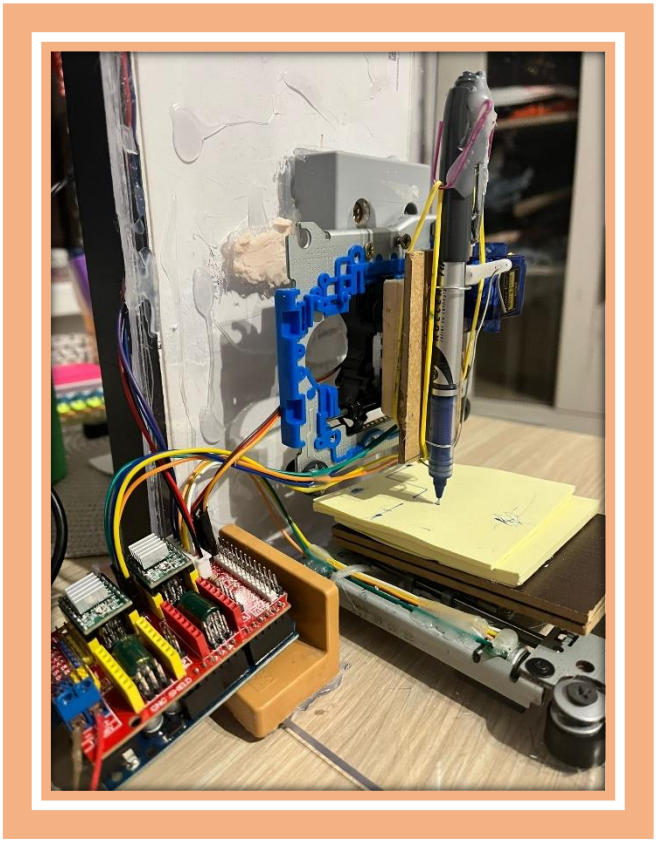
Universitatea Tehnică din Cluj-Napoca
Facultatea de Automatică și Calculatoare
Calculatoare și Tehnologia Informației

Program pentru controlul unui utilaj cu commandă numerică



Student: Lînaru Petra Rozalia

Cuprins

1. Introducere

Provocarea acestui proiect este implementarea unui program pentru comanda unui utilaj cu comandă numerică, a cărui mod de funcționare se aseamănă cu cel al unui plotter. Pe scurt, scopul acestui utilaj este acela de a primi comenzi de la computer, pe care apoi să le interpreteze și să transpună pe hârtie, cu ajutorul unui instrument de scris, puncte sau linii. Utilajul va trebui să deseneze ceea ce îi cere utilizatorul.

1.1 Interpretarea comenzilor primite

Programarea utilajelor/ mașinilor-unelte cu comandă numerică reprezintă secvențe de instrucțiuni utilizate pentru a controla mașina-unealtă. Aceste instrucțiuni de obicei constă într-un combo de litere și numere : fiecare literă corespunde unei categorii acțiuni pe care programul le execută, urmate uneori fie de coordonatele pe axele X,Y (exemplu : G0 X8 (coordonata-X) Y13 (coordonata-Y), fie de diverse constante. Instrucțiuni sunt primite printr-un port serial și indică direcția de mișcare a actuatoarelor(a markerului, în cazul nostru), în funcție de instrucțiunile primite. Astfel, omul reușește să comunice utilajului cum să facă un anumit lucru.


```
G00 X-25 Y20
G91 X-10
G02 X10 Y-10 I10 J0 F200 M3
G02 X-10 Y10 I0 J10;
G90 G00 M5 X0 Y0;
G00 X25 Y20
G91 X-10
G02 X10 Y-10 I10 J0 F200M3
G02 X-10 Y10 I0 J10;
G90 G00 M5 X0 Y0;
G00 X40 Y0
G02 X-40 Y0 I-40 J0 F200 M3
G90 G00 M5 X-55 Y0;
G02 X0 Y-55 I55 J0 F200 M3
G02 X-55 Y0 I0 J55
```

[1](Robotics&Automation, 2018)

Utilajele cu comandă numerică sunt mașinării a căror funcționare este bazată pe motoare stepper. În cazul nostru, utilajul de comandă dispune de două astfel de motoare stepper, pentru a putea permite mișcarea pe cele două axe : X și Y. Motoarele stepper sunt motoare care împart o rotație completă într-un număr egal de pași mai mici[2]. Astfel, poziția markerului va fi putea comandată pentru a se schimba, a aștepta sau pentru a apropia vârful markerului de hârtie. Acesta va primi comenzile de la utilizator, după care, cu ajutorul unui

braț de capătul căruia va fi legat un marker, motorul va acționa și va permite desenarea pe hârtie, conform instrucțiunilor primite. [3] (Wikipedia, n.d.)

1.2 Specificații

Creierul acestei întregi mașinării (codul) va fi încărcat pe o plăcuță Arduino, care va controla următoarele mișcări în funcție de instrucțiunile care vor fi transmise de către interfața grafică prin intermediul portului serial. Arduino este o companie open-source care produce atât plăcuțe de dezvoltare bazate pe microcontrolere, cât și partea de software destinată funcționării și programării acestora. Pe lângă acestea include și o comunitate uriașă care se ocupă cu creația și distribuirea de proiecte care au ca scop crearea de dispozitive care pot sesiza și controla diverse activități sau procese în lumea reală [4], motiv pentru care o astfel de plăcuță este perfectă pentru a crea utilajul cu comandă numerică dorit. De asemenea, pentru a mișca pixul pe cele două axe, se vor obține din două drive-uri de DVD/CD motoarele de tip stepper, iar pentru axa Z (baza care va susține pixul) se folosește un Mini Servo Motor. Pentru a atenționa utilizatorul cu privire la starea mașinăriei, se vor folosi două led-uri : unul verde și unul roșu.

