**INDEX**

1. INTRODUTION
2. APPARATUS REQUIRED
3. CIRCUIT DIAGRAM
4. CONSTRUCTION
5. CONCEPT OF G-CODE
6. CODING
7. BIBLIOGRAPHY

**INTRODUCTION**

Hand Writing Robot is a simplified version of a CNC (Computer Numerical Control) machine that is an automated control of machining tools and 3D printers by means of computer. It processes a piece of material to meet the required specification by coded programs.

Unlike an Industrial CNC machine, our robot is restricted only to 2D diagrams, it is a machine which moves all around the X and Y axis by using a pen draws the drawing which is uploaded using computer. It uses the chrome extension of G-code sender to send the graphic code, Inkscape software for generating the G-code and for our reference to decide how to draw we use the Camotics software.

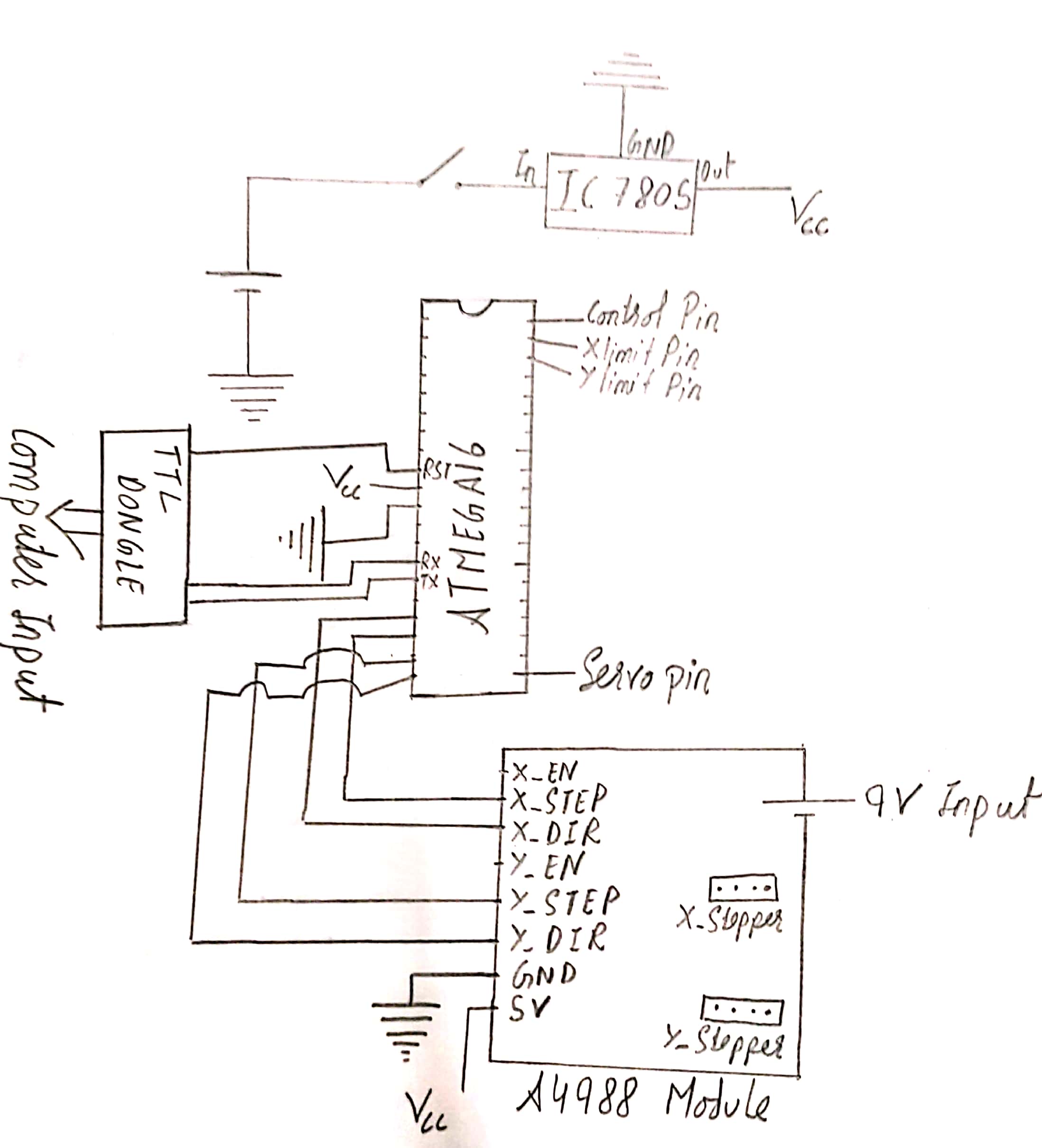
**APPARATUS REQUIRED**

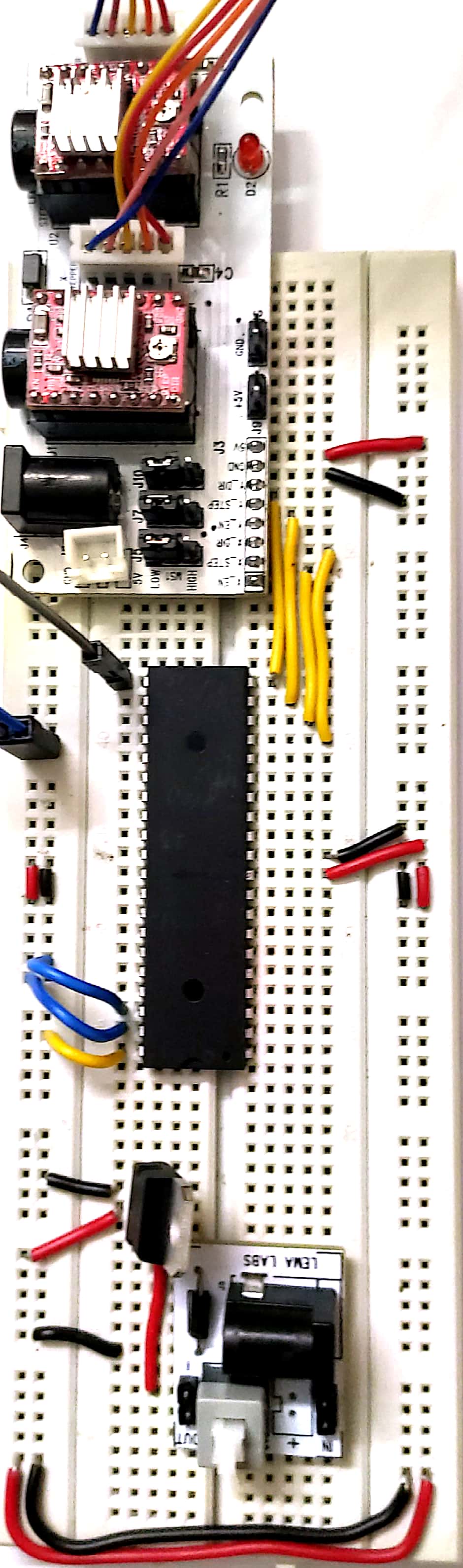
* Atmega16
* A4988 motor driver IC
* Stepper motor
* Servo motor
* AVR programmer
* IC 7805 – 5V voltage regulator
* TTL Dongle
* Switch Module
* Bread Board
* 9V Battery and Adaptor
* Connecting Wires
* Aluminum Rods
* Rotator

**SOFTWARE USED**

* G-Code Sender
* Inkscape
* CAMotics
* Arduino

**CIRCUIT DIAGRAM**





**CONSTRUCTION**

To make Hand Writing Robot, we have to consider x, y and z axis.

**Y and X AXIS:**

Firstly, on a plastic base we have to attach stand offs. Rotator and Stepper Motor are attached with different stand offs. Both rotator and belt are combined with belt.

Then, Aluminum rods are attached with the stand offs so as to give support to writing base. Ball Bearers are used to reduce the rotational friction and support Radial and Axial loads.

**Z AXIS:**

On the aluminium rods of x axis we have attached the base of z axis. Rotator and Servo Motor are attached with different stand offs. A Servo Motor is an electrical device which can push or rotate the belt with great precision. After this a pen stand is also attached with the aluminum rods.

**CONCEPT OF G-CODE**

Graphics Code:

G-Code is a set of instructions which tells the machine where and how to move.

