# Introducere

Pentru lucrarea de licență mi-am propus să fac o mașină CNC capabilă să deseneze sau să graveze in lemn diferite imagini sau text, prin intermediul unui limbaj standard folosit de CNC-uri sau imprimante 3D numit GCODE.

Aplicația este realizată pentru platforma hardware Arduino Uno. Aceasta primește comenzi GCODE prin intermediul portului USB si le execută cu ajutorul unor stepo-motoare rezultând astfel diferite tipuri de mișcări în spațiul cartezian XYZ.

Exista câteva firmware-uri similare. Am venit cu ceva diferit prin faptul că am ales să scriu aplicația intr-o maniera OOP, cu un cod abstractizat foarte mult de partea hardware, ușor de înțeles si configurat.

Pentru că m-am axat mai mult pe partea de software, pentru partea mecanică am folosit o platformă deja existentă pe care am adaptat-o la cerințele mele.

**<IMG: cu cnc>**

In primul capitol voi vorbi despre resursele software, hardware si cele mecanice utilizate pentru realizarea si utilizarea CNC-ului.

In cel de-al doilea capitol voi discuta despre asamblarea componentelor hardware si a componentelor mecanice. Voi discuta despre modul cum funcționează platforma hardware aleasă împreună cu celelalte componente hardware menționate în subcapitolul 1.2.

In al treilea capitol.

Contents

[Introducere 1](#_Toc536401814)

[1 Resurse utilizate 3](#_Toc536401815)

[1.1 Software 3](#_Toc536401816)

[1.1.1 Aplicația CNC Arduino 3](#_Toc536401817)

[1.1.2 Aplicația de comunicație cu Arduino 3](#_Toc536401818)

[1.1.3 *Tool*-uri pentru conversia imaginilor 3](#_Toc536401819)

[1.2 Hardware 3](#_Toc536401820)

[1.2.1 Microcontroler 4](#_Toc536401821)

[*1.2.2* CNC *shield* 5](#_Toc536401822)

[1.2.3 Motoare pas cu pas 6](#_Toc536401823)

[1.2.4 Drivere pentru motoarele pas cu pas 6](#_Toc536401824)

[1.2.5 *Endstop*-uri pentru limitarea axelor 8](#_Toc536401825)

[1.2.6 Sursă de alimentare 8](#_Toc536401826)

[1.3 Mecanice 8](#_Toc536401827)

[1.3.1 Componente printate din plastic 8](#_Toc536401828)

[1.3.2 Tije cromate 8](#_Toc536401829)

[1.3.3 Rulmenți 8](#_Toc536401830)

[1.3.4 Tije trapezoidale 8](#_Toc536401831)

[1.3.5 Cuplaj pentru motor 8](#_Toc536401832)

[1.3.6 Tije filetate, și piulițe 8](#_Toc536401833)

[1.3.7 Pat din plastic 8](#_Toc536401834)

[1.3.8 Unealtă multifuncțională pentru gravat 8](#_Toc536401835)

[2 Asamblare 9](#_Toc536401836)

[3 Limbajul GCODE 10](#_Toc536401837)

[3.1 Sintaxa limbajului GCODE 10](#_Toc536401838)

[3.2 Comenzi GCODE modale 10](#_Toc536401839)

[3.2.1 Grupul de comenzi de mișcare 11](#_Toc536401840)

[3.2.2 Grupul de comenzi pentru selectarea planului 13](#_Toc536401841)

[4 Algoritmi 14](#_Toc536401842)

[5 Implementare 15](#_Toc536401843)

[6 Utilizare 16](#_Toc536401844)

[7 Concluzii 17](#_Toc536401845)

# Resurse utilizate

In acest capitol voi discuta despre resursele hardware, software si mecanice folosite pentru realizarea proiectului.

## Software

In acest subcapitol sunt menționate atât resursele software folosite pentru aplicația dezvoltata pentru realizarea CNC-ului in Arduino, cât și pentru realizarea a unui program ce comunica prin portul USB cu Arduino.

Pe lângă asta mai sunt menționate si *tool*-uri pentru conversia imaginilor în fișiere GCODE.

