Automatizacija upravljanja brzinom rezanja, posmičnom brzinom i mjerenja snage rezanja na stolnoj glodalici

Kirasić, Oton

SVEUČILIŠTE U ZAGREBU

FAKULTET ŠUMARSTVA I DRVNE TEHNOLOGIJE

DRVNOTEHNOLOŠKI ODSJEK

SVEUČILIŠNI DIPLOMSKI STUDIJ

DRVNOTEHNOLOŠKI PROCESI

OTON KIRASIĆ

AUTOMATIZACIJA UPRAVLJANJA BRZINOM REZANJA, POSMIČNOM BRZINOM I MJERENJA SNAGE REZANJA NA STOLNOJ GLODALICI

DIPLOMSKI RAD

ZAGREB, 2024.

SVEUČILIŠTE U ZAGREBU

FAKULTET ŠUMARSTVA I DRVNE TEHNOLOGIJE

DRVNOTEHNOLOŠKI ODSJEK

AUTOMATIZACIJA UPRAVLJANJA BRZINOM REZANJA, POSMIČNOM BRZINOM I MJERENJA SNAGE REZANJA NA STOLNOJ GLODALICI

DIPLOMSKI RAD

Diplomski studij: Drvnotehnološki procesi

Predmet: Automatizacija i mjerna tehnika u DI

Ispitno povjerenstvo:

izv. prof. dr. sc. Igor Đukić

doc. dr. sc. Matija Jug

doc. dr. sc. Branimir Šafran

prof. dr. sc. Ružica Beljo Lučić (zamjenski član)

Student: Oton Kirasić

JMBAG: 0068234067

Datum odobrenja teme: 23.03.2023.

Datum predaje rada: 20.09.2024.

Datum obrane rada: 25.09.2024.

Zagreb, 2024.

DOKUMENTACIJSKA KARTICA

Naslov	AUTOMATIZACIJA UPRAVLJANJA BRZINOM REZANJA, POSMIČNOM BRZINOM I MJERENJA SNAGE REZANJA NA STOLNOJ GLODALICI
Title	Cutting speed, feed speed and cutting power measurement automation on woodworking spindle moulder
Autor	Oton Kirasić
Mjesto izrade	Sveučilište u Zagrebu, Fakultet šumarstva i drvne tehnologije
Vrsta objave	Diplomski rad
Mentor	izv. prof. dr. sc. Igor Đukić
Godina objave	2024.
Obujam	6 poglavlja, 71 stranica, 42 slike, 19 navoda literature, 1 prilog
Ključne riječi	Industrija 4.0, Raspberry Pi, automatizacija, posmična brzina, stolna glodalica
Key words	Industry 4.0, Raspberry Pi, automation, feed speed, table router
Sažetak	Drvna industrija definirana kao niskoprofitna grana kasni u implementaciji tehnologija koje donosi Industrija 4.0. Načela Industrije 4.0 se temelje na automatizaciji sustava, korištenju interneta, naprednih CAD/CAM programa te ostalih u svrhu stvaranja individualnog proizvoda po cijeni serijskog proizvoda. Rad obrađuje tematiku programiranja u jeziku Python i preinake klasične analogne stolarske glodalice u računalno upravljani stroj. Trofazni asinhroni elektromotori glavnog i posmičnog gibanja su kontrolirani pomoću mikroračunala, potrebnih konvertera i frekventnog pretvarača. Provedeno je mjerenje i analiza podataka posmične brzine, brzine rezanja i simulacija mjerenja potrebne djelatne snage.
Summary	The wood industry identifies as low-profitable buisness and therefore is more late then others in implementation of technologies provided by Industry 4.0. The core principles of Industry 4.0 movement are system automation, the use of the Internet, advanced CAD/CAM software and other in purpose of production a customizable product within costs of a mass produced product. This paper presents topics of Python programming and improvment of classic woodworking table router into computer-controlled machine. The three-phase motors of machine are controlled by microcomputer, necessary signal converters and a variable frequency drive. Measurements of feed speed, cutting speed and cutting power simulation has been done and the collected data were analyzed.

	IZJAVA **O AKADEMSKOJ ČESTITOSTI**	**OB FŠDT 05 07**
		Revizija: 2
		Datum: 29.04.2021.

„Izjavljujem da je moj diplomski rad izvorni rezultat mojega rada te da se u izradi istoga nisam koristio drugim izvorima osim onih koji su u njemu navedeni“.

U Zagrebu, 20. rujna 2024. godine

_____

vlastoručni potpis

Oton Kirasić

Sadržaj

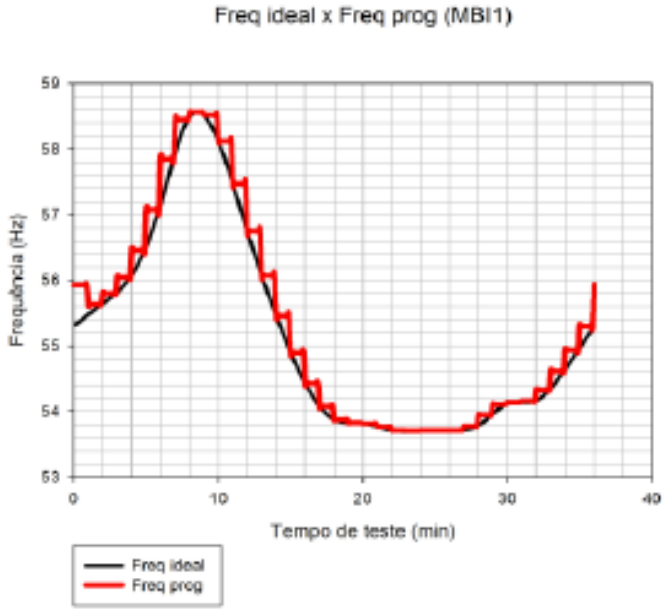
1. UVOD

Posljednjih 2 desetljeća se sveukupna industrija, među kojima i drvna, suočava sa poslovnim poteškoćama uzrokovanim eksponencijalnim razvojem tehnologije i povećanjem zahtjeva nad kvalitetom proizvoda. Problematika otežanog praćenja tržišta i zahtjeva kupaca je zasnovana na trendu globalizacije i početka četvrte industrijske revolucije (Industrija 4.0). U tijeku prilagodbe proizvodnje tehnologiji treće industrijske revolucije već se počela uvoditi još novija. Ovakav tip napretka nije dosad nikad zabilježen u povijesti, tj. istovremeni globalni utjecaj na različite segmente ljudskog života. Današnja generacija svjedoči uvođenju digitalizacije u sve procese proizvodnje koja uzrokuje takvu promjenu da postojeći sustavi više nisu dovoljni za zadovoljenje zahtjeva kupaca (Majstorović i dr., 2022). Kako bi se obnovilo stanje, tvrtke ubrzanim korakom uvode nove tehnologije za automatizaciju proizvodnje koja bi im smanjila jednični trošak proizvoda i povećala kvalitetu naspram konkurenata. Na krilima želje za automatizacijom koja sigurno isplativa nakon niza godina, ali ima veliku početnu cijenu investicije, pojavljuju se slučajevi prenamjene i restauracije starijih numerički kontroliranih (NC) strojeva u računalno numerički kontrolirane (CNC). U slučajevima kada tvrtka nije u mogućnosti investirati u potpunu automatizaciju nije neobično da se na starijim strojevima, koji su često još uvijek u zadovoljavajućem stanju, naprave preinake sa senzorima, mikrokontrolerima, mikroračunalima, novim elektromotorima upravljanih preko frekventnih pretvarača i slično. Ako se tom stroju upari rabljena robotska ruka koja će služiti za umetanje i odlaganje materijala, dobivaju se djelomične karakteristike modernog CNC stroja uz znatno manju investiciju. Provedene analize tržišta pokazuju da i dalje postoje poduzetnici koji su nesigurni i distancirani od ideje potpune digitalizacije proizvodnje ili kompjuterskog upravljanja stroja zbog kojih je potrebno detaljnije razjasniti pripadajuću tematiku (Peko, 2015). Sporijom tranzicijom tržišta u računalnu eru je poduzetnicima bilo donekle dopušteno zanemarivanje novih tehnologija, no prisustvom interneta i globalizacijom svjetsko tržište se prebrzo mijenja. Slabim ili nikakvim uvođenjem nove tehnologije nije garantiran samo pad prihoda u nekom vremenu, već potpuno gašenje proizvodnje.

1.1. Srodni projekti automatizacije

Trenutno je putem interneta moguće pristupiti velikom rasponu informacija o programiranju, spajanju hardverskih komponenti i zaobilaženju problema koji se pritom stvore. Zajednica dijeli veliku količinu informacija u znaku želje i potrebe za dodatnim brojem kvalificiranih ljudi u područjima robotike, razvijanja softvera, elektronike i energetike. Materijale je moguće naći na stranicama proizvođača komponenti, forumima na kojima obitavaju hobisti, ali i online knjižnicama akademske zajednice poput Google Scholara i hrvatskih nacionalnih knjižnica Dabar i Hrčak. Proučavanjem ove literature moguće je vlastiti projekt unaprijediti idejno i izvedbom. Pretpostavljene hipoteze ovog rada mogu se donekle opravdati zaključcima već provedenih sličnih projekata. Jedan od takvih su proveli Loureiro i

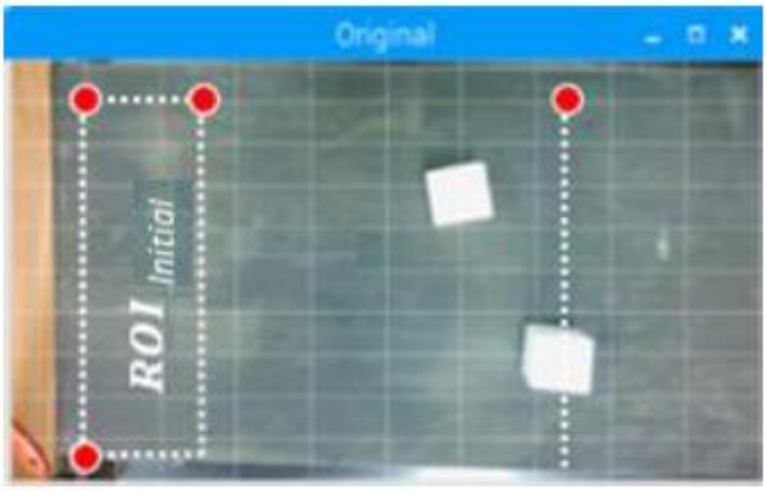
dr. (2024) u Brazilu. Njihov projekt se odnosio na problematiku utroška energije sustava prskalica u poljoprivredi. Postojeći sustav se sastojao od pumpe pogonjene elektromotorom spojene na veći mobilni spremnik s vodom. Problematika se vidjela u tome što je pumpa radila istom konstantnom frekvencijom vrtnje bez obzira na razinu vode u spremniku. Kako bi se uštedjelo na utrošku vode i električne energije bilo je potrebno osigurati manju frekvenciju vrtnje elektromotora prilikom visoke razine vode i tlaka u spremniku te obrnuto pri niskoj razini vode u spremniku. Hardverska nadogradnja sastojala se od mikroračunala Raspberry Pi 3 model B koje je služilo kao udaljeni server, mikrokontrolera Arduino Uno u kombinaciji s potenciometrom otpora 200 kΩ, mobilnim TP-Link 3G routerom te WEG frekventnim pretvaračem. Korišteni program je radio na 37 biranih frekvencija u rasponu od 3-60 Hz te se signal generirao na temelju podataka senzora razine vode u spremniku. Kontrola je bila spojena na 7 odvojenih pumpi čiji su podaci za traženom frekvencijom bili prije poznati. Analizom spremljenih podataka (frekvencija) u .csv formatu, bilo je moguće utvrditi korelaciju tražene i generirane frekvencije. Najniža razina Pearsonove korelacije je iznosila 0,9786, što je više nego zadovoljajuće za ovakav sustav. Grafički prikaz korelacije idealne i generirane frekvencije tijekom vremena jednog slučaja prikazan je na slici 1.



Slika 1. Prikaz idealne i generirane frekvencije (Loureiro i dr., 2024)

Osim problema utroška energije i optimizacije komponenti, u industriji se više javlja problem dostupnosti radne snage za poslove sortiranja, slaganja i pakiranja proizvoda. Neki autori navode i problematiku nedostatka visoko kvalificirane radne snage koja bi omogućila primjenu naprednih sustava (Loureiro i dr., 2024). Problematiku slaganja poluproizvoda u drvnoj industriji su obrađivali Costa i dr. (2019). Njihov projekt je prepoznao trenutačne poteškoće industrije koja je u intenzivnoj borbi za većom konkurentnošću i opstankom te su kao rješenje predstavili implementaciju tehnologije Industrije 4.0. Napretkom robotike, tehnologije procesuiranja slikovnih podataka i umjetne inteligencije odlučili su izraditi projekt prihvati i odloži (engl. *Pick and Place*) robota uparenog s kamerom za prepoznavanje predmeta i transportnom trakom pogonjenom asinhronim elektromotorom kontroliranim s frekventnim

pretvaračem. Kamera je korištena za prepoznavanje drvenih blokova koji su bili transportirani na pokretnoj traci, a dobivena snimka mora proći kroz niz digitalnih filtera kako bi se uspjeli odrediti parametri nadolazećeg objekta. Neki od njih su smanjivanje utjecaja pozadine, utvrđivanje rubova objekta, detekcija u crno-bijelom spektru, definiranje orijentacije objekta i drugo. Valja napomenuti da cijeli prikaz kamere nije pod područjem analize podataka već samo jedan dio nazvan područjem interesa. Ovaj princip rada je prikazan na slici 2,



Slika 2. Prikaz područja interesa (ROI) kamere(Costa i dr., 2019)

gdje je područje interesa vidljivo na lijevoj strani kadra te ono ne zauzima više od petine površine kadra. Na ovaj način se umanjuje količina obrađenih podataka, a prepoznati objekti se dalje prate, tj. nove koordinate računaju u skladu s brzinom kretanja transportne trake. Za upravljanje robotskom rukom i obradu podataka u sklopu sustava strojnog vida (engl. *Machine Vision*) isto je korišteno mikroračunalo Raspberry Pi 3 model B s programom napisanim u C++ programskom jeziku. Pomoću frekventnog pretvarača generiran je upravljački signal za regulaciju posmične brzine transportne trake, a na temelju obrađenih podataka dobivenih pomoću kamere i obradom slike, regulacijski sustav je davao signale robotskoj ruci.

Slika 3. Prikaz eksperimentalne robotske ruke upravljane mikroračunalom Raspberry Pi, pomoću sustava strojnog vida, u procesu automatskog sortiranja na trakastom transporteru (Costa i dr., 2019)

Jedan od problema koji se javljaju kod sustava koji uključuju strojni vid je i brzina obrade slike u sustavu povratne veze. U predstavljenom slučaju kamera je snimala procesni prostor (transportnu traku) s 30 slika u sekundi. Obrada slike prosječno je trajala 0,21 sekundu, a maksimalna brzina trake bila 1,7 m/min. Koordinate objekta ažurirane su svakih 0,59 cm pomaka. Za industrijske potrebe, u većini slučajeva ovo su premale brzine, ali poznato je da postoje industrijski sustavi strojnog vida koji zadane operacije mogu obavljati bitno brže. U ovom slučaju razvijen je sustav relativno male cijene, ali s obzirom na ograničene mogućnosti odabranog mikroračunala za zadane zahtjeve vezano uz obradu slike u sustavu strojnog vida, moguće je ostvariti prezentirane rezultate. Naravno, u nekim slučajevima i takve relativno male brzine, u nekim segmentima drvne industrije, kao što je prihvat i slaganje drvenih četvrtača su i više nego prihvatljivi. Analiza ovih radova od mnogih je potvrda da akademska zajednica intenzivno radi na približavanju tematike automatizacije i njoj srodnih disciplina, poslodavcima u industrijskom sektoru kao i studentima koji se orijentiraju u tom smjeru.

1.2. Industry 4.0

Pojam Industrija 4.0 je osmišljen u Njemačkoj 2011. godine na sajmu u Hannoveru. Glavno načelo ove revolucije je uvođenje novih sustava automatizacije koji pomoću interneta kao glavnog prenositelja informacija, kibernetsko-fizičkih sustava (engl. *cyber-physical systems*), računarstva u oblaku (engl. *cloud computinga*) i interneta stvari (engl. *Internet of things*) te umjetne inteligencije (engl. *Artificial Intelligence - AI*) sudjeluju u stvaranju „pametne proizvodnje“ (engl. *smart production*) (Kolberg i dr., 2015). H. Rauen, zamjenik direktora njemačke udruge industrije (*VDMA*) je za Industriju 4.0 rekao: „Implementacija će biti postepena te neće biti velikog praska uvođenja nove tehnologije, ali kada se nakon 10 godina pogleda unazad bit će vidljivo da se svijet značajno promijenio“ , što je danas nakon 13 godina i potvrđeno (Majstorović i dr., 2019). Nema segmenta ljudskog života u kojem je izostala promjena. Svijet velikom brzinom grabi prema automatizaciji, digitalizaciji i robotizaciji. Prema Majstoroviću i dr. (2019), ekonomska načela Industrije 4.0 su:

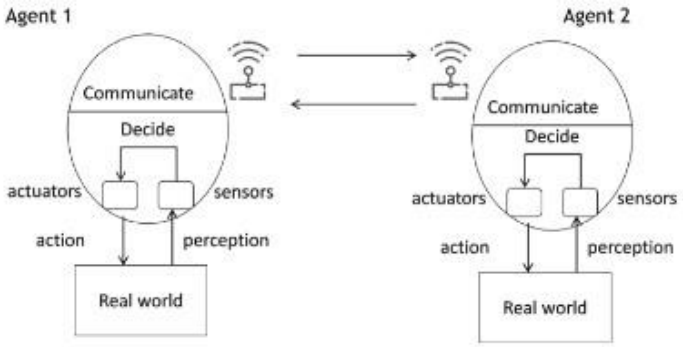
- Pametni dizajn i planiranje proizvoda i tehnologija - napredovanjem virtualne stvarnosti (engl. *virtual reality*) i proširene stvarnosti (engl. *augmented reality*) u kombinaciji sa CAD i CAM programima će dizajniranje proizvoda otići u „pametnu eru“. CAM programi će direktno u stvarnom vremenu komunicirati sa 3D printerima u razvoju prototipova.
- Pametni strojevi - putem mnoštva senzora strojevi će prikupljene podatke slati na obradu kako bi njihov digitalni blizanac (engl. *digital twin*) na drugoj lokaciji mogao pružiti bolju uslugu na temelju prikupljenih podataka.
- Pametno kontroliranje - senzorska kontrola širokog spektra parametara na jednom korisničkom sučelju za olakšano upravljanje.

- Pametni menandžment (upravljanje) - fleksibilna raznovrsna proizvodnja je razvijena pomoću kibernetsko-fizičkog sustava. Korisnici bi mogli upravljati strojevima ili robotima pomoću sučelja preko pametnih telefona.
- Pametno tempiranje - odnosi se na obradu prikupljenih podataka pomoću naprednih algoritama kako se stvorio pouzdan i hijerarhijski poredan rok izrade.

Cijeli idejni koncept Industrije 4.0 se zasniva na stvaranju pametne tvornice (engl. *smart factory*) u kojoj sofisticirani strojevi opremljeni senzorima i komuniciraju putem internetske veze. Oni svoje podatke spremaju u računalni oblak gdje se obrađuju na serverima velike moći obrade podataka. Cilj revolucije koristeći ova načela je stvaranje „pametnog proizvoda“ kojeg će krajnji korisnik dizajnirati na svom uređaju i kao narudžbu poslati pametnoj tvornici na izradu. To je princip personaliziranog proizvoda po cijeni onoga iz masovne proizvodnje. Na slici 4 prikazan je model rada pametne tvornice u kojoj se energija optimalno troši, obradak se prati u stvarnom vremenu, a proizvodne karakterstike mogu biti promijenjene u roku nekoliko sekundi.

Slika 4. Model rada pametne tvornice (Izvor: https://www.altamira.ai/blog/industry-4-0-smart-factory/)

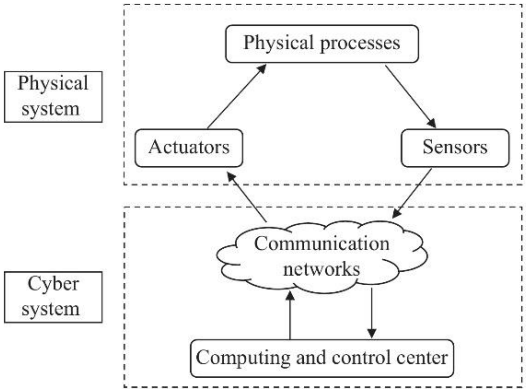
Podaci obrađeni složenom algoritamskom mrežom spremaju se prema načelu uzroka i posljedice kako bi strojevi u budućim slučajevima varijabilnosti proizvodnje mogli napraviti ispravnu reakciju. Ovi procesi se zbirno nazivaju strojno učenje (engl. *Machine Learning*). Cijelim sustavom tvornice upravlja kibernetsko-fizički sustav podržan umjetnom inteligencijom uz ljudsku kontrolu putem detaljnog korisničkog sučelja. Strojevi su međusobno povezani najčešće bežičnim putem industrijskog interneta, a sustavno posloženi prema hijerarhiji važnosti u tvornici. Kako je cijeli sustav snimljen i njegovi podaci spremljeni u oblak, stvara se digitalni blizanac (engl. *digital twin*) fizičkog svijeta koji na temelju simulacija upravljanih umjetnom inteligencijom stvara uzročno-posljedične veze za alternativne situacije u tvornici. Ovakav tip podataka će biti dostupan onim strojevima na drugim lokacijama koji se bave istim problemom. Tako će preventivnim radnjama biti u mogućnosti eliminirati potencijalne zastoje u proizvodnji (Ramos-Maldonado i dr., 2022). Na slici 5 je prikazan pojednostavljeni prikaz komunikacije digitalnih blizanaca koji razmjenjuju informacije o proizvodnji s različitih lokacija.



Slika 5. Komunikacija digitalnih blizanaca (Ramos-Maldonado i dr., 2022)

Uz pomoć svih proizvodnih parametara će se omogućiti veća varijabilnost proizvoda tvornice uz minimiziranje troškova vezanih za namještanje strojeva i puštanja linije u rad. Narudžbe se mogu obrađivati odmah nakon primitka uz garanciju točnog roka isporuke. Nove tehnologije i pojmovi koje generira Industrija 4.0 u velikom broju slučajeva nisu poznate prosječnom korisniku pa ih valja detaljnije objasniti.

- Internet stvari (engl. *Internet of things*), - pojam koji se odnosi na međusobno povezivanje stvari, ljudi i uređaja pomoću interneta, žično ili bezično, u svrhu maksimiziranja učinkovitosti radnje i individualnom pristupu korisniku ili proizvodu. Najjednostavniji primjer je na razini prosječnog kućanstva gdje hladnjak, robotski usisavač, televizor i klima uređaj budu povezani na *WLAN* mrežu. Analizom podataka o načinu i vremenu korištenja uređaja, nakon što se stvori uzorak ponašanja, uređaji mogu ekonomično trošiti energiju i prilagoditi parametre rada svom korisniku.
- Internet usluga (engl. *Internet of Services*) - označava internet kao izvor podataka koje korisnik treba kako bi savladao vještine potrebne za upravljanje određenom robom ili uslugom i tako ostvario korist.
- Kibernetsko-fizički sustav (engl. *Cyber-Pyhisical System-CPS*) - sustav koji temelji na računalnim algoritmima obrade podataka u kojemu su računalne i fizičke značajke povezane na višoj razini. Ono što razlikuje *CPS* od klasičnog računala je veća moć obrade podataka, tj. ulaza koji se zaprimaju sa senzora u tvornici i neometano upravljanje više strojeva odjednom. Najpoznatiji primjer *CPS-a* je autonomno vozilo. Primjer osnovnog oblika *CPS-a* koji svoje podatke izmjenjuje sa kontrolnim centrom i na temelju kojih donosi odluke.



Slika 6. Schematski prikaz kibernetsko-fizičkog sustava (Duo i dr., 2022)

- Računarstvo u oblaku (engl. *Cloud Computing-CC*) - označava pristup i korištenje podatkovnih centara (engl. *data center*) koji pružaju usluge korištenja servera, pohrane podataka, analize, dostupnih znanja za obradu vlastitih informacija. Praktički se radi o outsourcingu resursa potrebnih za analizu kako bi se smanjili jedinični troškovi proizvoda. U protivnom, bilo bi neisplativo da jedna tvornica posjeduje svoje superračunalo za vođenje poslovanja.
- Velika količina podataka (engl. *Big Data*) - ovaj pojam se referira na oblike podataka koji su preveliki i preopširni da bi se obrađivali na tradicionalan način. Za njihovu obradu su potrebni složeniji algoritmi i procesori višeg razreda moći. Dobiveni rezultati se koriste u vođenju tvornica visokog stadija autonomije, analizama tržišta ili poput razvitka olakšanog korištenja aplikacija ili tražilica krajnjem korisniku.
- Umjetna inteligencija (*engl. Artificial Intelligence-AI*) - oblik inteligencije koju koriste strojevi, a stvorio ju je čovjek. Trenutna dostupna inteligencija u sklopu svojih dopuštenih intervala koristi dostupne podatke i sustave na internetu u svrhu učenja i rješavanja problema koji su joj prezentirani. Po svojim algoritmima se ne razlikuje od načina ljudskog razmišljanja. Obzirom na ogromnu internetsku pretraživačku moć i obradu odabranih podataka u nekoliko sekundi postaje odličan resurs za rješenja na jasne, objektivne i logičke postavljene zadatke.
- Strojno učenje (engl. *Machine Learning-ML*) - grana umjetne inteligencije koju definira sposobnost stroja da imitira ljudsko učenje, tj. da usavršava svoje algoritme na temelju pokušaja i rezultata operacije koje mjeri pomoću svojih senzora. Ovim principom se smanjuju tvorničke pogreške i povećava proizvodnost stroja.
- Informacijsko-komunikacijske tehnologije (engl. *Information and Communications Technology-ICT*) - pojam koji u širokom smislu obuhvaća svu komunikacijsku tehnologiju poput interneta, bežičnih mreža, mobilnih telefona, računala i sve ostale audio-video medijske opreme. Odnosi se na svako moguće sredstvo prijenosa informacije od njenog generatora do korisnika. Prvim početkom ove tehnologije se smatra razvitak tiskarskog stroja.
- Automatsko vođeno vozilo (engl. *Automated Guided Vehicle-AGV*) - vozilo koje je zapravo mobilni robot nižeg stupnja komplikacije. Služi za transport materijala unutar velikih industrijskih zgrada, a za navigaciju koristi obilježene linije na podu, radiovalove, lasersko mjerenje udaljenosti ili tome sličnu tehnologiju.

