Šahovska tabla s automatiziranim pomicanjem figura

Klasić, Ivan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I

INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

ŠAHOVSKA TABLA S AUTOMATIZIRANIM POMICANJEM FIGURA

Završni rad

Ivan Klasić

Osijek, 2016.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Osijek, 23.09.2016.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu završnog rada na preddiplomskom stručnom studiju

Ime i prezime studenta:	Ivan Klasić
Studij, smjer:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI 4236, 06.10.2014.
OIB studenta:	20534829433
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	
Predsjednik Povjerenstva:	Doc.dr.sc. Tomislav Matić
Član Povjerenstva:	Dr.sc. Ivan Vidović
Naslov završnog rada:	Šahovska tabla s automatiziranim pomicanjem figura
Znanstvena grana rada:	**Arhitektura računalnih sustava (zn. polje računarstvo)**
Zadatak završnog rada	Temu rezervirao: Ivan Klasić U ovom završnom radu potrebno je izraditi maketu šahovske ploče na kojoj se figure pomiču pomoću udaljeno upravljanog pomičnog magneta. Potrebno je ugraditi mikroupravljački sustav za pomicanje i upravljanje elektromagnetom. Potrebno je obaviti pomicanje figura pomoću komunikacije s maketom.
Prijedlog ocjene pismenog dijela ispita (završnog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 3 Jasnoća pismenog izražavanja: 3 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	23.09.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 23.09.2016.

Ime i prezime studenta:	Ivan Klasić
Studij:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI 4236, 06.10.2014.
Ephorus podudaranje [%]:	

Ovom izjavom izjavljujem da je rad pod nazivom**: Šahovska tabla s automatiziranim pomicanjem figura**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Cilj ovog završnog rada je napraviti šahovsku tablu sa automatiziranim pomicanjem figura, koja će služiti kao prikaz partije šaha koja se trenutno igra na računalu s kojim je povezana. Za uspješno izradu sustava potrebno se upoznati sa svim tehnologijama i okruženjima koja se koriste pri izradi kao što je CNC, Arduino razvojno okruženje, rad sa elektromotorima i elektromagnetom, te njihovo upravljanje, što opisano u drugom poglavlju. Treće poglavlje donosi izradu projekta koji sam po sebi predstavlja jedan konačan sustav gdje korisnik, putem računalne aplikacije, šalje naredbe koje će sustav onda pomoću svih korištenih komponenti, povezanih u jednu cjelinu, upotrijebiti da bi uspješno odradio njemu naređen zadatak.

1.1. Zadatak završnog rada

U ovom završnom radu potrebno je izraditi maketu šahovske ploče na kojoj se figure pomiču pomoću udaljeno upravljanog pomičnog magneta. Potrebno je ugraditi mikroupravljački sustav za pomicanje i upravljanje elektromagnetom. Potrebno je obaviti pomicanje figura pomoću komunikacije s maketom.

2. PRIMJENJENE TEHNOLOGIJE I KOMPONENTE

Komponente korištene u projektu podijeljene su prema ulozi u nekoliko potpoglavlja, te su u daljnjem tekstu za svaku od njih navedene uloge, sastavni dijelovi i mehanizmi rada.

2.1. CNC (Computer Numeric Control)

Danas je industrija nezamisliva bez CNC strojeva (Sl.2.1.). Oni imaju manje više iste dijelove kao stari, ručno kontrolirani strojevi. Bitna razlika je dodana kontrolna (CNC) jedinica i motori, posebice koračni motori, na sve osovine. Sustav je upravljan precizno programiranim naredbama spremljenim na neki od medija za pohranu, najčešće na USB flash uređaj. Moderni CNC sustavi, kao upravljačke naredbe, koriste datoteke kreirane CAD[1] tehnologijom.

Sl. 2.1. – *CNC glodalica u metalskoj industriji.*

CNC sustavi koriste koordinate u radnom polju kao smjernice pri radu, te se praćenjem položaja postavljaju rute koje će pri izvedbi programa alat pratiti pomoću motora koji upravljaju tri osi prostora. X-os i Y-os za dužinu i širinu te Z-os za visinu ili dubinu.

Minimizirana verzija ovo sustava izvedena u projektu koristi samo dvije osi i koračne (*stepper*) motore sa remenom za pomicanje i sustav kotača na osovinama.

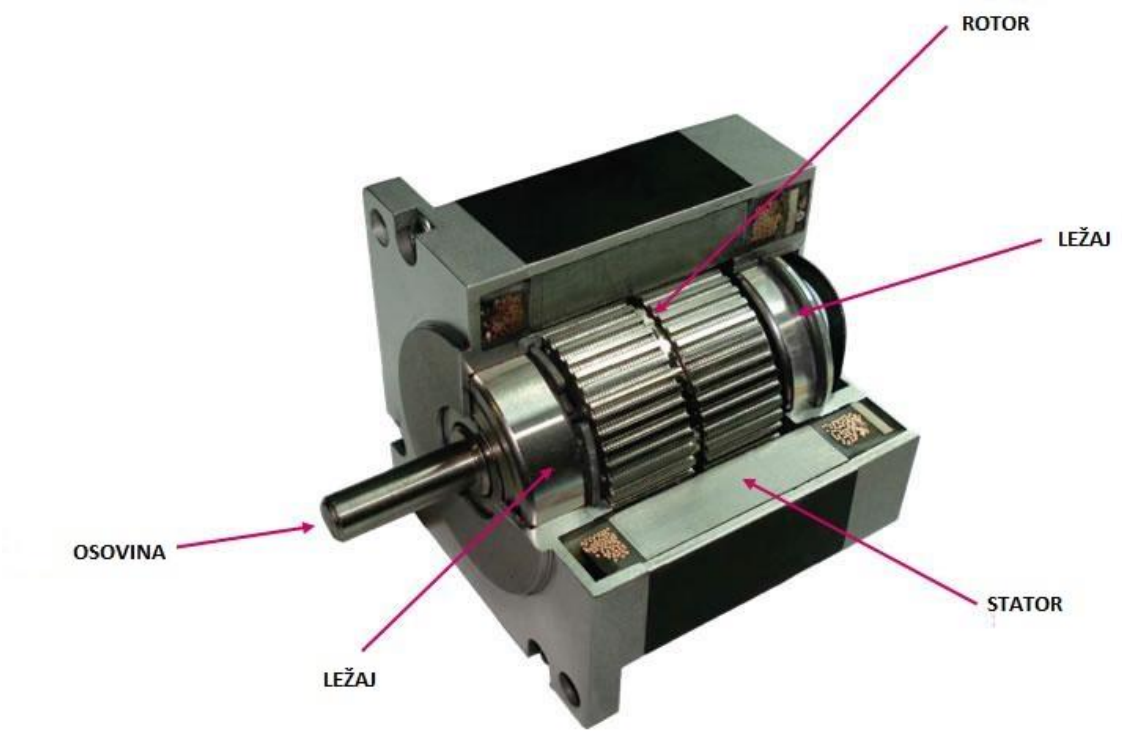
Sl. 2.2. – *Kotačić s ležajem*

[1] CAD je skraćenica od *Computer-aided Design* (dizajn potpomognut računalom), i označava uporabu računala kroz proces dizajna i stvaranja dokumentacije. Korištenjem CAD programa povećava se produktivnost, kvaliteta dizajna, točnost proračuna, te ono najvažnije, smanjuje se vrijeme od same ideje do izrade gotovog predmeta.[1]

2.2. Koračni motor

Koračni motor (*stepper motor*)(Sl. 2.3.) je izvedba istosmjernog motora bez četkica koji punu rotaciju dijeli u broj koraka, te se korisnik može motor zaustaviti u bilo kojem koraku punog kruga bez povratnog (*feedback*) senzora. Za razliku od klasičnih istosmjernih motora na polove koračnog motora dovodimo niz pravokutnih impulsa.[3]

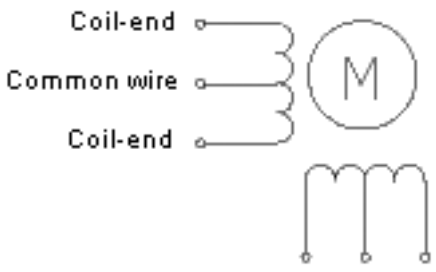
Kod dvofaznih koračnih motora imam dva načina namatanja elektromagnetskih namotaja: bipolarni i unipolarni.



