**SVEUČILIŠTE JOSIPA JURJA STORSSMAYERA U OSIJEKU**

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

**Sveučilišni studij**

**SAMOBALANSIRAJUĆI ROBOT**

**UGRADBENI RAČUNALNI SUSTAVI**

**Bruno Zahirović, Mislav Ćaleta**

**Osijek, 2022.**

**SADRŽAJ**

[1. UVOD 1](#_Toc114691242)

[1.1. Zadatak i struktura zadataka 1](#_Toc114691243)

[2. SAMOBALANSIRAJUĆI ROBOT NA PRINCIPU OBRNUTOG NJIHALA 3](#_Toc114691244)

[2.1. Teorijski osvrt na problem balansiranja prema principu obrnutog njihala 3](#_Toc114691245)

[2.2. Prijedlog sklopovskog rješenja 3](#_Toc114691246)

[2.3. Prijedlog programskog rješenja 5](#_Toc114691247)

[3. REALIZACIJA SAMOBALANSIRAJUĆEG ROBOTA 7](#_Toc114691248)

[3.1. Korišteni alati i programska okruženja 7](#_Toc114691249)

[3.2. Realizacija sklopovskog rješenja 8](#_Toc114691250)

[3.3. Realizacija programskog rješenja 16](#_Toc114691251)

[4. TESTIRANJE I REZULTATI 19](#_Toc114691252)

[4.1. Metodologija testiranja 19](#_Toc114691253)

[4.2. Rezultati testiranja 19](#_Toc114691254)

[5. ZAKLJUČAK 23](#_Toc114691255)

[LITERATURA 24](#_Toc114691256)

[PRILOZI I DODATCI 25](#_Toc114691257)

# UVOD

U ovom radu pristupa se rješenju problema balansiranja mobilnog robota na principu obrnutog njihala korištenjem PID regulatora. Održavanje ravnoteže robota jedan je od većih problema robotike. Jedan od sličnih primjera ovakvog proizvoda, je vozilo na dva kotača koje proizvodi tvrtka „Segway“. Vozilo tvrtke „Segway“ koristi se u raznim okruženjima i prigodama, to mogu biti, prema [1], turistički obilasci, tvornička okruženja, patrola po otvorenim površinama. Način na koji se vozilo koristi je premještanjem svojeg težišta prema naprijed ili nazad, odnosno naginjanjem. Vozilo će prepoznati promjenu pomoću nagibnih senzora i žiroskopa i zadržati potrebnu ravnotežu za kretanje prema naprijed ili nazad. Prema [2], još jedan od razvijenih proizvoda koji obavlja funkciju održavanja vlastite ravnoteže je robot koji za kretanje također koristi dva kotača i žiroskop kontrolnog momenta za kompenzaciju poremećaja. Ovakav način rješenja, odnosno ovakva konstrukcija mobilnog robota koji koristi dva kotača ima svoje prednosti u tome što za razliku od mobilnih robota koji koriste više kotača, lakše je upravljan na mjestima gdje je prostor ograničen. Također, svoje prednosti robot s kotačima nad robotima koji se kreću mehaničkim nogama ima u brzini i lakšoj okretljivosti. Budući da ipak takav robot, koji koristi samo dva kotača ima najveći problem u ravnoteži, rješavanjem tog problema, postižu se gore navedene prednosti nad robotima s više kotača ili s nogama, ali bez problema ravnoteže. Motivacija za ovaj rada je riješiti taj problem i dobiti robota koji ima prednosti kompaktnog robota s dva kotača bez njegovog glavnog problema. Način na koji se pokušava doći do rješenja uzet je iz teorije upravljanja. Ovakav problem prikladan je za korištenje regulatora, zato što je robot na dva kotača izložen poremećajima iz svoje okoline, od guranja do sudara u različite objekte. Te poremećaje potrebno je kompenzirati kako bi robot zadržao željeni položaj. Razlog za korištenje PID regulatora je njegova mogućnost brzine i stabilnosti, kao i uspješnosti u rješavanju sličnih problema gdje je potrebno precizno pratiti neku referentnu veličinu.

## Zadatak i struktura zadataka

Zadatak ovog projekta je izraditi robota koji će imati mogućnost zadržati što uspravniji položaj, ali će za kretanje koristiti samo dva kotača. Robot bi trebao imati mogućnost kompenzacije poremećaja kao što su sudari u različite prepreke ili blago odguravanje robota. U prvom poglavlju ovog pisanog rada nalazi se uvod gdje su navedeni neki slični proizvodi i glavni cilj i motivacija za ovaj projekt. U drugom poglavlju objašnjena je teorijska podloga rješenja problema, prijedlog sklopovskog rješenja korištenjem funkcionalnog blok dijagrama sklopovskog dijela projekta uz njegovo objašnjenje i blok dijagram tijeka programa koji predstavlja prijedlog rješenja programskog dijela. U trećem poglavlju kratko su opisani korišteni alati i programsko okruženje. U drugom dijelu trećeg poglavlja detaljno je objašnjena električna shema i kako funkcionira. U zadnjem potpoglavlju objašnjeno je detaljno programsko rješenje uz korištenje blok dijagrama tijeka programa. U četvrtom poglavlju objašnjena je metodologija testiranja i postupci koji su se koristili prilikom testiranja. Također su prikazani rezultati testiranja. U petom se poglavlju nalazi zaključak s istaknutim prednostima i manama projekta i pregledom svih prethodnih poglavlja. U šestom poglavlju mogu se pronaći navedene korištene literature, a na kraju rada su smješteni prilozi.

# SAMOBALANSIRAJUĆI ROBOT NA PRINCIPU OBRNUTOG NJIHALA

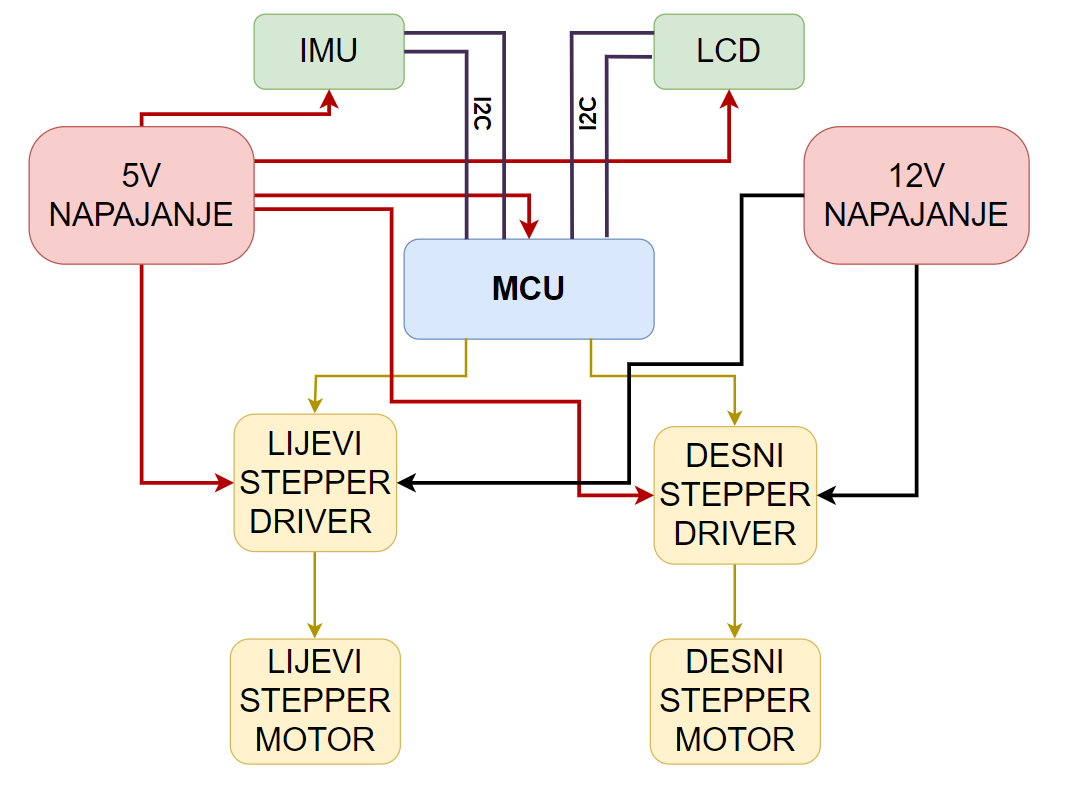
## Teorijski osvrt na problem balansiranja prema principu obrnutog njihala

