Realizacija sustava pravljanja primjenom OpenPLC platforme

Frančić, Filip

Undergraduate thesis / Završni rad

2017

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

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:047073

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-07-15**

Repository / Repozitorij:

University North Digital Repository

zir.nsk.hr

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

Sveučilište
Sjever

Završni rad br. 412/EL/2017

Realizacija sustava upravljanja primjenom OpenPLC platforme

Filip Frančić, 0020/336

Sveučilište
Sjever

Odjel za elektrotehniku

Završni rad br. 412/EL/2017

Realizacija sustava upravljanja primjenom OpenPLC platforme

Student

Filip Frančić, 0020/336

Mentor

Josip Srpak, dipl.ing.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za elektrotehniku

PRISTUPNIK: Filip Frančić | MATIČNI BROJ: 0020/336

DATUM: 12.09.2017. | KOLEGIJ: PLC sustavi upravljanja

NASLOV RADA: Realizacija sustava upravljanja primjenom OpenPLC platforme

NASLOV RADA NA ENGL. JEZIKU: Realization of control system using the OpenPLC platform

MENTOR: Josip Srpak | ZVANJE: predavač

ČLANOVI POVJERENSTVA:

1. mr.sc.Ivan Šumiga, viši predavač
2. Dunja Srpak, dipl.ing., predavač
3. Josip Srpak, dipl.ing., predavač
4. Miroslav Horvatić, dipl.ing., predavač - rezervni član
5.

Zadatak završnog rada

BROJ: 412/EL/2017

OPIS

OpenPLC je koncept funkcionalnog PLC-a otvorenog koda, prilagođen različitim razvojnim platformama od Windowsa i Linuxa, preko Raspberry PI do Arduina i ESP8266. Programski kod unosi se putem PLCopen Editora upotrebom standardnih PLC jezika što ga čini kompatibilnim postojećim programskim rješenjima, te otvara mogućnost povoljnih alternativnih projektnih rješenja nasuprot skupim komercijalno dostupnim proizvodima. U završnom radu potrebno je razviti sustav upravljanja primjenom OpenPLC koncepta pokrenutog na Raspberry Pi platformi. Ulazne i izlaze portove razviti kao izvršne module koristeći arduino platformu, a kojima će se upravljati putem Modbus RTU komunikacije. Dodatno za demonstraciju kompatibilnosti konfigurirati opisani sustav kao Modbus TCP/IP periferiju te ga putem VPN-a vizualizirati u Siemensovoj SCADA aplikaciji (WinCC).

U radu je potrebno:
- opisati osnovne elemente i sklopove od kojih se sustav sastoji,
- opisati problematiku razvoja i implementacije sustava do funkcionalnosti,
- objasniti programska i komunikacijska rješenja,
- izraditi funkcionalni prototip i potrebno okruženje za prikaz rada sustava,
- izvršiti testiranje sustava,
- analizirati i komentirati rezultate ispitivanja.

ZADATAK URUČEN: 18.09.2017. | POTPIS MENTORA: Josip Srpak

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

Zahvala

Zahvaljujem se svom mentoru dipl.ing. Josipu Srpaku na pruženoj potpori i usmjeravanju kod izrade završnog rada., te obitelji i prijateljima na potpori kroz obrazovanje.

Također zahvaljujem se svim djelatnicima Sveučilišta Sjever na pruženoj edukaciji koja je omogućila izradu ovog rada.

Posebna zahvala g. Stjepanu Vrpoljcu iz ITD-tima za izradu tiskane pločice.

Sažetak

U završnom radu opisuje se proces kreiranja PLC uređaja na Raspberry Pi uređaju pomoću OpenPLC platforme, te njegovo povezivanje sa SCADA sustavom upravljanja preko VPN-a. Također opisuje se izrada tiskane pločice koja služi kao ulazno – izlazni modul za OpenPLC kojom je moguće upravljati pomoću modbus protokola. U radu su opisani dijelovi sustava, te način njihovog korištenja.

Popis korištenih kratica

Mbit/s – Mega bitova po sekundi
PLC – eng. Programmable Logical Controller
ADC – eng. Analogno – digitalni converter
SCADA – eng. Supervisory Control and Data Acquisition
VPN - eng. Virtual Private Network
RTU – eng. Remote Terminal Unit
TCP – eng. Transmission Control Protocol
kB – kilobajt
PWM – eng. Pulse Width Modulation
MHz – Mega Hertz
V – Volt
A - Amper
bps – Bitova po sekundi
ms – milisekunda
LD – eng. Ladder
FBD – eng. Function Block Diagram
ST – eng. Structured Text
SFC – eng. Sequential Function Chart
IL – eng. Instruction List
UK – eng. United Kingdom
GPIO - eng. General Purpose Input/Output
DC – eng. Direct Current

Sadržaj

1. Uvod 1
2. Rad sustava 2
3. Elementi sustava 4
   3.1. Raspberry pi 4
   3.2. RS485 6
   3.3. PLC 6
   3.4. I2C 7
   3.5. SCADA 8
      3.5.1. WinCC 8
   3.6. Modbus 8
      3.6.1. Modbus RTU 9
      3.6.2. Modbus TCP 9
      3.6.3. Podaci u modbus protokolu 10
      3.6.4. Funkcijski kodovi 10
   3.7. VPN 11
   3.8. OpenPLC 12
4. Implementacija sustava 13
   4.1. VPN server 13
      4.1.1. Implementacija VPN-a u sustav 13
      4.1.2. Postavljanje VPN servera na Raspberry pi 14
      4.1.3. Spajanje na VPN poslužitelj 18
   4.2. PLC upravljanje 19
      4.2.1. OpenPLC software 19
      4.2.2. PLCopen editor 23
5. Hardversko rješenje 27
   5.1. Atmega328P 27
   5.2. Način programiranja 27
   5.3. Implementacija ATMega 328p mikrokontrolera u OpenPLC sustav 29
   5.4. El. Shema i tiskana pločica 30
      5.4.1. Dijelovi sheme 30
      5.4.2. Dizajn pločice [Prilog 3] 34
6. Programsko rješenje za Atmega 328P 35
   6.1. Blok dijagram 36
7. Programsko rješenje za OpenPLC 37
8. Kučište 38
9. Upravljanje pomoću SCADA sustava 40
   9.1. Latencija sustava 40
10. Zaključak 42
11. Literatura 43

Popis slika 44

Prilozi 45

1. Uvod

Proces automatskog upravljanja danas se koristi skoro u svim elementima postrojenja, ali i svakodnevnom životu. Cilj automatizacije je omogućiti lagan i jednostavan način upravljanja električnim i mehaničkim uređajima. Korištenje računala u svrhu automatskog upravljanja omogućilo je pristup upravljivim uređajima na jednostavan način. U industriji, najčešće se koriste PLC uređaji koji imaju status robusnih i izdržljivih uređaja kako bi upravljanje sustavom bilo pouzdano. Takvi uređaji iako su pogodni za upravljanje velikim sustavima, često zbog svoje velike cijene nisu isplativi u manjim sustavima. Zbog tog se razvijaju sustavi na raznim platformama koje su cijenom prihvatljive i razumljive. Jedna od takvih platforma je OpenPLC, koja je u mogućnosti koristiti brojne vrste računala za ostvarivanje upravljanja i time eliminira upotrebu industrijskih PLC uređaja. Pouzdanost takvog sustava se sa kvalitetnim elementima, te pažljivo pisanim programskim kodom može dovesti do industrijskih standarda. Mogućnost daljinskog upravljanja, te komunikacije po postojećim i vrlo raširenim protokolima čine sustav primjenjivim u svim okolnostima. Koristeći otvorene i besplatne protokole, te mogućnost prilagođenja sustava potrebama postrojenja čini OpenPLC sustav jednostavno i jeftino rješenje za automatizaciju postrojenja.

2. Rad sustava

Cijeli sustav ovog projekta zamišljen je da služi kao industrijski PLC sa mogućnošću daljinske komunikacije i upravljanja.

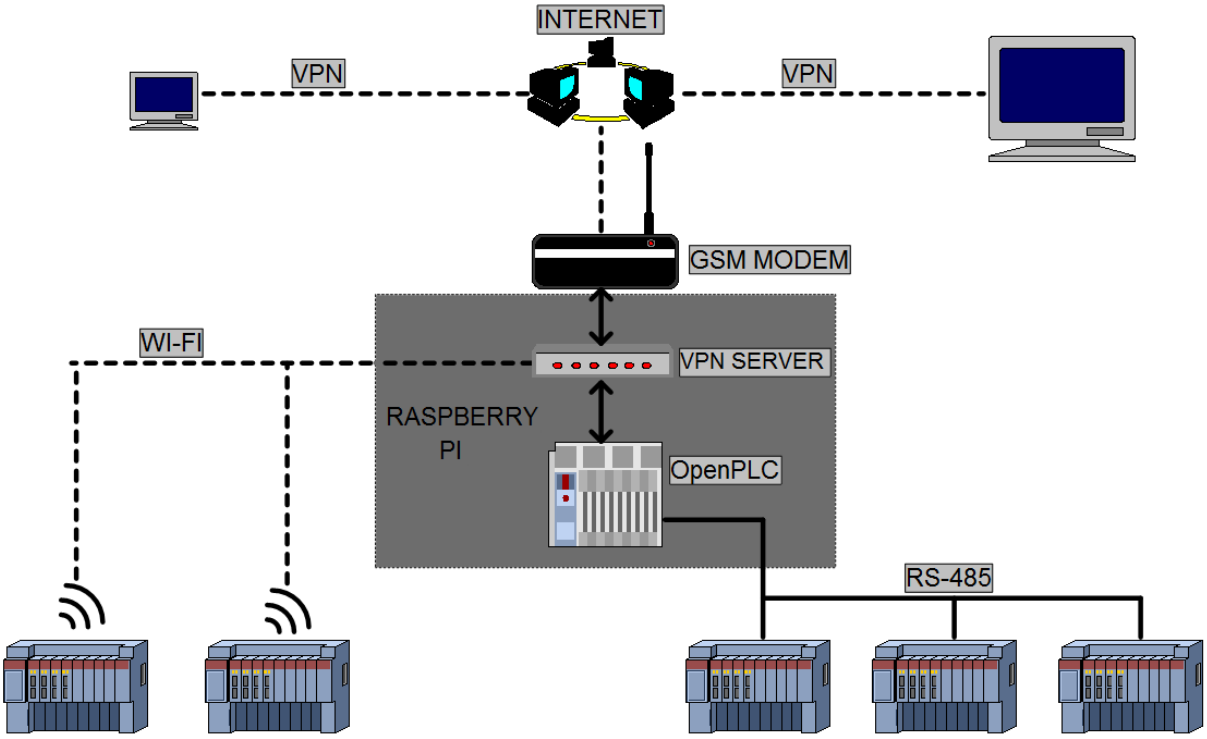
Raspberry pi služi kao centralna jedinica u sustavu koja prikuplja podatke i komunicira s vanjskim uređajima. Pomoću OpenPLC aplikacije raspberry pi se ponaša kao PLC uređaj, te ga je moguće programirati za svoje potrebe. Pri komunikaciji koristi se RTU i TCP verzija modbus protoka preko RS-485 serijske komunikacije, te ethernet veze. Također na Raspberry pi-u implementiran je i VPN server koji služi za sigurnu daljinsku komunikaciju sa cijelim sustavom preko mreže. U bilo kojem trenutku moguće se spojiti na VPN server te pristupiti cijeloj lokalnoj mreži nekog postrojenja koje može sadržavati i više Raspberry PLC uređaja. Moguće je daljinski mijenjati parametre programa, upravljati postrojenjem, te vidjeti stanje cijelog sustava.

OpenPLC može upravljati bilo kojim uređajem koji podržava modbus protokol. U ovom projektu izabran je ATMega 328P procesor koji se najčešće nalazi na arduino platformama. Izabran je zbog male cijene te velike pouzdanosti u izvršavanju programskog koda. Broj ulazno - izlaznih portova na procesoru se može mjeriti sa današnjim PLC uređajima i time služi kao savršen uređaj za upravljanje.

Za sam procesor izrađena je konceptna ulazno izlazna pločica, kako bi rad te komunikacija s procesorom bila jednostavna i spremna za upotrebu u industriji. Ulazni konektori su galvanski odvojeni od procesora kako bi se zaštitio od neželjenih kvarova, dok su izlazni portovi realizirani pomoću izlaznih tranzistora.

Procesor komunicira sa centralnom jedinicom preko modbus RTU protokola koristeći RS-485 serijsku komunikaciju.

Slika 2.1 prikazuje princip funkcioniranja cijelog sustava



Slika 2.1 Model sustava

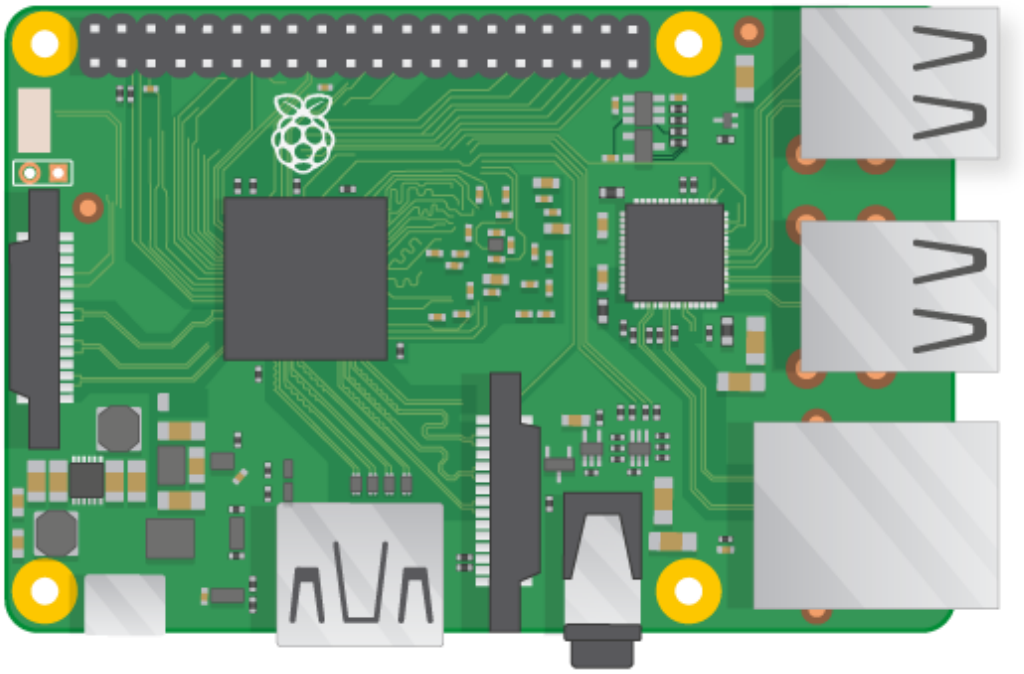
3. Elementi sustava

3.1. Raspberry pi

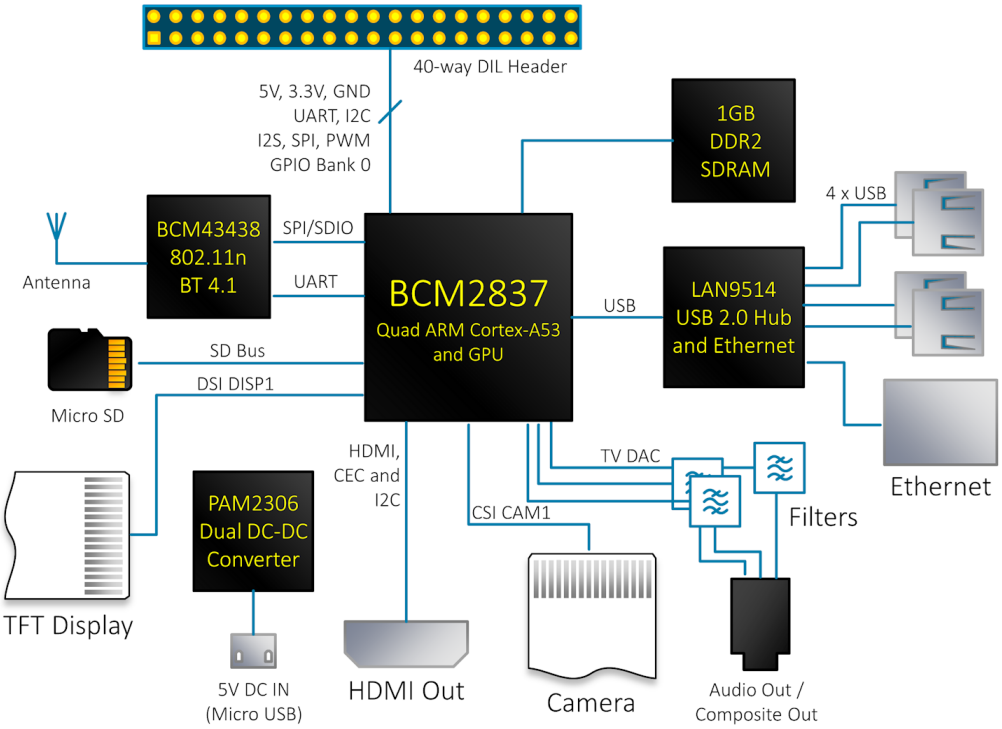
Raspberry pi je serija malih računala razvijena u svrhu edukacije računalnih vještina i programiranja. Razvijen je u UK od strane Raspberry pi Fundation.