Sl. 2.3. – *Presjek koračnog motora*

Unipolarni motor

Kod unipolarnog motora svaka faza ima namotaj sa dva krajnja kontakta i kontaktom izvedenim iz sredine namotaja kao što je prikazano na slici 2.4.. Takvom izvedbom omogućena je promjena smjera bez promjene polariteta napajanja, te je kontrolno sklopovlje vrlo jednostavno (npr. po jedan tranzistor za svaki namotaj).

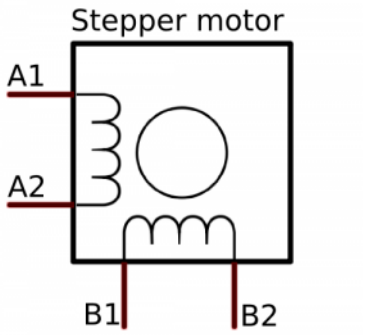


Sl. 2.4. – *Unipolarni koračni motor.*

Mikroupravljač može biti korišten za aktivaciju tranzistora pravilnim redoslijedom, te lakoća upravljanja čini ovu vrstu motora popularnu kod hobista i oni su najjeftiniji način za dobivanje preciznog kutnog pomaka.

Bipolarni motor

Bipolarni motori imaju jedan namotaj po fazi(Sl. 2.5.). Polaritet struje mora biti obrnut da bi se obrnulo magnetsko polje, pa je upravljački sklop za ovu vrstu motora kompliciraniji, tipično se koristi H-most (*H-bridge*)[2], iako danas postoje i integrirani krugovi koji to čine jednostavnijim. Zbog bolje iskorištenosti namotaja kod bipolarnih motora oni su snažniji od unipolarnih motora iste težine. Razlog tome je što unipolarni motor ima duplo više vodiča, ali je samo polovica korištena u određenom trenutku. Zaključak je dakle iako su bipolarni motori kompliciraniji za upravljanje pojavom gotovih upravljačkih sklopovlja postaju sve više popularni.



Sl. 2.5. – *Bipolarni koračni motor.*

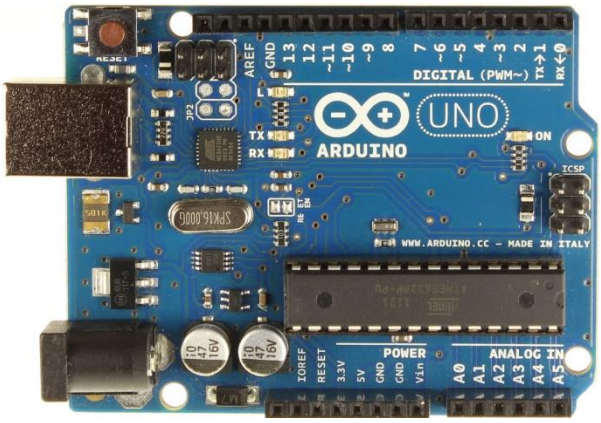
2.3. Arduino razvojno okruženje

„Arduino je ime za otvorenu računarsku i softversku platformu koja omogućava dizajnerima i konstruktorima stvaranje uređaja i naprava koje omogućava spajanje računala s fizičkim svijetom. Arduino je stvorila talijanska tvrtka SmartProjects 2005. rabeći 8-bitne mikrokontrolere Atmel AVR, da bi stvorili jedinstvenu malu i jeftinu platformu s kojom bi mogli lakše povezati računala s fizičkim svijetom.“, prema [4]. Arduino se može upotrijebiti za razvoj samostalnih interaktivnih objekata ili se može povezati sa osobnim računalom kojemu će slati podatke sa senzora na osnovu kojih će osobno računalo donositi neke odluke i preuzimati adekvatne akcije. Moguće ga je povezati sa sklopkama, raznim senzorima (npr. senzor temperature, pokreta, osvjetljenja, i dr.), svjetlosnim zaslonima, web kamerama, motorima, GPS[3] prijemnicima i sl. Postoji nekoliko različitih razvojnih sustava koji se razlikuju prema specifikacijama kao što je ukupan broj ulaza i izlaza, količina memorije, snaga procesora i dr.

[2] H-most je elektronički sklop koji omogućuje dovođenje napona na trošilo u oba smjera. Ovi sklopovi su često korišteni u robotici da bi se omogućilo okretanje istosmjernih (DC) motora u oba smjera.
[3] GPS (eng. Global Positioning System) je satelitski baziran navigacijski sustav načinjen od mreže 24 satelita postavljenih u orbitu. GPS je prvotno namijenjen za vojnu upotrebu, ali 1980-ih godina američka vlada je omogućila njegovu upotrebu i civilima.[5]

Iz Arduino obitelji najrasprostranjeniji i najdokumentiraniji razvojni sustav je Arduino UNO (Sl.2.6.). Baziran je na Atmega328P mikrokontroleru, ima četrnaest digitalnih ulaz-izlaz pinova (od kojih se šest pinova može koristit kao PWM[4] izlazi), šest analognih ulaza, 16 MHz kvarcni oscilator, te koristi napajanje od 5V DC.

Detaljne specifikacije Arduino Uno platforme nalaze se u prilogu P.2.1.



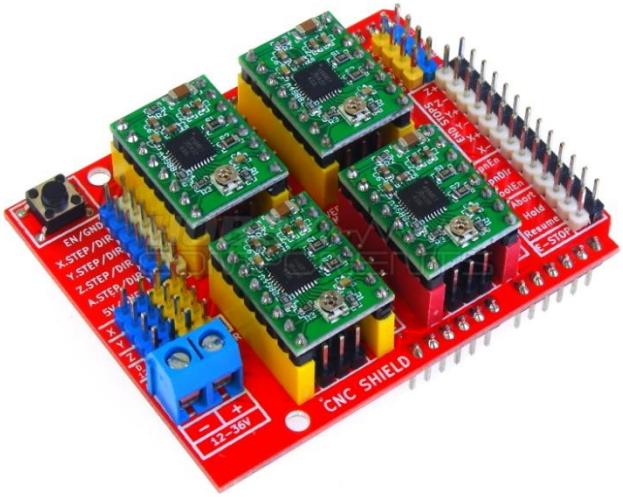
Sl. 2.6. – *Arduino UNO platforma sa ATmega328P mikrokontrolerom.*

Regulator za kontrolu motora (CNC Shield)

Regulator za kontrolu motora, prikazan na slici 2.7., služi za precizno i jednostavno upravljanje motorima. Kontrolu možemo vršiti na dva načina:

- Izvršavanjem programiranih petlji u arduino IDE-u
- Korištenjem GRBL skripte

Instalacijom željenih drivera za motore, koji se određuju prema potrebnoj struji, određenom izmjenom stanja pinova na arduinu pokrećemo motor u željenom smjeru. Regulator također ima i pinove za povezivanje tipkala za određivanje početnih i završnih pozicija (eng. End stops).



