Izrada i testiranje modela 3D pisača baziranog na Arduino platformi

Uršulin, Žarko

Undergraduate thesis / Završni rad

2015

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:769041

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-19**

Repository / Repozitorij:

University North Digital Repository

Sveučilište Sjever

Završni rad br.354/EL/2015

Izrada i testiranje modela 3D pisača baziranoga na Arduino platformi

Žarko Uršulin,4102/601

Varaždin, rujan 2015.godine

Sveučilište
Sjever

ELEKTROTEHNIKA

Završni rad br.354/EL/2015

Izrada i testiranje modela 3D pisača baziranoga na Arduino platformi

Student:

Uršulin Žarko,4102/601

Mentor:

mr.sc Matija Mikac dipl.ing

Predgovor

Zahvaljujem mentoru, mr.sc Matiji Mikacu na pomoći prilikom odabira teme završnog rada kao i usmjeravanju tijekom izrade završnog rada.

Zahvaljujem i svim profesorima koji su me pratili tijekom dosadašnjeg školovanja.

Sažetak

U ovome radu opisana je Arduino open-source platforma i navedena je njezina arhitektura te njezin hardverski i softverski dio. Opisana je i komunikacija između računala i Arduino pločice, cijela shema spajanja 3D pisača, princip rada ekstrudera, koračnih motora te kontrolne elektronike. Obrazložen je i program koji je bio izrađen i koji omogućuje kontroliranje 3D pisača. Ukratko opisana je funkcija komponente te način izrade 3D pisača.

Ključne riječi : Arduino, RAMPS, ekstruder, grijača ploča, koračni motor, grijač, termistor, 3D pisač

Popis korištenih kratica

U/I -Ulazno/Izlazni

USB - Universal Serial Bus

PWM - Pulse Width Modulation (pulsno-širinska modulacija)

Sadržaj:

1. UVOD **1**

2. ARDUINO MEGA2560 ..... **2**

2.1 FIZIČKE KARAKTERISTIKE 3

2.2 SPECIFIKACIJA 4

2.3 NAPAJANJE 4

2.4 ULAZNE I IZLAZNE JEDINICE 5

2.5 ARDUINO SOFTWARE 6

2.6 *MARLIN* FIRMWARE 7

2.7 SERIJSKA KOMUNIKACIJA SA RAČUNALOM 8

3. KORAČNI MOTORI **9**

3.1 BIPOLARNI NEMA 17 KORAČNI MOTOR 10

3.2 KARAKTERISTIKE KORAČNOG MOTORA NEMA17 11

4. KONTROLA ELEKTRONIKA **12**

5. EKSTRUDER **14**

6. GRIJAČA PLOČA **15**

7. LCD EKRAN **17**

8. PROGRAM ZA UPRAVLJANJE 3D PISAČEM **18**

8.1 PRIMJER ISPISANOMODELA 21

9. PROJEKT **23**

9.1 PROGRAMSKI KOD ZA UPRAVLJANJE 3D PISAČA 23

9.2 ISJEČAK KONFIGURACIJSKOG KODA NA ARDUINU 26

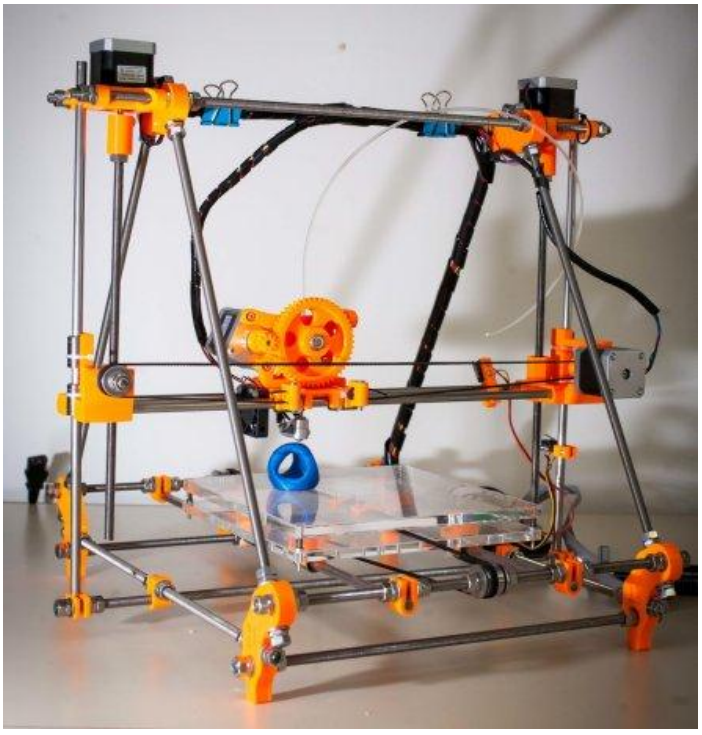
10. ZAKLJUČAK **28**

11. POPIS LITERATURE **30**

12. PRILOG **31**

1. Uvod

Budućnost sve više i više traži nove tehnologije i rješenja pa je tom namjerom izumljen 3D pisač. Prvi 3D pisač izumljen je 1981.g. na Institutu industrijskog istraživanja u Japanu. Izumio ga je Hideo Kodama [1]. Prvi 3D pisač koristio je materijal koji je služio da utvrđuje sliku dok današnji 3D pisači koriste malo naprednu tehnologiju. Oni koriste plastiku koja je dizajnirana samo za 3D pisače te ta plastika omogućuje visoku kvalitetu izrađenog dijela. Današnji 3D pisači koriste se kod izrade nekih vrlo preciznih industrijskih i komercijalnih dijelova. Sve više se ulaže u razvoj i poboljšanje 3D pisača zbog toga što su jeftini, njihovi izrađeni modeli su veoma kvalitetni te se koriste mnogo u industriji. Na Internetu postoji mnogo stranica na kojima je moguće skinuti gotove modele te ih samo ubaciti u program za kontroliranje 3D pisača te ispisati željene modele. Zbog ovih navedenih informacija o 3D pisaču te njegovih mogućnosti izabran je 3D pisač kao tema za završni rad te detaljno objašnjena njegova izrada, sve komponente korištene za tu izradu i njegovu funkciju te daljnji rad, tj.ispis.

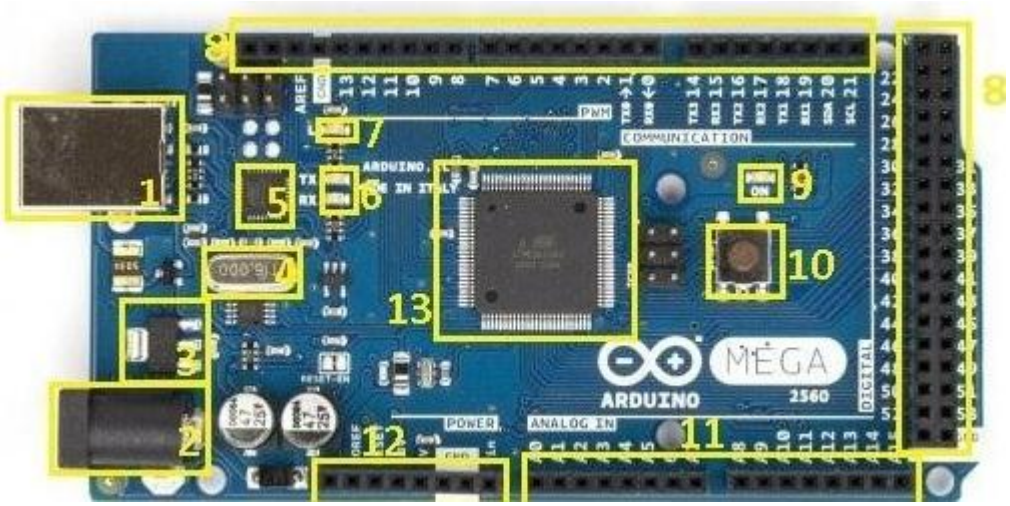


Slika 1.1 – izrađeni 3D pisač

2. ARDUINO MEGA2560

Arduino je open-source platforma za kreiranje elektroničkih prototipova baziranih na sklopovlju i programskom paketu koji je jednostavan za korištenje. Programiranje Arduino platforme se temelji na C jeziku. Arduino je baziran na 8-bitnom Atmel AVR mikrokontroleru ili 32-bitnom ARM procesoru [2]. Postoje različite verzije Arduina, a u ovom projektu je korišten Arduino MEGA2560 te modul RAMPS 1.4 koji služi za upravljanje koračnim motorima koji će kasnije biti opisani.

2.1 Fizičke karakteristike



Slika 2.1-glavne komponente

Popis najvažnijih komponenata Arduina MEGA2560:

1. USB konektor
2. Priključak za napajanje
3. Regulator napona
4. Kristalni oscilator
5. Atmega8U2 USB-to-TTL čip
6. TX RX diode za serijsku komunikaciju
7. Testna LED dioda
8. Digitalni pinovi (neki služe i za PWM modulaciju)
9. Dioda koja svijetli kada je Arduino uključen
10. RESET tipka
11. Analogni pinovi

12. Pinovi napajanja
13. Atmega2560

Na slici 2.1 su prikazani najbitniji dijelovi Arduino pločice. Arduino se povezuje s računalom putem USB konektora na pločici koji mu ujedno daje i napajanje. Na pločici postoji i čip USB–to-TTL koji nam služi za pretvaranje USB komunikacije u serijsku komunikaciju. RESET tipka nam služi da program koji se izvodi krene od početka. Tu su i digitalni te analogni pinovi koji nam služe za povezivanje elemenata na pločicu.