Raspberry pi je postao vrlo popularan kod raznih projekata zbog svojih velikih mogućnosti, brzine, programske podrške i vrlo malih dimenzija.

Raspberry koristi linux operativni sustav, te njegove brojne inačice od kojih su najpopularniji Raspbian, Ubuntu i OSMC. Pločica najnovije verzije Rasperry pi 3 [slika 3.1] sadrži Broadcom-ov čip BCM2837 kojeg pokreće 64 bitni četvero jezgreni procesor, te 1GB radne memorije. Podrškom za USB, Ethernet, WI-FI, HDMI, te zvučnom karticom raspberry pi može se koristiti kao stolno računalo. Sama pločica sadrži i 40 pomoćnih pinova (GPIO) za spajanje raznih uređaja i senzora. [1]



Slika 3.1 Raspberry pi 3 model B [2]



Slika 3.2 Raspberry pi 3 blok dijagram [3]

3.2. RS485

RS-485 je električni standard u serijskoj komunikaciji. Koristi dva vodiča za prijenos podataka (upletena parica) kod half-duplex rada.

RS-485 se u velikoj mjeri koristi u industrijskim mrežama zbog male cijene, visoke pouzdanosti i otpornosti na električne smetnje u postrojenju.

Brzine prijenosa podataka dostižu do 10Mbit/s, sa mogućnošću duljine voda do 1200 metara.

Princip rada se zasniva na razlici polariteta između A i B žica parice čime se interpretiraju stanja logičke jedinice i nule.

Komunikacija između 2 ili više uređaja u jednom trenutku može se odvijati samo u jednom smjeru, te se zbog tog dovodi treći signal koji postavlja RS-485 sklop za čitanje ili slanje podataka. [4]

3.3. PLC

PLC (Programmable logic controller) je industrijsko računalo koje služi za upravljanje automatiziranim postrojenjem. PLC čita ulazne varijable, te prema određenom programu postavlja stanja izlaza. Ima mogućnost kontinuiranog nadzora i upravljanja varijablama putem SCADA sustava.

PLC uređaji su izrađeni po industrijskim standardima za rad u uvjetima kakve nalazimo u postrojenju i time imaju status robusnih i pouzdanih uređaja.

Zamijenili su relejnu logiku, i time povećali funkcionalnost cijelog sustava upravljanja. Zbog svog jednostavnog dizajna uštedjeli su na prostoru, te donijeli nove logičke funkcije koje je prije bilo teško ostvariti.

Programiranje se vrši u jednom od 5 standardnih industrijskih jezika: LD,FBD,SFC,IL,ST. [5] [6]

3.4. I2C

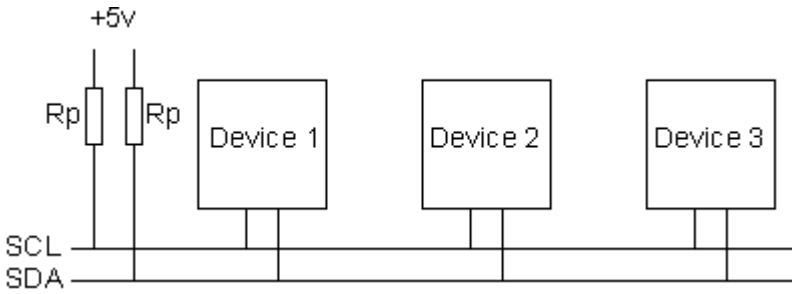
I2C je serijski protokol osmišljen 80-tih godina 20. stoljeća za jednostavno povezivanje i komunikaciju elektroničkih sklopova i uređaja. Brzina prijenosa podataka doseže do 3.4 Mbit/s što je i više nego dovoljno za kvalitetnu komunikaciju.

Protokol se koristi zbog svoje jednostavnosti, te se prijenos podataka odvija preko dva voda:

- **SDA- podatkovni vod,**
- **SCL- clock signal**

Komunikacija se odvija na principu master – slave, gdje svaki slave uređaj ima svoju adresu, te je time omogućeno spajanje više uređaja na istu liniju. [7]

U ovom projektu I2C komunikacija se koristi za povezivanje displeja sa mikrokontrolerom za ispis podataka.



Slika 3.3 Primjer I2C komunikacije [7]

3.5. SCADA

SCADA (Supervisory Control and Data Acquisition) je računalni sustav koji služi za prikupljanje podataka, upravljanje, te prikaz rada postrojenja. Najčešće se koristi sa PLC uređajima gdje je moguće direktno nadgledati i upravljati varijablama PLC-a. SCADA najčešće grafički prikazuje stanje postrojenja, gdje se grafički prikaz može prilagoditi svakom stroju i postrojenju po volji i time pruža jasnu sliku o stanju postrojenja.
WinCC SCADA je u mogućnosti komunicirati sa uređajima pomoću modbus protokola, te je iz tog razloga izabrana za ovaj projekt. [8]

3.5.1. WinCC

WinCC je sustav razvijen od strane Siemensa, te je jedan od najčešće korištenih SCADA sustava.
Mogućnosti WinCC-a se šire na sve aspekte upravljanja i nadgledanja postrojenja, od jednostavnog upravljanja i alarma, do složenih grafičkih prikaza. [9]

3.6. Modbus

Modbus je serijski komunikacijski protokol najčešće korišten u industrijskim postrojenjima za komunikaciju PLC uređaja.
Modbus je otvoren i besplatan protokol kojeg svi mogu koristiti bez plaćanja licenca.
Zbog široke primjene i podrške za SCADA sustave danas je jedan od najpopularnijih protokola za prijenos podataka u industriji.
Koristi se za prijenos podataka između PLC uređaja, raznih senzora i industrijskih računala.
Komunikacija se odvija pomoću serije binarnih podataka između glavnog (master) i sporednog uređaja (slave). Podaci se šalju preko serijske veze gdje se jedinice prikazuju kao negativni, a nule kao pozitivni potencijal (Modbus RTU).
Na svaki master uređaj moguće je spojiti do 247 slave uređaja. Takva komunikacija je moguća jer svaki slave uređaj ima svoju adresu. Master uređaj šalje podatke na sve slave uređaje, ali samo određeni uređaj sa odgovarajućom adresom prima poruku, dok je ostali ignoriraju.
Adresa slave uređaja određena je u prvom bajtu poslane poruke. [10]

3.6.1. Modbus RTU

Modbus RTU je serijski protokol i danas najčešće korišten u industriji. Podržava skoro sve vrste serijske komunikacije ali najčešće se koriste RS-232, RS-422 i RS-485.
Prednost takve komunikacije preko RS-485 standarda koji se i koristi u ovom projektu je mogućnost spajanja do 247 uređaja preko jedne parice na daljinama do 1.2 kilometra.
Modbus RTU poruka sastoji se od početnog bajta adrese slave uređaja, funkcijskog koda, podataka, te CRC bajtova koji služe za provjeru ispravnosti primljene poruke. [slika 3.104]
Na svakom početku i kraju poruke određeno je vrijeme kad nema komunikacije. []

START	ADDRESS	FUNCTION	DATA	CRC CHECK	END
T1–T2–T3–T4*	8 BITS	8 BITS	*n* x 8 BITS	16 BITS	T1–T2–T3–T4*

*For T1–T2–T3–T4, 3.5 character times at no communication.

Modbus Frame Structure-RTU Mode

Slika 3.4 Sastav modbus poruke [11]

3.6.2. Modbus TCP

Modbus TCP je vrsta modbus protokola koja je enkapsulirana u TCP paket za prijenos mrežom. Time se omogućava prijenos podataka LAN mrežom, ali i internetom upotrebom sigurnosnih prijenosnih protokola poput VPN-a.
Prednost takve komunikacije je mogućnost povezivanja i upravljanja postrojenjem sa udaljenih računala te mobilnih telefona. Time u svakom trenutku ovlaštena osoba može provjeriti stanje postrojenja aplikacijama poput SCADA sustava bez obzira gdje se nalazi. [10]

3.6.3. Podaci u modbus protokolu

Kod OpenPLC-a, podaci u modbus komunikaciji drže se u 4 tablice, gdje svaka tablica sadrži određene podatke o stanjima pojedinih registara [Slika 3.5].
Dvije tablice sadrže podatke o diskretnim veličinama, dok dvije sadrže stanja analognih veličina.
Svaka tablica sadrži 9999 polja u koja se spremaju podaci.
Veličina diskretnog polja u tablici je jedan bit, dok se analogne veličine spremaju u registre kao 16 bitne vrijednosti. [10]

POLJE/REGISTAR	TIP POLJA	TABLICA
1-9999	ČITAJ-PIŠI	DISKRETNI IZLAZ
10001-19999	ČITAJ	DISKRETNI ULAZ
30001-39999	ČITAJ	ANALOGNI ULAZ
40001-49999	ČITAJ-PIŠI	ANALOGNI IZLAZ

Slika 3.5 Modbus tablica

3.6.4. Funkcijski kodovi

Kako se modbus komunikacija bazira na principu zahtjev - odgovor primatelj mora znati koju informaciju poslati pošiljatelju. Tu nastupa funkcijski kod koji zauzima drugi bajt u modbus poruci. Ovisno o kodu, uređaj će poslati stanja određenih registara iz modbus tablice.
Popis funkcijskih kodova [Slika 3.6] [10]

Funkcijski kod	FUNKCIJA	Tablica
01	ČITAJ	DISKRETNI IZLAZ
05	PIŠI JEDNU	DISKRETNI IZLAZ
15	PIŠI VIŠE	DISKRETNI IZLAZ
02	ČITAJ	DISKRETNI ULAZ
04	ČITAJ	ANALOGNI ULAZ
03	ČITAJ	ANALOGNI IZLAZ
06	PIŠI JEDNU	ANALOGNI IZLAZ
16	PIŠI VIŠE	ANALOGNI IZLAZ

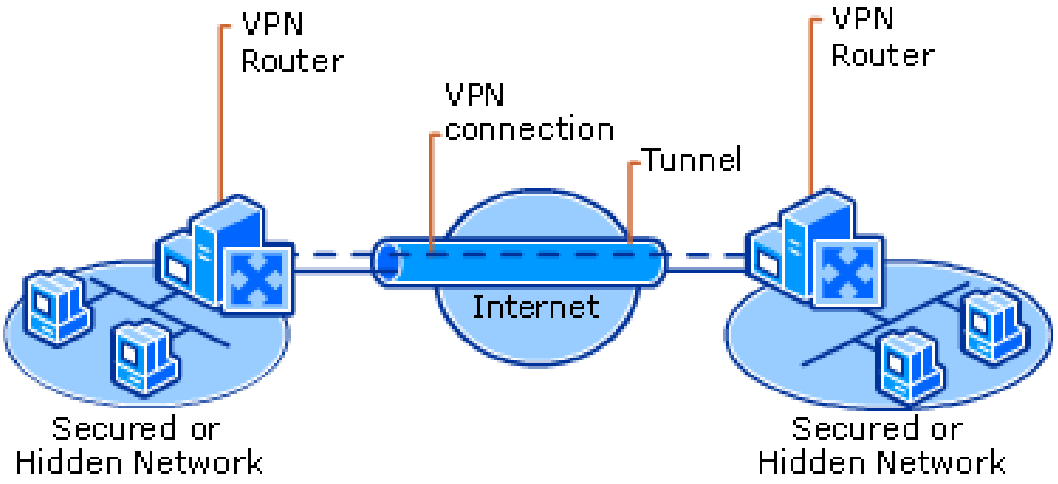
Slika 3.6 Funkcijski kodovi

3.7. VPN

VPN (Virtual Private Network) je mreža koja omogućava sigurno spajanje dvoje ili više privatnih mreža preko interneta. VPN krajnjem korisniku daje dojam da je direktno spojen na privatnu mrežu nekog servera dok u stvarnosti mogu biti udaljeni tisućama kilometara.

Tim principom korisik može sigurno pristupiti udaljenom serveru bez straha od prisluškivanja ili krađe podataka, te zbog tog mnoge tvrtke sa poslovnicama po cijelom svijetu koriste VPN tehnologiju za sigurnu komunikaciju [slika 3.7].

VPN koristi sigurnosne protokole poput tuneliranja ili point to point vezu za postizanje sigurne komunikacije. [12]



Slika 3.7 VPN sustav [12]

3.8. OpenPLC

OpenPLC je projekt otvorenog koda koji služi kao industrijski PLC uređaj.

U mogućnosti je upravljati svim vrstama postrojenja, od jednostavnih senzora i stanica, do kompleksnih strojeva i postrojenja.

Projekt je u stalnom razvitku, te već postoji podrška za mnoge platforme na kojima OpenPLC kod može raditi.

Prvenstveno projekt je dizajniran za rad s ATMega 2560 procesorom, ali s vremenom izveden je i za platforme poput Raspberry pi-a, Arduina, ESP8266 procesora, te može raditi na Windows i Linux operativnim sustavima.

Sam projekt je pisan u C programskom jeziku, te ima mnoge mogućnosti nadogradnje za nove platforme i protokole.

Podrškom za platforme poput Raspberry pi-a otvorena je mogućnost povezivanja i razmjene podataka preko mrežnih i komunikacijskih protokola. Protokol korišten za komunikaciju je modbus TCP i RTU, što omogućava povezivanje OpenPLC-a sa industrijskim uređajima poput PLC-a, te SCADA sustavima nadzora. OpenPLC služi kao modbus master uređaj na kojeg se preko modbus RTU i TCP protokola mogu spojiti ulazno - izlazne jedinice, te mnogi senzori i industrijski kontroleri. OpenPLC izveden na Raspberry pi platformi se istovremeno može ponašati kao master, ali i slave uređaj što omogućava povezivanje sa drugim PLC uređajima i SCADA sustavima.

Budući da je sam programski kod otvorenog tipa, i time besplatan, implementacija u sustav vrlo je jeftina i jednostavna za realizirati.

Najveća pogodnost kod implementacije OpenPLC-a u sustav je mogućnost dizajniranja ulazno izlaznih uređaja po volji, gdje se umjesto kupovanja skupih PLC uređaja koji možda neće iskoristiti svoj puni potencijal, može izraditi puno jednostavnije i jeftinije rješenje.

Izgradnjom kvalitetnih hardverskih ulazno - izlaznih jedinica kojima OpenPLC upravlja moguće je dovesti sustav na standard kvalitete i pouzdanosti današnjih PLC uređaja. [13]

4. Implementacija sustava

4.1. VPN server

4.1.1. Implementacija VPN-a u sustav

Kod ovog projekta VPN služi za siguran pristup PLC serveru, gdje osoba sa odgovarajućim sigurnosnim ključem te lozinkom može pristupiti i mijenjati parametre sustava preko interneta pomoću računala ili mobilnog uređaja. U današnje vrijeme pristup internetu je vrlo lako ostvariti pomoću mobilnih mreža, te sa time i spajanje nekog postrojenja na internet nije problem.
Najveći problem je sigurna komunikacija, te strah od krađe podataka i hakerskih napada, te tu nastupa VPN. Posebno je pogodan kod udaljenih postrojenja koje nemaju mogućnost stalnog ljudskog nadzora. Takvim postrojenjima omogućava se udaljeno upravljanje te siguran prijenos podataka.
Kao VPN server koristi se Raspberry pi računalo koje je spojeno na bežični mobilni usmjeritelj preko ethernet veze.
Omogućava se unaprijed određenim broju klijenata pristup računalu, te lokalnoj mreži na koju je računalo spojeno.