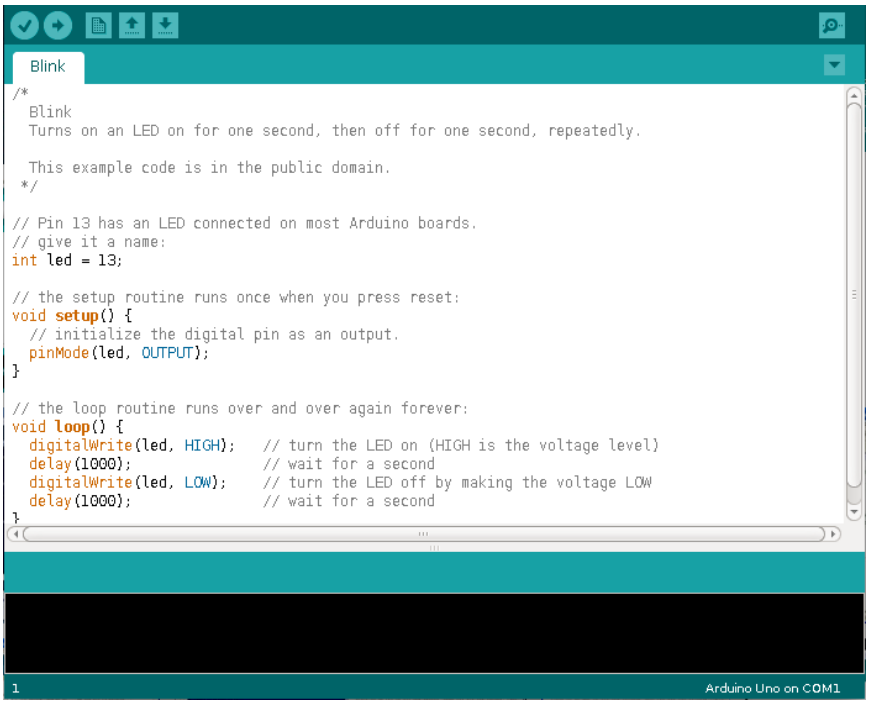
Sl. 2.7. – *Regulator za kontrolu motora*

[4] PWM – modulacija širine impulsa, karakterističan modilacijski postupak gdje se modulira vremen. ski položaj boka impulsa.

Arduino IDE

Programiranje Arduino platformi odvija se preko univerzalne serijske sabirnice[5] (engl. Universal Serial Bus, USB) korištenjem programa pod imenom *The Arduino Integrated Development Environment* ili *Arduino Software (IDE)* prikazanog na slici 2.8.

Program se sastoji od uređivača teksta (rabi se i anglizam editor teksta) za pisanje koda, zone za poruke, konzole, alatne trake sa standardnim funkcijma i serijom izbornika. Spaja se sa Arduino hardverom za slanje napisanih programa i komunikaciju.



Sl. 2.8. – *Arduino Software (IDE).*

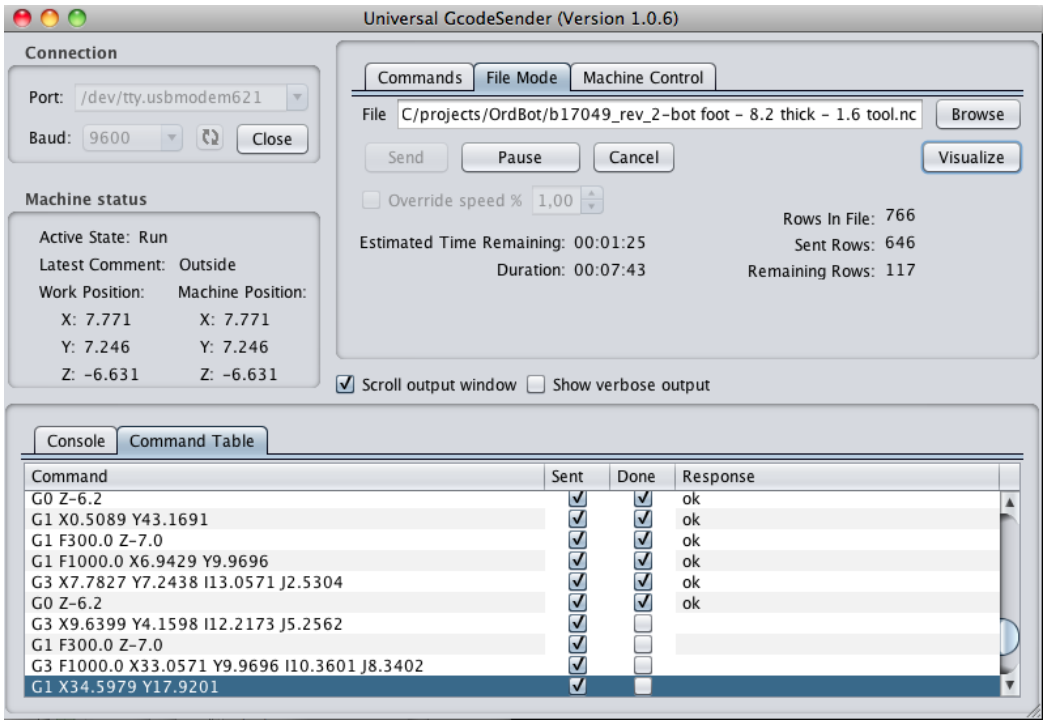
Programi napisani korištenjem *Arduino Software (IDE)* nazivaju se skice. Te skice napisane u uređivaču teksta i spremljene s ekstenzijom *.ino*. Uređivač teksta ima mogućnosti izrezivanja/lijepljenja i traženja/zamjene teksta. Zona za poruke daje povratne informacije prilikom spremanja i izvoza skica te prikaz grešaka. Konzola prikazuje kompletne poruke o pogreškama i ostale infor-

[5] Univerzalna serijska sabirnica (engl. Universal Serial Bus, USB) je tehnološko rješenje za komunikaciju računala s vanjskim uređajima pri čemu se podatci razmjenjuju serijski relativno velikom brzinom. USB je zamijenio razna dotadašnja serijska i paralelna sučelja na računalima.[6]

macije vezane za rad programa. U desnom donjem kutu programa nalazi se prikaz korištene Arduino platforme i serijskog porta. Dugmad alatne trake omogućava verificiranje i slanje programa na platformu, kreiranje, otvaranje te spremanje skica i otvaranje serijskog monitora, prema [7].

2.4. GRBL

Grbl je besplatan software, otvorenog koda i visokih performansi za kontrolu pokreta strojeva. Napisan je u programskom jeziku C i iskorištava sve mogućnosti Atmega328p[6] čipa.



Sl. 2.9. – *Program za slanje G-koda Grbl sučelju*

Kontrola grbl sučeljem vrši se slanjem G-koda mikroupravljaču. Primjer programa za slanje naredbi prikazan je na slici 2.9. Naredba za pomak se sastoji od nekoliko parametara, pa npr. imamo naredbu G90 G0 X2 Y3, gdje je G90 oznaka za apsolutni pomak, G0 oznaka za brzi pomak bez korištenja alata, X2 lokacija na x-osim Y3 lokacija na y-osi. Izmjenom ovih parametara postižemo željeni pomak te tako vršimo kontrolu nad CNC uređajem.

[6] Atmega328p – mikroupravljač (čip) koji koristi arduino uno platforma.

2.5. Računalo i aplikacija šah

Upravljanje arduinom je osobno računalo sa instaliranom pratećom windows aplikacijom, naime radi se računalnoj igrici „Quick chess“ otvorenog koda.[7] Idealna je zbog opcije igranja računalo protiv računala koja je pogodna za testiranje i predstavljanje.

Sama aplikacija je vrlo jednostavno programirana te je pogodna za preradu koja je predviđena i potrebna da bi aplikacija mogla komunicirati sa arduinom te pomoću njega pomicati figurice šaha na tabli. Aplikacija je također moćna i potiče čak i iskusne igrače šaha na razmišljanje jer u najtežem modu igre računalo predviđa buduće moguće poteze te time traži najpogodniji potez što od samoga igraća traži veliku količinu koncentracije i poznavanja trikova u šahu.

Grafičko sučelje aplikacije (Sl.2.10.) je vrlo jednostavno i osim table za igranje sadrži samo potrebne informacije koje su igraču potrebne za igranje kao što je prikaz povijesti poteza, vrijeme igranja pojedinog igrača, prikaz izbačenih figura, te alatnu traku sa izbornicima za odabir opcija igre, snimanje trenutne ili nastavak snimljene partije i dr.

