

3-osna mašina za graviranje bazirana na Arduino platformi

Posavec, Bruno

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Polytechnic of Međimurje in Čakovec / Međimursko veleučilište u Čakovcu**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:110:911834>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-02-05**



Repository / Repozitorij:

[Polytechnic of Međimurje in Čakovec Repository -
Polytechnic of Međimurje Undergraduate and
Graduate Theses Repository](#)





MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Bruno Posavec 0313027114

3-osna mašina za graviranje bazirana na Arduino platformi

Završni rad

Čakovec, srpanj 2024.



MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Bruno Posavec 0313027114

3-osna mašina za graviranje bazirana na Arduino platformi

3-axis engraving machine based on Arduino platform

Završni rad

Mentor: dipl. ing. Jurica Trstenjak

Čakovec, srpanj 2024.



MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

PRIJAVA TEME I OBRANE ZAVRŠNOG/DIPLOMSKOG RADA

Stručni prijediplomski studij:

Računarstvo

Održivi razvoj

Menadžment turizma i sporta

Stručni diplomski studij Menadžment turizma i sporta:

Pristupnik: Bruno Posavec, JMBAG: _____
(ime i prezime)

Kolegij: Digitalni elektronički sklopovi
(na kojem se piše rad)

Mentor: Jurica Trstenjak, v. pred.
(ime i prezime, zvanje)

Naslov rada: 3-osna mašina za graviranje bazirana na Arduino platformi

Naslov rada na engleskom jeziku: 3-axis engraving machine based on the Arduino platform

- Članovi povjerenstva: 1. Bruno Trstenjak, prof. struč. stud., predsjednik
(ime i prezime, zvanje)
2. Sanja Brekalo, prof. struč. stud., član
(ime i prezime, zvanje)
3. Jurica Trstenjak, v.pred., mentor
(ime i prezime, zvanje)
4. Marija Miščančuk, v.pred., zamjenski član
(ime i prezime, zvanje)

Broj zadatka: 2023-RAČ-9

Kratki opis zadatka: 3-osna mašina za graviranje bazirana na Arduino platformi će se sastajati od raznih rabljenih dijelova printera ili plotera, Arduino platforme, 3 koračna motora, napajanja te programskog dijela za Arduino.

Datum: 17.9.2024.

Potpis mentora: Jurica Trstenjak

Predgovor

Veliku zahvalnost iskazujem svojem mentoru, prof. Jurici Trstenjaku, dipl. ing., na iskazanom povjerenju, savjetima i vodstvu u izradi završnog rada.

Zahvaljujem svim djelatnicima Međimurskog veleučilišta u Čakovcu, svojoj obitelji i svima koji su bili uz mene tijekom cijelog obrazovanja.

Veliko hvala svima!

Sažetak

U završnom radu opisana je izrađena 3-osna mašina za graviranje bazirana na Arduinou čija je namjena graviranje raznih oblika u drvo ili plastiku. CNC stroj za graviranje sastoji se od mehaničkih i elektroničkih dijelova, uključujući okvir od materijala koji se koriste u namještaju, teleskopske vodilice za ladice, vijke i trapezna navojna vretena. Od elektroničkih dijelova koriste se sklopke, senzor za kalibraciju Z osi, tri koračna motora, jedan istosmjerni motor, pretvarač istosmjernog napona, napajanje, modul H-mosta, moduli upravljača, CNC *shield* proširenje za Arduino i Arduino UNO. Za sam rad stroja nužno je i računalo preko kojeg se šalje kôd za izvršavanje. Stroj ima 3 osi: X, Y i Z. Svaka je izrađena kao modul, baza je napravljena od šperploče ili medijapana, na njoj su vijcima pričvršćeni nosači koračnog motora, ležaja, po dvije vodilice te blok (klizač) koji putuje niz os. Na Z osi montiran je istosmjerni motor koji vrti alat. Softver stroja radi u tri dijela. Grafičko sučelje za upravljanje preko računala izrađeno u C# komunicira sa strojem. Softver na Arduinou izrađen je u C++/C koji upravlja strojem, a za rad se koristi G-kôd koji se napravi u CAM alatu kao što je Fusion 360.

Ključne riječi: *CNC, stroj, graviranje, Arduino, računalo*

Summary

This paper talks about the 3 axis machine for engraving based on Arduino. The main use of this machine is engraving various shapes in wood or plastic. The CNC engraving machine consists of mechanical and electronic parts including a frame made of materials used in furniture, telescopic guides for drawers ,screws and threaded screw rods . Electronic parts include switches, Z-axis calibration sensor, three stepper motors, one DC motor, boost converter, power supply, H-bridge module, stepper drivers, CNC shield and Arduino UNO. A computer is also necessary for the operation of machine itself as it's used for sending code for execution. The machine has 3 axes X, Y, Z. Each one is made as a module. The base is made out of plywood or mediapan and on it are bolted stepper motor mount, bearings with mounts and two guides that have a block(slider) which travels down the axis mounted on them. A DC motor is mounted on the Zaxis, which rotates the tool. The machine software works in three parts. A graphical interface for computer was made in C# which communicates with machine. Software on the Arduino is made in C++/C that controls the operation of machine according to G-côde made in CAM tool such as Fusion 360.

Key words: *CNC, machine, engraving, Arduino, computer*

Popis korištenih kratica

CNC - (eng. Computer Numerical Control)

UART – (eng. Universal asynchronous reciver/transmitter)

RAM – (eng. Random-access memory)

NEMA – (eng. National Electrical Manufacturers Asociacion)

PWM – (eng. Pulse With Modulation)

USB - (eng. Universal Serial Bus)

CAD- (eng. Computer aided Design)

CAM – (eng. Computer adied Manufacturing)

CAE - (eng. Computer aided Engineering)

PCB - (eng. Printed Circuit Bord)

GRBL – (eng. G-code Reference Block Library)

Sadržaj

1. Uvod	2
2. MATERIJALI	3
2.1. Konstrukcija	3
2.2. MEHANIČKI DIJELOVI	4
2.2.1. Ležajevi	4
2.2.2. Trapezno navojno vreteno	5
2.2.3. Stalak za koračni motor	6
2.2.4. Blok za maticu	6
Izvor: autor	6
2.2.5. Stalak za ležaj	7
2.2.6. Steznik za istosmjerni motor	8
2.2.7. Teleskopske vodilice	9
2.2.8. Prihvat za motor 5 na 8 mm	10
2.2.9. Ostali dijelovi	10
2.3. Arduino	11
2.4. CNC modul	12
2.5. Koračni motor	13
2.6. Modul A4988	15
2.7. Istosmjerni motor s četkicama	17
2.8. Modul BTS7960	18
2.9. Prekidači	20
2.10. CNC <i>touch</i> senzor	21
2.10. Napajanje	22
2.11. Pretvarač istosmjernog napona	24
3. SHEMA SPOJA PROJEKTA	25
4. SOFTVER	26
3.1. Arduino IDE	26
3.2. Biblioteke	26
3.3. G-KÔD	27

3.4 Arduino CNC V0_0_8U_D.....	28
3.5. C# Windows Forms .NET	37
3.6. Arduino CNCpcontroller.....	38
3.7. Fusion 360	39
4. IZRADA G-KÔDA U FUSIONU 360	40
5. TESTIRANJE RADA	44
6. ZAKLJUČAK	46
7. LITERATURA	47
8. POPIS SLIKA	54

1. Uvod

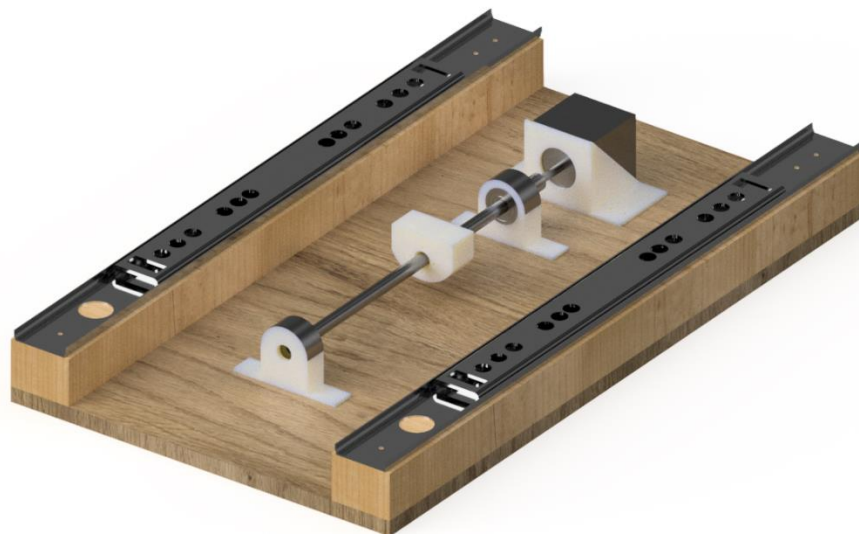
Ideja za ovakav projekt proteže se već duži vremenski period, najviše iz želje da bi se njime mogle izraditi tiskane pločice koje bi služile za izradu strujnih krugova. Cilj projekta bio je izraditi 3-osnu mašinu za graviranje baziranu na Arduino platformi. Graviranje je postupak kod kojeg se u površinu materijala urezuju znakovi ili ukrasi. CNC je upravljanje alatom pomoću računala. Arduino UNO će upravljati modulima, a računalo će putem upravljačkog programa slati naredbe na Arduino. Stroj je izrađen od dijelova koji su bili lako dostupni. Sastoji se od 3 osi, a svaka os sadrži koračni motor koji ostvaruje pokrete, zatim navojnog vretena koje pretvara kružno gibanje motora u pravocrtno te vodilica koje usmjeravaju pokrete. Na Z osi nalazi se istosmjerni motor s glodalom. Arduino upravlja koračnim motorima putem A4988 modula, a istosmjernim motorom pomoću BTS7960 modula. Za napajanje se koristi 12 V prekidačko napajanje, a istosmjerni motor napaja pretvarač istosmjernog napona koji povisuje napon na 24 V kako bi mogao postići punu brzinu od 20 000 okretaja u minuti. *Fusion 360* će se koristiti za generiranje G-kôda koji se putem C# programa šalje preko UBS-a na Arduino, koji zatim izvodi operacije prema kôdu. Arduino na sebi ima Arduino CNC softver koji podržava osnovne G-kôd naredbe kako bi upravljao strojem, izrađen u Arduino IDE koji je primarno u C++. Za naprednije upravljanje konačnim motorima koristi se biblioteka *AccelStepper* koja daje mogućnost naprednog upravljanja brzinom te akceleracijom konačnih motora uz objektno orijentirano sučelje.

2. MATERIJALI

2.1. Konstrukcija

Konstrukcija stroja je izrađena od raznih dostupnih materijala iverice, šperploče, medijapana, jele i bukve. Baza stroja je iverica. Na bazi stroja nalazi se Y os i konstrukcija za X osi. Baza Y os izrađena je od medijapana, a ostale od šperploče. Bočne stranice na bazama izrađene su od jele i vijcima su pričvršćene za dno. Na njima su montirane teleskopske vodilice za ladice. U središnji dio montirana je os s pogonom kojeg čini koračni motor s nosačem, dva nosača trapeznog vretena. Kroz njih prolazi trapezno navojno vreteno koje na sebi ima maticu. Matica je pričvršćena u pokretni blok. Pokretni blok odnosno klizač je pričvršćen za pokretni dio vodilice kako bi se kretao po osi zajedno s radnim stolom. Sam radni stol na Y osi napravljen je od iverice i sadrži niz utora s navojnim maticama kako bi se materijali za obradu mogli pričvrstiti. X os je montirana iznad Y osi na konstrukciju za X os, a orijentirana je tako da ostvari okomiti pokret u odnosu na Y os. Na klizaču X osi montirana je Z os koja je okomita na X os. Na klizaču Z osi montiran je prihvat za istosmjerni motor koji se sastoji od spojne pločice na kojoj su zavarene dvije spojnice za stezanje glavnog motora.

Slika 1. 3D model-Modul(X,Y,Z) osi



Izvor: autor

2.2. MEHANIČKI DIJELOVI

2.2.1. Ležajevi

U projektu su korišteni ležajevi promjera 22 mm sa središnjim otvorom od 8 mm i debljinom od 7 mm. Njihova uloga je držati trapezno navojno vreteno u centru s pogonskim motorom te omogućiti okretanje osi.

Slika 2. Ležajevi



Izvor: autor

2.2.2. Trapezno navojno vreteno

Trapezno navojno vreteno sastoji se od šipke s trapeznim navojem i matice. Okretanjem vretena matica putuje te se time kružno gibanje pretvara u pravocrtno. Nema elemenata između kontakta. Jeftinija su od kugličnih, ali manje učinkovita zbog trenja i praznog hoda. Na X i Y osi nalaze se vretena promjera 8 mm s 8 mm hodom dužine 350 mm, dok je na Z osi 8 mm s 2 mm hoda i 200 mm dužine. Vretena promjera 8 mm najčešće se mogu pronaći na 3D printerima.

Slika 3. Trapezno navojno vreteno



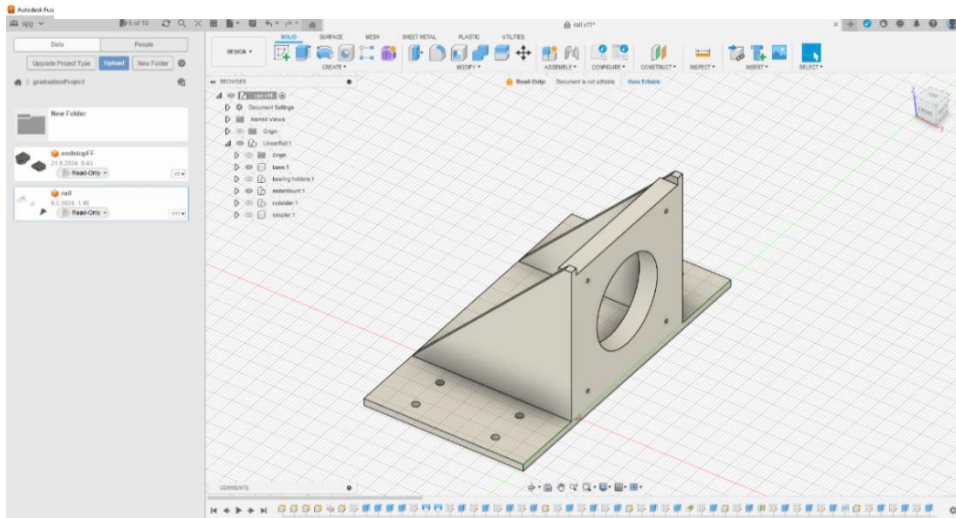
Izvor:

https://vi.aliexpress.com/item/4001332632943.html?spm=a2g0o.order_list.order_list_main.110.74201802CuD8sO&gatewayAdapt=glo2vnm. (pristup 14. 8.2024.)

2.2.3. Stalak za koračni motor

Stalak za koračni motor je modeliran i izrađen pomoću 3D printera uz korištenje PLA plastike.

Slika 4. Model stalaka za koračni motor

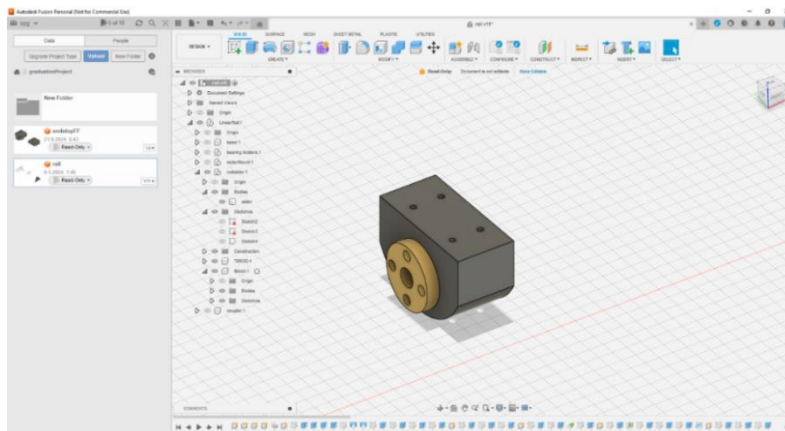


Izvor: autor

2.2.4. Blok za maticu

Blok za maticu je 3D printani dio koji pričvršćuje maticu navojnog vretena za klizač koji putuje po osi.