Samo-balansirajući robot na dva kotača, ili robot na principu obrnutog njihala, prema [3] spada u nelinearne sustave, koji se sastoji od baze s dva kotača i tijela koje se ponaša kao obrnuto njihalo. Takav sustav implementira tri stupnja slobode, uključujući nagib, rotaciju oko uspravne osi (kako bi se omogućilo skretanje) te kretanje u jednom smjeru, kako bi se postigla potpuna mogućnost kretanja robota u 2D prostoru, kao i mogućnost uspinjanja po podlogama koje su pod kutom te balansiranje na istima. Iako su sami po sebi nestabilni i zahtijevaju konstanto praćenje nagiba i poduzimanje radnji kako bi ostali uspravni, prednost robota na principu obrnutog njihala je ta što za razliku od robota s više kotača mogu provući kroz mnogo manja mjesta te imaju mogućnost potpune rotacije oko uspravne osi stojeći na identičnom mjestu.

Kako je spomenuto, potrebno je konstantno pratiti nagib robota te djelovati pomoću aktuatora kako bi robot ostao u uspravnom položaju, a u sljedeća dva potpoglavlja prikazat će se prijedlozi sklopovskog i programskog rješenja kojima bi se mogli postići željeni rezultati, dok će u trećem poglavlju oni biti detaljnije razjašnjeni.

## Prijedlog sklopovskog rješenja

Kako bi se robot mogao samostalno balansirati, potrebno je osmisliti sklopovlje koje će imati mogućnost praćenja trenutnog nagiba robota, te ovisno o njemu slati upravljačke signale na aktuatore robota. Prijedlog sklopovskog rješenja za ovaj problem prikazan je na slici 2.1.



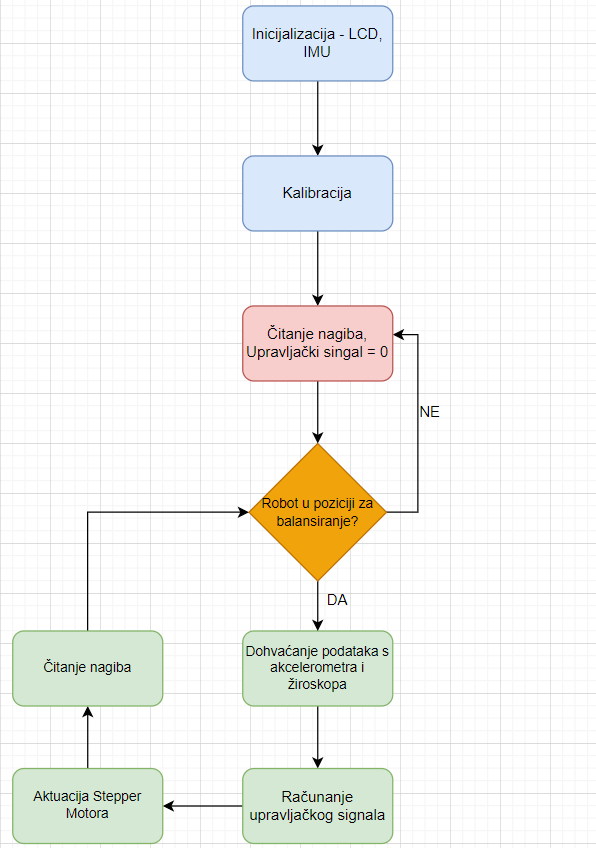
**Slika 2.1.** Sklopovski prijedlog rješenja problema samo-balansirajućeg robota.

Kao i kod svakog sustava, potrebna je jedinica koja će se brinuti o logici sustava, i predložen je mikroupravljač (eng. Microcontroller Unit, MCU na shemi), zbog jednostavnijeg komuniciranja s ostalim uređajima u odnosu na korištenje samog mikroprocesora. Specifični mikroupravljač koji će se koristiti za realizaciju projekta bit će naveden u trećem poglavlju, kao i ostale specifične komponente. Kako bi mogli pratiti nagib robota te o tom nagibu mogli izračunati potreban upravljački signal za balansiranje robota, potrebna nam je inercijskom mjerna jedinica (eng. Inertial Measurement Unit, IMU na shemi). Takva komponenta često sadrži podatke o akceleracijama u 3 dimenzije, kao i o brzini rotacije u tri dimenzije, i tim će se podacima moći izračunati nagib robota te koristiti ih za izračune upravljačke veličine. Kada je upravljačka veličina izračunata, potrebno ju je proslijediti na aktuatore kako bi oni izvršili radnju koja će robota balansirati, odnosno takvu radnju da robot ostane u uspravnom položaju. Predloženo je da su aktuatori u ovoj realizaciji projekta koračni (eng. stepper) motori. Naravno, za takve aktuatore potrebni su i pokretači (eng. driver) koji će od mikroupravljača primiti upravljačke signale te proslijediti ih na motore. Još je predložen LCD zaslon kako bi se korisniku mogle prenositi bitne informacije, kao dovršetak kalibracije ili greška prilikom rada. Pošto LCD zasloni i inercijske mjerne jedinice često dolaze u paketu koji pruža komunikaciju putem I2C sučelja, predložen je taj oblik komunikacije između te dvije komponente i mikroupravljača. Još su potrebni izvori energije kako bi sve komponente mogle pravilno raditi, stoga je predloženo odvojeno napajanje od pet volta kao izvor energije svih komponenti kojima je to potrebno za normalan logički rad, te napajanje od dvanaest volta koje je potrebno pokretačima koračnih motora kako bi motorima mogli opskrbiti dovoljno snage za neometan rad.

U sljedećem potpoglavlju prikazan je prijedlog programskog rješenja problema samo-balansirajućeg robota.

## Prijedlog programskog rješenja

Imajući prijedlog sklopovskog rješenja, potrebno je predložiti i programsko rješenje koje će implementirati logiku te povezati sve dijelove sa sklopovskog rješenja u jednu cjelinu, koja će na kraju imati mogućnost ostvariti željeno vladanje robota, a taj prijedlog dan je na slici 2.2.



**Slika 2.2.** Programski prijedlog rješenja problema samo-balansirajućeg robota.

Prijedlog programskog rješenja kreće s inicijalizacijom komponenti kojima je to potrebno, a to će zasigurno biti komponente koje će s mikroupravljačem komuniciraju preko I2C sučelja, kako bi se osiguralo da su na pravim adresama za komunikaciju. Nakon što je inicijalizacija obavljena (i uspješna), bit će potrebna kalibracija sustava, pošto korisnik robota može uključiti u bilo kojoj poziciji. Nakon obavljene kalibracije, program će promatrati nagib robota, te čekati da ga korisnik postavi u poziciju za balansiranje, odnosno da ga uspravi. Dok taj uvjet nije ostvaren, program će za upravljački signal postavljati vrijednost nula, odnosno neće biti nikakvog upravljanja. Kada korisnik uspravi robota, program će krenuti u glavnu petlju, koja će računati potrebne upravljačke signale za pročitani nagib robota kako bi robot ostao u uspravnom položaju. Ako se dogodi da robot prijeđe granicu za koju je balansiranje moguće, odnosno prijeđe nagib iz kojeg se može vratiti u neutralan uspravni položaj, upravljački signal će se ponovno postaviti na nulu kako se kotači i motori ne bi okretali dok robot nije u mogućnosti održavanja ravnoteže.