### Aplicația CNC Arduino

Deoarece am ales sa folosesc platforma hardware Arduino, am folosit ca și limbaj de programare, limbajul C++, aceasta fiind singura opțiune disponibilă, însa după părerea mea si una dintre cele mai bune alegeri pentru programare *embedded*. Oferă toate facilitățile *low level* ale limbajului C, cat si posibilitatea de a programa la un nivel cat mai *high level* folosind concepte de programare orientata pe obiect.

Compilatorul folosit pentru a compila limbajul C++ în limbaj mașină pentru *microcontroler-*ulAtmega328p de pe Arduino este avr-g++.

Petru operațiile de input/output pe pini cât si pentru partea de comunicație cu portul serial, am folosit biblioteca oferită de Arduino si biblioteca AVR.

**<REF: arduino reference sau avr datasheet>**

### Aplicația de comunicație cu Arduino

Pentru aplicația de comunicație prin USB cu Arduino am ales sa folosesc limbajul Python 3.7, cu ajutorul pachetului PySerial.

**TODO: add more**

### *Tool*-uri pentru conversia imaginilor

Pentru conversia imaginilor in limbajul GCODE, am folosit aplicația *desktop* inkscape, împreună cu alte tool-uri online de conversie a fișierelor cu extensia SVG în GCODE.

**<REF: inkscape si jscut>**

## Hardware

În această secțiune voi enumera toate resursele hardware împreuna cu specificațiile lor, folosite la realizarea CNC-ului.

### Microcontroler

Așa cum am menționat și în introducere, placa de dezvoltare folosită pentru realizarea CNC-ului este Arduino Uno Rev3. Această placă folosește microcontroler-ul Atmega328p.

|  |  |
| --- | --- |
| Fig. – Arduino UNO | Specificații tehnice generale:  - frecventa microcontroler: **16Mhz**  - memorie flash: **32KB**  - memorie SRAM: **2KB**  - memorie EEPROM: **1KB**  - pini digitali I/O: **14**  - pini analogici: **6**  - tensiune alimentare placa: **7-20V**  - tensiune operare microcontroler: **5V** |

Am ales această placă deoarece este o platformă foarte populară. Din acest motiv există o comunitate destul de mare, iar documentația este bine pusă la punct.

Unul dintre cele mai importante motive pentru care am ales această placă este faptul că se poate monta peste ea un *CNC shield,* componenta hardware despre care voi vorbi in subcapitolul 1.2.2. Acest *shield*simplifica foarte mult conexiunile fizică dintre stepo-motoare si *driver*-ele pentru motoare, pe care voi le discuta în subcapitolul 1.2.4.

Un alt avantaj este dat de faptul ca interfațarea fizica dintre laptop si Arduino este făcută prin intermediul unui cablu USB de tip A către tip B. Un alt motiv pentru care am ales această platformă este faptul ca *uploadarea* software-ului pe microcontroler se face foarte ușor prin intermediul acestui USB fără a mai fi nevoie de un programator special pentru a face acest lucru.

Un dezavantaj în domeniul programării *embedded* este faptul ca se poate face *debug* doar cu hardware specializat, iar in ajutorul acestei probleme, platforma Arduino, vine cu trimiterea mesajelor de *debug* prin intermediului portului USB. Lucru ce rezolva parțial această problemă fără prea multe costuri în plus.

Toate aceste avantaje menționate mai sus legate de simplitatea folosirii plăcii Arduino prin intermediul USB-ului, sunt datorită faptului că această platformă are un *chip* de convertire a semnalului USB către interfața serial a microcontroler-ului.

**IMG: CH340G**

Iar nu in ultimul rând, legat de costuri, aceasta platforma împreuna cu celelalte componente hardware, *shield-ul* CNC si driverele pentru stepo-motoare, sunt o soluție relativ ieftina din punct de vedere financiar, iar cerințele hardware pentru implementarea CNC-ului sunt suficiente.

### CNC *shield*

In figura următoare este ilustrat *shield*-ulpentru CNC folosit în acest proiect.