Za jednostavnu konfiguraciju VPN servera koristi se gotova skripta PiVPN projekta koju je moguće pronaći na stranicama PiVPN-a. [14]

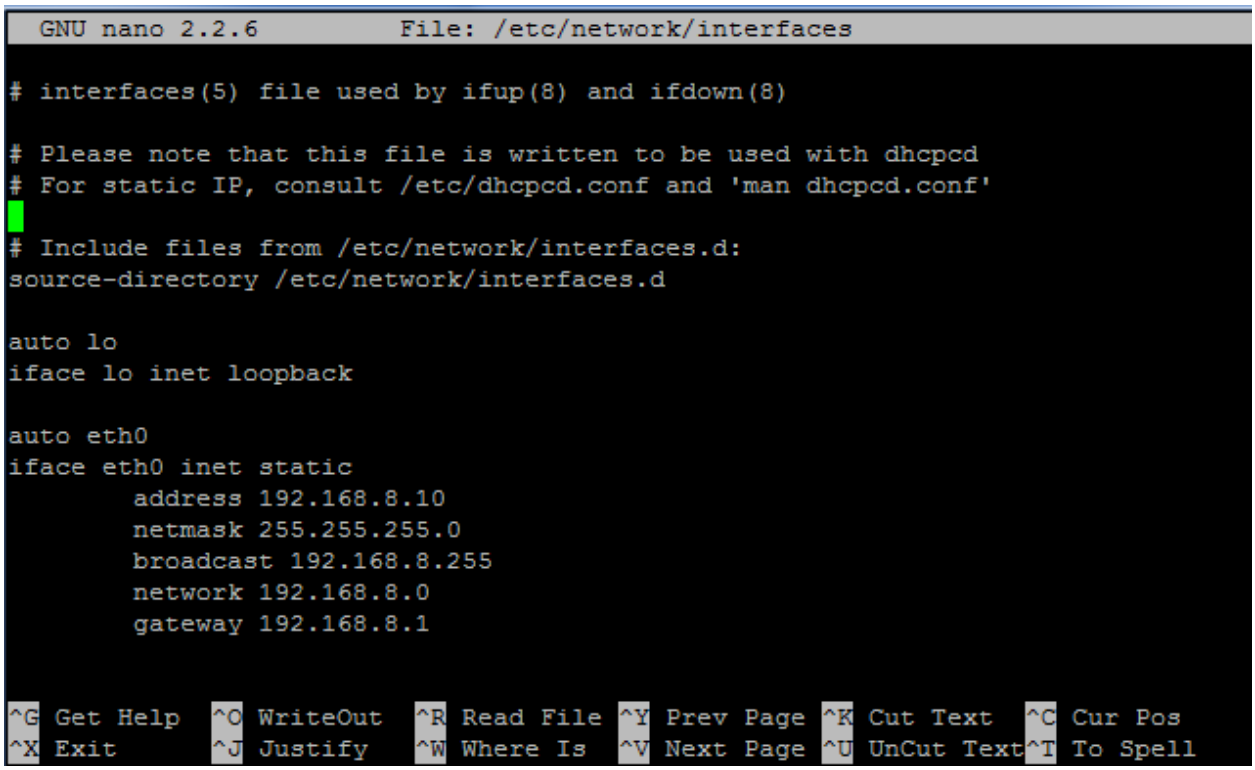
4.1.2. Postavljanje VPN servera na Raspberry pi

Prva stvar kod postavljanja VPN sustava na Raspberry pi računalu je dodjeljivanje statičke IP adrese. Sam VPN server ima određenu IP adresu i port preko kojeg je poslužiteljima dozvoljeno spajanje, te kod spajanja Raspberry pi računala na usmjeritelj moramo osigurati da se Raspberry-u dodijeli željena IP adresa.
Za postavljanje statičke IP adrese moramo urediti konfiguracijsku datoteku u Raspbian sustavu [slika 4.1]

Datoteci pristupamo sa nano editorom koristeći naredbu:

sudo nano /etc/network/interfaces

Postavlja se željena konfiguracija mreže (IP adresa, maska, broadcast adresa te adresa mreže). Ta konfiguracija će se koristiti kod spajanja Raspberry računala sa usmjeriteljem i time se osigurava dodjeljivanje određene IP adrese kod spajanja na lokalnu mrežu.



Slika 4.1 Konfiguracija statičke IP drese

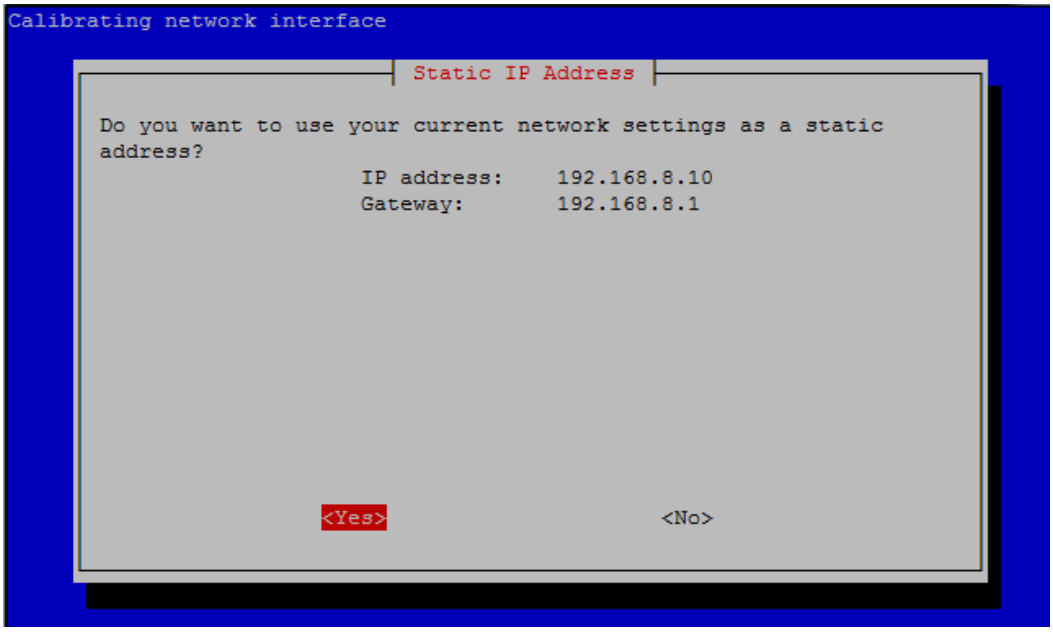
Za dohvat skripte za instalaciju PiVPN servera u komandnom prozoru upišemo naredbu [14]:

```
curl -L https://install.pivpn.io | bash
```

Za par sekundi pokreće se skripta gdje treba postaviti parametre VPN servera.

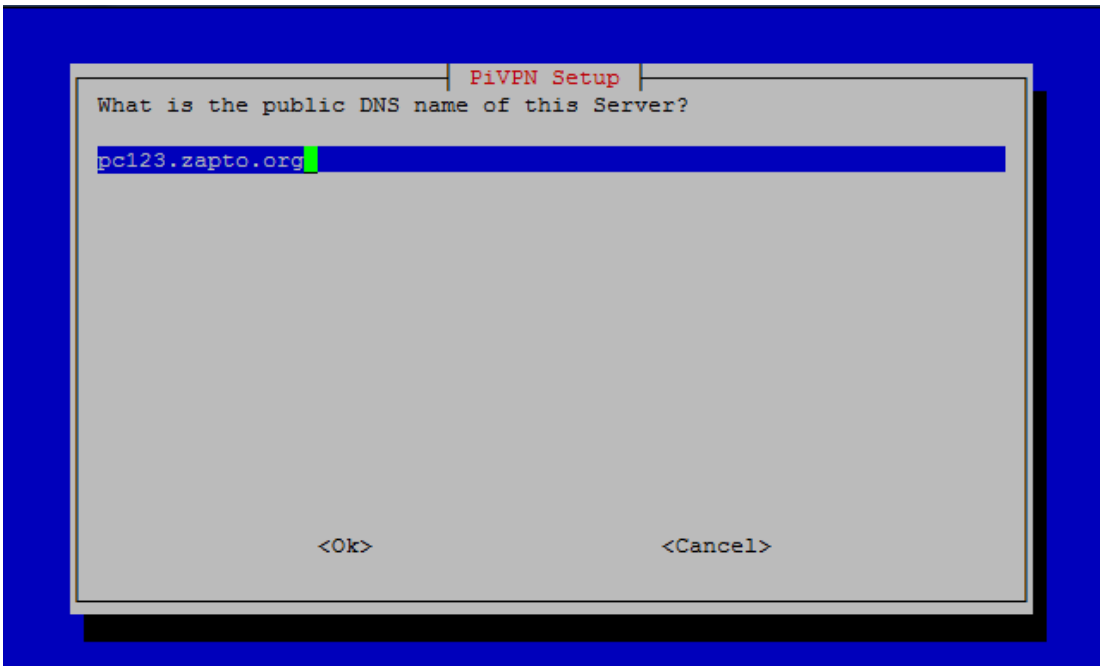
Za početak postavlja se IP adresa preko koje će se server imati pristup internetu, odabire se statička IP adresa koja se dodijelila raspberry-u u prethodnom koraku [slika 4.2]. Zatim se odabire korisničko ime (u ovom slučaju pi), te je potrebno pričekati da se server ažurira.

Nakon tog odabire se protokol i port koji će VPN koristiti. U ovom se projektu koristi TCP protokol koristeći port 443. Broj porta moguće je samostalno odrediti, ali pritom je potrebno pravilno konfigurirati prosljeđivanje portova na modemu kako bi se sav promet preusmjerio na željeni port.



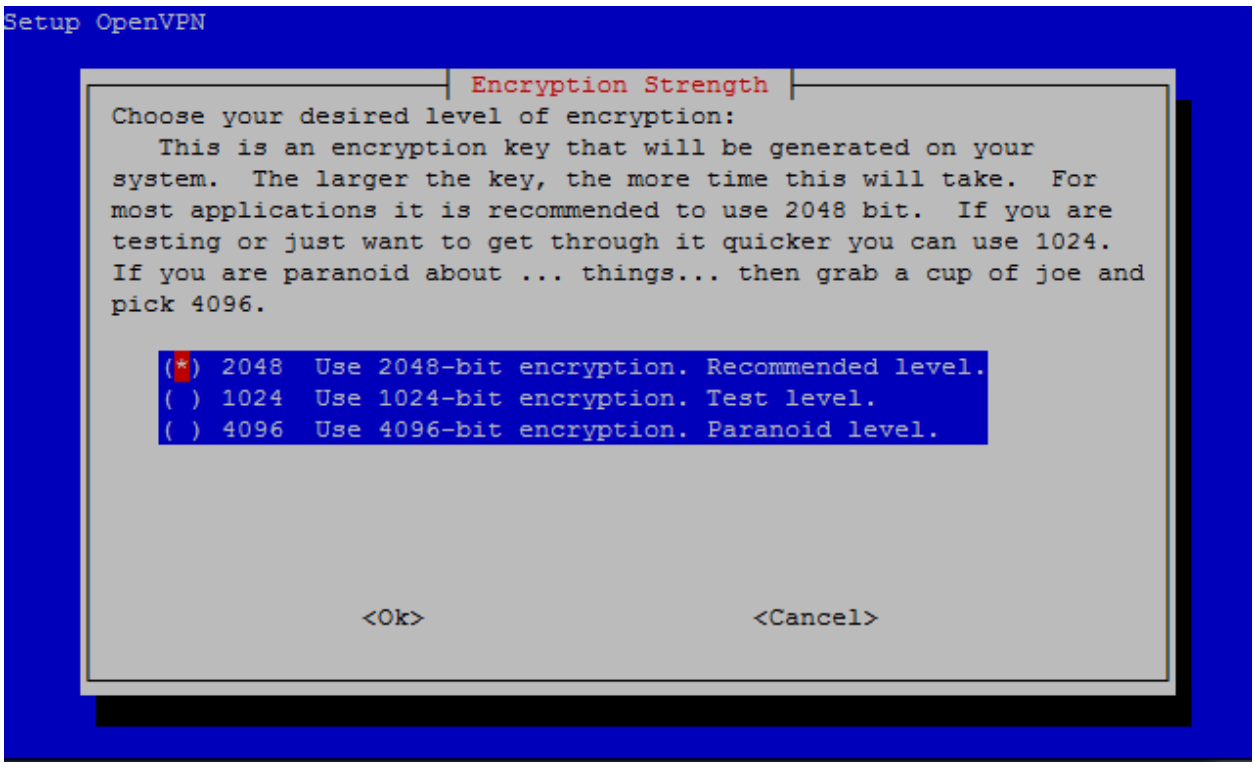
Slika 4.2 Postavljanje IP adrese

Kako bi se klijent mogao u bilo koje vrijeme spojiti na poslužitelj potrebno je odrediti javnu IP ili DNS adresu preko koje će se klijent spajati [slika 4.3]. Pošto se konfiguriranje statičke javne IP adrese obično naplaćuje od strane internetskih operatera, u ovom projektu koristi se noIP servis. NoIP servis [15] je usluga koja konstantno motri bilo kakvu promjenu javne IP adrese od strane internetskog operatera, te kod svake promjene spaja novu IP adresu sa definiranom DNS adresom. Na taj način moguće je pristupiti određenom korisniku mreže, bez potrebe za znanjem njegove javne IP adrese, već mu se pristupa preko DNS adrese. Za korištenje servisa potrebno je instalirati noIP aplikaciju na računalu u mreži kojoj želimo osigurati statičku adresu, ali danas mnogi modemi sami po sebi podržavaju DNS servere, te oni sami motre promjene u mreži, pa u ovom slučaju to nije potrebno.



Slika 4.3 Odabir DNS adrese

U sljedećem koraku potrebno je odabrati razinu enkripcije kojom će se šifrirati privatni ključ. Odabire se enkripcija od 2048 bita [slika 4.4], što je dovoljno za sigurnu komunikaciju. Kreiranje Ključa od 2048 bita traje oko 5 minuta, te je nakon tog potrebno ponovo pokrenuti Raspberry.



Slika 4.4 Odabir razine enkripcije

Potrebno je konfigurirati profile klijenata kojima će se dozvoliti pristup VPN serveru. Kreiranje korisničkog profila se vrši naredbom `[14]`:

```
pivpn add
```

Tu se određuje korisničko ime i lozinka, te se generira konfiguracija koja sadrži privatni ključ kojim klijent pristupa VPN serveru

Konfiguraciju je potrebno unijeti u OpenVPN aplikaciju na računalu kojim se klijent želi povezati sa serverom.

Postupak se ponavlja za željeni broj klijenata.

4.1.3. Spajanje na VPN poslužitelj

Povezivanje na VPN server vrši se preko OpenVPN aplikacije.
Konfiguracijsku datoteku koja se izradila na Raspberry pi-u potrebno je kopirati na mjesto instalacije OpenVPN-a, te se potom pomoću lozinke spaja na VPN server.
Nakon pokretanja OpenVPN aplikacije na Windows operativnom sustavu, u donjem desnom uglu potrebno je odabrati OpenVPN, te odabrati opciju spoji.
Nakon toga započinje spajanje na server, te se dodjeljuje privatna IP adresa preko koje je moguće pristupiti računalu u lokalnoj mreži Raspberry pi-a. [16]

4.2. PLC upravljanje

4.2.1. OpenPLC software

Instalacija OpenPLC softvera na raspberry pi vrši se pomoću komandne linije. Prije instalacije potrebno je instalirati programske pakete koji su potrebni za ispravan rad OpenPLC-a. Za instalaciju paketa, te promjenu sistemskih datoteka potrebno je biti prijavljen kao root korisnik što se čini naredbom:

```
sudo su
```

Prije svake instalacije dobro je ažurirati cijeli sustav koristeći naredbu:

```
apt-get upgrade
```

Instalacija paketa vrši se naredbama [13]:

```
apt-get install build-essential pkg-config bison flex
apt-get install autoconf automake libtool make nodejs git
```

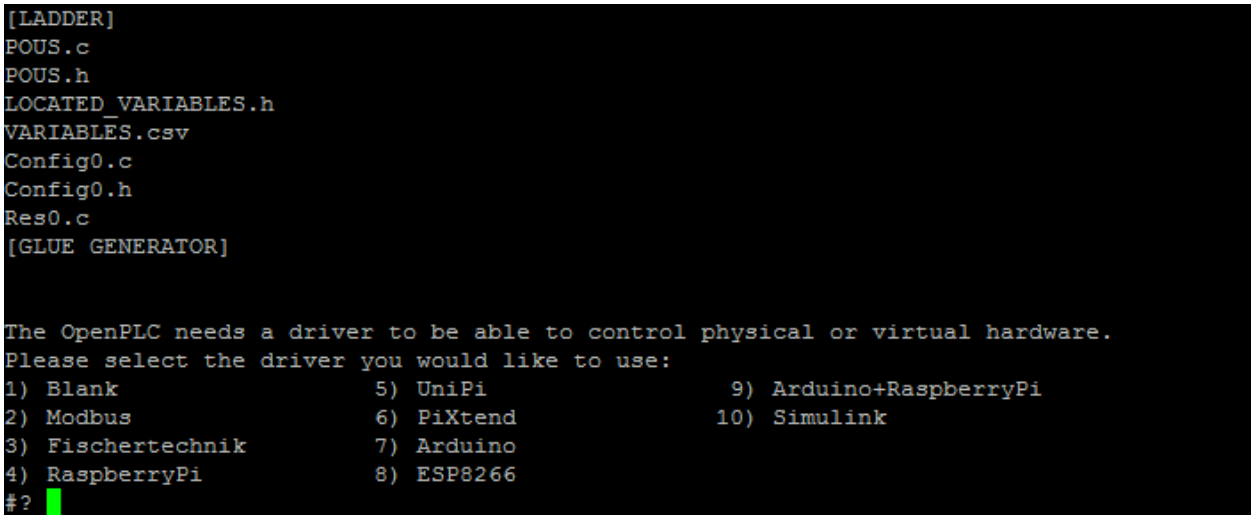
Nakon instalacije svih potrebnih programskih paketa potrebno je kopirati OpenPLC na raspberry pi. [17]
To se vrši naredbom:

git clone https://github.com/thiagoralves/OpenPLC_v2.git

Nakon kopiranja, potrebno je ući u mapu OpenPLC_v2, te pokrenuti proces instalacije:

```
cd OpenPLC_v2
./build.sh
```

Počinje proces instalacije, te je potrebno odabrati vrstu ulazno-izlaznog uređaja kojim se želi upravljati pomoću OpenPLC-a [slika 4.5]. OpenPLC može raditi na raznim platformama, ali u ovom projektu koristi se raspberry pi. Međutim unatoč korištenja raspberry platforme, ne odabire se ta platforma, već Modbus opcija. Razlog tome je potreba za upravljanjem ulazno-izlaznim uređajima preko modbus protokola. Odabirom raspberry platforme moguće je upravljati samo ulazno-izlaznim portovoma na samom raspberry pi uređaju.