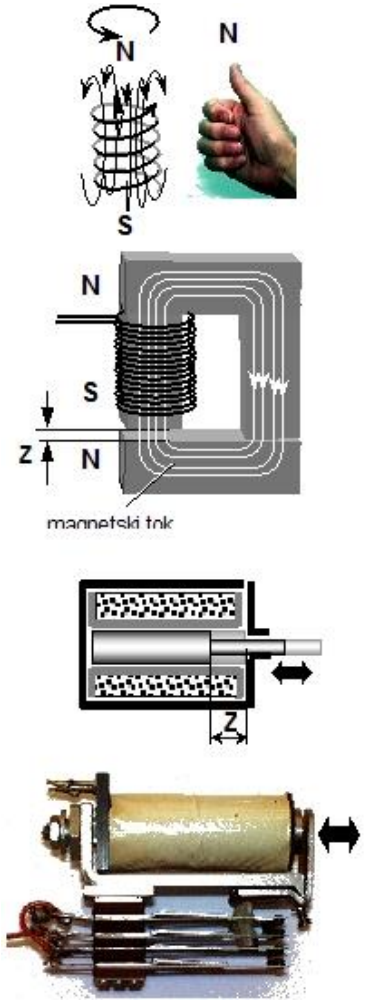
Sl. 2.10. – *Grafičko sučelje windows aplikacije "Quick Chess".*

[7] Otvoreni kod – softver čiji je izvorni kod i/ili nacrti (dizajn) dostupan javnosti na uvid, korištenje, izmjene i daljnje raspačavanje.[8]

2.6. Elektromagnet

Elektromagnet je komponenta (naprava) koja pod naponom stvara magnetsko polje (Sl.2.11.). Naime, električna struja oko električnog vodiča kroz koji prolazi stvara magnetsko polje. Razlika elektromagneta od trajnog magneta je ta da kada isključimo napajanje elektromagneta on gubi svoje magnetsko polje.

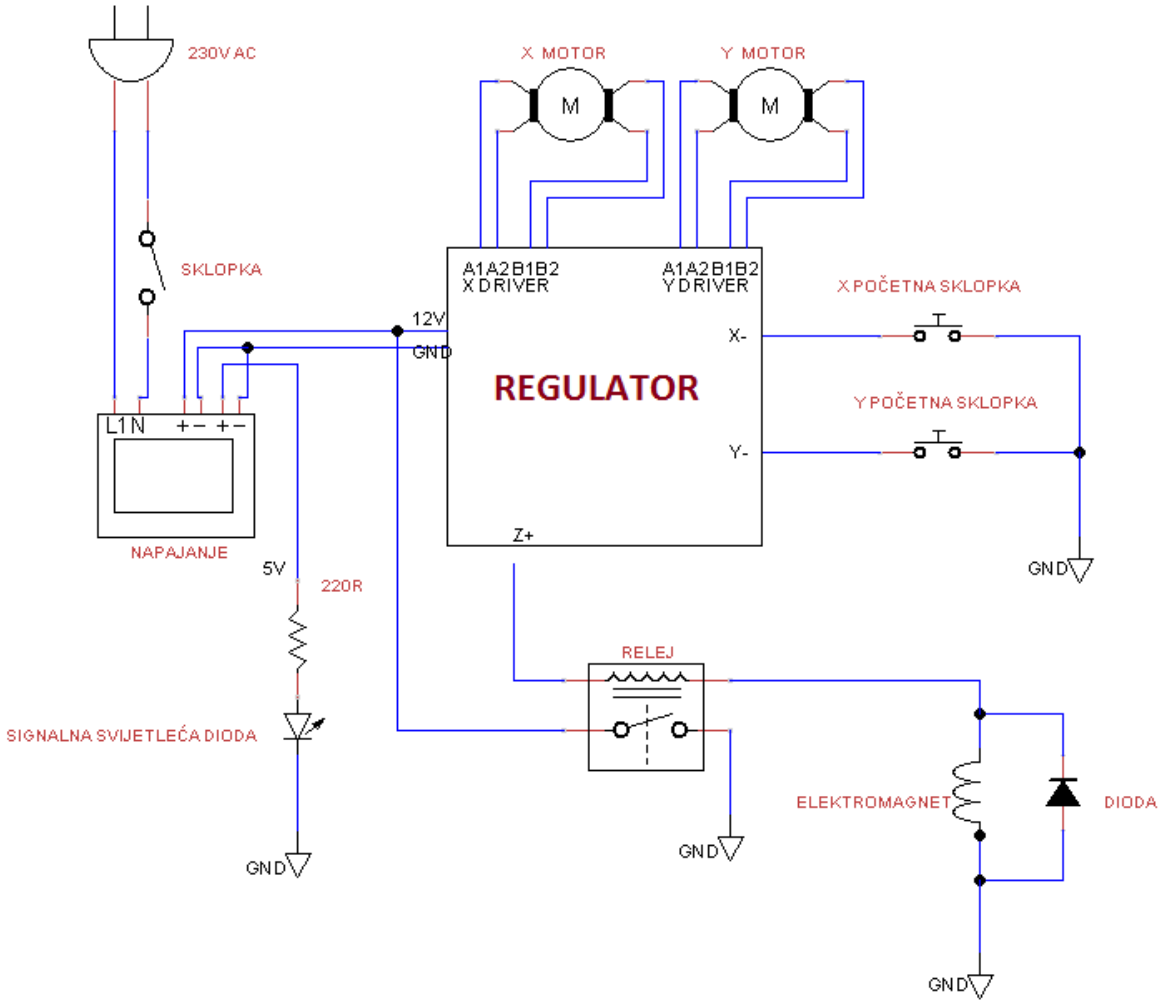
Elektromagnet je u principu električna zavojnica kroz koju može teći električna struja. Što je više namotaja zavojnice, to je magnetsko polje uz jednaku električnu struju jače, a ono se pogotovo silno poveća ako se u unutrašnjost zavojnice umetne jezgra od željeza ili čelika velike magnetske permeabilnosti. Kao i trajni magnet, elektromagnet privlači željezne predmete koji se nalaze u njegovoj blizini, a jednako tako ima najmanje dva magnetska pola, prema [9].



Sl. 2.11. – *Prikaz smjera magnetskog polja, presjeka elektromagneta, te primjerak elektromagneta.*

3. REALIZACIJA SUSTAVA

3.1. Shema sustava



Sl. 3.1. – *Pojednostavljena shema sustava*

Na slici 3.1. prikazana je pojednostavljena shema sustava. Za napajanje sustava koristimo računalno napajanje. Da bi izračunali potrebnu snagu napajanja u ovakvom sustavu potrebno je provesti mjerenja ili pogledati informacije za motore i elektromagnet pošto su to najveći potrošači. Pošto su korištene komponente skupljene iz starih uređaja nemamo dostupne informacije pa se mora mjerenjem odrediti potrebna struja. Koristi se bipolarni koračni motor sa četiri izvoda. Mjerenje snage motora vršimo tako da pronađemo odgovarajuće izvode za namotaje te mjerimo otpor na jednom namotaju (otpor je jednak na oba namotaja motora) te korištenjem Ohmovog zakona (3-1) izračunavamo jakost struje potrebne na tom namotaju i množimo ju s dva jer su dva namotaja

u motoru. Npr. ako za motor čiji je otpor na namotaju 10Ω pri naponu od 12V jakost struje će biti 1.2A, te će sveukupna potrebna jakost struje biti 2.4A.

Ohmov zakon glasi:

$$U = I * R \tag{3-1}$$

gdje je:

- U – Napon, V
- I – Jakost struje, A
- R – Otpor, Ω

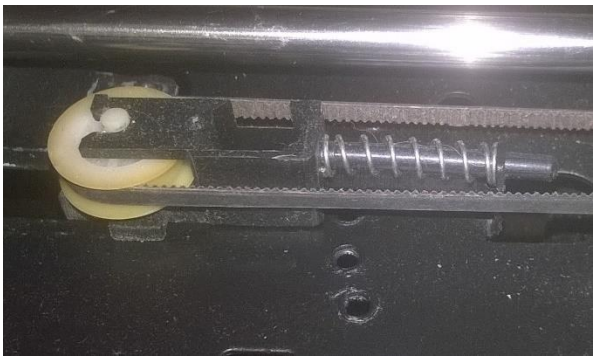
Na isti način provodimo mjerenje na elektromagnetu te zbrajanjem svih dobivenih vrijednosti dobivamo potrebnu snagu napajanja na 12V istosmjerne struje. Spajamo napajanje na regulator za kontrolu motora na tome predviđeno mjesto, zatim odabiremo potrebne drivere koračnih motora prema njihovoj maksimalnoj izlaznoj struji, jer što je manja jakost struje na motoru to je on slabiji. U maketi je korištena Pololu A4988 platforma sa driverom koja nam omogućava regulaciju izlazne snage drivera i time ograničava mogućnost kvara motora ili drivera. Nadalje koristimo relej za aktivaciju magneta, te paralelno sa magnetom spajamo diodu radi samopražnjenja. Zbog slabe snage magneta koristimo figurice koje imaju permanentni magnet. Da bi elektromagnet mogao privlačiti figurice mora se obratiti pozornost na polaritet permanentnih magneta na figuricama, svi magneti moraju biti isto polarizirani inače će elektromagnet neke privlačiti a neke odbijati. I na kraju spajamo završne sklopke koje omogućavaju sustavu da pronađe početnu poziciju.