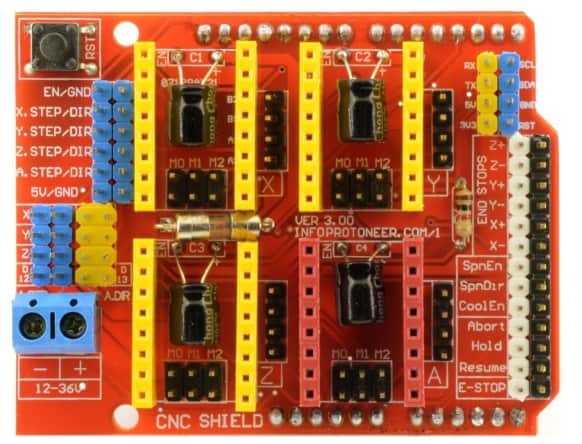


Fig. – CNC shield

După cum am menționat și în subcapitolul precedent unul dintre cele mai importante motive pentru care am ales platforma Arduino este faptul că acest *shield* se poate monta peste Arduino.

Pe acest *shield* se pot monta 3 drivere pentru motoare pas cu pas pentru fiecare dintre cele 3 axe X, Y si Z, respectiv 3 motoare pas cu pas. În plus se mai poate monta încă un driver plus încă un motor pentru a ajuta una dintre axe.

Pentru a detecta când axele s-au deplasat la limitele lor fizice, pe *shield* se pot monta 6 *endstop*-uri, câte 2 pe fiecare axă pentru a semnala sfârșitul axei, poziția zero a unei axe sau celalalt capăt.

Tensiunea de alimentare pentru acest *shield* este între 12 si 36 de volți. Acest curent electric de intrare este folosit doar pentru a alimenta motoarele pas cu pas. Placa Arduino este alimentată prin USB, nu prin curentul de alimentare al *shield-*ului. Acest lucru facilitează separarea curentului de intensitate și amperaj mai mare folosit de motoarele pas cu pas de curentul folosit de microcontroler (5V tensiunea maxima și 200mA).

Voi intra în mai multe detalii despre modul cum funcționează acest CNC *shield* împreună cu Arduino și driverele pentru motoarele pas cu pas în capitolul 2 de asamblare.

### Motoare pas cu pas

Motoarele pas cu pas sunt niște motoare electrice în care o rotație completă de 360 de grade se împarte la un număr egal de pași, de obicei 200 de pași.

Astfel, aceste motoare sunt de foarte mare precizie și sunt foarte des utilizate în proiecte precum CNC-uri, imprimante 3D, sau în robotică.

Pentru acest proiect am avut nevoie de 3 motoare, fiecare motor pune în mișcare câte o axa, X, Y, respectiv axa Z. Modelul pe care l-am ales este: **Motor pas cu pas 17HS8401S Nema17**.

|  |  |
| --- | --- |
| Fig. – Motor Pas cu Pas Nema17 | Specificații tehnice:  - pas: **1.8grade, adică 200 pași in 360 grade**  - cuplu: **0.59Nm**  - curent: **1.7 A**  - dimensiuni: **specifice standardului Nema17** |

Mișcarea acestor motoare va fi controlată de către *driverele* pentru motoarele pas cu pas despre care voi discuta în subcapitolul următor.

### Drivere pentru motoarele pas cu pas

Driverele pentru motoarele pas cu pas sunt componente special construite pentru a controla rotația precisă a unui motor. Cu ajutorul lor se poate ajusta limita maximă de curent trimisă către motor, direcția in care motorul se deplasează, precum si executarea pașilor.

În plus de asta, aceste drivere vin cu opțiunea de a face *microstepping*. Un motor pas cu pas face intr-o rotație completă de 360 grade exact 200 de pași egali. Cu ajutorul *microstepping-ului* controlează fazele curentului către motor astfel încât o rotație completă poate avea mai mult de 200 de pași. Astfel putem activa aceste opțiuni ajungând pana la 6400 de micro pași pe o rotație completă.

Cu alte cuvinte aceste drivere simplifică foarte mult interfațarea dintre motoare și microcontroler. Folosind astfel doar 2 pini de pe microcontroler, putem controla pașii și direcția unui astfel de motor pas cu pas.

Pentru alegerea *drivere*-lor existau mai multe posibilități, însa cele mai accesibile opțiuni au fost următoarele 2 *drivere*.

|  |  |
| --- | --- |
| Fig. – Driver A4988 | Fig. – Driver DRV8825 |
| Specificații tehnice:  Driver chip: **A4988**  Tensiune motor: **8V - 35V**  Tensiune chip: **3V - 5.5V**  Curent maxim: **2A**  Moduri microstepping: **1/2, 1/4, 1/8, 1/16** | Specificații tehnice:  Driver chip: **DRV8825**  Tensiune motor: **8V - 45V**  Tensiune chip: **3V - 5.5V**  Curent maxim: **2.5A**  Moduri microstepping: **1/2, 1/4, 1/8, 1/16, 1/32** |

Pentru proiect am folosit 2 *drivere* DRV8825 pentru motoarele de pe axele X si Y, și un *driver* A4988 pentru motorul de pe axa Z.