Slika 4.5 Odabir uređaja za upravljanje

Nakon instalacije sama OpenPLC aplikacija pokreće se iz instalacijske mape naredbom:

```
nodejs server.js
```

Kako bi se izbjeglo ručno pokretanje aplikacije prilikom svakog uključenja uređaja, potrebno je osigurati automatski start, te garanciju da se prilikom bilo kakve greške aplikacija ponovno pokrene.

Automatski start se osigura izradom skripte za pokretanje, te dodavanje skripte u konfiguracijsku datoteku koja se pokreće prilikom pokretanja Raspbian sustava.
Postupak izrade skripte započinje izradom novog dokumenta pomoću nano editora u init.d datoteci.

```
nano /etc/init.d/skripta
```

Skripta sadrži konfiguraciju za nodejs servis, te je potrebno upisati put do nodejs instalacije, te put do aplikacije koja se želi pokrenuti:

```
#!/bin/sh
#/etc/init.d/myService
export PATH=$PATH:/usr/local/bin
export NODE_PATH=$NODE_PATH:/usr/lib/node_modules
case "$1" in
start)
exec forever --sourceDir=/home/pi/OpenPLC_v2 server.js -p
/home/pi/OpenPLC_v2 server.js
;;
stop)
exec forever stop --sourceDir=/home/pi/OpenPLC_v2/ server.js
;;
*)
echo "Usage: /etc/init.d/skripta {start|stop}"
exit 1
;;
esac
exit 0
```

Nakon izrade skripte potrebno joj je dati dozvolu za izvršenje:

```
chmod 755 /etc/init.d/skripta
```

Kako bi pokrenuli skriptu kod svakog pokretanja sustava, potrebno je dodati naredbu za pokretanje u rc.local skriptu:

```
#!/bin/sh -e
#
# rc.local
sh /etc/init.d/skripta start
exit 0
```

Pokretanjem OpenPLC aplikacije, automatski se pokreće cijeli server,te počinje sa radom.

Za pristup serveru, te konfiguraciju modbus uređaja i postavljanje programa koristi se web aplikacija kojoj se pristupa preko internetskog preglednika.
Za pristup serveru potrebno je upisati IP dresu raspberry uređaja, te port 8080.
Primjer: *http://192.168.8.10:8080*

Kad se pristupi web aplikaciji, moguće je pokrenuti i zaustaviti rad OpenPLC-a, te vidjeti dnevnik svih događaja poput spajanja modbus uređaja ili promjene programa.
Također ovdje se stavlja željeni program, te konfiguracija za spajanje modbus uređaja.
Web aplikaciji je moguće pristupiti iz lokalne mreže, a implementacijom VPN servera moguće je upravljati aplikacijom i preko interneta što otvara pristup daljinskom upravljanju.
Kod uploada programa, odabiremo datoteku programa spremljenu na računalu, te se OpenPLC server automatski ponovno pokreće, te počinje izvoditi novi program.
Konfiguracija modbus uređaja postavlja se na način da se u običnom tekst dokumentu ispišu parametri uređaja koji će se spojiti na OpenPLC, te se spremi kao konfiguracijska datoteka (.cfg).
Također moguće je stvoriti gotovu konfiguracijsku datoteku pomoću web aplikacije kojoj je moguće pristupiti putem OpenPLC-ovih internet stranica.
Konfiguracijska datoteka sadrži osnovne podatke o modbus uređaju, te je tako kod RTU protokola potrebno odrediti adresu serijskog porta na koji je uređaj spojen, u ovom slučaju to je "/dev/ttyUSB0", što predstavlja USB port raspberry pi-a na koji je spojen USB na RS-485 konverter. Tip serijske veze, brzinu, paritet, te broj podatkovnih bitova su bitni parametri kod RTU protokola potrebni za ispravan rad. Kod TCP protokola potrebna je IP adresa i port. Koristi se port 502 koji je standard kod modbus protokola.
Također vrlo je bitno točno definirati broj ulazno - izlaznih registara koje će slave uređaj slati, jer u protivnom će se dogoditi greška u komunikaciji.
Tako se odabire broj digitalnih i analognih ulaza i izlaza, također odabire se i odstupanje (eng offset) pojedinih polja u modbus tablici. Bitno je odrediti odgovarajući offset koji je kompatibilan sa strukturom modbus podataka u samom uređaju koji se kontrolira. Ako se krivo definira offset registara može doći do greške u komunikaciji ili krivo očitanih podataka.

4.2.2. PLCopen editor

PLCopen editor je softver koji služi za pisanje PLC programa, te njihovo kompiliranje u format pogodan za izvedbu u OpenPLC softveru.
Sam editor dio je beremiz projekta i pisan je u python programskom jeziku.
Beremiz je programska platforma osmišljena za pisanje PLC programa, koja radi po industrijskim standardima za automatizaciju i programsku podršku PLC-a poput IEC-61131.
Beremiz je besplatna platforma, te je osmišljena u svrhu jednostavnijeg načina pisanja PLC programa, te mogućnosti korištenja svakog tipa procesora kao PLC platforme. Time se razlike u pisanju programa za PLC uređaje različitih proizvođača uklanjaju, te koncept PLC programa postaje standard.
PLCopen editor je dio beremiz sustava, te pojednostavljuje upotrebu beremiza za pisanje PLC programa za openPLC platformu. Pisanje programa moguće je u 5 standardnih programskih jezika po IEC-61131.
U PLCopen editoru moguće je napisati program u bilo kojem od 5 standardnih jezika, te kompilirati ga u ST jezik za korištenje u OpenPLC-u. [13]

Vrste programskih jezika:

Ladder logic (LD): Ladder je grafički način pisanja PLC programa koji je sličan relejnoj tehnici upravljanja.Svaki programski blok predstavlja jedan ili više elemenata u nizu upravljanja postrojenjem, te je savršen uvod u PLC programiranje.

Function block diagram (FBD): FBD je također grafički način programiranja, gdje se pisanje programa temelji na stvaranju veza, te logičkih funkcija između ulazne i izlazne varijable.

Sequential function chart (SFC): SFC je grafički programski jezik koji služi za programiranje procesa pojedinačnim koracima.

Instruction list (IL): IL je vrsta osnovnog programskog jezika koja se temelji na asemblerskom jeziku.

Structured text (ST): ST je vrsta programskog jezika bazirana na Pascalu, te je jezik koji OpenPLC softver koristi za izvedbu programa.

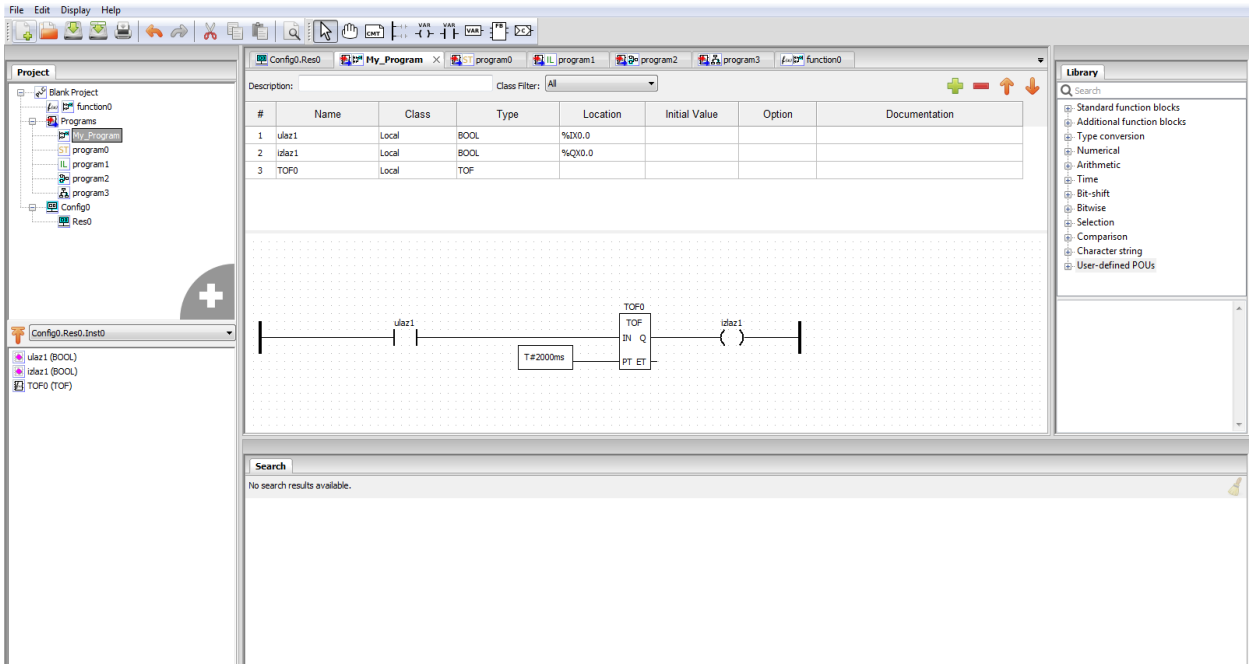
PLCopen editor moguće je instalirati na Windows, Linux i MacOS operacijskim sustavima. Proces stvaranja novog programa sastoji se od odabira vrste programskog jezika, pisanja programa, stvaranja konfiguracije, te kompiliranje programa u ST programski jezik.

Biblioteka funkcijskih blokova sastoji se od:

- **Standardnih funkcija: tajmeri, brojači i bistabili**
- **Matematičkih i aritmetičkih funkcija**
- **Funkcije za pretvorbu tipa podataka**
- **Vremenskih funkcija**
- **Logičkih funkcija**
- **Funkcije za usporedbu**

Varijable mogu biti ulazne, izlazne i pomoćne te se označavaju sa:

- **%IX za ulazne varijable**
- **%QX za izlazne varijable**
- **%MX za pomoćne varijable**



Slika 4.6 Primjer pisanja programa u ladder programskom jeziku

Lokacija varijabli u programu mora odgovarati memorijskim lokacijama u modbus tablici. OpenPLC kod rada sa modbus protokolom sprema varijable na memorijsku lokaciju koja je definirana u konfiguracijskoj datoteci za modbus uređaje. Ako se u konfiguracijskoj datoteci navede da ima 5 digitalnih ulaza, varijable će se u OpenPLC-u pojaviti kao %IX0.0 - %IX0.4.
Kod spajanja više uređaja pomoću modbus protokola, za svaki sljedeći uređaj koji se definira u konfiguracijskoj datoteci lokacija varijabli će se nastavljati. Time ako prvi uređaj obuhvaća varijable od %IX0.0 do %IX2.2, sljedeći deklariran uređaj obuhvaćati će varijable počevši od %IX2.3. Nakon gašenja OpenPLC sustava stanje varijabla se poništava na početnu vrijednost (0). [13]

- **Pisanje programa**

Samo programiranje se vrši stvaranjem ulaznih, izlaznih i pomoćnih varijabla, te njihovim povezivanjem sa funkcijama. Kod grafičkih programskih jezika odabiremo funkcijski blok iz biblioteke mogućih funkcija, te ih linijama povezujemo sa varijablama. Kod IL i ST jezika uz mogućnost ručnog pisanja programa editor nam omogućava korištenje funkcijskih blokova na način da ih pretvara u programski kod koji odgovara odabranom jeziku.
Moguće je i stvarati posebne funkcijske blokove, gdje se definira program koji će funkcijski blok izvoditi, i time pojednostaviti komplicirane programe.
Kod pisanja programa potrebno je konfigurirati parametre po kojima će se izvršavati program, tako se pod opcijom config mora odrediti TaskMain kod kojeg određujemo brzinu izvršavanja PLC programa.
Nakon pisanja programa, potrebno je prevesti program pomoću kompilatora u oblik koji OpenPLC može čitati. Kompiliranje se obavlja u OpenPLC editoru na način da se odabere funkcija generate program ili kraticom CTRL+G. Nakon kompiliranja program se sprema na računalo, te je potrebno napraviti upload u OpenPLC sustav. To se vrši pomoću web aplikacije, kojoj se dostupa na način da se u preglednik upiše adresa Raspberry pi-a, te port 8080.
Upload se vrši odabirom programa koji je spremljen na računalu, te pritiskom na naredbu Upload Program. Nakon toga OpenPLC počinje izvoditi novi program.
Kod konfiguracije modbus uređaja stvara se konfiguracijska datoteka [slika 4.8], te se upload vrši odabirom datoteke, te pritiskom na naredbu Upload Configuration.