2.2 Specifikacija

Tablica 2.1 nam prikazuje tehničku specifikaciju Arduino mikrokontrolera:

MIKROKONTROLER	Atmega2560
RADNI NAPON	5V
ULAZNI NAPON(preporučeno)	7-12V
ULAZNI NAPON(limitirani)	6-20V
BROJ DIGITALNIH ULAZA/IZLAZA	54 (od kojih 12 pruža PWM
BROJ ANALOGNIH ULAZA	16
DC STRUJA	20mA
DC STRUJA ZA 3.3V PIN	50mA
FLASH MEMORIJA	256kB (od kojih 8kB
SRAM	8kB
EEPROM	4kB
OSCILATOR(frekvencija)	16MHz

Tablica 2.1 – tehnička specifikacija

2.3 Napajanje

Radni napon mikrokontrolera Atmega2560 je 5V, kao što je prikazano na tablici 2.1. Arduino se može napajati putem USB kabla ili preko vanjskog adaptera (baterija ili neki AC/DC pretvarač). Minimalni napon na kojem de Arduino raditi iznosi 7V, ako se Arduino priključi na napon manji od 7V postoji mogućnost da cijeli sustav neće funkcionirati., isto tako ako se priključi napon veći od 12V postoji mogućnost od pregaranja pinova.

Pinovi koji se mogu koristiti za napajanje:

- **VIN** – pin za ulazni napon
- **5V** – ovaj pin daje 5V koji se može napajati preko USB kabla ili vanjskog izvora (baterije)
- **3V3** – pin koji daje 3.3 V te struju jakosti 50mA
- **GND** – pin koji se koristi za uzemljivanje
- **IOREF**–pin koji kontrolira napon s kojim mikrokontroler izvršava zadaću

2.4 Ulazne i izlazne jedinice

Arduino pločica koja je korištena u ovom završnom radu ima na sebi 54 digitalna pina. Svaki od tih digitalnih pinova mogu se koristiti kao ulazni ili izlazni pomoću funkcija pinMode(), digitalWrite() i digitalRead() [11]. Digitalni pinovi funkcioniraju na radnom naponu od 5V te svaki digitalni pin može podnijeti struju jakosti 40mA.

Digitalni pinovi sa specijaliziranim funkcijama:

- **SERIAL** – na ovom Arduinu postoje 4 para serijski pinova, Serial1: 0 (RX) i 1 (TX), Serial2: 19 (RX) i 18 (TX), Serial3: 17 (RX) i 16 (TX), Serial4: 15 (RX) i 14 (TX)
    - služe za primanje (RX) te slanje (TX) serijskih podataka, a povezani su sa Atmega2560 USB-to-TTL serijskim čipom koji omogućava serijsku komunikaciju
- **PWM** – 2,3,4,5,6,7,8,9,10,11,12,13,44,45,46 ovih 15 pinova podržavaju PWM izlaze odnosno „pulsno-širinsko-modulaciju“ koja daje mogućnost da se iz digitalnih pinova koji daju 0 ili 5 V dobije bilo koja vrijednost između 0 i 5V.
- **VANJSKI PREKIDI** – 2,3,18,19,20,21 ovi pinovi se mogu programirati tako da omogućuju prekid prilikom nailaska na nisku vrijednost, rastući ili padajući brid
- **SPI,MOSI, MISO, SCK** – 50,51,52,53 pinovi omogućavaju SPI (SERIJSKO- PERIFERNO SUČELJE) komunikaciju

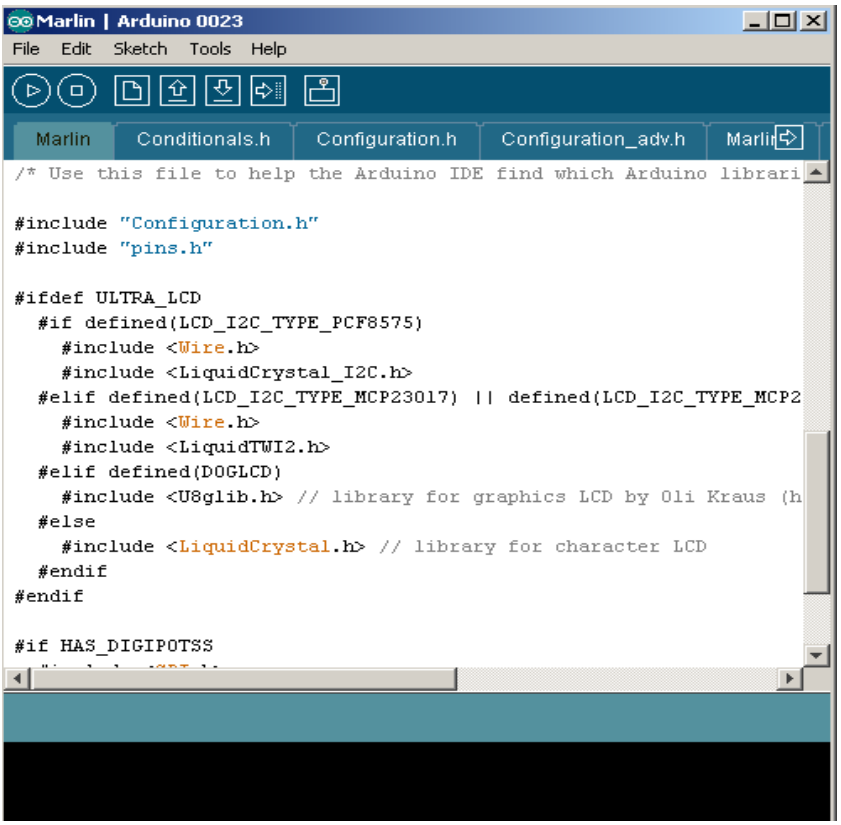
- **LED13**- na Arduino postoji LED dioda koja je spojena na digitalni pin 13 koja se uključuje ako je vrijednost HIGH, a isključuje ako je LOW

Osim 54 digitalnih pinova postoji i 16 analognih ulaza, a svaki od njih posjeduje 10 bita prostora. Ti analogni ulazi su konfigurirani da rade od 0-5 V, no tu je granicu moguće promijeniti koristeći AREF pin i naredbu analog Refrence. Analogni pinovi sa specijaliziranim funkcijama:

- **TWI**–20pin(SDA) I 21pin(SCL) podržavaju komunikaciju koristeći WIRE
- **AREF** – odabire napon za analogne ulaze
- **RESET** – ako ovaj pin dobije naredbu mikrokontroler

2.5 Arduino software

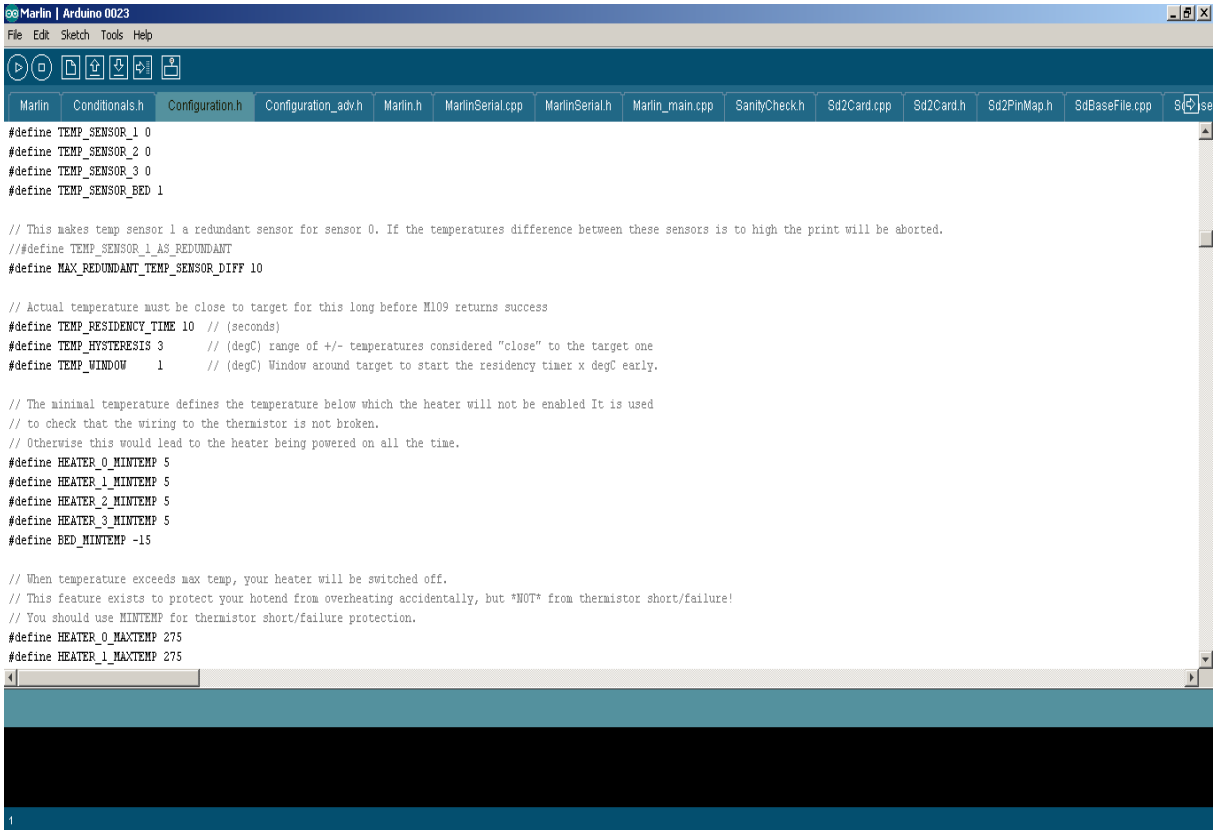
Arduino pločica se programira u C jeziku putem besplatnog IDE (Integrated Development Environment) softverskog paketa. Taj softverski paket je potpuno besplatan te se može preuzeti sa službenih stranica Arduina.[2] Prilikom završetka instalacije Arduino IDE programskog paketa potrebno je instalirati upravljačke programe za Arduino pločice. Programsko okruženje također nudi i gotove primjere jednostavnih programa, koji služe i za testiranje pločice ali pomaže i programeru da jednostavnije prilagodi program prema projektu kojeg izrađuje. Postoji velika prednost Arduina što preko USB kabla dobiva napajanje, pa je time omogućeno da se neposredno nakon „uploada“ programa u mikrokontroler može isprobati ili vidjeti je li se postigao željeni rezultat, pošto se petlja vrti od onog trenutka kad pločica dobije napajanje. Ukoliko se želi u nekom trenutnku program pokrenuti od početka nemora se isključivati napajanje i ponovno ga uključivati, ponovno pokretanje možemo izvršiti pritiskom tipke RESET. U projektu je korišten Arduino MEGA2560 te su upravljački programi automatski instalirani tijekom instalacije programskog paketa. Slika 2.2 prikazuje programsko okruženje Arduino programskog paketa.