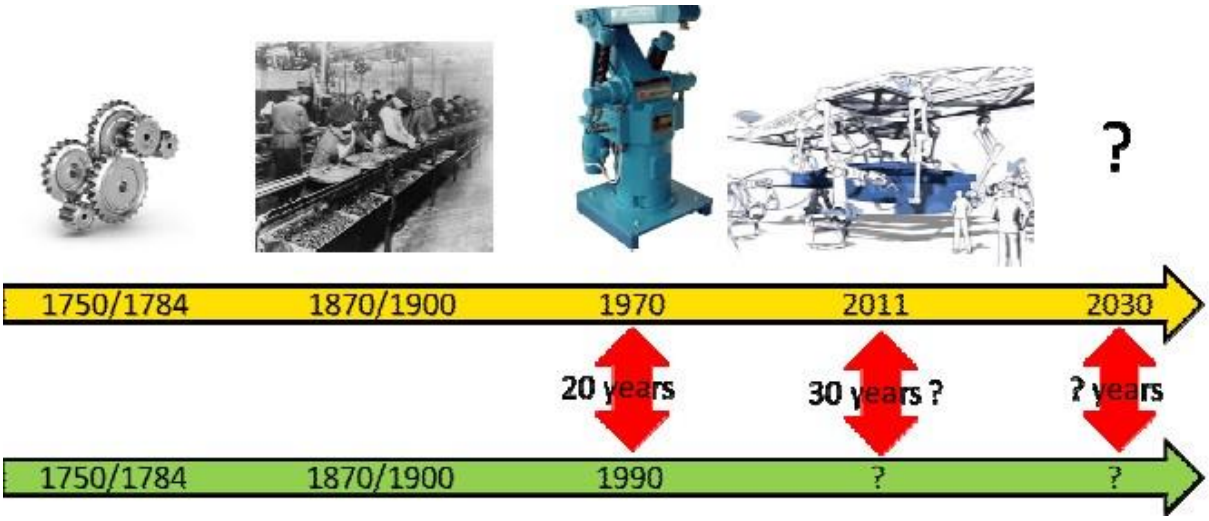
Također postoje sličnosti metodologije Industrija 4.0 i *Lean production* principa zasnovanog u Japanu 1950-ih godina. Prema Kolberg i Zühlke (2015) Toyotin inženjer Ono je stvorio skup metoda i principa rada koje se fokusiraju na način kako proizvodnja mora biti organizirana i radnje obavljenje kako bi se postiglo najkraće vrijeme izrade uz minimalne troškove sa najvišom kvalitetom. Ovaj princip rada karakterizira maksimalno smanjenje otpada uz povećanje produktivnosti do 25 %. Zbog toga je još i danas atraktivan poslovni model za mnoga poduzeća. Prilikom svog nastanka je *Lean Production* kvalitetno odgovarao na promjene tržišta s dobrom reakcijom u sistemima maloserijske i velikoserijske proizvodnje. Daljnim razvojem tržišta i otvaranjem svjetske scene 1980-ih i 1990-ih godina je *Lean production*

lagano dosegao svoje limite zbog novog načina tržišnog zahtijevanja. Njegovi rezultati su bili kvalitetni dugovječni proizvodi koji se uglavnom masivno skladištili čekajući narudžbu. Dizajn i trendovi su bili sporiji te je takav način proizvodnje bio općeprihvaćen. Danas se moda, trendovi, dizajn i želje mijenjaju na mjesečnoj bazi te su preostale rijetke industrije koje mogu sa sigurnošću proizvoditi jedan te isti model na skladište u velikim količinama. Analizom načela na kojima se zasniva Industrija 4.0 se može napraviti poveznica s *Lean Production* kao spoj principa s automatizacijom izrade. Težnja najvećoj kvaliteti, ubrzanje procesa, minimiziranje otpada, brzi odgovor na tržišne zahtjeve dovode do stvaranja pojma *Lean Automation*. *Lean Automation* kombinira osnovna načela *Lean Production* i tehnologija koje nudi Industrija 4.0. Bitno je znati da su ovi principi ostvarivi zbog isprepletenosti s pojmovima poput *JIT* (engl. *Just-In-Time*) i japanskog termina *Kaizen*. Prema Kootanaee i dr. (2013) princip *JIT* se odnosi na japansku poslovnu filozofiju koja zahtijeva ispravne materijale u točnoj količini i kvaliteti pozicionirane na točnom mjestu u određenom trenutku. Smatra se da ispravno korištenje *JIT* sistema rezultira u povećanju proizvodnosti i kvalitete s usporednim smanjenjem jediničnih troškova i otpada. Razlika *JIT-a* je bila da njegovi proizvodi nisu zahtijevali velika skladišta zbog ciljanog tempiranja vremena početka proizvodnje i isporuke. *Kaizen* metodologija se bazira na konstantnom napretku poslovanja. Onog trenutka kada je japanska industrija osjetila pad stručnih radnika predstavljen je model cjeloživotnog napretka unutar iste tvrtke (Singh i Singh, 2009). Radnici su bili potaknuti rastom plaće i pozicije u hijerarhiji ako se njihov prijedlog unaprijeđenja određenog radnog mjesta ispostavi korisnim. Ovaj princip je bio potpuna suprotnost dotadašnjem strogo gledanom hijerarhijskom odnosu prema radniku, a većinu radnika je potakla da obavljaju dužnosti odgovorno u želji za napretkom. Svi navedeni principi su u nekom obliku uvedeni u ključne značajke *Lean Automation* i Industrije 4.0. Od svakog su preuzete najbolje stavke uz koje je evolucijski razvijeno do današnjeg oblika. Ključni novi element je *CPS* koji je sposoban voditi cijeli proces uz ljudsko nadgledanje i minimiziranje fizičkog rada. Rezultat je proizvod visoke kvalitete, maksimalno iskorištenje sirovine, serije s malom varijabilnošću te isporuka u točno definiranom najbržem roku. Principe *Lean Automation* su uvele njemački *Würth* i danski *Lego* (Kolberg i Zühlke, 2015).

1.2.1. Industry 4.0 u drvnoj industriji

Usvajanje novih tehnologija u drvnoj industriji je tradicionalno usporeno i kasni za ostalim industrijama. Ovakva pojava ima i svoja djelomična opravdanja. Drvna prerada je oblik proizvodnje sa niskom dodanom vrijednošću kojoj je potreban velik broj obrtaja kapitala kako bi se poslovalo pozitivno. Nadalje, drvo je materijal pun varijabilnosti u estetskim, mehaničkim i kemijskim svojstvima te je zasada teško stvoriti bazu podataka kojom bi stroj uz pomoć senzora mogao klasificirati drvnu građu s istom postotkom uspješnosti kao čovjek. Nepovoljna je situacija što je drvo anizotropan i higroskopan materijal sklon oscilacijama oblika pri promjeni ambijentalnih uvjeta. Ipak, razvijaju se sistemi na principu strojnog učenja koji daju obećavajuće rezultate. U testiranjima gdje je baza podataka bila 6000 zapisa o klasifikaciji listova furnira sustav opremljen brojnim senzorima čije odluke kontrolira *CPS* bio je točan u

80% slučajeva. Istom metodom je bilo obavljeno testiranje i na pločama MDF-a. Baza podataka je bila 14 000 mjerenja, a točnost klasifikacije je bila 95% što je razumljivo više obzirom da ploče vlaknatice imaju uniformnija svojstva od masivnog drva (Ramos-Maldonado i Aguilera-Carrasco, 2022). Nove tehnologije se još uvijek pokušavaju prilagoditi drvnoj industriji, ali ne mogu garantirati funkcionalnost sustava u svim slučajevima. Prema Landscheidt i Kans (2016), promatrana švedska drvna industrija djelomično još uvijek nije spremna na sljedeći korak automatizacije koju pruža Industrija 4.0. Nespremnost se odnosi na segment izrade namještaja u kojoj je zastupljen način rada sa puno fizičkog rada te velike oscilacije proizvoda i dimenzija, kao i problematičnost automatizacije finalne obrade brušenja i lakiranja gdje strojna obrada još nije našla načina da kvalitetno nadmaši ručnu obradu. Preporuka o stagnaciji inovacija se temelji na tome da ni ostale visokorazvijene zemlje nisu ušle u intezivan proces revolucije industrije. Koristi se načelo promatranja konkurenata u nadi da drugi posluže kao testni uzorak. Tim modelom švedska drvna industrija koja sudjeluje sa 12 % u bruto domaćem proizvodu države i drži 18 % svjetskog tržišta piljene građe promatra stanje u Danskoj. Prema ovoj analizi, kada bi švedska prerada drva dosegla razinu automatizacije koju posjeduje Danska, prognoze su da mogla povećati izvozni promet za 17 %. Prema praćenju trendova uvođenja tehnologije u svim industrijskim revolucijama, švedska analiza pokazuje da se svakoj revoluciji sve više kasnilo. S takvim trendom potpuna automatizacija u Švedskoj bi trebala nastupiti za 30 godina. Na slici 7 je vidljiv prikaz trenda zakašnjenja uvođenja novih tehnologija unazad posljednje 3 industrijske revolucije na nacionalnoj razini.

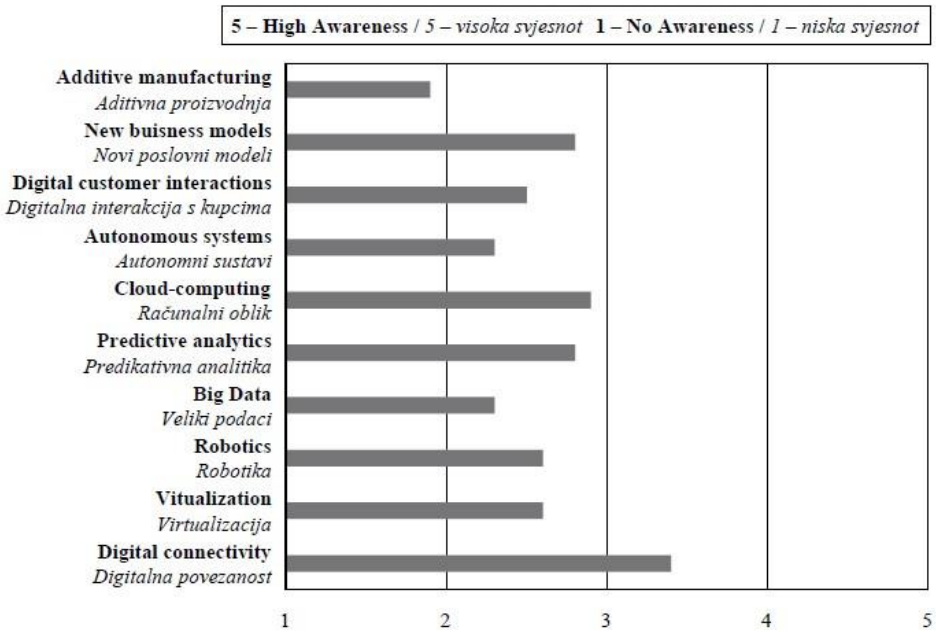


Slika 7. Prikaz kašnjenja za uvođenjem novih tehnologija (Landscheidt i Kans, 2016)

Usprkos tome što je vidljivo da automatizacija dugoročno donosi isplativost investicije, slučajevi potpune automatizacije su vidljivi samo na primjerima primarne prerade drva. Takvi primjeri su postignuti ponudom fiksnih dimenzija piljene građe, bez mogućnosti rada po individualnim željama, korištenjem trupaca četinjača manjih dimenzija kojima se lakše manipulira i čije drvo je manje osjetljivo na greške prilikom sušenja. Na slici 8 je prikazan primjer unutarnjeg transporta svježe piljene građe u sušionice pomoću automatskog vođenog vozila u jednoj njemačkoj pilani.

Slika 8. Automatsko vođeno vozila za punjenje sušionice (Izvor: https://www.youtube.com/watch?v=qIcqRVl7N5c&ab_channel=PollmeierMassivholz)

Pilanska sirovina četinjača generalno ima jednoličniju formu oblika, tj. sličniju pravilnom valjku (krnjem stošcu) nego što je to sirovina listača. Greške četinjača se mogu okarakterizirati kao lokalne za razliku od globalnih kod listača. Globalne greške značajnije mijenjaju fizički izgled sirovine te utječu na kvalitetu drva kroz veći broj piljenica. Iz tog razloga su današnji primjeri automatizacije pilanske proizvodnje uglavnom za preradu mekog drva. Pomoću definiranih normi dopuštenog broja kvrga, pukotina i tražene širine goda skeneri mogu odrediti klase na temelju kojih se generiraju barkodovi s podacima o piljenici za prodaju. U sličnoj problematičnoj situaciji se nalazi i američko tržište primarne prerade drva. Postoje primjeri djelomične uporabe naprednih tehnologija, no većinu i dalje koči visoka cijena koštanja i manjak educiranog kadra koji bi implementirao nove tehnologije. Prema provedenoj anketi (Legg i dr., 2021) potvrđeni su očekivani loši podaci o znanju dostupne tehnologije Industrije 4.0. Na anketu poslanu na 444 adrese od kojih su sve različite pilane dobiveno je samo približno 25 % odgovora što ukazuje na potpunu nezainteresiranost o temi. Iz analiziranih podataka je vidljivo da 53 % uzorka koji se odnosi na poduzeća sa 100 ili više zaposlenika ima više kapaciteta za educiranje radnika, već posjeduju stručnjake za određena područja i prepoznaju potencijale Industrije 4.0. Samo 6 tvornica je izjavilo da koriste pametno upravljanje ili da je napisana strategija investicija u ovaj segment za nekoliko godina. Gotovo 80 % ispitanika nije bilo uopće upoznato sa osnovnim pojmovima poput pametne tvornice, velikih količina podataka, Industrija 4.0 i slično. Na slici 9 je prikazana razina svjesnosti o aktualnim principima koje donosi Industrija 4.0 iz koje se može zaključiti da treba poticati razinu obrazovanja vodećeg kadra američkih pilanara.

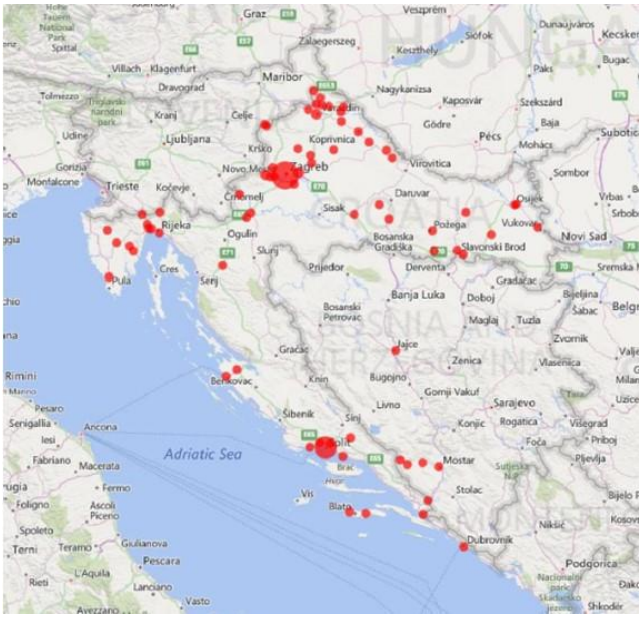


Slika 9. Analiza svijesti o Industriji 4.0 (Legg i dr., 2021)

Četvrtina ispitanika je izjavila da nemaju potrebe za uvođenjem robota iako se većina ispitanika muči sa pronalaskom kvalitetne radne snage. Na pitanje koje IT sustave koriste u proizvodnji približno trećina je izjavila da ih uopće ne koristi, a manje od polovice koristi *CAD* (engl. *Computer Aided Design*) programe. Korištenje *CAD-a* se može tumačiti kao loš signal budući da se vjerojatno odnosi na jednog ili dva čovjeka u tvornici koji su morali predstaviti ideju novog rasporeda strojeva što nije reprezentativna slika uzorka (Legg i dr., 2021). Neki prerađivači mogu imati obrambeni stav prema novoj tehnologiji temeljen na neznanju o dostupnim proizvodima na tržištu.

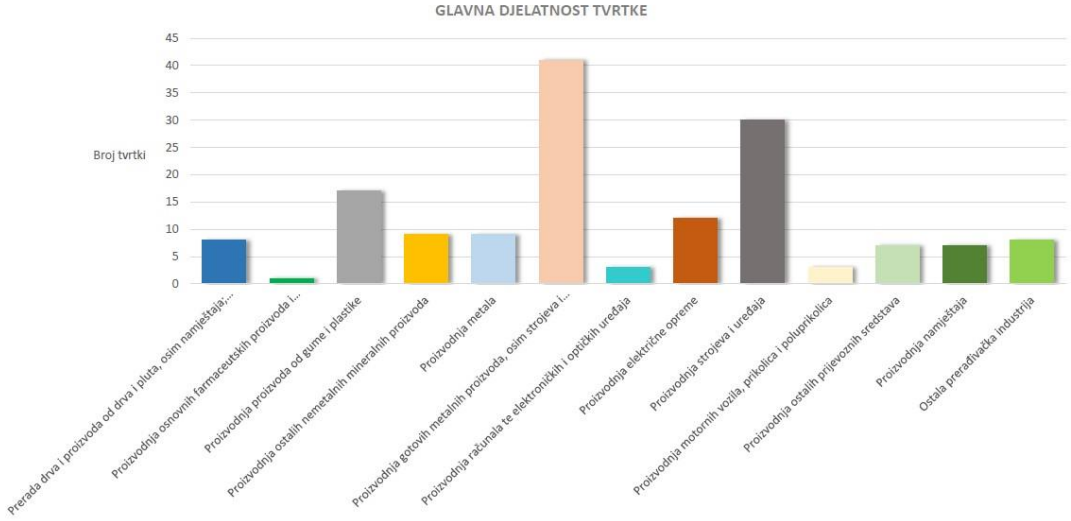
1.2.2. Industry 4.0 u Hrvatskoj i okolini

U Hrvatskoj se daje malo značaja dostupnoj novoj tehnologiji na tržištu. Razlog tomu je cijena investicije, potpuno nepoznavanje tehnologije, slabo poznavanje stranih jezika i generalni konzervativni stav prema novitetima. Iako je Industrija 4.0 relativno mlad pojam za domaću javnost, postoje studije o provedenim anketama koje prezentiraju spremnost i znanje hrvatskog poduzetništva. U jednoj anketi sudjelovalo je 160 tvrtki s područja Hrvatske i Bosne i Hercegovine (Peko, 2015). Obavljale su djelatnosti u rasponu od C16 – C32 prema NKDD.



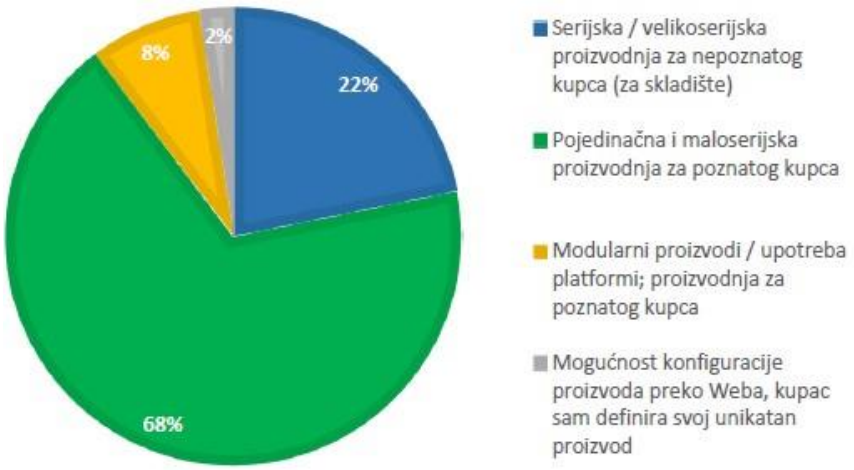
Slika 10. Geografski prikaz ispitanih subjekata (Peko, 2015)

Na slici 11 je prikazano apsolutno stanje broja anketiranih tvrtki prema njihovim djelatnostima. Najbrojniji u uzorku su bili iz područja proizvodnje proizvoda od gume i plastike, proizvodnje strojeva i uređaja i proizvodnje metalnih proizvoda.



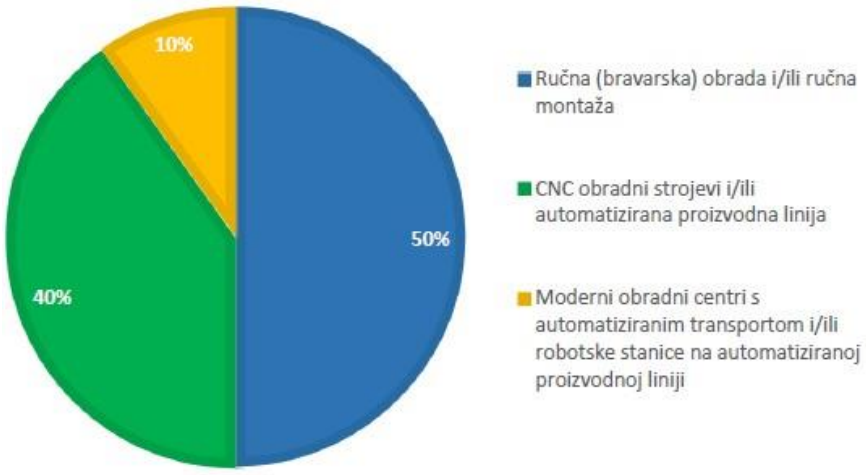
Slika 11. Dijagram djelatnosti ispitanih subjekata (Peko, 2015)

Anketa je uključivala samo prerađivačku industriju s obzirom da su one realni glavni stupovi gospodarstva. Neočekivano, 68 % tvrtki se bavi pojedinačnom i maloserijskom proizvodnjom dok je 22 % izjavilo kako radi velikoserijsku proizvodnju za nepoznatog kupca, tj. za skladište.



Slika 12. Dijagram modela proizvodnje ispitanih subjekata (Peko, 2015)

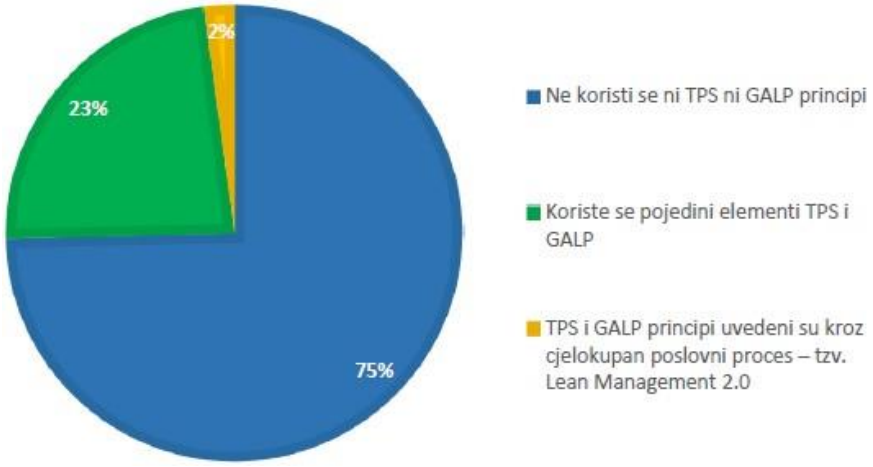
Na slici 12 je kružnim dijagramom prikazan udio različitosti načina rada ispitanika. Iz prikaza je vidljivo da modularna proizvodnja za poznatog kupca i individualna konfiguracija proizvoda putem interneta imaju tek desetinu ukupne količine. Nadalje, u razvoju proizvoda 84 % ispitanika koristi *CAD* programe, 9 % koristi virtualnu stvarnost (engl. *Virtual Reality*) i brzi razvoj prototipova (engl. *Rapid Prototyping*), a 7 % u svom razvoju koristi sustave „digitalne tvornice“ i simulacije pri razvoju. Svi ispitanici su se izjasnili da koriste računalnu opremu u bar jednom dijelu proizvodnog procesa. Na pitanje o dostupnoj tehnologiji u tvrtki čak polovica se definirala kao ručna (bravarska) obrada ili montaža što ukazuje da im je određena kompjuterizacija zastupljena izvan procesa mehaničke obrade materijala.