Slika 5. Model bloka za maticu.

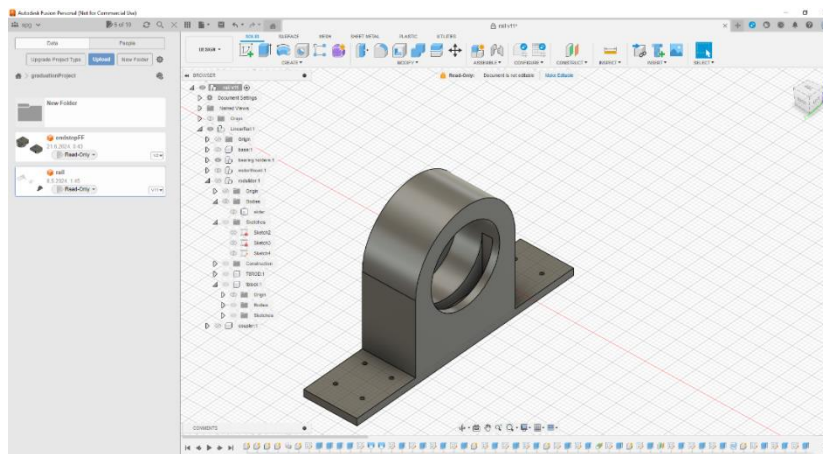


Izvor: autor

2.2.5. Stalak za ležaj

3D printani dijelovi koji drže ležajeve na osi kako bi bili u ravni s motorom.

Slika 6. 3D model stalaka za ležaj



Izvor: autor

2.2.6. Steznik za istosmjerni motor

Stežnik za motor sastoji se od dvije spojnice koje su zavarene na pločicu i služe za pričvršćivanje istosmjernog motora.

Slika 7. Spojnica i pločica



Izvor: autor

2.2.7. Teleskopske vodilice

Vodilice su ključan dio mašine jer određuju smjer kretanja i nose osi ili podlogu. Postoji niz vrsta vodilica, no često se dijele u dvije skupine: one s valjnim tijelima i bez valjnih tijela. One s valjnim tijelima smanjuju trenje i time osiguravaju lakši pomak. Linearne vodilice sastoje se od tračnice i bloka u kojem se nalazi niz valjnih tijela. Teleskopske vodilice za ladice i stol sastoje se od tračnice i kliznog dijela koji u zidovima sadrži loptice. Znatno su jeftinije i pristupačnije od linearnih i zbog toga su korištene u ovom projektu.

Slika 8. Teleskopske vodilice



Izvor: autor

2.2.8. Prihvat za motor 5 na 8 mm

Potrebno je pričvrstiti trapezno vreteno za motor te se za tu svrhu koristi prihvat koji ima dva utora, jedan od 5 mm, koji odgovara promjeru šipke na motoru, i drugi od 8 mm koji odgovara trapeznom vretenu.

Slika 9. Prihvat za motor



Izvor: autor

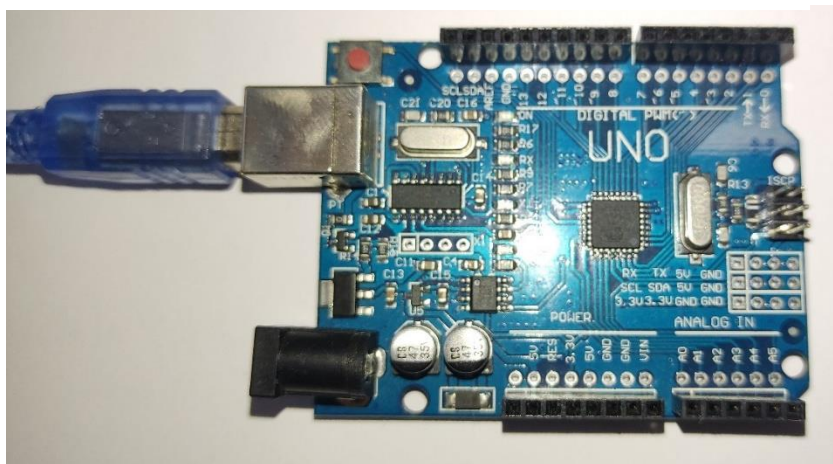
2.2.9. Ostali dijelovi

Vijci, matice, trio matice, spojne pločice, kutnici, akrilno staklo.

2.3. Arduino

Arduino pločica je razvojna platforma bazirana na Atmelovom mikroupravljaču, a cilj Arduina je omogućiti svima lagani ulaz u programiranje mikroupravljača. Mikroupravljač u sebi sadrži *bootloader* koji omogućuje učitavanje programa putem USB-a. Komunikacija se izvodi preko sučelja koje prevodi USB u UART¹ i zbog toga pločica ne treba zasebni programer za programiranje. Arduino UNO je baziran na Atmega328P-PU koji je 8-bitni mikroupravljač s 32 KB *flash* memorije i 2 KB RAM memorije. U projektu Arduino vrši upravljanje dijelovima stroja prema uputama zadanim u obliku G-kôda. Programiranje se vrši putem Arduino IDE. Arduino UNO ima 12 digitalnih ulaza i izlaza od kojih 6 ima mogućnost PWM izlaza, a 6 pinova sadrži 8-bitni ADC.

Slika 10. Arduino UNO



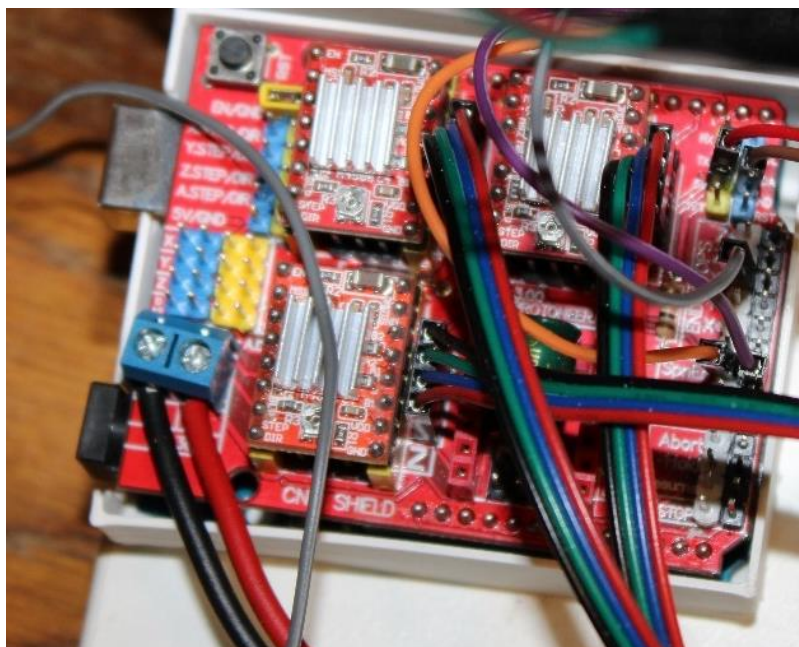
Izvor: autor

¹ UART - Universal Asynchronous Receiver/Transmitter, protokol za komunikaciju između dvaju uređaja. Koristi dvije linije za zaseban smjer, umjesto *clock* signala koristi se *baud rate*. Uređaji prethodno moraju imati postavljenu brzinu slanja podataka.

2.4. CNC modul

CNC modul je proširenje za Arduino UNO koje se priključuje na Arduino pločicu. Dizajniran je tako da je kompatibilan s GRBL-om, softverom otvorenoga kôda koji razumije G-kôd i upravlja hardverom. Na sebi ima izvedene priključke za module upravljača koračnih motora kao što su A4988 ili DRV8825 te ulaze kao što su granični prekidači.

Slika 11. Arduino UNO s CNC modulom te 3 upravljača



Izvor: autor

2.5. Koračni motor

Koračni motor je vrsta istosmjernog motora koji se u odnosu na klasični vrti korak po korak i time osigurava precizno kretanje. Ovisno o izradi postoji nekoliko vrsta: sa stalnim magnetima, varijabilnom reluktancijom i hibridni. Također mogu imati različiti broj para zavojnica, odnosno faza, pri čemu najčešće korišteni imaju dvije ili četiri. Postoji u više dostupnih veličina, gdje su veći motori snažniji. NEMA² standard kod koračnih motora govori o veličini prednje strane motora dok ostale karakteristike ovise o modelu. Različiti koračni motori mogu imati različiti broj koraka po revoluciji. U projektu je korišten NEMA 17 HS4401 koji je bipolarni hibridni koračni motor koji ima 200 koraka po okretaju što daje pomak od 1,8 stupnjeva. Hibridni koračni motor sastoji se od dva zupčanika od feromagnetnog materijala između kojih se nalazi magnet. Na taj način jedan zupčanik je južni pol, a drugi sjeverni. Zupčanici su zamaknuti tako da je zubac drugog između prvog. Na statoru se nalaze zavojnice koje tvore elektromagnet s feromagnetnom jezgrom i na njihovim su vrhovima oblikovani zupci koji su poravnati sa zupcima na rotoru. Pri uključivanju faze zavojnica s obadje strane nalaze se dva različita pola u blizini jednog zupca zavojnice. S obzirom na to da se suprotni polovi privlače, a istoimeni odbijaju, suprotan će se rotirati prema zavojnici. Zatim se uključuje drugi niz zavojnica kod kojih se ponovo nalaze dva različita pola. Polaritet polova je također različit s obje strane i samim time će se, ovisno o smjeru struje kroz zavojnice, odrediti smjer okretaja. Maksimalna struja mu je 1.5 A i daje 40 Ncm okretnog momenta.

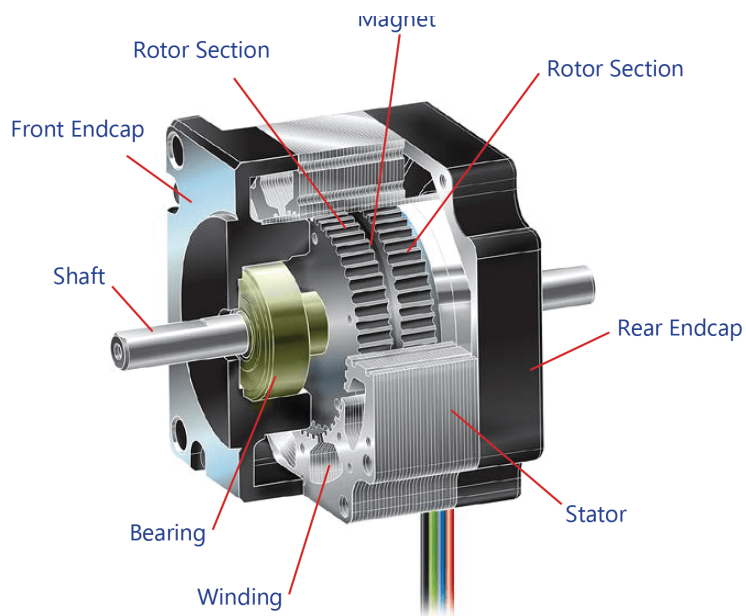
² NEMA (National Electrical Manufacturers Association) – definira standarde za razne uređaje, a u slučaju steppera govori o veličini prednje strane motora

Slika 12. Koračni motor NEMA 17 hs4401



Izvor: autor

Slika 13. Koračni motor iznutra



Izvor: <https://www.linengineering.com/technology/hybrid-stepper-motors>

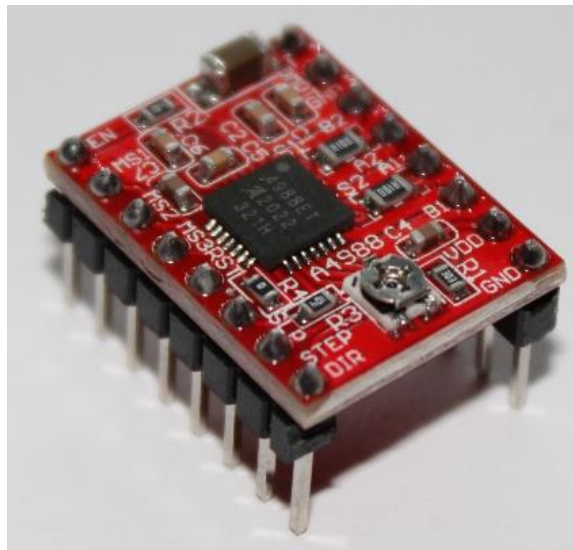
(pristup : 17.9.2024)

2.6. Modul A4988

A4988 je upravljač bipolarnih koračnih motora. Unutar sebe sadrži pretvarač koji, ovisno o stanju na ulazima, upravlja kontrolerom i pretvaračima. Moguće je odabrati od punog koraka do jedne šesnaestine koraka što se radi pomoću četiri ulaza, a s tri ulaza se upravlja smjerom, rotacijom i omogućavanjem okretanja motora. Također na sebi ima i ograničavanje izlazne struje pomoću potenciometra (na modulu), a njime se postavlja U_{ref} , dok se praćenje provodi na dva vanjska otpornika R_s za svaku fazu. Maksimalna izlazna struja se računa prema formuli 2.1. R_s ovisi o proizvođaču, najčešće iznosi 0.1Ω .

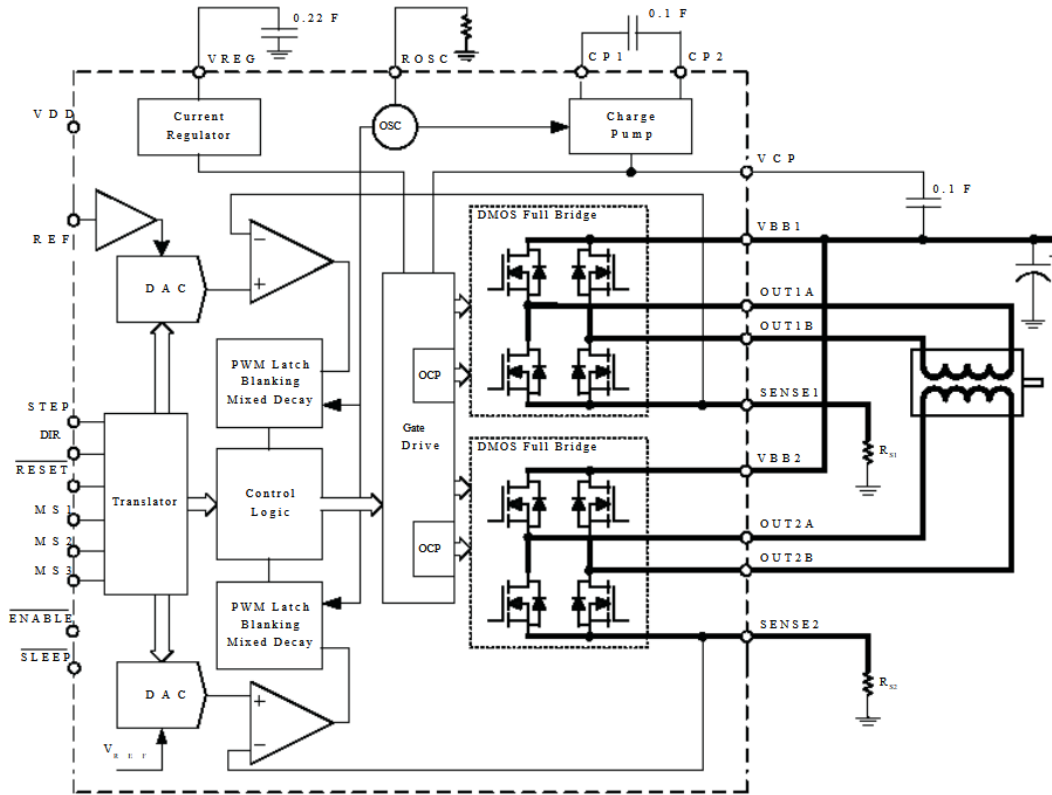
$$I = \frac{U_{ref}}{8 * R_s} \quad (2.1)$$