Slika 2.2 – Arduino programsko okruženje

2.6 MARLIN firmware

U ovom projektu program koji je stavljen na Arduino pločicu zove se *MARLIN* jer taj firmware jedini pruža podršku LED zaslona koji je objašnjen kasnije u projektu te pruža više mogućnosti kontrole 3D pisača nego ostali [3]. Sam program ovisno o konfiguraciji može biti između 50kB do 200kb [3]. U ovom projektu je program zauzimao oko 152kB memorije Arduina. Može se prikazivati poruke na LED zaslonu te vidjeti sve potrebne informacije vezane za 3D pisač, čitati program sa SD kartice bez upotrebe računala. Tvorci toga softver su EvdZ te bkubicek. Taj program nam omogućava da sami postavimo postavke koje nam najbolje odgovaraju kao na primjer maksimalnu temperaturu grijača, brzinu kretanja grijača, postavki matične ploče, postavke senzora te mnoge druge postavke. Na slici 2.3 prikazan je konfiguracijski dio toga softvera u kojem se unose željene postavke.

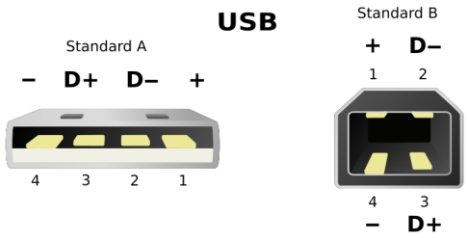


Slika 2.3 – konfiguracijski dio softvera

2.7 Serijska komunikacija sa računalom

Serijska komunikacija je tip komunikacije gdje prijamnik i predajnik razmjenjuju podatke putem jednog para signala odnosno način na koji se program koji je napisan na računalu pohranjuje na mikrokontroler. Isto tako moguće je informacije sa Arduina odnosno raznih senzora i sličnog prikazati na računalu. Komunikacija se odvija preko USB kabla koji se koristi i za programiranje mikrokontrolera. Prilikom serijske komunikacije kao što je već bilo spomenuto koriste se pinovi 0(RX) i 1(TX) te su oni zauzeti pa se njihovo korištenje ne preporučuje za ništa drugo osim kada je sigurno da se neće više mijenjati kod programa. Za serijsku komunikaciju na Arduino pločici zaslužan je FTDI čip koji omogućava da se Arduino i računalo sporazumijevaju.

Na slici 2.4 je prikazan USB port u standardu A te standard B



Pin1 V_{CC} (+5 V,CRVENA)

Pin2 Data– (BIJELA)

Pin3 Data+(ZELENA)

Pin4 Ground(CRNA)

Slika 2.4 – USB port

3. Koračni motori

Koračni motor je vrsta elektromotora koji se koristi u industriji robotike. Osim toga, oni su elektromehanički pretvornici energije, koji pulsnu električnu pobudu pretvaraju u koračni mehanički pomak (rotacijski ili translacijski). Premještaju poznati interval za svaki impuls snage. Te impulse snage omogućuje driver koračnog motora i smatra ih se jednim korakom. Kako svaki korak pomiče motor za nama već poznatu duljinu to ih čini korisnim uređajima za vrlo male pokrete što nam omogućava veliku preciznost kod 3D ispisa. Postoje tri tipa koračnih motora, a to su: peromanentnomagnetski, koračni motori s promjenjivom reluktancijom te hibridni koračni motori. Na slici 3.1 je prikazan koračni motor koji je korišten u projektu.

Slika 3.1 – Koračni motor

3.1 Bipolarni NEMA 17 koračni motor

Bipolarni motori imaju jedno namatanje po fazi. Struja mora biti obrnuta kako bi se mogao preokrenuti magnetski pol tako da strujni krug mora biti složeniji tipično s H-mostom. Efekti statičkog trenja koristeći H-most bili su zamijećeni s određenim pogonskim topologijama. Sjenčanjem step signala (koraka) na višoj frekvenciji motor će smanjiti efekt „statičkog trenja“. Budući da su namatanja bolje korištena, ona su moćnija od unipolarnog motora iste težine. To je zbog fizičkog prostora kojeg zauzimaju namatanja. Unipolarni motor ima duplo više žica u istome prostoru, ali samo pola ih se koristi u bilo kojem trenutku u vremenu [4]. Iako je bipolarnim koračnim motorom kompliciranо upravljati, obilje čipova za upravljanje znači daje lakše to postići.

Ova vrsta motora može biti spojena u nekoliko konfiguracija:

- UNIPOLARNI
- BIPOLARNI SA SERIJAMA NAMATANJA (daje veći induktivitet, ali manju struju po jednom namatanju)
- BIPOLARNI S PARALELNIM NAMATANJIMA (zahtijeva više struje, ali može izvesti bolje kako je induktivitet namatanja smanjen)
- BIPOLARNI S JEDNIM NAMATANJEM PO FAZI (ova metoda će pokrenuti motor na samo polovicu raspoloživih namatanja, što će smanjiti dostupnu nisku brzinu okretnih momenata, ali zahtjeva manje struje)

3.2 Karakteristike koračnog motora NEMA17

NEMA 17 koračni motor je koračni motor s dimenzijama 1,7 x 1,7 inča (43,2 x 43,2 mm). NEMA17 je veća i generalno teža od na primjer NEMA14, ali također znači da ima više prostora za stavljanje većeg okretnog momenta [6].

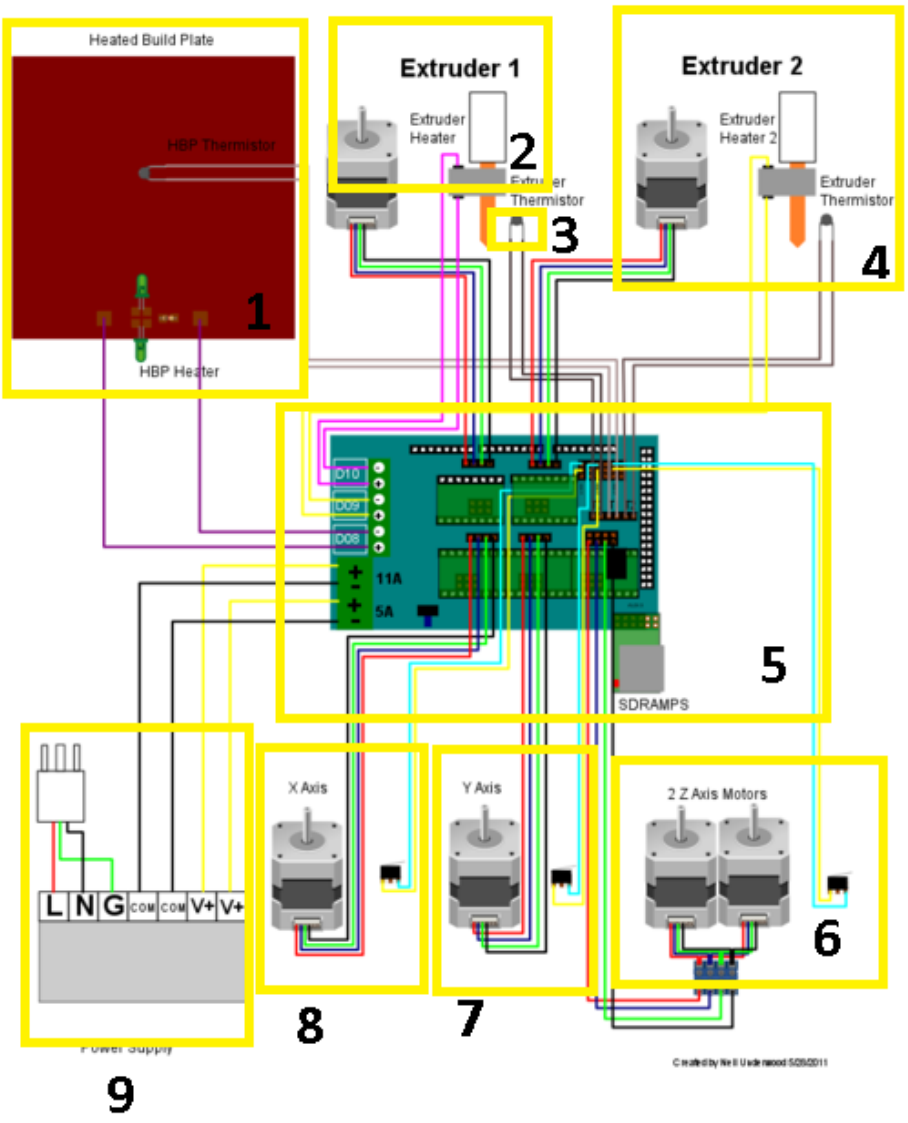
U tablici 3.1 prikazana su najbitnija svojstva koračnog motora koji su upotrijebljeni u projektu:

PARAMETAR	SPECIFIKACIJA
VELIČINA	NEMA 17 ILI 14 (PROTOTIP NEMA 14)
TIP	BIPLOARNI
OSOVINA	DUPLA OSOVINA – POTREBNO DA LAKŠE SAVLADA JEDAN KORAK
OKRETNI MOMENT	13.7NM-CM
OTPOR	MORA BITI IZNAD 6Ω

Tablica 3.1 – karakteristike koračnog motora NEMA17

4. Kontrola elektronika

3D pisač izrađen kao praktični dio ovog završnog rada koristi ranije objašnjeni Arduino te modul/dodatak RAMPS (v1.4) koji služi za upravljanje koračnim motorima. Dodatak RAMPS za rad koristi 12 V napajanje i najveću struju od 11 A. Na slici 4.1 se nalazi shema spajanja cijele elektronike 3D pisača sa shieldom te će kasnije biti objašnjeni.