TIP REGISTRA	ADRESA	VELIČINA	VRIJEDNOST
DIGITALNI ULAZ	%IX0.0 - %IX99.7	1 BIT	0/1
DIGITALNI IZLAZ	%QX0.0 -%QX99.7	1 BIT	0/1
ANALOGNI ULAZ	%IW0 - %IW99	16 BIT	0 – 65535
ANALOGNI IZLAZ	%QW0 - %QW99	16 BIT	0 - 65535
POMOĆNI	%MW0 - %MW1023	16 BIT	0 - 65535
POMOĆNI	%MD0 - %MD1023	32 BIT	0 - 4294967295
POMOĆNI	%ML0- %ML1023	64 BIT	

Slika 4.7 Memorijske lokacije u OpenPLC sustavu

```
1  Num_Devices = "1"
2
3  device0.name = "device1"
4  device0.protocol = "RTU"
5  device0.slave_id = "10"
6  device0.address = "/dev/ttyUSB0"
7  device0.IP_Port = ""
8  device0.RTU_Baud_Rate = "9600"
9  device0.RTU_Parity = "N"
10 device0.RTU_Data_Bits = "8"
11 device0.RTU_Stop_Bits = "1"
12 device0.Discrete_Inputs_Start = "0"
13 device0.Discrete_Inputs_Size = "6"
14 device0.Coils_Start = "16"
15 device0.Coils_Size = "4"
16 device0.Input_Registers_Start = "4"
17 device0.Input_Registers_Size = "2"
18 device0.Holding_Registers_Start = "2"
19 device0.Holding_Registers_Size = "2"
```

Slika 4.8 Modbus konfiguracija

5. Hardversko rješenje

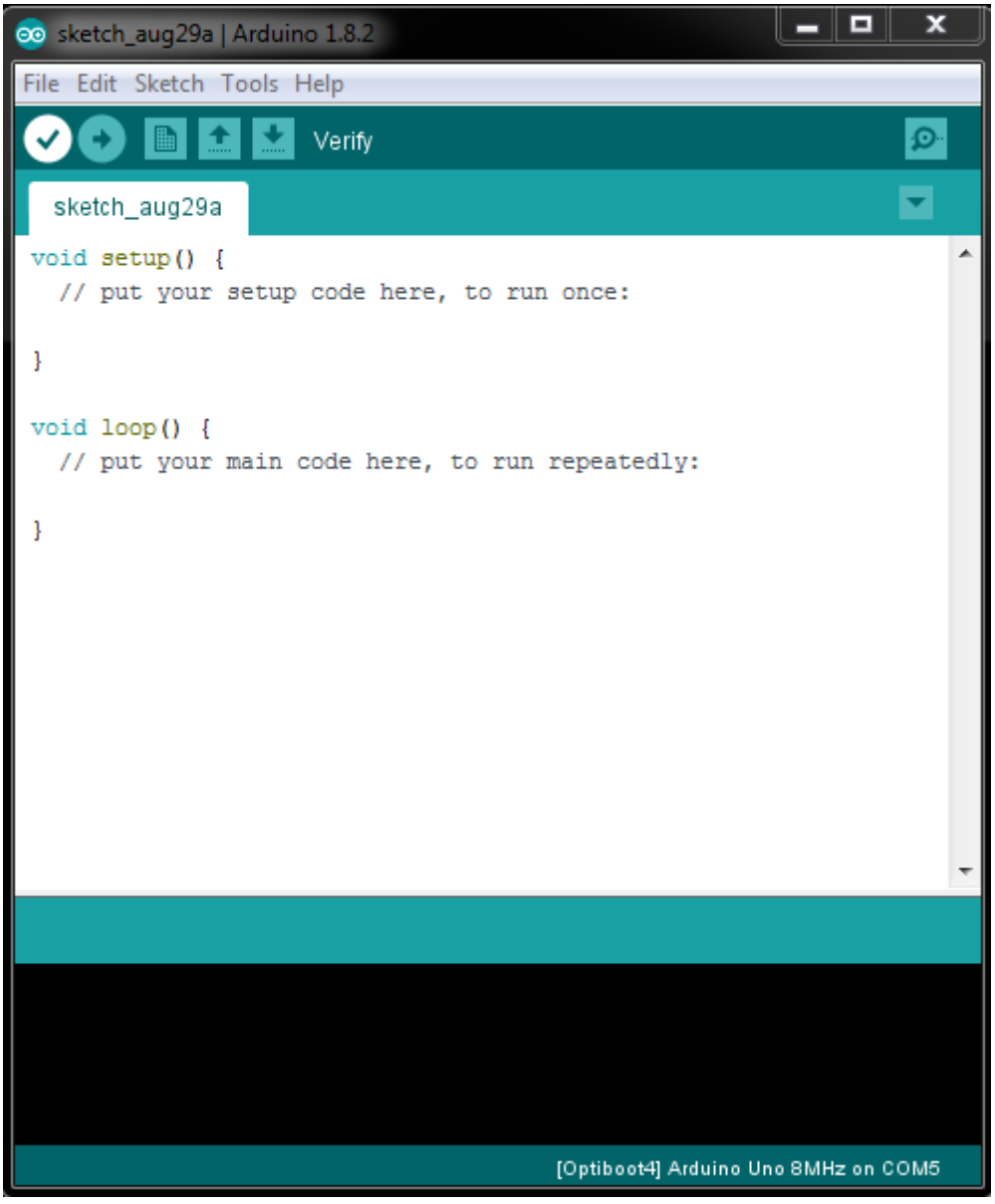
5.1. Atmega328P

ATMega 328P je 8 bini AVR mikrokontroler izrađen od strane Atmela i Mikrochipa.
Središte mikrokontrolera je 8 bitni AVR procesor koji je u mogućnosti raditi na brzinama do 20MHz. Sadrži 32kB programske memorije, 2kB RAM memorije, te 1kB EEPROM memorije.
Za komunikaciju koristi UART, 2 SPI porta, te I2C. Implementirani su dva 8 bitna, te jedan 16 bitni tajmer ili brojač.
Sadrži 23 ulazno izlazna porta, gdje je 8 portova implementirano u 10 bitni analogno-digitalni konverter (ADC), te sadrži 6 PWM portova.
Radi pomoću vanjskog oscilatora najčešće od 16MHz, ali ima ugrađeni oscilator od 8MHz.
U mogućnosti je raditi na naponima od 2.7V do 5.5V, te ima vrlo malu potrošnju električne energije.
ATMega 328p se koristi u projektima upravljanja i komunikacije, te ga najčešće možemo sresti na Arduino platformama. [18]

5.2. Način programiranja

ATMega328p je moguće programirati na razne načine, ali najčešće se koristi Arduino IDE [Slika 5.1], te ATMEL studio softver za pisanje i upload programa. Sam program se piše u C programskom jeziku. U ovom projektu koristi se Arduino IDE zbog jednostavnosti te podrške za razne biblioteke.
Proces programiranja se bazira na deklaraciji varijabli i pojmova, konfiguraciji ulazno - izlaznih pinova i serijske komunikacije, te pisanju programa koji će se periodično izvoditi.
Kod uploada programa sa Arduino IDE potrebno je odabrati odgovarajuću vrstu bootloadera koja je instalirana na mikrokontroler. Bootloader je softver koji omogućava programiranje mikrokontrolera bez korištenja vanjskog programatora. Prednost bootloadera je jednostavno programiranje preko serijske veze, ali sam bootloader zauzima mjesto u programskoj memoriji.
U slučaju da nam treba više memorije, moguće je programirati mikrokotroler pomoću serijskog programatora (AVR ISP). Također na taj način se instalira bootloader.
Nakon odabira bootloadera ili odabira programiranja preko ISP-a programski kod koji želimo instalirati na mikroprocesor se kompilira, te uploada pritiskom na naredbu Upload.

Nakon uploada programa on automatski započinje sa radom.



Slika 5.1 Arduino IDE okruženje

5.3. Implementacija ATMega 328p mikrokontrolera u OpenPLC sustav

Odabran je 328p mikrokontroler zbog velikog broja ulazno izlaznih portova, te mogućnost korištenja modbus protokola za serijsku komunikaciju.
Ideja je koristiti mikrokontroler za direktno upravljanje industrijskim procesima i strojevima, gdje je mikrokontroler u mogućnosti izvoditi dio upravljanja sam, te slanja i primanja bitnih podataka pomoću modbus protokola koristeći RS-485 komunikaciju. Mogućnost izvođenja programskog koda uz modbus komunikaciju rasterećuje OpenPLC sustav, te omogućava rad uređaja upravljanog mikrokontrolerom čak i u slučaju prekida komunikacije.

Za mikrokontroler izrađena je konceptna pločica kako bi bio u mogućnosti upravljati industrijskim signalima. Pločica služi kao ulazno izlazni dodatak mikrokontroleru, kako bi se mali i slabi signali do 5V koje je mikrokontroler u mogućnosti proizvesti, pretvorili u industrijske signale, te se time otvara mogućnost upravljanja relejima i senzorima.

Zbog pristupa dodatnim portovima, mikrokontoler se koristi bez vanjskog oscilatora, i zbog toga je potrebno kod programiranja koristiti Optiboot bootloader [19] ili ISP programiranje gdje se odabire način rada sa unutarnjim oscilatorom od 8MHz.

5.4. El. Shema i tiskana pločica

Tiskana pločica izrađena je kao ulazno – izlazna platforma kojom je moguće upravljati putem modbus protokola. Srce pločice je Atmega328p mikrokontroler koji omogućuje upravljanje i komunikaciju. Pločica sadrži zaštitne komponente koje odvajaju mikrokontroler od industrijskih signala i time štite od kvara. Radni napon pločice je 24V DC što odgovara industrijskim standardima i time omogućuje lako korištenje u postrojenju.

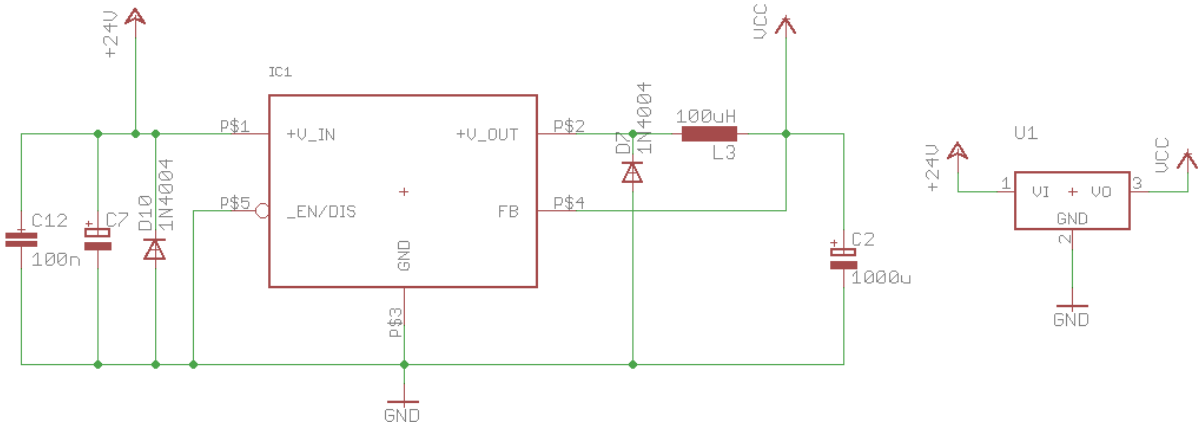
Kompletna shema te dizajn pločice mogu se vidjeti u [Prilog 2] i [Prilog 3].

5.4.1. Dijelovi sheme

- **Napajanje [Slika 5.2]**

Koristi se LM2576 step-down regulator kojem je dopušteni ulazni napon od 7 do 40V DC, a izlazni napon regulirani 5V, sa maksimalnom strujom od 3 ampera što je i više nego dovoljno za rad mikrokontrolera, te ostalih komponenti.

Prednost step-down regulatora je velika učinkovitost, i time malo zagrijavanje samog čipa, pa nije potrebno hlađenje. Također implementirani su kontakti za jednostavniji regulator.



Slika 5.2 Napajanje

- **Ulazni sklop [Slika 5.3] [Slika 5.4]**

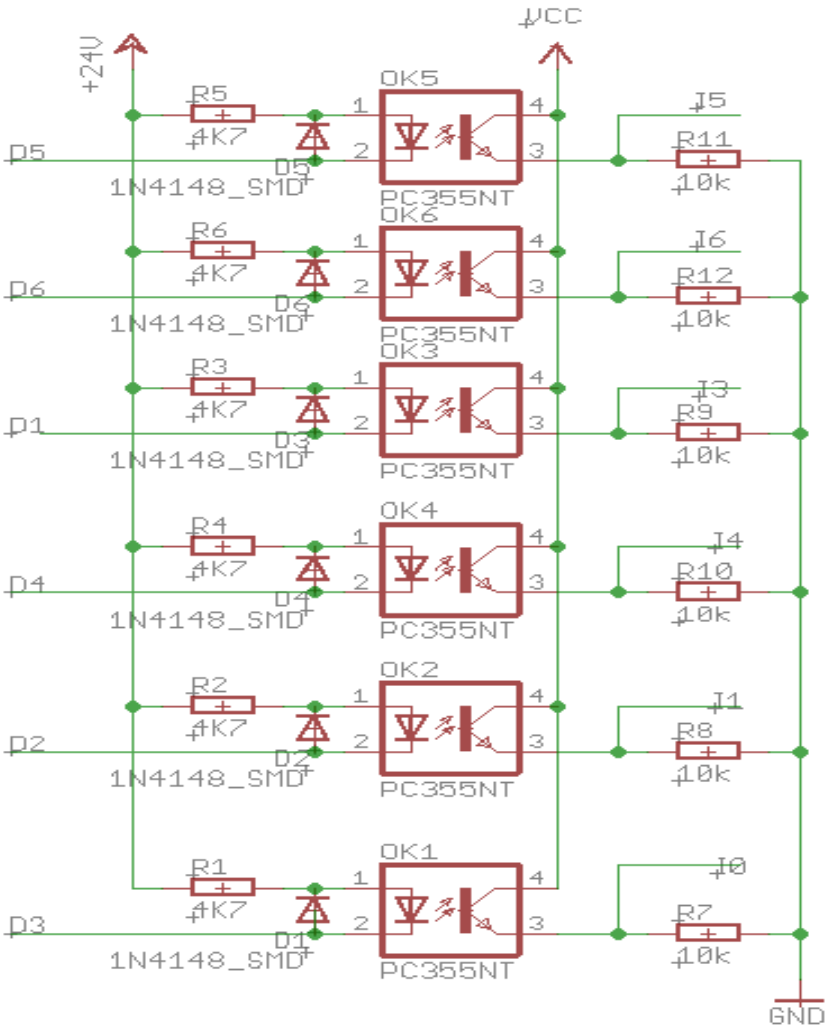
Ulazni dio pločice temelji se na galvanskom odvajanju industrijskih signala, te njihove prilagodbe na naponske i strujne razine primjerene mikrokontroleru.

Implementirano je šest digitalnih ulaznih portova, te dva analogna ulaza koji su u mogućnosti raditi kao ulaz za uređaje sa rasponom od 4-20mA.

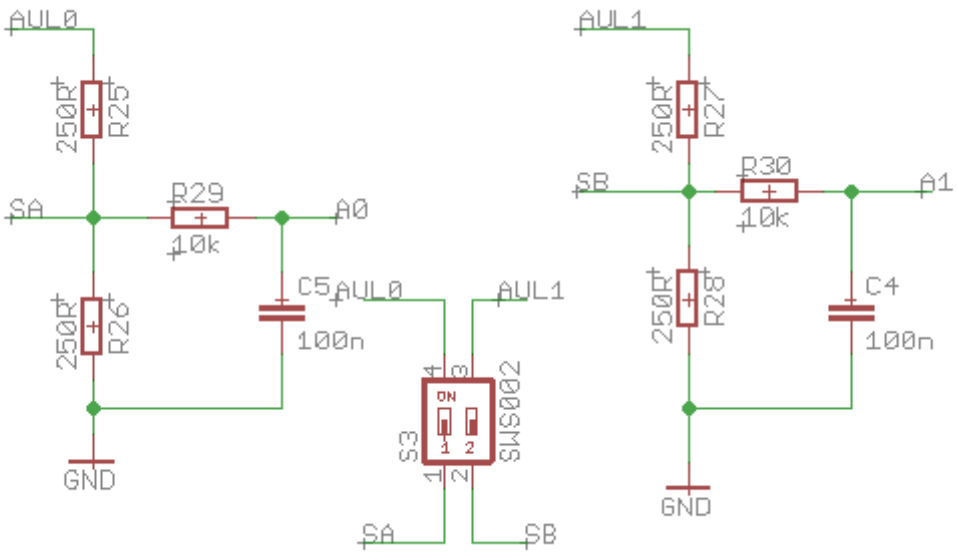
Vanjski dio ulazne sheme, gdje se ulazni uređaji spajaju pomoću industrijskih konektora radi na 24V, što je standard kod PLC uređaja u industriji. Ulazni signali se vode na ulaz optocouplera, gdje je u seriju sa ulazom spojen otpornik koji ograničava struju kako bi se prilagodila specifikacijama optocouplera. Koriste se PC357 optocoupleri, kojima je dopuštena ulazna stuja 5mA, i time se za 24V odabire otpornik od 4.7k ohma što ograničava struju na 5mA.

Također stavljena je zaštitna dioda paralelno ulazu koja služi za zaštitu optocouplera u slučaju spajanja uređaja obrnutim polaritetom. Kod takvog slučaja dioda će provesti, te spriječiti štetu na optocoupleru. Nakon galvanskog odvajanja na izlaznoj strani optocouplera signal se vodi na ulaz mikrokontrolera. Korišten je otpornik od 10k ohma spojen od ulaza u mikrokontroler prema uzemljenju kako bi se spriječilo neželjeno pogrešno očitanje logičke jedinice kad nema signala na ulazu zbog raznih smetnji. Taj princip se naziva pull down, gdje otpornik spriječava variranje napona na portu,te ga drži na nultom potencijalu. Kod dolaska ulaznog signala, napon napajanja je dovoljan da se registrira logička jedinica.

Ulazni dio također sadrži dva analogna ulaza, koji su u mogućnosti raditi od 0-10V, te 0-5V kao i za uređaje sa 4-20mA. Odabir mjernog područja vrši se pomoću malih sklopki na samoj pločici, gdje se bira između naponskog djelila za 0-10V, te direktnog očitanja 0-5 V koji također služi kao ulaz za 4-20mA. Naponsko djelilo je izvedeno pomoću dva otpornika od 250 ohma, te kad se napon od 10 volta spoji na ulaz, na izlazu naponskog djelila dobije se 5V koji su prigodni za mikrokontroler. U slučaju da ne koristimo naponsko djelilo, otpornik od 250 ohma služi za registriranje signala 4-20 mA. Kod 20mA pad napona na otporniku će biti 5V, dok će kod 4mA pad napona bit 1V, te na taj način mikrokontroler očitava vrijednost signala. Paralelno ulazu spojen je filterski kondenzator od 100nF koji služi za uklanjanje smetnji, te točnije očitanje analogne vrijednosti.