Slika 13. Dijagram dostupne tehnologije ispitanih subjekata (Peko, 2015)

Na slici 13 je kružnim dijagramom prikazana udjelna raznolikost korištenja dostupne tehnologije u proizvodnji ispitanika. Nekorištenjem unificiranih računalno stvorenih radnih naloga se često događa slučajni škart uzrokovan lošom interpretacijom zadanih uputa. Prema tome još se u 21 % posto slučajeva radni nalog objašnjava usmenom komunikacijom, a u 44 % voditelj predaje pisani radni nalog radniku. Jedna od lakših i jeftinijih inovacija u tvornicu je

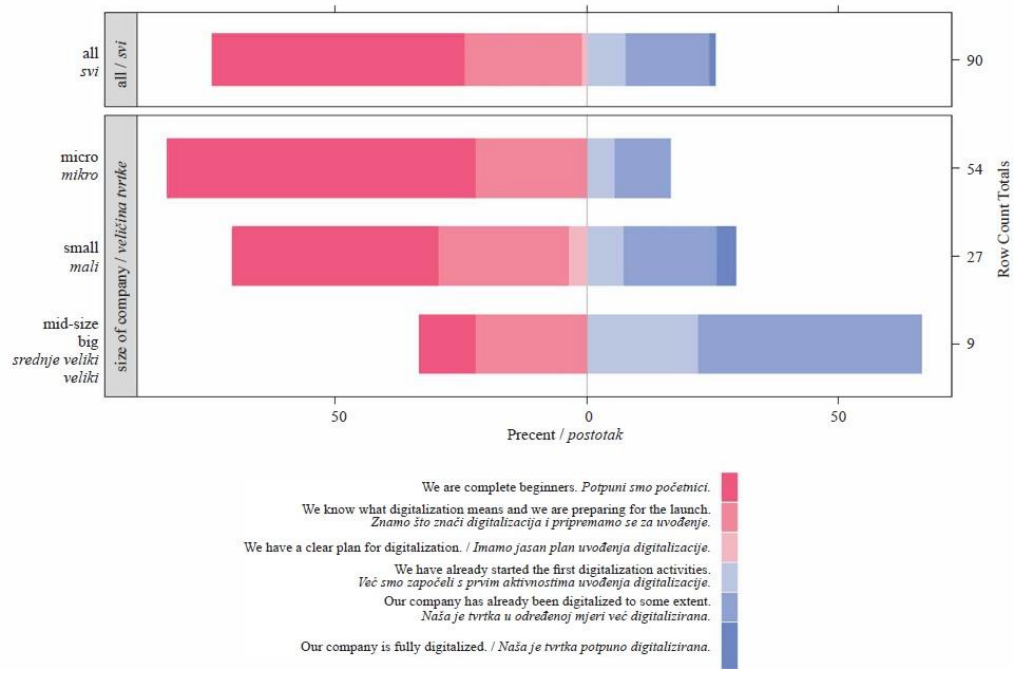
praćenje robe pomoću generiranog barkoda na pripadajućem poluproizvodu ili sirovini. Nakon takve inovacije i postepenog privikavanja sistemu upravljačko tijelo počne otvorenije gledati na automatizaciju. Nažalost, prema podacima ankete 87 % ispitanika ili ne vodi evidenciju ili roba ima zaljepljen papir na koji se upisuje tko je što napravio. Trećina tvornica ne zna koliko robe imaju u obradi i na skladištu pa se koriste metodama procjene koja je subjektivna i neprecizna, a uglavnom ovisi o jednoj osobi čija se odluka temelji na više godina iskustva.



Slika 14. Dijagram upoznatosti s Industrijom 4.0 (Peko, 2015)

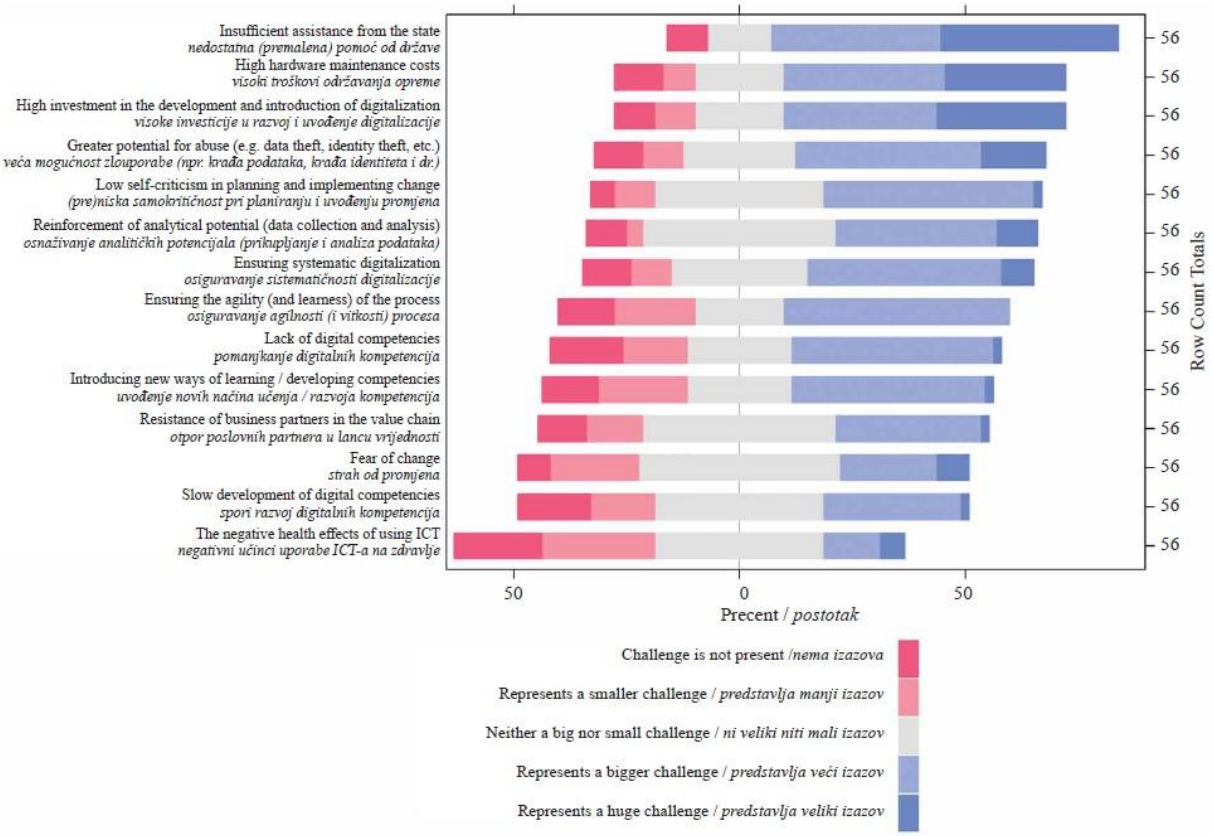
Na slici 14 je prikazan kružni dijagram o upoznatosti ispitanika s pojednim pojmovima vezanih za Industriju 4.0. Tri četvrtine ispitanika nije znalo ili ne koristi principe *Toyota Production System* ili *Green and Lean Production* (Peko, 2015). Dobrih primjera postoji, ali većina je onih koji ne barataju dobro značajkama koje su dovele treća i četvrta industrijska revolucija. Hrvatska prerađivačka industrija i dalje postoji te polagano raste što je pozitivno, a svoje vodeće mjesto na tržištu mora tražiti kroz inovacije na koje drugi još uvijek nisu spremni i tako izbiti na vrh. Rizik za to je velik, ali nagrade koje donosi su još veće. Prema analizi Bergera (2014), Hrvatska spada u skupinu „oklijevalista“ koje definira kao one koji vide prednosti inovacije, ali uvijek nađu neki razlog zašto to nije za njih trenutno. Potrebni kapital je moguće potražiti iz europskih fondova ili banki za obnovu i razvoj, ali je prije svega potrebno povećati svijest o prednostima Industrije 4.0. Jedan od načina je ugledati se na zemlje u bliskom okruženju koje imaju približno sličnu ekonomsku situaciju. Situacija u Sloveniji je generalno malo bolja nego u ostalim balkanskim zemljama što je već duži niz godina karakteristično. Ipak je pojam Industrije 4.0 dovoljno mlad da se i slovenske tvrtke ne mogu pohvaliti s velikim faktorom implementacije pripadajućih tehnologija u proizvodnju. Istraživanje stanja u slovenskoj drvnoj industriji, točnije sektorima djelatnosti C16 i C31 je provedeno prije nekoliko godina (Kropivšek i dr., 2019). Njihova anketa s dvadesetak pitanja je bila poslana više od 1000 prijavljenih poslovnih subjekata koji su članovi nacionalne gospodarske komore, a od kojih su dobili 131 nepotpun ili potpuni odgovor. Prema dobivenim rezultatima je moguće dobiti koeficijent korelacije između veličine subjekta i implementacije digitalnih tehnologija. Mikro i mali poslovni subjekti su uglavnom ograničeni za rast iz raznih razloga te nemaju jasnu viziju za budućnost tvrtke te stoga nema ni interesa za veća ulaganja u nove tehnologije. Takvi subjekti najčešće zapošljavaju i visok udio starijeg radnog stanovništva kojemu je digitalna

pismenost minimalna ili nepostojeća. Takvim okruženjem se stvara kontradikcija između predložene tehnologije i predviđenog operatera istom. Srednji i veći subjekti su uglavnom i prije bili primorani uvesti neke sustave olakšanog praćenja ili kontrole proizvodnje koji sada stvaraju veći liberalizam oko uvođenja tehnologija Industrije 4.0. Ovakvim ponašanjem će se sustavno na tržištu vršiti principi Darwinove evolucije, a industriju predvoditi nekoliko velikih imena u branši.



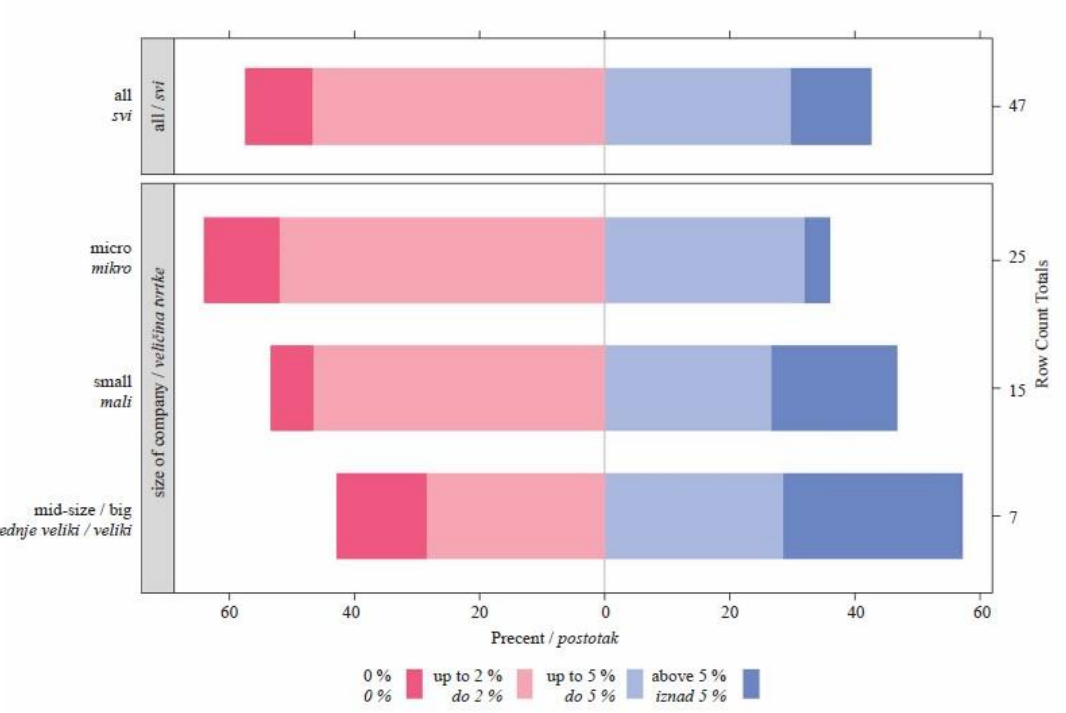
Slika 15. Odnos uvedene digitalizacije i veličine ispitanih subjekata (Kropivšek i dr., 2019)

Odnos uvedene digitalizacije i veličine poslovnog subjekta je vidljiv na slici 15. Njenim tumačenjem je vidljivo da razina implementacije novih tehnologija drastično raste s veličinom promatranog poduzeća, odnosno mikro, malih, srednjih i velikih tvrtki. Na pitanja o razlozima neinvestiranja navode slične kao i u drugim državama. Najčešće se to odnosi na visoke troškove, malu podršku od strane države, manjak kvalificiranih radnika kao i opasnosti koje nosi pretjerano izlaganje internetu, tj. moguće otkrivanje poslovne tajne. Grafički prikaz prezentiranih prepreka je vidljiv na slici 16.



Slika 16. Zastupljenost prepreka ispitanih subjekata (Kropivšek i dr., 2019)

Promatranjem danog grafa je vidljivo koje razloge u postotnom obliku poslovni subjekti smatraju kao ograničavajuće faktore. Poražavajuća je činjenica da je čak trećina ispitanika izjavila kako ne vjeruje kako će korištenje predloženih tehnologija trenutno ili u budućnosti biti presudne za poslovanje (Kropivšek i dr., 2019). Temelj takvog razmišljanja je da će se s drvom zbog svojih grešaka uvijek morati raditi s ljudskim okom i dodirom, bez obzira na razinu napretka tehnologije. Bez obzira na to, uvođenje nekih tehnologija ispitivanja kvalitete i praćenja proizvodnje su već potvrdile svoje benefite što potvrđuje sve veće ulaganje velikih tvrtki u digitalni sektor. Hipoteza da digitalizacija neće uspjeti savladati varijabilnosti masivnog drva će biti osporena onog trenutka kada cijena rada višestruko prijeđe cijenu sirovine. U takvoj situaciji će postojati samo zahtjev da se rad obavi uz minimalnu ljudsku intervenciju. Ispitanici su u anketi trebali i potvrditi planirano ulaganje u tehnologiju Industrije 4.0 u narednom trogodišnjem razdoblju. Velike tvrtke ulažu najviše, što ih trenutno čini prihvatljivijim krajnjem kupcu, rezultirajući eksponencijalnim rastom prihoda. Planirana ulaganja u odnosu na veličinu poduzeća vidljiva su na slici 17.



Slika 17. Odnos planiranih ulaganja i veličine poduzeća (Kropivšek i dr., 2019)

Odgovori na pitanje o planiranim ulaganjima u nove tehnologije pokazuju veoma sličan trend ponašanja kao i graf odnosa uvedene digitalizacije i veličine poduzeća. Oni koji su već implementirali neke tehnologije vide njene koristi i odlučuju se na daljnje investiranje u tom segmentu. Pozitivna je činjenica da više od trećine mikro poduzeća ipak vidi da će biti primorani na investiciju u neke od tehnologija Industrije 4.0 i sljedećih nekoliko godina.

2. CILJ RADA

Potaknuto slabom tranzicijom drvne industrije prema modernoj tehnologiji koje generira Industrija 4.0, potrebno je njenu problematiku i rješenja predstaviti populaciji (Legg, 2021). Prema Peko (2017) i dostupnim anketnim podacima u Hrvatskoj i okruženju nažalost još uvijek postoji tvrtke koje u svim segmentima poslovanja nisu prošle treću industrijsku revoluciju, tj. kompjuterizacije cijelog sistema. Onim načinom kako razvitak tržišta potiče drvnu industriju na implementaciju tehnologije Industrije 4.0, tako i proizvođači strojeva moraju odgovoriti na zahtjeve kupaca. Takve tvrtke moraju svoje kupce pridobiti sa predstavljanjem najboljih rješenja po pitanju kvalitete materijala, dostupnih kapaciteta i ekonomskog aspekta. Jedan od primjera koji pokazuju da se novi sistemi razvijaju je autonomna ćelija njemačkog proizvođača strojeva Homag. Njihovo rješenje se odnosilo na zajedničku operaciju oblaganja rubne trake i bušenja za pločaste materijale. Prvo predstavljanje je bilo na sajmu HOLZ-HANDWERK 2018. godine. Ćelijom upravlja jedan osposobljeni radnik koji umeće potrebni materijal koji je označen bar kodom koji sadrži sve potrebne operacije koje obradak mora proći. Na temelju skeniranog koda robot odlučuje da li obradak ide na daljnje kantiranje ili ga odlaže na *AGV*. *AGV* služi za odvoz materijala u međuskladište ili ako je vertikalni CNC stroj slobodan, odvodi ga na sljedeću operaciju. Ponovno pomoću skeniranja bar koda drugi robot odlučuje koje obratke ulaže u stroj, a koje odlaže na *AGV* kao završene. Prikazanom konfiguracijom na slici 18 se potpuno minimizirao broj potrebnih radnika, fizičkog napora i problem transporta materijala između radnih mjesta.

Slika 18. Homagovo rješenje autonomne ćelije (Izvor: https://www.homag.com/en/)

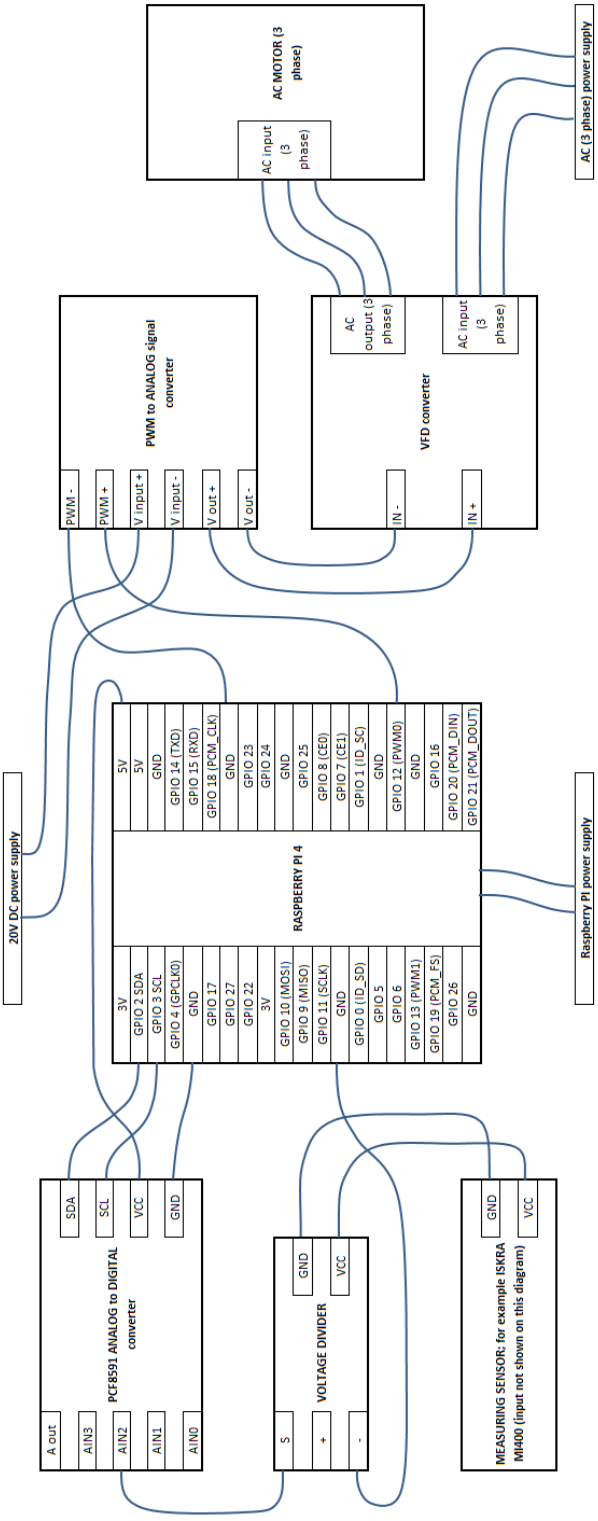
Homag tvrdi da je njihovo rješenje zadovoljavajuće za serije veličine jednog komada kao i masovne proizvodnje te da je prikladno za tvrtke od 10-25 radnika (HOMAG Group AG). Kako financijske mogućnosti svih tvrtki nisu jednake, potrebno je razmotriti i alternativna rješenja. Cilj ovog rada je približiti moguće načine jeftinijeg oblika automatizacije klasičnih strojeva

pomoću relativno jeftinih i jednostavno dostupnih komponenata. Razradit će se projekt preinake stolne glodalice kojom će biti omogućeno automatsko upravljanje glavnom brzinom rezanja i posmičnom brzinom. Obzirom da su pogonski elektromotori trofazno asinhroni, moguće ih je kontrolirati putem frekventnih regulatora. Upravljanje frekventnim regulatorima će biti omogućeno putem grafičkog korisničkog sučelja (engl. *Graphical User Interface-GUI*), a upravljanje frekventnim regulatorima će biti ostvareno preko naponskih upravljačkih signala. Usporedno tome, razvit će se sustav za mjerenje snage potrebne za glodanje pomoću mjernog pretvornika djelatne električne snage, kao jednog od bitnijih faktora u praćenju rada i opterećenja stolne glodalice za vrijeme obrade. U izračunima će se posvetiti pažnja na karakteristike mjernih instrumenata kako bi se minimizirale greške izmjerenih vrijednosti. Kao upravljačko računalo koristit će se mikroračunalo Raspberry Pi, s Rasbian, odnosno Debian GNU/Linux operativnim sustavom (standardni *multithreaded* operativni sustav bez *real-time patcha*), a *software* će biti napisan u programskom jeziku Python.

3. MATERIJALI I METODE

3.1. Shema spajanja strujnog kruga

Prije detaljnog opisa pojedinih komponenti razvijenog prototipa upravljačkog i mjernog sustava, pojasnit će se cjelovita blok-shema (slika 19) razvijenog prototipa sustava.



Slika 19. Blok-shema strujnog kruga

Kao upravljačko mikroračunalo korišten je Raspberry Pi 4 model B s vlastitim napajanjem. Za upravljanje naponskim ulazima frekventnih pretvarača korišteni su digitalno-analogni (D/A) pretvornici koji su upravljani pulsno-širinski moduliranim (engl. *Pulse Width Modulation* – PWM) signalima s odgovarajućih izlaza upravljačkog mikroračunala. Iz prikazane sheme (slika 19) može se vidjeti da su spojeni prema sljedećem: PWM- ulaz s GND na mikroračunalu, a PWM+ ulaz s PWM GPIO pinom 32 (koristi se za upravljanje frekventnim pretvaračem za kontrolu posmične brzine) i PWM GPIO pinom 33 (koristi se za upravljanje frekventnim pretvaračem za kontrolu frekvencije vrtnje radnog vretena, odnosno brzine rezanja). Napajanje pretvornika je na ulazima Vinput- i Vinput+ s vanjskim izvorom napajanja u iznosu od U_{cc} = 20 V. Izlazni naponski analogni upravljački signal u rasponu (0-10) V se s izlaza Voutput- i Voutput+ spaja na ulaze IN- i IN+ na frekventnom pretvaraču, koji je podešen za upravljanje izlaznom frekvencijom vanjskim naponskim upravljačkim signalom u rasponu (0-10) V. Frekventni pretvarač koristi vlastito trofazno napajanje te je kontrolna grana spojena na električnu kutiju elektromotora. Mjerni dio sustava, sastoji se od analogno-digitalnog (A/D) pretvornika koji je predviđen za mjerenje naponskog analognog signala s odgovarajućeg izlaza mjernog pretvornika snage. Mjerni pretvornik snage ima izlazni naponski analogni signal u rasponu (0-10) V, proporcionalan mjerenoj djelatnoj električnoj snazi. S obzirom da je ulazni raspon A/D pretvornika u rasponu od (0-5) V, bilo je potrebno prilagoditi izlazni naponski signal s mjernog pretvornika snage. Za to je upotrebljeno programibilno otporničko dijelilo napona. Naponski izlaz VCC s pretvornika je spojen s VCC ulazom na djelilu napona, a uzemljenje GND s GND ulazom na djelilu napona. Djelilo napona s faktorom djeljenja 5 je potrebno za smanjenje naponske vrijednosti jer mikroračunalo može primiti signal najvećeg napona 3,3 V. Upotrebljen je 8-bitni A/D pretvornik oznake PCF8591. Analogni ulaz pretvornika AIN2 spojen je s naponskim izlazom S djelila napona. S obzirom da odabrani A/D pretvornik koristi digitalni I2C komunikacijski protokol moguće ga je direktno spojiti na odgovarajuće GPIO pinove upravljačkog mikroračunala, te su izlazni portovi SDA, SCL, 5 V i GND spojeni s istoimenim pinovima na mikroračunalu. SDA i SCL portovi služe za komunikaciju pretvornika i mikroračunala.

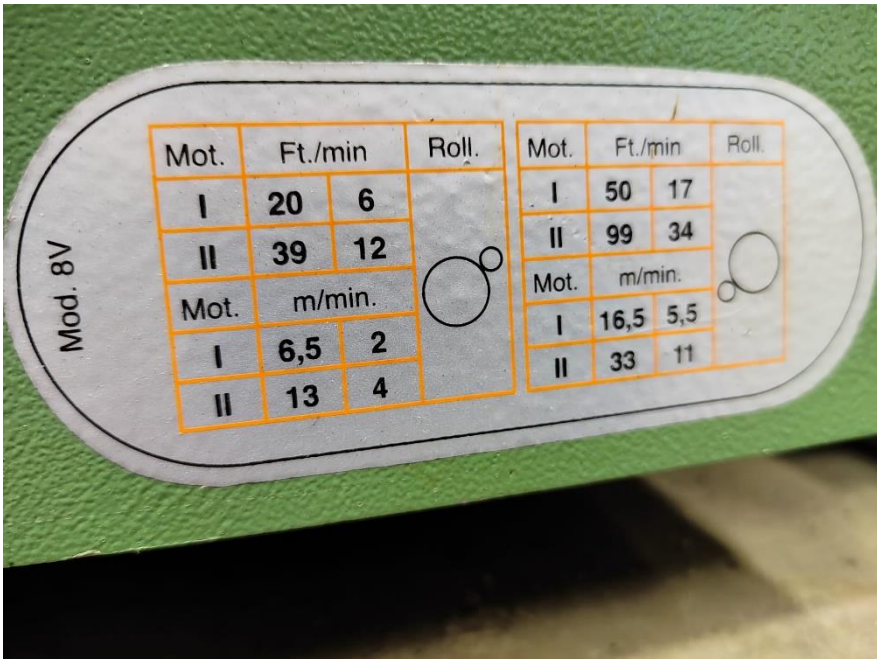
3.2. Stolna glodalica

Eksperimentalni dio rada proveden je na stolnoj glodalici u Laboratoriju za mehaničku obradu drva pri Sveučilištu u Zagrebu, Fakultetu šumarstva i drvne tehnologije. Sama vrsta stroja ne čini ulogu u ovome radu jer je kao primjer mogla biti uzeta i tračna pila, debljača, ravnalica ili stolna kružna pila. Osnovni uvjet izbora je bio da se radi o standardnom stroju bez numeričke ili kompjutersko-numeričke kontrole s trofaznim asinhronim elektromotorima bez upuštanja u rad pomoću frekventnih pretvarača. Glodalica je češkog proizvođača strojeva za preradu drva Rojek, modela FSN 300A. Opremeljena je glavnim trofaznim asinhronim elektromotorom snage 3,7 kW spojenog na mrežu 3x400 V u trokut, frekvencije 50 Hz. Glavni motor putem stepenaste remenice na osovini i remenskog klinastog prijenosa pokreće vratilo

u 4 brzine; 1400, 3500, 6000, 8000 o/min. Prikaz sustava za prijenos snage i kretanja s glavnog motora na pogonsko radno vreteno prikazan je na slici 20.

