Razvoj web aplikacije za upravljanje PLC-om

Jasnić, Josip

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

Razvoj web aplikacije za upravljanje PLC-om

Završni rad

Josip Jasnić

Osijek, 2022.

Sadržaj

1. UVOD 1
1.1. Zadatak završnog rada 1
2. TEORIJSKA PODLOGA 2
2.1. Programirljivi logički kontroler (PLC) 2
2.1.1. Siemens 3
2.1.4. Komunikacija 4
2.2. Programski jezici za izradu web stranica 6
3. KORIŠTENI ALATI I UREĐAJI 7
3.1. SIMATIC S7-1200 7
3.1.2. Signalni modul SM 1223 DI16/DQ16 x relay 8
3.1.2. Napajanje 9
3.2. TIA Portal 10
3.3. Ljestvičasti dijagram 11
3.4. Maketa dizala 13
4. IZRADA PROGRAMA I VIZUALIZACIJA 14
4.1. Opis zadatka 14
4.2. Postavljanje web servera 14
4.2.1. Čitanje vrijednosti 16
4.2.2. Mijenjanje vrijednosti 18
4.3. Izrada programa 18
4.4. Vizualizacija pomoću web sučelja 19
5. ZAKLJUČAK 21
6. LITERATURA 22
Sažetak 23
Abstract 23
Životopis 24
Prilog 25
Prilog 1. Index.html 25
Prilog 2. IO.html 29

1. UVOD

U radu je opisano upravljanje dizalom pomoću programirljivog logičkog kontrolera. Radom programiljivog logičkog kontrolera (PLC) će se upravljati preko web sučelja. Dizalo je prijevozno sredstvo koje služi za prijevoz ljudi i tereta, a može se kretati samo po vertikalnoj osi. Radom dizala upravlja PLC koji prima zahtjeve od korisnika te obrađuje podatke senzora, te prosljeđuje naredbe aktuatorima. Dizalo će raditi tako da će po pozivu ići na pojedini kat, a ukoliko nema poziva, ostat će na zadnjem mjestu na koje je bilo pozvano. Za stimulaciju rada sustava biti će korištena maketa dizala koja se nalazi u zgradi fakulteta.

1.1. Zadatak završnog rada

Potrebno je razviti web aplikaciju koja će demonstrirati mogućnosti upravljanja rada PLC-a putem web sučelja.