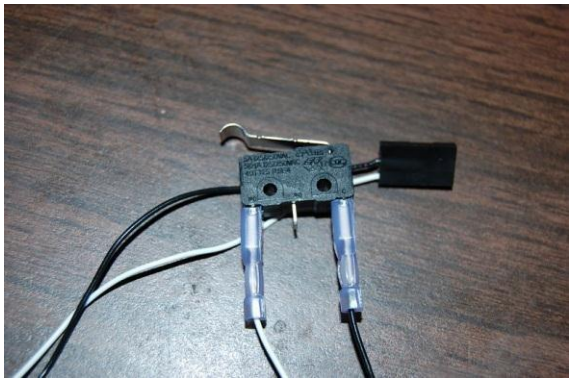
Slika 4.1 – shema spajanja elektronike

Popis komponenata te njihova spajanja na RAMPS 1.4:

1. Grijača ploča – služi da bi se ispisani dio postepeno hladio te da ne bi došlo do savijanja toga dijela
2. Ekstruder – koristi se da bi zagrijao plastiku na zadanu temperaturu te ju polagano istiskivao
3. Termistor ekstrudera – isključuje grijač na ekstruderu kada se dostigne zadana temperatura
4. Ekstruder broj 2 – može se koristiti kao dodatni ekstruder koji može istiskivati novi dio koji se mijenja u programu
5. Kontrolna elektronika RAMPS 1.4
6. Koračni motori koji se koriste za kontroliranje Z osi
7. Koračni motori koji se koriste za kontroliranje Y osi
8. Koračni motori koji se koriste za kontroliranje X osi
9. Napajanje elektronike

Na 3D pisaču postoje i 3 krajnje sklopke koje nam služe da isključe koračne motore kada ekstruder dostigne kraj svake osi. Njihova shema spajanja prikazana je na slici 4.1.

Slika 4.2 prikazuje sliku krajnjih sklopki, a tablica 4.1 prikazuje spajanje tih krajnjih sklopki.



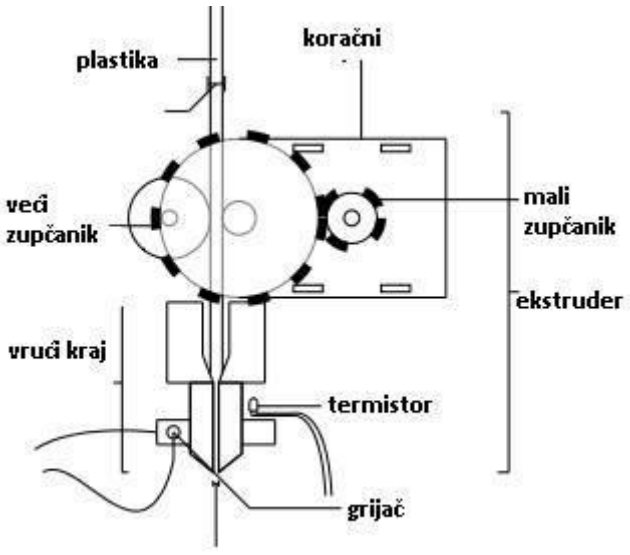
Slika 4.1 – krajnja sklopka

RAMPS krajnje sklopke	
SIG(S)	Bijela
GND(-)	Crna
VCC(+)	Crvena

Tablica 4.1 – spajanje krajnjih sklopki

5. Ekstruder

Istiskivač (eng. extruder, u tehničkoj dokumentaciji i ekstruder) je komponenta 3D pisača koja služi da zagrije plastiku te ju istiskuje ravnomjerno po površini. Sastoji se od koračnog motora koji je spojen na set zupčanika koji se pomalo okreće te svojim koracima uzima plastiku u grijač. Zasada postoje dvije vrste plastike: ABS i PLA. Glavna razlika između tih dviju plastika je u tvrdoći i kvaliteti ispisa. PLA plastika topi se na otprilike 110 do 130˚C , a ABS plastika na 210 do 230˚C. Plastika se može topiti i istiskivati na dva načina. Prvi način je da se topi u nekoj zatvorenoj posudi te se potopljena masa plastike prenosi pomoću plastične cijevi iz posude u ekstruder koji istiskuje tu masu na površinu. Drugi način je da se plastika topi na vrućem kraju koji se nalazi ispod ekstrudera, a taj kraj zagrijava grijač kojemu unaprijed definiramo zadanu temperaturu. Kada se postigne zadana temperatura, termistor koji se nalazi u grijaču isključuje grijač te po potrebi postepeno grije dokad ne dođe do zadane temperature. Kraj toga grijača nalazi se ventilator koji je spojen na kontrolnu elektroniku te nam služi da hladi grijač ako temperatura grijača prelazi zadanu vrijednost temperature. Upravo taj način je korišten u ovom projektu. Slika 5.1 prikazuje presjek ekstrudera.



Slika 5.1- presjek ekstrudera

6. Grijača ploča

Zagrijana platforma napravljena je da poboljša kvalitetu ispisa tako što pomaže u sprječavanju savijanja. Kako ekstruder topi plastiku ta se plastika pomalo počinje savijati. Kada se proces savijanja ne događa jednako kroz vrijeme ispisa, rezultat je deformirani dio. To savijanje se obično vidi kao da se uglovi podižu s izgrađene podloge. Ispis na grijaćoj ploči omogućuje da ispisani dio ostaje topao za vrijeme procesa ispisa i omogućuje još više savijanja plastike kako se hladi. Grijače ploče poboljšavaju kvalitetu ispisanog djela s materijalima kao što su ABS ili PLA. Prijašnja prihvatljiva metoda je bila da se prekrije PCB grijača ploča s trakom. Alternativna metoda, koja je postala popularna od siječnja 2015. godine je da se koristi papir od 3 mm debelog borosilikatnog stakla, koje se drži koristiti pričvršćivače.

Raspon temperature:

- ABS: 180-220ºC
- PLA : 50-70ºC

Staklo štiti ploču od udaraca i lako se može zamijeniti. Dimenzije ploče su identične dimenzijama MK1. Izolacija između grijače ploče i izolatora poboljšava vrijeme zagrijavanja te smanjuje potrošnju energije. Postoje rupe na grijaćoj ploči kroz koje se navode žice za napajanje. Treba provjeriti da žice koje se koriste dosta debele za 10 A i zalemiti za gornju stranu grijače ploče. LED diode nisu obavezne, ali ako se ipak odluči za korištenje LED dioda, mora se instalirati otpornik. Sa MK2, konvencionalni žičani dijelovi mogu se zamijeniti za montirane površinske dijelove. Grijača ploča može koristiti staklo kao izolator dobre kvalitete (npr. 3 mm debljine). Mada je jeftin i lako ga je za naći, staklo može biti i nepredvidljiv materijal te može puknuti kada ga se zagrije. Normalno staklo je najbolje koristiti na vrhu aluminijske ili bakrene ploče tako da se temperatura ravnomjerno širi. Ako su potrebne vrlo visoke temperature, postoji opcija da se koristi isti tip stakla kao i kod vrata od pećnice. Zbog slabe vodljivosti topline, temperatura na vrhu ploče u odnosu na donju stranu ploče može biti različita. Da bi išla struja kroz ploču, treba se koristiti PC PSU ili univerzalno napajanje koje može reproducirati najmanje 10A [5].

Elementi za zagrijavanje mogu biti: nikalna žica, razni otpornici za više temperature, a za niže ravni grijači. Nikalna žica je jeftinija i uzima manje prostora od raznih otpornika. U ovom projektu korišteno je napajanje snage 400 W te napona 12 V. Može dati 11 A bez pregaranja osigurača. Kontrolirani je preko Arduinovog 6. izlaza koji može dati PWM modelirani signal od 5 V. Dioda D1 je zaštićena od mogućih variranja u voltaži. Što se tiče TSIC101, on je opskrbljen od Arduina i radi na 5 V. Njegov izlaz je linearan i iznosi između 0 i 1 V. 0 V za 50ºC i 1 V za 150ºC. Pošto TSIC101 ima izlaz od 5mV za svaki 1ºC. Arduino može mjeriti pomak temperature od 0.2 stupnjeva. Neće se dopustiti da TSIC101 ide preko 140ºC za sigurnost budući da je 150ºC maksimum. Slika 6.1 prikazuje grijaču ploču korištenu u projektu.