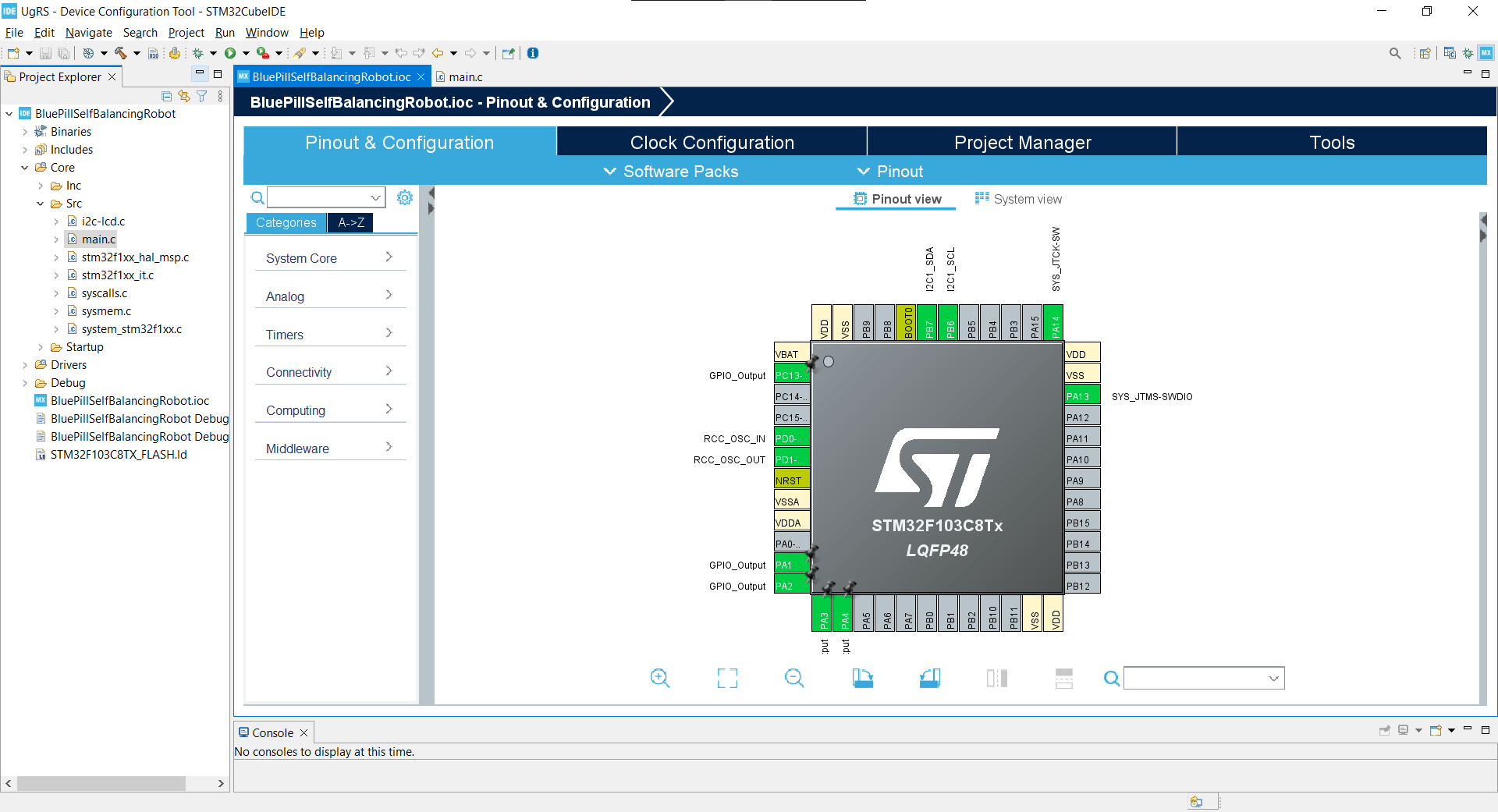
U sljedećem poglavlju, predstavit će se korišteni alati i programska okruženja te će se detaljnije opisati realizacija sklopovskog i programskog rješenja.

# REALIZACIJA SAMOBALANSIRAJUĆEG ROBOTA

## Korišteni alati i programska okruženja

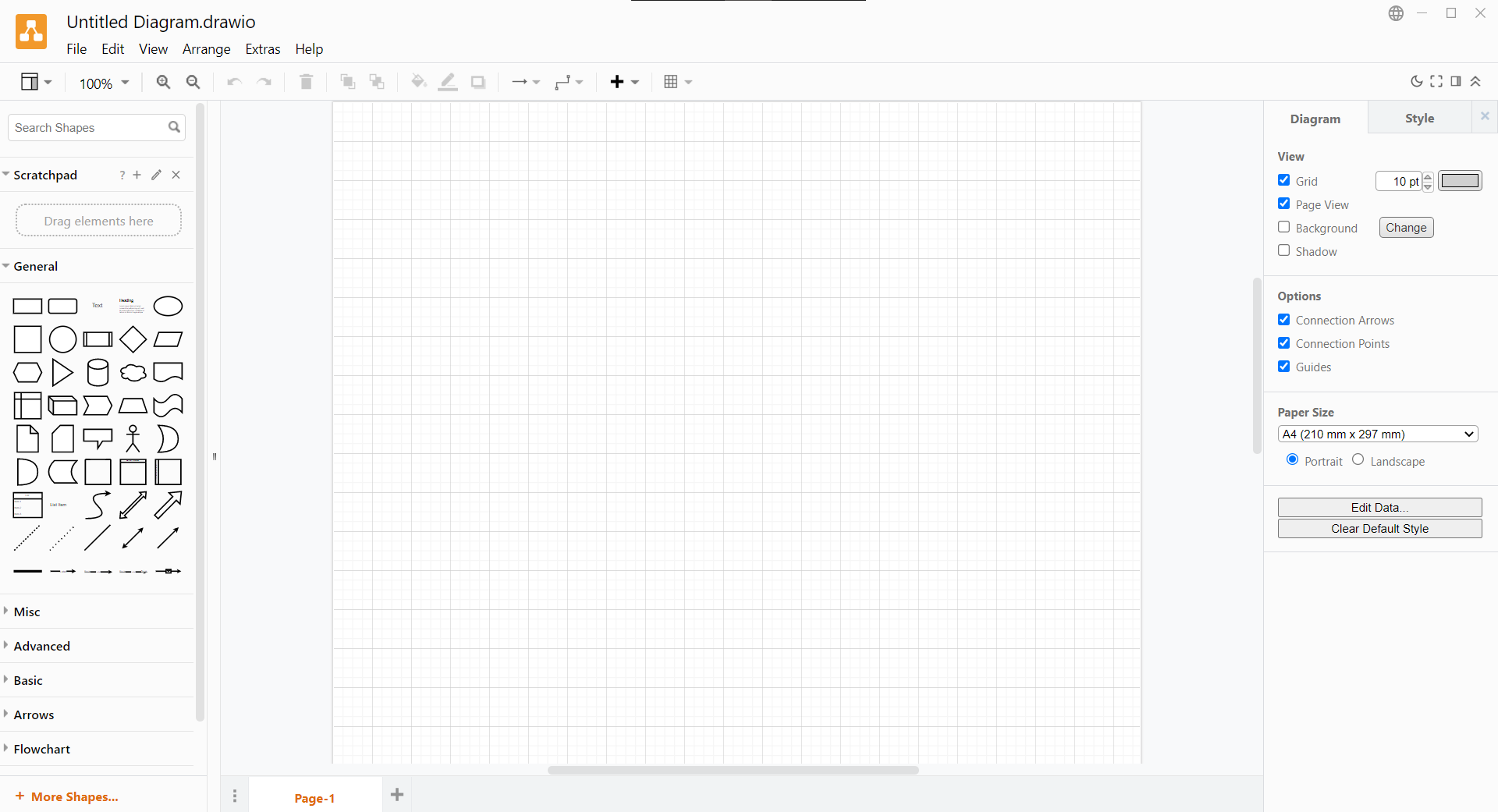
Kako bi se projekt realizirao, korišteno je nekoliko alata kojima je omogućeno razvijanje te testiranje.