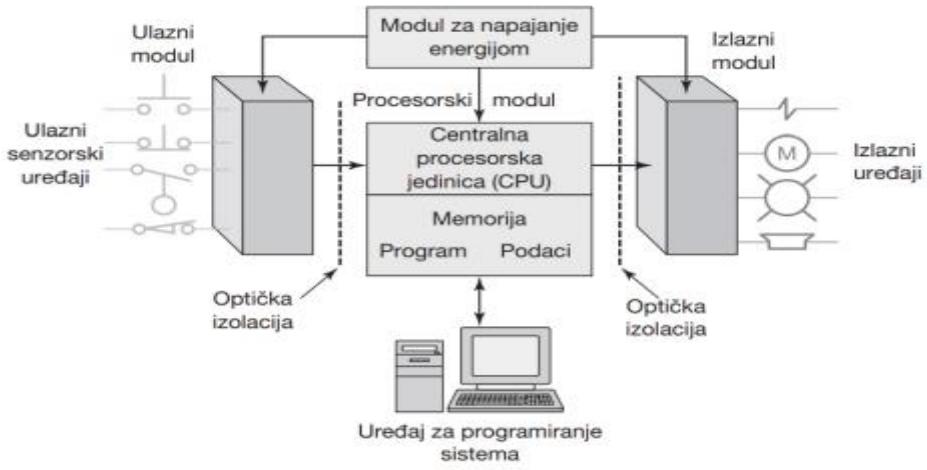
2. TEORIJSKA PODLOGA

2.1. Programirljivi logički kontroler (PLC)

Programirljivi logički kontroler (eng. programmable logic controller, PLC) je industrijski kontroler zasnovan na mikroprocesoru koji je zamijenio relejnu tehnologiju. Relej je elektromehanički prekidač koji se sastoji od elektromagneta, armature, opruge i električkih kontakata. Kada se na oprugu dovede energija, magnetska sila povlači prekidač u stanje upaljeno ili gašeno. Nestankom sile prekidač se vraća u početno stanje. Upravljanje procesa bilo je izvedeno pomoću releja. Budući da je ožičenje komplicirano, promjene programa, tj. funkcije ili nadogradnje su bile teško ostvarive. Iz tog razloga se u General Motorsu javila potreba za unaprjeđenjem upravljanja procesa. Zahtijevali su da novi sustav može funkcionirati u industrijskom okruženju, da bude jednostavno modificirati ulazne i izlazne jedinice te da ga električari u pogonu mogu lako održavati i mijenjati. Prvi PLC uređaji pojavili su se krajem 1960ih. Imali su iste mogućnosti kao i releji, ali ih se moglo reprogramirati, iskoristiti na drugom mjestu i bili su pouzdaniji. Prednosti PLC-a su fleksibilnost, brz odziv, modularan dizajn, lako povezivanje, manja cijena, jednostavnije ožičenje, programiranje i otklanjanje kvarova. Dizajniran je za rad u realnom vremenu, s velikim brojem ulaznih i izlaznih modula, u teškim uvjetima kao što su visoka ili niska temperatura, vlaga i prašina.

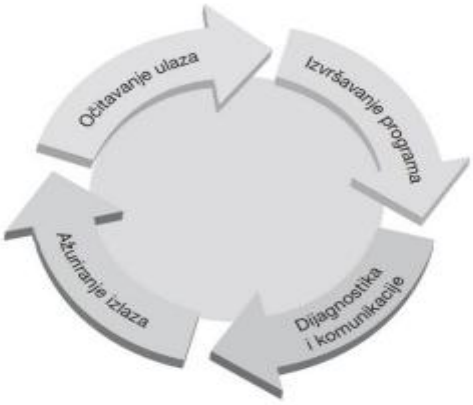
Budući da su bili direktna zamjena za relejnu tehnologiju, bilo je potrebno osmisliti programski jezik koji će omogućiti što lakši prijelaz na novu tehnologiju. Iz tebe potrebe je nastala ljestvičasta logika (engl. ladder logic, LAD). Kasnije su u upotrebu ušli i drugi načini programiranja, između ostalog su to funkcijski blokovi (engl. Function Block Diagram, FBD), strukturirani tekst (engl. structured Text, ST) te sekvencijalni dijagram toka (engl. Sequential Flow Charts, SFC).

Programirljivi logički kontroleri se sastoje od centralne procesne jedinice (engl. central processing unit, CPU), napajanja, memorije, modula za komunikaciju te ulaznih i izlaznih modula. CPU je mozak PLC-a. CPU se sastoji od mikroprocesora koji izvodi program i komunicira s ostalim modulima. Za rad procesora je potrebna memorija kako bi mogao spremiti korisnički program, izračunate vrijednosti te pratiti stanja na ulazu i izlazu.



Slika 2.1.: Prikaz dijelova PLC-a

Korisnički program se izvršava kao dio ponavljajućeg ciklusa. Ciklus počinje čitanjem stanja na ulazima PLC-a te se zatim izvršava program u CPU. Nakon toga se obavljaju poslovi dijagnostike i komunikacije, te se zatim ažuriraju vrijednosti na izlazima PLC-a. Ulazno/izlazni moduli su sučelje preko kojih PLC komunicira sa vanjskim svijetom. Pojam "vanjski svijet" se koristi kako bi se razlikovali fizički uređaji kojima PLC upravlja od onih programskih koji imaju ulogu releja ili brojača u samom programu.