Slika 14. Modul upravljača koračnih motora A4988



Izvor: autor

Slika 15. Blok-shema upravljača koračnog motora

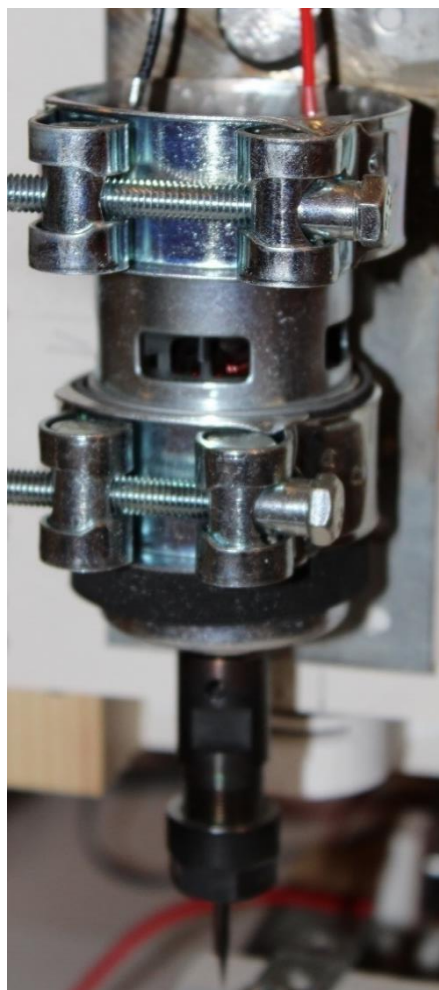


Izvor: <https://www.alldatasheet.com/html-pdf/338780/ALLEGRO/A4988/888/3/A4988.html> (pristup 17.9.2024)

2.7. Istosmjerni motor s četkicama

Postoji više vrsta istosmjernih motora: sa stalnim magnetima, elektromagnetima, s četkicama i bez četkica. Istosmjerni motor s četkicama i stalnim magnetima sastoji se od statora na kojem se nalaze magneti i rotora na kojem su zavojnice. Za prijenos struje na zavojnice kontakt se ostvaruje pomoću četkica i komutatora. Radi na temelju Lorentzove sile koja nastaje uslijed gibanja elektrona u magnetskom polju. Smjer sile određuje se pravilom lijeve ruke. U projektu se koristi za vrtnju glodala i drugih alata. Nedostatak istosmjernog motora s četkicama je kraći životni vijek jer se četkice uslijed vrtnje troše. Prednost mu je da nema potrebu za dodatnom upravljačkom jedinicom. U projektu je korišten motor koji na sebi ima steznu glavu za alat promjera do 3.15 mm.

Slika 16. Istosmjerni motor s četkicama.

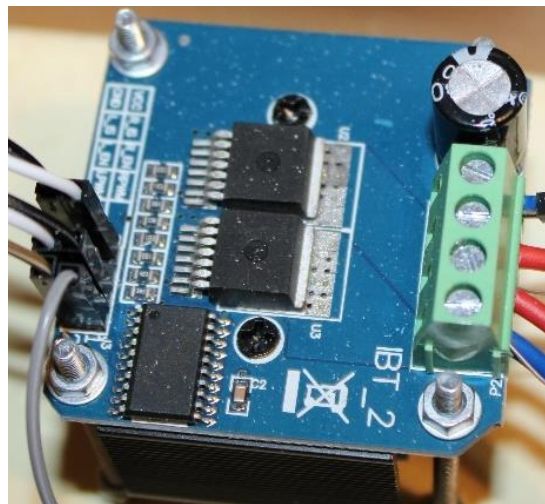


Izvor: autor

2.8. Modul BTS7960

Arduino sam ne može pokretati velika trošila, odnosno veće struje ni napone iznad 5 V, pa se za upravljanje istosmjernim motorom koristi BTS7960 modul. To je modul „H-most“ (engl. H-bridge) za velike struje. Na modulu se nalaze dva integrirana polovična H-mosta. Maksimalna struja mu je 43 A, a napon od 6 do 27 V. Za spajanje na mikroupravljač ima dva izlaza za mjerenje struje i četiri za kontrolu, od kojih su dva za omogućavanje upravljanja. H-most je elektronički sklop pomoću kojeg se može upravljati smjerom vrtnje istosmjernog motora, a sastoji se od četiri sklopke tako da dvije spajaju svaku stranu motora s oba pola izvora. Uključivanjem suprotnih sklopki omogućuje se prolaz struje kroz motor u jednom smjeru, a njihovim gašenjem te paljenjem drugih motor će se vrtjeti u drugom smjeru, dok se brzina istosmjernog motora može upravljati pomoću PWM-a³. U BTS7960 ulogu sklopki preuzimaju MOSFETI. U projektu se koristi samo jedan ulaz jer će se motor vrtjeti samo u jednom smjeru te je dodana obrnuto polarizirana dioda za zaštitu u slučaju EMF-a iz motora.

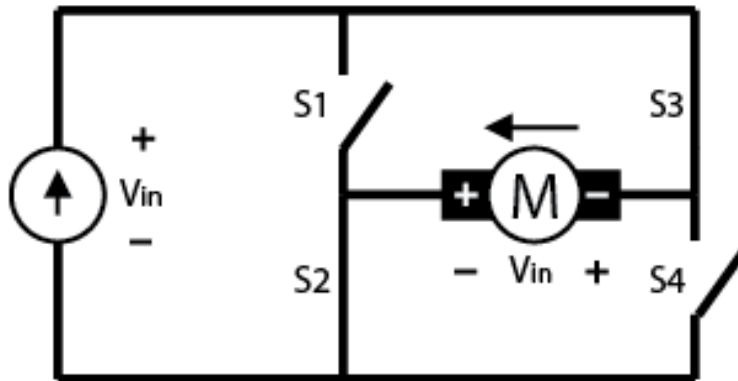
Slika 17. BTS7960 modul



Izvor : autor

³ PWM (engl. *Pulse With Modulation*) – tehnika kod koje se pomoću digitalnog signala simulira analogni variranjem vremena trajanja napona visoko/nisko.

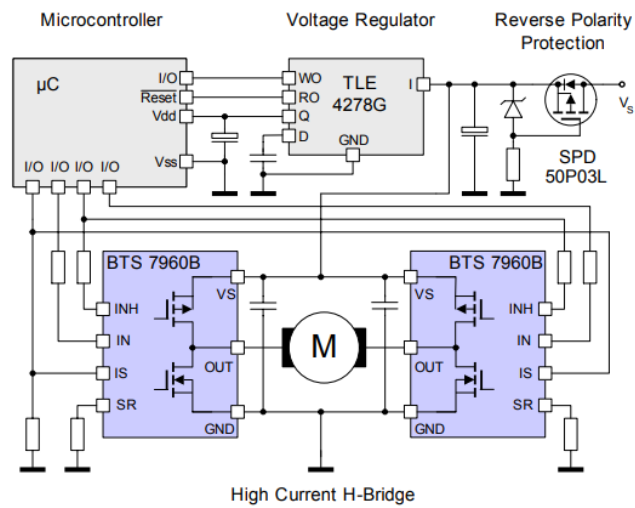
Slika 18. H-most



Izvor: <https://digilent.com/blog/what-is-an-h-bridge/>

(pristup: 17.9.2024)

Slika 19. Shema BTS7960 modula



Izvor:

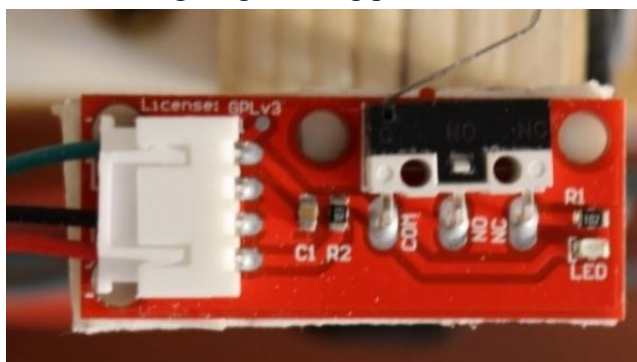
<https://www.handsontec.com/dataspecs/module/BTS7960%20Motor%20Driver.pdf>

(pristup: 17.9.2024)

2.9. Prekidači

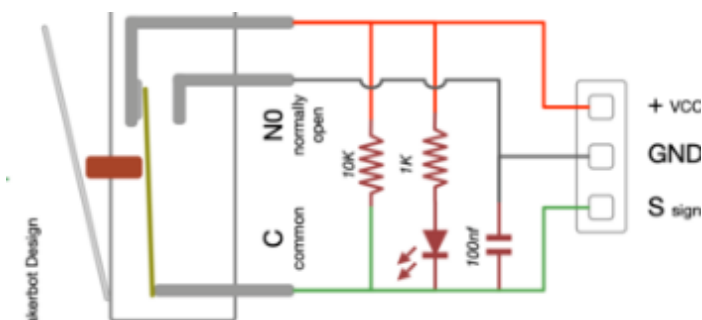
Prekidači služe za uključivanje i isključivanje strujnog kruga. U projektu se koriste na krajevima osi kako bi Arduino dobio informacije da je došao do kraja jedne osi. Na sami uređaj montirana je „gljiva“ odnosno sigurnosna sklopka, koja pritiskom na tu „gljivu“ isključuje cijeli uređaj. Za granične prekidače korišteni su moduli koji u sebi imaju kondenzatore, otpornike i LED diodu. Kondenzator služi za smanjenje smetnji, a otpornici jedan za podizanje napona signalne žice na logičku jedinicu, dok drugi služi za ograničavanje struje kroz LED diodu. S obzirom na to da su moduli prekidača spojeni na isti ulaz, nije moguće koristiti 5 V jer dovodi do velike struje kroz Arduino te su korišteni samo signalni izlaz i zemlja. Za spajanje na isti ulaz korištena je eksperimentalna pločica i na nju je montiran otpornik $10\text{ K}\Omega$ koji služi za podizanje napona na logičkih 5 V.

Slika 20. Izgled graničnog prekidača



Izvor: autor

Slika 21. Shema graničnog prekidača



Izvor: https://reprap.org/wiki/Mechanical_Endstop. (pristup 13. 8. 2024.)

Slika 22. Gljiva tipkalo



Izvor: autor

2.10. CNC touch senzor

Za preciznije namještanje alata koristi se *CNC touch probe* koja radi tako da se ostvari kontakt preko metala na motoru, odnosno alata, i dobro izolirane metalne površine. Pritiskom na Z-cal gumb poziva se Zcal funkcija. Ona pomiče Z os sve dok ne ostvari kontakt sa senzorom, koji je spojen zajedno s graničnim prekidačima te funkcionira na isti način. Nakon ostvarenog kontakta glava stroja vraća se 5 mm iznad materijala kako bi se senzor mogao maknuti. Zatim se pozicija stroja postavlja na 24.5 mm visine te ostale osi na 0 mm.

Slika 23. CNC touch senzor

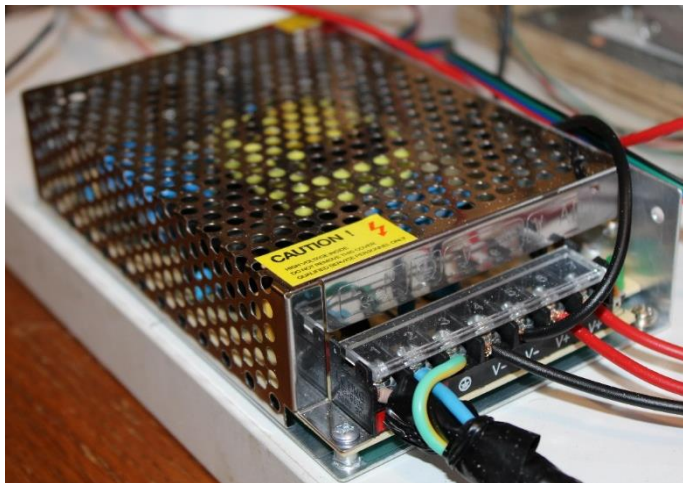


Izvor: autor

2.10. Napajanje

Za komponente je potreban izvor istosmjerne struje niskog napona, a za tu svrhu koristi se prekidačko napajanje. Prekidačko napajanje sastoji se od filtera, ispravljača koji pretvara izmjeničnu u istosmjernu, prekidača, transformatora i kruga za upravljanje. Izlaz je izoliran u odnosu na ulaz preko transformatora i optokaplera⁴. Na izlazu se prati napon te po potrebi smanjuje ili povećava širina PWM kojim se prekida ulaz. Izlaz se dodatno filtrira kako bi se dobio što stabilniji napon. Glavna prednost prekidačkog napajanja je visoka učinkovitost, a većom frekvencijom smanjuje se veličina potrebnog transformatora. Paljenjem i gašenjem odbacuje se potreba za disipacijom nekoristene snage. Jedini nedostatak je što paljenjem i gašenjem ostaju manje fluktuacije u naponu koje nije moguće potpuno otkloniti. U projektu je korišteno 12 V 10 A napajanje.

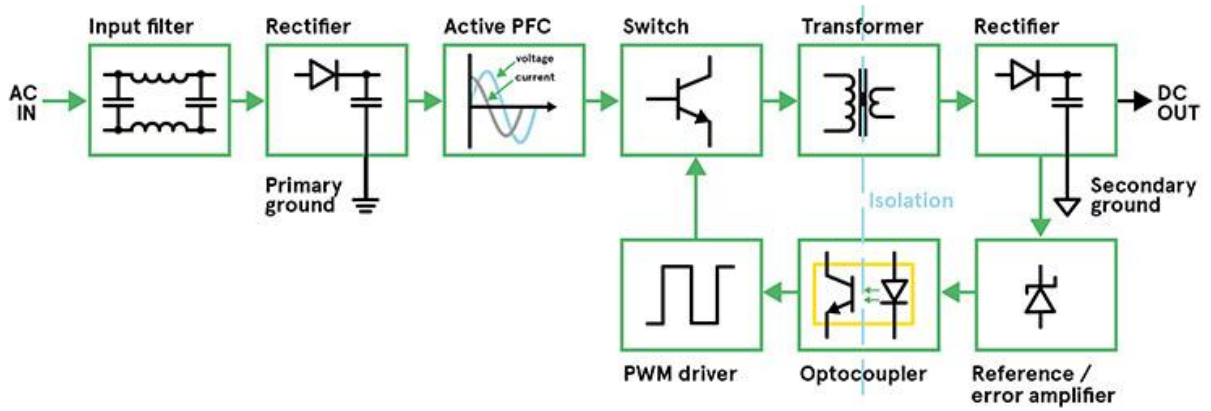
Slika 24. Prekidačko napajanje



Izvor: autor

⁴ Optokapler – komponenta koja pomoću svjetla prenosi upravljački signal iz jednog strujnog kruga u drugi

Slika 25. Blok-shema prekidačkog napajanja

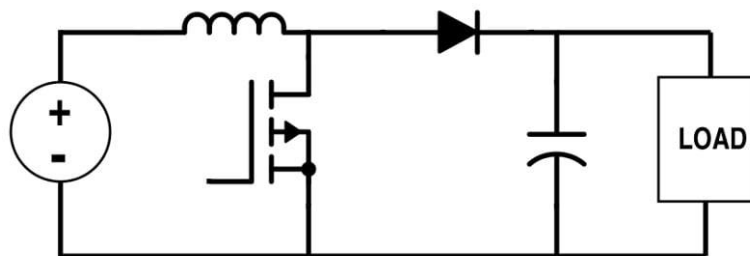


Izvor: <https://elektronikaupraksi.com/napajanje/prekidacko-napajanje-switch-mode-power-supply-smps/> (pristup 17.9.2024)

2.11. Pretvarač istosmjernog napona

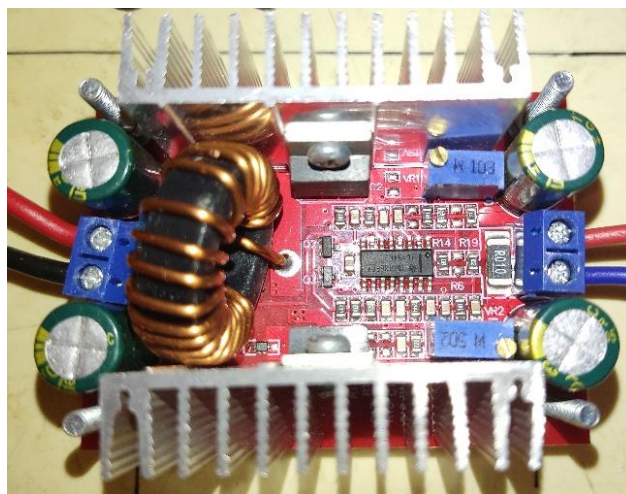
Budući da se u projektu koristi 12 V napajanje, a istosmjerni motor zahtijeva 24 V za maksimalnu brzinu vrtnje, da bi se izbjegla potreba za 2 napajanja, koristi se pretvarač istosmjernog napona koji povisuje napon na 24 V. Pretvarač je oblik prekidačkog napajanja koji radi na bazi Faraday-Lenzovog zakona. Osnovni pretvarač sastoji se od: zavojnice, sklopke, diode i kondenzatora. Paljenjem sklopke struja prolazi kroz zavojnicu uslijed čega nastaje magnetsko polje. Gašenjem sklopke energija u obliku magnetskog polja pohranjena u zavojnici ispušta se kroz diodu u kondenzator, dok dioda onemogućuje pražnjenje kondenzatora kroz sklopku. Na samu sklopku može se dodati još jedan sklop koji daje PWM kako bi se regulirao i održao stabilan napon. U projektu je korišten 400 W 15 A 10-60 V modul koji ima dva potencijometra s kojima se može regulirati izlazni napon i struja.