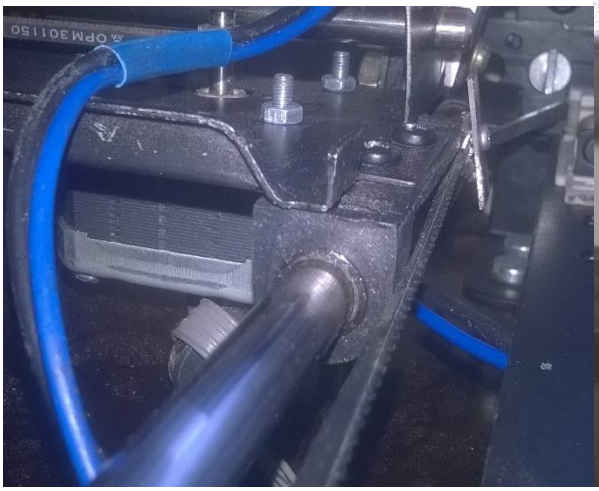
3.2. Izrada makete sustava

Maketa sustava izrađena je od komponenata starog printera. Sastoji se od dvije platforme od kojih je jedna pomična i nosi magnet(Sl. 3.2.).

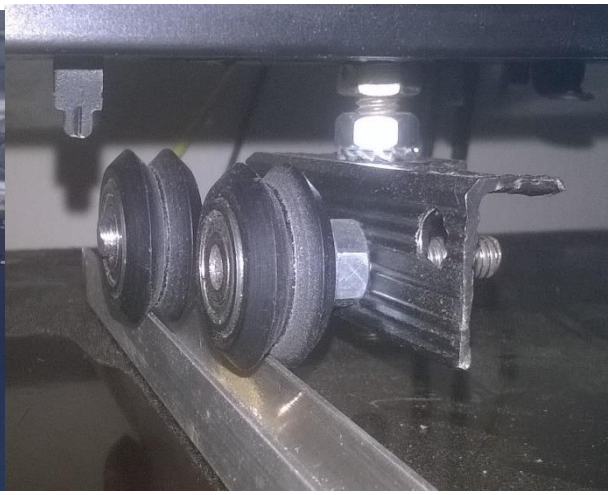
Sl. 3.2. – *Pomična os s postoljem za magnet i magnetom.*

Osi za pomicanje magneta sastoje se od koračnih motora, remenja i remenica od koje je jedna na svakoj osi povezana na oprugu radi održavanja napetosti na remenu(Sl. 3.3). Za vođenje y-osi koristimo sustav osovine i linearnog ležaja (Sl. 3.4.) na jednoj strani i dva gumena kotačića s ležajevima na U aluminijskom profilu(Sl. 3.5.).





Sl. 3.4. – *Linearni ležaj i osovina.*



Sl. 3.5. – *Konstrukcija s kotačićima na aluminijskom ležaju.*

Šahovska tabla je izrađena je od komada drvene plohe debele tri milimetra radi što manjeg smanjenja efikasnosti elektromagneta. Izrađena maketa prikazana je na slici 3.6.

Sl. 3.6. – *Maketa šahovske table sa automatiziranim pomicanjem figurica*

3.3. Programsko rješenje sustava

Konfiguracija postavki grbl-a

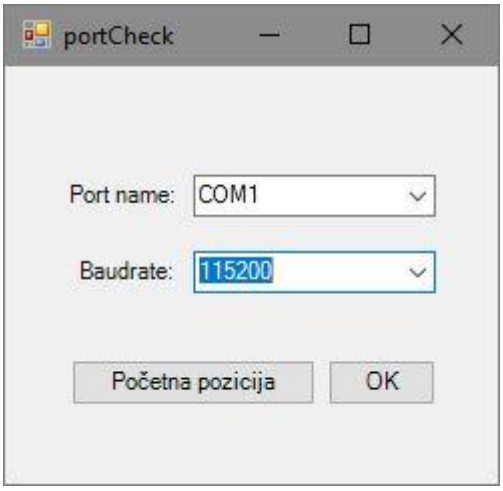
Da bi maketa uspješno odradila precizan pomak potrebno je postaviti parametre sustava tako da odgovaraju korištenim motorima i remenicama. U tablici 3.1. prikazane su neki od osnovnih parametara potrebnih za pravilno funkcioniranje makete.

Tablica 3.1. – *Osnovni parametri grbl sučelja.*

Oznaka i vrijednost	Opis
$3=6	Maska smjera vrtnje, maska:00000110
$22=1	Postavljanje početne pozicije pri paljenju, bool
$23=2	Maska smjera vrtnje tijekom postavljanja početne pozicije
$100=450.000	Veličina pomaka X motora, korak/mm
$101=150.000	Veličina pomaka Y motora, korak/mm

Izmjena parametara vrši se slanjem naredbe u obliku $x=vrijednost, gdje x označava broj naredbe a vrijednost je željena vrijednost izražena u unaprijed određenim mjernim jedinicama.

Kako bi imali mogućnost povezivanja makete sa igricom na računalu se koristi „Quick chess“ računalna igrica šah otvorenog koda. Aplikacija je pisana u programskom jeziku C#. Sve izmjene i dopune izvedene su pomoću programskog sustava Visual Studio 2013.



Sl. 3.6. – *Forma za postavljanje komunikacije i makete.*

Pošto je aplikacija otvorenog koda imamo mogućnost izmjene i nadopune istog. Pa tako postavljamo da se pri otvaranju aplikacije prikazuje forma sa izbornicima za odabir serijskog porta i brzine prijenosa, gumbom za postavljanje početne pozicije makete (Homing cycle), i gumbom OK kojim se pokreće sama igrica(Sl. 3.6.).

Pokrećemo novu igru, kada se odigra potez računalo vrši provjeru, ako je potez valjan potez se izvodi, u isto vrijeme koordinate početnog i odredišnog polja poteza kao i koordinate odredišnog

polja se šalju u konstruktor klase Command koja onda kreira novi objekt klase Path koji, pri izračunavanju putanje prolaženjem kroz nekoliko if petlji, sadrži potreban G-kod koji se onda formira i prosljeđuje serijskim portom maketi te se aktivira dijalog čekanja (Sl. 3.7.) dok maketa ne odradi pomak.

Sl. 3.7. *– Dijalog čekanja.*

Dio algoritma za izrađivanje putanje poteza.

```
...
 if (magnetX < endX)
                {
                    if (magnetY < endY)
                    {
                        if (magnetAdjust == 0)
                        {
                            magnetAdjust = 1;
                            magnetY += 0.5;
                            magnetX += 0.5;
                            setAdjust(magnetX, magnetY);
                            Adjust = "HDR";
                            x = 1;
                            continue;
                        }
                        dir[1] = 'R';
                        magnetY++;
                    }
                    else if (magnetY > endY)
                    {
                        if (magnetAdjust == 0)
                        {
                            magnetAdjust = 1;
                            magnetX += 0.5;
                            magnetY -= 0.5;
                            setAdjust(magnetX, magnetY);
                            Adjust = "HDL";
                            x = 1;
                            continue;
                        }
...
```

Konstantnom korekcijom varijabli magnetX i magnetY kontrolira se tok te se po rezultatu usporedbe tih varijabli i koordinata odredišta određuju točke po kojima se kreira putanja nizom naredbi. Varijabla magnetAdjust služi za početno pomicanje magneta sa središta polja na rub bliži odredištu jer se pomak figurica odvija po rubovima polja da se izbjegnu ostale figurice.