2. Studiu Bibliografic

2.1 Codul G

Pentru a obține o mai bună înțelegere a lucrurilor pe care le pot face mașinăriile și modul în care acestea acționează, înțelegerea codului care le ghidează este esențială. Codul G (G-Code) este limbajul simplu pentru programarea utilajelor cu comandă numerică. Acesta conține linii de cod organizat în blocuri. La rândul lor, fiecare bloc de instrucțiuni controlează un utilaj de comandă numeric și este notat cu litera N și un anumit lucru.

Categoriile de acțiuni sunt evidențiate prin diverse litere:

- M : coduri pentru acțiuni ale mașinii (M00 – oprirea programului , M02 – încheierea programului)
- G : coduri pentru mișcarea mașinăriei într-un anumit mod, urmate de coordonatele spațiale (G0 – mișcare rapidă, G01 – mișcare controlată, G02 – mișcare în sensul acelor de ceasornic, exemplu G02 X10 Y7 I0 J-5)

- T : codul care specifică unealta cu care se va lucra
- S : codul care controlează viteza cu care acționează unealta
- F : codul care controlează cât de rapid mașinăria se va mișca în timp ce aceasta urmărește coordonatele spațiale [5]

3.Analiză

Scopul acestui capitol este acela de a prezenta funcționalitățile proiectului, precum și cazurile de utilizare si distribuirea eficientă a sarcinilor care trebuie îndeplinite astfel încât planificarea proiectului să fie una clară și bine delimitată.

3.1 Propunere de proiect

Acest proiect are ca și scop aprofundarea cunoștințelor despre mașinăriile cu comandă numerică, precum și despre limbajul folosit pentru programarea acestora, a componentelor care întră în componența acestora, dar și a modului în care acestea funcționează și interacționează cu utilizatorii.

Astfel, propunerea mea de proiect se referă la introducerea unei interfețe grafice prin care comunicarea utilizator-mașinărie să fie una mult mai prietenoasă, iar utilizatorul va putea specifica ce anume dorește să deseneze cu ajutorul mașinăriei, având la dispoziție o serie de butoane prin care fie poate controla mișcarea axelor. Odată cu această propunere sosesc multe alte curiozități legate de librăriile care vor fi implementate și modul în care se va realiza această "comunicare", curiozități care vor fi explicate mai pe larg pe parcursul capitolelor viitoare.

O altă propunere de proiect ar fi aceea de a permite utilizatorului să deseneze linii, ale căror coordonate să fie transformate în G-Code și să fie apoi transmise către mașinărie. Această propunere vine, de asemenea, cu multe alte oportunități de a aprofunda Codul G, precum și interpretarea input-ului de la utilizator astfel încât să poată fi "tradus" automat în limbajul mașinăriei.

Ambele propuneri vor avea la bază o interfată grafică, care are la bază ca și limbaj de programare Java și care va fi realizată în Processing. Utilizatorilor li se va pune la dispoziție să aleagă fie o direcție în care să deseneze liber, fie să traseze o linie.

Processing este o bibliotecă grafică gratuită și un mediu de dezvoltare integrat construit pentru comunitățile de arte electronice, arta new media și design vizual, cu scopul de a preda

non-programatorilor elementele fundamentale ale programării computerelor într-un context vizual.[6]. Interfața comunică cu Arduino prin intermediul portului serial, comenzile fiind trimise prin acesta de fiecare dată când utilizatorul alege o opțiune sau apasă un buton.

3.2 Funcționalitățile proiectului

Din punctul de vedere al funcționalităților, acest proiect trebuie să bifeze următoarele cerințe:

- Să permită utilizatorului să deseneze liber folosindu-se de butoanele “UP”, “DOWN”, “LEFT”, “RIGHT”, corespunzătoare direcțiilor de pe axele X și Y
- Să alerteze utilizatorul în cazul în care input-ul este necorespunzător
- Să imprime corect liniile dorite/ să urmeze direcțiile introduse de la interfață.