Slika 26. Shema osnovnog pretvarača napona.



Izvor: <https://www.allaboutcircuits.com/technical-articles/understanding-the-operation-of-a-boost-converter/> (pristup: 13.8.204)

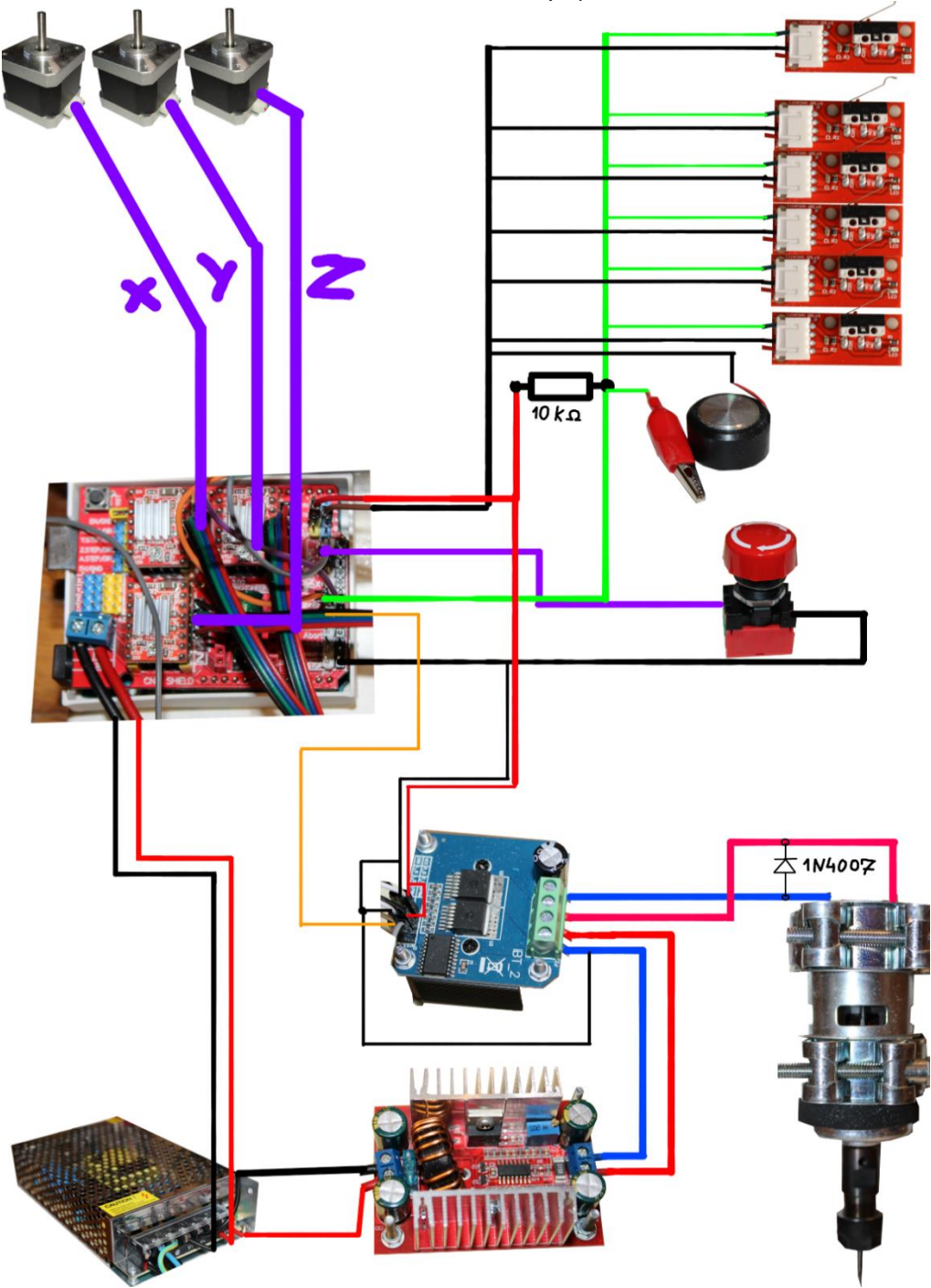
Slika 27. Pretvarač napona



Izvor: autor

3. SHEMA SPOJA PROJEKTA

Slika 28. Shema spoja



Izvor: autor

4. SOFTVER

3.1. Arduino IDE

Arduino IDE je razvojno okruženje za programiranje Arduino pločica i drugih kompatibilnih razvojnih pločica. U novijoj verziji 2.0 promijenjeno je sučelje i dodane su nove mogućnosti kao što je osnovno automatsko popunjavanje. U projektu je korišten za izradu softvera koji upravlja strojem. Programiranje se primarno vrši u C++-u, no može se i u C-u. U samom programu moguće je preuzeti i instalirati biblioteke, odabrati pločicu priključenu na određen USB ulaz, slati i primiti podatke putem USB-a. Arduino ima već izveden niz funkcija koje olakšavaju samo programiranje mikroupravljača te nije nužno poznavati hardver na izrazito niskoj razini.

3.2. Biblioteke

U projektu je korištena biblioteka AccelStepper koja omogućava naprednu kontrolu koračnih motora na bazi objektno orijentiranog pristupa. Napravio ju je Mike McCauley, a održava ju Patrick Wasp .

AccelStepper ,Arduino , Dostupno na:

<https://www.arduino.cc/reference/en/libraries/accelstepper/> (Pristup 17.9.2024)

Može se upravljati s više motora istovremeno i podržava jako spore brzine. Također daje mogućnost akceleracije i deceleracije. Korištenje same biblioteke je jednostavno, započinje se izradom objekata motora. Na samom objektu postavljaju se parametri te se zatim pokreće motor pomoću *run* metode koja se mora uzastopno pozivati sve dok udaljenost stavljena na motoru ne dođe na 0.

3.3. G-KÔD

G-kôd je jezik za upravljanje CNC strojevima i 3D printerima. Djelomično je standardiziran jer neke mogućnosti mogu ovisiti o proizvođaču strojeva, no postoji određeni niz koji je isti za sve (G-code Tutor: CNC G Codes , <https://gcodetutor.com/cnc-machine-training/cnc-g-codes.html>) . Neke od njih su: G00, G01, G02, G03, G04, G20, G21, G90, G91, G40, G41, G42, G70-G76, G80-G86, G54-G59.

G00 pravocrtno gibanje maksimalnom brzinom

G01 pravocrtno gibanje

G02 kružno gibanje u smjeru kazaljke na satu

G03 kružno gibanje suprotno smjeru kazaljke na satu

G04 pauziranje

G20, G21 mjerne jedinice u mm ili inčevima

G90 parametri su zadani apsolutno, u odnosu na početnu poziciju stroja

G91 parametri su zadani relativno, u odnosu na trenutnu poziciju

G40 prekidanje kompenzacije alata

G41, G42 lijeva, desna kompenzacija alata

G70, G72, G74, G76 ciklusi za otklanjanje velikih količina materijala

G81, G82, G83, G84, G85, G86 ciklusi za bušenje rupa

G10, G54-G59 postavljanje točaka na materijalu, odabir točaka na materijalu.

3.4 Arduino CNC V0_0_8U_D

Arduino CNC je naziv softvera razvijenog u izradi projekta. Njegova funkcija je preuzimanje komandi s USB-a te interpretacija G-kôda i upravljanje strojem. Podržava osnovne G-kôd naredbe G0, G1, G2, G3, G17, G18, G19, G20, G21, G90, G91, M3, M4 i M30. Za početak Arduino očekuje samo USB naredbe koje se šalju kroz funkciju *doUsbCommands*, a primitkom \$BEGIN# započinje se zaprimanje G-kôda.

Slika 29. doUsbCommands funkcija

```
bool doUsbCommands(String cmd) {
  // Return false if the command is empty or c
  if (cmd.length() == 0) {
    return false;
  }

  if (cmd == startUsbCommunication) {
    SendReply(usbCommBeginReply);
    if (!usbCommRunning) {
      usbRun();
    }
    return true;
  }

  if (usbCommRunning || running) {
    if (cmd == usbCommEnd) {
      usbCommRunning = false;
      return true;
    }
    if (cmd == startGcodeStream) {
      SendReply(okA);
      if (!running) {
        receiveGcode();
      }
      return true;
    }
    if (cmd == sendSingleGCode) {
      receiveSingleGcode();
      return true;
    }
    if (cmd == beginHomeSequence) {
      SendReply(okA);
      home();
      SendReply(sequenceDone);
      return true;
    }
    if (cmd == beginZcalSequence) {
      SendReply(okA);
      zCal();
      SendReply(sequenceDone);
      return true;
    }
    if (cmd == disableES) {
      SendReply(okA);
      disableEndstops();
    }
  }
}
```

Izvor: autor

Slika 30. Zaprimanje G-kod naredbi

```
bool usbCommRunning = false;
> void usbRun() { ...
}

String usbLine = "";
bool running = false;
bool done = false;
void receiveGcode() {
  running = true;
  while (running) {
    checkStopBtn();
    if (checkEndstops()) {
      SendReply(stopGcodeStream);
      break;
    }

    usbLine = readUsbLine();
    usbLine.trim();

    if (usbLine.length() > 0) {
      if (!doUsbCommands(usbLine)) {
        readCommandLine(usbLine);
        delay(100);
      }
      if (!done) {
        SendReply(gcodeStreamSendNext);
      }
    } else {
      delay(100);
      if (!done) {
        SendReply(gcodeStreamSendNext);
      }
    }
    printStatus();
  }
  if (done) {
    home();
  }
}
> void receiveSingleGcode() { ...
}

// homing and calibration commands-----
bool ignoreStopExecution = false;
```

Izvor: autor

Nakon izvršene linije šalje \$NEXT# kako bi dao do znanja računalu da pošalje sljedeću liniju. G-kôd *streaming* radi tako da će, u slučaju da ne naiđe na određenu USB naredbu, preusmjeriti dobivenu liniju na čitanje u *readCommandLine()*. Čitanje radi tako da dobivenu liniju rastavlja na dijelove, prvo koristeći razmak. U slučaju da naiđe na zagrade, ignorirat će tu liniju. Nakon toga pojedina dobivena riječ se rastavlja na slovo i broj. Ako se radi o M, prevodi se M-komanda, a ako se radi o G, odradit će se G-komanda. Broj se zatim koristi za postavljanje vrijednosti varijabli kao što su varijabla (*motionMode*) koja se koristi za određivanje vrste kretanja u slučaju G0-G3 ili za odabir glavnih osi G18-G19, dok kod M komande utječu na vrtnju motora i prekid izvođenja programa. Ako se radi o drugim slovima, broj se preusmjerava kao određeni parametar i, nakon što se pročita linija, stroj izvršava kôd.

Slika 31. Dio koda koji čita linju G-koda i statusne varijable

```

486 }
487 //i za debug
488 int i = 1;
489 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
490 // Process G-code line
491 void readCommandLine(String gCodeLine) {
492
493     checkStopBtn();
494     Serial.println(gCodeLine);
495     resetPosVars();
496     calcStuff();
497
498     for (unsigned int i = 0; i < gCodeLine.length(); i++) {
499         if (gCodeLine[i] == '\n') {
500             ignore = false;
501         } else if (gCodeLine[i] == '(') {
502             ignore = true;
503         }
504
505         if (!ignore) {
506             if (validChar(gCodeLine[i])) {
507                 readPC = true;
508                 output += gCodeLine[i];
509             } else if (readPC) {
510                 gcodeSegmentReceiv(output);
511                 readPC = false;
512                 output = "";
513             }
514         }
515     }
516
517     if (readPC && !(output == "")) {
518         gcodeSegmentReceiv(output);
519         output = "";
520         readPC = false;
521     }
522
523     if (!ignore) {
524         executeCommand();
525     }
526
527     i++;
528 }
529
530 // Handle received G-code segment
531 void gcodeSegmentReceiv(String code) {
532     char c = code[0];
533     code.remove(0, 1);
534     double num = code.toDouble();
535
536     switch (c) {
537         case 'G': gCom(num); break;
538         case 'M': mCom(num); break;
539         case 'F': feedRate = num; break;
540         case 'I': Ivalue = num; break;
541         case 'J': Jvalue = num; break;
542         case 'K': Kvalue = num; break;
543         case 'R': Rvalue = num; break;
544         case 'S': Svalue = num; break;
545         case 'X': Xvalue = num; break;
546         case 'Y': Yvalue = num; break;
547         case 'Z': Zvalue = num; break;
548     }
549
550     // Reset nonModal variables
551     void resetPosVars() { ...
552 }
553
554 // set origin
555 void setOrigin() { ...
556 }
557
558 // -----Gocde iz
559
560 bool readPC = false;
561 bool ignore = false;
562 String output = "";
563
564 // G-code status variables
565 bool spindleOn = false;
566 bool spindleDirClock = true;
567 bool unitModeMM = true;
568 bool distanceModeAbs = false;
569 bool toolLengthOffset = false;
570 bool cutterCompensation = false;
571 bool exactStop = false;
572 bool feedRateModeUnitPain = true;
573 byte motionMode = 0; // g0 1 / g1
574 bool xyPlane = true;
575 bool xzPlane = false;
576 bool yzPlane = false;
577
578 // G-code parameter status variabl
579 float feedRate = 0;
580 float Ivalue = 0;
581 float Kvalue = 0;
582 float Jvalue = 0;
583 float Rvalue = 0;
584 float Svalue = 0;
585 float Xvalue = 0;
586 float Yvalue = 0;
587 float Zvalue = 0;
588
589 //speed
590 float speedVar;
591
592 //stuff thats recalculated
593 bool calc = false;
594 float xAxisResolution;
595 float yAxisResolution;
596 }

```

Izvor: autor

Kod izvršavanja kôda postoji nekoliko slučajeva, a to su paljenje i gašenje vretena te pravocrtno i kružno gibanje. Kod paljenja motora broj okretaja u minuti izračunava se u postotak PWM-a prema kojem se izračuna PWM. Zatim se izlazni PWM postupno povećava svaki određeni vremenski razmak dok ne postigne traženu vrijednost. Za to vrijeme stroj ne radi ništa. Na isti način radi i gašenje. Za PWM izlaz koristi se Arduino funkcija `analogWrite(8-bitna vrijednost)`.

