
long over = 0;

if (dx > dy) {
  for (i=0; i<dx; ++i) {
    myStepperX.step(sx);
    over+=dy;
    if (over>=dx) {
      over-=dx;
      myStepperY.step(sy);
    }
    delay(SDe);
  }
}
else {
  for (i=0; i<dy; ++i) {
    myStepperY.step(sy);
    over+=dx;
    if (over>=dy) {
      over-=dy;
      myStepperX.step(sx);
    }
    delay(SDe);
  }
}

if (verbose)
{
```

```
    Serial.print("dx, dy:");
    Serial.print(dx);
    Serial.print(",");
    Serial.print(dy);
    Serial.println("");
   }

   if (verbose)
   {
    Serial.print("Going to (");
    Serial.print(x0);
    Serial.print(",");
    Serial.print(y0);
    Serial.println(")");
   }
  delay(LDe);
   Xpos = x1;
   Ypos = y1;
}
void pnUp() {
   penServo.write(pnZUp);
   delay(LDe);
   Zpos=Zma;
   if (verbose) {
    Serial.println("Pen up!");
   }
}
```

```
void pnDown() {
  penServo.write(pnZDown);
  delay(LDe);
  Zpos=Zmi;
  if (verbose) {
    Serial.println("Pen down.");
  }
}
```

FLOWCHART