Slika 5.3 Sklop digitalnih ulaza

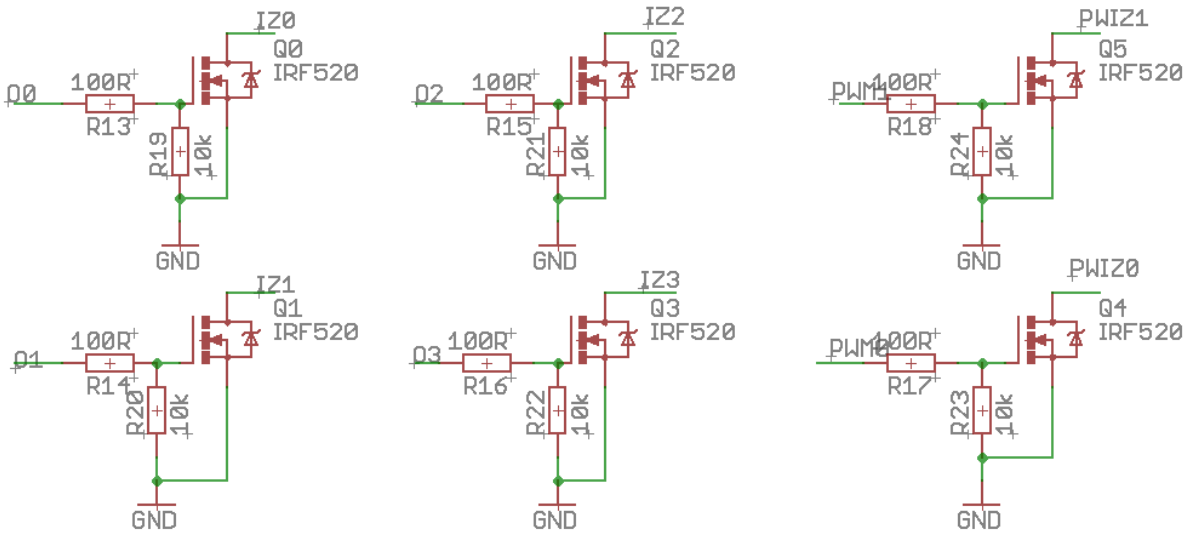


Slika 5.4 Sklop analognih ulaza

- **Izlazni sklop [Slika 5.5]**

Izlazni dio pločice se sastoji od šest MOSFET tranzistora koji funkcioniraju kao tranzistorske sklopke, gdje su dva izlaza u mogućnosti raditi kao PWM izlazi. Izlazni tranzistori su potrebni za prilagođavanje signala mikrokontrolera na industrijskih 24 volti. Korišteni su tranzistori IRF 520 koji mogu izdržati do 10 Ampera struje u normalnom radu. Bit tranzistora je da prilagode napon, te rasterete mikrokontroler koji može dati vrlo malu struju kako bi se moglo upravljati relejima i ostalim uređajima.

Izlazna shema također koristi pull down otpornik kako bi se osigurao pouzdani rad tranzistora.



Slika 5.5 Izlazni sklop

- **Komunikacija**

Za komunikaciju sa OpenPLC-om koristi se modbus protokol preko RS-485 serijske veze. Takva komunikacija ostvarena je preko MAX485 čipa, koji pretvara serijsku komunikaciju iz mikrokontrolera u RS-485 vezu. Serijska veza iz mikrokontrolera se dovodi na ulaz MAX485 čipa, zajedno sa jednim portom mikrokontrolera koji služi za upravljanje komunikacije.
Izlaz iz čipa sadrži A i B pinove koji služe za prijenos podataka preko RS-485, te se oni odvode na konektore na rubu pločice. Paralelno A i B pinovima spojen je otpornik od 250 ohma koji služi za korekciju impedancije u slučaju prijenosa podataka na duže staze.
Na putu od mikrokontrolera do čipa postavljen je konektor koji odvaja RX vod od čipa, te je na njega moguće spojiti sklopku. Razlog toga je da se prekine komunikacija sa čipom, i time dozvoli serijska uart komunikacija i instaliranje programa. Ako se odvija komunikacija sa MAX485 čipom nije moguće koristiti serijski port u drugu svrhu, tako se prekidom komunikacije omogućuje njegovo korištenje. Sami RX i TX vodovi serijskog porta spojeni su na konektor s lijeve strane pločice, zajedno sa napajanjem i pinom za reset kako bi bilo moguće programiranje. Pločica također sadrži ISP konektor za programiranje, te konektor za I2C komunikaciju za moguće spajanje displeja. Napajanje za I2C konektor odvojeno je od napajanja pločice tranzistorskom sklopkom, za mogućnost isključenja napajanja displeja. Konektor SV2 također sadrži I2C, ISP pinove, napajanje, te pin P0 na kojem je spojena statusna led dioda. Pločica sadrži dvije led diode, jedna spojena direktno na napajanje za signalizaciju rada uređaja, te druga spojena na pin mikrokontrolera kojom se može upravljati po volji.

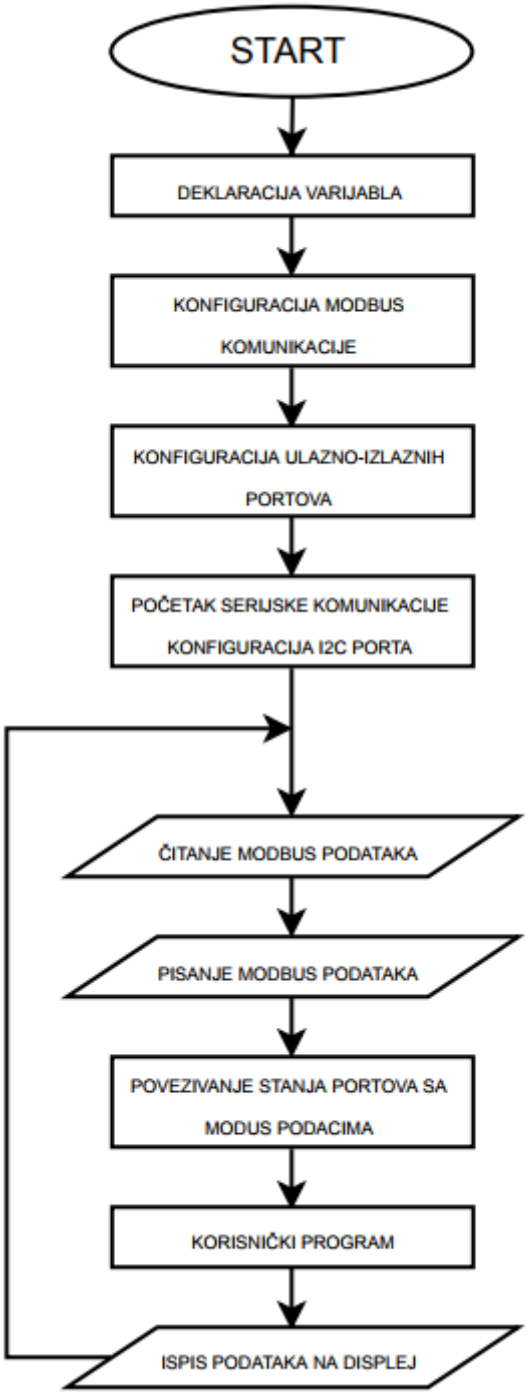
5.4.2. Dizajn pločice [Prilog 3]

Pločica je dizajnirana u programu Autodesk Eagle 8.2.2.
Kod dizajniranja pločice potrebno je paziti na dimenzije, kako bi pločica odgovarala kučištu.
Pločica je dizajnirana u skladu sa industrijskim standardima. Tako su ulazni i komunikacijski konektori stavljeni s gornje strane pločice, dok su izlazni konektori i napajanje stavljeni s donje strane. Konektori za programiranje stavljeni su sa strane pločice.
Kod samog crtanja tiskanih vodova pažnja je posvećena na urednost, te razmak između vodova kako nebi došlo do greške. Pločica je zalivena bakrom u spoju sa GND kako bi se uklonile elektromagnetske smetnje.

6. Programsko rješenje za Atmega 328P

Programsko rješenje se bazira na modbus biblioteci koja omogućuje modbus komunikaciju između mikrokontrolera i ostalih modbus uređaja. Podaci dobiveni od ulazih portova povezuju se sa podacima u modbus tablici, dok se podaci dobiveni od OpenPLC sustava povezuju sa izlaznim portovima. Cijela modbus tablica se šalje putem modbus protokola u OpenPLC sustav.
U programski kod implementiran je ispis podataka na displeju koji se priključuje putem I2C porta. Korištenje displeja je predviđeno za ispis važnih varijabla i upozorenja neposredno uz mikrokontroler, što omogućuje brz i jednostavan pogled u stanje sustava.
Također, mikroprocesor osim slanja i primanja podataka u OpenPLC sustav, može i sam izvoditi logiku za upravljanje. Kod takvog upravljanja vrlo je bitno da je programski kod kratak, te ne zahtjeva mnogo vremena da se izvrši kako bi se komunikacija s OpenPLC sustavom odvijala bez poteškoća. Zbog toga upravljanje je ograničeno, ali može poslužiti za jednostavnu obradu podataka prije slanja modbus protokolom kao i nastavak rada mikroprocesora kod prekida komunikacije sa OpenPLC sustavom, što pridonosi sigurnosti i pouzdanosti sustava.
Kompletan program može se vidjeti u Prilogu 1.

6.1. Blok dijagram



Slika 6.1 Blok dijagram

7. Programsko rješenje za OpenPLC

Program za OpenPLC pisan je u Ladder programskom jeziku, te je osmišljen za jednostavan prikaz mogućnosti, te funkcionalnosti OpenPLC sustava.

Program manipulira ulaznim varijablama I0.0 i I0.1, izlaznim varijablma Q0.1 i Q0.2, te analognim ulazima te izlazima: IW0, IW1, QW0, QW1.

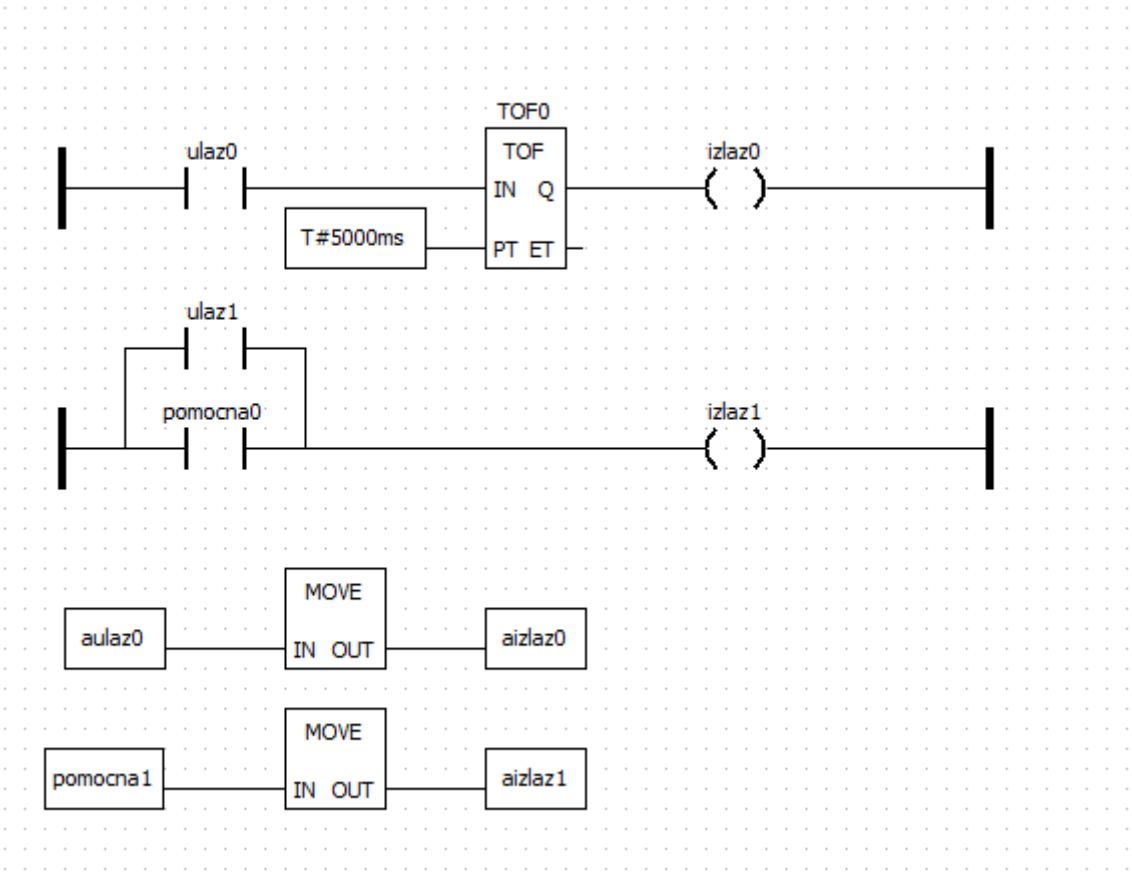
U program su implementirane pomoćne varijable kako bi se osiguralo lakše upravljanje pomoću SCADA sustava.

Program omogućuje uključenje varijable Q0.0 (izlaz1) pomoću I0.0 (ulaz0), te pomoću tajmera isključenje za 5 sekundi.

Izlaz I0.1 (izlaz1) moguće je uključiti pomoću ulaza I0.1(ulaz1) te pomoćne varijable (pomoćna0) koja je povezana sa SCADA sustavom.

Analogni ulaz AI0 (aulaz0) povezan je sa izlazom AQ0 (aizlaz0)

Analogni izlaz AQ1 (aizlaz1) povezan je sa pomoćnom varijablom (pomocna1) za upravljanje SCADA sustavom.