Slika 32. funkcije za pokretanje i gašenje vrtnje alata

```
1118 byte spindleDelay = 100;
1119 void runSpindle(byte spdPWM) {
1120     Serial.println(spdPWM);
1121
1122     if (spindleCurrent == spdPWM) {
1123         return;
1124     }
1125     if (spdPWM == 0) {
1126         spindleOFF();
1127     } else {
1128         spindleON(spdPWM);
1129     }
1130 }
1131
1132 void spindleOFF() {
1133
1134     while (spindleCurrent > 0) {
1135         spindleCurrent--;
1136         analogWrite(spindleSpeedOut, spindleCurrent);
1137         delay(spindleDelay);
1138     }
1139 }
1140 void spindleON(byte &PWM) {
1141     if (PWM < spindleCurrent) {
1142         while (spindleCurrent > PWM) {
1143             spindleCurrent--;
1144             analogWrite(spindleSpeedOut, spindleCurrent);
1145             delay(spindleDelay);
1146         }
1147     }
1148     while (spindleCurrent < PWM) {
1149         spindleCurrent++;
1150         analogWrite(spindleSpeedOut, spindleCurrent);
1151         delay(spindleDelay);
1152     }
1153 }
1154 byte calcSpindleSpeed(float rpm) {
1155     return (rpm / spindleMaxRPM) * 255;
1156 }
```

Izvor : autor

Kod pravocrtnog kretanja prvo se izračunava broj potrebnih koraka te smjer. Zatim se traži motor s najviše koraka te se prema tome izračunaju brzine koristeći omjer broja koraka. Nakon svih izračuna osvježi se apsolutna pozicija stroja i postavljaju se parametri za svaki motor te se pokreće funkcija *moveMotors()*. U njoj se nalazi petlja koja radi dokle god motori ne postignu traženu poziciju. U međuvremenu stroj prati stanje graničnih prekidača i prekidača za hitno zaustavljanje putem funkcija *checkEndstops* i *checkStopBtn*. U slučaju da se neki od njih aktivira, motori će prestati s radom i vreteno će se polako zaustaviti.

Slika 33. Funkcija koja priprema linearni pomak

```

98 // prep data-----1042
99 void moveAllAxis(float x, float y, float z, float spd) { 1043
100     enableSteppers(); 1044
101     checkEndstops(); 1045
102     1046
103     checkStopBtn(); 1047
104     1048
105     float xt = 0, yt = 0, zt = 0; 1049
106     int xSteps, ySteps, zSteps; 1050
107     float xSpd, ySpd, zSpd; 1051
108     float xAcc, yAcc, zAcc; 1052
109     1053
110     1054
111     bool xDir = true, yDir = true, zDir = true; 1055
112     xSpd = calculateSpeedFrMMS(xAxisResoultion, spd); 1056
113     ySpd = calculateSpeedFrMMS(yAxisResoultion, spd); 1057
114     zSpd = calculateSpeedFrMMS(zAxisResoultion, spd); 1058
115     1059
116     1060
117     if (distanceModeAbs) { 1061
118         1062
119         xt = calcMove(machineAbsPosX, x); 1063
120         yt = calcMove(machineAbsPosY, y); 1064
121         zt = calcMove(machineAbsPosZ, z); 1065
122     } else { 1066
123         xt = x; 1067
124         yt = y; 1068
125         zt = z; 1069
126     } 1070
127     1071
128     if (unitModeMM) { 1072
129         xSteps = floor((xt * xAxisStepMM)); 1073
130         ySteps = floor((yt * yAxisStepMM)); 1074
131         zSteps = floor((zt * zAxisStepMM)); 1075
132     } else { 1076
133         xSteps = floor(xt * (xAxisStepMM * 25.4)); 1077
134         ySteps = floor(yt * (yAxisStepMM * 25.4)); 1078
135         zSteps = floor(zt * (zAxisStepMM * 25.4)); 1079
136     } 1080
137     1081
138     1082
139     if (xSteps < 0) { 1083
140         xSteps = xSteps * -1; 1084
141         xDir = false; 1085
142     }
143     if (ySteps < 0) {
144         ySteps = ySteps * -1;
145         yDir = false;
146     }
147     if (zSteps < 0) {
148         zSteps = zSteps * -1;
149         zDir = false;
150     }
151     if (xDir) {
152         machineAbsPosX += (xAxisResolution * xSteps);
153     } else {
154         machineAbsPosX -= (xAxisResolution * xSteps);
155     }
156     if (yDir) {
157         machineAbsPosY += (yAxisResolution * ySteps);
158     } else {
159         machineAbsPosY -= (yAxisResolution * ySteps);
160     }
161     if (zDir) {
162         machineAbsPosZ += (zAxisResolution * zSteps);
163     } else {
164         machineAbsPosZ -= (zAxisResolution * zSteps);
165     }
166     int primarySteps = max(xSteps, max(ySteps, zSteps));
167     if (primarySteps == 0) { return; }
168     float xRelSpd = (float)xSteps / primarySteps * xSpd;
169     float yRelSpd = (float)ySteps / primarySteps * ySpd;
170     float zRelSpd = (float)zSteps / primarySteps * zSpd;
171     setMotorParameters(stepperX, xRelSpd, xAcc, xSteps, xDir);
172     setMotorParameters(stepperY, yRelSpd, yAcc, ySteps, yDir);
173     setMotorParameters(stepperZ, zRelSpd, zAcc, zSteps, zDir);
174     moveMotors();
175 }

```

Izvor: autor

Slika 34. Pokretanje motora

```
void moveMotors() {  
  
    while (stepperX.distanceToGo() != 0 || stepperY.distanceToGo() != 0 || stepperZ.distanceToGo() != 0) {  
        checkStopBtn();  
  
        if (checkStopBtn() || checkEndstops()) { return; }  
  
        stepperX.run();  
        if (checkStopBtn() || checkEndstops()) {  
            Serial.println("X");  
            return;  
        }  
        stepperY.run();  
        if (checkStopBtn() || checkEndstops()) {  
            Serial.println("Y");  
            return;  
        }  
        stepperZ.run();  
        if (checkStopBtn() || checkEndstops()) {  
            Serial.println("Z");  
            return;  
        }  
    }  
}
```

Izvor: autor

Slika 35. Funkcije koje provjeravaju status prekidača

```
1210 }  
1211 bool checkStopBtn() {  
1212     if (digitalRead(stpBtnPin) == LOW) {  
1213         Serial.println("$STOPBTN#");  
1214         stopExecutionSB();  
1215         spindleOFF();  
1216         running = false;  
1217         return true;  
1218     }  
1219     return false;  
1220 }  
1221  
1222 bool checkEndstops() {  
1223  
1224     if (digitalRead(endStpPin) == LOW) {  
1225         if (!ignoreStopExecution) {  
1226             running = false;  
1227             spindleOFF();  
1228             stopExecution();  
1229             return true;  
1230         }  
1231     }  
1232  
1233     return false;  
1234 }
```

Izvor: autor

Za kružni pokret ovisno o odabranim primarnim osima G17-G19 šalju se parametri u funkcije za izračun točaka kružnice. Funkcija za početak zaokruži točke, izračuna radijus i odredi gdje se na kružnici nalazi početna točka putem funkcije *getPointPosOnCircle*.

Slika 36. followCircle, dio 1

```
float outputA;
float outputB;
bool doFullCircle = false;
void followCircle(float startA, float startB, float centerA, float centerB, float targetA, float targetB, float increment, float threshold, bool dirClock) {

    float ctrA = centerA / increment;
    float ctrB = centerB / increment;

    float ta = targetA / increment;
    float tb = targetB / increment;

    float sa = startA / increment;
    float sb = startB / increment;

    ctrA = floor(ctrA) * increment;
    ctrB = floor(ctrB) * increment;

    ta = floor(ta) * increment;
    tb = floor(tb) * increment;

    sa = floor(sa) * increment;
    sb = floor(sb) * increment;

    if (sa == ta && sb == tb) {
        doFullCircle = true;
    } else {
        doFullCircle = false;
    }

    debugLn("follow circle: " + String(sa) + "," + String(sb) + " center: " + String(ctrA) + "," + String(ctrB) + " target: " + String(ta) + "," + String(tb));

    float pstarta = sa;
    float pstartb = sb;
    float radius = getTwoPointDistance(ctrA, ctrB, ta, tb);

    int checker = (4 * radius) / increment;
```

Izvor: autor

Slika 37. followCircle, dio 2

```
if (radius <= increment) {
    debugLn("too small circle!");
    followCircleOutput(targetA, targetB);
    return;
}

bool runCircle = true;
while (runCircle) {

    if (getTwoPointDistance(pstarta, pstartb, ta, tb) < threshold && !doFullCircle) {
        debugLn("treshold reached!");
        runCircle = false;
        break;
    }

    if (checker == 0) { break; }

    checker--;
    calcNextPoint(pstarta, pstartb, radius, ctrA, ctrB, increment, dirClock);
    pstarta = outputA;
    pstartb = outputB;

    followCircleOutput(pstarta, pstartb);
}

followCircleOutput(targetA, targetB);
```

Izvor: autor

Zatim šalje parametre u funkciju *calcNextPoint* koja služi kao središnja funkcija nakon obrade točaka, a *findCirclePoint* za traženje točke.

Slika 38. calcNextPoint funkcija

```
void calcNextPoint(float startA, float startB, float radius, float centerA, float centerB, float increment, bool dirClock) {
    int pointPos = getPointPosOnCircle(startA, startB, centerA, centerB);
    debugLn("point pos:" + pointPos);
    // 1-> POINT 1/4    4->POINT 4/4
    // 5 0DEG, 7 180  6 90 AND 8 270
    switch (pointPos) {
        case 1:
        case 2:
            if (dirClock) {
                findCirclePoint(startA, centerA, centerB, radius, increment, true, true);
            } else {
                findCirclePoint(startA, centerA, centerB, radius, increment, false, true);
            }
            break;
        case 3:
        case 4:
            if (dirClock) {
                findCirclePoint(startA, centerA, centerB, radius, increment, false, false);
            } else {
                findCirclePoint(startA, centerA, centerB, radius, increment, true, false);
            }
            break;
        case 5:
            if (dirClock) {
                findCirclePoint(startA, centerA, centerB, radius, increment, false, false);
            } else {
                findCirclePoint(startA, centerA, centerB, radius, increment, false, true);
            }
            break;
        case 6:
            if (dirClock) {
                findCirclePoint(startA, centerA, centerB, radius, increment, true, true);
            } else {
                findCirclePoint(startA, centerA, centerB, radius, increment, false, true);
            }
            break;
        case 7:
    }
```

Izvor: autor

U funkciji *findCirclePoint* jedna se os iterira odmah, a druga se iterira tako dugo dok udaljenost nije veća ili jednaka radijusu kružnice.

Slika 39. *findCirclePoint* funkcija

```
void findCirclePoint(float axisA, float centerAxisA, float centerAxisB, float radius, float increment, bool right, bool up) {
    outputA = axisA;
    outputB = centerAxisB;
    if (right) {
        outputA += increment;
    } else {
        outputA -= increment;
    }

    while (getTwoPointDistance(outputA, outputB, centerAxisA, centerAxisB) < radius) {
        if (up) {
            outputB += increment;
        } else {
            outputB -= increment;
        }
    }

    // DECREASE IF ITS MORE THAN 20% OF RADIUS OUTSIDE CIRCLE
    if ((getTwoPointDistance(outputA, outputB, centerAxisA, centerAxisB) - radius) > radius * 0.2) {
        if (up) {
            outputB -= increment;
        } else {
            outputB += increment;
        }
    }
}
```

Izvor: autor

Nakon izračunatih točaka os koja se iterirala, ako je udaljenost veća od radijusa za 20 %, oduzima se jedna vrijednost iteracije. Izračunate točke u početnoj funkciji šalju se u funkciju za izlaz, koja prije slanja u funkciju za linearni pomak vraća osi nazad u pravilan redoslijed ovisno o odabranim G17-G19. Nakon izvršenog pomaka ponavlja se računanje sve dok udaljenost ciljane i trenutne točke nije manja od određene vrijednosti. Kôd uz to podržava relativne i apsolutne vrijednosti (G90-G91), mogućnost korištenja mm ili inčeva (G20, G21). Postoje i dodatne funkcije koje se mogu pokrenuti uz određene USB komande, a to su *Home* i *Zcal*. *Home* će alatnu glavu stroja staviti u gornji lijevi kut, a *Zcal* će pomicati Z os sve dok ne udari u senzor te je vratiti 5 mm iznad njega. S obzirom na to da je senzor visine 19.5 mm, glava će biti 24.5 mm iznad materijala. To radi tako da šalje G0 Z-5 F20 liniju tako dugo dok ne udari u *endstop*. Budući da je senzor spojen zajedno s graničnim gumbima, očitat će *true* te nakon toga onemogućuje prekid pokretanja te šalje liniju pomak +5 mm gore i nazad omogućuje prekidače. *Home* funkcija radi na isti način ponavljajući postupak za svaku os.

Slika 40. zCal funkcija

```
bool zcal = false;
void zCal() {
    SendReply(okA);
    zcal = true;
    enableEndstops();
    running = true;

    String line = "G0 Z-5 F20";
    while (running) {
        checkStopBtn();
        if (checkEndstops()) {
            running = false;
            break;
        }
        delay(1000);
        readCommandLine(line);
    }
    disableEndstops();
    readCommandLine("G0 Z5 F20");
    enableEndstops();

    setOrigin();
    machineAbsPosZ = 24.5;
    zcal = false;
}
```

Izvor: autor

Slika 41. Dio Home funkcije

```
void home() {
    //ZAXIS
    enableEndstops();
    running = true;

    String line = "G0 Z15 F20";
    while (running) {
        checkStopBtn();
        if (checkEndstops()) {
            running = false;
            break;
        }
        delay(1000);
        readCommandLine(zAxisHome);
    }
    checkStopBtn();

    disableEndstops();
    readCommandLine(zAxisClear);
    enableEndstops();

    //XAXIS
    running = true;

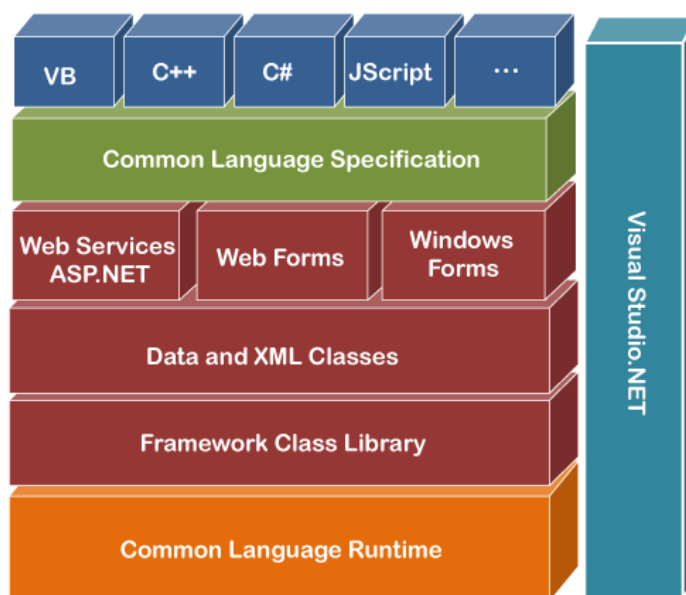
    line = "G91 G0 X-50 F200";
    while (running) {
        checkStopBtn();
        if (checkEndstops()) {
            running = false;
            break;
        }
        delay(100);
        readCommandLine(xAxisHome);
    }
}
```

Izvor: autor

3.5. C# Windows Forms .NET

Windows Forms je programski okvir za izradu grafičkog sučelja na računalu i dio je .NET Frameworka. Daje mogućnost lagane izrade desktop programa s elementima koji se postavljaju na sam okvir. .NET je okruženje za razvoj softvera koji se sastoji od mnogo komponenata u različitim jezicima koji se prevode na zajednički jezik. Uključuje puno biblioteka za izradu raznih aplikacija te predložaka za njih. U projektu je korišten za izradu programa koji će slati komande i G-kôd na Arduino za izvršavanje. U .NET Framework postoje biblioteke za rad s datotekama te portovima koji olakšavaju rad s njima. Korištene su klase: *Port*, *openFileDialog*, *File* te *FileReader*.