Slika 20. Remenski prijenos gibanja glavnog motora

Promatrani stroj kao dodatnu opremu ima i nadstolni agregat koji služi za mehanizirani posmak obratka. Agregatu pripada dvobrzinski elektromotor snage 0,75 kW i mjenjačke kutije kojom se omogućuje stupnjeviti izbor posmaka obratka u osam posmičnih brzina kako je vidljivo na Slici 21.



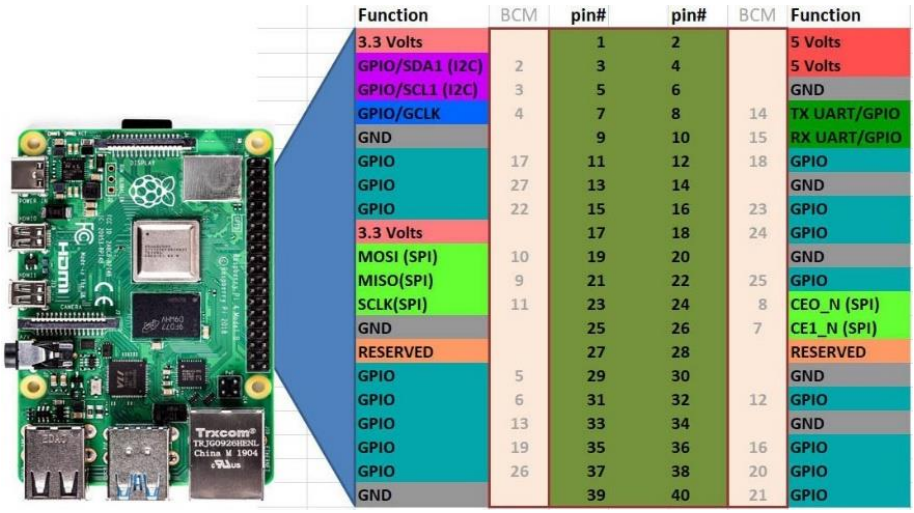
Slika 21. Dostupne brzine posmičnog gibanja

Posmične brzine se mijenjaju zamjenom zupčanika u kućištu agregata te grebenastim prekidačem s dvije brzine na elektromotoru i samom kućištu prijenosa.

3.3. Mikroračunalo Raspberry Pi

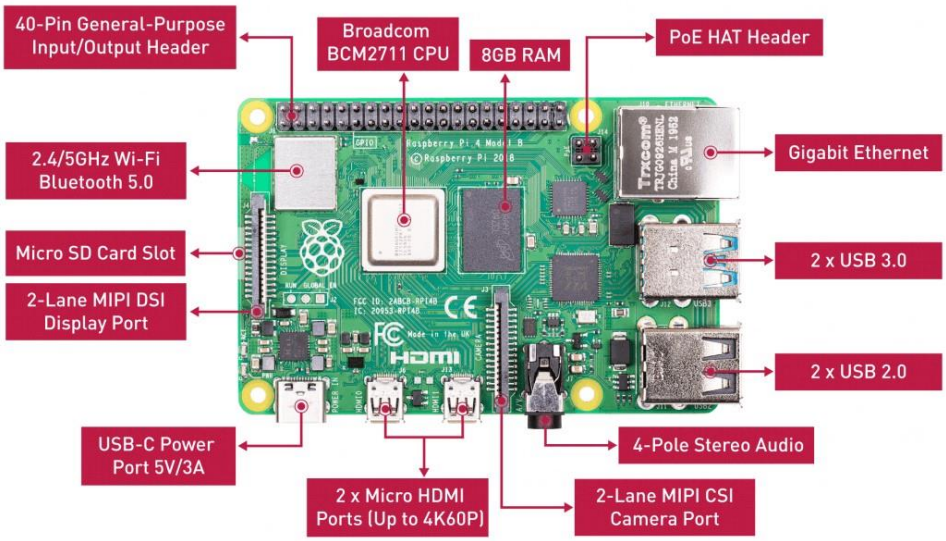
Kao upravljačko mikroračunalo je odabran Raspberry Pi koji slovi kao najprodavanije i najpopularnije i lako dostupno mikroračunalo relativno niske cijene. Analizom uvjeta koji je

trebalo zadovoljiti, Raspberry Pi je odabran naspram ostalih mikroračunala i mikrokontrolera poput Arduino Uno, Rasberry Pi Pico ili Banana Pi iz više razloga. Ovakva konfiguracija računala je imala značaja zbog dostupnosti tiskanih i online materijala za učenje. Python se smatra objektno orijentiranim programskim jezikom te je prikladan za one bez prijašnjeg znanja u pisanju koda. Pored njega, Raspberry Pi podržava programiranje u C++ i Scratch jezike. Teoretski, za projekt automatizacije analognog stroja je dovoljan mikrokontroler poput Rasberrya Pi Pico ili Arduino Una, ali u tim slučajevima ne bi bio moguć razvitak grafičkog korisničkog sučelja ili bi bili potrebni postprocesori koji bi uklađivali kod programa iz Pythona u kontroleru prikladan operativni programski jezik. Korišten model u ovome radu će biti Raspberry Pi četvrte generacije s 8 GB radne memorije te mogućnosti WiFi ili Ethernet konekcije na internetsku mrežu. Od ostalih karakteristika valja napomenuti da posjeduje 2.0 i 3.0 USB ulaze za periferne uređaje, sklop s 40 GPIO (engl. *General Purpose Input/Output*) iglica, MIPI portove za kameru i zaslon. Od 40 dostupnih iglica, nemaju sve istu namjenu. Računalo putem njih može dati signal u obliku 0 ili 1, točnije propušta struju napona 3,3 ili 0 V. U spajanju hardverskih komponenti i programiranju koda treba voditi obzira na način označavanja ovih 40 iglica. Postojeća dva načina su BOARD i BCM sustav. BOARD se odnosi na fizičku i kronološku numeraciju od ruba prema sredini pločice, a BCM na Broadocomov SoC sustav prepoznavanja iglica. Oba sustava su prikazana na slici 22.



Slika 22. Tipovi numeracije GPIO sklopa(Izvor: https://toptechboy.com/understanding-raspberry-pi-4-gpio-pinouts/)

U programiranju koda za ovaj rad je korištena BOARD numeracija prema osobnoj preferenci. Također, donekle je lakša orijentacija koristeći kronološki poredak prilikom fizičkog spajanja pinova. Od prikazanih, u radu će se koristiti SDA, SCL, GND i PWM portovi. Detaljnija arhitektura ovog mikroračunala je vidljiva na slici 23.

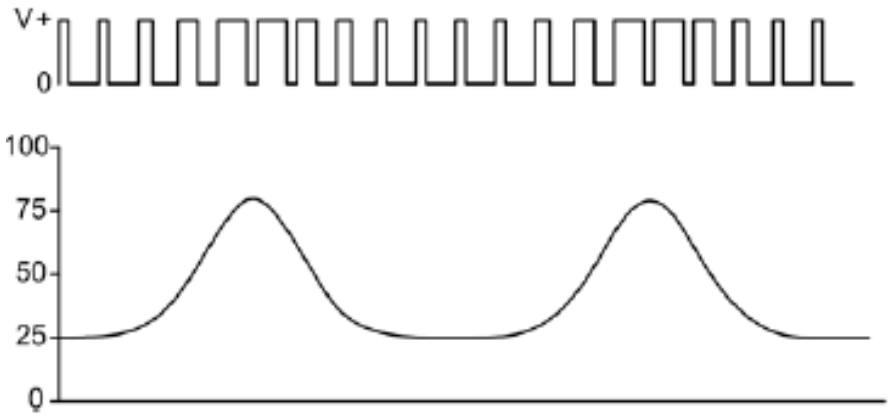


Slika 23. Shema Raspberry Pi-ja s označenim komponentama(Izvor: https://www.amazon.com/Raspberry-Pi-Computer-Suitable-Workstation/dp/B0899VXM8F)

3.4. Komponente upravljačkog sustava

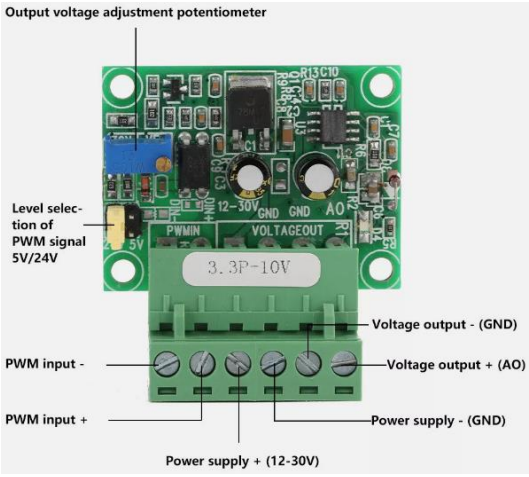
Komponente upravljačkog sustava se sastoje od frekventnog pretvarača, djelila napona, A/D konvertera i D/A konvertera upravljanog PWM signalom. Prvenstveno, za upravljanje broja okretaja vratila elektromotora je potrebno generiranje frekvencije pomoću frekventnog pretvarača. Jedna od funkcija frekventnog pretvarača je kontrola frekvencije vrtnje upravljanog asinhronog elektromotora regulacijom frekvencije i napona. U radu je korišten jedan pretvarač upravljan putem jednog D/A pretvornika čija se konekcija na Raspberry Pi zamijenjivala s pina 32 na 33 i obrnuto, ovisno o tome koji se elektromotor upravljao. Korišteni frekventni pretvarač je marke Mitsubishi, modela FREQROL-CS80. Pojednostavljenja radi, bit će predstavljene samo najbitnije karakteristike za ovaj rad. Pretvarač može upravljati motorima do 3,7 kW snage i 8 A maksimalne jakosti struje za trofazne motore u mreži 380-480 V. Napajanje je trofazno (380-480 V, 50/60 Hz), a hlađenje prirodnom cirkulacijom zraka. Moguća izlazna frekvencija je u rasponu od 0,2-400 Hz, analogno upravljanje moguće u rasponima napona 0-5, 0-10 V te rasponu struje 4-20 mA. Korišteni raspon je bio od 0 V do 10 V. Točnost izlazne frekvencije se nalazi u ±1 % maksimalne izlazne frekvencije. U promatranom slučaju i rasponu 0-50 Hz, točnost je u apsolutnom iznosu bila ±0,5 Hz. Odabrani frekventni pretvarač je vidljiv na SLICI strujnog kruga. Nadalje, bilo je potrebno adekvatno prilagoditi signal birane frekvencije s upravljačkog računala prema frekventnom pretvaraču. Kako Rasberry Pi generira digitalne, a frekventni prima analogni signal potrebno je napraviti konverziju. Mikroračunalo može dati signal maksimalnog napona 3,3 V, a frekventnom pretvaraču je potreban napon u rasponu od 0 V do 10 V što znači da treba povećati napon. Napon signala se poveća korištenjem pretvornika signala s vanjskim napajanjem. Promjenu naponske vrijednosti unutar intervala omogućuje korištenje PWM pinova na mikroračunalu. PWM je pulsno-širinski moduliran signal korišten za varijabilno kontroliranje okretaja motora (Huges, 2010). Ovisi o definiranim parametrima frekvencije i

radnog ciklusa (engl. *duty cycle*). Radni ciklus se odnosi na omjer vremena aktivnog signala i vremena trajanja perioda i označava se postotno. Na slici 24 je prikazana ovisnost promjene radnog ciklusa na izlazni naponski signal.



Slika 24. Prikaz ovisnosti radnog ciklusa o širini vala generiranog signala (Huges, 2010)

Kako graf pokazuje, naponske vrijednosti se ne mijenjaju, već ostaju na 0 ili 3,3 V. Primjera radi, postavka radnog ciklusa na 50 % znači da je napon jednak 3,3 V polovicu vremena jednog perioda. Pretpostavka da je naponski signal jednak 1,65 V je u ovom slučaju pogrešna. Promjena parametra radnog ciklusa na 50 % će u projektu rezultirati s frekvencijom pretvarača od 25 Hz. Kako bi predstavljeni uvjeti bili ispunjeni, korišten je D/A pretvornik vidljiv na slici 25.



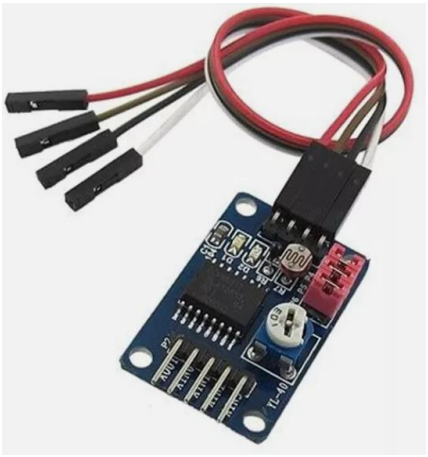
Slika 25. Prikaz korištenog D/A konvertera (Izvor: https://www.ebay.com/itm/196048675241?mkcid=16&mkevt=1&mkrid=711-127632-2357-0&ssspo=MQE01S0JRTi&sssrc=4429486&ssuid=hojns4mBQ7q&var=&widget_ver=artemis&media=MORE)

Prikazani konverter ima izlazni signal istosmjernog napona u rasponu 0-10 V koji je omogućen vanjskim istosmjernim napajanjem u rasponu 12-30 V. U radu je na potenciometru izvora napajanje bilo podešeno na 20 V. Kalibracija konvertera se vršila na potenciometru konvertera s ciljem izjednačenja greške izlaznog napona pri krajevima radnog intervala. U idealnom slučaju bez gubitaka i grešaka komponenti postavka frekvencije vrtnje na 25 Hz treba rezultirati izlaznim naponom od 5 V. U specifikaciji konvertera je navedena preporučena frekvencija PWM signala od 1 kHz do 3 kHz te je rađeno na postavki od 500 Hz. Odluka o radu na frekvenciji niže nego preporučene je donešena eksperimentalno prilikom kalibracije

sustava. Pri preporučenim frekvencijama rada je dolazilo do većih oscilacija naponskih signala i stabilnijom frekvencijom na frekventnom pretvaraču. Pri 500 Hz je naponski signal bio u području prihvatljivijih oscilacija te neznatno većim odstupanjem frekvencije na frekventnom pretvaraču. Navedena rezolucija pretvornika je 0,1 V. Sljedeće komponente su potrebne za konvertiranje povratnog signal s mjernog uređaja. Navedeni signal je analogni istosmjerni napon u rasponu od 0 V do 10 V. Kako bi mikroračunalo očitalo podatke s GPIO pina, signal je potrebno konvertirati u digitalni te mu smanjiti maksimalni napon na 3,3 V jer u protivnom može doći do preopterećenja sklopa mikroračunala. U svrhu smanjenja napona je na povratnoj vezi ugrađeno djelilo napona vidljivo na slici 26.

Slika 26. Prikaz korištenog djelila napona (Izvor: https://www.ebay.com/itm/173860251282?mkcid=16&mkevt=1&mkrid=711-127632-2357-0&ssspo=4o--qSjnR6m&sssrc=4429486&ssuid=hojns4mBQ7q&var=&widget_ver=artemis&media=MORE)

Djelilo napona koristi određeni broj otpornika kako bi smanjio napon u strujnom krugu. Korišteno otporničko djelilo može se koristiti do najviše 25 V s faktorom djeljenja 5. Svaka ulazna vrijednost se dijeli s faktorom 5 te se time napon smanjuje na petinu ulazne vrijednosti. U promatranom primjeru, napon od 10 V se smanjuje na napon od 2 V. Ova karakteristika je bitna za upis u izvorni kod programa kako bi izračun mjerene vrijednosti bio ispravan. Rezolucija djelila je 0,00489 V. Nije dostupna specifikacija djelila napona o točnosti bez koje se ne može računati granična pogreška istog. Istosmjerni signal smanjenog napona se na A/D pretvorniku prilagođuje za očitanje na Raspberry Pi-ju. Korišteni 8-bitni A/D pretvornik je vidljiv na slici 27.



Slika 27. Prikaz korištenog 8-bitnog A/D pretvornika (Izvor: https://www.ebay.com/itm/355019371747?mkcid=16&mkevt=1&mkrid=711-127632-2357-0&ssspo=OJ2q66xfTOi&sssrc=4429486&ssuid=hojns4mBQ7q&var=&widget_ver=artemis&media=MORE)

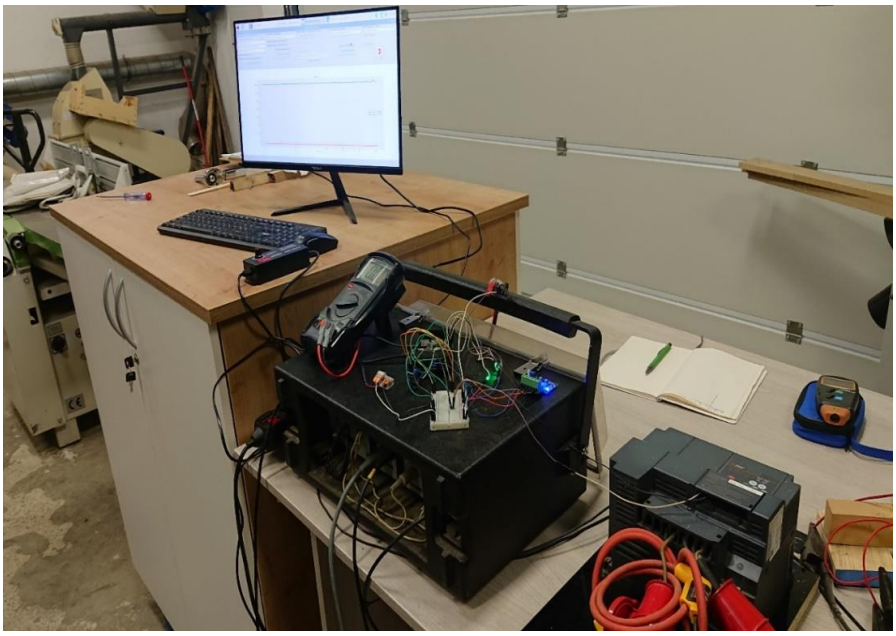
Prikazani konverter ima 4 ulazna pina od kojih se u radu koristi samo jedan te sabirni izlaz prema mikroračunalu. Najveći ulazni naponski signal pretvornika je 5 V. Rezolucija pretvornika nije definirana u specifikaciji proizvoda te se ona računa i iznosi prema sljedećem:

$$r = \frac{R}{2^n} = \frac{5}{2^8} = 19{,}53\ mV.$$

Naponski raspon *R* je jednak referentnom naponu od 5 V. Komunikacija između računala i konvertera se odvija putem I2C protokola. Ovaj protokol se koristi za povezivanje s senzorskim uređajima, a moguć je samo na SDA (engl. *Serial Data Line*) i SCL (engl. *Serial Clock Line*) pinova na Raspberry Pi. Pretvorniku je potrebno napajanje u rasponu od 2,5 V do 6 V pa je spojen i na pinove 5 V i GND (uzemljenje). Sam princip izračuna izmjerene vrijednosti prema konvertiranom signalu će biti objašnjen prilikom opisa funkcije analog read u izvornom kodu programa.

3.5. Metode mjerenja

Ovo poglavlje će opisati metode mjerenja napona i mjerenja frekvencije vrtnje. Prema vrijednostima električne snage će se prikazati i analizirati približni pogonski dijagram asinhronog motora glavnog gibanja u zasebnom potpoglavlju. Za mjerenje napona ulaznog signala u frekventni pretvarač, odnosno naponskog signala na izlazu PWM D/A pretvornika korišten je multimetar UNI-T UT51. Multimetar koji se koristi kao voltmetar je spojen paralelno. U provednom radu je multimetar stezaljkama bio spojen na eksperimentalnu pločicu mikroračunala. Poštujući smjerove povezivanja eksperimentalne pločice potrebno je bilo da se PWM signal, stezaljke multimetra i žična veza prema frekventnom regulatoru nalaze u istom nizu. Schematski prikaz spajanja je objašnjen u poglavlju Schema strujnog kruga. Široki prikaz korištenog i funkcionalnog spojenog strujnog kruga u laboratoriju je vidljiv na slici 28.



Slika 28. Spojeni strujni krug za upravljanje glodalice

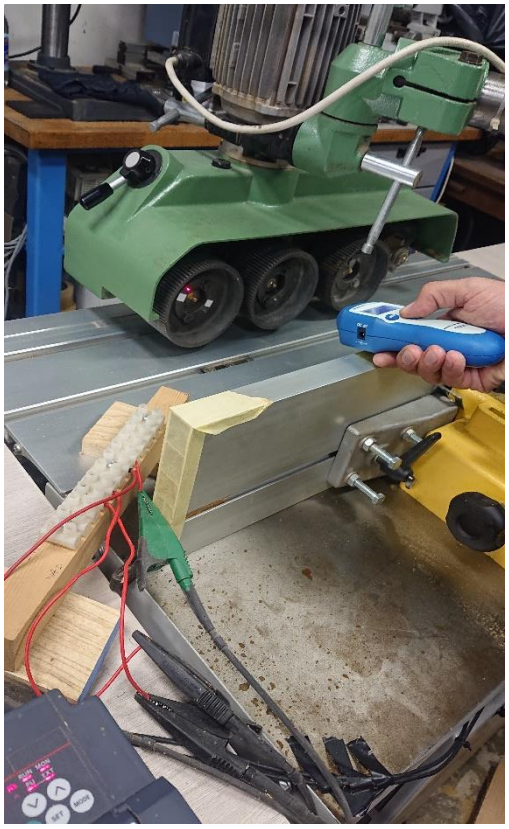
Dani prikaz još uključuje komponente monitora, bežične tipkovnice i miša te izvora napajanja s mogućnošću regulacije izlaznog napona u rasponu od 3 V do 24 V. Ove komponente su potrebne za korištenje mikroračunala te kao izvor napajanja pretvornika signala. Uz to, univerzalne su te ne utječu na rezultate rada i stoga se neće detaljnije opisivati. Rezultantne vrijednosti naponskih signala i zadane frekvencije bit će analizirane u odnosu s mjerenim podacima tahometra čije su karakteristike i granična pogreška objašnjene u narednim poglavljima. Za točno određivanje frekvencije vrtnje valjaka na agregatu za posmično gibanje i na osovini glodala je korišten tahometar. Uređaj koristi reflektirajuću naljepnicu kao referentne točke na objektu vrtnje u koju se odbijaju uperene svjetlosne zrake samog uređaja. Mjerena vrijednost se generira svakih pola sekunde, a objekt mjerenja ne smije biti udaljen više od 500 mm.

3.6. Mjerni instrumenti

U predstavljenom projektu će se pripremiti konfiguracija komponenti za mjerenje i simulirati mjerenje djelatne električne snage glavnog, frekvencije vrtnje alata te promjene naponskih vrijednosti pri ulaznim i izlaznim portovima novostvorenog upravljačkog hardvera. Stoga je potrebno prikazati karakteristike korištenih mjernih instrumenata te detaljnije objasniti njihovu svrhu. Ovakvim pristupom se odabirom hardverskih komponenti i uvrštavanja statičkih osjetljivosti odabrane mjerne tehnike u izvorni kod programa stvorila zamjenska mjerna tehnika koja bi mogla služiti u slučajevima neispunjenih uvjeta za mjerenje u industrijskim pogonima. Predviđeni mjerni instrumenti su programibilni mjerni pretvornik snage oznake Iskra MI400, tahometar, multimetar oznake UNI-T, modela UT51. Mjerni pretvornik Iskra MI400 (slika 29) ima raspon mjerenja do 600 V i 12,5 A uz raspone napajanja od 24-300 V (istosmjerni izvor) i 40-276 V (izmjenični izvor). Klasa točnosti instrumenta je 0,5 prema normi EN 60 688. Podržava do 4 izlazno-ulazna modula: analogni, impulsni, alarmni i digitalni. Od ponuđenih u radu je predviđeno snimanje vrijednosti analognog izlaza u rasponu od 0 V do 10 V.

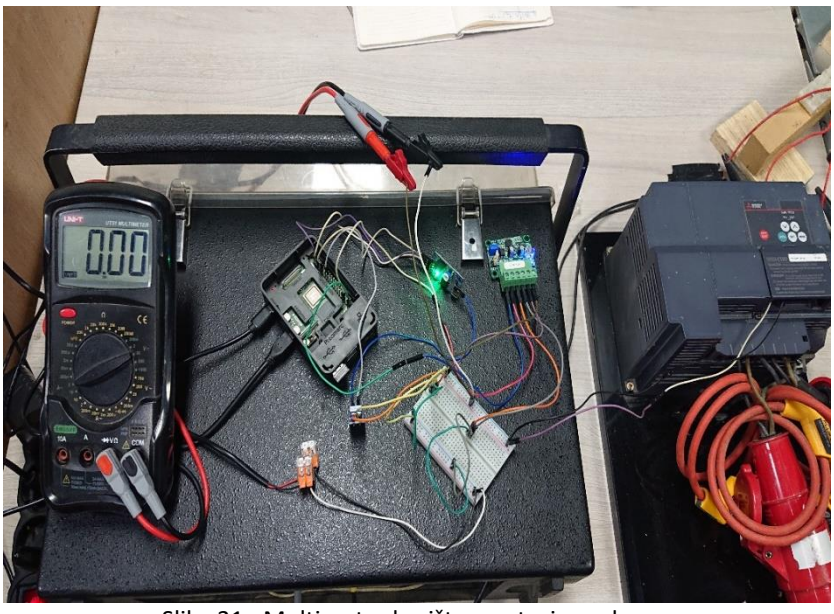
Slika 29. Mjerni pretvornik snage Iskra

Korišteni digitalni tahometar u procesu mjerenja frekvencije vrtnje je vidljiv na slici 30. Nazivno područje pokazivanja tahometra je od 2 min^{-1} do 99999 min^{-1} uz točnost od ±(0,5 % očitane vrijednosti + 1 digit).