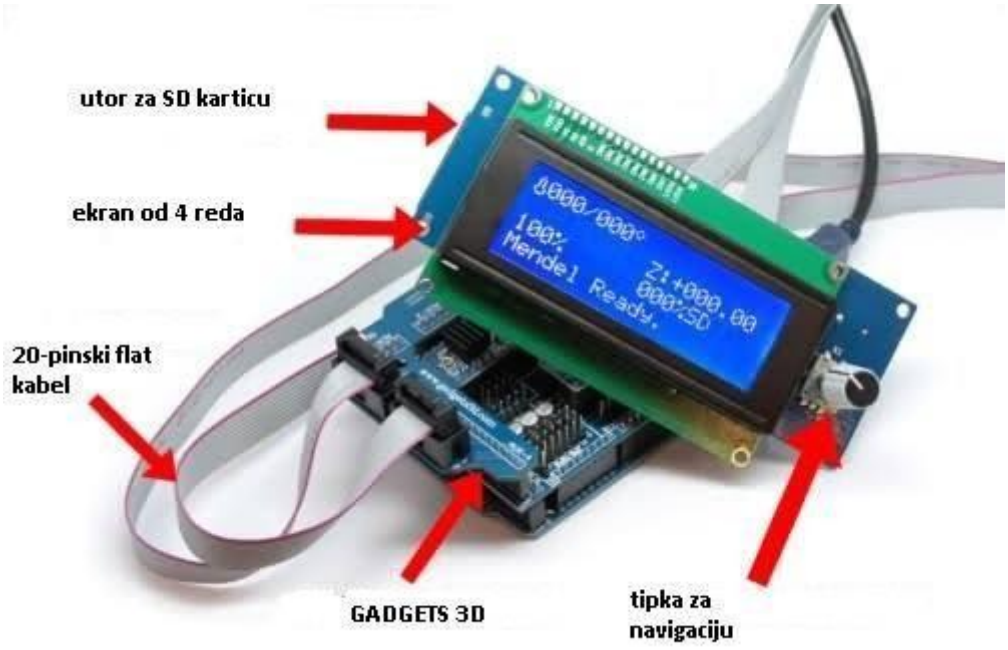
Slika 6.1 – grijača ploča

7. LCD ekran

LCD ekran korišten u projektu služi za upravljanje 3D pisača bez upotrebe osobnog računala. Sastoji se od: upravljačkog ekrana, tipke za navigiranje, zujalice i podrške za SD karticu. On može učitati program sa SD kartice te ga pretvoriti u G-kod i taj program ispisati. Na njegovom ekranu nalaze se sve informacije koje su potrebne kao što su: temperatura grijača, temperatura grijače ploče, pozicija svih triju osi, tekst projekta te razne druge informacije [9]. Spaja se na poseban konektor kojega su izradili ljudi iz tvrtke Gadgets 3D.

U ovom projektu korišten je *MARLIN* firmware upravo jer samo on daje mogućnost priključivanja tog LCD ekrana. Kada se LCD ekran priključi na 3D pisač, potrebno je napraviti neke preinake na softveru. U konfiguracijskom dijelu potrebno je pronaći liniju # G3_PANEL te ispred nje izbrisati samo dvije kose crte. Softver će sve dalje uraditi sam. Kada se ne bi izbrisale te dvije crte, LCD panel ne bi radio, tj. uvijek bi se prikazivala poruka da se panel programira.

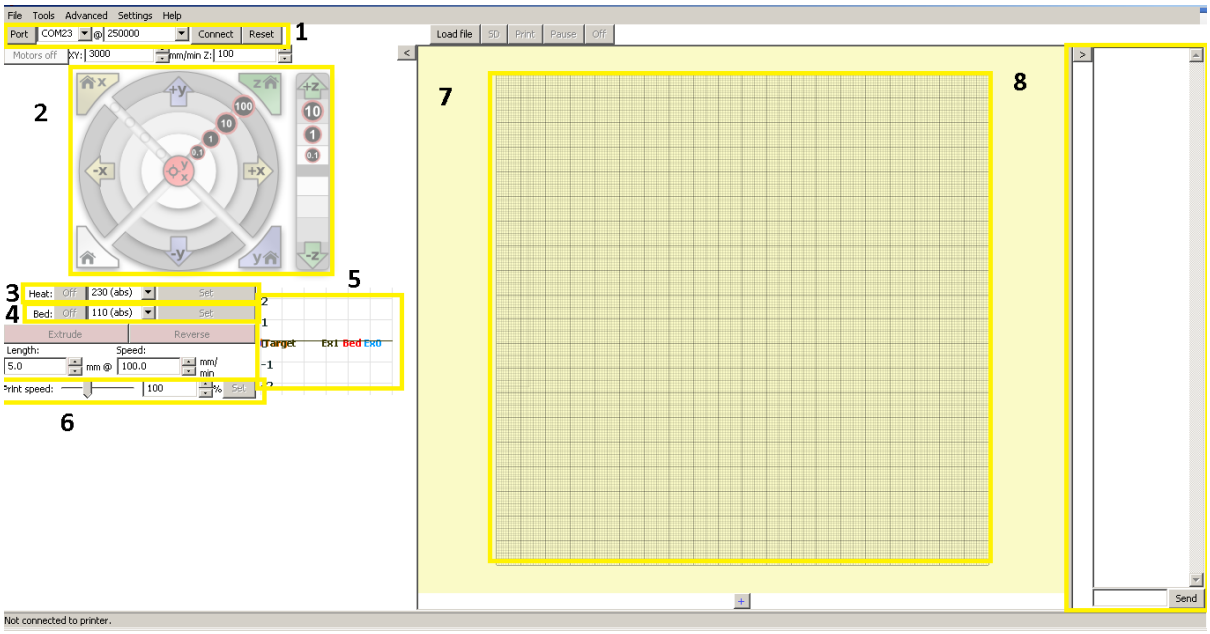
Na slici 7.1 nalazi se popis dijelova LCD panela.



Slika 7.1 - dijelovi LCD panel

8. Program za upravljanje 3D pisačem

Program za kontroliranje 3D pisača korišten u projektu naziva se Pronterface. Radi na Windows i Linux platformi a u radu je korišten na Windows platformi. Napisan je u programskom jeziku Python. U programu se klikne na gumb Load File te se s računala odabere model koji je prethodno izrađen u nekom programu za 3D modeliranje te program automatski taj model pretvara u G-kod. To je kod koji je razumljiv kontrolnoj elektronici samog 3D pisača tj. za numeričko upravljanje. G-kod je programski jezik u kojemu korisnik „govori“ stroju kako da nešto izradi [8]. Primjer G-koda: G1 X90.6 Y13.8 E22.4 (kreći se 90.mm po X osi i 13.8 mm po Y osi te ispiši 22.4 mm plastike). Kada program završi s kompajliranjem 3D pisač se automatski kalibrira (stavlja u početni položaj) te se zagrijava plastika na definiranu temperaturu. Kada se dostigne potrebna temperatura 3D pisač počinje s ispisom. Njegov izgled, dizajn i funkcije će biti označene na slici 8.1 i objašnjeni ispod slike.

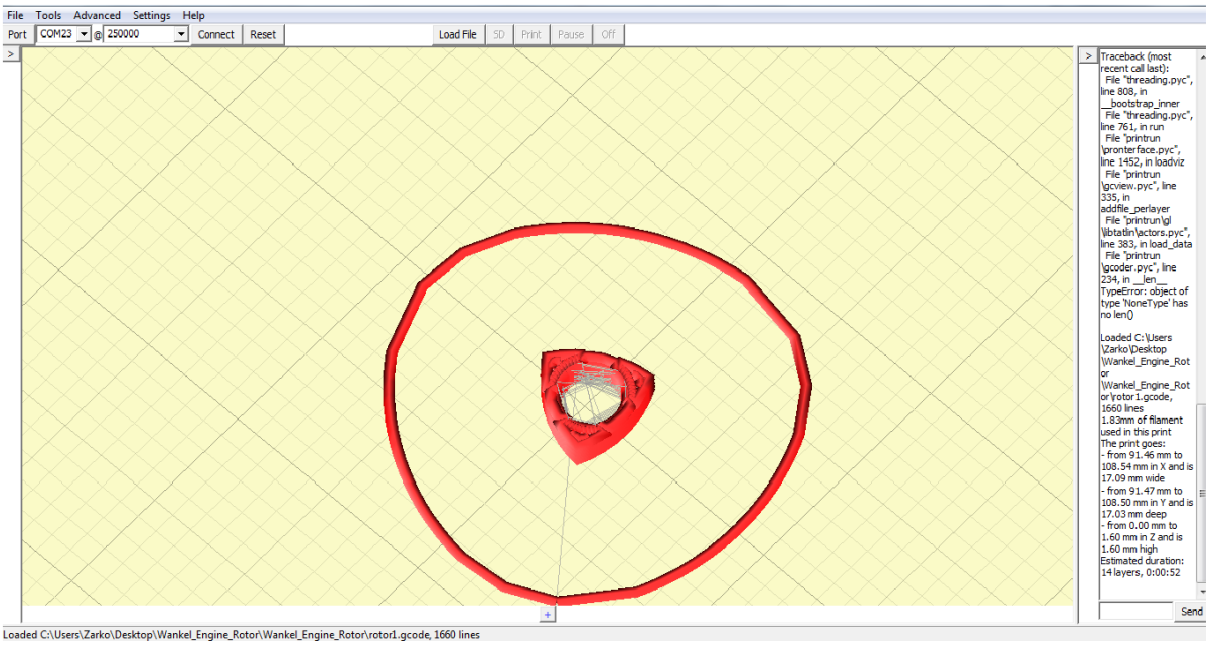


Slika 8.1-Pronterface

- U ovom prozoru odabire se PORT na kojem se spaja 3D pisač, broj bitova koji će se slati te dva gumba: CONNECT koji služi da se selektirani port otvori te započne konekcija 3D pisača s računalom te RESET koja vraća sve postavke na zadane.