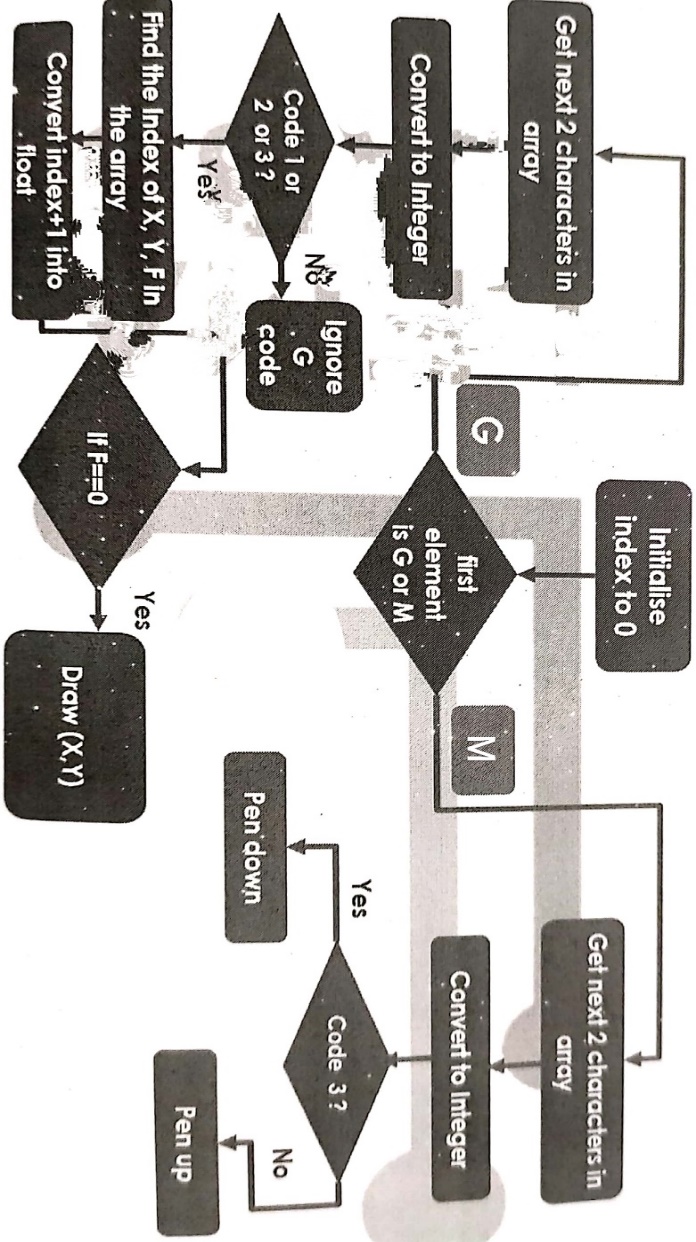
**A G-Code has initials as G or M.**

* G -Gives the X, Y and Z coordinates.
* M – Gives the End effector movements like Pen or Laser.

**G-Code Instructions**

1. M05 – Move the pen up
2. G1 F3000 - Accelerate movement to 3000 mm/min
3. G1 X19.38 Y3.77 – X, Y coordinates
4. M03 – Move the pen down and start writing
5. G1 – Moves the machine in a linear direction
6. G2 – Moves the machine to draw an arc in the clockwise direction
7. G3 – Moves the machine to draw an arc in the counter-clockwise direction

**CODING**

****

**ACTUAL CODE**

#include <Servo.h>

Servo pen;

// Servo on PWM pin 15

const int penServoPin = 15;

const int penDelay = 150; //delay for servo to reach position

//servo angle for up and down

const float penDown = 127;

const int penUp = penDown-30;

// defines pins numbers

const int xstepPin = 12;

const int xdirPin = 11;

const int ystepPin = 13;

const int ydirPin = 14;

const int Control = 24;

const int xlimit = 25;

const int ylimit = 26;

const float away = 1;

const float origin = 0;

int xDir, yDir;

// Motor steps to go 1 millimeter.