4. ZAKLJUČAK

U ovom završnom radu izrađena je šahovska tabla s automatiziranim pomicanjem figura. Tehnologije koje su duže vrijeme poznate čovjeku, te su kroz vrijeme usavršavanjem i pojednostavljivanjem postale dostupne svakome, omogućile su izradu sustava automatizacije za pomicanje šahovskih figura. Poznavanjem osnova elektrotehnike uspješno su implementirane i povezane različite komponente u jednu cjelinu. Arduino razvojna platforma omogućila je vrlo jednostavno testiranje i upravljanje sustavom za pomicanje figura. Korištenjem windows aplikacije (igrice šah) otvorenog koda uz male izmjene koda uspješno su ugrađene naredbe za slanje podataka (odigranih poteza) arduino platformi. Željom i znanjem sustav, zbog svoje modularnosti, je moguće dalje razvijati i unapređivati sa novim tehnologijama i funkcionalnostima.

Treba istaknuti još da kvaliteta izrađenog projekta ovisi o korištenim materijalima i komponentama. Pošto su korištene reciklirane komponente nisu postignuti očekivani rezultati, te se kupnjom novih, kvalitetnijih komponenata može poboljšati i kvaliteta samoga projekta.

LITERATURA

[1] – CAD, Wikipedia – https://hr.wikipedia.org/wiki/CAD, lipanj, 2016.

[2] – Električni strojevi, Wikipedia – https://hr.wikipedia.org/wiki/Električni_strojevi, lipanj, 2016.

[3] – Koračni motori, FER, Zavod za automatiku i procesno računarstvo – https://www.fer.unizg.hr/_download/repository/EAP_VIII_dio_KM%5B1%5D.pdf, rujan, 2016.

[4] – Arduino, Wikipedia – https://hr.wikipedia.org/wiki/Arduino, lipanj, 2016.

[5] – GPS, Garmin – http://www8.garmin.com/aboutGPS/, lipanj, 2016.

[6] – USB, Wikipedia – https://hr.wikipedia.org/wiki/USB, lipanj, 2016.

[7] – Arduino, Vodić – https://www.arduino.cc/en/Guide/Environment, lipanj, 2016.

[8] – Otvoreni kod, Wikipedia – https://hr.wikipedia.org/wiki/Otvoreni_kod , lipanj, 2016.

[9] – Elektromagnet, Wikipedia – https://hr.wikipedia.org/wiki/Elektromagnet, lipanj, 2016.

SAŽETAK

Naslov: Šahovska tabla s automatiziranim pomicanjem figura

U ovom radu prikazana je izrada šahovske table s automatiziranim pomicanjem figura. U prvom poglavlju navedene su korištene tehnologije i komponente, te su prikazane njihove funkcionalnosti. U drugom poglavlju se nalazi postupak realizacije projekta, njegova izrada i slike finalnog proizvoda zajedno sa shemama i ostalim popratnim materijalom.

Ključne riječi: šah, šahovska tabla, automatizacija, arduino, CNC.

ABSTRACT

Title: Chessboard with automated moving of chess pieces.

In this paper is presented development of a chessboard with automated moving of chess pieces. First chapter lists the used tehnology and components, and shows their functionality. In the second chapter is the process of realization of the project, its development and the images of the final product together with schemes and other accomanying material.

Key words: chess, chessboard, automation, arduino, CNC.

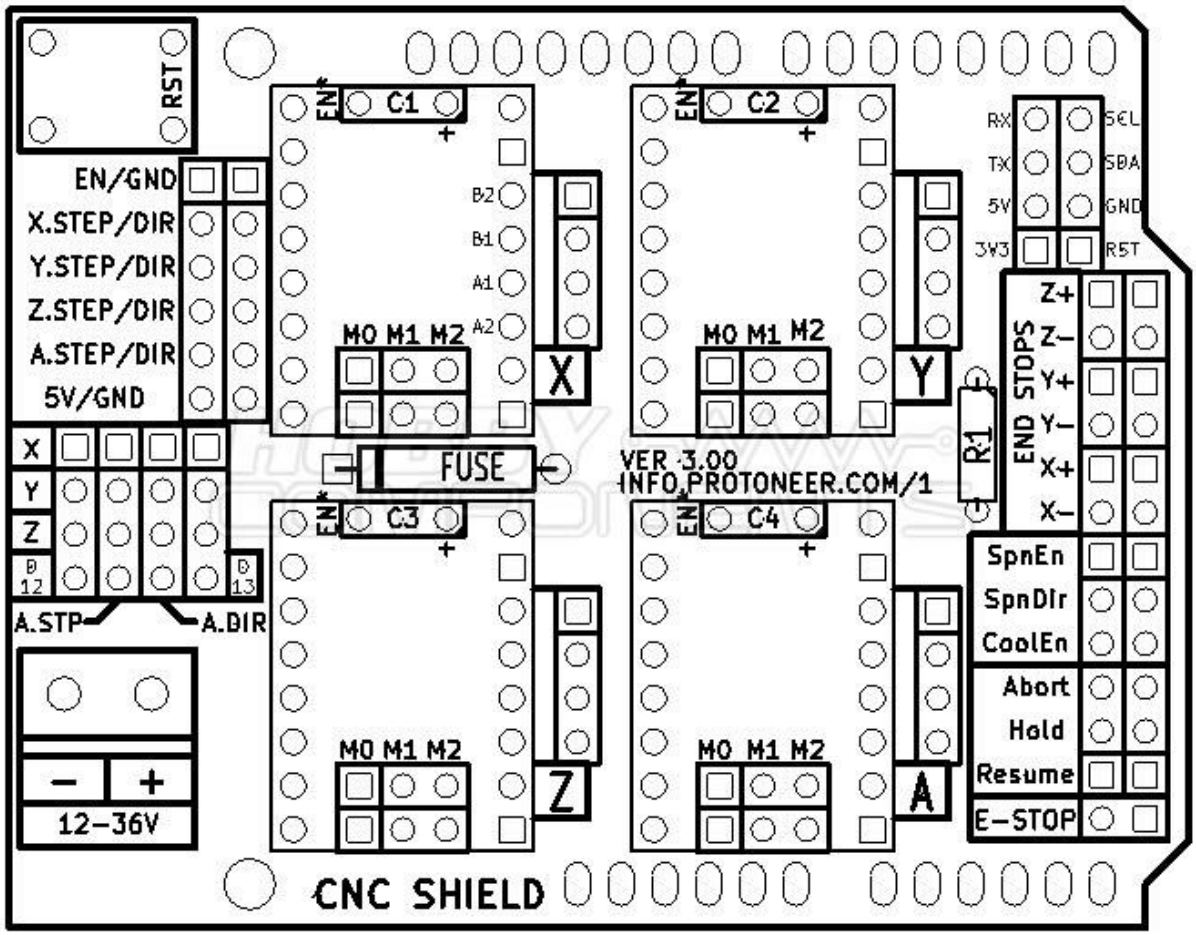
ŽIVOTOPIS

Ivan Klasić rođen 8.4.1993.g. u Osijeku. Nakon završetka osnovne škole OŠ Josipovac u Josipovcu, upisuje Poljoprivrednu i veterinarsku školu u Osijeku te maturira 2011.g. 2012.g. upisuje stručni studij na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija, smjer informatika. Nakon četiri godine studija položio je sve ispite i sada se nalazi pred obranom svog završnog rada.