**STM32CubeIDE** je napredna C/C++ razvojna platforma s mogućnosti konfiguriranja perifernih uređaja, generacije programskog koda, kompilatora programskog koda te mogućnosti praćenja stanja prilikom pokretanja koda za sve STM32 mikroupravljače i mikroprocesore [4]. Razvojna platforma STM32CubeIDE vrlo je korisna zbog mogućnosti konfiguriranja perifernih uređaja te automatskog generiranja koda za svaki korišteni periferni uređaj, kako se korisnik ne bi morao brinuti o implementiranju koda kojima će komunicirati s perifernim uređajima u svakom novom projektu ili za različite mikroupravljače ili mikroprocesore, a uz to ima i poznato korisničko sučelje koje je bazirano na „Eclipse“ integriranom razvojnom okruženju. Na slici 3.1 prikazano je korisničko sučelje razvojne platforme STM32CubeIDE.



**Slika 3.1.** Prikaz korisničkog sučelja razvojne platforme STM32CubeIDE.

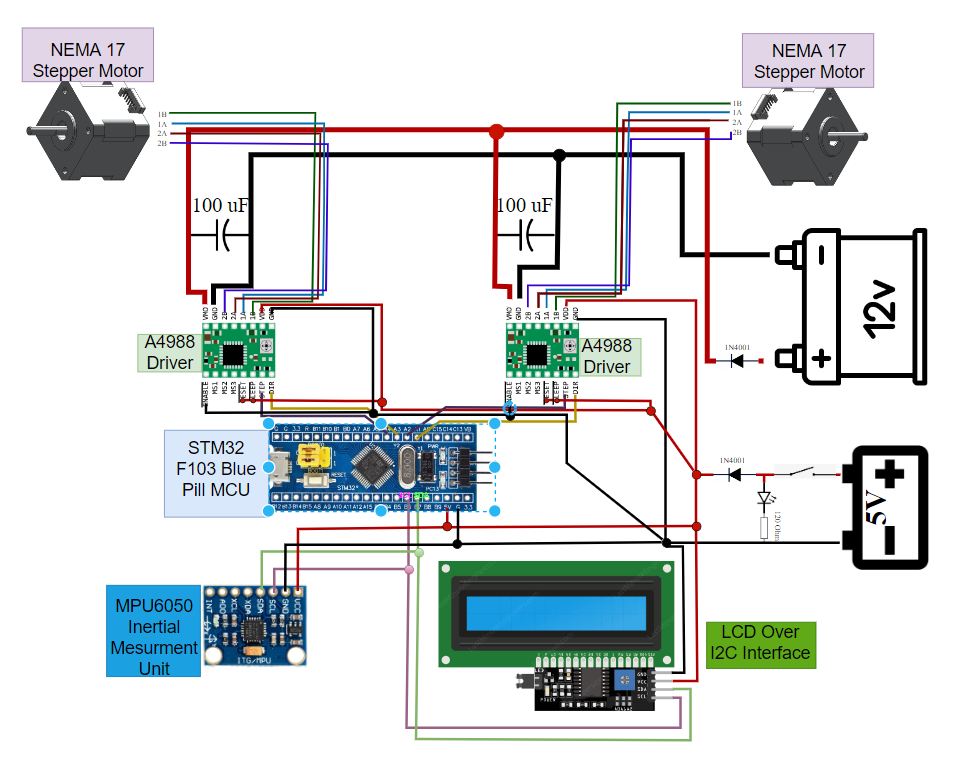
**Draw.io** je program za izradu dijagrama i shema, bilo to programskih ili sklopovskih. Program korisniku daje mogućnost biranja između automatskog raspoređivanja komponenti ili izrade prilagođenog rasporeda po korisnikovim željama [5]. Također, postoji i mogućnost spremanja izrađenih dijagrama na oblak. Draw.io je korišten prilikom izrade prijedloga rješenja te prilikom izrade sheme realizacije sklopovskog rješenja te blok dijagrama realizacije programskog rješenja. Na slici 3.2 prikazano je korisničko sučelje programa Draw.io.



**Slika 3.2.** Prikaz korisničkog sučelja programa Draw.io

## Realizacija sklopovskog rješenja

Na slici 3.3 prikazana je detaljna elektronička shema kojom je realizirano sklopovsko rješenje danog problema, na kojoj su prikazane i označene sve korištene komponente te sve poveznice među njima.