Nu este o diferență foarte mare între aceste 2 alegeri, una dintre diferențe ar fi că DRV8825 suportă *microstepping* mai mare. Cu un DRV8825 având opțiunea maxima de *microstepping* **1/32** activată atunci o rotație completă a unui motor pas cu pas ajunge la 6400 de micro pași, pe când la A4988 cu opțiunea maximă de *microstepping* **1/16** activată vom ajunge la 3200 de pași pe o rotație completă.

Din punct de vedere electric, ambele *drivere* suportă fără probleme curentul si voltajul necesar funcționării optime ale motoarelor pas cu pas alese.

### *Endstop*-uri pentru limitarea axelor

### Sursă de alimentare

## Mecanice

În acest ultim subcapitol sunt prezentate toate componentele mecanice folosite la realizarea proiectului, rolul lor, precum si specificațiile unor componente mai importante.

### Componente printate din plastic

- cele printate de pe site-ul mentionat in referinta si

- cele componente adaptate la nevoile mele, endstop, z axis, y axis modificat.

- adaptor pentru tool axa Z

### Tije cromate

### Rulmenți

- precizare figget spinner

### Tije trapezoidale

### Cuplaj pentru motor

### Tije filetate, și piulițe

### Pat din plastic

-ultima componenta montata

### Unealtă multifuncțională pentru gravat

# Asamblare

# Limbajul GCODE

Limbajul GCODE(Geometric CODE) este un limbaj de programare foarte popular folosit de imprimantele 3D si CNC-uri. Instrucțiunile limbajului GCODE indică unde și cum o astfel de mașina trebuie să se deplaseze pentru a desena, decupa, sau grava anumite forme in diferite materiale.

Ideea de ansamblu a acestui capitol este că toate instrucțiunile ce vor fi discutate mai jos, trebuiesc implementate în software-ul de pe Arduino. astfel încât CNC-ul să fie funcțional și să poată executa orice tip de fișier GCODE trimis linie cu linie prin USB către CNC.

## Sintaxa limbajului GCODE

O instrucțiune generică în GCODE are următoarea formă, unde **nn** reprezintăun număr, iar parantezele pătrate sugerează faptul că anumite simboluri pot lipsi:

**[Gnn] [Xnn] [Ynn] [Znn] [Fnn] [Mnn]**

* **G -** simbol folosit pentru a selecta o comandă de mișcare
* **X -** semnifică o poziție de deplasare pe axa X, în planul orizontal
* **Y -** semnifică o poziție de deplasare pe axa Y, în planul vertical
* **Z -** semnifică o poziție de deplasare pe axa Z, in adâncime
* **F -** setează viteza cu care se deplasează mașina, în milimetrii pe minut (mm/min)
* **M -** alte funcții diverse

Exemplu de comandă GCODE: **G01 X50.25 Y20.1** **F1000**

Această comanda executa o mișcare de interpolare liniara (**G01**) de către un nou punct având coordonata **X: 50.25** și coordonata **Y: 20.1**, cu o viteza de **1000** de milimetrii pe minut.

Pe lângă acestea, mai sunt și câteva simboluri specifice pentru comenzile **G02** și **G03** despre care voi vorbi în unul dintre următoarele capitole.

Limbajul GCODE nu este unul standard, pot exista anumite variații. De exemplu pentru abordarea mea am ales să folosesc simbolul **#** pentru a afișa diferite mesaje de diagnosticare, de a modifica anumiți parametrii interni, sau de a executa diferite rutine.

## Comenzi GCODE modale

Un aspect foarte important de menționat este faptul că anumite comenzi GCODE sunt modale, în sensul că unele comenzi le pot dezactiva pe cele precedente. Daca o comanda este dintr-un anumit grup, execuția unei comenzi din același grup dezactivează comanda precedenta.

In acest subcapitol voi menționa cele mai importante dintre aceste grupuri de comenzi modale.