3.3 Cazuri de utilizare

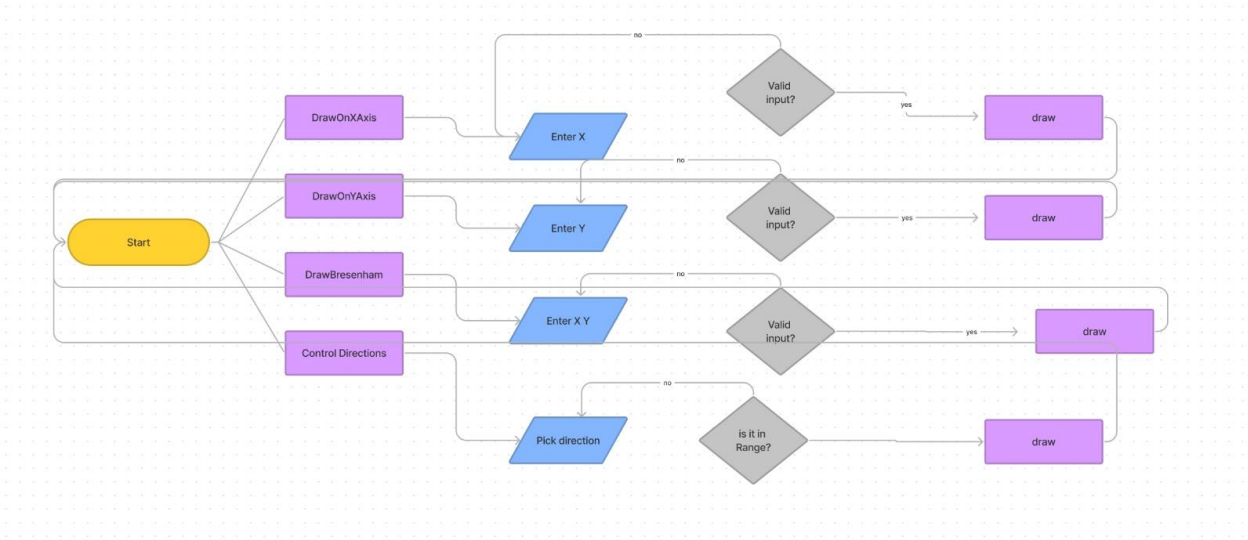
Cazurile de utilizare sunt ilustrate în diagrama de mai jos.

Utilizatorul pornește aplicația și poate să aleagă între atrasa o linie, fie a desena liber cu ajutorul butoanelor.

În cazul în care utilizatorul alege să traseze o linie. Pentru a trasa o linie presupunem că punctul inițial se află la (0,0) și se deplasează spre punctul indicat de utilizator, iar mașinăria trebuie doar să interpreteze acest cod, și să cunoască timpii cât pixul se află pe hârtie și când trebuie să stea ridicat.

În cazul în care utilizatorul alege sa deseneze o formă proprie, pe ecran va avea la dispoziție 4 butoane corespunzătoare celor 4 direcții de desenare, precum și două butoane cu ajutorul cărora poate ridica sau pune pixul pe hartie.

De asemenea, mașinăria va face distincția, în momentul în care analizează formele, între mișcare stânga-dreapta, pe axa X, mișcare sus-jos, pe axa Y și rapiditatea cu care,pornind dintr-un punct, trebuie să ajungă în alt punct, fie cu pixul pus jos, fie cu pixul ridicat.



[7]

3.4 Plan de dezvoltare

- •Alegerea proiectului
- •Studiu asupra temei, a mașinilor cu comandă numerică si a codului G
- •Analiza proiectului, descrierea funcționalităților, a cazurilor de utilizare
- •Studiu asupra îmbunătățirilor care pot fi aduse proiectului
- •Începerea implementării, obținerea stepperelor din DVD
- •Îmbinarea componentelor, efectuarea primelor teste asupra mișcărilor
- •Testarea primelor imagini desenate cu ajutorul GCTRL program
- •Aprofundarea codului G - primul cod G generat cu ajutorul Python
- •Îmbunătățirea proiectului cu ajutorul sugestiilor
- •Implementarea interfeței grafice cu opțiunea de a desena o imagine deja existenta
- •Testarea primei faze în care se află interfața împreună cu mașinăria
- •Îmbunătățirea proiectului cu ajutorul sugestiilor
- •Completarea interfeței cu opțiunea de a desena într-o fereastră
- •Testarea noii interfețe împreună cu mașinăria
- •Testarea întregului proiect
- •Formularea unor concluzii

4. Implementare

4.1 Îmbinarea componentelor și controlul acțiunilor pe axele X, Y, Z

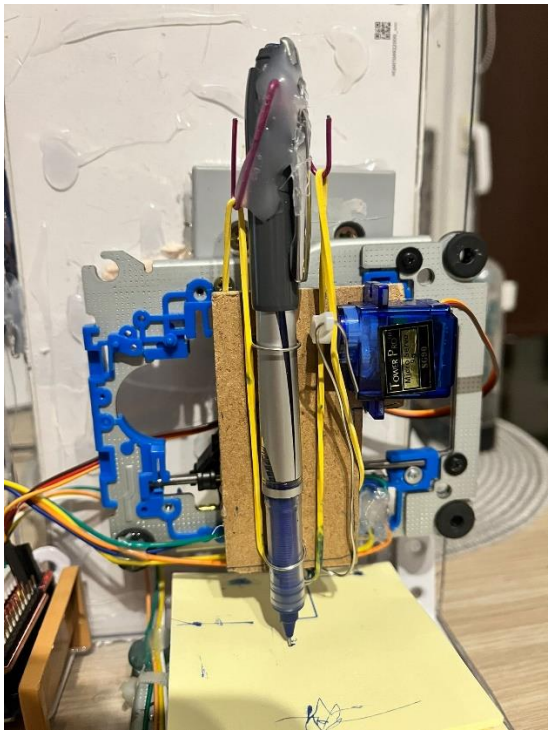
Pentru mișcarea pe axele X și Y voi folosi două mecanisme de citire-scriere pe care le-am obținut din două CD Drivere și care vor juca rolul suportului pentru hârtie și rolul suportului pentru pix. Aceste două componente vor fi așezate pe un suport din plexiglas.