**Slika 3.3.** Prikaz električne sheme kojom je realizirano sklopovsko rješenje.

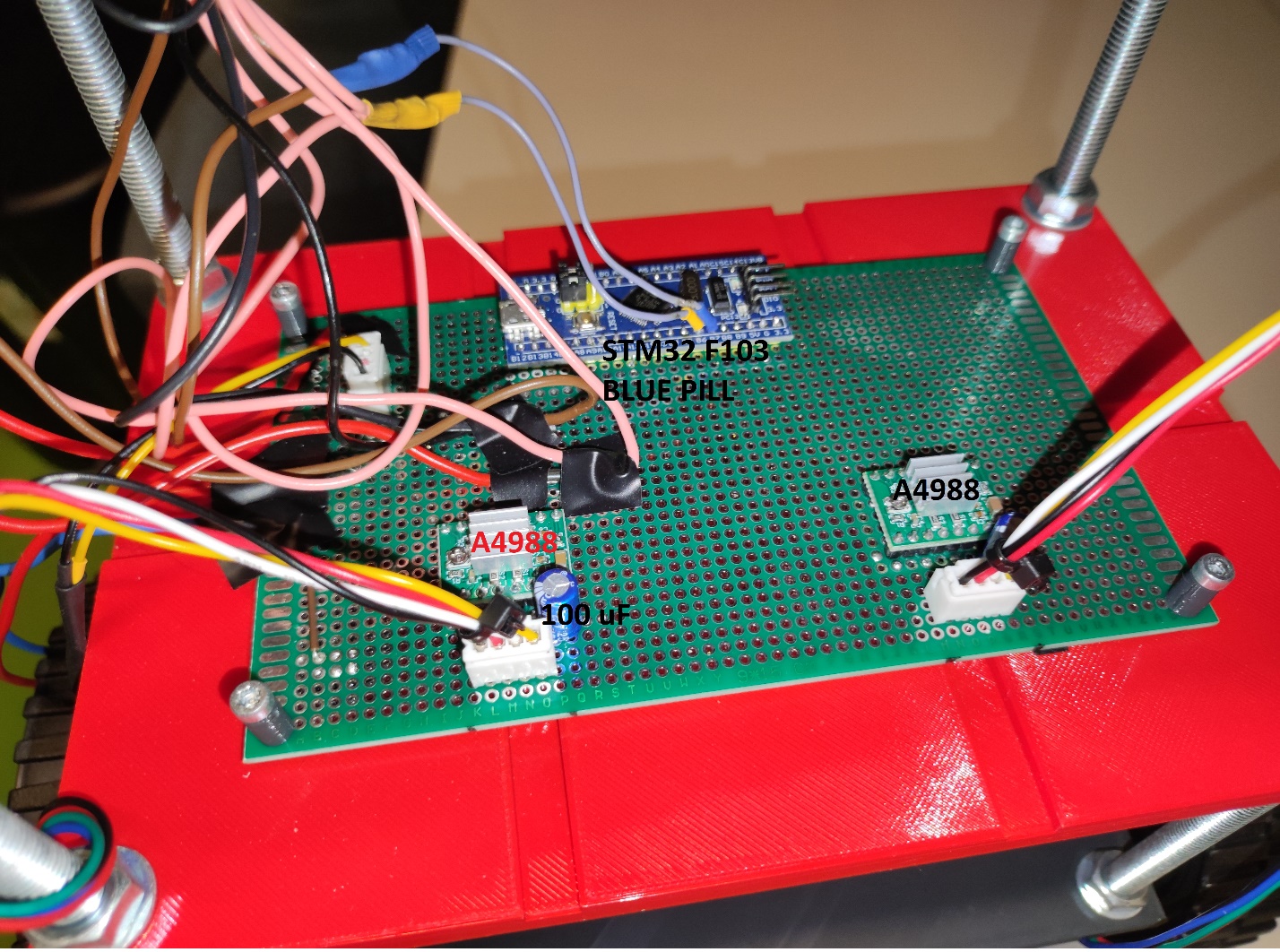
Kako je moguće vidjeti, za mikroupravljač odabran je STM32 F103 Blue Pill, te je na njemu implementirana cijelo programsko rješenje, i on služi kao glavna logička komponenta koja omogućuje komunikaciju između ostalih komponenti. Na njemu je omogućeno I2C sučelje kako bi mogao komunicirati s LCD zaslonom te inercijskom mjernom jedinicom, a pinovi na kojima je to omogućeno su B6 (linija za sat) te B7 (podatkovna linija). LCD zaslon koji je korišten je generički LCD 2x16 zaslon, uz dodatak I2C sučelja za jednostavniju komunikaciju s mikroupravljačem. Inercijska mjerna jedinica koja je korištena je MPU6050 IMU, koja u sebi sadrži kombinaciju žiroskopa i akcelerometra, koji su oboje potrebni kako bi se postiglo točno mjerenje nagiba robota. Korišteni koračni motori su NEMA 17, koji su dvofazni te se na korišteni pokretač, A4988, spajaju preko četiri žice. Korišteni pokretači imaju mogućnost smanjivanja veličine koraka motora, tzv. *microstepping,* ili mikro koračanje, ali za realizaciju ovog projekta nisu korišteni, pošto mikro koračanje smanjuje moment motora. Kao izvor napajanja za motore, korištena je LiPo baterija, u shemi označena kao izvor od dvanaest volti. Ona se spaja na pokretače koračnih motora, na linije VMOT i GND. Bitno je napomenuti da su prije nego što se direktno spoje linije izvora spojeni kondenzatori od 100 mikro Farada, koji služe kako pokretači ne bi opazili nagli pad napona kada je motorima u jednom trenutku potrebno više snage. Također, kao sigurnosna mjera, stavljena je i 1N4001 dioda na pozitivni izlaz iz baterije, da se spriječi tok struje u obrnutom smjeru, a ista sigurnosna mjera stavljena je i na izvor napajanja od pet volti. Izvor napajanja od pet volti je generički prijenosni punjač za mobitel. Može se primijetiti da je na izvor napajanja od pet volti, odnosno na pozitivnu liniju postavljena sklopka, kojom se upravlja radom robota. Korisnik odlučuje kada želi upaliti i ugasiti robota pomoću spomenute sklopke, a dodatni indikator da je robot upaljen je LED dioda, koja zasvijetli kada je robot upaljen. Na kraju, mikroupravljač komunikaciju s pokretačima koračnih motora obavlja preko četiri pina: A1 – DIR pin desnog motora (služi za postavljanje smjera rotacije), A2 – STEP pin desnog motora (služi za mijenjanje brzine vrtnje), A3 – DIR pin lijevog motora te A4 – STEP pin lijevog motora.

Električna shema u potpunosti odgovara stvarnom fizikalnom elektroničkom sklopu koji je sastavljen za potrebe ovog zadatka, a realizacija fizikalnog sklopa prikazana je na slikama 3.4 – 3.8.

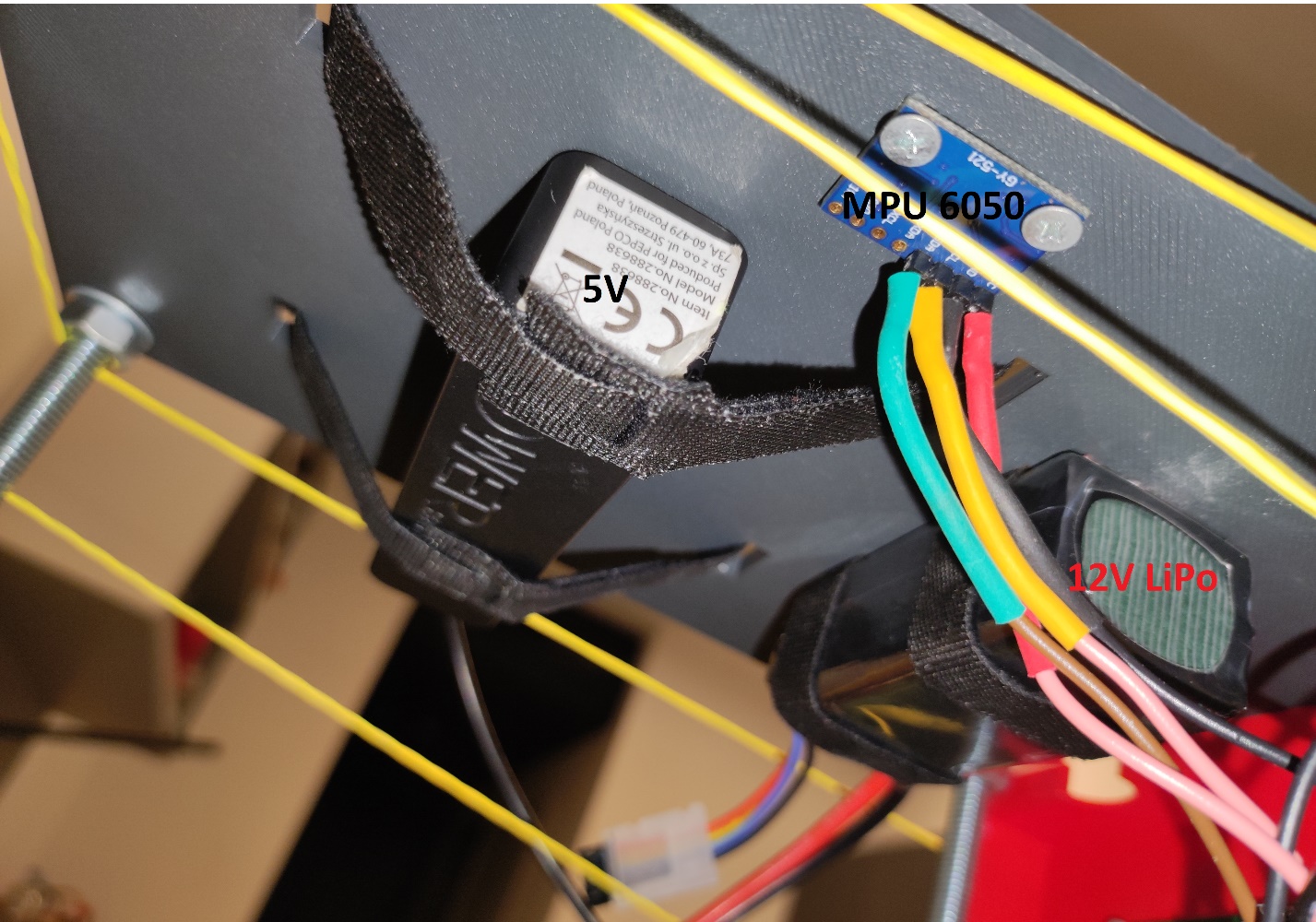
Slika na kojoj se prikazuje na zatvorenom, zid, radni stol