Slika 2.2.: Prikaz cikličkog izvršavanja programa [4]

2.1.1. Siemens

Kako je spomenuto ranije, PLC uređaj dizajniran je za potrebe General Motorsa, a dizajnirala ga je američka kompanija Modicon, zajedno s komunikacijskim protokolom Modbus. Paralelno s njima, svoj rad na PLC-ima započela je također još jedna američka kompanija,

Allen-Bradley te Njemačka tvrtka Siemens u Europi. Danas u svijetu postoji niz proizvođača PLC-a, kao što su ABB, Bechoff, Bosch, Eaton, Rexroth i mnogi drugi .

Slika 2.3.: Modularni Siemens S7-1200

Tijekom godina, Siemens je razvio nekoliko serija PLC-a, među kojima se izdvajaju SIMATIC S3, SIMATIC S5 i SIMATIC S7.[4] Glavna karakteristika SIMATIC S3 je prijelaz s upravljanja procesom pomoću ožičenja na upravljanje pomoću programa. SIMATIC S7 serija je prvi put predstavljena 1994. godine, a sastojala se od tri vrste uređaja, S7-200, S7-300 i S7-400, a danas su oni zamijenjeni s S7-1200 i S7-1500. Zajedno sa SIMATIC S7 predstavljen je i Profibus. S7-200 i S7-1200 pogodni su za upravljanje manjim sustavima, dok su S7-400 i S7-1500 namijenjeni upravljanju velikih postrojenja.

2.1.4. Komunikacija

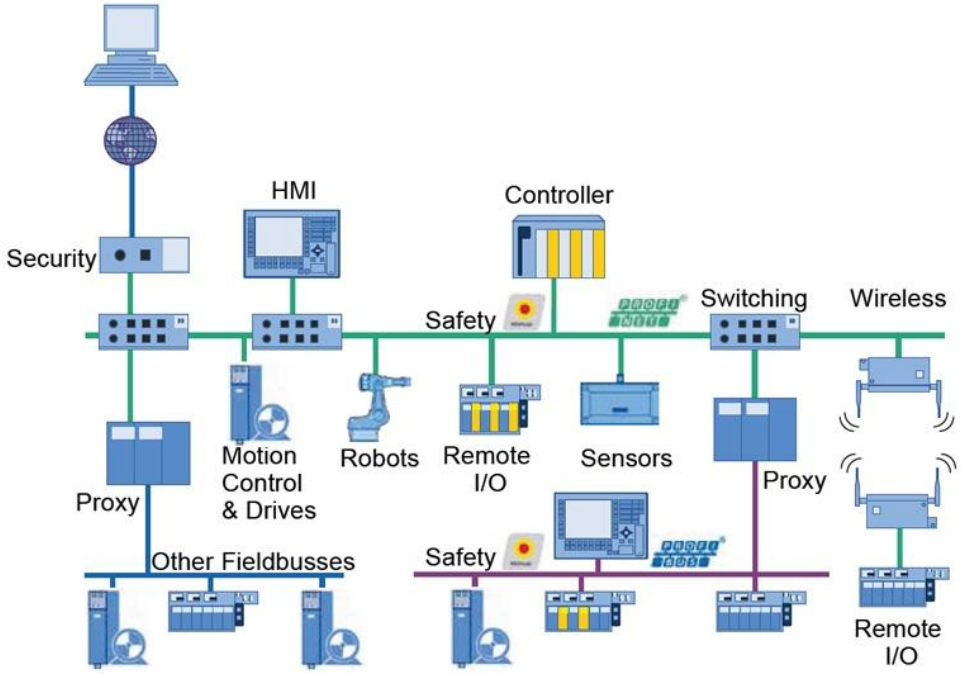
Načini povezivanja i komunikacije PLC-a su mijenjali tijekom godina jer je trebalo povezivati sve više uređaja. U početku je svaki proizvođač imao svoju tehnologiju za komunikaciju te je bilo teško povezati uređaje različitih proizvođača. Danas je situacija ipak malo drugačija i koristi se svega nekoliko protokola, a među najčešće korištenima su Modbus, Ethernet, Profibus i Profinet.