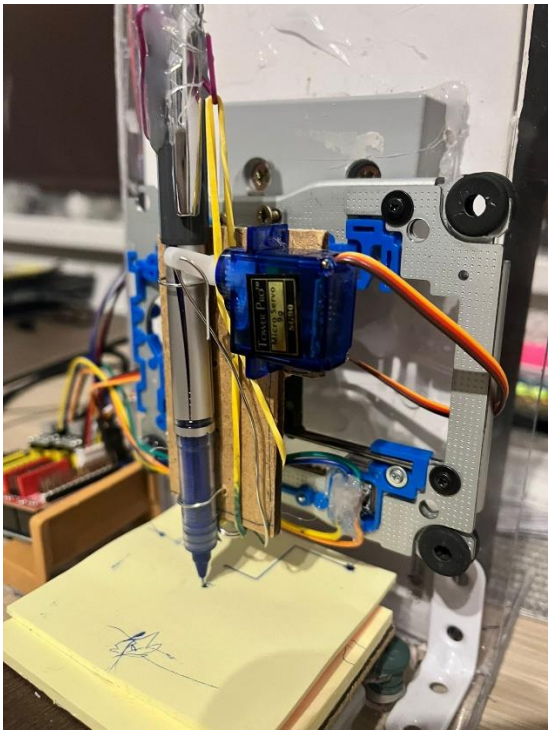
Suportul va consta din două plăci plexiglas perpendiculare una pe cealaltă, care vor fi îmbinate cu ajutorul unor colțari la 90 de grade și a unei cutii. Pe suprafața acestora voi lipi cele două mecanisme :

- mecanismul care va susține suportul de hartie
- mecanismul care va susține motorul servo și pixul împreună cu suportul său

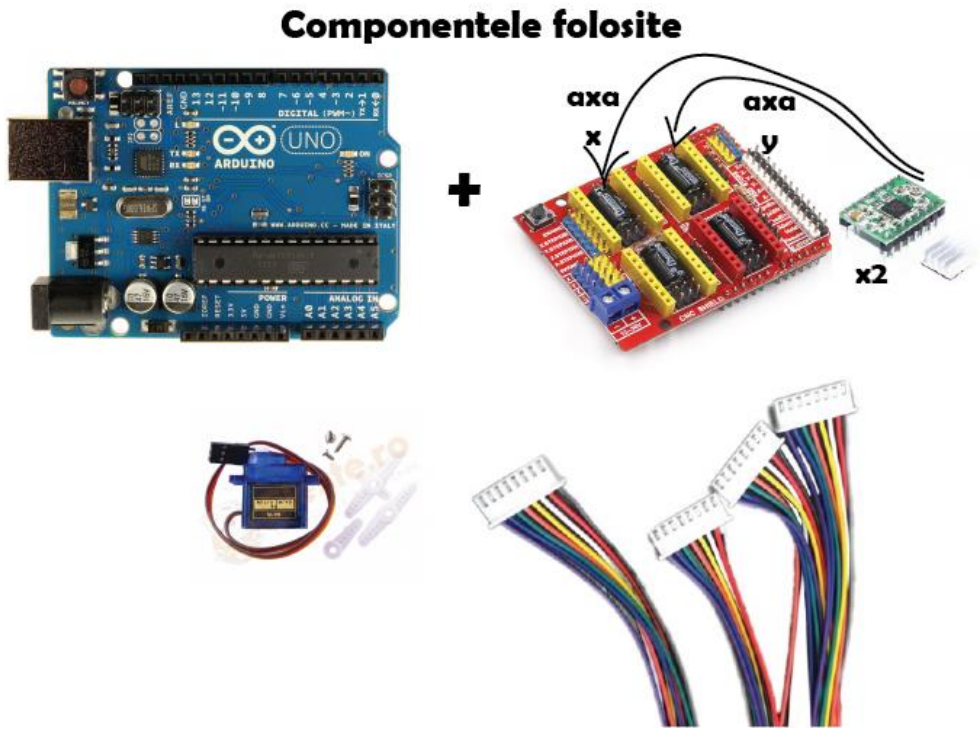
Problema care apare vizează axa Z, respectiv legătura dintre motorul servo și pixul pe care va trebui să îl acționeze. Soluția mea constă în construirea unui mecanism de susținere a pixului care să îndeplinească două roluri: suport stabil pentru pix, dar care să poată fi acționat de motorul servo. Astfel, soluția mea este una creativă : modelez un arc care să susțină vârful pixului și care sa fie legat de aripa motorului servo. Mai mult de atât, fixez pixul pe suportul axei Z cu ajutorul unor cleme care să permită mișcarea acestuia sus-jos. Pentru a conferi acestuia mai multă stabilitate, pe capacul pixului lipesc o agrafă de birou, modelată pe forma cilindrică și orientez cele două vârfuri în sus. Astfel, cu ajutorul a două elastice, leg arcul care susține vârful pixului de cele două vârfuri ale agrafei de birou și forțez pixul să stea în poziție dreaptă, și să poată fi mișcat cu ușurință sus-jos.

O altă problemă întâlnită în construirea suportului a fost cea a organizării firelor și a plasării plăcuței și a shield-ului astfel încât să permită mișcări ușoare și să nu încurce cele două mecanisme. Astfel, firele le-am lipit în puncte cheie, iar pentru plăcuță și shield am lipit un colțar de plastic pe care să le poziționez (arată amuzant, ca o canapea).