Slika 30. Postupak mjerenja frekvencije vrtnje tahometrom

U navedenom primjeru su korištene dvije reflektirajuće naljepnice zbog kojih treba izmjerene rezultate dijeliti s faktorom 2. Naponske vrijednosti vrijednosti izlaznog signala generiranog od strane mikroračunala kao i provjera ispravnosti povratne veze očitanja vrijednosti s mjernog pretvornika snage su mjerene multimetrom oznake UNI-T UT51 (slika 31).

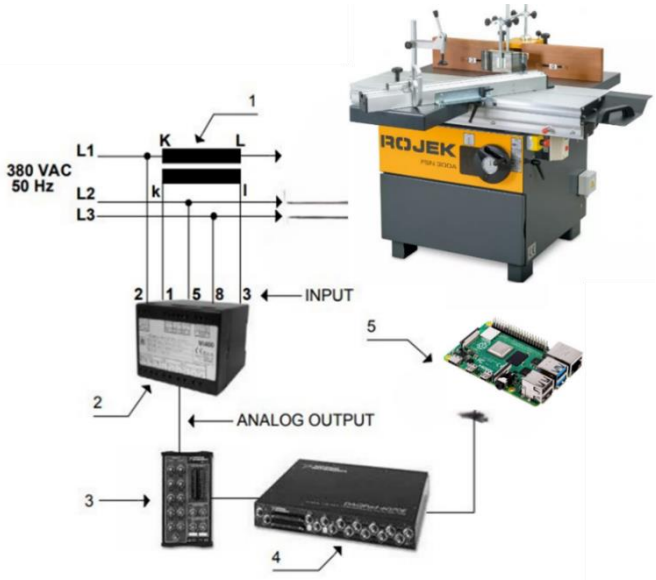


Slika 31. Multimetar korišten u strujnom krugu

U mjerenju istosmjernog napona je korišten raspon od 0 V do 20 V za kojega prema specifikaciji vrijedi rezolucija od 10 mV te točnost od ±(0,5 % očitane vrijednosti + 1 digit). Valja napomenuti da se mjerni instrumenti nisu rekalibrirali obzirom da to nisu dopuštali uvjeti projekta već da se pouzdanost temeljila na tvornički kalibriranim vrijednostima. Kalibriranje mjernog pretvornika snage je složen postupak kojeg ovaj rad nije obuhvatio, ali za pretpostaviti je da izmjerene vrijednosti ne osciliraju u tolikoj mjeri da dobiveni podaci i zaključci se ne mogu primijeniti u drvnoj industriji.

3.7. Mjerenje djelatne električne snage na glavnom pogonskom elektromotoru stolne glodalice

Često se zbog jednostavnosti mjerenja, cijene i drugih utjecajnih faktora se sila rezanja, jedinični otpori rezanja i ostale njima zavisne veličine određuju preko veličine prosječne potrebne snage za rezanje, odnosno na temelju izmjerene prosječno potrebne djelatne električne snage za vrijeme obrade. Ova metoda je manje točnija, ali i dalje najpraktičnija i ekonomski prihvatljiva od mjerenja sila rezanja pomoću piezoelektričnih dinamometara ili dinamometara s tenzometarskim trakama. Određivanje mehaničke snage na osovini motora je važno kako bi se utvrdilo je li elektromotor ispravno izabran u skladu s zahtjevima koji se pred njega stavljaju. Najlakši pristup izračunu mehaničke snage je putem izrade približnog pogonskog dijagrama na temelju kojega se može izračunati mehanička snaga trofaznog asinhronog elektromotora te čija maksimalna greška ne odstupa više o 2 % od stvarne krivulje elektromotora. Za dobivanje nekih potrebnih varijabli je predviđeno mjerenje mjernim pretvornikom snage Iskra MI400 čije su karakteristike opisane u potpoglavlju Mjerni instrumenti. Opis načina rada i vrste mjernih pretvornika snage detaljno su opisana u odgovarajućoj literaturi. Prema Bego (1979), ovi uređaji upotrebljavaju zbog mogućnosti daljinskog mjerenja i upravljanja, kao i računske obrade podataka. Sve moguće mjerene veličine ovog uređaja on pretvara u proporcionalan istosmjerni napon. Moguće je pojednostavljeno prikazati strujni krug potreban za mjerenje djelatne električne snage na elektromotorima glodalice. Predloženi strujni krug za mjerenje je vidljiv na Slici 32.



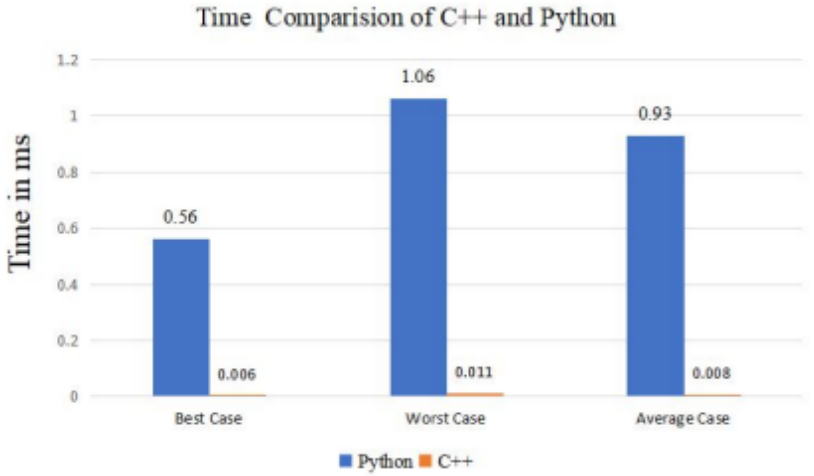
Slika 32. Shema predloženog sustava za mjerenje djelatne električne snage na glanom pogonskom elektromotoru stolne glodalice (1 - strujni transformator, 2 - mjerni pretvornik Iskra MI400, 3 - BNC pretvornik, 4 - DAQ sustav NI Daq Pad 6070 E, 5 - Raspberry Pi 4 model B)

U provedenom eksperimentu mjerni pretvornik snage nije fizički bio spojen na elektromotor već za potrebe ispitivanja upravljačkog programa uzeta je pretpostavka prijašnjih mjerenja o podešenoj statičkoj osjetljivosti (k) u iznosu 346,6 W/V, a mjerni signal je simuliran naponskim signalom u rasponu od 0 V do 10 V. Upisom osjetljivosti u kod programa se odvija mjerenje i grafički prikaz na monitoru principom umnoška izlazne vrijednosti napona i osjetljivosti mjernog pretvornika snage. Poznavanjem svih potrebnih karakteristika uređaja stvoreni strujni krug će služiti za simulirano mjerenje djelatne električne snage prema čijim podacima se može izraditi približni pogonski dijagram za elektromotor u zasebnom poglavlju. Opisani pristup može u nekim hitnim industrijskim slučajevima služiti kao zamjena za nedostupnu mjernu opremu. U tom slučaju je nužno da korisnik temeljito poznaje karakteristike mjerne opreme. Prikazanim sustavom se dobiva podatak o prosječno potrebnoj snazi na ulazu u elektromotor na temelju kojih je potrebno izračunati snagu rezanja u postavljenim uvjetima. Opis potrebnih podataka i procesa izrade je obradio Hamm (1970). Prema njemu, za konstrukciju dijagrama je potrebno nekoliko podataka: nazivna mehanička snaga motora P, nazivnu električnu snagu koju troši elektromotor pri nazivnom opterećenju P_{el}, gubitak električne snage pri nazivnom opterećenju P_g i gubitak elekrične snage pri neopterećenom kretu P_0. Mehanička snaga motora P se očitava s pločice motora. Gubitak električne snage pri nazivnom opterećenju se određuje iz razlike nazivne električne snage i nazivne mehaničke snage. Ukoliko gubitak električne snage pri neopterećenom kretu nije moguće izmjeriti se može ugrubo uzeti jednak polovici iznosa gubitka pri nazivnom opterećenju. Nakon utvrđenja potrebnih vrijednosti, konstruira se koordinatni sustav s dvije ordinatne osi. Osi trebaju biti međusobno okomite te u ispravnom mjerilu (npr. 1 cm = 1 kW). P_{el} i P_g se nanose na okomitim osima desne strane sustava, a P_o na lijevoj strani. Potom se povlače pravci s ishodištem u P_o prema P_{el} i P_g. Pravac P_o-P_{el} je dovoljan sa većinu praktične primjene te prema jednadžbi pravca koji prolaze kroz dvije točke se mogu izračunati vrijednosti mehaničke snage i utrošak

električne snage (Hamm, 1970). Iako linija između danih vrijednosti u stvarnosti nije linearna već plosnati luk, greške pojednostavljenog izračuna ne prelaze 2,4 %. Pogreška ove veličine u praktičnim primjerima je gotovo zanemariva te se ne uzima u obzir. Za određivanje mehaničke snage na osovini motora postoje i druge metode koje nisu u ovom radu obrađene iz razloga nedostatka mjerne opreme ili nepoznatih podataka elektromotora.

3.8. Općenito o programskom jeziku Python

Python je objektno-orijentiran jezik stvoren 90-ih godina u Nizozemskoj i prikladan je za sve stadije programa poput analize, dizajn, stvaranja prototipa, kodiranja, testiranja, otklanjanja grešaka, usavršavanja, dokumentacije, izdavanja i održavanja što ga čini poželjnim za programere. Objektno-orijentiran jezik ne stvara strukturu podataka i metoda odvojeno već stvara statičke entitete u programu koji sadrže određene podatke i definirane metode kako obraditi te podatke (Buršić, 2017). Radi se o jeziku vrlo visoke razine (VHLL), tj. koristi višu razinu apstrakcije nego što je koriste klasični kompajlirani jezici poput C, C++ i Fortrana. Kompajlirani jezik je svaki onaj koji generira strojni jezik kako bi hardverski dio brže funkcionirao. Takvi klasični jezici su uglavnom puno brži, ali za potrebe ovog rada i većine slučajeva gdje nije potrebna automatizacija na razini milisekunda su brzine Pythona zadovoljavajuće (Martelli, 2006). Sam princip izrade programskog koda u Pythonu je dosta sličan logičkom tijeku ljudskih misli zbog čega je olakšan za čitanje i učenje korisniku nižeg stupnja vještina. Jedan o alternativnih jezika koji se mogao koristiti je i C ili njegova naprednija varijanta C++ programski jezik, koji se obično i koriste u programiranju tzv. engl. *embedded* sustava. Jednu od komparativnih analiza Pythona i C++ po parametrima memorije i vremena izvršenja su izradili Zehra i dr. (2020). Njihovo istraživanje se temeljilo na analizi najboljeg, najgoreg i prosječnog slučaja trajanja izvršenja i korištenja memorije prilikom 4 različita algoritma. Testirani su algoritmi traženja, sortiranja, umetanja i brisanja. U svakom slučaju je najbolje, najgore i prosječno vrijeme kao i korištenja memorija bilo kraće i manje kod C++ programskog jezika (slika 33).



Slika 33. Usporedba brzine rada C++ i Python-a (Zehra i dr., 2020)

Prema njihovim zaključcima, Python je sporiji uglavnom iz razloga što se svaka linija najprije prevodi u bytecode prije prijevoda u machine code te potom izvršava liniju po liniju programskog koda, dok C++ direktno generira machine code. Kao preporuku navode korištenje Pythona u početnim fazama, kao i kod projekata umjetne inteligencije i strojnog učenja, a C++ za brze softvere i računalne igre. Promatrajući njihove rezultate čije su razlike vremena izvršenja programa u milisekundama u projektu ovog rada nije bilo potrebe za korištenje C++ jezika. Zahtjev brzine rada će u potpunosti zadovoljiti odabrani Python.

3.9. Upravljački softver

Kako je bilo prije navedeno, programski kod se pisao u jeziku Python unutar popularnih uređivača PyCharm i Visual Studio Code na računalu s operativnim sustavom Windows. Preinake na samom mjestu rada su se pisale u uređivaču Thonny na operacijskom sustavu Raspbian Raspberrya Pi. Veliki dio koda se odnosi na grafička uređenja koja u ovom radu nisu presudna te zbog toga neće biti detaljno analizirana. Na početku koda vidljivog ispod su uvedene knjižnice potrebne funkcioniranje grafičkog i hardverskog dijela.

```
import time
import math
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog
from tktooltip import ToolTip
from PIL import Image, ImageTk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import
FigureCanvasTkAgg, NavigationToolbar2Tk
from matplotlib.animation import FuncAnimation
import csv
import RPi.GPIO as GPIO
import smbus
```

Knjižnice (engl. *library*) koje se koriste u programiranju se odnose na skupove već napisanih funkcija, metoda i klasa koje omogućuju olakšano stvaranje koda. Na ovaj način programeri ne moraju stvarati nove funkcije iznova već ih jednostavno pozovu iz knjižnice. Neke knjižnice su unaprijed ugrađene u Python, a neke je bilo potrebno skinuti i instalirati. Primjerice, protjek vremena je bitan prilikom osvježavanja podataka grafa mjerenja unutar prozora programa, kao i kod zapisa podataka u .csv oblik. Knjižnica math daje mogućnost izračuna vrijednosti s funkcijama poput pi, sinus i kosinus te sličnima. Konstanta pi se koristi prilikom izračuna posmičnih i brzina rezanja. Tkinter knjižnica omogućuje stvaranje grafičkog korisničkog sučelja te izbornike, prozore, gumbe, okvire i klizača unutar njega. Uvozom kao tk se skraćuje broj znakova kojima se poziva funkcija iz tkintera. Uvozom modula ttk iz tkintera je omogućeno generiranje tematskih grafičkih elemenata koji su atraktivniji. Modul filedialog je potreban

prilikom funkcije spremanja .csv datoteke s mjerenim vrijednostima. U ovoj funkciji se odvija upit korisnika gdje na računalu želi spremiti dokument. Tktooltip je modul kojim se mogu dodavati „info“ značajke. U programu se takvi prozori otvore prilikom dolaska pokazivača miša na ikonu „Info“. Knjižnica PIL (engl. *Python Imaging Library*) omogućuje obradu fotografija na način prikladan prepoznavanju Pythona kao podatka. PIL je potreban prilikom funkcije paljenja i gašenja stroja jer je „prekidač“ u prozoru u obliku slike „Off“ ili „On“. Matplotlib.pyplot je korišten za izradu grafova prikaza mjerenja. Vrste grafova je moguće odabrati pozivanjem različitih funkcija. Nadalje, matplotlib.backends.backend_tkagg i moduli unutar nje povezuju grafove i aplikacije tkintera kako bi se graf nalazio unutra prozora aplikacije. Modul NavigationToolbar2Tk na graf dodaje izbornik operacija navigacije, približavanja, smanjivanja i tome slično. Kako bi se graf konstantno ažurirao i bio sukladan mjerenju u jedinici vremena potrebno je raditi s modulom FuncAnimation iz knjižnice matplotlib.animation. Za stvaranje .csv (engl. *Comma-Separated Values*) s podacima mjerenja je potreban uvoz knjižnice csv za fukcionalno čitanje i stvaranje dokumenta. Definiranje i funkcionalnost GPIO pinova na Raspberry Pi je moguće s uvozom RPi.GPIO. Definiranje se odnosi na tip numeracije, broj pina te početna stanja frekvencije i radnog ciklusa. Komunikacija Raspberry Pi i A/D pretvornika opisanog u prijašnjim poglavljima je moguća uvozom smbus modula za interakciju putem I2C protokola na SDA i SCL pinovima. Ovaj modul se koristi i kod drugih perifernih uređaju poput senzora ili zaslona. Sljedeća bitna značajka ove aplikacija izrađene aplikacije je izrada klizača (engl. *slider*) koje služe za grafičko postavljanje željene frekvencija vrtnje motora. Ispod prikazan dio koda zaslužan za grafičku izradu i povezivanje s funkcijom stvaranja PWM signala.

```
#slider frekvencije
fr_slider=0.0
fr_range=50.0
scale1=tk.Scale(root, from_=0, to=fr_range,
orient=tk.HORIZONTAL, label="Odabrana frekvencija (Hz):
"+str(fr_slider), length=180)
scale1.set(fr_slider)
def horizontal1(event):
    global fr_slider
    if rucno1.get()==True:
        fr_slider=scale1.get()
        scale1.config(label="Odabrana frekvencija (Hz):
"+str(fr_slider))
        pwm_1()
        update1()
    else:
        scale1.set(fr_slider)
scale1.bind("<ButtonRelease-1>", horizontal1)
```

Prikazani segment koda opisuje definiranje funkcionalnih parametara klizača za određivanje frekvencije. Grafičke odredbe stvaranja i pozicioniranja klizača su definirane u redovima prije te postupak nije pretjerano zahtjevan. Prikazano određuje početnu vrijednost i raspon

frekvencije. Orijentacija klizača je horizontalna s definiranim tekstom koji će se prikazivati iznad njega. Linija scale1.set(fr_slider) postavlja početnu vrijednost na 0 Hz. Dalje je potrebno definirati funkciju kojom će se klizač pomicati prilikom klika miša i pomakom. Funkcija je nazvana horizontal1 i ima dva slučaja rada obzirom na to je li kvadratić ručnog upravljanja potvrđen ili ne. Postavljanje slučajeva je moguće pomoću engl. *if-else* uvjeta. U slučaju označenog kvadratića program povlači vrijednost s klizača, mijenja tekstualni natpis o odabranoj frekvenciji i pokreće funkcije generiranja PWM signala i ažuriranja prikaza aplikacije. U slučaju da ručno upravljanje nije odabrano, pomicanje klizača nije moguće te je postavljeno na početnu vrijednost od 0 Hz. Povezivanjem događaja buttonrelease-1 s funkcijom se postiže osvježavanje podataka nakon prestanka pomicanja miša i otpuštanja klika miša. Pozvana funkcija stvaranja PWM signala nije moguća bez definiranja parametara istih. Dio koda te svrhe je prikazan u nastavku.

```
#
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
FREQUENCY=500
GPIO.setup(33, GPIO.OUT)
pwm_glavni=GPIO.PWM(33, FREQUENCY)
pwm_glavni.start(0)
def pwm_1():
    global fr_range
    global fr_slider
    if stroj_state.get()==True:
        duty_cycle_glavni=(100/fr_range*fr_slider)
        pwm_glavni.ChangeDutyCycle(duty_cycle_glavni)
    else:
        duty_cycle_glavni=0
        pwm_glavni.ChangeDutyCycle(duty_cycle_glavni)
```

Prvi korak je određivanje tipa numeracije na Raspberry Pi u tip BOARD. Potom se stavlja naredba kojom će se gasiti sva moguća upozorenja. Ovom naredbom se otklanjaju sitna upozorenja koja se mogu dogoditi. Nakon toga se postavljaju početne vrijednosti frekvencije, radnog ciklusa i definiranje signala u PWM oblik. Nakon što je pin postavljen, potrebno ga je povezati s funkcijom koja će pokrenuti signal. Ova funkcija teče u dva smjera pomoću if-else uvjeta te povlači promjenjivu varijablu postavljene vrijednosti klizača. Ako je glodalica označena kao upaljena na slikovnom stanju On u prozoru aplikacije PWM signal se pokreće prema promjenjenom radnu ciklusu koji je jednak omjeru stanja i raspona klizača u postotnom obliku. U alternativnom slučaju radni ciklus iznosi 0 % i pokreće se promjena radnog ciklusa na 0 % PWM signala. Iduća bitna funkcija koda je očitanje analognog signala što ga proizvodi mjerni pretvornik snage. Sam princip djelovanja A/D konvertera je pojašnjen u prijašnjem poglavlju, no preostalo je objasniti interakciju između koda i komponente. Dio koda kojim se definira funkcija analog read je vidljiv ispod.

```
def analog_read():
    global volt_avg
    global snaga
    bus = smbus.SMBus(1)
    steps = 255
    i=0
    lista=[] # s pomoću liste se može napraviti average od
većeg broja očitanja
    while i<1:
        bus.write_byte(0x48, 0x42) #0x41 kanal
        value = bus.read_byte(0x48)
        vref=5.02
        volt=vref/steps*value
        i = i + 1
        lista.append(volt)
    volt_avg = sum(lista) / len(lista)
    volt_avg = round(volt_avg, 3)
    volt_avg = volt_avg*5 # Voltage divider x5
    snaga=volt_avg*346,6 #u watima
    root.after(500, analog_read)
analog_read()
```

Cilj funkcije je prikaz mjerenog napona signala i snage. Ponajprije je potrebno pokrenuti I2C komunikaciju komponenti i odrediti njihove parametre. Poziva se funkcija iz knjižnice smbus i definiraju koraci zapisa podatka. Korišteni 8-bitni konverter može očitati vrijednosti od 0-255 pri čemu je 255 maksimalna vrijednost napona. Potom se stvara lista u koju će se upisivati podaci periodično dok traje while petlja. Lista se stvara kako bi se dobila preciznija vrijednost na temelju prosječne vrijednosti više mjerenja. Unutar petlje se određuje adresa 0x48 na koju će se slati naredba za očitanje vrijednosti s kanala 0x42. Referentna vrijednost napona se postavlja na 5,02 V zbog točnosti rezultata, a mjerena vrijednost je jednaka omjeru referentne vrijednosti i ukupnog broja koraka zapisa umnožen s brojem aktivnih koraka. Konačna vrijednost napona istosmjernog signala je jednaka prosječnoj vrijednosti omjera sume članova liste i ukupnog broja članova zaokružena na 3 decimalna mjesta i umnožen s faktorom 5 kojeg uzrokuje djelilo napona. Mjerena snaga je tada jednaka umnošku izračunate volt avg i podešene osjetljivosti mjernog pretvornika k u iznosu 346,6 W/V. Na kraju, funkcija i mjerenje se ponavljaju svakih 500 milisekundi. Nakon što su željene vrijednosti izmjerene, konvertirane i očitane potrebno ih je snimiti u .csv oblik i spremiti na odgovarajuće mjesto u računalu. Prikaz ispod se odnosi na postupak definiranja funkcije snimanje_csv.

```
def snimanje_csv():
    global fr_slider
    global fr_slider2
    global volt_avg
    global snaga
```

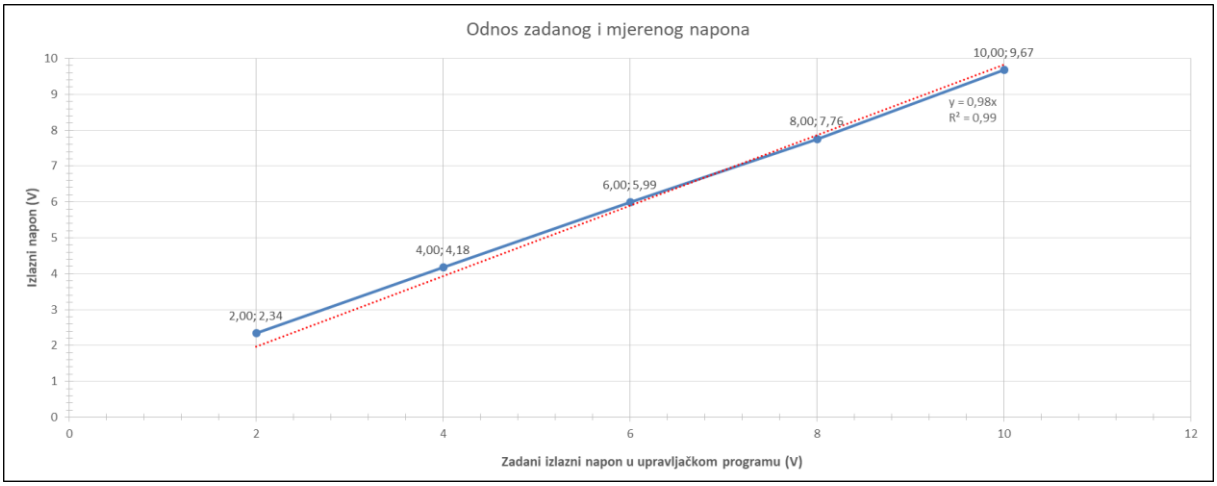
```
    if snimanje_state.get()==True and
mjerac_vremena.get()==False:
        csv_list.append((vrijeme_csv, fr_slider, fr_slider2,
volt_avg, snaga))
        root.after(100, snimanje_csv)
    elif snimanje_state.get()==True and
mjerac_vremena.get()==True:
        if vrijeme_csv<upis_vrijeme:
            csv_list.append((vrijeme_csv, fr_slider,
fr_slider2, volt_avg, snaga))
            root.after(100, snimanje_csv)
        else:
            snimanje_state.set(False)
    else:
        switch_csv() #samo da promjeni text
        file_path =
filedialog.asksaveasfilename(initialfile=str(ime_projekta),
defaultextension=".csv", filetypes=[("CSV file", ".csv")])
        headers=["Vrijeme", "Frekvencija glavnog motora",
"Frekvencija posmicnog motora", "Struja", "Snaga rezanja"]
        with open(file_path, "a", newline='') as file:
            writer = csv.writer(file, delimiter="\t")
            writer.writerow(headers) #prvo headeri
            writer.writerows(csv_list)
        csv_list.clear()
```

Ova funkcija povlači globalne varijable vrijednosti klizača te izmjerene vrijednosti napona i snage koje se ažuriraju u nekim drugim dijelovima koda. Potom se postavljaju uvjeti rada funkcije. Prvi uvjet se odnosi na potvrdu pokretanja snimanja i negativno stanje mjerača vremena kako bi se podaci pohranjivali u listu bez vremenskog ograničenja. Potom se definira pohrana podataka svakih 100 milisekundi. Ukoliko je potvrđeno pokretanje snimanja i pozitivno stanje mjerača vremena podaci se pohranjuju svakih 100 milisekundi dok zbrojeno vrijeme ne dođe do zadanog vremena mjerenja. U slučaju prekoračenja zadanog vremena se snimanje ne odvija. Nadalje, kada je snimanje završeno se otvara prozor gdje korisnik odabire mjesto pohrane na računalu te su definirani parametri imena i vrste dokumenta. Potrebno je i definirati zaglavlja te redoslijed upisa podataka sukladan onome kako su se pohranjivali u listu. Konačno se određuju tip razmaka na tabulator i organiziraju zaglavlja i podaci te se privremeno spremljena lista unutar koda prazni od prethodnih podataka. S time su objašnjeni oni dijelovi koda za koje se smatra funkcionalno najbitnijima za potrebe ovog rada. Dijelovi grafičkog uređenja aplikacije prividno zauzimaju najveći dio koda te se kao takvi najzahtjevnijima za razumijevanje. Ipak se radi o generiranju jednog polju unosa ili gumba za potvrdu koji se puno puta ponavljaju zbog velikog broja varijabli. Valja napomenuti da je moguće kod programa napisati na više različitih načina od prikazanog. Učenju i rješavanju

problematike su uvelike pomogli dostupni online materijali više od onih fizički tiskanih zahvaljujući programerskoj zajednici koja funkcionira na velikoj količini dijeljenog materijala.