Slika 42. .NET struktura



Izvor: <https://www.javatpoint.com/vb-net-dot-net-framework-introduction> (pristup 13.8.2024)

3.6. Arduino CNCController

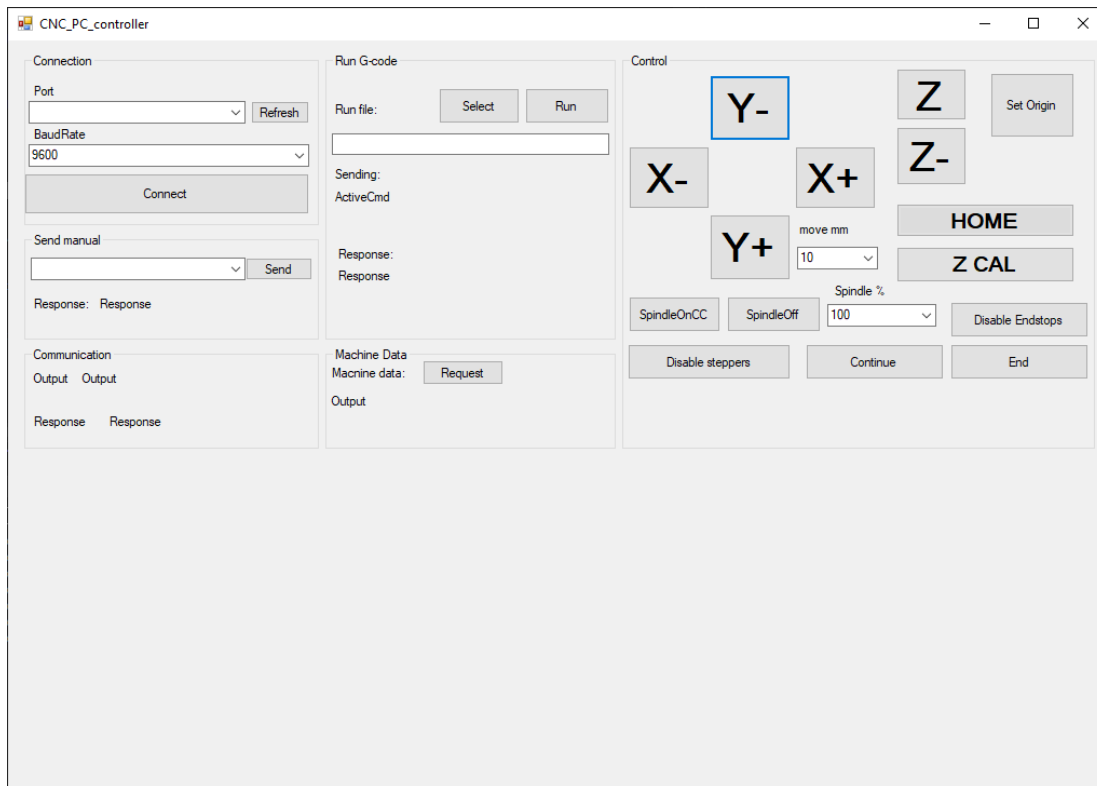
Arduino CNCController je naziv za softver razvijen za potrebe projekta. Izrađen je u C# Windows Forms .NET okruženju. Cilj ovog programa je slati komande na Arduino. Koristi .NET za datoteke i USB komunikaciju. Pri pokretanju program traži dostupne portove koji se zapisuju u listu. Nakon što odabere dostupan port, korisnik odabire brzinu slanja podataka te se pritiskom na gumb *connect* spaja na Arduino. Ima mogućnost kontroliranja svake osi pojedinačno koje se mogu koristiti kako bi se glava dovela na početnu poziciju. Odabirom datoteke može se pokrenuti izvršavanje G-kôda nakon čega program šalje liniju po liniju na Arduino. USB komunikacija se sastoji od jednostavnog protokola gdje Arduino očekuje određenu komandu te šalje određen odgovor po primitku. U početku se ostvaruje veza slanjem *\$START* na što Arduino odgovara sa *\$CONNECTED #*. Kako bi započelo slanje G-kôda, šalje se naredba *\$BEGIN*, nakon čega Arduino odgovara *\$OK#* te zatim očekuje G-kôd liniju i, nakon što se izvrši, šalje *\$NEXT#*. Sama logika komunikacije je prepuštena drugoj klasi nazvanoj *ArduinoCommunicator* koja sadrži niz funkcija i definirane varijable za komunikaciju.

Slika 43. Udio kôda za čitanje datoteke i slanje G-kôda

```
1: namespace
2: public async Task<bool> StreamGcodeFileAsync(string filePath, System.Windows.Forms.Label inputLbl,
3: System.Windows.Forms.Label responseLbl, System.Windows.Forms.Label machinePos){
4:     Streaming = true;
5:
6:     string begin = await Task.Run(() => SendAndReply(startGcodeStream, 3, 100));
7:     responseLbl.Text = begin;
8:
9:     if (!begin.Contains(ok))
10:    {
11:        LogLn("failed to start gcode stream");
12:        return false;
13:    }
14:
15:    if (File.Exists(filePath))
16:    {
17:        try
18:        {
19:            using (StreamReader reader = new StreamReader(filePath))
20:            {
21:                string line;
22:                while ((line = await reader.ReadLineAsync()) != null)
23:                {
24:
25:                    inputLbl.Invoke(new Action(() => inputLbl.Text = line));
26:                    await Task.Run(() => SendCmd(line));
27:
28:                    bool wait = true;
29:                    string response = "";
30:
31:                    while (wait)
32:                    {
33:                        response = ReadResponseCMD(3600000);
34:                        LogLn("response =" + response);
35:
36:                        if (response.Contains(gcodeStreamSendNext))
37:                        {
38:                            wait = false;
39:                            LogLn("line executed!");
40:                        }
41:                        if (response.Contains(stopGcodeStream))
42:                        {
43:                            wait = false;
44:                            LogLn("gcode stream stopped: pressed endstop or stop btn");
45:                            Streaming = false;
46:                            return false;
47:                        }
48:                    }
49:                    response = "";
50:                }
51:            }
52:        }
53:    }
54: }
```

Izvor: autor

Slika 44. Izgled sučelja programa



Izvor: autor

3.7. Fusion 360

Autodesk Fusion je profesionalni program Autodesk-a, kombinira niz alata CAD⁵, CAM⁶, CAE⁷ i za PCB kako bi se moglo dizajnirati bilo što. U projektu je korišten za generiranje G-kôda i za izradu modela za 3D printanje. Postoji verzija za osobne potrebe koja je besplatna ukoliko prihodi ne iznose više od 1000 \$ godišnje. Sam rad u Fusionu u CAD modu nalik je na tehničko crtanje s time da se 2D profil pretvara u 3D oblik na razne načine, kojeg se zatim može uređivati alatima, drugim profilima i tijelima. CAD služi za dizajniranje modela, CAM

⁵ CAD (Computer aided design) – softver za izradu dizajna

⁶ CAM (Computer aided manufacturing) – softver za izradu programa za CNC strojeve

⁷ CAE (Computer aided engineering) - softver za simulaciju

je program za izradu pokreta CNC strojeva odnosno G-kôda, PCB za izradu tiskanih pločica. CAE je alat za analizu i simulaciju dizajna.

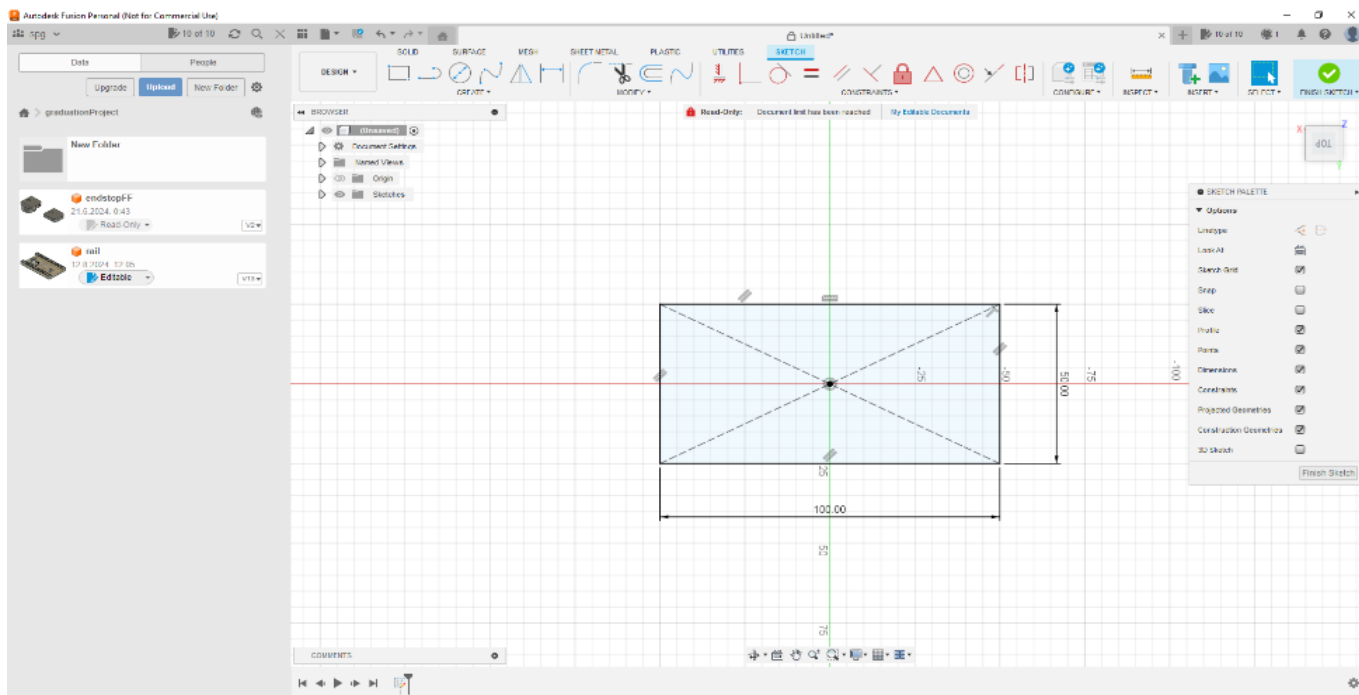
4. IZRADA G-KÔDA U FUSIONU 360

Prije izrade potrebno je postaviti alat koji će se koristiti za izradu. Prvo u gornjem lijevom kutu treba odabrati *manufacture* opciju kako bi se otvorilo sučelje za rad s CNC strojevima. Nakon toga se pod *upravljaj menijem* nalazi opcija *biblioteka alata*. Na lijevoj strani treba odabrati *lokalni*, zatim je potrebno odabrati *biblioteka* i kliknuti na gornji znak plusa, pa odabrati vrstu alata. Zatim je nužno uvrstiti niz parametara za određeni alat. Nakon dodavanja alata treba na sličan način dodati model stroja, oznaka desno od biblioteke alata. Kod stroja se odabire sličan stroj iz biblioteke Fusion360 te se povlači u lokalnu biblioteku. Nakon dodavanja moguće je podesiti parametre stroja. Posljednji nužni korak je biblioteka koja govori o strukturi G-kôda te obuhvaćanje naredbe, a kako je trenutni stroj napravljen prema GRBL⁸-u, na isti način kao i kod stroja, uzima se GRBL⁸ post. Nakon što je Fusion spreman za rad, potrebno se vratiti u sučelje za dizajniranje te napraviti oblik materijala ili dio materijala koji će biti radni prostor. Klikom na gumb za izradu sheme treba napraviti 2D oblik materijala. Zatim treba potvrditi s desne strane alatne trake kako bi se završilo crtanje sheme. Odabirom oblika te pritiskom na E, ili gornji alat za ekstruziju, unosom debljine materijala izrađuje se 3D oblik materijala. Zatim se odabirom gornje površine može nacrtati željeni oblik ili tekst. Tekst se može označiti te se napravi ekstruzija u željenu dubinu. Nakon crtanja ponovo treba otići u *manufacture* sučelje te odabrati alat 2D kontura (alat koji se nalazi na stroju kod izrade projekta podržava samo 2D konturu). Potrebno je odabrati oblike koje se želi gravirati, a može se i raditi samo o liniji u shemi. Odabirom oblika otvara se odabir alata gdje je nužno odabrati odgovarajući alat. Pritiskom na *enter* ili *ok* u skočnom

⁸ GRBL - softver otvorenog kôda koji upravlja CNC strojem

prozoru završava se izrada puteva za alat. Na alatnoj traci pod menijem akcije treba kliknuti *post proces*. Otvorit će se sučelje u kojem se datoteci daje ime te se izradi G-kôd.