4.2 Schema

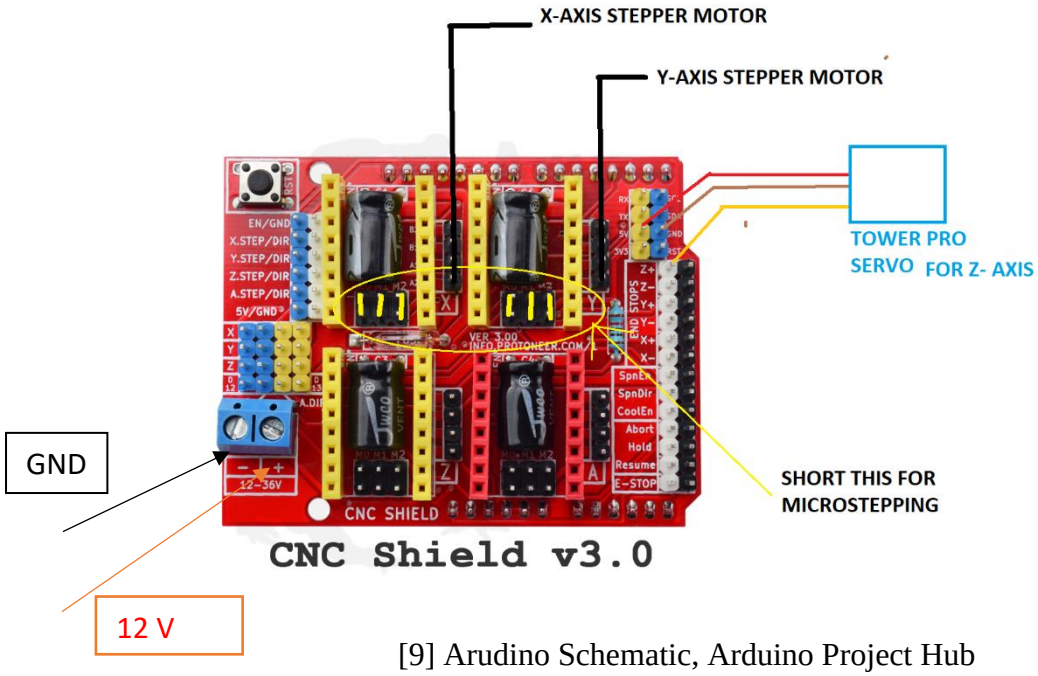


[8] Componente folosite

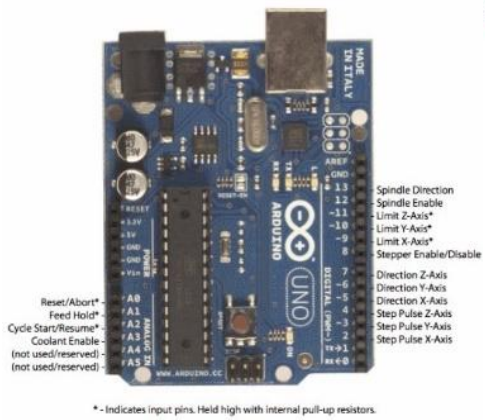
Logica care sta în spatele controlului acestui CNC este una complexă, dar care poate fi ușor înțeleasă urmărind figurile prezente în acest sub-capitol. Componentele folosite sunt cele

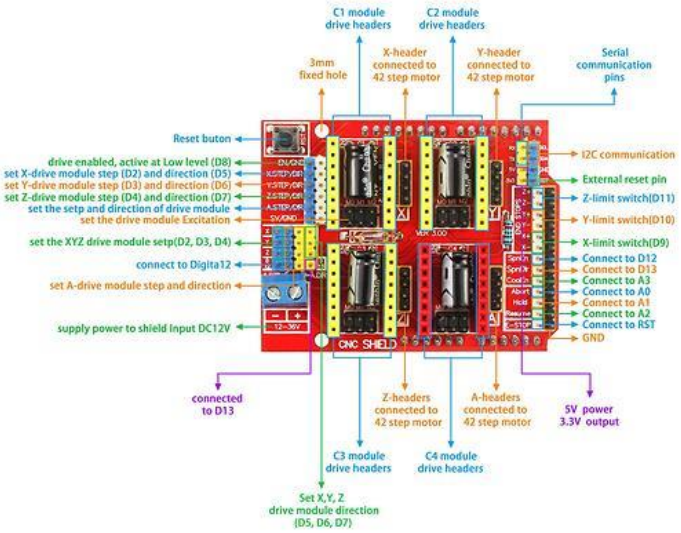
prezentate in figura [8] – O placă de dezvoltare compatibilă cu Arduino UNO R3, un shield CNC A4988 V3, un micro servo motor SG90, cabluri conector pentru motoare pas cu pas și două drivere pentru motoare pas cu pas, A4988.