4. REZULTATI I RASPRAVA

Provjera izvedenog upravljanja sustavom posmičnog kretanja, provedena je mjerenjem frekvencije vrtnje valjka na sustavu za posmično kretanje i to pri nazivno zadanim posmičnim brzinama od (5,5; 11; 16,5; i 33) m/min. Isto tako je provjerena izvedba upravljanja na sustavu za osiguranje glavnog kretanja i to mjerenjem frekvencije vrtnje radnog vretena ispitivane stolne glodalice na kojoj je prijenosni omjer remenica bio tako postavljen, da bi prema specifikaciji proizvođača stroja, frekvencija vrtnje radnog vretena trebala biti 6000 min^{-1}. Tijekom provjere upravljanja radom sustava glavnog kretanja, frekvencija vrtnje radnog vretena mjerena je za zadane ulazne frekvencije od (10; 20; 30; 40 i 50) Hz, što onda odgovara očekivanim frekvencijama vrtnje od (1200; 2400; 3600; 4800 i 6000) min^{-1}. Sukladno tome odgovarajući izlazni naponi na upravljačkom dijelu sustava trebali bi biti (2,0; 4,0; 6,0; 8,0 i 10,0) V. Teoretski, moguće je oba motora držati u razredima maksimalnih brzina rotacije te iste smanjivati pomoću frekventog pretvarača. Ova praksa ipak nije preporučljiva obzirom da je poželjniji rad elektromotora pri višim frekvencijama vrtnje. Rezultantni generirani naponi su mjereni multimetrom te nisu bili idealnog iznosa. Nije u potpunosti poznato koji je uzrok takvim oscilacijama električnog napona upravljačkog signala. Jedan od vjerojatnijih uzroka takvim varijacijama može biti korišteni D/A pretvornik koji je predviđen za ulazni PWM signal razine od 5 V, a u projektu je poteciometrom podešen na 3,3 V koje generira mikroračunalo na svojim GPIO portovima, te je moguće da uslijed toga dolazi do lošijih perfomansi. Kako za korišteni konverter nismo uspjeli pribaviti detaljnu specifikaciju, nismo u mogućnosti tvrditi hipoteze o potencijalnim greškama na istom. Mjerenje napona na izlazima D/A pretvornika provedeno je pomoću multimetra UNI-T UT51, koji je s obzirom na raspon mogućih vrijednosti upravljačkog signala, postavljen za mjerenje istosmjernog napona u granicama od 0 V do 20 V, što odgovara mjernom rasponu R = 19,99 V. Iz karakteristike korištenog multimetra, za zadano mjerno područje, vidljivo je da je njegova točnost ±(0,5 % očitane vrijednosti + 1 digit) uz rezoluciju 0,01 V. Na slici 34 prikazani su rezultati izmjerenih vrijednosti napona na izlazu D/A pretvornika u ovisnosti o zadanim vrijednostima u upravljačkom programu.



Slika 34. Ovisnost očekivanog i generiranog naponskog signala

U idealnom slučaju očekivali bi da je jednadžba tog pravca dana izrazom U_{izl} = 1,00 * U_{zadano}, ali iz prezentiranih rezultata može se vidjeti da praktično dobivena karakteristika odstupa od teorijske karakteristike. Za svaku zadanu vrijednost upravljačkog napona na izlazu D/A pretvornika izmjereno je pet vrijednosti generiranog izlaznog napona. Oscilacije naponske mjerene vrijednosti su matematički obrađene. Na temelju izmjerenih vrijednosti izračunate su srednje vrijednosti pojedinih mjerenja i eksperimentalna standardna odstupanja – $s(U_{izl})$ te su iznosila: (2,34 ± 0,01) V; (4,18 ± 0,01) V; (5,99 ± 0,01) V; (7,76 ± 0,02) V i (9,67 ± 0,02) V. S obzirom na karakteristiku multimetra (± 0,5% očitane vrijednosti + 1 digit) granična pogreška pojedinih mjerenja iznosi: 0,02 V; 0,03 V; 0,04 V; 0,05 V i 0,06 V. Za primjer izračuna granične pogreške digitalnog uređaja će se obraditi primjer najveće oscilacije od zadane vrijednosti. Ona iznosi 0,33 V pri traženoj frekvenciji 50 Hz. Proporcionalna (p_p) i konstantna pogreška (p_a) je u tim uvjetima jednaka:

$$p_p = \frac{\%}{100} * očitanje = \frac{0{,}5}{100} * 9{,}67 = 0{,}05\ V$$

$$p_a = \frac{z}{N_D} * R = \frac{1}{1999} * 19{,}99 = 0{,}01\ V.$$

Ukupna pogreška jednaka zbroju proporcionalne i konstantne pogreške te iznosi:

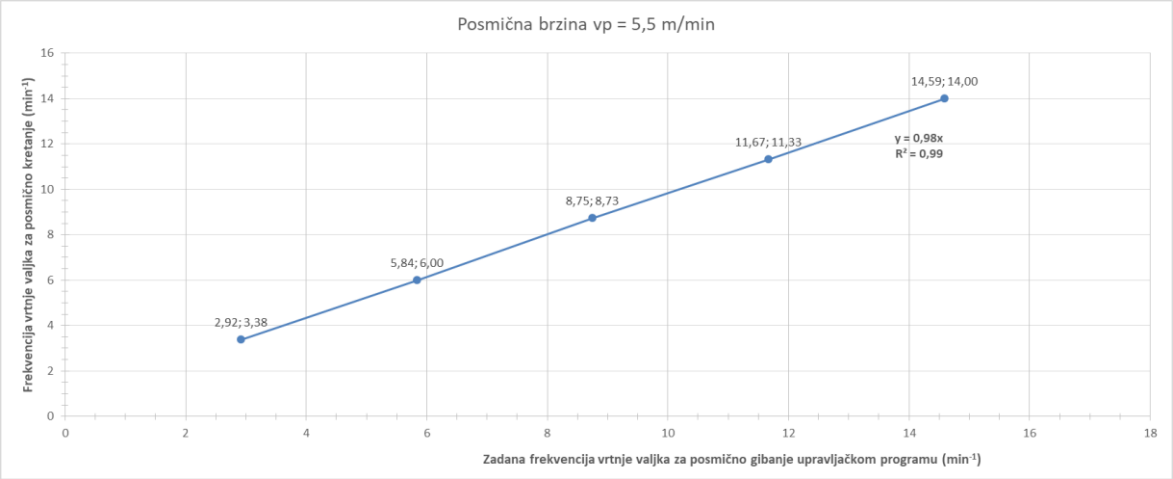
$$p = p_a + p_p = 0{,}05 + 0{,}01 = 0{,}06\ V.$$

Maksimalna relativna pogreška (δ_{max}) za taj slučaj onda je jednaka:

$$\delta_{max} = \frac{2 * p}{\overline{U_{izlazno}}} * 100 = \frac{2 * 0{,}06}{9{,}67} * 100 = 1{,}24\ \%.$$

Utjecaj greške mjerenja u odnosu na očitanu frekvenciju u najgorem slučaju rezultira oscilacijom od 0,6 Hz. Izračunati podaci apsolutne i relativne pogreške na mjerenja multimetrom su u skladu s zahtjevima te je njen utjecaj u traženim uvjetima rada gotovo zanemariv. Oscilacije frekvencije ove veličine mogu zadovoljiti uvjete rada prerade drva. Blago povećanje standardnih devijacija s porastom mjerene vrijednosti je sukladno rastu proporcionalnog dijela greške. Sljedeći uzročnik greške osciliranja traženih frekvencija vrtnje je frekventni pretvarač. Prema specifikaciji proizvođača izlazna frekvencija se nalazi unutar ±1 % maksimalne frekvencije pri 25 °C (±10 °C). Prema ovom podatku, izlazna frekvencija se nalazi unutar ±0,5 Hz. Prosječni iznosi frekvencija očitanih barem 5 puta unutar razreda su iznosili: (11,71; 21,01; 30,04; 39,03 i 48,69) Hz. Najveća oscilacija frekvencije prema generiranom naponskom razredu u ovom slučaju je pri najvišoj frekvenciji i iznosi 0,34 Hz. Rezultat ove analize ne može potvrditi da li su odstupanja frekvencije uzrokovana nepreciznim očitanjem napona na multimetru ili greškama unutar strujnog kruga frekventnog pretvarača. Ipak, najveća oscilacija ovog dijela sustava u iznosu od 0,34 Hz je više nego zadovoljajuća za sustave s kojim se susrećemo u drvnoj industriji. Daljnje mjerenje se odnosilo na provjeru broja okretaja rotirajućih dijelova u odnosu na postavljenu frekvenciju vrtnje. Korišteni tahometar

prema specifikaciji ima točnost ±(0,05 % očitane vrijednosti + 1 digit) i rezoluciju 0,1 min^{-1}. Na slici 35 je prikaz graf ovisnosti mjerenog i postavljenog broja okretaja pri posmičnoj brzini 5,5 m/min.



Slika 35. Ovisnost postavljene i mjerene frekvencije vrtnje

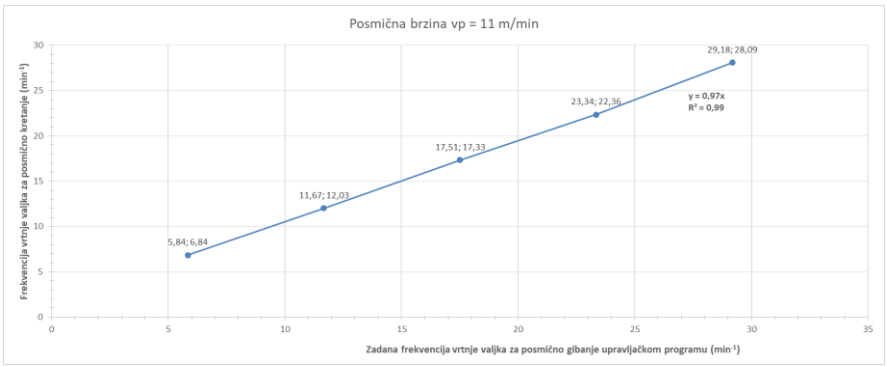
Izračunom standardne devijacije očitanih mjernih vrijednosti po razredima frekvencije vrtnje iznosi su: (3,4 ± 0,1) min^{-1}; (6,0 ± 0,1) min^{-1}; (8,7 ± 0,1) min^{-1}; (11,3 ± 0,2) min^{-1} i (14,0 ± 0,2) min^{-1}. Primjer izračuna granične pogreške u najgorem slučaju oscilacije se računa prema sljedećem:

$$p = \left(\frac{0{,}05}{100} * 14\right) + \left(\frac{1}{99999} * 99999\right) = 0{,}007 + 1 = 1{,}007 \approx 1{,}00 \ \frac{o}{min}.$$

Ovaj podatak govori da se stvarna vrijednost mjerene frekvencije vrtnje nalazi u intervalu između 13,00 min^{-1} i 15,00 min^{-1}. Prema rezultatu granične pogreške moguće je izračunati maksimalnu relativnu pogrešku u najgorem slučaju i iznosi:

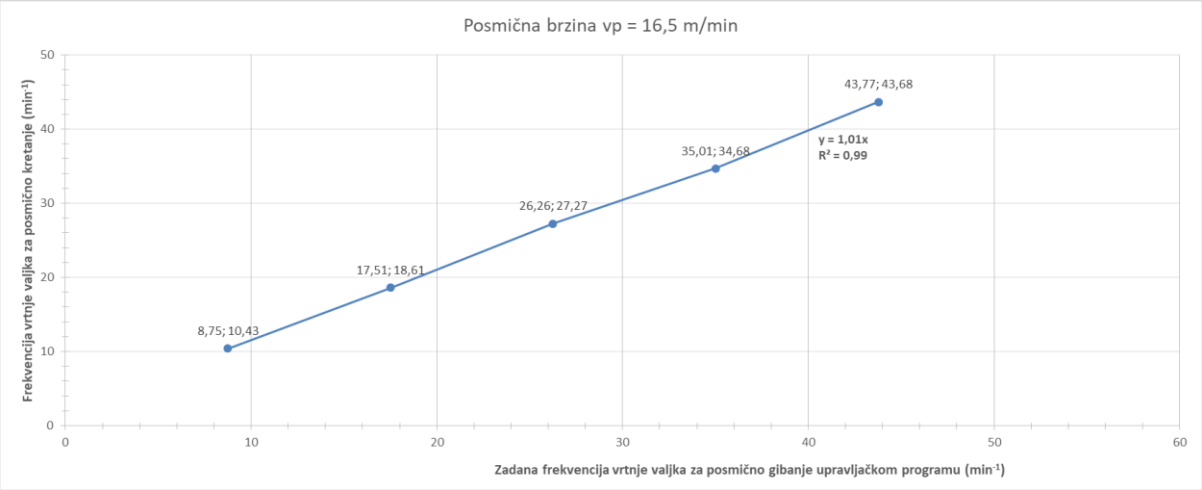
$$G_{max} = \frac{2 * p}{\overline{n_v}} * 100 = \frac{2 * 1{,}01}{14} * 100 = 14{,}43\ \%.$$

Ovaj podatak ukazuje na postotno učešće granične pogreške u izmjerenoj vrijednosti najveće oscilacije od zadanih postavki. Veliko učešće apsolutne greške u rezultatu je objašnjivo kroz to da se uređaj koristi pri niskim područjima mjernog raspona. Sljedeći mjereni slučaj je bio u razredu posmične brzine 11 m/min. Graf ovisnosti je prikazan na slici 36.



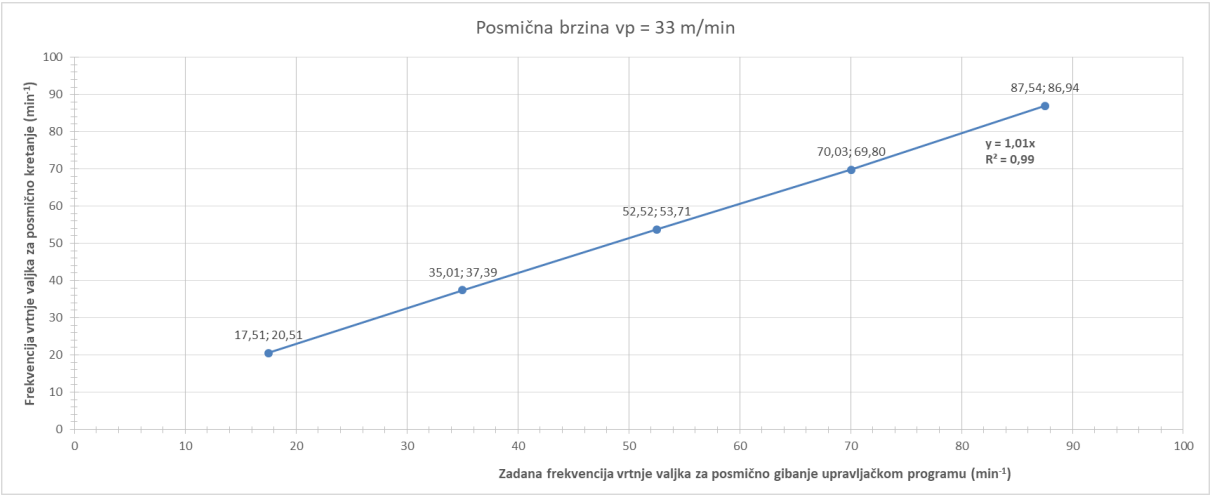
Slika 36. Ovisnost postavljene i mjerene frekvencije vrtnje

Izračunom standardne devijacije očitanih mjernih vrijednosti sljedećem razredu frekvencije vrtnje iznosi su: (6,8 ± 0,3) min^{-1}; (12,0 ± 0,1) min^{-1}; (17,3 ± 0,3) min^{-1}; (22,4 ± 0,3) min^{-1} i (28,1 ± 0,3) min^{-1}. Idući razred mjerenja se odnosi na posmičnu brzinu 16,5 m/min. Graf ovisnosti tog slučaja je prikazan na slici 37.



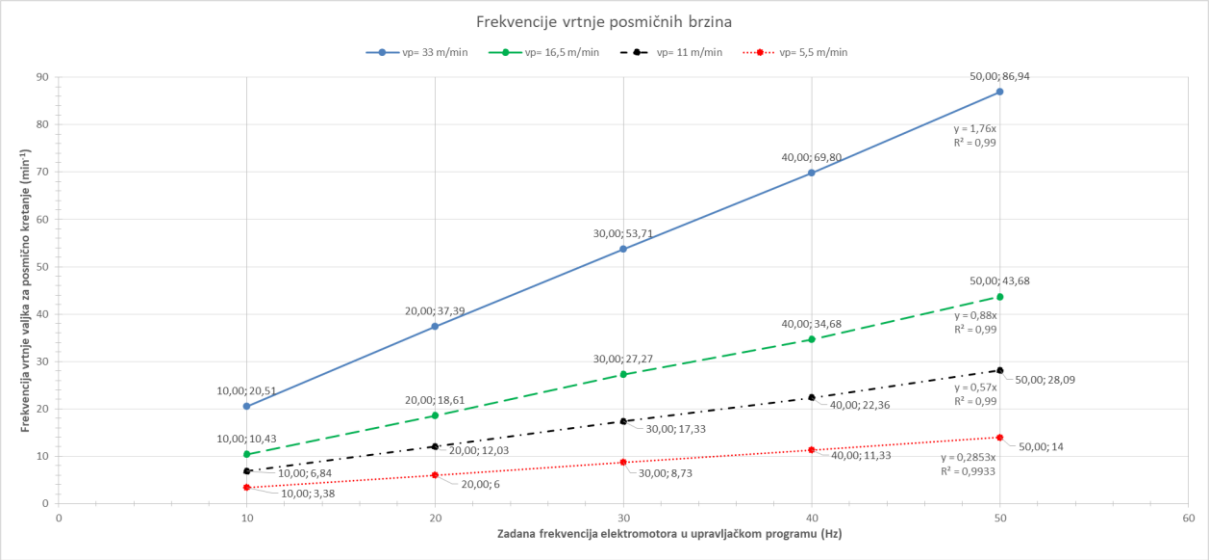
Slika 37. Ovisnost postavljene i mjerene frekvencije vrtnje

Izračunom standardne devijacije očitanih mjernih vrijednosti sljedećem razredu frekvencije vrtnje iznosi su: (10,4 ± 0,3) min^{-1}; (18,6 ± 0,2) min^{-1}; (27,3 ± 0,3) min^{-1}; (34,7 ± 0,3) min^{-1} i (43,7 ± 0,4) min^{-1}. Četvrti i posljednji mjereni slučaj je bio pri razredu posmične brzine 33 m/min. Graf ovisnosti je prikazan na slici 38.



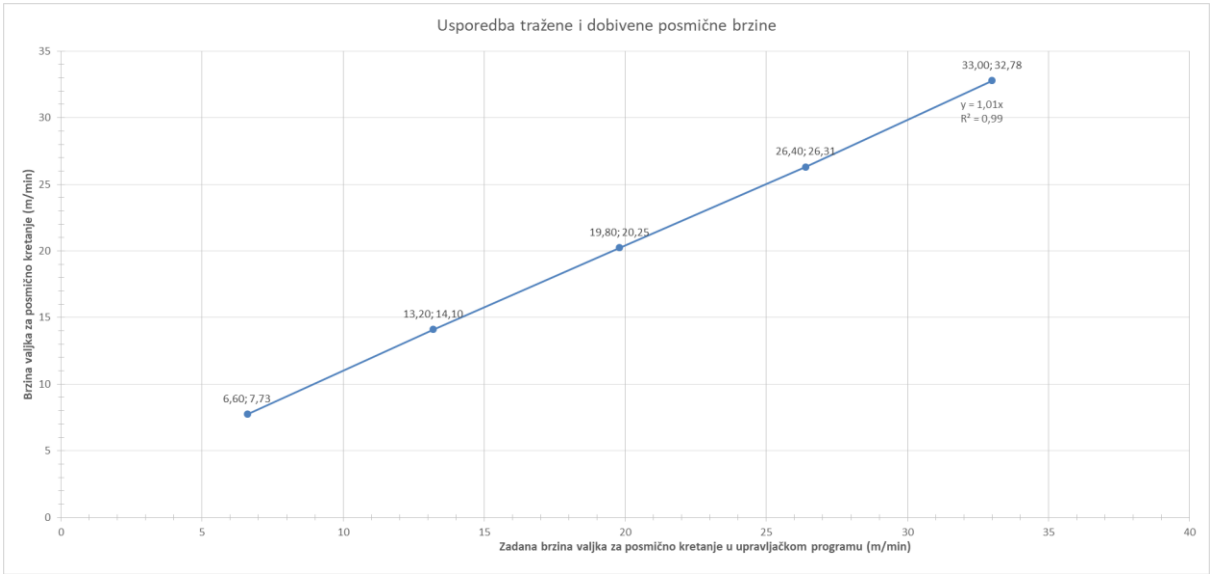
Slika 38. Ovisnost postavljene i mjerene frekvencije vrtnje

Izračunom standardne devijacije očitanih mjernih vrijednosti sljedećem razredu frekvencije vrtnje iznosi su: (20,5 ± 0,3) min^{-1}; (37,4 ± 0,1) min^{-1}; (53,7 ± 0,2) min^{-1}; (69,8 ± 0,6) min^{-1} i (86,9 ± 0,6) min^{-1}. Zbirni prikaz statičkih karakteristika promatranih posmičnih brzina u odnosu za zadanu frekvenciju je vidljiv na slici 39.



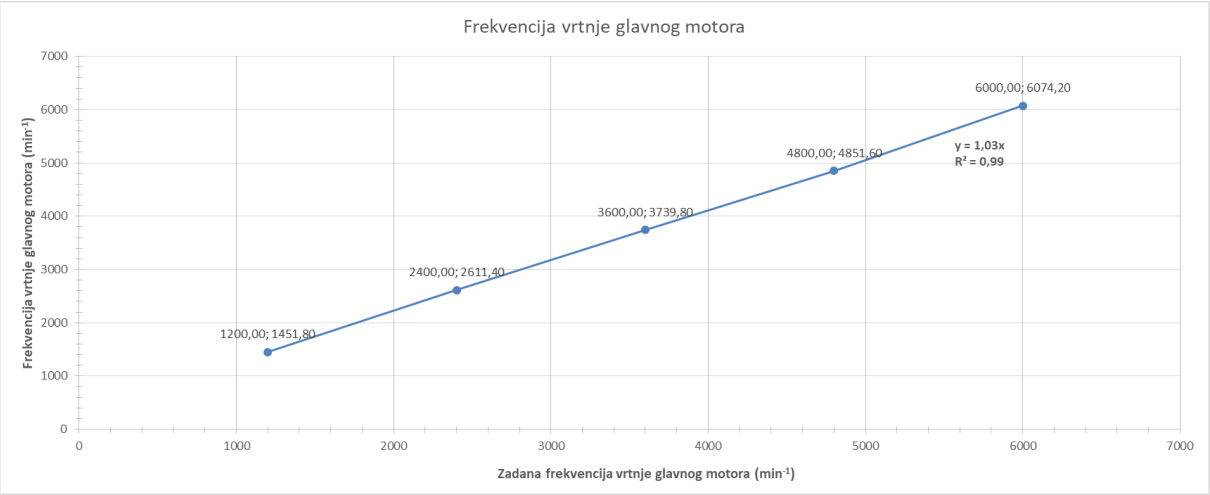
Slika 39. Zbirni prikaz statičkih karakteristika posmičnog gibanja

Analizirani su najgori slučajevi pri svakom mjerenju. Kroz sve grafove je vidljivo da su manje oscilacije prisutne pri većem mjerenom broja okretaja. Potvrda tome je generalno smanjenje maksimalne relativne pogreške u svim slučajevima. Kako je već prije spomenuto, ovim sustavom je moguća kontrola posmične brzine od 0 m/min do 33 m/min unutar najveće postavljene brzine na stroju. Promjer pogonskih valjaka posmičnog gibanja na glodalici je 120 mm. Prikaz ovisnosti zadane i prosječne izmjerene posmične brzine obratka je vidljiv na slici 40.



Slika 40. Ovisnost postavljene i mjerene posmične brzine

Iako je ovakav način rada moguć, nije preporučljivo dovoditi motor u područje niske frekvencije vrtnje. Na sličan način provedena su mjerenja i na sustavu za osiguravanje glavnog kretanja. Generirani naponski signali i frekvencije su identične kao i kod posmičnog gibanja. Ovisnost mjerene i očekivane vrijednosti frekvencije vrtnje je vidljiv na slici 41.



slika 41. – Ovisnost postavljene i mjerene frekvencije vrtnje

Izračunom standardne devijacije očitanih mjernih vrijednosti sljedećem razredu frekvencije vrtnje iznosi su: (1451,80 ± 5,02) min^{-1}; (2611,40 ± 2,70) min^{-1}; (3739,80 ± 25,59) min^{-1}; (4851,60 ± 7,13) min^{-1} i (6074,20 ± 1,3) min^{-1}.