// Calculate steps per mm. Enter here.

const float StepsPerMillimeterX = 64.1025;

const float StepsPerMillimeterY = 64.1025;

// Drawing robot limits, in mm

const float Xmin = 0;

const float Xmax = 120;

const float Ymin = 0;

const float Ymax = 120;

//initialise the positions of x and y

float Xpos = Xmin;

float Ypos = Ymin;

char line[40]; //initialise an array to store received G codes

int lineIndex = 0;

char temp[15]; //temporary array to process G code

void xstepper\_rev(int xdir)

{

//set the direction of rotation

digitalWrite(xdirPin, xdir);

//one pulse for the motor to take one step

digitalWrite(xstepPin, HIGH);

delayMicroseconds(800);

digitalWrite(xstepPin, LOW);

delayMicroseconds(800);

}

void ystepper\_rev(int ydir)

{

//set the direction of rotation

digitalWrite(ydirPin, ydir);

//one pulse for the motor to take one step

digitalWrite(ystepPin, HIGH);

delayMicroseconds(800);

digitalWrite(ystepPin, LOW);

delayMicroseconds(800);

}

void not\_writing()

{

pen.write(penUp); //pen up

delay(penDelay); //delay for theservo to reach position

Serial.println("pen is not writing");

}

void writing()

{

pen.write(penDown); //pen down

delay(penDelay); //delay for theservo to reach position

Serial.println("pen is writing now");

}

void setup()

{

//declare all outputs

pinMode(xstepPin, OUTPUT);

pinMode(xdirPin, OUTPUT);

pinMode(ystepPin, OUTPUT);

pinMode(ydirPin, OUTPUT);

//declare input for homing controls

pinMode(xlimit, INPUT);

pinMode(ylimit, INPUT);

//pullup and set control pin as input

pinMode(Control, INPUT\_PULLUP);

//servo pin

pen.attach(penServoPin);

//initially set servo up

pen.write(penUp);

delay(penDelay);

Serial.begin( 9600 );

Serial.println("Mini CNC Plotter alive and kicking!");

Serial.print("X range is from ");

Serial.print(Xmin);

Serial.print(" to ");

Serial.print(Xmax);

Serial.println(" mm.");

Serial.print("Y range is from ");

Serial.print(Ymin);

Serial.print(" to ");

Serial.print(Ymax);

Serial.println(" mm.");

}

void loop()

{

//Manual homing of motors to (0,0)

if (digitalRead(Control) == LOW)

{

pen.write(penDown);

int c = digitalRead(xlimit);

int d = digitalRead(ylimit);

if ((c == 0) && (d == 0))

{

xstepper\_rev(origin);

}

else if ((c == 0) && (d == 1))

{

xstepper\_rev(away);

}

else if ((c == 1) && (d == 0))

{

ystepper\_rev(origin);

}

else if ((c == 1) && (d == 1))

{

ystepper\_rev(away);

}

}

//\*\*\*\*\*\*\*\*\*\* Handwriting \*\*\*\*\*\*\*\*\*\*\*\*\*\*

else if (digitalRead(Control) == HIGH)

{

// Serial reception

while ( Serial.available() > 0 ) //if data is received from the g code sender

{

char e = Serial.read(); //read the data

line[lineIndex] = e; //store the data in the array

//increment the index so that the next char is stored in the next position

lineIndex = lineIndex + 1;

//check if a new line or return catridge is received - indicates end of line

if ((e == '\n') || (e == '\r'))

{

line[lineIndex] = '\0'; //add a null character in th end to indicate end of char array

Serial.print("Received :");

Serial.println(line);

Serial.print("lineIndex =");

Serial.println(lineIndex);

//check if it is a g code or a M code

if ((line[0] == 'G') || (line[0] == 'M'))

{

//process the g code

gcode\_processing(line);

delay(penDelay);

//make the index to 0

memset(line, 0, sizeof(line));

lineIndex = 0;

//send "OK" to indicate that processing is done

//and the machine is ready to receive the next code

Serial.println("ok");

}

}

}

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Processing Gcode\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void gcode\_processing(char\* line)

{

//initialise the index to zero

int currentIndex = 0;

//temporary array

if (line[currentIndex] == 'G') // if it is a G code

{

//read the next two indices to check the code and copy it into temp array

temp[0] = line[currentIndex + 1];

temp[1] = line[currentIndex + 2];

//add null char to temp array

temp[2] = '\0';

//delay(10);

Serial.print("G Command: ");

Serial.println(temp);

//convert the values of the temp array to integer

int code = atoi(temp);

Serial.print("code =");

Serial.println(code);

//check the code

if ((code == 1) || (code == 2) || (code == 3))