Cele două drivere sunt montate în cele două spații dedicate driverelor pentru axele X, respectiv Y și ele vor fi responsabile pentru controlul mișcărilor pe cele două axe (practic ele funcționează ca niște intermediari între comenzile trimise de la calculator și motoarele fizice). Motorul Servo este montat pe partea dreaptă a Shield-ului, având firul roșu conectat la 5V, cel maroniu la GND, iar firul galben fiind conectat la pin-ul care face legătura cu axa Z. Aceste legături sunt prezentate în figura 8. De asemenea, avem o sursă de alimentare de curent continuu la care am conectat un adaptor de 12V, 1A



[9] Arudino Schematic, Arduino Project Hub





[10] r/CNC forum on reddit [11] Arduino CNC V3, Mikroelectron

Partea interesantă (și care mi-a ocupat cel mai mult timp) este cea legată de transpunerea în cod a tuturor acestor explicații, motiv pentru care figurile [9] și [10] sunt extrem de utile. Pinii care sunt responsabili de mișcarea axelor sunt organizați astfel:

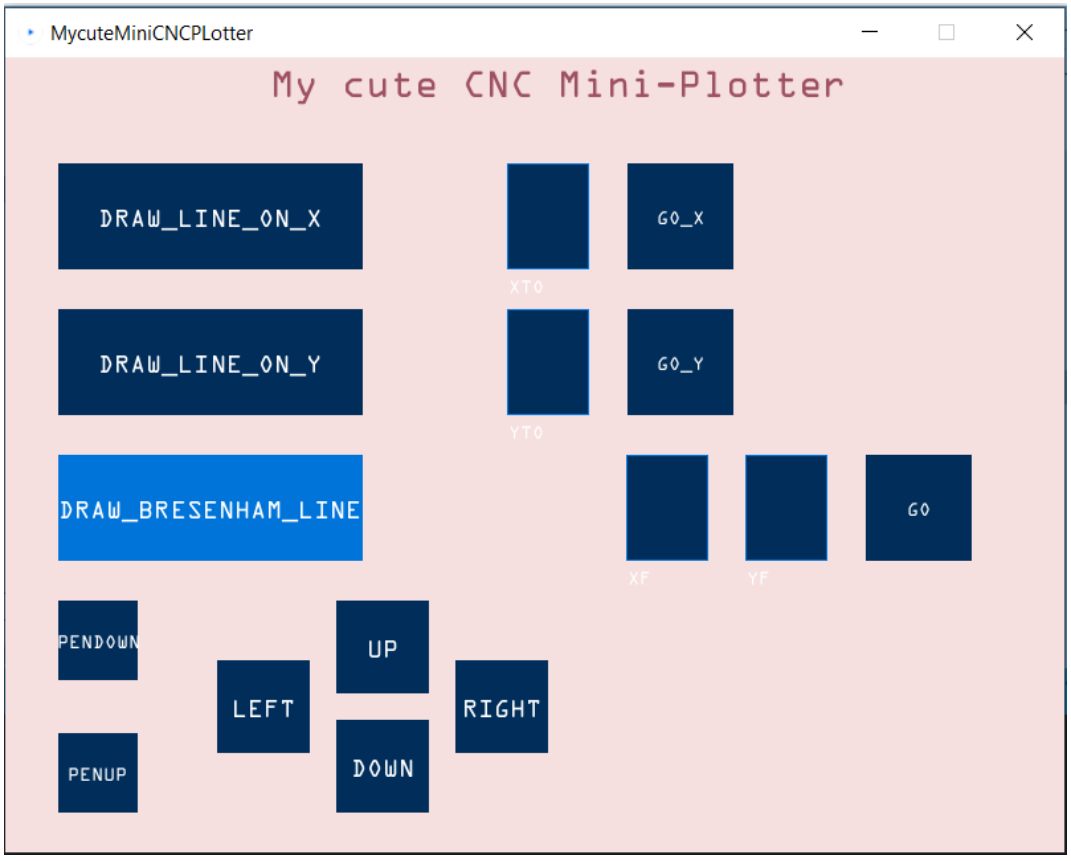
- Axa X : 2 (Step Pulse X), 5 (Direction X)
- Axa Y : 3 (Step Pulse Y), 6 (Direction Y)
- Axa Z : 4 (Step Pulse Z), 7 (Direction Z)

Un alt lucru extrem de important care trebuie menționat este acela că enable este setat pe pinul 8 și este activ pe 0!

De asemenea, pentru a acționa motorul servo ne folosim de librăria Servo.h și de 2 funcții extrem de importante : attach(servoPin) și write(angle). Astfel, pinul pe care este atașat motorul servo este pinul 11 și ridicarea, respectiv coborârea acestuia se face cu ajutorul funcției write(angle). Pentru a ridica pixul, am declarat unghiul de ridicare ca fiind penZUp=180, iar pentru a-l coborî am declarat unghiul de coborâre ca fiind penZDown=90. Astfel, în combinație cu sistemul de prindere al pixului și suportul acestuia, motorul servo funcționează conform așteptărilor : pixul este acționat pe axa Z, iar mișcarea este una controlată, care ne ajută să plasăm puncte în locațiile dorite / să traseze o linie.

Cu toate acestea, schema este completă, și putem efectua primele teste pentru mișcarea pe axa X, pe axa Y, apoi miscări combinate pe axele XY.

4.3 Interfața grafică



Interfața grafică este un punct important al acestui proiect, deoarece ea are un rol destul de important : ea preia comenzile de la utilizator, le transformă în GCode și le transmite mai departe spre Arduino.

4.3.1 Preluarea comenzilor de la utilizator

În momentul în care utilizatorul deschide aplicația, acestuia îi sunt prezentate opțiunile pe care le are la dispoziție : draw line on X, draw line on Y, draw bresenham line, fie mișcare liberă.