.

4.1. Izrada približnog pogonskog dijagrama elektromotora

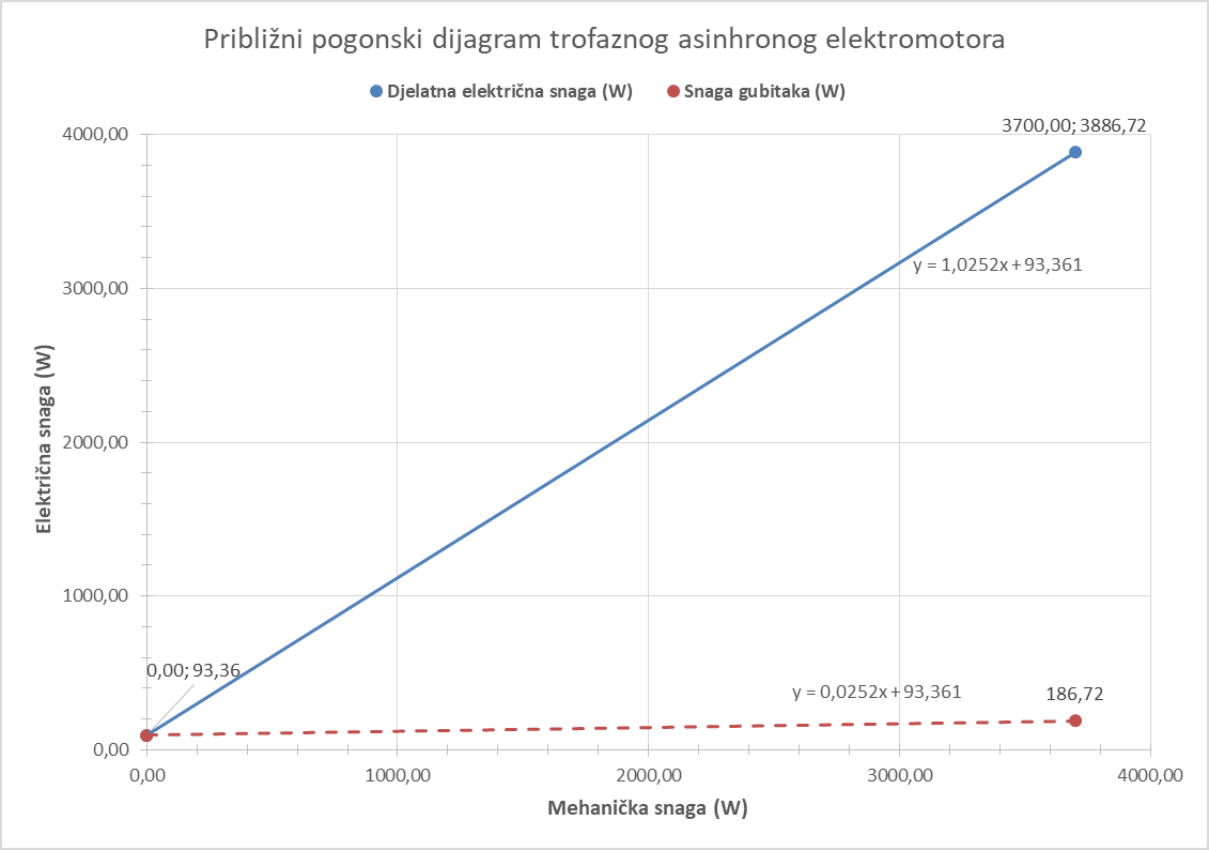
Poznavajući postupak izrade približnog pogonskog dijagrama i očitanjem podataka s pločica elektromotora moguće je izraditi dijagrame i za elektromotore glodalice koji su korišteni u projektu ovog rada. Glavni elektromotor glodalice sljedeće očitane karakteristike: P = 3,7 kW, U = 400 V, I = 6,6 A, $cos\varphi$ = 0,85, n = 2830 o/min. Prema tim podacima i izračunom ostalih slijedi:

$$P_{el} = \sqrt{3} * U * I * cos\varphi = \sqrt{3} * 400 * 6{,}6 * 0{,}85 = 3886{,}72\ W.$$

$$P_{g0} = P_{el} - P = 3886{,}72 - 3700 = 186{,}72\ W.$$

$$P_{g0} \approx \frac{P_g}{2} = \frac{186{,}72}{2} = 93{,}36\ W.$$

Unosom ovih vrijednosti prema opisanom postupku dobiven je približni pogonski dijagram trofaznog asinhronog elektromotora prikazan na slici 42.



Slika 42. Graf približnog pogonskog dijagrama glavnog motora

Dijagrami su stvoreni prema podacima s pločice motora pod pretpostavkom da su spajani u trokut. Daljnji razvoj dijagrama je moguć mjerenjem i unosom podataka djelatne snage pri uvjetima glodanja i praznom hodu. Pomoću tih podataka je moguće izračunati mehaničku snagu u praznom hodu. Nadalje unosom djelatne snage pri maksimalnim parametrima glodanja se može intervalno odrediti snaga rezanja prema razlici potrebne mehaničke snage i mehaničke snage u praznom hodu. Ovaj postupak rad nije obradio zbog problematike kalibriranja mjernog pretvornika snage, ali je projekt obuhvatio stvaranje hardvera za očitanje povratne veze uz poznatu statičku osjetljivost predloženog pretvornika snage. Ovim putem je moguća simulacija mjerenja djelatne električne snage.

5. ZAKLJUČAK

Dobiveni rezultati mjerenja i razvijeni sustav za kontrolu brzine rezanja i posmične brzine stolne glodalicezadovoljili su tražene zahtjeve. Uspješno je provedeno upravljanje oba trofazna asinhrona elekromotora te testiran sustav za očitanje analognog signala što bi ga generirao mjerni pretvornik snage Iskra MI400. U radu je korištena različita oprema od kojih većini nije poznata potpuna specifikacija koja bi dodatno utvrdila uzročno-posljedične veze mjerenih varijabli. Iako su rezultati mjerenja odstupali od očekivanog, prihvaćeni su za mogućnost rada u stvarnim uvjetima obzirom da prerada drva dopušta veće promjene parametara nego što je slučaj u npr. metaloprerađivačkoj industriji. Najveće greške sustava su se javljale prilikom rada niskih frekvencija i malih broja okretaja rotirajućih dijelova što ionako nije poželjno u stvarnim uvjetima rada. Nije bilo moguće odrediti da li su odstupanja vrijednosti uzrokovane greškama unutar sustava ili granične pogreške mjernih instrumenata. Ipak je potrebno težiti daljnjem unapređenju sustava s manjim oscilacijama te testirati drugačije pristupe generiranja PWM signala. Potvrđena je i hipoteza o preinaki standardnog stroja u računalno kontroliran pomoću relativno jeftinih komponenti. Radu je dopušten daljnji razvitak kroz mjerenje djelatne električne snage u uvjetima rada i završetak izračuna potrebne snage rezanja pomoću približnog pogonskog dijagrama elektromotora.

6. LITERATURA

1. Bego V., 1979: Mjerenja u elektrotehnici, četvrto dopunjeno izdanje. Tehnička knjiga Zagreb
2. Berger R., 2014: Strategy Consultants Gmbh – Think act Industry 4.0
3. Buršić P., 2017: Usporedna analiza objektivno-orijentiranih programskih jezika. Završni rad. Odjel za informacijsko-komunikacijske tehnologije, Sveučilište Jurja Dobrile u Puli.
4. Costa &, J., Almeida, G., & Cordeiro, A. (2019): Automated Pick and Place Solution Based on Raspberry Pi. Interantional Journal of Engineering Trends and Technology, Vol 67, Is. 10.
5. Hamm Đ., 1970: Približni pojednostavljeni način određivanja utroška el. Energije i predane mehaničke energije trofaznih asinhronih indukcionih elektromotora. Drvna industrija br. 7-8.
6. Huges J., 2010: Real World Instrumentation with Python. O'Reilly Media Inc., Sebastopol, Kalifornija, SAD.
7. HOMAG Group AG - https://www.homag.com/en/
8. Kolberg D., Zühlke D., 2015: Lean Automation enabled by Industry 4.0 technologies. IFAC-PapersOnLine 48-3: 1870-1875.
9. Kootanaee A., Babu K., Talari H., 2013: Just-In-Time manufacturing system: From introduction to implement. International Journal of Economics, Business and Finance.
10. Kropivšek J., Grošelj P., 2019: Digital development of slovenian wood industry. Drvna industrija 71(2) 139-148(2020).
11. Legg B., Dorfner B., Leavengood S., Hansen E., 2021: Industry 4.0 implementation in US primary wood products industry. Drvna industrija 72(2): 143-153.
12. Loureiro, G. F., & Colombo, A. (2024). Reduction of energy consumption using remote Variable Frequency Drive (VFD). Caderno Pedagógico, 21(1), 1076-1085.
13. Landscheidt S., Kans M., 2016: Automation practices in wood product industries: lessons learned, current practices an future perspectives. The 7th Swedish Production Symposium SPS, Lund, Švedska.
14. Majstorović V., Mitrović R., Mišković Ž., 2022: Industry 4.0 in Serbia – state of development. Serbian Journal of Management 17(1): 5-14.
15. Martelli A., 2006: Python in a Nutshell, Second Edition. O'Reilly Media Inc., Sebastopol, Kalifornija, SAD.
16. Peko I., 2015: Na putu prema četvrtoj industrijskoj revoluciji: analiza stanja hrvatske industrije. Seminarski rad. Fakultet elektrotehnike, strojarstva i brodogradnje, Sveučilište u Splitu.
17. Ramos-Maldonado M., Aguilera-Carrasco C., 2022: Trends and opportunities of Industry 4.0 in wood manufacturing processes. Engineered wood products for construction, London, Ujedinjeno Kraljevstvo.
18. Singh J., Singh H., 2009: Kaizen philosophy: A review of literature. The Icfai University Journal of Operation Manegement, Vol. VIII, No.2.

19. Zehra, F., Javed, M., Khan, D., & Pasha, M. (2020). Comparative analysis of C++ and python in terms of memory and time.