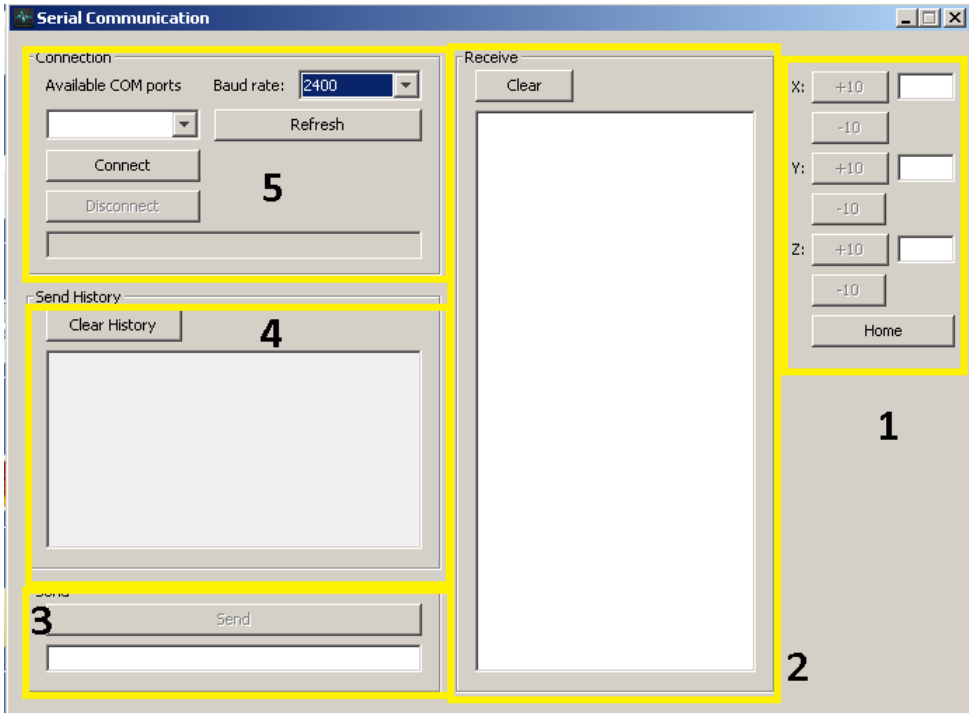
2. Ovaj prozor služi da se koračni motori kalibriraju te postave na početne pozicije koje definiraju ranije spominjani krajnici.
3. Postavljanje temperature grijača: može se unaprijed odabrati zadana temperatura za PLA i ABS plastiku u padajućem izborniku.
4. Postavljanje temperature grijače ploče.
5. Graf koji pokazuje temperaturu u ovisnosti o vremenu.
6. Podešavanje brzine ispisa.
7. Prostor u kojem se iscrtava željeni model te kako će oni izgledati.
8. U ovom prozoru moguće je poslati komande pisaču koje su napisane u G-kodu.

Nakon što se odabere model pomoću gumba Load File, taj model se prikazuje unutar programa kako će izgledati nakon ispisa te se u desnom kutu prikaže broj linija koda, koliko će se plastike utrošiti za taj ispis, kolike će mu biti dimenzije te vrijeme ispisa i slojeva. Na slici 8.2 je prikazan prozor u kojemu se vidi model te njegove informacije.



Slika 8.2 – Prikaz modela u Pronterface-u

Uz izvedbu 3D pisača, njegovo konfiguriranje i povezivanje sa sustavom za upravljanje, u okviru praktičnog dijela rada izrađena je pomoćna aplikacija kojom je moguće izvršiti jednostavnu kontrolu pisača, slanje G-kod naredbi. Aplikacija je napisana u programskom jeziku Java, korištenjem NetBeans razvojnog okruženja. Aplikacija je bazirana na standardnom sučelju za izvedbu serijske komunikacije računala s uređajima u okolini, uz dodatne elemente sučelja koji omogućavaju slanje standardnih naredbi (npr. pomak po radnim osima) u obliku G-koda. Njegov dizajn je prikazan na slici 8.2 te pojašnjen, a njegov programski kod će biti prikazan kasnije u projektu.



Slika 8.2- izrađeni program

1. Prozor u kojem se nalaze gumbi za kontroliranje X,Y i Z osi te sa strane su oblačići u kojima piše koliko koraka je bila pomaknuta koja os.

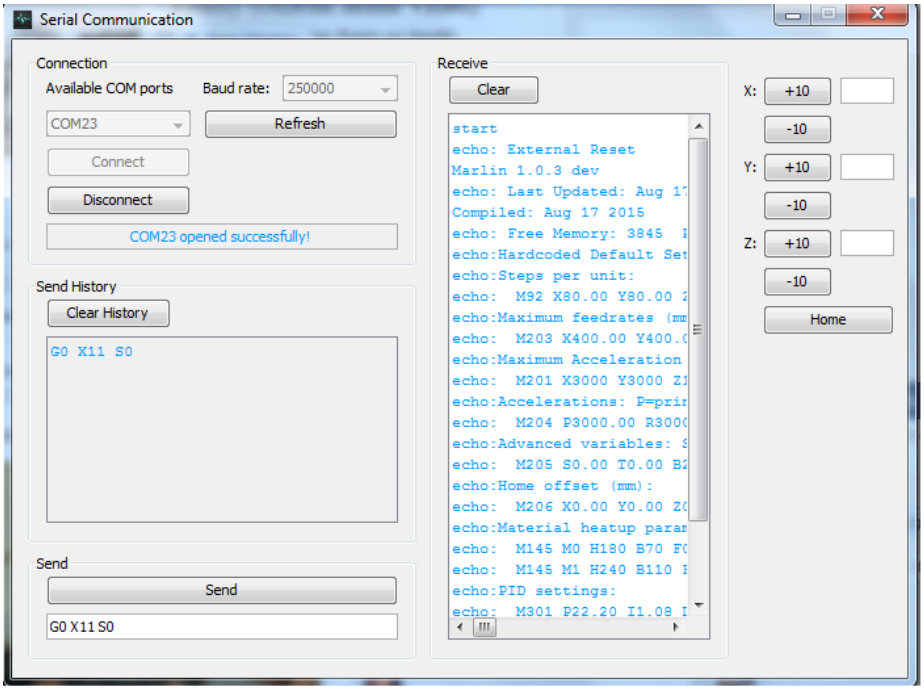
2. Prozor u kojemu se izlistaju informacije s Arduina kada se spoji s računalom.

3. Prozor koji služi za unos komandi te za slanje u 3D pisač.

4. Popis komandi koje smo prethodno slali 3D pisaču.

5. Kada se 3D pisač spoji s računalom, program automatski odabire port te postoji gumb s kojim se 3D pisač spaja s računalom.

Naredbe koje se šalju printer zapisane su u G-kodu te se one šalju iz prozorčića 3 sa slike 8.2.Kada se pošalje komanda na prozoru 4. je vidljivo koja komanda je poslana te da li je pisač "razumio" tu komandu ili ne. Na slici 8.3 je poslana jedna naredba na pisač te je vidljivo na prozoru 4. da je pisač dobio tu naredbu.

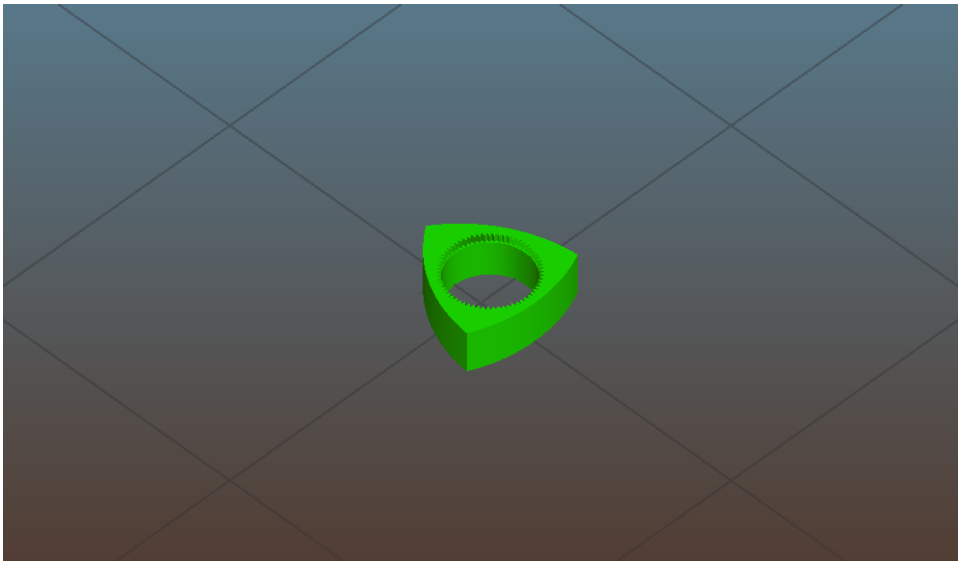


Slika 8.3 - isječak rada izrađenog programa

8.1 Primjer ispisanog modela

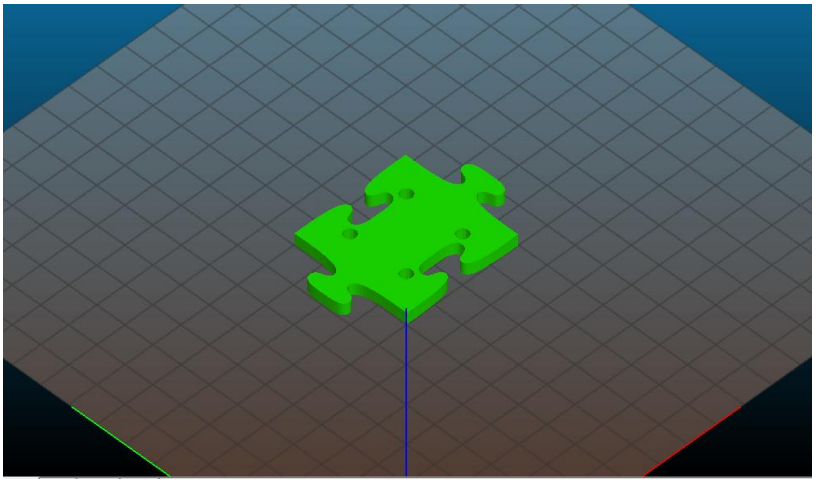
Najprije se u program stavi model kojeg želimo ispisati. 3D pisač korišten u projektu može ispisivati modele veličine 20cm x 20cm x 20cm pa je to potrebno uzeti u obzir. Program

automatski podesi početno stanje ekstrudera te temperature grijača i grijače ploče. Na dnu programa vidi se preostalo vrijeme ispisa, koliko linija ima kod te koliko plastike će biti utrošeno prilikom ispisa. Na slikama 8.4 je prikazan rotor Wankel motora u programu a na slici 8.5 se vidi ispisani model a na slikama 8.6 je prikaz puzzle u programu za modeliranje a na slici 8.7 je ispisani model.