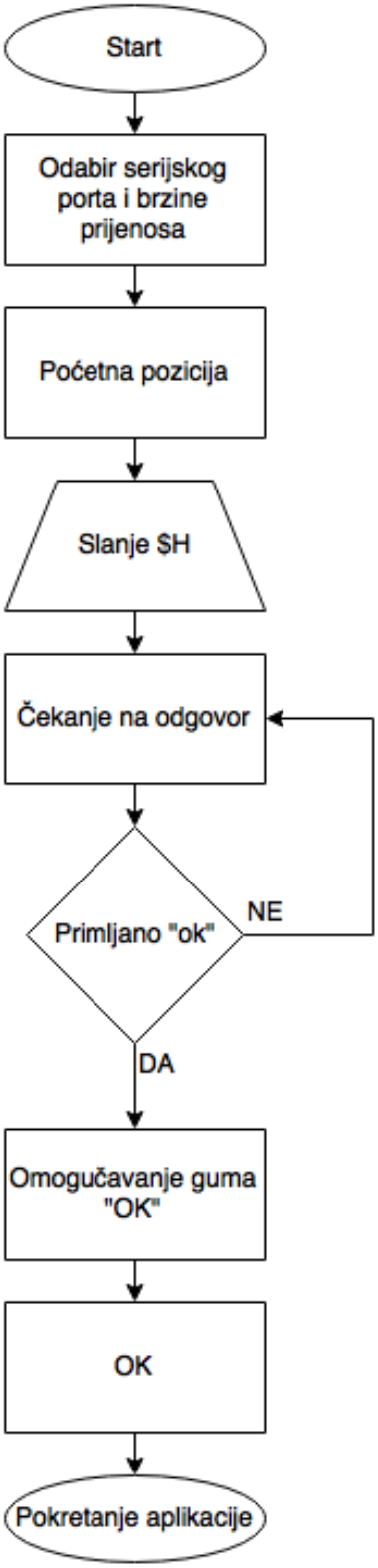
PRILOG A. TEHNIČKE ZNAČAJKE ARDUINO UNO PLATFORME

Mikroupravljač	ATmega328P
Radni napon	5V
Ulazni napon(preporučeno)	7-12V
Ulazni napon (limit)	6-20V
Digitalni UI pinovi	14(6 PWM izlaza)
Analogni ulazni pinovi	6
DC struja po UI pinu	20mA
DC struja za 3.3V pin	50mA
Memorija	32 KB
SRAM	2 KB
EEPROM	1 KB
Brzina mikroupravljača	16 MHz
Duljina	68.6 mm
Širina	53.4 mm
Težina	25 g

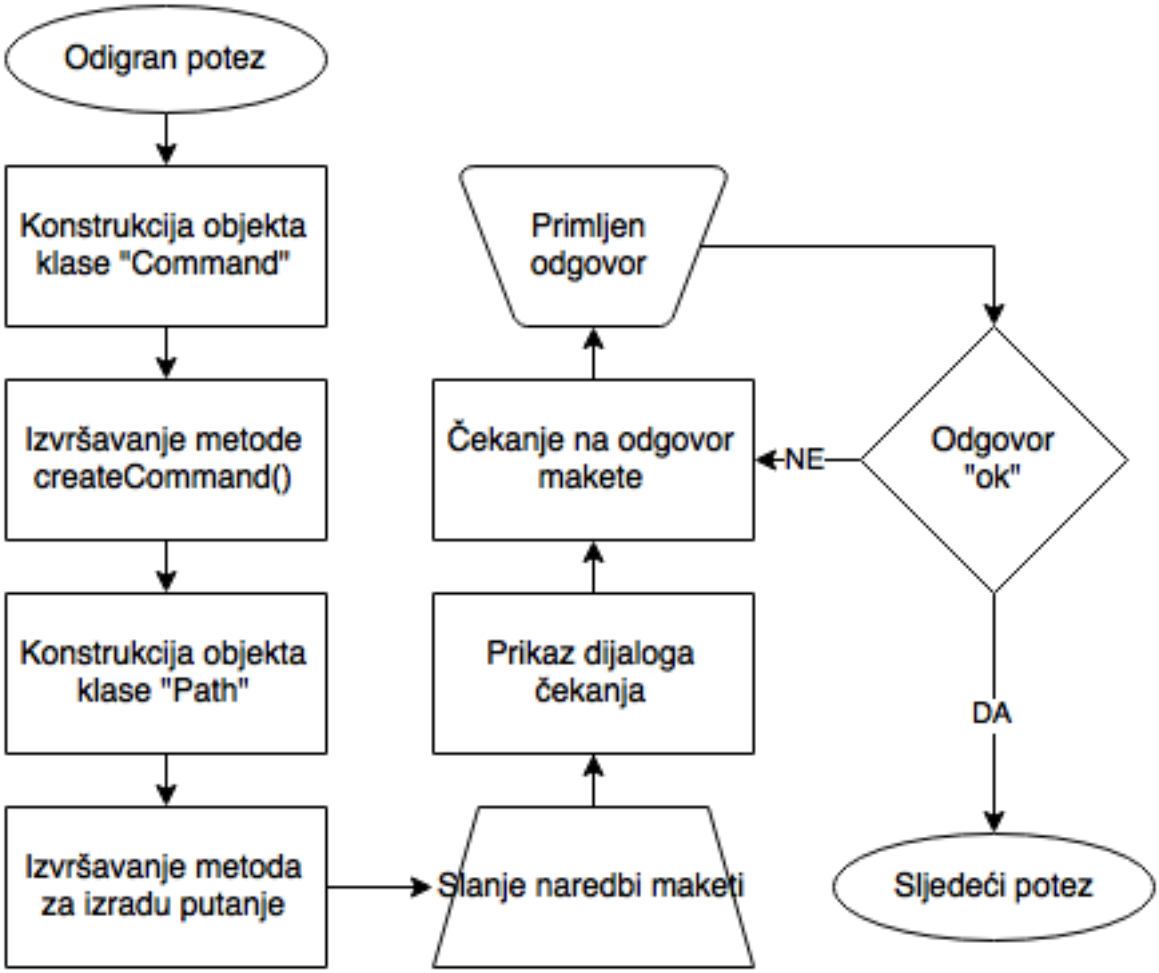
PRILOG B. TLOCRT REGULATORA ZA KONTROLU MOTORA