{

//if the code is 1 or 2 or 3

//find the position of X, Y, F in the array

char\* xIndex = strchr(line, 'X');

char\* yIndex = strchr(line, 'Y');

char\* FIndex = strchr(line, 'F');

// delay(10);

//convert the values after X,Y, F index to float to check the cordinates

float newXpos = atof(xIndex + 1);

float newYpos = atof(yIndex + 1);

float Fvalue = atof(FIndex + 1);

Serial.print("Fvalue =");

Serial.println(Fvalue);

Serial.print("xIndex =");

Serial.println(xIndex);

Serial.print("yIndex =");

Serial.println(yIndex);

Serial.print("newXpos, newYpos =");

Serial.print("(");

Serial.print(newXpos);

Serial.print(",");

Serial.print(newYpos);

Serial.println(")");

delay(1);

//if there is no F value,Draw

if (Fvalue == 0)

{

draw(newXpos, newYpos);

}

}

//else ingore the G code

else

{

//ignore other gcodes

}

}

// if it is a M code

else if (line[currentIndex] == 'M')

{

//read the next two indices to check the code and copy it into temp array

temp[0] = line[currentIndex + 1];

temp[1] = line[currentIndex + 2];

//add null char to temp array

temp[2] = '\0';

// delay(10);

Serial.print("M Command: ");

Serial.println(temp);

//convert the values of the temp array to integer

int code = atoi(temp);

Serial.print("code =");

Serial.println(code);

//if the code is 3, the pen should write

if (code == 03)

{

writing();

}

//if the code is 5, the pen should not write

else

{

not\_writing();

}

}

}

void draw(float x, float y)

{

Serial.println("entered draw");

//convert the millimeters into steps

x = float(x \* StepsPerMillimeterX);

//convert the millimeters into steps

y = float(y \* StepsPerMillimeterY);

// delay(10);

//old positions -> initially zero

float x0 = Xpos;

float y0 = Ypos;

//initialize change

float ChangeInX = 0;

float ChangeInY = 0;

delay(1);

//if new x is greater than old x, move away from the origin

if (x > x0)

{

Serial.println("x>x0");

xDir = away;

Serial.print("X direction =");

Serial.println(xDir);

//calculate the change in x to find the number of steps

ChangeInX = x - x0;

}

else //if old x is greater than new x, move towards the origin

{

Serial.println("x0>x");

xDir = origin;

//calculate the change in x to find the number of steps

ChangeInX = x0 - x;

Serial.print("X direction =");

Serial.println(xDir);

}

// delay(20);

//if new y is greater than old y, move away from the origin

if (y > y0)

{

Serial.println("y>y0");

yDir = away;

//calculate the change in y to find the number of steps

ChangeInY = y - y0;

Serial.print("Y direction =");

Serial.println(yDir);

}

else //if old y is greater than new y, move towards the origin

{

Serial.println("y0>y");

yDir = origin;

//calculate the change in y to find the number of steps

ChangeInY = y0 - y;

Serial.print("Y direction =");

Serial.println(yDir);

}

// delay(20);

// print the number of steps

Serial.print("Change in X =");

Serial.println(ChangeInX);

Serial.print("Change in Y =");

Serial.println(ChangeInY);

float i;

int over = 0;

//if the change in x is more than the change in Y then move xstepper first and theny stepper

if (ChangeInX > ChangeInY)

{

for (i = 0; i < ChangeInX; i++)

{

xstepper\_rev(xDir);

over += ChangeInY;

if (over >= ChangeInX)

{

over -= ChangeInX;

ystepper\_rev(yDir);

}

delay(1);

}

}

//if the change in Y is more than the change in X then move xstepper first and theny stepper

else if (ChangeInY >= ChangeInX)

{

for (i = 0; i < ChangeInY; i++)

{

ystepper\_rev(yDir);

over += ChangeInX;

if (over >= ChangeInY)

{

over -= ChangeInY;

xstepper\_rev(xDir);

}

delay(1);

}

}

Serial.println("Finished Drawing");

delay(6); //line delay

//set the current values to xpos and ypos to use for next calculations

Xpos = x;

Ypos = y;

}

**BIBLIOGRAPHY**

* wikipedia.org
* [semmimd.com](https://semmimd.com/)
* semiwiki.com