Opis je automatski generiran

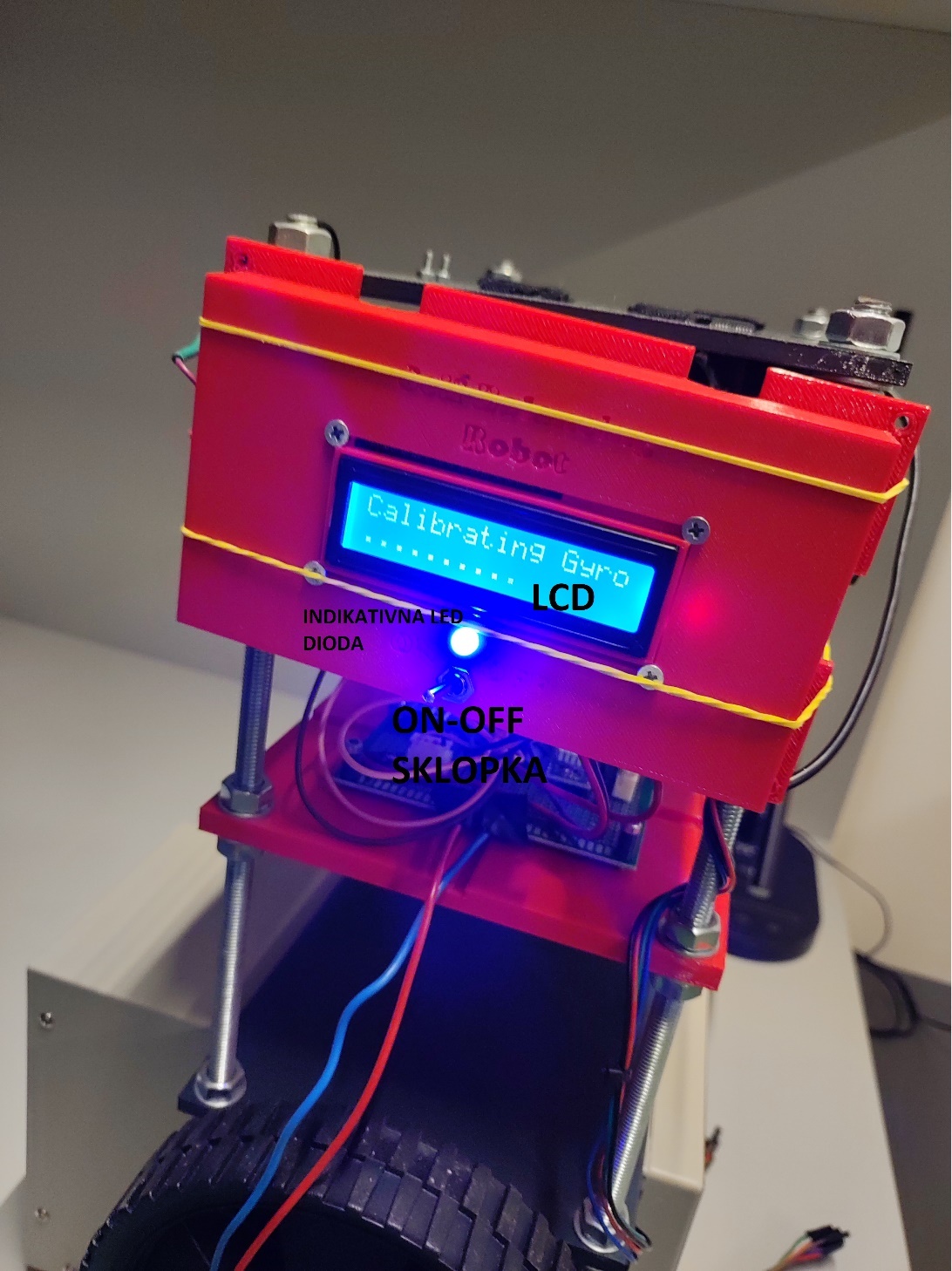
**Slika 3.4.** Prikaz elektroničkog sklopa robota (1).



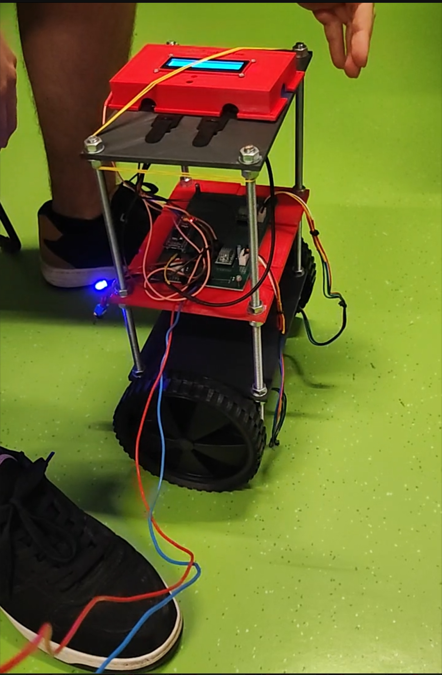
**Slika 3.5.** Prikaz elektroničkog sklopa robota (2).



**Slika 3.6.** Prikaz elektroničkog sklopa robota (3).



**Slika 3.7.** Prikaz robota prilikom rada (1).



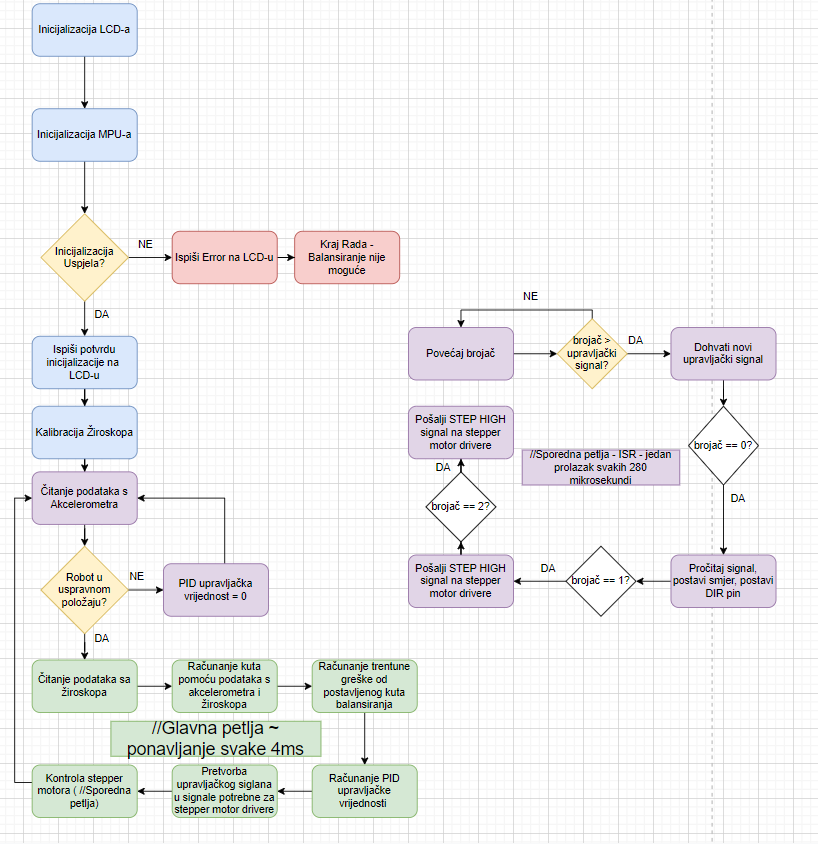
**Slika 3.8.** Prikaz robota prilikom rada (2).

Prilikom fotografiranja gore postavljenih fotografija bio je na mjestu koje ne zaklanja pregled elektroničkih komponenti, pričvršćen elastičnim gumicama, ali inače njegovo mjesto je nad proto pločicom na kojoj je većina komponenti.

U sljedećem potpoglavlju, prikazan je i objašnjen detaljni blok dijagram implementiranog programskog koda.