Slika 8.4 – 3D model Wankel motora

Slika 8.5 – Ispisani 3D model Wankel motora veličine 4cm x 4cm



Slika 8.6 – 3D model puzzle

Slika 8.7 – ispisani 3D model puzzle veličine 7cm x 3cm

9. Projekt

Glavni dio projekta bio je izraditi program koji će moći kontrolirati 3D pisač. Dakle, bilo je potrebno isprogramirati programski kod kojim će Arduino primati naredbe s računala te ih preko serijske komunikacije slati na 3D pisač. Kada se cijeli projekt sastavi i poveže prema uputama s prethodnih cjelina može se početi izrađivati programski kod. U cjelini 9.1 bit će stavljen programski kod programa za kontroliranje 3D pisača te biti pojašnjen, a u cjelini 9.2 bit će isječak konfiguracijskog koda koji je stavljen u Arduino pločicu.

9.1 Programski kod za upravljanje 3D pisača

Programski kod za upravljanje pisača je napisan u programskom jeziku Java. Kada se program otvori najprije se odabere port na koji se otvara konekcija između računala i 3D pisača. Kada se odabere port tada se odabire broj bitova koji će biti poslani prema 3D pisaču. Taj kod je vidljiv u nastavku:

```
serialPort = (SerialPort)commPort;
serialPort.setSerialPortParams(Integer.valueOf(mainWindow.boxBaudRate.getSelected
Item().toStrin g()),

                    SerialPort.DATABITS_8,

                    SerialPort.STOPBITS_1,

                    SerialPort.PARITY_NONE);
```

Ako je port otvoren i ostvarena je konekcija s 3D pisačem javlja se poruka da je ostvarena komunikacija te se svi gumbi prikažu I moguće je stisnuti na njih i poslati komandu Arduinu da pomakne koračni motor za neku vrijednost u x,y,z osi

```
statusText = selectedPort + " openedsuccessfully!";

mainWindow.buttonConnect.setEnabled(false)
mainWindow.boxComPorts.setEnabled(false);
mainWindow.boxBaudRate.setEnabled(false);
mainWindow.buttonXminus10.setEnabled(true);
mainWindow.buttonXplus10.setEnabled(true);
mainWindow.buttonYminus10.setEnabled(true);
```

```
mainWindow.buttonYplus10.setEnabled(true);
mainWindow.buttonZminus10.setEnabled(true);
mainWindow.buttonZplus10.setEnabled(true);
mainWindow.buttonDisconnect.setEnabled(true);
mainWindow.buttonSend.setEnabled(true);
```

Ako je port već u upotrebi tada se javlja poruka sa tekstom:„ERROR_MESSAGE“

```
statusText = selectedPort + " is in use! (" +
e.toString() + ")";
JOptionPane.showMessageDialog(null,statusText,
                    "Serial communication",
                    JOptionPane.ERROR_MESSAGE);
```

Ako se port nije uspio otvoriti, odnosno ako nije izabran dobar port na koji se spaja 3D pisač, javlja se ista poruka kao i ako je port u upotrebi.

```
statusText = "Failed to open " + selectedPort + "! (" +
e.toString() + ")";
JOptionPane.showMessageDialog(null,statusText,
                    "Serial communication",
                    JOptionPane.ERROR_MESSAGE);
```

Upravljanje 3D pisača se izvodi pomoću gumba koji se nalaze na desnoj strani programa. Za svaku os nalaze se dva gumba. Jedan gumb se koristi za pomak 3D pisača 1 cm po toj osi udesno, a drugi gumb se koristi za pomak po 1 cm-u lijevo. Kraj tih gumba se nalazi prozorčić koji računa pomake. Komanda koja se šalje Arduinu pisana je u G-kodu te glasi : „E0 S1“ . Dakle pomakni se za 1 cm ako nije uključena krajnja sklopka.

```
 private void
    buttonXplus10ActionPerformed(java.awt.event.ActionEvent evt)
    { counterX +=10;
    String sendCommand = "G0 X" + counterX.toString() + " E0S1";

    for(int x = 0; x < sendCommand.length();x++)
    {
      communication.writeData(Integer.valueOf(sendCommand.charAt(x)));
    }
    textCounterX.setText(counterX.toString());
    communication.writeData(13);
    communication.writeData(10);
```

U programu postoji još jedan gumb koji se zove „Home“ , a služi da se sve osi postave u početni položaj te se brojač resetira. Način na koji pisač funkcionira je taj da računalo šalje

Arduino pločici pozicije ekstrudera u G-kodu, a Arduino pločica ovisno o tim pozicijama otvara svoje izlaze na kojima su spojeni koračni motori.

```
private void

    buttonHomeActionPerformed(java.awt.event.ActionEvent evt)

    { String sendCommand ="G28";
    for(int x = 0; x < sendCommand.length();x++)
    {
      communication.writeData(Integer.valueOf(sendCommand.charAt(x)));
    }
    counterX =0;
    counterY =0;
    counterZ = 0;
    textCounterX.setText(counterX.toString());
    textCounterY.setText(counterY.toString());
    textCounterZ.setText(counterZ.toString());
    communication.writeData(13);
    communication.writeData(10);
```

9.2 Isječak konfiguracijskog koda na Arduinu

Na Arduino je potrebno konfigurirati parametre za 3D pisač kao što su:

- port na kojeg se spaja 3D pisač
  ```
  #define SERIAL_PORT 0
  ```

- broj bitova koji se šalju 3D pisaču tj. Brzinu komunikacije
  ```
  #define BAUDRATE 250000
  ```

- odabire se kontrolna elektronika
  ```
  #ifndefMOTHERBOARD
   #define MOTHERBOARDBOARD_RAMPS_13_EFB
  #endif
  ```

- broj ekstrudera na 3D pisaču
  ```
  #define EXTRUDERS 1
  ```

- definiraju se temperaturni senzori
  ```
  #define TEMP_SENSOR_0 1
  #define TEMP_SENSOR_1 0
  #define TEMP_SENSOR_2 0
  #define TEMP_SENSOR_3 0
  #define TEMP_SENSOR_BED 1
  ```

- minimalna i maksimalna temperatura grijača te grijače ploče
  ```
  #define HEATER_0_MINTEMP 5
  ```

```
#define HEATER_1_MINTEMP 5
#define HEATER_2_MINTEMP 5
#define HEATER_3_MINTEMP 5
#define BED_MINTEMP -15
#define HEATER_0_MAXTEMP 275
#define HEATER_1_MAXTEMP 275
#define HEATER_2_MAXTEMP 275
#define HEATER_3_MAXTEMP 275
#define BED_MAXTEMP 150
```

- invertiranje x,y i z osi te puno drugih mogućnosti

```
#define INVERT_X_DIR false
#define INVERT_Y_DIR false
#define INVERT_Z_DIR true
```

Kada se podese svi parametri počinje zapisivanje programa na Arduino pločicu.

10. Zaključak

U ovom završnom radu opisana je konstrukcija, komponente te način rada 3D pisača. 3D pisač ima mogućnost proizvodnje različitih vrsta preciznih i postojanih trodimenzionalnih formi bez alata i strojeva jednostavnim pretvaranjem 3D datoteka u funkcionalne prototipove od tvrde plastike. 3D pisači su mini sustavi s numeričkim upravljanjem u tri osi (x, y, z). Osim toga, opisan je rad Arduino pločice, njezino spajanje na elektroniku 3D pisača te kompletan rad 3D pisača. Zapravo, korištena je Arduino opet-source platforma. Arduino je puno jednostavniji za programiranje, prednost mu je i ta što je njegov mikrokontroler već implementiran na tiskanu pločicu pa nije potrebna izrada dodatne platforme. Program stavljen na Arduino pločicu naziva se *Marlin*. U završnom radu najviše problema je stvarala kalibracija z osi jer širina ispisa treba biti debljine A4 papira što je otprilike 0.05 mm. Tu je nastao problem jer z os treba doći u početnu poziciju, a to određuje krajnja sklopka te je bilo veoma teško namjestiti na tako malu vrijednost. Izrada programa za kontroliranje 3D pisača nije bila nešto zahtjevna jer smo programsko okruženje NetBeans koristili na mnogim kolegijima tijekom školovanja pa se trebalo samo malo prisjetiti principa rada programa. Kao što je u projektu bilo prikazano, 3D pisač koristi koračne motore koji mogu dati visoku preciznost ispisa. Korišten je NEMA 17 bipolarni koračni motor. Važno je i za spomenuti da je prisutna serijska komunikacija s računalom gdje se razmjenjuju podaci putem jednog para signala te sve informacije koje nas zanimaju u vezi 3D pisača možemo vidjeti i na računalu što mi je dosta pomoglo tijekom ovoga projekta. Jedna od komponenta 3D pisača koja je također korištena je ekstruder. Njegova zadaća je zapravo da zagrije plastiku te da ju ravnomjerno istiskuje po površini. LCD ekran je korišten u svrhu upravljanja 3D pisača bez upotrebe računala te ima mogućnost učitavanja programa sa SD kartice i pretvaranje u G-kod. Najvažniji dio projekta bio je da izraditi program koji će moći kontrolirati 3D pisač. Na kraju svega ovoga zaključujem da je vrlo korisno imati jedan 3D pisač kod kuće jer pomoću njega je moguće izraditi sve stvari koje mogu biti bitne, a mogu biti nama samo za zabavu. Vrlo je jednostavan za izradu te nije potrebno imati nikakvih prijašnjih iskustava u 3D modeliranju i ispisu. Bio sam vrlo zainteresiran za izradu 3D pisača te sljedeća ideja koju imam je da izradim CNC stroj koji radi na istom principu kao i 3D pisač, samo što on ne upotrebljava plastiku, nego ima posebno dizajnirano svrdlo umjesto ekstrudera koje se okreće velikom brzinom te tako

obrađuje metal.