Dacă utilizatorul alege una dintre opțiunile de a desena o linie, la apăsarea unui buton apar textfield-urile în care acesta trebuie să introduce coordonatele necesare. Pentru a desena o linie pe axa X sau Y, utilizatorul este rugat să introducă punctul final pe unul dintre cele cele 2 axe, desenarea punctului se face începând cu originea. Comanda este trimisă la Arduino prin portul serial, prin concatenarea unor string-uri cu valorile coordonatelor. Pentru a desena o linie folosind algoritmul lui Bresenham, utilizatorul este rugat să introducă coordonatele punctului final pentru prima trasare a liniei, pornind din origine, apoi punctul inițial de desenare se

schimbă cu punctul în care se află pixul. De asemenea, comanda finală trimisă se obține prin concatenarea string-urilor care reprezintă codul comenzii, respectiv G1, și coordonatele.

Dacă utilizatorul alege să deseneze liber, prin apăsarea butoanelor care reprezintă direcțiile de desenare, prin portul serial sunt trimise comenzi personalizate care apoi să fie interpretate și să pună în acțiune stepperele. Comenzile personalizate sunt „U”, „D” , „L”, „R”, care apoi sunt procesate în interpretorul de comenzi, alături de toate celelalte comenzi.

4.4 Interpretorul de comenzi

Comenzile trec prin două etape : procesarea liniei care vine de la interfață și procesarea comenzii. Astfel, fiecare linie este citită și trece printr-un process în care sunt eliminate spațiile libere și eventualele greșeli de scriere, după care în cazul în care s-a ajuns la finalul liniei, aceasta este trimisă spre următoarea etapă, cea de procesare a comenzii.

Pentru procesarea comenzii, linia este împărțită în funcție de codul comenzii, printr-un switch. Fiecare ramură a switch-ului conține alte operații care trebuie effectuate asupra liniei pentru a obține coordonatele finale, fie comenzi care trebuie executate.

Funcția declară mai întâi o structură numită „punct”, care are două variabile x și y, și o variabilă „currentIndex” care este folosită pentru a ține evidența poziției curente în șirul de comandă. Apoi, inițializează două variabile newPos.x și newPos.y la zero și declară un buffer de 64 de caractere.

Funcția intră apoi într-o buclă while care iterează prin fiecare caracter al șirului de comandă. În cadrul buclei, există o instrucțiune switch care verifică valoarea caracterului curent și efectuează o acțiune în funcție de caracter.

Codul folosește caracterele „0” și „1” pentru a controla pixul, unde „0” pune pixul jos, iar „1” pune pixul sus. „U” și „D” sunt folosite pentru a muta cursorul în sus și, respectiv, în jos. „L” și „R” sunt folosite pentru a muta cursorul la stânga și la dreapta.

Cel mai important caz este cazul „G”, în care codul folosește funcția strchr pentru a găsi pozițiile caracterelor „X” și „Y” în șirul de comandă, apoi folosește funcția atof pentru a converti caracterele care urmează caracterele „X” și „Y” în numere în float. Aceste valori sunt apoi folosite pentru a actualiza poziția cursorului. Funcția apelează apoi drawBresenhamLine cu noua poziție ca argumente, care este responsabilă pentru actualizarea poziției cursorului pe ecran folosind algoritmul Bresenham de trasare a liniei.

4.5 Algoritmul de trasare a liniei

Algoritmul de trasare a unei linii folosit este algoritmul Bresenham.

Algoritmul Bresenham de trasare a unei linii este un algoritm ce determină care puncte dintr-un raster n-dimensional trebuie aprinse pentru a forma o aproximaţie a unei linii drepte între două puncte date. Acest algoritm este folosit pentru a trasa linii pe ecranele calculatoarelor, folosind doar adunarea numerelor întregi, scădere şi schimbare de bit, operaţii rapide pe calculatoarele din zilele noastre. Este unul dintre primii algoritmi apărut în domeniul graficii pe calculator.

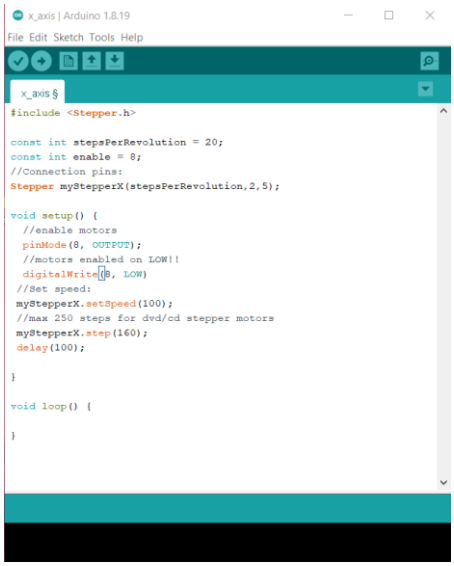
Printr-o mică modificare, algoritmul original pentru trasarea unei linii poate fii folosit şi pentru a trasa cercuri. De asemenea, se poate rezolva cu operaţii aritmetice simple, ecuaţiile pătratice şi expresiile trigonometrice pot fii evitate sau sparte recursiv în bucăţi mai mici.[12]

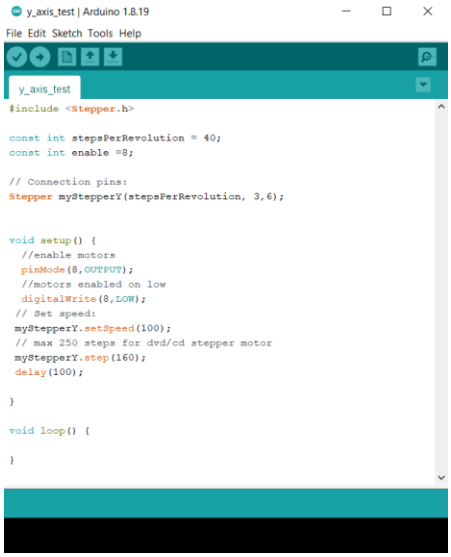
Funcția primește doi parametri, x1 și y1, care reprezintă punctul final al liniei. Funcția verifică apoi dacă aceste valori se încadrează în valorile maxime și minime ale grilei (Xmax, Xmin, Ymax, Ymin). Dacă sunt în afara limitelor, sunt setate la cea mai apropiată limită.