Modbus je prvi protokol koji se koristio za komunikaciju PLC-a, a funkcionirao je na principu master/slave. U ovom obliku komunikacije master uređaj može pisati i čitati iz registara slave uređaja te inicirati zahtjeve, a slave može samo čekati te zahtjeve i odgovarati na njih. RS-232 je omogućavao komunikaciju između dva uređaja, jedan master i jedan slave, na udaljenosti do 15 metara uz brezinu 1Mbs. Uvođenje RS-485 omogućilo je rad s više slave uređaja i na

udaljenostima do 1200 metara, a dodavanjem pojačivača moguće je postići da jedan glavni uređaj upravlja s do maksimalno 247 slave uređaja, uz povećanje brzine na približno 10Mbs.

PROFINET je Ethernet standard za automatizaciju koji omogućava siguran prijenos podataka velikom brzinom prijenosa, koja iznosi 100Mbs. Za razliku od Modbus protokola koji se temelji na master/slave odnosu, PROFINET koristi konzument/pružatelj(eng. consumer/provider) odnos, što znači da svaki uređaj može biti i konzument i pružatelj, npr. PLC u ulozi pružatelja šalje informacije IO uređajima, a u ulozi konzumenta prima podatke s IO uređaja[7]. Broj uređaja koji se mogu spojiti na mrežu je praktički neograničen. PROFINET kablovi koji se koriste su prilagođeni industrijskom okruženju, kao i sam PLC, da mogu podnijeti loše radne uvjete bez kompromitiranja podataka. Korištenje PROFINET-a omogućava pristup uređaju koristeći običan internetski preglednik.

PROFIBUS se koristi za brzi ciklički prijenos podataka. Koriste se dva protokola, PROFIBUS PA i PROFIBUS DP. PROFIBUS DP se zasniva na RS-485, a koristi se u industrijama koje zahtijevaju visoku brzinu prijenosa, ali ne i zaštitu od eksplozija.[12] Brzina prijenosa se može konfigurirati da iznosi između 9.6 kb/s i 12 Mb/s. PROFIBUS PA predstavlja poboljšanje PROFIBUS DP jer zamjenjuje konvencionalne signale 4-20 mA i HART te je otporan na eksplozije. Brzina prijenosa podataka je puno manja, 31 kb/s, što omogućava veću udaljenost između uređaja.



Slika 2.4.: Povezivanje uređaja

2.2. Programski jezici za izradu web stranica

HyperText Markup Language (HTML), Cascading Style Sheets (CSS) i JavaScript su jezici kojima se kreiraju web stranice. HTML je jezik koji koristi označavanje korištenjem oznaka (eng. tag), kojima se strukturiraju elementi HTML dokumenta. Oznake govore internet pregledniku kako prikazati tekst koji slijedi. HTML se sastoji od elemenata i atributa. Elementi su blokovi koji korištenjem oznaka identificiraju dijelove HTML stranice. Prva oznaka u HTML dokumentu je <html> koja internet pregledniku govori da slijedi početak HTML dokumenta, sve do oznake </html> koja označava kraj HTML dokumenta. Atributi daju podatke o određenoj instanci elementa te tako omogućavaju raznovrsnost opisa sadržaja. Na slici 2.5. je prikazan element <img/> koji govori internet pregledniku da se radi o slici, a src atribut gdje se ta slika nalazi. Atributi width i height određuju prikaz slike na stranici, dok alt označava tekst koji će se prikazati ako se slika ne bude mogla učitati. Dizajniranje stranice pomoću HTML je vrlo ograničeno te moguće koristiti tablice, određivati font te stil teksta.

```
<img src="siemens_Logo.png" width="464px" height="75px" alt="SIEMENS"/>
```

Slika 2.5.: HTML atributi i elementi

CSS, odnosno kaskadni stilovi, je donio niz novih mogućnosti u dizajniranju stranica. Postalo je puno lakše dodavanje stilova, fontova i boja, što je HTML kod učinilo preglednijim i manjim, a samim time i lakšim za kontroliranje. CSS omogućuje podešavanje pozadine, kontroliranje veličine i pozicije elemenata, uređivanje margina i tablica. Sintaksa CSS-a se sastoji od selektora i deklaracije. Selektor određuje koji HTML element želimo mijenjati, a u deklaraciji se navode svojstva i pridjeljuju im se vrijednosti. Na slici 2.6 je prikazano oblikovanje za element <body/> unutar HTML-a. Tekst napisan unutar tog elementa će biti ispisani plavom bojom na sredini stranice.

```
body {
  color: blue;
  text-align: center;
}
```