Slika 7.1 PLC program

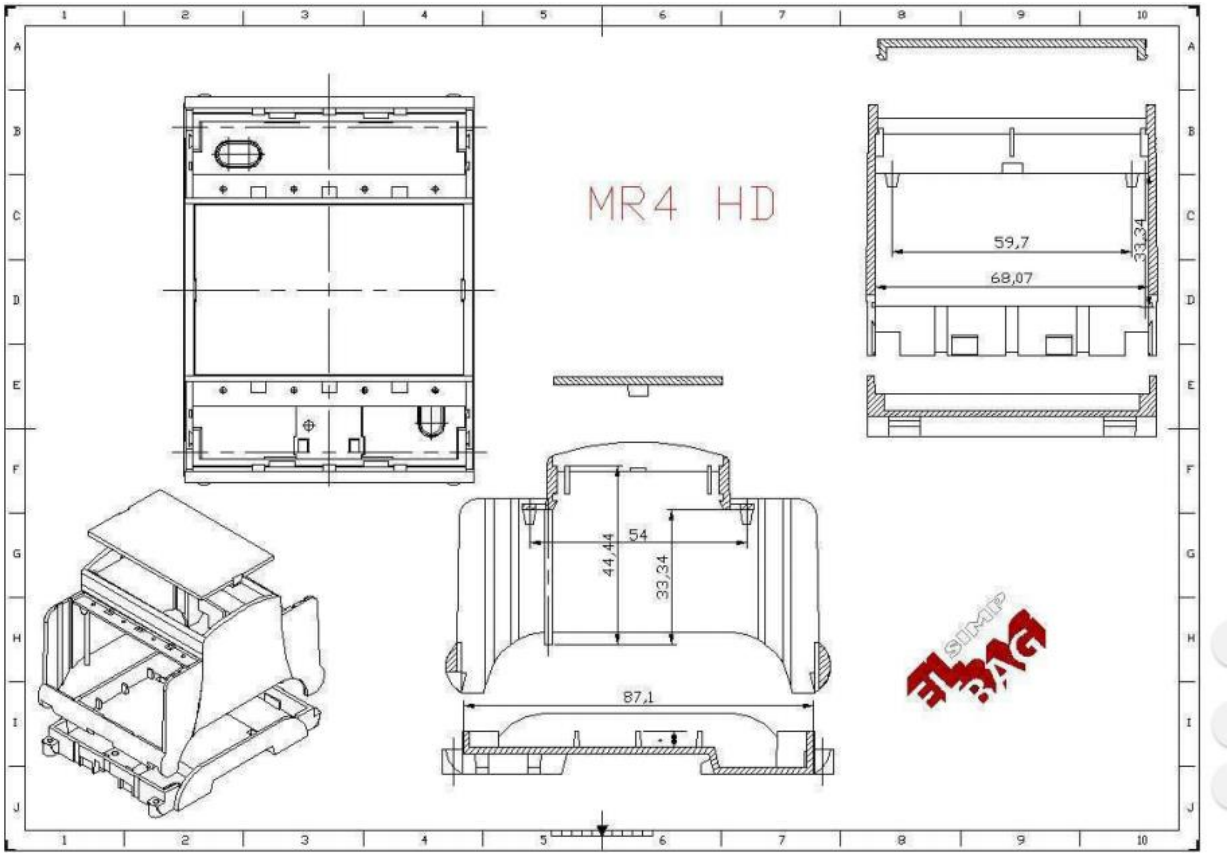
8. Kučište

Kučište Schukat MR-4 odabrano je u skladu sa dimenzijama i standardima industrijskih električnih ormara za jednostavnu instalaciju u sustav. [20]

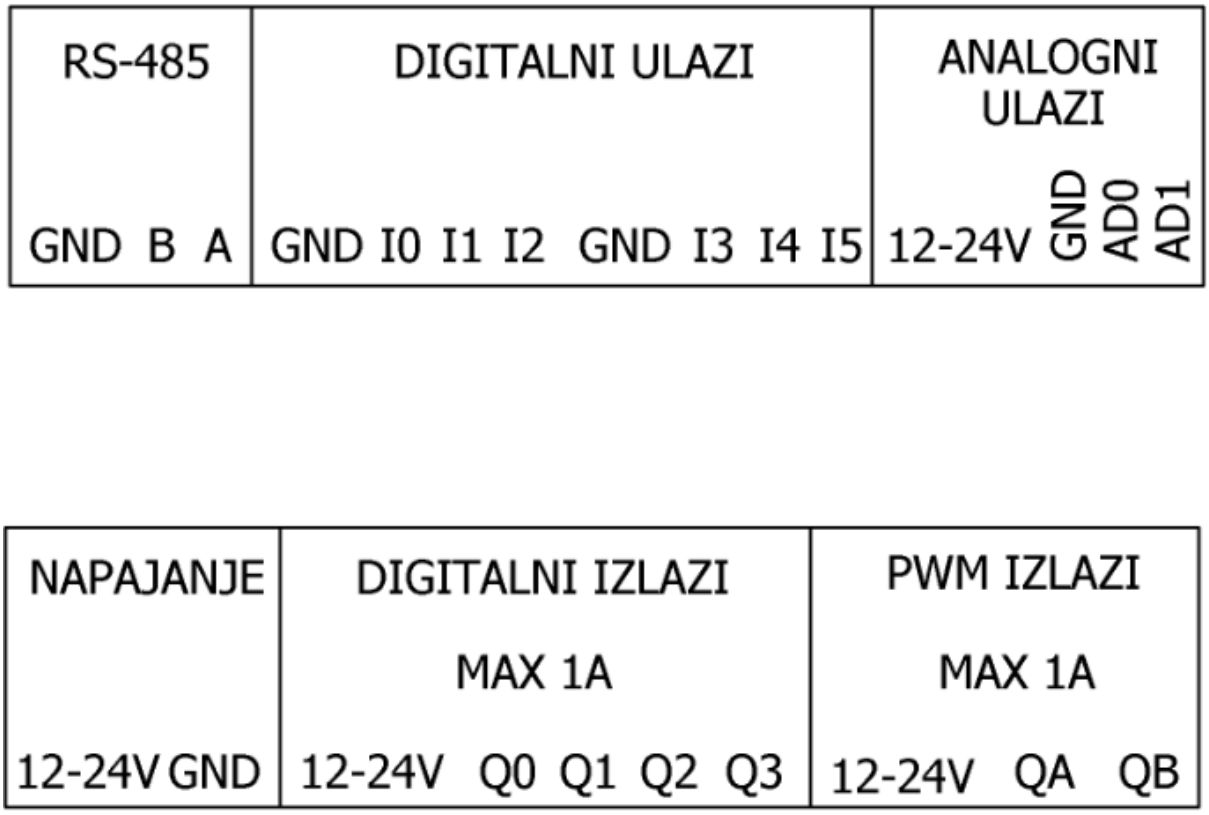
Kučište ima mogućnost ugradnje u električne ormare na standardnu šinu.

Sa gornje i donje strane kučišta nalaze se poklopci koji su obrađeni da odgovaraju konektorima na pločici. Također naljepnice sa nazivima pojedinih konektora nalaze se s prednje strane poklopca [Slika 8.2]. Na gornji poklopac montiran je displej, te svjetlovod za dvije led diode koje se nalaze na pločici. Sama tiskana pločica izrađena je po dimenzijama koje odgovaraju kučištu.

Shema kučišta: [Slika 8.1]



Slika 8.1 Shema kučišta [20]



Slika 8.2 Naljepnice

9. Upravljanje pomoću SCADA sustava

Za upravljanje sustava koristi se WinCC SCADA sustav.

Prikaz sustava [Slika 9.1] rađen je prema PLC programu [Slika 7.1]

Pri korištenju WinCC sustava potrebno je izraditi grafički prikaz koji će označavati pojedine komponente sustava, te omogućiti njihovo upravljanje.

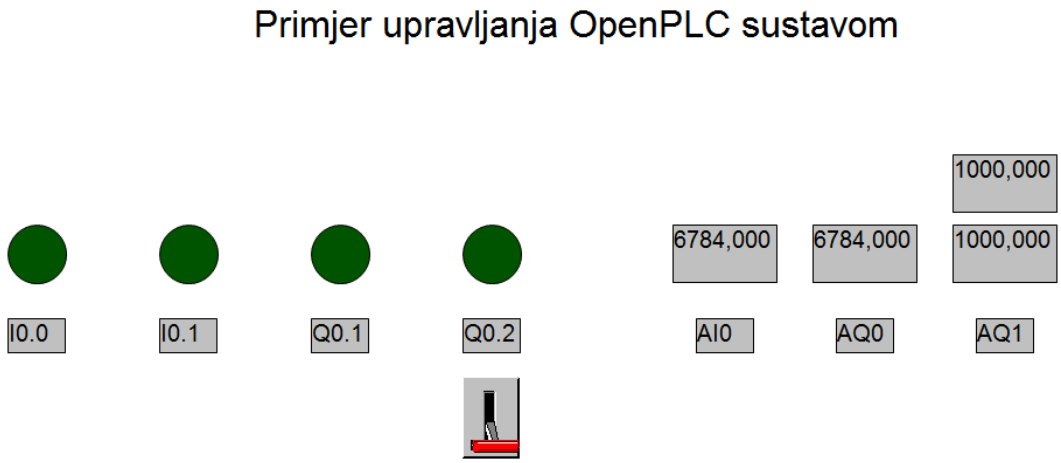
Nakon grafičkog dizajna potrebno je povezati elemente u WinCC sustavu sa registrima u OpenPLC sustavu.

Povezivanje elemenata obavlja se u Tag managmentu. U Tag manageru potrebno je stvoriti novu vezu gdje se odabire Modbus TCP/IP protokol kako bi mogli komunicirati sa OpenPLC-om. Odabire se vrsta procesora, koja kod OpenPLC sustava nije bitna, pa se preskače na odabir porta 502, te je potrebno navesti IP adresu Raspberry pi uređaja, te adresu slave uređaja (u ovom slučaju 255). Nakon stvaranja veze potrebno je stvoriti tagove kako bi povezali OpenPLC registre sa objektima u WinCC-u. OpenPLC adresira varijable dobivene od svojih modbus uređaja prema modbus tablici, te je kod povezivanja potrebno odabrati odgovarajuće adrese u WinCC sustavu. Kod stvaranja tag-a također se odabire veličina podataka, tako za digitalne tagove odabiremo binarni tag, a za analogne 16 bitnu riječ. Nakon stvaranja tagova preostaje samo povezivanje grafičkih objekata sa određenim tagovima kako bi se omogućio
prikaz te upravljanje.

9.1. Latencija sustava

Latencija sustava ovisi o brzini serijske komunikacije između OpenPLC-a i modbus uređaja, te brzini izvođenja PLC programa. Kod Atmega328p procesora izabrana je brzina prijenosa od 9600bps, dok je brzina izvođenja PLC programa 1 ms. Kod praktičnog mjerenja izmjerena je latencija od 400ms kod direktnog upravljanja pomoću digitalnih ulaza, te 100 ms kod upravljanja pomoću SCADA sustava.

Takva latencija pogodna je za osnovne funkcije čitanja i pisanja analognih i digitalnih portova, dok je kod aplikacija koje su vremenski zahtjevnije potrebno smanjiti latenciju. Smanjenje latencije može se osigurati povećanjem brzine serijske komunikacije između mikrokontrolera i OpenPLC sustava.



Slika 9.1 Primjer SCADA upravljanja

10. Zaključak

Sustav upravljanja pomoću OpenPLC-a pokazao se kao pouzdan te učinkovit način za automatizaciju sustava. Niska cijena, te princip otvorenog koda čine sustav povoljnim i prikladnim za modifikaciju po potrebi sustava. Princip implementiranja VPN-a u sustav pokazao se učinkovitim načinom za daljinski pristup sustavu. Upravljanje pomoću SCADA sustava omogućava nadzor sustava, te približava sustav industrijskim standardima. Mogućnost razvoja ulazno – izlaznih modula prema potrebama sustava omogućava maksimalno iskorištenje sredstava. OpenPLC sustav je u stalnom razvoju, te se može očekivati više mogućnosti i podrške za mnoge platforme u budućnosti.

Filip Frančić

_____

U Varaždinu 20.10.2017.

11. Literatura

[1] https://www.raspberrypi.org dostupno 19.09.2017.

[2] https://www.raspberrypi.org/learning/hardware-guide/components/raspberry-pi/ dostupno 19.09.2017.

[3] https://www.element14.com/community/community/raspberry-pi/blog/2017/01/16/raspberry-pi-3-block-diagram dostupno 19.19.2017

[4] http://spvp.zesoi.fer.hr/seminari/2003/sipic/RS-485.htm dostupno 19.09.2017.

[5] https://www.amci.com/industrial-automation-resources/plc-automation-tutorials/what-plc/ dostupno 19.09.2017.

[6] https://moodle.vz.unin.hr/
Automatizacija strojeva i uređaja - Dunja Srpak dipl. ing. el.
dostupno 19.09.2017.

[7] http://www.robot-electronics.co.uk/i2c-tutorial dostupno 19.09.2017.

[8] https://moodle.vz.unin.hr/
Nadzor i vizualizacija tehnoloških procesa - Dunja Srpak dipl. ing. el.
dostupno 19.09.2017.

[9] http://www.its-ltd.co.uk/siemens-wincc.aspx dostupno 19.09.2017.

[10] http://www.simplymodbus.ca/faq.htm dostupno 19.09.2017.

[11] http://www.rfwireless-world.com/Terminology/Modbus-message-frame.html dostupno 19.09.2017

[12] https://technet.microsoft.com/en-us/library/cc731954(v=ws.10).aspx dostupno 19.09.2017.

[13] http://www.openplcproject.com dostupno 19.09.2017.

[14] http://www.pivpn.io dostupno 19.09.2017.

[15] https://www.noip.com dostupno 19.09.2017.

[16] https://openvpn.net dostupno 19.09.2017.

[17] https://github.com/thiagoralves/OpenPLC_v2.git dostupno 19.09.2017.

[18] http://www.microchip.com/wwwproducts/en/ATmega328P dostupno 19.09.2017.

[19] https://github.com/Optiboot dostupno 19.09.2017.

[20] https://www.schukat.com/ dostupno 19.09.2017.

Popis slika

Slika 2.1 Model sustava 3
Slika 3.1 Raspberry pi 3 model B [2] 4
Slika 3.2 Raspberry pi 3 blok dijagram [3] 5
Slika 3.3 Primjer I2C komunikacije [7] 7
Slika 3.4 Sastav modbus poruke [11] 9
Slika 3.5 Modbus tablica 10
Slika 3.6 Funkcijski kodovi 10
Slika 3.7 VPN sustav [12] 11
Slika 4.1 Konfiguracija statičke IP drese 14
Slika 4.2 Postavljanje IP adrese 15
Slika 4.3 Odabir DNS adrese 16
Slika 4.4 Odabir razine enkripcije 17
Slika 4.5 Odabir uređaja za upravljanje 20
Slika 4.6 Primjer pisanja programa u ladder programskom jeziku 24
Slika 4.7 Memorijske lokacije u OpenPLC sustavu 26
Slika 4.8 Modbus konfiguracija 26
Slika 5.1 Arduino IDE okruženje 28
Slika 5.2 Napajanje 30
Slika 5.3 Sklop digitalnih ulaza 32
Slika 5.4 Sklop analognih ulaza 32
Slika 5.5 Izlazni sklop 33
Slika 6.1 Blok dijagram 36
Slika 7.1 PLC program 37
Slika 8.1 Shema kučišta [20] 38
Slika 8.2 Naljepnice 39
Slika 9.1 Primjer SCADA upravljanja 41