Funcția calculează apoi numărul de pași necesari pentru a trage linia determinând modificarea în x și y între punctul de început și punctul final și semnele acestor modificări. Algoritmul folosește apoi aceste valori pentru a calcula poziția următorului punct pe linie folosind o variabilă „p” care este utilizată pentru a determina dacă punctul curent este deasupra sau sub linie.

Algoritmul folosește apoi o buclă for pentru a itera fiecare punct de pe linie. În fiecare iterație, funcția verifică valoarea lui „p” pentru a determina dacă următorul punct ar trebui să fie deasupra sau sub linie și apoi actualizează poziția următorului punct în consecință. Funcția apelează, de asemenea, funcția putPen() care este responsabilă pentru punerea pixului pe hârtie.

5.Testare





Pentru a testa axele X și Y, precum și limitele de pe fiecare axă, precum și numărul de pași efectuați într-o direcție, am folosit testele de mai sus și am ajuns la anumite concluzii.

Dimensiunea suprafeței de desenat este de 40 x 40 mm, dar vom lua în considerare doar 39 x 39 mm pentru a lăsa loc și pentru margini.

Unghiul de rotație al motorului îl considerăm a fi de 1.8 grade (cel mai des întâlnit la motoarele stepper din CD/DVD Drive). Astfel, atunci când motorul primește un impuls, axul se va roti cu 18 grade și vom avea nevoie de 360 / 18 = 20 de pași pentru ca un motor să realizeze o rotație completă. În altă ordine de idei, avem nevoie de 20 de impulsuri pentru a efectua o rotație completă, adică pentru a parcurge întreaga distanță pe care o putem desena pe una dintre cele două axe.

MS1	MS2	MS3	STEP RESOLUTION	STEPS PER REVOLUTION
0	0	0	1	20

[13] “Back to basics, MINI CNC PLOTTER”, Instructables

De asemenea, pentru a obține mișcări cât mai exacte, vom avea nevoie de câteva calcule, luând în considerare dimensiunea șurubului care intră în componența motorului (50 mm pe axa X, considerăm egal și pe axa Y) și dimensiunea unui segment de pe șurub (aprox 3 mm)

Pentru a calcula câți pași vom face pe mm vom folosi un calcul simplu, experimental.

Motoarele sunt configurate în așa fel încât fac 20 de pași/revoluție, adică au nevoie de 20 de impulsuri a câte x pași pentru a ajunge de la 0 mm la 40 mm. După mai multe încercări, am concluzionat că la un impuls, un motor poate face 55 de pași și acei 55 de pași să reprezinte 1/20 din linia finală, pentru o obține o linie cât de cât aspectuoasă.

La 10 impulsuri a 55 de pași, un motor parcurge 20 mm pe direcția setată. Astfel, ajungem la un calcul rapid :dacă la 10 impulsuri a câte 55 de pași (= 550 pași), parcurgem 20 mm, rezultă că la 40 mm vom efectua 1100 pași. Astfel, punctul limită exprimat în nr. Pași va fi 1100, iar pentru increment vom folosi 55 de pași.

Bibliografie

[1]"How to become a G-Code master with a complete list of G-Codes". (2018). Preluat de pe Robotics&Automation: https://roboticsandautomationnews.com/2018/01/26/how-to-become-a-g-code-master-with-a-complete-list-of-g-codes/15807/

[2]"G-Code Knowledge is Key to Mastering Any CNC Machine". (fără an). Preluat de pe fictive: https://fictive.com/articles

[3]"CNC Drawing Plotter". (fără an). Preluat de pe Arduino Project Hub: https://create.arduino.cc/projecthub

[4] "Programarea mașinilor-unelte cu comandă numerică". (fără an). Preluat de pe Wikipedia: http://ro.wikipedia.org.wiki

[5]"Arduino". (fără an). Preluat de pe Wikipedia: https://ro.wikipedia.org/wiki

[6] https://en.wikipedia.org/wiki/Processing

[9] "How to make GRBL+ CNC V3 Shield based MINI CNC" , https://create.arduino.cc/projecthub/mrinnovative01/how-to-make-grbl-cnc-v3-shield-based-mini-cnc-5a2e69

[10]" Need help with Connecting an arduino with GRBL to a cNC-controller with a db25 LPT-port ", Reddit, https://www.reddit.com/r/CNC/comments/5depj0/need_help_with_connecting_an_arduino_with_grbl_to/

[11]"Arduino CNC v3", https://mikroelectron.com/Product/ARDUINO-CNC-SHIELD-V3/

[12] https://digipedia.ro/algoritmul-bresenham-de-trasare-a-unei-linii/

[13]“Back to basics, MINI CNC PLOTTER”, Instructables, https://www.instructables.com/BACK-TO-BASIC-MINI-CNC-PLOTTER/