### Grupul de comenzi de mișcare

In acest grup se află următoarele comenzi: G00, G01, G02 si G03.

Acesta este cel mai important grup de comenzi. Din acest grup fac parte toate cele 4 tipuri de mișcări pe care mașina le poate face:

* **G00** reprezintă o mișcare liniară a axelor într-un mod sincron. De exemplu dacă suntem la poziția P0(0, 0) și următoarea comandă urmează să fie executată: **G00 X40 Y20**

Atunci se executa mișcarea ca în figura de mai jos. Mai întâi se execută mișcarea primei axe până când noua poziție, în cazul de față **X: 40**, este atinsă. Apoi se executa mișcarea pe cea de a doua axă până când poziția **Y: 20** a fost atinsă.



Fig. – Comanda G00, mișcare liniară simplă

* **G01** reprezintă o mișcare liniară de interpolare. De exemplu dacă suntem la poziția P0(0, 0) și următoarea comandă urmează să fie executată: **G00 X40 Y20**

Atunci se execută o mișcare liniară pe dreapta formată dintre cele doua puncte, punctul P0 și punctul destinație P1(40, 20). Mișcarea obținută va fi echivalentă cu cea din imaginea următoare.



Fig. – Comanda G01, interpolare liniară

* **G02** si **G03** reprezintă mișcări de interpolare circulară. Cu ajutorul acestor comenzi se pot crea cercuri sau arcuri de diferite dimensiuni.

Singura diferența dintre G02 si G03 este că comanda G02 produce cercuri **în sensul acelor de ceasornic**, iar comanda G03 produce cercuri **invers acelor de ceasornic**.

Pentru aceste două tipuri de comenzi exista încă alte 3 simboluri esențiale:

* + **I** - distanța incrementală pe coordonata X relativă la centrul cercului
  + **J** - distanța incrementală pe coordonata Y relativă la centrul cercului
  + **R** - raza cercului

Pentru a desena cercul cerut, CNC-ul are nevoie să-și calculeze centru cercului. Exista doua metode prin care putem desena un cerc. Prima metodă este sa trimitem parametrii **I** si **J**, iar cea de-a doua metoda este sa trimitem doar raza cercului prin parametrul **R**, atunci ceilalți doi parametrii pot sa lipsească.

Pentru a desena un arc trebuie sa trimitem prin parametrii **X** si **Y** coordonatele celuilalt capăt al arcului. Atunci CNC-ul va trasa un arc începând de la punctul curent pana la punctul destinație. Punctul destinație trebuie neapărat sa se afle pe cerc. In caz contrar CNC-ul va răspunde cu un mesaj de eroare.

**<IMG>**

### Grupul de comenzi pentru selectarea planului

Aceste comenzi au impact doar asupra comenzilor de interpolare circulara **G02** si **G03**. Cu ajutorul lor putem alege planul în care dorim să executăm o comandă de interpolare circulară. Exista 3 comenzi pentru selectarea planului:

|  |  |
| --- | --- |
| * G17 – selectează planul **XY**, acesta este modul selectată in mod implicit * G18 – selectează planul **XZ** * G19 – selectează planul **YZ** | C:\Users\ciprian\AppData\Local\Microsoft\Windows\INetCache\Content.Word\G17_G18_G19.jpg |

### Grupul de comenzi pentru poziționare

Acest grup de comenzi conține două comenzi pentru selectarea modului de poziționare, referitor la cei trei parametrii, X, Y si Z.

Comanda **G90** reprezintă poziționarea absolută a parametrilor X, Y si Z relativ la poziția zero a axelor. De exemplu dacă suntem la poziția P0(10, 10, 10) și avem de executat următoarea comanda: **G01 X20 Y5 Z10** atunci CNC-ul se va poziționa la punctul P1(20, 5, 10).

Comanda **G91** reprezintă poziționarea incrementala a parametrilor X, Y si Z relativ la poziția curenta a axelor. De exemplu dacă suntem la poziția P0(10, 10, 10) și avem de executat următoarea comanda: **G01 X20 Y5 Z10** atunci CNC-ul se va poziționa la punctul P1(40, 15, 20).

### Grupul de comenzi pentru setarea unității de măsură

# Algoritmi

# Implementare

# Utilizare

# Concluzii