Slika 45. Izrada sheme materijala



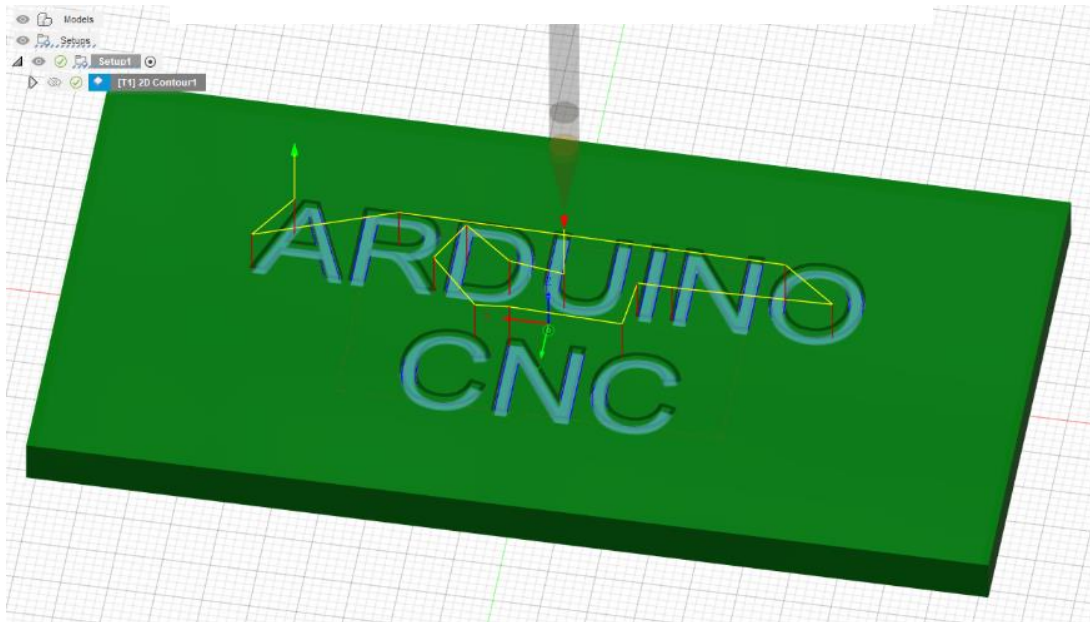
Izvor: autor

Slika 46. Crtanje teksta dizajn1

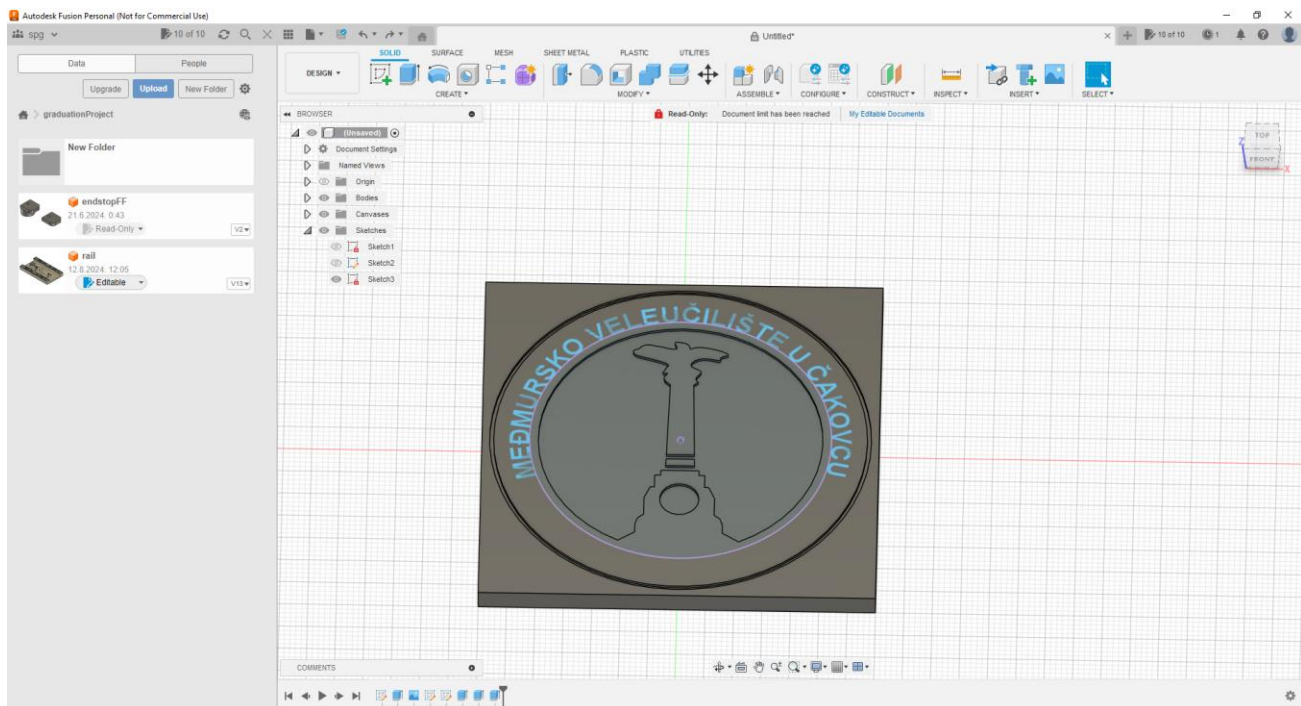


Izvor: autor

Slika 47. Tekst s putevima alata nakon odabira 2D konture

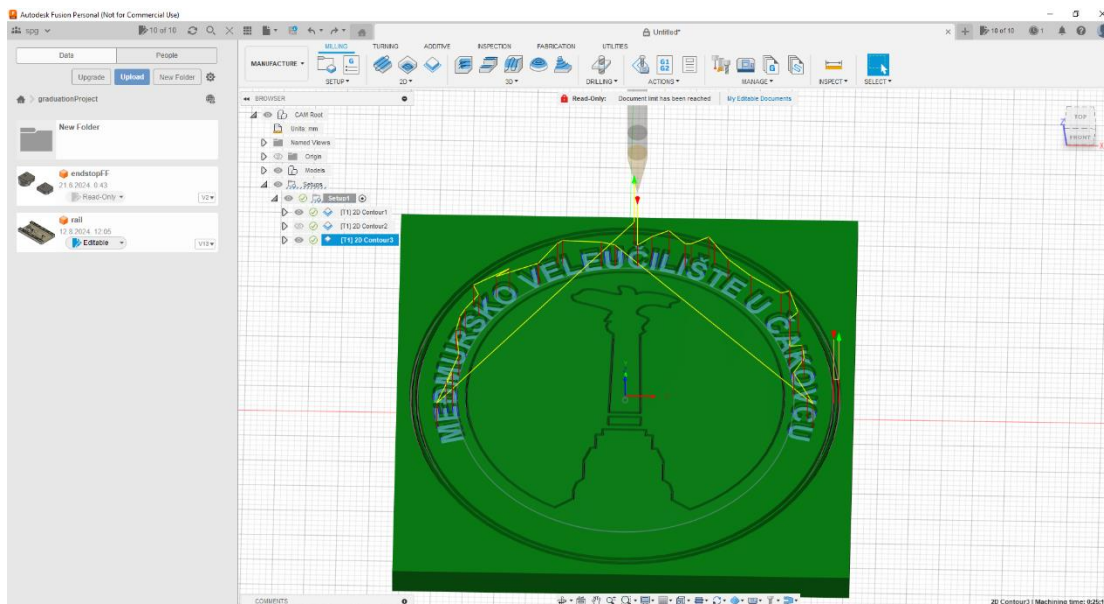


Slika 48. Dizajn 2



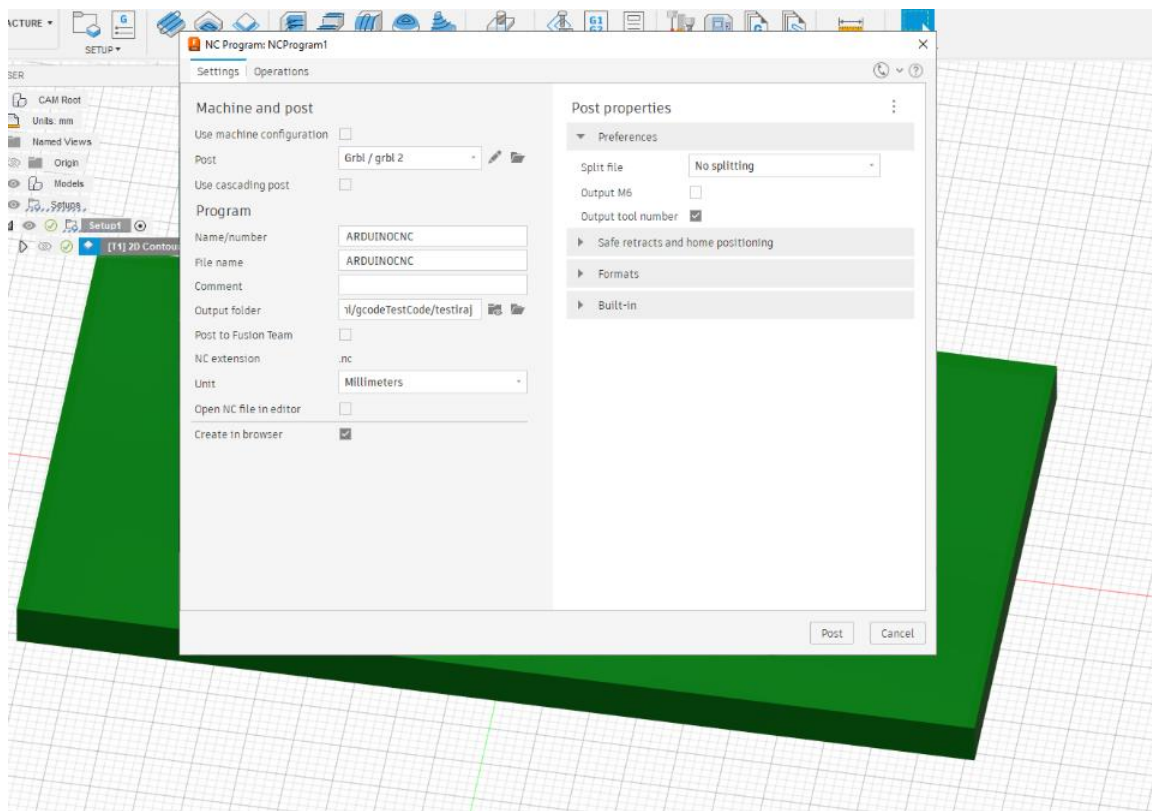
Izvor:autor

Slika 49. Dizajn 2 nakon odabira 2D kontura



Izvor: autor

Slika 50. Izrada G-kôd datoteke

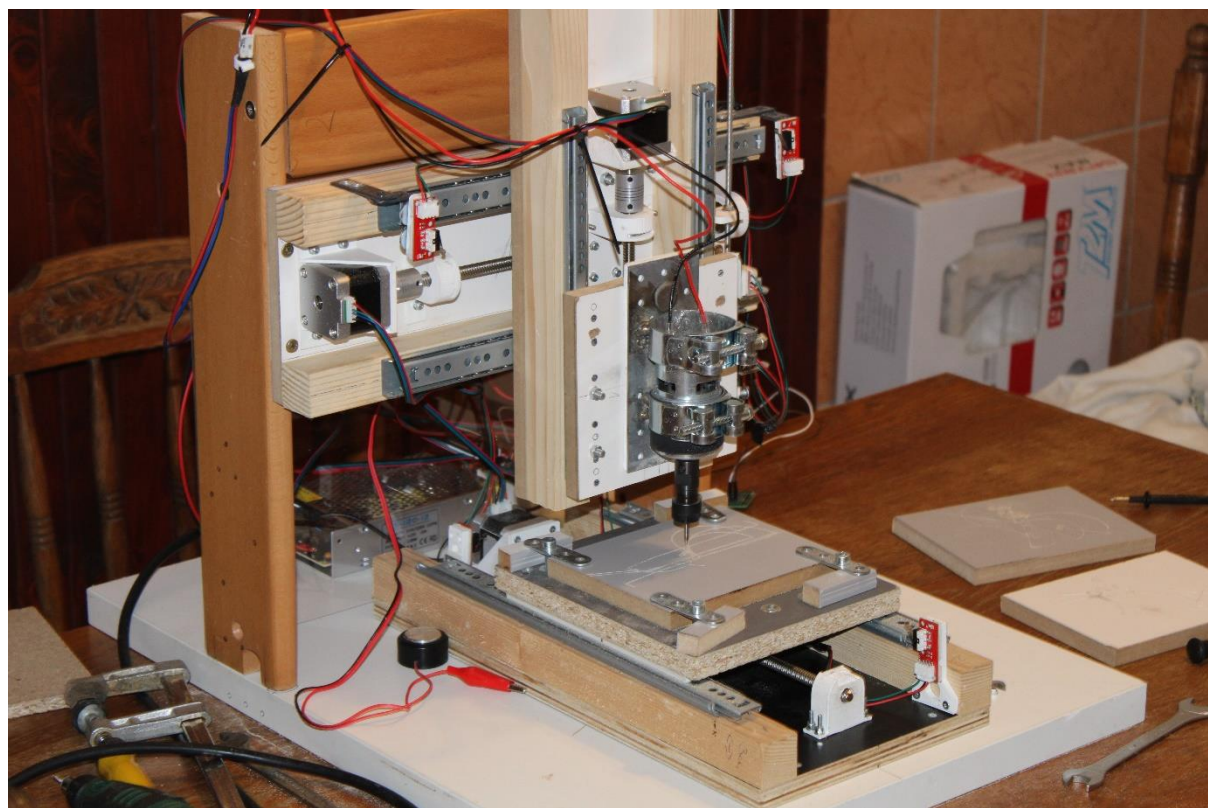


Izvor: autor

5. TESTIRANJE RADA

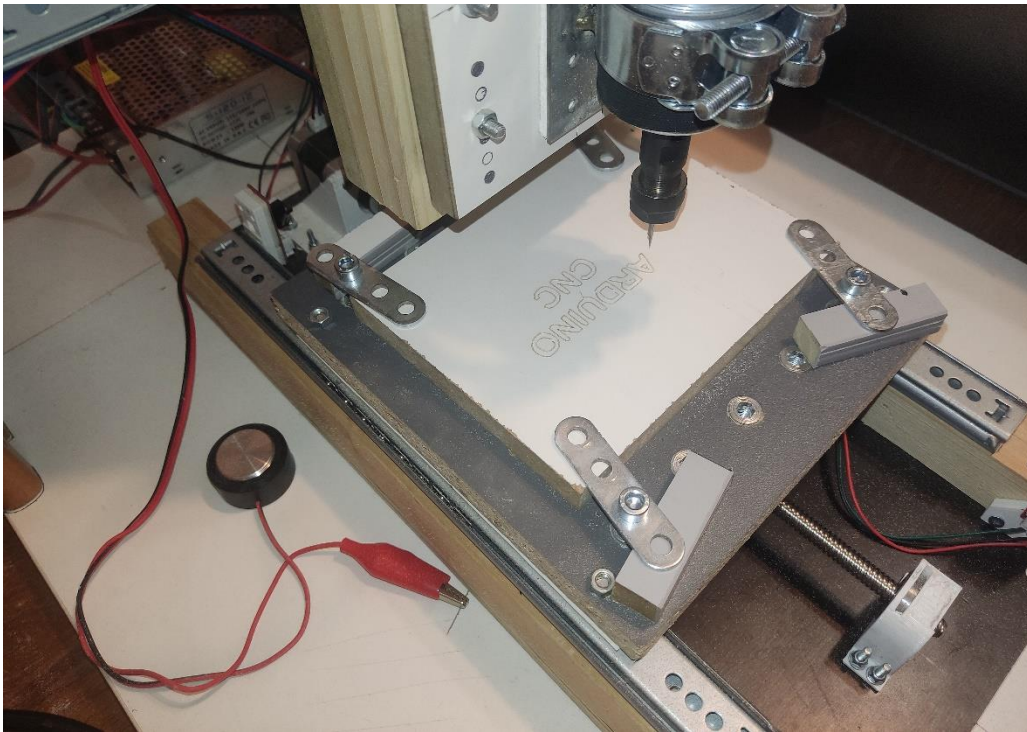
Nakon izrade G-kôd datoteke potrebno je otvoriti Arduino CNCController i povezati se sa strojem. Nakon uspješnog povezivanja nužno je dovesti alatnu glavu na poziciju za rad, odnosno na nultu točku na materijalu. To se može uraditi ručno prije spajanja Arduina ili se može sa sučeljem dovesti alat na poziciju iznad materijala. Zatim treba korištenjem opcije ZCAL postaviti alat te ga pomaknuti za 24.5 mm dolje i kliknuti *set origin*. To će osigurati da je alat točno na materijalu. Zatim se klikom na *otvori* odabire datoteka i pritiskom na *run* se pokreće izvođenje G-kôda. Prije rada važno je dobro stegnuti materijal te staviti zaštitu od letećih krhotina.

Slika 51. Izgled Projekta



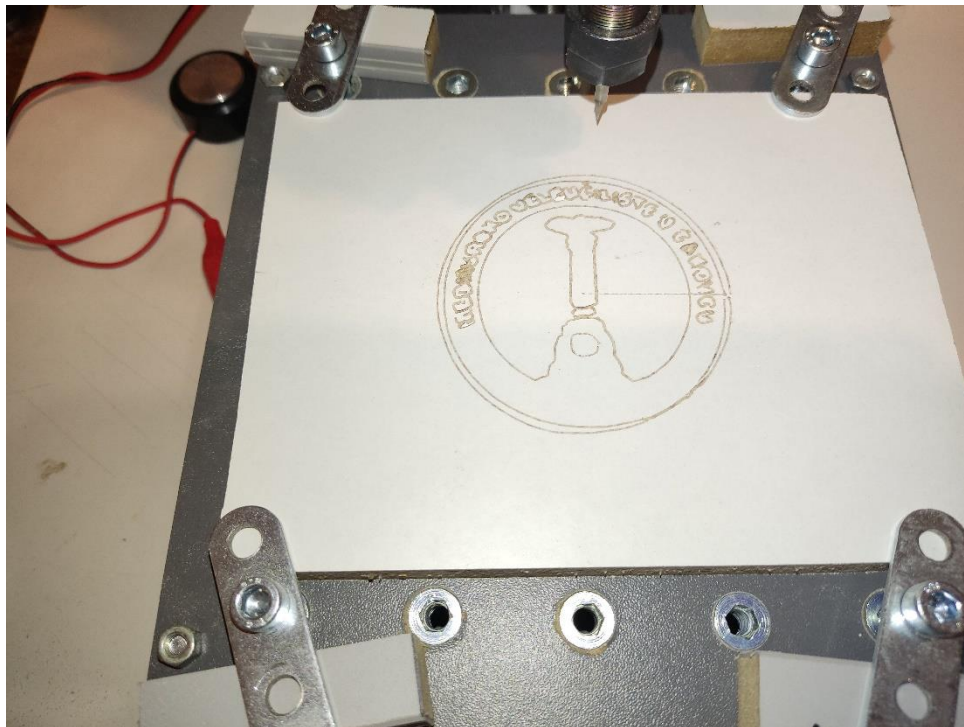
Izvor: autor

Slika 52. Rezultat prvog dizajna



Izvor: autor

Slika 53. Rezultat drugog dizajna



Izvor: autor

6. ZAKLJUČAK

Nakon same izrade utvrđeno je da treba koristiti trapezna vretena sa što manjim hodom kako bi se izbjegli neželjeni prazni hodovi zbog tolerancije u navoju. Programiranje vlastitog softvera za stroj zahtjevan je proces jer, kako bi se što kvalitetnije napravio, zahtijeva dobro poznavanje G-kôda, programiranja, hardvera, matematike, algoritama i elemenata strojarstva. S trenutnim softverom mogu se urezivati osnovni oblici, no preciznost varira te kao takav nije dobar za sve namjene. Kako bi se poboljšala preciznost, ubuduće bi trebalo zamijeniti trapezna vretena na X i Y osi tako da bi imale hod od 2 mm te bi, umjesto teleskopskih vodilica, za ladice trebalo koristiti linearne vodilice koje bi trebalo montirati što više paralelno s vretenom. Za rad s čvršćim materijalima odlično bi bilo zamijeniti konstrukciju za aluminijsku ili umjesto istosmjernog motora koristiti laser. Cijena tih poboljšanja dosegla bi cijenu gotovog CNC usmjerivača te kao takva nije prihvatljiva osim ako bi se radilo o većem radnom prostoru. Uz sama mehanička unaprjeđenja, softver bi trebalo proširiti kako bi podržao sve naredbe, potrebno je poboljšati izračune točaka na kružnici korištenjem drugih metoda i algoritama, provjeriti ispravnost ulaznog kôda, a treba i bolji protokol za USB komunikaciju, korištenje asinkronih funkcija ili dretvi kod slanja kôda te dodati više mogućnosti za upravljanje, kao i pokušati smanjiti upotrebu memorije na samom upravljaču korištenjem kôda niže razine uz efikasnije korištenje resursa. Ukupno gledano, ovaj projekt uspješno demonstrira potencijal Arduino platforme u izradi CNC mašina i potvrđuje da su „uradi sam“ rješenja praktična, pristupačna i primjenjiva u širokom spektru područja.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