PRILOG – Potpuni kod upravljačkog programa

```
import time
import math
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog
from tktooltip import ToolTip
from PIL import Image, ImageTk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg,
NavigationToolbar2Tk
from matplotlib.animation import FuncAnimation
import csv
import RPi.GPIO as GPIO
import smbus


root=tk.Tk()
root.title("Upravljanje strojem")
#root.state("zoomed")

#frame za entry
frame_alat=tk.LabelFrame(root, text="Glavni pogon - izračun maksimalne
brzine", width=400, height=240)
frame_alat.place(x=5, y=0)
frame_em12=tk.LabelFrame(root, text="Glavni pogon - upravljanje",
width=400, height=240)
frame_em12.place(x=390, y=0)
frame_em21=tk.LabelFrame(root, text="Posmični pogon - izračun maksimalne
brzine", width=400, height=240)
frame_em21.place(x=775, y=0)
frame_em22=tk.LabelFrame(root, text="Posmični pogon - upravljanje",
width=400, height=240)
frame_em22.place(x=1160, y=0)
frame_snimanje=tk.LabelFrame(root, text="Snimanje podataka - CSV file",
width=400, height=240)
frame_snimanje.place(x=1545, y=0)

#promjer glodala
promjer_glodala=0.0
text_promjer_glodala=tk.StringVar()
text_promjer_glodala.set("Promjer glodala (mm):
"+str(round(promjer_glodala, 3)))
e_promjer_glodala=ttk.Entry(root, width=40,
textvariable=text_promjer_glodala)
def promjer_alata(event):
    global promjer_glodala
    if event.type=="9":
        text_promjer_glodala.set("")
    elif event.keysym=="Return":
        try:
            promjer_glodala=float(e_promjer_glodala.get())
            update1()
```

```
            root.focus_set()
        except:
            root.focus_set()
    else:
        text_promjer_glodala.set("Promjer glodala (mm):
"+str(round(promjer_glodala, 3)))
e_promjer_glodala.bind("<FocusIn>", promjer_alata)
e_promjer_glodala.bind("<FocusOut>", promjer_alata)
e_promjer_glodala.bind("<Return>", promjer_alata)

#
info1= "Za prikaz parametara i točnost izračunatih podataka potrebno je
unesti što točnije podatke u sva otvorena polja."\
     " Unos se vrši pritiskom na tipku 'Enter'."
button_info1=tk.Button(root, text="Info", foreground="green")
tip1=ToolTip(button_info1, msg=info1)
tip1.config(bd=1, bg="blue")

#nazivni okretaji EM
okretaji_em1=0.0
text_okretaji_em1=tk.StringVar()
text_okretaji_em1.set("Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em1, 3)))
e_okretaji_em1=ttk.Entry(root, width=40, textvariable=text_okretaji_em1)
def okretaji_m1(event):
    global okretaji_em1
    if event.type=="9":
        text_okretaji_em1.set("")
    elif event.keysym=="Return":
        try:
            okretaji_em1=float(e_okretaji_em1.get())
            update1()
            root.focus_set()
        except:
            root.focus_set()
    else:
        text_okretaji_em1.set("Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em1, 3)))
e_okretaji_em1.bind("<FocusIn>", okretaji_m1)
e_okretaji_em1.bind("<Return>", okretaji_m1)
e_okretaji_em1.bind("<FocusOut>", okretaji_m1)

#nazivni okretaji vratila
okretaji_vratilo1=0.0
text_okretaji_vratilo1=tk.StringVar()
text_okretaji_vratilo1.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo1, 3)))
lista_okretaji_vratilo1=["1400.0", "3500.0", "6000.0", "8000.0"]
e_okretaji_vratilo1=ttk.Combobox(root, textvariable=text_okretaji_vratilo1,
width=39, values=lista_okretaji_vratilo1, exportselection=False)
def okretaji_v1(event):
    global okretaji_vratilo1
    global koef_emv1
```

```
    if event.num==1:
        text_okretaji_vratilo1.set("")
    elif event.type=="10":
        text_okretaji_vratilo1.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo1, 3)))
    else:
        try:
            okretaji_vratilo1=float(text_okretaji_vratilo1.get())
            update1()
            root.focus_set()
        except:
            root.focus_set()
e_okretaji_vratilo1.bind("<<ComboboxSelected>>", okretaji_v1)
e_okretaji_vratilo1.bind("<Return>", okretaji_v1)
e_okretaji_vratilo1.bind("<Button-1>", okretaji_v1)
e_okretaji_vratilo1.bind("<FocusOut>", okretaji_v1)

#koeficijent prijenosa
koef_emv1=0.0
text_koef_emv1=tk.StringVar()
text_koef_emv1.set("Izračun koeficijenta prijenosa (i):
"+str(round(koef_emv1, 3)))
e_koef_emv1=ttk.Entry(root, width=40, textvariable=text_koef_emv1)

#maksimalna brzina rezanja
max_brzina1=0.0
text_max_brzina1=tk.StringVar()
text_max_brzina1.set("Maksimalna brzina rezanja pri 50Hz (m/s):
"+str(round(max_brzina1, 3)))
e_max_brzina1=ttk.Entry(root, width=40, textvariable=text_max_brzina1)
def max_vr(event):
    global max_brzina1
    global fr_slider
    if event.num==1:
        text_max_brzina1.set("")
    elif event.type=="10":
        text_max_brzina1.set("Maksimalna brzina rezanja pri 50Hz (m/s):
"+str(round(max_brzina1, 3)))
    else:
        try:
            max_brzina1=float(text_max_brzina1.get())
            update1()
            root.focus_set()
        except:
            root.focus_set()
e_max_brzina1.bind("<<ComboboxSelected>>", max_vr)
e_max_brzina1.bind("<Return>", max_vr)
e_max_brzina1.bind("<Button-1>", max_vr)
e_max_brzina1.bind("<FocusOut>", max_vr)

#slider frekvencije
fr_slider=0.0
fr_range=50.0
```

```
scale1=tk.Scale(root, from_=0, to=fr_range, orient=tk.HORIZONTAL,
label="Odabrana frekvencija (Hz): "+str(fr_slider), length=180)
scale1.set(fr_slider)
def horizontal1(event):
    global fr_slider
    if rucno1.get()==True:
        fr_slider=scale1.get()
        scale1.config(label="Odabrana frekvencija (Hz): "+str(fr_slider))
        pwm_1()
        update1()
    else:
        scale1.set(fr_slider)
scale1.bind("<ButtonRelease-1>", horizontal1)

#rucni odabir frekvencije
rucno1=tk.BooleanVar()
rucno1.set(False)
def slider1_reset_checkbutton():
    global fr_slider
    if rucno1.get()==True:
        e_brzina_fr1.config(state="readonly")
    else:
        e_brzina_fr1.config(state="normal")
rucno_check=tk.Checkbutton(root, text="Ručno upravljanje frekvencijom",
foreground="green", variable=rucno1, command=slider1_reset_checkbutton)

#brzina nakon frekventnog
brzina_fr1=float((max_brzina1/fr_range)*fr_slider)
text_brzina_fr1=tk.StringVar()
text_brzina_fr1.set("Odabrana brzina rezanja (m/s): "+str(round(brzina_fr1,
3)))
e_brzina_fr1=ttk.Entry(root, width=40, textvariable=text_brzina_fr1)
def vr_fr(event):
    global brzina_fr1
    global fr_slider
    if event.type=="9":
        text_brzina_fr1.set("")
    elif event.keysym=="Return":
        try:
            if float(e_brzina_fr1.get())<=max_brzina1:
                brzina_fr1=float(e_brzina_fr1.get())
                fr_slider=float(brzina_fr1/(max_brzina1/fr_range))
                slider_frekvencija_show=round(fr_slider, 3)
                scale1.set(fr_slider)
                scale1.config(label="Odabrana frekvencija (Hz):
"+str(slider_frekvencija_show))
                update1()
                pwm_1()
                root.focus_set()
            else: root.focus_set()
        except:
            root.focus_set()
    else:
```

```
        text_brzina_fr1.set("Odabrana brzina rezanja (m/s):
"+str(round(brzina_fr1, 3)))
e_brzina_fr1.bind("<FocusIn>", vr_fr)
e_brzina_fr1.bind("<FocusOut>", vr_fr)
e_brzina_fr1.bind("<Return>", vr_fr)

#okretaji motora nakon frekventnog
okretaji_em_fr1=float((okretaji_em1/fr_range)*fr_slider)
text_okretaji_em_fr1=tk.StringVar()
text_okretaji_em_fr1.set("Odabrani okretaji motora (n/min):
"+str(round(okretaji_em_fr1, 3)))
e_okretaji_em_fr1=ttk.Entry(root, width=40,
textvariable=text_okretaji_em_fr1)

#okretji vratila nakon frekventnog
okretaji_vratila_fr1=float((okretaji_vratilo1/fr_range)*fr_slider)
text_okretaji_vratila_fr1=tk.StringVar()
text_okretaji_vratila_fr1.set("Odabrani okretaji vratila (n/min):
"+str(round(okretaji_vratila_fr1, 3)))
e_okretaji_vratila_fr1=ttk.Entry(root, width=40,
textvariable=text_okretaji_vratila_fr1)

#checkbutton izracuna prijenosa
prijenos1=tk.BooleanVar()
prijenos1.set(True)
def checkbutton_prijenos1():
    global okretaji_em1
    global okretaji_vratilo1
    global koef_emv1
    global okretaji_em_fr1
    global okretaji_vratila_fr1
    global max_brzina1
    global brzina_fr1
    if prijenos1.get()==True:
        e_promjer_glodala.config(state="normal")
        promjer_glodala=0.0
        text_promjer_glodala.set("Promjer glodala (mm):
"+str(round(promjer_glodala, 3)))
        e_okretaji_em1.config(state="normal")
        okretaji_em1=0.0
        text_okretaji_em1.set(f"Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em1, 3)))
        e_okretaji_vratilo1.config(state="normal")
        okretaji_vratilo1=0.0
        text_okretaji_vratilo1.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo1, 3)))
        e_koef_emv1.config(state="readonly")
        koef_emv1=0.0
        text_koef_emv1.set("Koeficijent prijenosa (i):
"+str(round(koef_emv1, 3)))
        e_max_brzina1.config(state="readonly")
        max_brzina1=0.0
        text_max_brzina1.set("Maksimalna brzina rezanja pri 50Hz (m/s):
```

```
"+str(round(max_brzina1, 3)))
        e_okretaji_em_fr1.config(state="readonly")
        brzina_fr1=0.0
        text_brzina_fr1.set("Odabrana brzina rezanja (m/s):
"+str(round(brzina_fr1, 3)))
        okretaji_em_fr1=0.0
        text_okretaji_em_fr1.set("Odabrani okretaji motora (n/min):
"+str(round(okretaji_em_fr1, 3)))
        e_okretaji_vratila_fr1.config(state="readonly")
        okretaji_vratila_fr1=0.0
        text_okretaji_vratila_fr1.set("Odabrani okretaji vratila (n/min):
"+str(round(okretaji_vratila_fr1, 3)))
    else:
        e_promjer_glodala.config(state="disabled")
        promjer_glodala=0.0
        text_promjer_glodala.set("Promjer glodala (mm):
"+str(round(promjer_glodala, 3)))
        e_okretaji_em1.config(state="disabled")
        okretaji_em1=0.0
        text_okretaji_em1.set(f"Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em1, 3)))
        e_okretaji_vratilo1.config(state="disabled")
        okretaji_vratilo1=0.0
        text_okretaji_vratilo1.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo1, 3)))
        e_koef_emv1.config(state="disabled")
        koef_emv1=0.0
        text_koef_emv1.set("Koeficijent prijenosa (i):
"+str(round(koef_emv1, 3)))
        e_max_brzina1.config(state="normal")
        max_brzina1=0.0
        text_max_brzina1.set("Maksimalna brzina rezanja pri 50Hz (m/s):
"+str(round(max_brzina1, 3)))
        brzina_fr1=0.0
        text_brzina_fr1.set("Odabrana brzina rezanja (m/s):
"+str(round(brzina_fr1, 3)))
        e_okretaji_em_fr1.config(state="disabled")
        okretaji_em_fr1=0.0
        text_okretaji_em_fr1.set("Odabrani okretaji motora (n/min):
"+str(round(okretaji_em_fr1, 3)))
        e_okretaji_vratila_fr1.config(state="disabled")
        okretaji_vratila_fr1=0.0
        text_okretaji_vratila_fr1.set("Odabrani okretaji vratila (n/min):
"+str(round(okretaji_vratila_fr1, 3)))
check_prijenos1=tk.Checkbutton(root, text="Izračun putem prijenosa",
foreground="green", variable=prijenos1, command=checkbutton_prijenos1)
checkbutton_prijenos1()

#updateovi u slučaju izmjene ikojeg entrya
def update1():
    global brzina_fr1
    global max_brzina1
    global fr_slider
```

```
    global okretaji_em_fr1
    global okretaji_vratila_fr1
    global korisnost_em1
    if prijenos1.get()==True:
        try:
            koef_emv1=float(okretaji_em1/okretaji_vratilo1)
            text_koef_emv1.set("Izračun koeficijenta prijenosa (i):
"+str(round(koef_emv1, 3)))

max_brzina1=float((promjer_glodala/1000)*math.pi*(okretaji_vratilo1/60))
            text_max_brzina1.set("Maksimalna brzina rezanja pri 50Hz (m/s):
"+str(round(max_brzina1, 3)))
            okretaji_em_fr1=float((okretaji_em1/fr_range)*fr_slider)
            text_okretaji_em_fr1.set("Odabrani okretaji motora (n/min):
"+str(round(okretaji_em_fr1, 3)))

okretaji_vratila_fr1=float((okretaji_vratilo1/fr_range)*fr_slider)
            text_okretaji_vratila_fr1.set("Odabrani okretaji vratila
(n/min): "+str(round(okretaji_vratila_fr1, 3)))
            brzina_fr1=float((max_brzina1/fr_range)*fr_slider)
            text_brzina_fr1.set("Odabrana brzina rezanja (m/s):
"+str(round(brzina_fr1, 3)))
        except:
            pass
    else:
        try:
            brzina_fr1=float((max_brzina1/fr_range)*fr_slider)
            text_brzina_fr1.set("Odabrana brzina rezanja (m/s):
"+str(round(brzina_fr1, 3)))
        except:
            pass
#karakteristika1
def karakteristika_1():
    global fr_range
    global frekvencija_graf
    global napon_range
    window=tk.Toplevel()
    app_width=1450
    app_height=725
    screen_width=root.winfo_screenwidth()
    screen_height=root.winfo_screenheight()
    x=(screen_width/2)-(app_width/2)
    y=(screen_height/2)-(app_height/2)
    window.geometry(f'{app_width}x{app_height}-{int(x)}-{int(y)}')
    ###
    brzina_rezanja_list1=[]
    slider_frekvencija_list=[]
    napon_range=10
    frekvencija_graf=0
    for frekvencija_graf in range(51):
        napon_graf=frekvencija_graf*(napon_range/fr_range)
        slider_frekvencija_list.append(napon_graf)
        brzina_rezanja=float((max_brzina1 / fr_range) * frekvencija_graf)
```

```
        brzina_rezanja_list1.append(brzina_rezanja)
    fig=plt.figure()
    canvas=FigureCanvasTkAgg(fig, master=window)
    canvas.get_tk_widget().pack(fill=tk.BOTH, expand=True)
    canvas.draw()
    toolbar=NavigationToolbar2Tk(canvas, window, pack_toolbar=False)
    toolbar.place(x=10, y=10)
    toolbar.update()
    plt.style.use('fivethirtyeight')
    ax1=plt.subplot(111)
    plt.subplots_adjust(right=0.93, top= 0.93)
    ax1.set_title("Statička karakteristika glavnog motora",
fontsize="medium")
    ax1.set_xlabel("Ulazni napon u frekventni pretvarač (V)",
fontsize="medium")
    ax1.yaxis.labelpad=+10
    ax1.set_ylabel("Brzina rezanja (m/s)", fontsize="medium")
    ax1.plot(slider_frekvencija_list, brzina_rezanja_list1, label="Brzina
1")
    ax1.legend(fontsize="small", framealpha=1, ncol=2)
button_karakteristika_em1=tk.Button(root, text="Statička karakteristika",
foreground="green", command=karakteristika_1)

### isto sve samo za posmični motor
#promjer kotaca
promjer_kotac=0.0
text_promjer_kotac=tk.StringVar()
text_promjer_kotac.set("Promjer vođača/kotača posmaka (mm):
"+str(round(promjer_kotac, 3)))
e_promjer_kotac=ttk.Entry(root, width=40, textvariable=text_promjer_kotac)
def kotac(event):
    global promjer_kotac
    if event.type=="9":
        text_promjer_kotac.set("")
    elif event.keysym=="Return":
        try:
            promjer_kotac=float(e_promjer_kotac.get())
            update2()
            root.focus_set()
        except:
            root.focus_set()
    else:
        text_promjer_kotac.set("Promjer kotača posmaka (mm):
"+str(round(promjer_kotac, 3)))
e_promjer_kotac.bind("<FocusIn>", kotac)
e_promjer_kotac.bind("<FocusOut>", kotac)
e_promjer_kotac.bind("<Return>", kotac)

#okretaji glavnog motora
okretaji_em2=0.0
text_okretaji_em2=tk.StringVar()
text_okretaji_em2.set("Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em2, 3)))
```

```
e_okretaji_em2=ttk.Entry(root, width=40, textvariable=text_okretaji_em2)
def okretaji_m2(event):
    global okretaji_em2
    if event.type=="9":
        text_okretaji_em2.set("")
    elif event.keysym=="Return":
        try:
            okretaji_em2=float(e_okretaji_em2.get())
            update2()
            root.focus_set()
        except:
            root.focus_set()
    else:
        text_okretaji_em2.set("Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em2, 3)))
e_okretaji_em2.bind("<FocusIn>", okretaji_m2)
e_okretaji_em2.bind("<Return>", okretaji_m2)
e_okretaji_em2.bind("<FocusOut>", okretaji_m2)

#
okretaji_vratilo2=0.0
lista_okretaji_vratilo2=[]
lista_max_brzina2=["2.0", "4.0", "5.5", "6.5", "11.0", "13.0", "16.55",
"33.0"]
text_okretaji_vratilo2=tk.StringVar()
text_okretaji_vratilo2.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo2, 3)))
e_okretaji_vratilo2=ttk.Entry(root, width=40,
textvariable=text_okretaji_vratilo2)
def okretaji_v2(event):
    global okretaji_vratilo2
    global koef_emv2
    if event.num==1:
        text_okretaji_vratilo2.set("")
    elif event.type=="10":
        text_okretaji_vratilo2.set("Broj okretaja vratila (n/min):
"+str(round(okretaji_vratilo2, 3)))
    else:
            try:
                okretaji_vratilo2=float(text_okretaji_vratilo2.get())
                update2()
                root.focus_set()
            except:
                root.focus_set()
e_okretaji_vratilo2.bind("<<ComboboxSelected>>", okretaji_v2)
e_okretaji_vratilo2.bind("<Return>", okretaji_v2)
e_okretaji_vratilo2.bind("<Button-1>", okretaji_v2)
e_okretaji_vratilo2.bind("<FocusOut>", okretaji_v2)

#
koef_emv2=0.0
text_koef_emv2=tk.StringVar()
text_koef_emv2.set("Koeficijent prijenosa (i): "+str(round(koef_emv2, 3)))
```

```
e_koef_emv2=ttk.Entry(root, width=40, textvariable=text_koef_emv2)

#
max_brzina2=0.0
text_max_brzina2=tk.StringVar()
text_max_brzina2.set("Maksimalna brzina posmaka pri 50Hz (m/min): " +
str(round(max_brzina2, 3)))
e_max_brzina2=ttk.Combobox(root, textvariable=text_max_brzina2, width=39,
values=lista_max_brzina2, exportselection=False)
def max_vp(event):
    global max_brzina2
    global fr_slider2
    if event.num==1:
        text_max_brzina2.set("")
    elif event.type=="10":
        text_max_brzina2.set("Maksimalna brzina posmaka pri 50Hz (m/min): "
+ str(round(max_brzina2, 3)))
    else:
        try:
            max_brzina2=float(text_max_brzina2.get())
            update2()
            root.focus_set()
        except:
            root.focus_set()
e_max_brzina2.bind("<<ComboboxSelected>>", max_vp)
e_max_brzina2.bind("<Return>", max_vp)
e_max_brzina2.bind("<Button-1>", max_vp)
e_max_brzina2.bind("<FocusOut>", max_vp)

#
fr_slider2=0.0
fr_range2=50.0
scale2=tk.Scale(root, from_=0, to=fr_range2, orient=tk.HORIZONTAL,
label="Odabrana frekvencija (Hz): " + str(fr_slider2), length=180)
scale2.set(fr_slider2)
def horizontal2(event):
    global fr_slider2
    if rucno2.get()==True:
        fr_slider2=scale2.get()
        scale2.config(label="Odabrana frekvencija (Hz): " +
str(fr_slider2))
        update2()
        pwm_2()
    else:
        scale2.set(fr_slider2)
scale2.bind("<ButtonRelease-1>", horizontal2)

#
rucno2=tk.BooleanVar()
rucno2.set(False)
def slider2_reset_checkbutton():
    global fr_slider2
    if rucno2.get()==True:
```

```
        e_brzina_fr2.config(state="readonly")
    else:
        e_brzina_fr2.config(state="normal")
rucno_check2=tk.Checkbutton(root, text="Ručno upravljanje frekvencijom",
foreground="green", variable=rucno2, command=slider2_reset_checkbutton)

#
brzina_fr2=float((max_brzina2 / fr_range2) * fr_slider2)
text_brzina_fr2=tk.StringVar()
text_brzina_fr2.set("Odabrana brzina posmaka (m/min): " +
str(round(brzina_fr2, 3)))
e_brzina_fr2=ttk.Entry(root, width=40, textvariable=text_brzina_fr2)
def vr_fr2(event):
    global brzina_fr2
    global fr_slider2
    if event.type=="9":
        text_brzina_fr2.set("")
    elif event.keysym=="Return":
        try:
            if float(e_brzina_fr2.get())<=max_brzina2:
                brzina_fr2=float(e_brzina_fr2.get())
                fr_slider2=float(brzina_fr2 / (max_brzina2 / fr_range2))
                slider_frekvencija_show=round(fr_slider2, 3)
                scale2.set(fr_slider2)
                scale2.config(label="Odabrana frekvencija (Hz): " +
str(slider_frekvencija_show))
                update2()
                pwm_2()
                root.focus_set()
            else: root.focus_set()
        except:
            root.focus_set()
    else:
        text_brzina_fr2.set("Odabrana brzina posmaka (m/min): " +
str(round(brzina_fr2, 3)))
e_brzina_fr2.bind("<FocusIn>", vr_fr2)
e_brzina_fr2.bind("<FocusOut>", vr_fr2)
e_brzina_fr2.bind("<Return>", vr_fr2)


okretaji_em_fr2=float((okretaji_em2 / fr_range2) * fr_slider2)
text_okretaji_em_fr2=tk.StringVar()
text_okretaji_em_fr2.set("Odabrani okretaji motora (n/min): " +
str(round(okretaji_em_fr2, 3)))
e_okretaji_em_fr2=ttk.Entry(root, width=40,
textvariable=text_okretaji_em_fr2)


okretaji_vratila_fr2=float((okretaji_vratilo2 / fr_range2) * fr_slider2)
text_okretaji_vratila_fr2=tk.StringVar()
text_okretaji_vratila_fr2.set("Odabrani okretaji vratila (n/min): " +
str(round(okretaji_vratila_fr2, 3)))
e_okretaji_vratila_fr2=ttk.Entry(root, width=40,
```

```
textvariable=text_okretaji_vratila_fr2)

#
prijenos2=tk.BooleanVar()
prijenos2.set(False)
def checkbutton_prijenos2():
    global okretaji_em2
    global okretaji_vratilo2
    global koef_emv2
    global okretaji_em_fr2
    global okretaji_vratila_fr2
    global max_brzina2
    global brzina_fr2
    if prijenos2.get()==True:
        e_promjer_kotac.config(state="normal")
        promjer_kotac=0.0
        text_promjer_kotac.set("Promjer vođača/kotača posmaka (mm):
"+str(round(promjer_kotac, 3)))
        e_okretaji_em2.config(state="normal")
        okretaji_em2=0.0
        text_okretaji_em2.set("Nazivni broj okretaja motora (n/min):
"+str(round(okretaji_em2, 3)))
        e_okretaji_vratilo2.config(state="normal")
        okretaji_vratilo2=0.0
        text_okretaji_vratilo2.set("Broj okretaja vratila (n/min): " +
str(round(okretaji_vratilo2, 3)))
        e_koef_emv2.config(state="readonly")
        koef_emv2=0.0
        text_koef_emv2.set("Koeficijent prijenosa (i): " +
str(round(koef_emv2, 3)))
        e_max_brzina2.config(state="readonly")
        max_brzina2=0.0
        text_max_brzina2.set("Maksimalna brzina posmaka pri 50Hz (m/min): "
+ str(round(max_brzina2, 3)))
        brzina_fr2=0.0
        text_brzina_fr2.set("Odabrana brzina posmaka (m/min): " +
str(round(brzina_fr2, 3)))
        e_okretaji_em_fr2.config(state="readonly")
        okretaji_em_fr2=0.0
        text_okretaji_em_fr2.set("Odabrani okretaji motora (n/min): " +
str(round(okretaji_em_fr2, 3)))
        e_okretaji_vratila_fr2.config(state="readonly")
        okretaji_vratila_fr2=0.0
        text_okretaji_vratila_fr2.set("Odabrani okretaji vratila (n/min): "
+ str(round(okretaji_vratila_fr2, 3)))
    else:
        e_promjer_kotac.config(state="disabled")
        promjer_kotac=0.0
        text_promjer_kotac.set("Promjer kotača posmaka (mm):
"+str(round(promjer_kotac, 3)))
        e_okretaji_em2.config(state="disabled")
        okretaji_em2=0.0
        text_okretaji_em2.set("Nazivni broj okretaja motora (n/min):
```

```
"+str(round(okretaji_em2, 3)))
        e_okretaji_vratilo2.config(state="disabled")
        okretaji_vratilo2=0.0
        text_okretaji_vratilo2.set("Broj okretaja vratila (n/min): " +
str(round(okretaji_vratilo2, 3)))
        e_koef_emv2.config(state="disabled")
        koef_emv2=0.0
        text_koef_emv2.set("Koeficijent prijenosa (i): " +
str(round(koef_emv2, 3)))
        e_max_brzina2.config(state="normal")
        max_brzina2=0.0
        text_max_brzina2.set("Maksimalna brzina posmaka pri 50Hz (m/min): "
+ str(round(max_brzina2, 3)))
        brzina_fr2=0.0
        text_brzina_fr2.set("Odabrana brzina posmaka (m/min): " +
str(round(brzina_fr2, 3)))
        e_okretaji_em_fr2.config(state="disable")
        okretaji_em_fr2=0.0
        text_okretaji_em_fr2.set("Odabrani okretaji motora (n/min): " +
str(round(okretaji_em_fr2, 3)))
        e_okretaji_vratila_fr2.config(state="disable")
        okretaji_vratila_fr2=0.0
        text_okretaji_vratila_fr2.set("Odabrani okretaji vratila (n/min): "
+ str(round(okretaji_vratila_fr2, 3)))
check_prijenos2=tk.Checkbutton(root, text="Izračun putem prijenosa",
foreground="green", variable=prijenos2, command=checkbutton_prijenos2)
checkbutton_prijenos2()

#
def update2():
    global brzina_fr2
    global max_brzina2
    global fr_slider2
    global okretaji_em_fr2
    global okretaji_vratila_fr2
    global korisnost_em2
    if prijenos2.get()==True:
        try:
            koef_emv2=float(okretaji_em2/okretaji_vratilo2)
            text_koef_emv2.set("Izračun koeficijenta prijenosa (i):
"+str(round(koef_emv2, 3)))

max_brzina2=float((promjer_kotac/1000)*math.pi*(okretaji_vratilo2))
            text_max_brzina2.set("Maksimalna brzina posmaka pri 50Hz
(m/min): "+str(round(max_brzina2, 3)))
            okretaji_em_fr2=float((okretaji_em2/fr_range2)*fr_slider)
            text_okretaji_em_fr2.set("Odabrani okretaji motora (n/min):
"+str(round(okretaji_em_fr2, 3)))

okretaji_vratila_fr2=float((okretaji_vratilo2/fr_range2)*fr_slider2)
            text_okretaji_vratila_fr2.set("Odabrani okretaji vratila
(n/min): "+str(round(okretaji_vratila_fr2, 3)))
            brzina_fr2=float((max_brzina2/fr_range2)*fr_slider2)
```

```
            text_brzina_fr2.set("Odabrana brzina posmaka (m/min):
"+str(round(brzina_fr2, 3)))
        except:
            pass
    else:
        try:
            brzina_fr2=float((max_brzina2/fr_range2)*fr_slider2)
            text_brzina_fr2.set("Odabrana brzina posmaka (m/min):
"+str(round(brzina_fr2, 3)))
        except:
            pass

#def karakteristika_2():
    global fr_range2
    global frekvencija_graf
    global napon_range
    window=tk.Toplevel()
    app_width=1450
    app_height=725
    screen_width=root.winfo_screenwidth()
    screen_height=root.winfo_screenheight()
    x=(screen_width/2)-(app_width/2)
    y=(screen_height/2)-(app_height/2)
    window.geometry(f'{app_width}x{app_height}-{int(x)}-{int(y)}')
    brzina_rezanja_list1=[]
    slider_frekvencija_list=[]
    napon_range=10
    frekvencija_graf=0
    for frekvencija_graf in range(51):
        napon_graf=frekvencija_graf*(napon_range/fr_range2)
        slider_frekvencija_list.append(napon_graf)
        brzina_rezanja=float((max_brzina2 / fr_range2) * frekvencija_graf)
        brzina_rezanja_list1.append(brzina_rezanja)
    fig=plt.figure()
    canvas=FigureCanvasTkAgg(fig, master=window)
    canvas.get_tk_widget().pack(fill=tk.BOTH, expand=True)
    canvas.draw()
    toolbar=NavigationToolbar2Tk(canvas, window, pack_toolbar=False)
    toolbar.place(x=10, y=10)
    toolbar.update()
    plt.style.use('fivethirtyeight')
    ax1=plt.subplot(111)
    plt.subplots_adjust(right=0.93, top= 0.93)
    ax1.set_title("Statička karakteristika posmičnog motora",
fontsize="medium")
    ax1.set_xlabel("Ulazni napon u frekventni pretvarač (V)",
fontsize="medium")
    ax1.yaxis.labelpad=+10
    ax1.set_ylabel("Brzina posmaka (m/min)", fontsize="medium")
    ax1.plot(slider_frekvencija_list, brzina_rezanja_list1, label="Odabrana
brzina")
    ax1.legend(fontsize="small", framealpha=1, ncol=2)
button_karakteristika_em2=tk.Button(root, text="Statička karakteristika",
```

```
foreground="green", command=karakteristika_2)

#mjesta entrya
button_info1.place(x=300, y=20)
check_prijenos1.place(x=15, y=25)
e_promjer_glodala.place(x=15, y=55)
e_okretaji_em1.place(x=15, y=85)
e_okretaji_vratilo1.place(x=15, y=115)
e_koef_emv1.place(x=15, y=145)
e_max_brzina1.place(x=15, y=175)
button_karakteristika_em1.place(x=115, y=205)
e_brzina_fr1.place(x=400, y=25)
e_okretaji_em_fr1.place(x=400, y=55)
e_okretaji_vratila_fr1.place(x=400, y=85)
scale1.place(x=500, y=115)
rucno_check.place(x=450, y=180)
check_prijenos2.place(x=785, y=25)
e_promjer_kotac.place(x=785, y=55)
e_okretaji_em2.place(x=785, y=85)
e_okretaji_vratilo2.place(x=785, y=115)
e_koef_emv2.place(x=785, y=145)
e_max_brzina2.place(x=785, y=175)
button_karakteristika_em2.place(x=885, y=205)
e_brzina_fr2.place(x=1170, y=25)
e_okretaji_em_fr2.place(x=1170, y=55)
e_okretaji_vratila_fr2.place(x=1170, y=85)
scale2.place(x=1270, y=115)
rucno_check2.place(x=1220, y=180)

#
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
FREQUENCY=500
GPIO.setup(33, GPIO.OUT)
pwm_glavni=GPIO.PWM(33, FREQUENCY)
pwm_glavni.start(0)
def pwm_1():
    global fr_range
    global fr_slider
    if stroj_state.get()==True:
        duty_cycle_glavni=(100/fr_range*fr_slider)
        pwm_glavni.ChangeDutyCycle(duty_cycle_glavni)
    else:
        duty_cycle_glavni=0
        pwm_glavni.ChangeDutyCycle(duty_cycle_glavni)

GPIO.setup(32, GPIO.OUT)
pwm_posmak=GPIO.PWM(32, FREQUENCY)
pwm_posmak.start(0)
def pwm_2():
    if stroj_state.get()==True:
        duty_cycle_posmak=(100/fr_range2*fr_slider2)
        pwm_posmak.ChangeDutyCycle(duty_cycle_posmak)
```

```
    else:
        duty_cycle_posmak=0
        pwm_posmak.ChangeDutyCycle(duty_cycle_posmak)

def analog_read():
    global volt_avg
    global snaga
    bus = smbus.SMBus(1)
    steps = 255
    i=0
    lista=[]
    while i<1:
        bus.write_byte(0x48, 0x42)
        value = bus.read_byte(0x48)
        vref=5.02
        volt=vref/steps*value
        i = i + 1
        lista.append(volt)
    volt_avg = sum(lista) / len(lista)
    volt_avg = round(volt_avg, 3)
    volt_avg = volt_avg*5
    snaga=volt_avg*100
    root.after(500, analog_read)
analog_read()

#CSV dio
vrijeme_live=0
def live_timer():
    global vrijeme_live
    vrijeme_live=vrijeme_live+0.1
    vrijeme_live=round(vrijeme_live,2)
    root.after(100, live_timer)
live_timer()

#lista podataka za live graf
live=[]
def list4csv_main():
    global volt_avg
    global snaga_show
    snaga_show=snaga/1000
    live.append((vrijeme_live, volt_avg, snaga_show))
    root.after(500, list4csv_main)
list4csv_main()

#animiranje grafa
def animate(i):
    xList, y1List, y2List=[], [], []
    for data in live:
        xList.append(data[0])
        y1List.append(data[1])
        y2List.append(data[2])
    if len(xList)>0 and len(y1List)>0 and len(y2List)>0:
        ax1.clear()
```

```
        ax1.set_title("Mjerenje", fontsize="medium")
        ax1.set_xlabel("Vrijeme (s)", fontsize="small")
        ax1.set_ylabel("")
        ax1.plot(xList, y1List, "g", label="Napon = " + str(round(y1List[-
1], 2)) +" (V)")
        ax1.plot(xList, y2List, "r", label="Snaga = " + str(round(y2List[-
1], 2))+" (kW)")
        ax1.legend(fontsize="small")
fig=plt.figure(figsize=(26, 9), dpi=75)
canvas=FigureCanvasTkAgg(fig, master=root)
canvas.get_tk_widget().place(x=0, y=270)
plt.style.use("fivethirtyeight")
plt.tight_layout()
ax1=fig.add_subplot(111)
#
ani=FuncAnimation(fig, animate, interval=500)


#
def reset_graph():
    global live
    global vrijeme_live
    vrijeme_live=0
    live.clear()
    ani.frame_seq=ani.new_frame_seq()
    root.after(60000, reset_graph)
reset_graph()

# polje za upis naziva projekta i gumb
ime_projekta="Projekt 1"
ime_projekta_text=tk.StringVar()
ime_projekta_text.set("Naziv projekta: "+str(ime_projekta))
entry_naziv=ttk.Entry(root, width=38, textvariable=ime_projekta_text)
entry_naziv.place(x=1550, y=25)
def projekt(event):
    global ime_projekta
    if event.type=="9":
        ime_projekta_text.set("")
    elif event.keysym=="Return":
        try:
            ime_projekta=str(entry_naziv.get())
            root.focus_set()
        except:
            root.focus_set()
    else:
        ime_projekta_text.set("Naziv projekta: "+str(ime_projekta))
entry_naziv.bind("<FocusIn>", projekt)
entry_naziv.bind("<FocusOut>", projekt)
entry_naziv.bind("<Return>", projekt)

# CSV entry vrijeme snimanja
upis_vrijeme=60.0
upis_vrijeme_text=tk.StringVar()
```

```
upis_vrijeme_text.set("Vrijeme snimanja za CSV file (s):
"+str(upis_vrijeme))
mjerac_vremena=tk.BooleanVar()
mjerac_vremena.set(False)
def upis_csv(event):
    global upis_vrijeme
    if event.type=="9":
        upis_vrijeme_text.set("Vrijeme snimanja za CSV file (s):
"+str(upis_vrijeme))
    elif event.keysym=="Return":
        try:
            upis_vrijeme=float(entry_upis_csv.get())
            root.focus_set()
        except:
            root.focus_set()
    else:
        upis_vrijeme_text.set("Vrijeme snimanja za CSV file (s):
"+str(upis_vrijeme))

#
def entry_csv():
    global entry_upis_csv
    if mjerac_vremena.get()==True:
        entry_upis_csv=ttk.Entry(root, width=35,
textvariable=upis_vrijeme_text)
        upis_vrijeme_text.set("Vrijeme snimanja za CSV file (s):
"+str(upis_vrijeme))
        entry_upis_csv.place(x=1550, y=85)
        entry_upis_csv.bind("<FocusIn>", upis_csv)
        entry_upis_csv.bind("<FocusOut>", upis_csv)
        entry_upis_csv.bind("<Return>", upis_csv)
    else:
        entry_upis_csv.destroy()
button_mjerac_vremena=tk.Checkbutton(root, text="Mjerač vremena",
foreground="green", variable=mjerac_vremena, command=entry_csv)
button_mjerac_vremena.place(x=1550, y=55)

# csv_snimanje
vrijeme_csv=0
def timer_csv():
    global vrijeme_csv
    if snimanje_state.get()==True:
        vrijeme_csv=vrijeme_csv+0.1
        vrijeme_csv=round(vrijeme_csv,3)
        root.after(100, timer_csv)
    else:
        vrijeme_csv=0

#csv switch gumb
snimanje_state=tk.BooleanVar()
snimanje_state.set(False)
def switch_csv():
    global snimanje_state
```

```
    global vrijeme_live
    if snimanje_state.get()==True:
        button_switch_csv.config(text="Zaustavi snimanje",
foreground="red")
        snimanje_csv()
        timer_csv()
    else:
        button_switch_csv.config(text="Pokreni snimanje",
foreground="green")
button_switch_csv=tk.Checkbutton(root, indicatoron=False, text="Pokreni
snimanje", foreground="green", variable=snimanje_state, command=switch_csv)
button_switch_csv.place(x=1550, y=115)

#
csv_list=[]
def snimanje_csv():
    global fr_slider
    global fr_slider2
    global volt_avg
    global snaga
    if snimanje_state.get()==True and mjerac_vremena.get()==False:
        csv_list.append((vrijeme_csv, fr_slider, fr_slider2, volt_avg,
snaga))
        root.after(100, snimanje_csv)
    elif snimanje_state.get()==True and mjerac_vremena.get()==True:
        if vrijeme_csv<upis_vrijeme:
            csv_list.append((vrijeme_csv, fr_slider, fr_slider2, volt_avg,
snaga))
            root.after(100, snimanje_csv)
        else:
            snimanje_state.set(False)
    else:
        switch_csv()
        file_path =
filedialog.asksaveasfilename(initialfile=str(ime_projekta),
defaultextension=".csv", filetypes=[("CSV file", ".csv")])
        headers=["Vrijeme", "Frekvencija glavnog motora", "Frekvencija
posmicnog motora", "Struja", "Snaga rezanja"]
        with open(file_path, "a", newline='') as file:
            writer = csv.writer(file, delimiter="\t")
            writer.writerow(headers)
            writer.writerows(csv_list)
        csv_list.clear()

#paljenje/gašenje stroja sa gumbom i prikazom
switch_stroj_on = Image.open("on.png")
switch_stroj_on=ImageTk.PhotoImage(switch_stroj_on.resize((50, 50)))
switch_stroj_off = Image.open("off.png")
switch_stroj_off=ImageTk.PhotoImage(switch_stroj_off.resize((50, 50)))
label_switch_stroj=ttk.Label(root, text="Stroj je ugašen",
foreground="gray")

#
```

```
stroj_state=tk.BooleanVar()
stroj_state.set(False)
def switch_stroj():
    if stroj_state.get()==True:
        button_switch_stroj.config(image=switch_stroj_on)
        label_switch_stroj.config(text="Stroj je upaljen",
foreground="green")
        pwm_1()
        pwm_2()
    else:
        button_switch_stroj.config(image=switch_stroj_off)
        label_switch_stroj.config(text="Stroj je ugašen",
foreground="gray")
        pwm_1()
        pwm_2()
button_switch_stroj=tk.Checkbutton(root, indicatoron=False,
image=switch_stroj_off, borderwidth=0, foreground="gray",
variable=stroj_state, command=switch_stroj)
label_switch_stroj.place(x=1675, y=215)
button_switch_stroj.place(x=1700, y=155)

#zatvaranje prozora
def closing():
    GPIO.cleanup()
    root.destroy()
root.protocol("WM_DELETE_WINDOW", closing)

#pokretanje petlje
root.mainloop()
```