Slika 2.6.: CSS

JavaScript je programski jezik namijenjen razvoju interaktivnih HTML stranica. Dodavanje JavaScripta u statički HTML omogućava se interakcija s korisnikom i dinamički prikaz sadržaja. Tako je moguće kreiranje sadržaja koji će se aktivirati tek kada se mišom pređe preko njega, a pri tome se stranica ne mora ponovo učitavati.

3. KORIŠTENI ALATI I UREĐAJI

3.1. SIMATIC S7-1200

Serija S7-1200 sastoji se od 5 CPU modela, a to su 1211C, 1212C, 1214C,1215C i 1217C. Arhitektura samog PLC-a varira ovisno o tipu CPU-a. Primjer arhitekture biti će prikazan na CPU 1214C, koji će se koristiti za izradu ovoga rada. U tablici 3.1. nalaze se osnovne tehničke karakteristike CPU 1214C.[5] PLC uređaji serije S7-1200 sastoje se od modula koji se mogu međusobno spajati, kao što je prikazano na slikama 2.3. i 3.1. Ispred centralne procesorske jedinice ili CPU-a nalaze se komunikacijski moduli koji se koriste za serijsko povezivanje s ostalim uređajima. Zatim dolazi centralna procesorska jedinica ili CPU modul.

Slika 3.1.: PLC u upravljačkom ormaru

Na njemu se nalaze LED diode za signalizaciju stanja CPU-a, ulaza i izlaza i konektori za spajanje signala, napajanja te PROFINET konektor. CPU moduli se razlikuju po broju ulaza i izlaza, po naponu napajanja i vrsti izlaznog signala. Nakon CPU-a se dodaju signalni moduli koji služe za povećanje broja ulaza i izlaza te za prilagodbu procesnih signala. Broj signalnih modula se određuje prema složenosti projekta, te je tako za potrebe ovog rada korišten samo jedan signalni modul i to 6ES7223-1PL32-0XB0 koji se sastoji od 16 digitalnih ulaza i 16 digitalnih izlaza[6].

Dimenzije (mm)	110x100x75
Napajanje	24V
Radna memorija	50kb
Podatkovna memorija	2MB
Trajna memorija	50kb
Digitalni ulazi	14
Digitalni izlazi	10
Analogni ulazi	2

Tablica 3.1.: Osnovne tehničke karakteristike CPU 1214C

S7-1200 također ima ugrađen web server. Web server prihvaća zahtjeve poslane iz internet preglednika i zatim mijenja vrijednosti varijabli te tako upravlja s radom sustava. Prije početka korištenja mora se omogućiti njegovo korištenje u postavkama. Nakon što je web server aktiviran, unosom IP adrese PLC uređaja u internet preglednik možemo pristupiti samom uređaju. Tamo možemo vidjeti osnovne podatke o uređaju, o spojenim modulima, stanju varijabli, vršiti dijagnostiku te učitati korisničke stranice i mijenjati PLC tagove. Za korištenje opcije web servera potrebno ga je pozvati u samom projektu pomoću funkcije WWW, te poslati projekt na PLC. Funkcija WWW obrađuje poslane podatke iz preglednika i usklađuje ih s bazom podataka. Korisničke stranice moguće je razvijati pomoću HTML-a, CSS-a i Javascripta, o čemu će više biti rečeno u narednim poglavljima.