Prilozi

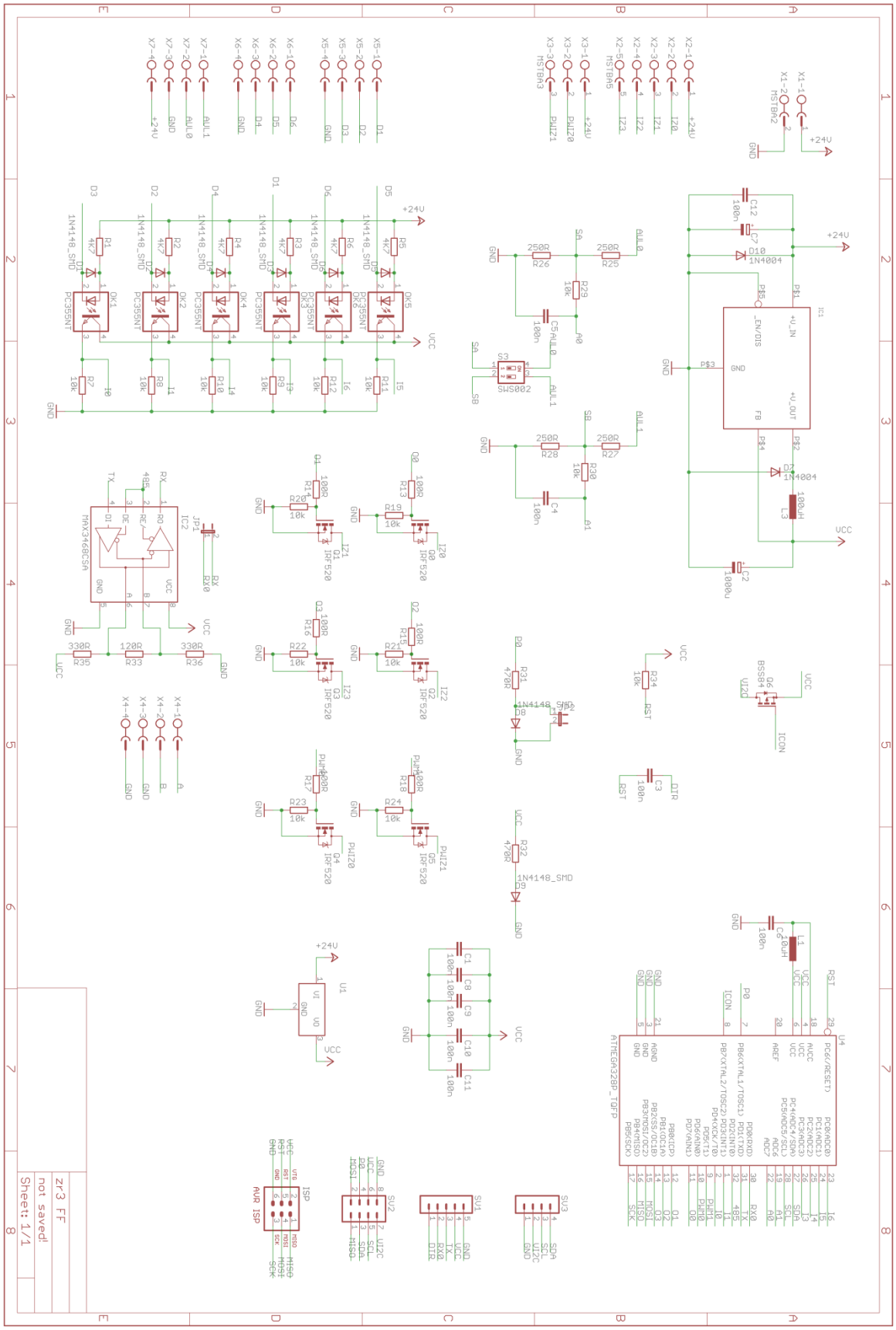
- **Prilog 1: Atmega 328 program**

```
#include <SoftwareSerial.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET PB6
Adafruit_SSD1306 display(OLED_RESET);
#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2
#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH  16
unsigned int ulaz0;
unsigned int ulaz1;
unsigned int izlaz0;
unsigned int izlaz1;
int analog0;
int analog1;
int analog2;
int analog3;
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

#if (SSD1306_LCDHEIGHT != 32)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif

// Modbus konfiguracija
#include <ModbusRtu.h>
#define ID 1
Modbus slave(ID, 0, 2); //
int8_t state = 0;
unsigned long tempus;

// Modbus podaci
uint16_t aul6data[16];

void setup()
{
  io_setup(); // pozivanje ulazno - izlazne konfiguracije

  // početak komunikacije
  Serial.begin(9600); // odabir brzine serijske komunikacije
  slave.begin( 9600 );

  // Konfigurcija I2C porta

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C
  (for the 128x32)
  display.display();
  display.clearDisplay();

}

void loop() {
```

```
71
72      // Povezivanje modbus podataka sa serijskom komunikacijom
73      state = slave.poll( au16data, 16 );
74      io_poll();
75      // displej
76
77      // prikaz displeja
78      // digitalni ulazi
79      display.setCursor(0,0);
80      display.setTextSize(1);
81      display.setTextColor(WHITE);
82      if(digitalRead(4)== HIGH)
83      {display.print("I0 "); }
84      display.setCursor(20,0);
85       if(digitalRead(3)== HIGH)
86      {display.print("I1 "); }
87      display.setCursor(40,0);
88      if(digitalRead(A3)== HIGH)
89      {display.print("I2 "); }
90      display.setCursor(60,0);
91      if(digitalRead(A2)== HIGH)
92      {display.print("I3 "); }
93      display.setCursor(80,0);
94      if(digitalRead(A1)== HIGH)
95      {display.print("I4 "); }
96      display.setCursor(100,0);
97      if(digitalRead(A0)== HIGH)
98      {display.print("I5 "); }
99
100
101     //digitalni izlazi
102     display.setCursor(0,8);
103     if(digitalRead(7)== HIGH)
104     {display.print("Q0 "); }
105     display.setCursor(20,8);
106     if(digitalRead(8)== HIGH)
107     {display.print("Q1 "); }
108     display.setCursor(40,8);
109     if(digitalRead(9)== HIGH)
110     {display.print("Q2 "); }
111     display.setCursor(60,8);
112     if(digitalRead(10)== HIGH)
113     {display.print("Q3 "); }
114
115     //analogni ulazi
116     display.setCursor(0,16);
117     display.print("AD0= ");
118      analog0 = analogRead(A7) / 10;
119      analog0 = analog0 / 1.02;
120     display.print(analog0);
121      display.print("%");
122     display.setCursor(80,16);
123     display.print("AD1= ");
124      analog1 = analogRead(A6) / 10;
125      analog1 = analog1 / 1.02;
126     display.print(analog1);
127     display.print("%");
128
129     //analogni izlazi
130     display.setCursor(0,24);
131     display.print("QA= ");
132      analog2 = aizlaz0 / 10;
133      analog2 = analog2 / 1.02;
134     display.print(analog2);
135     display.print("%");
136     display.setCursor(80,24);
137     display.print("QB= ");
138      analog3 = aizlaz1 / 10;
139      analog3= analog3 / 1.02;
140   display.print(analog3);
141   display.print("%");
```

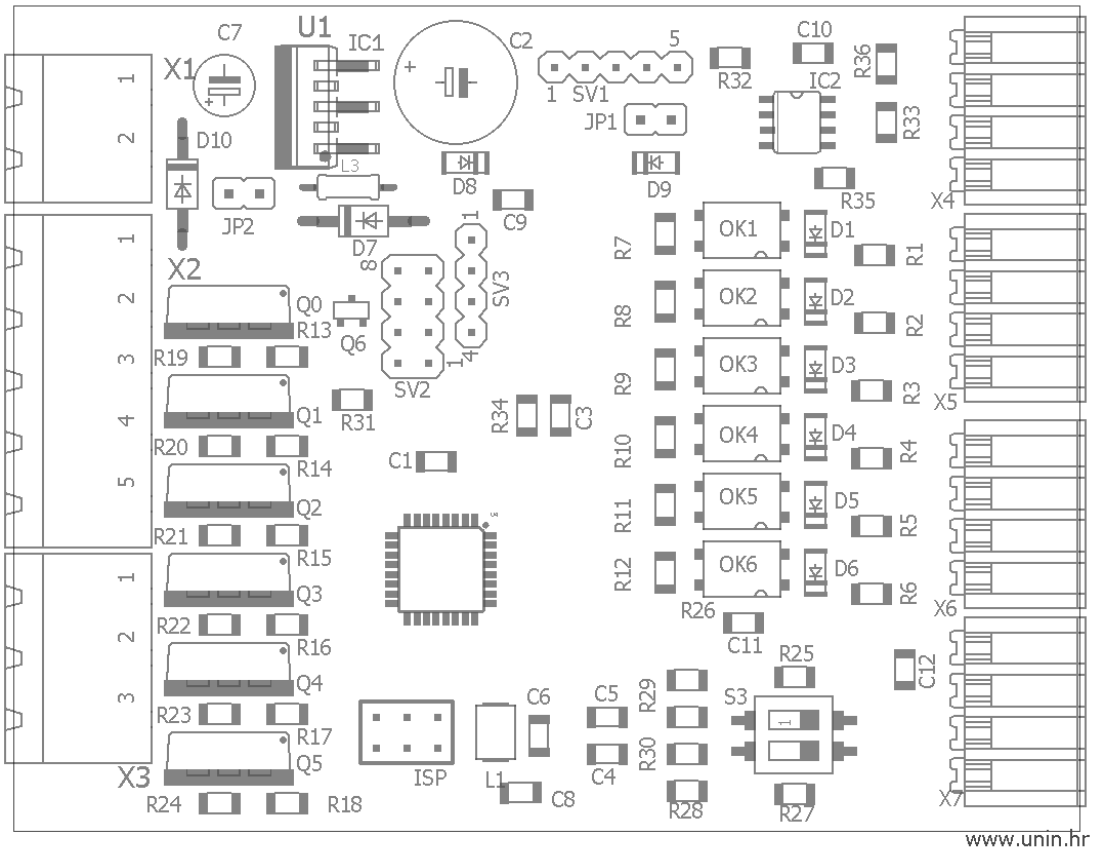
```
142
143    display.display();
144    display.clearDisplay();
145   }
146   void io_setup() {
147     // Konfiguracija ulazno izlaznih portova
148     pinMode(A0, INPUT);
149     pinMode(A1, INPUT);
150     pinMode(A2, INPUT);
151     pinMode(A3, INPUT);
152     pinMode(3, INPUT);
153     pinMode(4, INPUT);
154
155     pinMode(7, OUTPUT);
156     pinMode(8, OUTPUT);
157     pinMode(9, OUTPUT);
158     pinMode(10, OUTPUT);
159
160     DDRB |= (1 << DDB6) | (1 << DDB7);
161     PORTB |= (1 << PORTB7);
162   }
163
164
165     // Veza između ulazno - izlaznih pinova i modbus podataka
166
167   void io_poll() {
168     // digitalni ulazi
169     bitWrite( aul6data[0], 0, digitalRead( 4 ));
170     bitWrite( aul6data[0], 1, digitalRead( 3 ));
171     bitWrite( aul6data[0], 2, digitalRead( A3 ));
172     bitWrite( aul6data[0], 3, digitalRead( A2 ));
173     bitWrite( aul6data[0], 4, digitalRead( A1 ));
174     bitWrite( aul6data[0], 5, digitalRead( A0 ));
175
176     // digitalni izlazi
177     digitalWrite( 7, bitRead( aul6data[1], 0 ));
178     digitalWrite( 8, bitRead( aul6data[1], 1 ));
179     digitalWrite( 9, bitRead( aul6data[1], 2 ));
180     digitalWrite( 10, bitRead( aul6data[1], 3 ));
181
182
183     // analogni ulazi
184     aulaz0 = analogRead( A7 );
185     aulaz1 = analogRead( A6 );
186
187     //povezivanje sa modbus podacima
188     aul6data[4] = aulaz0;
189     aul6data[5] = aulaz1;
190
191     // analogni izlazi
192     aizlaz0 = aul6data[2];
193     aizlaz1 = aul6data[3];
194
195     //povezivanje sa modbus podacima
196     analogWrite( 6, aizlaz0);
197     analogWrite( 5, aizlaz1);
198
199   }
200
```

- **Prilog 2: Električna shema pločice**

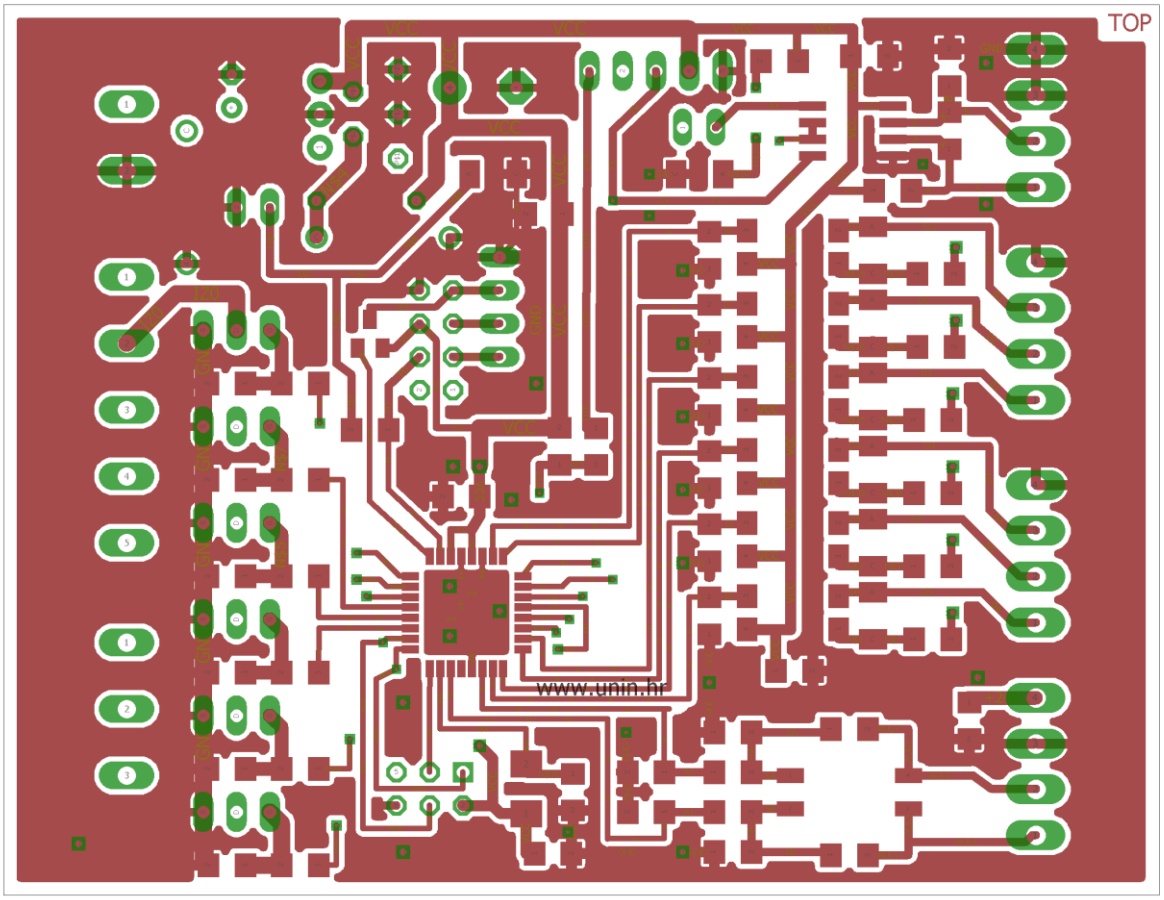


- **Prilog 3: Dizajn pločice**

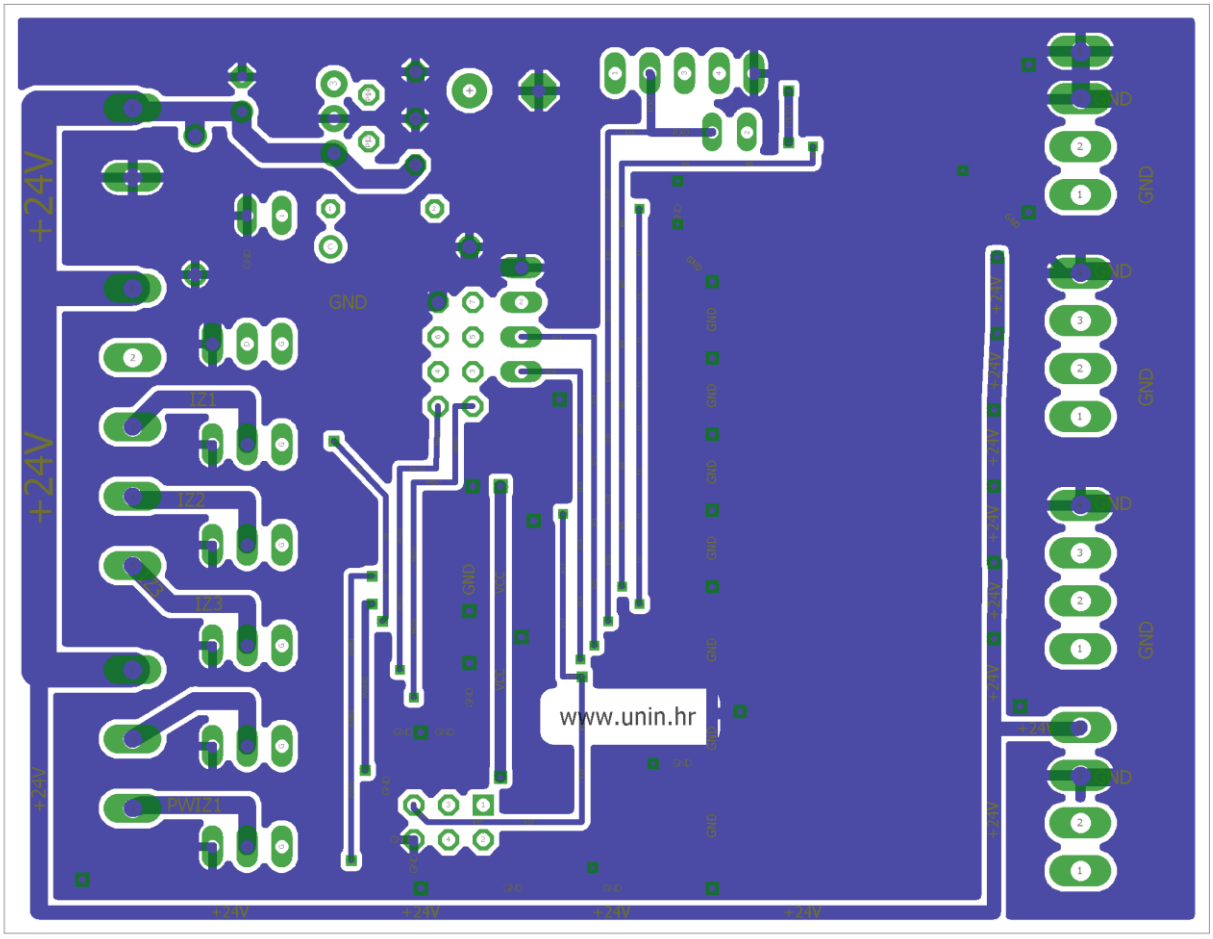
Razmještaj elemenata:



Gornji dio pločice:



Donji dio pločice



Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

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

Ja, Filip Frančić pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor završnog rada pod naslovom Realizacija sustava upravljanja primjenom OpenPLC platforme te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student:
(Filip Frančić)

Frančić Filip
(vlastoručni potpis)

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

Ja, Filip Frančić neopozivo izjavljujem da sam suglasan s javnom objavom završnog rada pod naslovom Realizacija sustava upravljanja primjenom OpenPLC platforme čiji sam autor.

Student:
(Filip Frančić)

Frančić Filip
(vlastoručni potpis)