11. Popis literature

[1] https://en.wikipedia.org/wiki/3D_printing

[2] https://www.arduino.cc

[3] http://reprap.org/wiki/Marlin

[4] http://reprap.org/wiki/NEMA_17_Stepper_motor

[5] http://reprap.org/wiki/Heated_Bed#Glass

[6] http://reprap.org/wiki/Stepper_motor

[7] http://reprap.org/wiki/NEMA_Motor

[8] http://reprap.org/wiki/RepRapPro_Mendel

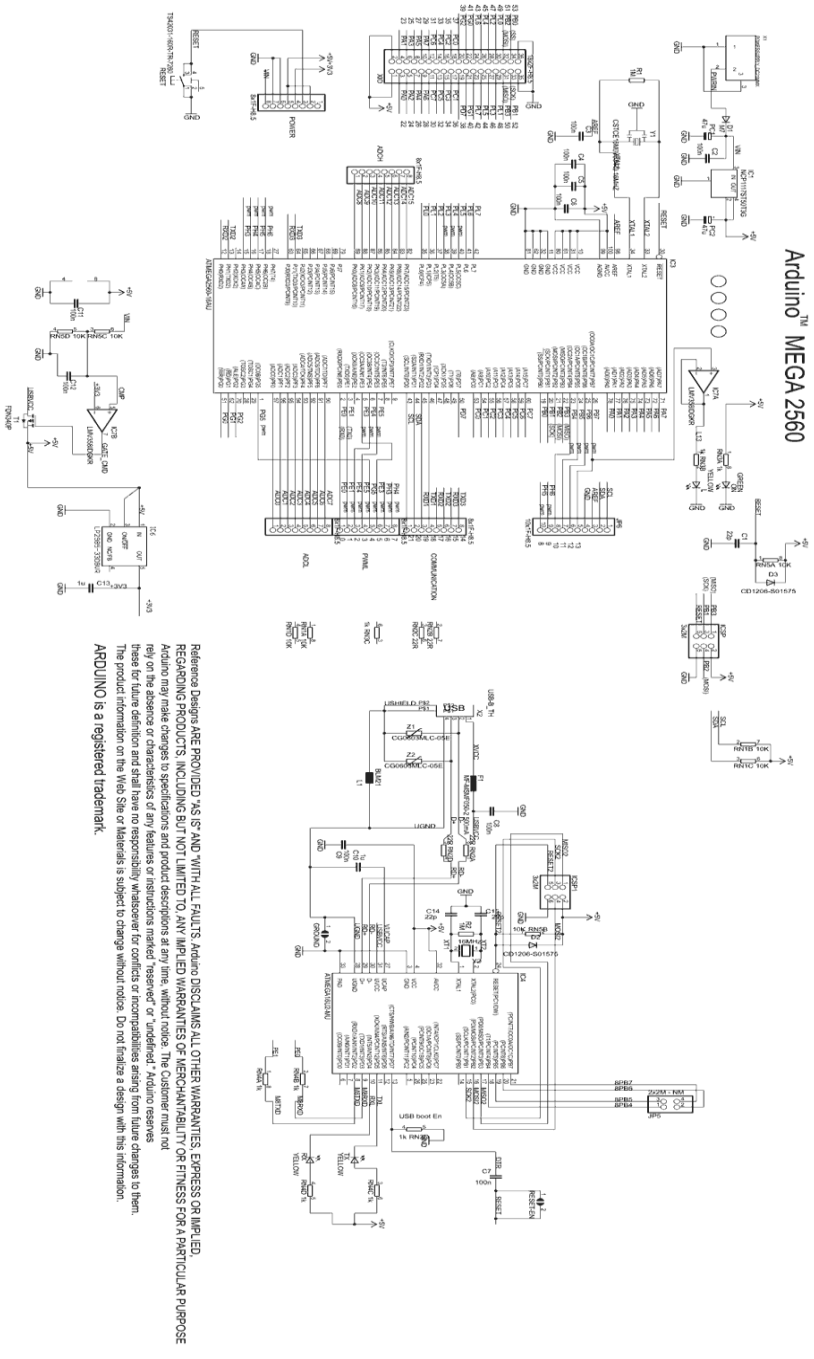
[9] http://reprap.org/wiki/Mendel

[10] https://bs.wikipedia.org/wiki/3D_printanje

[11] Michael Margolis: Arduino Cookbook, 2nd Edition, O'Reilly Media, 2011

[12] Isaac Budmen: The Book on 3D Printing, Anthony Rotolo, Paperback, 2013

12. Prilog



Prilog 1 – shema Arduina

```
package SerialCommunication;

import gnu.io.*;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.TooManyListenersException;
import javax.swing.JOptionPane;

public class Communication implements SerialPortEventListener {

    MainWindow mainWindow = null;
    private Enumeration ports = null;
    private final HashMap portMap = new HashMap();
    private CommPortIdentifier selectedPortIdentifier = null;
    private SerialPort serialPort = null;
    private InputStream input = null;
    private OutputStream output = null;
    private boolean bConnected = false;
    final static int TIMEOUT = 2000;
    final static int SPACE_ASCII = 32;
    final static int DASH_ASCII = 45;
    final static int NEW_LINE_ASCII = 10;


    String statusText = "";
    String receiveData = "";

    public Communication(MainWindow mainWindow)
    {
        this.mainWindow = mainWindow;
    }


    public void searchForPorts()
    {
        ports = CommPortIdentifier.getPortIdentifiers();

        while (ports.hasMoreElements())
        {
            CommPortIdentifier curPort = (CommPortIdentifier)ports.nextElement();

            if (curPort.getPortType() == CommPortIdentifier.PORT_SERIAL)
            {
                mainWindow.boxComPorts.addItem(curPort.getName());
                portMap.put(curPort.getName(), curPort);
            }
        }
    }


    public void connect() {

        String selectedPort = (String)mainWindow.boxComPorts.getSelectedItem();
        selectedPortIdentifier = (CommPortIdentifier)portMap.get(selectedPort);

        CommPort commPort = null;
```

```
        try
        {

            commPort = selectedPortIdentifier.open("SerialCommunication",
TIMEOUT);
            serialPort = (SerialPort)commPort;

serialPort.setSerialPortParams(Integer.valueOf(mainWindow.boxBaudRate.getSelectedI
tem().toString()),
                           SerialPort.DATABITS_8,
                           SerialPort.STOPBITS_1,
                           SerialPort.PARITY_NONE);

            setConnected(true);

            statusText = selectedPort + " opened successfully!";
            mainWindow.textPortStatus.setText(statusText);

            mainWindow.buttonConnect.setEnabled(false);
            mainWindow.boxComPorts.setEnabled(false);
            mainWindow.boxBaudRate.setEnabled(false);

            mainWindow.buttonXminus10.setEnabled(true);
            mainWindow.buttonXplus10.setEnabled(true);
            mainWindow.buttonYminus10.setEnabled(true);
            mainWindow.buttonYplus10.setEnabled(true);
            mainWindow.buttonZminus10.setEnabled(true);
            mainWindow.buttonZplus10.setEnabled(true);

            mainWindow.buttonDisconnect.setEnabled(true);
            mainWindow.buttonSend.setEnabled(true);
        }
        catch (PortInUseException e)
        {
            statusText = selectedPort + " is in use! (" + e.toString() + ")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
        }
        catch (Exception e)
        {
            statusText = "Failed to open " + selectedPort + "! (" + e.toString() +
")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
        }
    }


    public boolean initIOStream()
    {
        boolean successful = false;

        try {

            input = serialPort.getInputStream();
            output = serialPort.getOutputStream();
```

```
            successful = true;
            return successful;
        }
        catch (IOException e) {
            statusText = "I/O Streams failed to open. (" + e.toString() + ")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
            return successful;
        }
    }
    public void initListener()
    {
        try
        {
            serialPort.addEventListener(this);
            serialPort.notifyOnDataAvailable(true);
        }
        catch (TooManyListenersException e)
        {
            statusText = "Too many listeners! (" + e.toString() + ")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
        }
    }

    public void disconnect()
    {
         try
        {
            serialPort.removeEventListener();
            serialPort.close();
            input.close();
            output.close();
            setConnected(false);


            statusText = "Disconnected!";
            mainWindow.textPortStatus.setText(statusText);

            mainWindow.buttonDisconnect.setEnabled(false);
            mainWindow.buttonSend.setEnabled(false);
            mainWindow.buttonXminus10.setEnabled(false);
            mainWindow.buttonXplus10.setEnabled(false);
            mainWindow.buttonYminus10.setEnabled(false);
            mainWindow.buttonYplus10.setEnabled(false);
            mainWindow.buttonZminus10.setEnabled(false);
            mainWindow.buttonZplus10.setEnabled(false);

            mainWindow.boxComPorts.setEnabled(true);
            mainWindow.boxBaudRate.setEnabled(true);
            mainWindow.buttonConnect.setEnabled(true);

        }
        catch (Exception e)
        {
            statusText = "Failed to close " + serialPort.getName() + "(" +
```

```
e.toString() + ")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
        }
    }

    final public boolean getConnected()
    {
        return bConnected;
    }

    public void setConnected(boolean bConnected)
    {
        this.bConnected = bConnected;
    }
    public void serialEvent(SerialPortEvent evt) {
        if (evt.getEventType() == SerialPortEvent.DATA_AVAILABLE)
        {
            try
            {
                byte singleData = (byte)input.read();
                    receiveData = new String(new byte[] {singleData});
                    mainWindow.textReceive.append(receiveData);
            }
            catch (Exception e)
            {
                statusText = "Failed to read data! (" + e.toString() + ")";
                JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    public void writeData(int data)
    {
        try
        {

            output.write(data);
            output.flush();
        }
        catch (Exception e)
        {
            statusText = "Failed to write data! (" + e.toString() + ")";
            JOptionPane.showMessageDialog(null, statusText,
                                          "Serial communication",
                                          JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

Prilog 2. Kod programa za upravljanje 3D pisač

Sveučilište Sjever

SVEUČILIŠTE SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio 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, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, _____ (ime i prezime) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor završnog rada pod naslovom _____ _____ (upisati naslov) te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Žarko Uršulin

_____

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, _____ (ime i prezime) neopozivo izjavljujem da sam suglasan s javnom objavom završnog rada pod naslovom _____ _____(upisati naslov) čiji sam autor.

Žarko Uršulin

_____