PRILOG C. DIJAGRAM TOKA FORME ZA POSTAVLJANJE POČETNE POZICIJE



PRILOG D. DIJAGRAM TOKA KOMUNIKACIJE IZMEĐU APLIKACIJE I MAKETE



PRILOG E. KLASA „PATH“ ZA ODREĐIVANJE PUTANJE

```
using System;
using System.Collections.Generic;
using System.Text;
using ChessLibrary;
/*
            CompletePath[0][0] = ToStart;
            CompletePath[1][0] = "M05";
            CompletePath[2][0] = grblAdjust;
            CompletePath[3][i] = grblToEnd[i];
            CompletePath[4][0] = grblDestination;
            CompletePath[5][0] = "M03";
*/
namespace PathCalculation
{
    class Path : Game
    {
        public string[][] CompletePath = new string[6][];
        private string ToStart;
        private string[] ToEnd = new string[50];
        private string[] grblToEnd = new string[50];
        private string Adjust;
        private string grblAdjust;
        private string Destination;
        private string grblDestination;
        private int startX;
        private int startY;
        private double endX;
        private double endY;
        private double magnetX;
        private double magnetY;
        private int magnetAdjust;

        public Path(int SX, int SY, int EX, int EY, int MX, int MY)
        {
            startX = SX;
            startY = SY;
            magnetX = MX;
            magnetY = MY;
            endX = EX - 0.5;
            endY = EY - 0.5;
            magnetAdjust = 0;

        }
        public void PathToStart()
        {
            ToStart = String.Format("G00 X{0} Y{1} \r\n", startX, startY);
        }
        public int PathToEnd()
        {
            char[] dir = new char[2] { 'S', 'S' };
            int i = 0;
            int x = 0;
            int MX;
            int MY;

            while (true)
```

```
{

    MX = Convert.ToInt32(magnetX - 0.5);
    MY = Convert.ToInt32(magnetY - 0.5);


    if (x == 1)
    {
        string check = checkCell(Adjust, MX, MY);
        if (check != "Clear")
        {
            ToEnd[i] = check;
            setToEnd(magnetX, magnetY, i);
            if (i != 0 && ToEnd[i - 1] == ToEnd[i])
            {
                grblToEnd[i - 1] = grblToEnd[i];
                i--;
                continue;
            }
            i++;
        }
        x = 0;
    }
    if (i > 0)
    {
        string check1 = checkCell(ToEnd[i - 1], MX, MY);
        if (check1 != "Clear")
        {
            ToEnd[i] = check1;
            setToEnd(magnetX, magnetY, i);
            i++;
        }

    }
    if (magnetX < endX)
    {
        if (magnetY < endY)
        {
            if (magnetAdjust == 0)
            {
                magnetAdjust = 1;
                magnetY += 0.5;
                magnetX += 0.5;
                setAdjust(magnetX, magnetY);
                Adjust = "HDR";
                x = 1;
                continue;
            }
            dir[1] = 'R';
            magnetY++;
        }
        else if (magnetY > endY)
        {
            if (magnetAdjust == 0)
            {
                magnetAdjust = 1;
                magnetX += 0.5;
                magnetY -= 0.5;
                setAdjust(magnetX, magnetY);
```

```
                Adjust = "HDL";
                x = 1;
                continue;
            }
            if (magnetY - 1 == endY)
            {
                dir[0] = 'D';
                dir[1] = 'S';
                magnetX++;
                setToEnd(magnetX, magnetY, i);
                ToEnd[i] = new string(dir);
                if (i != 0 && ToEnd[i - 1] == ToEnd[i])
                {
                    grblToEnd[i - 1] = grblToEnd[i];
                    continue;
                }
                i++;
                continue;
            }

            dir[1] = 'L';
            magnetY--;
        }
        else if (magnetY == endY)
        {
            dir[1] = 'S';
        }
        dir[0] = 'D';
        magnetX++;
    }
    else if (magnetX > endX)
    {
        if (magnetY < endY)
        {
            if (magnetAdjust == 0)
            {
                magnetAdjust = 1;
                magnetX -= 0.5;
                magnetY += 0.5;
                setAdjust(magnetX, magnetY);
                Adjust = "HUR";
                x = 1;
                continue;
            }
            if (magnetX - 1 == endX && magnetY + 1 == endY)
            {
                dir[0] = 'S';
                dir[1] = 'R';
                magnetY++;
                setToEnd(magnetX, magnetY, i);
                ToEnd[i] = new string(dir);
                if (i != 0 && ToEnd[i - 1] == ToEnd[i])
                {
                    grblToEnd[i - 1] = grblToEnd[i];
                    continue;
                }
                i++;
                continue;
            }
```

```
    dir[1] = 'R';
    magnetY++;
}
else if (magnetY > endY)
{
    if (magnetAdjust == 0)
    {
        magnetAdjust = 1;
        magnetX -= 0.5;
        magnetY -= 0.5;
        setAdjust(magnetX, magnetY);
        Adjust = "HUL";
        x = 1;
        continue;
    }
    if (magnetX - 2 == endX && magnetY - 1 == endY)
    {
        dir[0] = 'U';
        dir[1] = 'S';
        magnetX--;
        setToEnd(magnetX, magnetY, i);
        ToEnd[i] = new string(dir);
        if (i != 0 && ToEnd[i - 1] == ToEnd[i])
        {
            grblToEnd[i - 1] = grblToEnd[i];
            continue;
        }
        i++;
        continue;
    }
    if (magnetX - 1 == endX && magnetY - 2 == endY)
    {
        dir[0] = 'S';
        dir[1] = 'L';
        magnetY--;
        setToEnd(magnetX, magnetY, i);
        ToEnd[i] = new string(dir);
        if (i != 0 && ToEnd[i - 1] == ToEnd[i])
        {
            grblToEnd[i - 1] = grblToEnd[i];
            continue;
        }
        i++;
        continue;
    }
    if (magnetX - 1 == endX && magnetY - 1 == endY)
    {
        Destination = "HUL";
        magnetX -= 0.5;
        magnetY -= 0.5;
        setDestination(magnetX, magnetY);
        break;
    }
    dir[1] = 'L';
    magnetY--;
}
else if (magnetY == endY)
{
    if (magnetAdjust == 0)
```

```
            {
                magnetAdjust = 1;
                magnetX -= 0.5;
                magnetY -= 0.5;
                setAdjust(magnetX, magnetY);
            }
            if (magnetX - 1 == endX && magnetY == endY)
            {
                Destination = "HUR";
                magnetX -= 0.5;
                magnetY += 0.5;
                setDestination(magnetX, magnetY);
                break;

            }
            dir[1] = 'S';
        }
        dir[0] = 'U';
        magnetX--;
    }
    else if (magnetX == endX)
    {
        if (magnetY < endY)
        {
            if (magnetAdjust == 0)
            {
                magnetAdjust = 1;
                magnetX -= 0.5;
                magnetY += 0.5;
                setAdjust(magnetX, magnetY);
                Adjust = "HUR";
                x = 1;
                continue;
            }
            dir[1] = 'R';
            magnetY++;
        }
        else if (magnetY > endY)
        {
            if (magnetAdjust == 0)
            {
                magnetAdjust = 1;
                magnetX -= 0.5;
                magnetY -= 0.5;
                setAdjust(magnetX, magnetY);
                Adjust = "HUL";
                x = 1;
                continue;
            }
            if (magnetX == endX && magnetY - 1 == endY)
            {
                Destination = "HDL";
                magnetX += 0.5;
                magnetY -= 0.5;
                setDestination(magnetX, magnetY);
                break;
            }
            dir[1] = 'L';
            magnetY--;
```

```
            }
            else if (magnetY == endY)
            {
                Destination = "HDR";
                magnetX += 0.5;
                magnetY += 0.5;
                setDestination(magnetX, magnetY);
                break;
            }
            dir[0] = 'S';
        }
        setToEnd(magnetX, magnetY, i);
        ToEnd[i] = new string(dir);
        if (i != 0 && ToEnd[i - 1] == ToEnd[i])
        {
            grblToEnd[i - 1] = grblToEnd[i];
            continue;
        }
        i++;
    }
    return i;
}
public string checkCell(string dir, int row, int col)
{
    ArduinoBoard board = copyBoard();
    bool result = false;
    string newDir = "SS";
    switch (dir)
    {
        case "UR":
        case "HUR":
            result = board.GetCell(row - 1, col);
            break;

        case "UL":
        case "HUL":
            result = board.GetCell(row - 1, col - 1);
            break;

        case "DR":
        case "HDR":
            if (board.GetCell(row, col) == true)
            {
                newDir = avoidCell(dir);
                return newDir;
            }
            break;

        case "DL":
        case "HDL":
            result = board.GetCell(row, col - 1);
            break;
        default:
            return "Clear";
    }
    if (result == true)
    {
        newDir = avoidCell(dir);
        return newDir;
```

```
        }
        else
        {
            return "Clear";
        }
    }
    public string avoidCell(string dir)
    {
        string newDir = "SS";
        switch (dir)
        {
            case "UR":
            case "UL":
            case "HUR":
            case "HUL":
                newDir = "US";
                magnetX--;
                break;

            case "DR":
            case "DL":
            case "HDR":
            case "HDL":
                newDir = "DS";
                magnetX++;
                break;
        }
        return newDir;
    }
    public void setAdjust(double X, double Y)
    {
        grblAdjust = String.Format("G01 X{0} Y{1} \r", X, Y);
    }
    public void setDestination(double X, double Y)
    {
        grblDestination = String.Format("G01 X{0} Y{1} \r", X, Y);
    }
    public void setToEnd(double X, double Y, int i)
    {
        grblToEnd[i] = String.Format("G01 X{0} Y{1} \r", X, Y);
    }
    public string[][] CreatePath()
    {
        PathToStart();
        int moveSteps = PathToEnd();

        CompletePath[0] = new string[1];
        CompletePath[1] = new string[1];
        CompletePath[2] = new string[1];
        CompletePath[3] = new string[moveSteps];
        CompletePath[4] = new string[1];
        CompletePath[5] = new string[1];
        CompletePath[0][0] = ToStart;
        CompletePath[1][0] = "M05 \r";
        CompletePath[2][0] = grblAdjust;
        for (int i = 0; i < moveSteps; i++)
        {
            CompletePath[3][i] = grblToEnd[i];
        }
```

```
            CompletePath[4][0] = grblDestination;
            CompletePath[5][0] = "M03 \r";

            return CompletePath;

        }


    }
}
```