Bana Josipa Jelačića 22/a, Čakovec

IZJAVA O AUTORSTVU

Završni/diplomski rad isključivo je autorsko djelo studenta te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, internetskih i drugih izvora) bez pravilnog citiranja. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom i nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, _____ (ime i prezime studenta) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog rada pod naslovom

te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:



(vlastoručni potpis)

7. LITERATURA

CNC:

[1] Autor: Zdravko Blažević, rujan 2004. Programiranje CNC Tokarilice i glodalice, pristup 16. 6. 2024., dostupno na:

https://zoranpericsplit.weebly.com/uploads/1/2/4/9/12491619/skripta_cnc_-_blazevic.compressed.pdf

[2] Autori: Suk-Hwan Suh, Seong-Kyon Kang, Dae-Hyuk Chung Lan Stroud.

Theory and design of CNC Systems, pristup : 16. 6 .2024.

Dostupno na: https://books.google.hr/books?hl=hr&lr=&id=c_3TxZlnpMC&oi=fnd&pg=PR19&dq=cnc+design&ots=P_cZ6PKKtH&sig=MZ5jp-FB8Py493V5ZY40pbrM21s&redir_esc=y#v=onepage&q=cnc%20design&f=false

[3] Autor: Graham T. Smith. CNC Machining Technology, 27.studen 2013. pristup:16. 6. 2024.

Dostupno na:

https://www.google.hr/books/edition/CNC_Machining_Technology/NcPcBwAAQBAJ?hl=hr&gbpv=1&dq=cnc+design&printsec=frontcover

[4] Carbide 3D, learn, CNC engraving. Autor nije naveden . Guide to CNC Engraving .

Pristup: 16. 6. 2024 . Dostupno na: <https://carbide3d.com/learn/cnc-engraving/>

DC motor:

[5] Uk.rs-online. Autor nije naveden. The Complete guide to DC motors . Pristup: 16.6 .2024.

Dostupno na:<https://uk.rs-online.com/web/content/discovery/ideas-and-advice/dc-motors-guide>

[6] Vedantu. Working of DC motor, electric motor-Principle and Application for JEE.Autor nije naveden , pristup: 17. 6. 2024. Dostupno na:

<https://www.vedantu.com/jee-main/physics-working-principle-of-dc-motor>

[7] Ins Tolls . Autor nije naveden: Dc Motor Theory, pristup: 17. 6. 2024. , Dostupno na:

<https://instrumentationtools.com/dc-motor-theory/>

[8] Magnetic innovations. Autor nije naveden: How DC motor actually works, pristup 16. 6. 2024. Dostupno na: <https://www.magneticinnovations.com/faq/dc-motor-how-it-works/>

[9] ASPINA. Autor nije naveden: What is a DC motor? DC motor types how they work and how to control them, pristup 16. 6. 2024. Dostupno na: <https://eu.aspina-group.com/en/learning-zone/columns/what-is/030/>

Koračni motori:

[10] V.V. Athani Stepper motors fundamental, applicatons and design 2005., pristup 16. 6. 2024. Dostupno na:

https://books.google.hr/books?hl=hr&lr=&id=0m8NTozFZL8C&oi=fnd&pg=PA1&dq=stepper+motors&ots=MG8_KhxJBa&sig=17idCLoTzqLtJ5WfMMVtf4_uZuc&redir_esc=y#v=onepage&q=stepper%20motors&f=false

[11] ELPROCUS. Autor nije naveden: What is Hybrid Stepper Motor: Working & Its Applications, pristup 16. 6. 2024. Dostupno na: <https://www.elprocus.com/what-is-hybrid-stepper-motor-working-its-applications/>

[12] [LinEngineering](https://www.linengineering.com/technology/hybrid-stepper-motors), autor nije naveden: What is Stepper motor. Pristup 16. 6. 2024. Dostupno na:

<https://www.linengineering.com/technology/hybrid-stepper-motors>

NEMA:

[13] StepperOnline, autor nije naveden, Nema Stepper Motor Sizes. Pristup 18. 6. 2024. Dostupno na:<https://www.omc-stepperonline.com/nema-stepper-motor-sizes>

A4988:

[14] ALLDATASHEET. 4988 Datasheet(HTML)-Allegro MicroSystems, autor nije naveden. Pristup 16. 6. 2024. Dostupno na: <https://www.alldatasheet.com/datasheet-pdf/pdf/338780/ALLEGRO/A4988.html>

[16] How to mechatronics. Dejan Nedelkovski: How to control a stepper motor with a4988 driver and arduino, pristup 16. 6. 2024. Dostupno na:

<https://howtomechatronics.com/tutorials/arduino/how-to-control-stepper-motor-with-a4988-driver-and-arduino/>

Vodilice:

[17] Aolisheng. Autor nije naveden: what are ball bearing drawer glides.

Pristup 16. 6. 2024. Dostupno na: <https://www.aolisheng.com/blogs/news/what-are-ball-bearing-drawer-glides>

[20] SLS. Autor nije naveden. Linear Guide explained: Understanding Linear Motion Systems. Pristup: 16. 6. 2024., dostupno na: <https://www.slsbearings.com/sg-en/blog/linear-guide-explained-understanding-linear-motion-systems>

[22] THK. Autor nije naveden: Linear motion guide. Pristup: 16. 6. 2024.

Dostupno na:

<https://www.thk.com/opm/jp/en/linear/thklinearguide/>

[23] Industrial quick shearch – Linear rails pristup: 16. 6. 2024.

Dostupno na: <https://www.iqsdirectory.com/articles/linear-slides/linear-rails.html>

Trapezno vreteno:

[18] Aerobik. Autor nije naveden. Trapezno vreteno je osnovica linearnog prijenosa pomaka. Pristup: 16. 6. 2024.

Dostupno na: <https://www.aerobik.ba/trapezno-vreteno-je-osnovica-linearnog-prijenosa-pomaka/>

[19] WEB STRANICA. Autor nije naveden: Trapezno vreteno i linearni sustavi za strojeve. Pristup: 16. 6. 2024 .Dostupno na:<https://www.web-sajt.rs/trapezno-vreteno-i-linearni-sustavi-za-strojeve/>

[24] Anđelka Ređep: finomehanika. Udžbenik za strukovne škole, 2005.

Arduino:

[25] Starmotech. Autor: Deni Ajanić: Šta je to Arduino. Pristup 16. 6. 2024. Dostupno na: <https://starmotech.com/sta-je-to-arduino/>

[26] Sparkfun. Autor nije naveden: what is Arduino. Pristup 16. 6. 2024. Dostupno na: <https://learn.sparkfun.com/tutorials/what-is-an-arduino/all>

Bts7960:

[28] Handson technology. Autor nije naveden: BTS7960 High current 43A H-Bridge Motor driver. Pristup 16. 6. 2024. Dostupno na:

<https://www.handsontec.com/dataspecs/module/BTS7960%20Motor%20Driver.pdf>

[29] Ovaga. Autor: Stella Brinkley: BTS7960 Driver datasheet and circuit diagram 11. 10. 2023. Pristup 16. 6. 2024. Dostupno na:

<https://www.ovaga.com/blog/transistor/bts7960-motor-driver-datasheet-and-circuit-diagram>

H-most:

[30] Diligent.com. Autor: Kaityn Franz: What is H-Bridge 18.3.2023 . Dostupno na:

<https://diligent.com/blog/what-is-an-h-bridge/>

G-Kod:

[38] G-CODE tutor. Autor nije naveden: CNC G Codes

Pristup: 16. 6. 2024 . Dostupno na: <https://gcodetutor.com/cnc-machine-training/cnc-g-codes.html>

Napajanje:

[40] Elektronika u praksi: Prekidačko napajanje (Switch Mode Power Supply-SMPS). Autor nije naveden. Pristup 12. 8. 2024. Dostupno na:

<https://elektronikaupraksi.com/napajanje/prekidacko-napajanje-switch-mode-power-supply-smps/>

[41] Analog Devices: Switch Mode Power Supply Basics. Dostupno na:
<https://www.analog.com/en/resources/technical-articles/switch-mode-power-supply-basics.html>

Boost converter:

[42] MPS: Boost convertets [Step-Up converter]. Autor nije naveden . Pristup 12. 8. 2024.
Dostupno na:

<https://www.monolithicpower.com/en/learning/mpscholar/power-electronics/dc-dc-converters/boost-converters.net>

[43] Microsoft.Learn: Desktop Guide (Windows Forms.NET). Autori nepoznati. Pristup 12. 8.2024. Dostupno na:

<https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-8.0>

CNC shield:

[44] Maker Store. Autor nije naveden. Pristup 13. 8. 2024. Dostupno na:
<https://www.makerstore.com.au/wp-content/uploads/filebase/publications/CNC-Shield-Guide-v1.0.pdf>

AccelStepper:

[55] Arduino-references-Librariers: Accelstepper. Autor: Mike McCauley. Pristup: 9. 9. 2024.

Dostupno na: <https://www.arduino.cc/reference/en/libraries/accelstepper/>

[56] Airspayce: AccelStepper. Autor: Mike McCauley. Pristup 9. 9. 2024.

Dostupno na: <http://www.airspayce.com/mikem/arduino/AccelStepper/>

PWM:

[57] Arduino - Home-Learn: Basics of PWM. Autor: Timothy Hirzel. Pristup: 9. 9. 2024.

Dostupno na: <https://docs.arduino.cc/learn/microcontrollers/analog-output/>

UART:

[58] ROHDE&SCHWARZ: Understanding UART. Autor nije naveden. Pristup: 9. 9. 2024.

Dostupno na:

https://www.rohde-schwarz.com/us/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart_254524.html

GRBL:

[59] All3DP: GRBL Software: All you need to know. Autor: Priyank Pal. Pristup: 9. 9. 2024.

Dostupno na: <https://all3dp.com/2/grbl-software-guide/>

Optokapler:

[60] JAMECO: What is Optocoupler and How it Works. Autor: Megan Thung. Pristup 9. 9. 2024.

Dostupno na: <https://www.jameco.com/Jameco/workshop/Howitworks/what-is-an-optocoupler-and-how-it-works.html>

Fusion:

[61] Autodesk Fusion. Autor nije naveden. Pristup: 12. 9. 2024.

Dostupno na: <https://www.autodesk.com/products/fusion-360/overview?term=1-YEAR&tab=subsription>

8. POPIS SLIKA

SLIKA 1. 3D MODEL-MODUL(X,Y,Z) OSI	3
SLIKA 2. LEŽAJEVI.....	4
SLIKA 3. TRAPEZNO NAVOJNO VRETENO	5
SLIKA 4. MODEL STALKA ZA KORAČNI MOTOR	6
SLIKA 5. MODEL BLOKA ZA MATICU.	6
SLIKA 6. 3D MODEL STALAKA ZA LEŽAJ.....	7
SLIKA 7. SPOJNICA I PLOČICA	8
SLIKA 8. TELESKOPSKE VODILICE	9
SLIKA 10. PRIHVAT ZA MOTOR	10
SLIKA 11. ARDUINO UNO	11
SLIKA 12. ARDUINO UNO S CNC MODULOM TE 3 UPRAVLJAČA	12
SLIKA 14. KORAČNI MOTOR NEMA 17 HS4401.....	14
SLIKA 13. KORAČNI MOTOR IZNUTRA.....	14
SLIKA 15. MODUL UPRAVLJAČA KORAČNIH MOTORA A4988	15
SLIKA 16. BLOK-SHEMA UPRAVLJAČA KORAČNOG MOTORA.....	16
SLIKA 17. ISTOSMJERNI MOTOR S ČETKICAMA.	17
SLIKA 18. BTS7960 MODUL.....	18
SLIKA 20. SHEMA BTS7960 MODULA.....	19
SLIKA 19. H-MOST	19
SLIKA 21. IZGLED GRANIČNOG PREKIDAČA	20
SLIKA 22. SHEMA GRANIČNOG PREKIDAČA.....	20
SLIKA 23. GLJIVA TIPKALO	21
SLIKA 24. CNC TOUCH SENZOR.....	21
SLIKA 26. PREKIDAČKO NAPAJANJE.....	22
SLIKA 25. BLOK-SHEMA PREKIDAČKOG NAPAJANJA.....	23
SLIKA 27. SHEMA OSNOVNOG PRETVARAČA NAPONA.....	24
SLIKA 28. PRETVARAČ NAPONA	24
SLIKA 29. SHEMA SPOJA	25
SLIKA 30. ZAPRIMANJE G-KOD NAREDBI	28
SLIKA 31. DOUSBCOMMANDS FUNCKCIJA	28
SLIKA 32. DIO KODA KOJI ČITA LINJU G-KODA I STATUSNE VARIJABLE	29
SLIKA 33. FUNKCIJE ZA POKRETANJE I GAŠENJE VRTNJE ALATA	30
SLIKA 34. FUNKCIJA KOJA PRIPREMA LINEARNI POMAK	31
SLIKA 35. POKRETANJE MOTORA	32
SLIKA 36. FUNKCIJE KOJE PROVJERAVAJU STATUS PREKIDAČA	32
SLIKA 37. FOLLOWCIRCLE, DIO 1	33
SLIKA 38. FOLLOWCIRCLE, DIO 2	33
SLIKA 39. CALCNEPTPOINT FUNKCIJA	34

SLIKA 40. FINDCIRCLEPOINT FUNKCIJA	35
SLIKA 42. ZCAL FUNKCIJA.....	36
SLIKA 41. DIO HOME FUNKCIJE	36
SLIKA 43. .NET STRUKTURA.....	37
SLIKA 44. UDIO KÔDA ZA ČITANJE DATOTEKE I SLANJE G-KÔDA	38
SLIKA 45. IZGLED SUČELJA PROGRAMA.....	39
SLIKA 46. IZRADA SHEME MATERIJALA.....	41
SLIKA 47. 3D MODEL MATERIJALA NAKON EKSTRUZIJE	POGREŠKA! KNJIŽNA OZNAKA NIJE DEFINIRANA.
SLIKA 48. CRTANJE TEKSTA DIZAJN1	41
SLIKA 49. TEKST S PUTEVIMA ALATA NAKON ODABIRA 2D KONTURE	42
SLIKA 50. DIZAJN 2	42
SLIKA 51. DIZAJN 2 NAKON ODABIRA 2D KONTURA	43
SLIKA 52. IZRADA G-KÔD DATOTEKE	43
SLIKA 53. IZGLED PROJEKTA	44
SLIKA 54. REZULTAT PRVOG DIZAJNA.....	45
SLIKA 55. REZULTAT DRUGOG DIZAJNA	45