## Realizacija programskog rješenja

Zadnji dio realizacije rješenja je programsko rješenje. Na slici 3.9 prikazan je detaljni blok dijagram programskog rješenja.



**Slika 3.9.** Prikaz blok dijagrama programskog rješenja.

Nakon što se sklopka za paljenje upali, program kreće s inicijalizacijom. Prvo se inicijalizira LCD zaslon, te se na njemu ispisuje odgovarajuća poruka da je zaslon uspješno inicijaliziran. Nakon toga, inicijalizira se inercijska mjerna jedinica, gdje se prilikom inicijalizacije rezolucija akcelerometra postavlja na od -2g do 2g, rezolucija žiroskopa od -250 stupnjeva po sekundi do 250 stupnjeva po sekundi, te je uključeno dodatno filtriranje podataka prilikom čitanja kako bi se smanjile smetnje u podacima. Ako se prilikom inicijalizacije inercijske mjerne jedinice otkrije greška, na LCD zaslonu se ispisuje odgovarajuća poruka, te sustav neće ući u dijelove programskog koda gdje je napisana glavna logika i potrebno je resetiranje sustava. Ako inicijalizacija uspješno prođe, ispisuje se odgovarajući tekst na ekranu te se kreće u kalibraciju žiroskopa. Kalibracija žiroskopa odvija se tako da se sa žiroskopa pročita tisuću vrijednosti te uzima srednju vrijednost svih očitanja kao početni kut. Pošto žiroskop očitava promjenu kuta po sekundi, bitno je da se prilikom kalibracije žiroskopa robot ne kreće, ali nije bitno u kojem je položaju, tako da prilikom kalibriranja robot može biti i polegnut. Nakon kalibracije, počinje izvođenje glavnog dijela programa. Glavni dio programa podijeljen je u dvije petlje; glavna petlja koja se vrti svake četiri milisekunde, u kojoj se dohvaćaju vrijednosti s inercijske mjerne jedinice, računa trenutni kut robota i na temelju njega računa se PID upravljačka vrijednost za upravljanje motorima te sporedna petlja, kroz koju se prolazi svakih 280 mikro sekundi, koja je zadužena za pravilno slanje smjera i brzine na pokretače koračnih motora na temelju pročitane upravljačke vrijednosti.

U glavnoj petlji se prvo pročita vrijednost akcelerometra, kako bi se dobio približno dobar trenutni kut robota. Ako trenutni kut nije nula stupnjeva, odnosno robot nije u uspravnom stanju, a balansiranje se trenutno ne odvija, vrijednost PID regulatora se postavlja na nulu. Ako korisnik uspravi robota dok se balansiranje ne odvija, i vrijednost s akcelerometra pokazuje da je robot uspravan, tada kreće pravo računanje PID vrijednosti te balansiranje. Kako bi se dobio točan kut, pročitaju se i vrijednosti sa žiroskopa. Kombinacijom vrijednosti sa žiroskopa i akcelerometra, dobijemo najtočnije očitanje trenutnog nagiba robota. Bitno je napomenuti kako prilikom računanja kuta većinu utjecaja ima žiroskop, dok akcelerometar ima jak gotovo zanemariv utjecaj. To je iz razloga što akcelerometar čita sva ubrzanja, što znači da vibracije jako utječu na njegova očitanja. To nije slučaj kod žiroskopa, i on je vrlo otporan na vibracije, ali zbog načina kako je implementiran, njegovo očitanje postaje netočno s vremenom zbog rotacije Zemlje, tako da se on ne može samostalno koristiti za očitanje kuta ako se robot pokušava balansirati duže vrijeme. Iz tog razloga utjecaj očitanja žiroskopa na sveukupni kut je mnogo veći od utjecaja akcelerometra (žiroskop > 99%, akcelerometar < 1%). Kada je dohvaćen trenutni nagib robota, od zadane točke balansiranja se računa trenutna greška, kako bi se mogao izračunati točan upravljački signal. Nakon što je izračunat upravljački signal, potrebno ga je pretvoriti u vrijednost vremenskog razmaka između impulsa koji će se poslati na pokretače koračnih motora. Pošto brzina okretaja korištenih koračnih motora nije linearna u ovisnosti o vremenskom razmaku između impulsa, na temelju poznate karakteristike implementirana je kompenzacija na nelinearnost, te nakon te kompenzacije je signal spreman da bude pročitan od strane sporedne petlje i da se upravlja sustavom u odnosu na upravljački signal PID regulatora. Ovo je zadnji korak u glavnoj petlji, koja će krenuti od početka nakon što četiri milisekunde od početka trenutnog prolaska petlje istekne.

Sporedna petlja je ta koja se brine da pokretači motora dobiju ispravne signale za trenutno postavljeni upravljački signal iz glavne petlje. Ona je prekidna rutina koja se odvija svakih dvjesto osamdeset mikro sekundi. Pri ulasku u sporednu petlju, za svaki pokretač se povećava brojač kojim se prati je li prošli pročitani upravljački signal izvršen ili ne. Ako taj brojač prekorači vrijednost prošlog upravljačkog signala, čita se nova vrijednost, te se brojač postavlja na nulu. Kada je vrijednost brojača nula, znači da je pročitana nova upravljačka vrijednost i iz nje se izvlači smjer vrtnje motora te se šalje na DIR pinove pokretača. Ako je vrijednost brojača jednaka broju jedan, šalje se logička jedinica na STEP pinove pokretača, a ako je vrijednost brojača jednaka broju dva, šalje se logička nula na STEP pinove. Potrebno je samo jednom poslati logičku jedinici pa zatim logičku nulu na STEP pinove, jer će se brojač nastaviti uvećavati prilikom svakog ulaska u sporednu petlju, što služi kao regulacija brzine, jer što je upravljačka vrijednost veća, to će duže trebati da brojač prijeđe staru vrijednost i dohvati novu i time je brzina manja, te obrnuto.

Ovim objašnjenjem predstavljeno je programsko rješenje, a u sljedećem poglavlju može se vidjeti testiranje realizacije projekta, te rezultati samog testiranja. U prilogu dokumenta može se pronaći programski kod opisanog programskog rješenja, kao i poveznica na github repozitorij na kojem je postavljen kod skupa s raznim korištenim dijagramima u drugom i trećem poglavlju, u originalnim formatima za bolju vidljivost.

# TESTIRANJE I REZULTATI

## Metodologija testiranja

Glavno mjerilo uspješnosti je vrijeme u kojem robot uspješno održava ravnotežu, odnosno, koliko sekundi se robot uspijeva održati uspravnim. Prvo testiranje izvedeno je bez poremećaja, a drugo uz vanjski poremećaj, pred robota je postavljena nepomična prepreka u koju se sudara. Što dulje robot uspije održati ravnotežu to će se njegova funkcija smatrati uspješnijom. Robot se nalazi u zatvorenom prostoru gdje je utjecaj vanjskih uvjeta kao što su vjetar i kiša uklonjen, također podloga po kojoj se robot kreće, ravna je i nema nikakvih udubljenja ili izbočenja. U prvom načinu testiranja štoperica se pokreće u trenutku pokretanja robota i zaustavlja se u trenutku njegovog gubitka ravnoteže. U drugom načinu testiranja robot se postavlja pred prepreku i štoperica se pokreće u trenutku sudara robota s preprekom i zaustavlja se u trenutku gubitka ravnoteže robota. Ovakvo mjerenje ponovljeno je 25 puta. Mjerenja su zapisana i iz njih je izračunata srednja vrijednost vremena održavanja kao i standardna devijacija vremena održavanja. Na kraju su rezultati protumačeni i prikazani grafički.

## Rezultati testiranja

|  |  |
| --- | --- |
| Mjerenje | Rezultat [sekunde] |
| 1 | 9.37 |
| 2 | 1.23 |
| 3 | 10.03 |
| 4 | 10.72 |
| 5 | 5.41 |
| 6 | 11.34 |
| 7 | 9.28 |
| 8 | 3.05 |
| 9 | 8.56 |
| 10 | 6.80 |
| 11 | 11.18 |
| 12 | 7.06 |
| 13 | 5.56 |
| 14 | 10.89 |
| 15 | 2.45 |
| 16 | 10.51 |
| 17 | 12.01 |
| 18 | 5.87 |
| 19 | 10.02 |
| 20 | 6.32 |
| 21 | 5.67 |
| 22 | 2.08 |
| 23 | 10.09 |
| 24 | 3.74 |
| 25 | 9.49 |
| Srednja vrijednost | 7.55 |
| Standardna devijacija | 3.26 |

Tablica 4.1

|  |  |
| --- | --- |
| Mjerenje | Rezultat [sekunde] |
| 1 | 3.11 |
| 2 | 2.03 |
| 3 | 1.86 |
| 4 | 3.32 |
| 5 | 3.03 |
| 6 | 2.89 |
| 7 | 2.01 |
| 8 | 1.05 |
| 9 | 2.13 |
| 10 | 2.78 |
| 11 | 3.01 |
| 12 | 2.11 |
| 13 | 3.21 |
| 14 | 1.45 |
| 15 | 1.02 |
| 16 | 2.20 |
| 17 | 2.44 |
| 18 | 3.54 |
| 19 | 2.03 |
| 20 | 2.87 |
| 21 | 1.17 |
| 22 | 3.06 |
| 23 | 1.65 |
| 24 | 1.04 |
| 25 | 2.76 |
| Srednja vrijednost | 2.31 |
| Standardna devijacija | 0.778 |

Tablica 4.2

U tablici 4.1 navedena su sva mjerenja koja su provedena u prvom načinu testiranja bez sudara s preprekom. U tablici 4.2 navedena su sva mjerenja koja su provedena u drugom načinu testiranja uz sudar s preprekom. U prvom stupcu nalazi se broj mjerenja, a u drugom stupcu nalaze se sekunde koje predstavljaju vrijeme održavanja robota. U dva zadnja reda tablice nalaze se srednja vrijednost i standardna devijacija vremena održavanja.

Dijagram 4.1

Dijagram 4.2

Na dijagramu 4.1 prikazana su mjerenja kod prvog načina testiranja bez ikakvog korištenja prepreka, a na dijagramu 4.2 prikazana su mjerenja kod drugog načina testiranja, uz korištenje prepreke. Može se iščitati da je 7.55 sekundi srednja vrijednost održavanja ravnoteže robota bez sudara u prepreku, 12.01 sekundi je najbolje postignuto vrijeme, a 1.23 sekunde najlošije postignuto vrijeme održavanja. Kod drugog načina testiranja, srednja vrijednost održavanja nakon sudara u prepreku je 2.31 sekunda, maksimalna vrijednost je 3.54 sekunde, a 1.02 sekunde je najlošije vrijeme postignuto nakon sudara u prepreku. Standardna devijacija u prvom načinu testiranja, koja se nalazi u tablici 4.1, puno je veća nego kod drugog načina testiranja. Iz tog se može vidjeti da su odstupanja vremena održavanja ravnoteže oko aritmetičke sredine bez sudara u prepreku primjetna kao i razlika između minimalnog i maksimalnog vremena. To se također može iščitati i iz dijagrama 4.1. U drugom načinu testiranja standardna devijacija, koja se može pročitati iz tablice 4.2, je manja, pa se može zaključiti da je srednja vrijednost bolji pokazatelj vremena održavanja koje se može očekivati nakon sudara robota u prepreku. To se također uočava na dijagramu 4.2.

# ZAKLJUČAK

U ovom radu prikazan je način rješavanja problema samo-balansirajućeg robota. Napisan je teorijski osvrt rješenja uz prijedloge funkcioniranja sklopovlja i programa. Također je detaljno objašnjen način konačnog rješenja i korištenih alata pri ostvarivanju tog rješenja. Na kraju je opisana metodologija testiranja i protumačeni su rezultati testiranja. Prednost ovakvog rješenja nad robotima s više kotača je, kao što je već spomenuto u uvodu, veličina robota koja omogućava kretanje u uskim prostorima. Problem se pojavljuje pri održavanju ravnoteže, u slučaju robota u ovom radu, najveći problem bila je njegova konstrukcija koja je imala masu veću od mogućnosti koračnih motora. Masa robota uzrokovala je problem u održavanju ravnoteže kada bi robot promijenio smjer. Motori bi naglo izgubili okretni moment, i robot bi pao. Prijedlog rješenja ovog problema je izgradnja lakše konstrukcije bez metalnih komponenti. U ovom slučaju utjecaj je umanjen programskim smanjivanjem brzine okretaja motora do granica gdje je moment motora maksimalan, a brzina još uvijek dovoljna, ali na kraju ni to nije bilo dovoljno da motori pri promijeni smjera imaju dovoljan zakretni moment da održe konstrukciju uspravnom. Još jedan od problema u ovom radu bilo je podešavanje konstanti PID regulatora, gdje je integralno djelovanje uzrokovalo probleme. Promijene u pojačanju integralnog djelovanja uzrokovale bi primjetne promijene u valjanosti regulatora. To može biti iz razloga što je inercijska mjerna jedinica postavljena na vrh robota, koji je naspram ostalih izvedbi visoko od tla. Svaka promjena kuta je zabilježena tako da postavljanje različitog pojačanja integralnog djelovanja koje pamti sve greške je osjetljivije naspram ostalih pojačanja, ali svakako je i ovdje ulogu igrala teška konstrukcija te nedovoljni momenti motora, pa je moguće da ostale konstante nisu dobro određene i iz tog razloga je pojačanje integralnog djelovanja pravilo problem. Jedan dodatak koji se mogao napraviti za lakše podešavanje konstanti PID regulatora je ugradnja vanjskog sučelja za takvo podešavanje. To sučelje može biti skup od tri potenciometra kojima bi se podešavale konstante ili može biti aplikacija kojom bi se putem bluetooth-a slale informacije o vrijednosti konstanti.

# LITERATURA

[1] Want A Thriving Tour Operator Business? Focus On Segway Tours!, <https://www.orioly.com/thriving-business-focus-on-segway-tours/>, Pristup: 21.09.2022.

[2] J., Park, B., Cho, Development of a self-balancing robot with a control moment gyroscope, SAGE journals, 2, 15, travanj 2018.

[3] S., Kim, S., Kwon, Dynamic Modeling of a Two-wheeled Inverted Pendulum Balancing Mobile Robot, International Journal of Control, Automation, and Systems, br. 13, sv. 4, str. 926 – 933, svibanj 2015.

[4] Integrated Development Environment for STM32,

<https://www.st.com/en/development-tools/stm32cubeide.html>, Pristup: 21.09.2022.

[5] Draw.io,

<https://www.computerhope.com/jargon/d/drawio.htm>, Pristup: 21.09.2022.

# PRILOZI I DODATCI

**P.2, P.3 –** Poveznica na github repozitorij s potpunim programskim kodom te dijagramima i shemama - <https://github.com/bruno-zahirovic/Self-Balancing-Robot>

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.1.** Inicijalizacija LCD zaslona te inercijske mjerne jedinice.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.2.** Beskonačna petlja programa – dodatne informacije o stanju preko LED diode mikroupravljača.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.3.** Funkcija za inicijalizaciju LCD zaslona.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.4.** Funkcija za inicijalizaciju inercijske mjerne jedinice.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.5.** Glavna petlja od četiri milisekunde (1).

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.6.** Glavna petlja od četiri milisekunde (2).

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.7.** Glavna petlja od četiri milisekunde (3).

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.8.** Glavna petlja od četiri milisekunde (4).

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.9.** Sporedna petlja od dvjesto osamdeset mikro sekundi (1).

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

**P3.3.10.** Sporedna petlja od dvjesto osamdeset mikro sekundi (2).