3.1.2. Signalni modul SM 1223 DI16/DQ16 x relay

Ako je za upravljanje procesom potrebno više ulaza ili izlaza nego što ih ima glavni modul dodaju se signalni moduli. Pomoću signalnog modula povećava se broj ulaza i izlaza PLC-a bez potrebe da se nabavlja novi. Najčešće se dijele na module za digitalne ulaze ili izlaze te analogne ulaze ili izlaze. Korišteni modul se sastoji od 16 digitalnih ulaza te 16 digitalnih izlaza izvedenih pomoću releja. Također, svaki signalni modul se mora napajati.

Slika 3.2.: Signalni modul SM 1223 DI16/DQ16

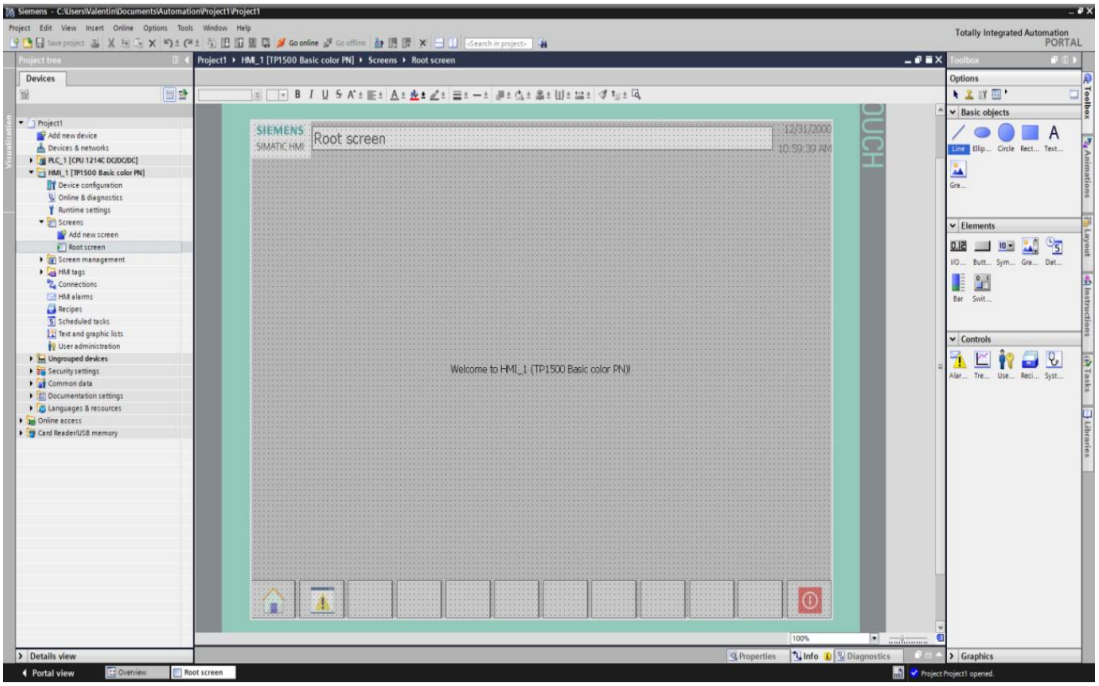
3.1.2. Napajanje

Napajanje je ključan dio za rad električnih uređaja. Modul napajanje je neosjetljiv na smetnje električne mreže te na kraće ispade. Na ulaz napajanja dovodimo 120 ili 230 VAC, a kao izlaz dobivamo 24 VDC koji se koristi za napajanje PLC-a te signalnih modula.

Slika 3.3.: Napajanje

3.2. TIA Portal

Totally Integrated Automation Portal ili TIA Portal je razvojno okruženje za upravljanje proizvodima linije SIMATIC S7. Budući da TIA Portal objedinjuje sve podatke u jedan projekt, moguće je u istom sustavu raditi PLC program i vizualizaciju. Za programiranje samog PLC-a koristi se STEP7, a za izradu vizualizacije WinCC. Vizualizacija omogućava praćenje stanja sustava te zadavanje naredbi sustavu. Kreiranje tog prikaza vrši se po principu "drag and drop" te se zatim u postavkama dodanog elementa namještaju željene opcije. Na slici 3.4. prikazano je sučelje za izradu vizualizacije.

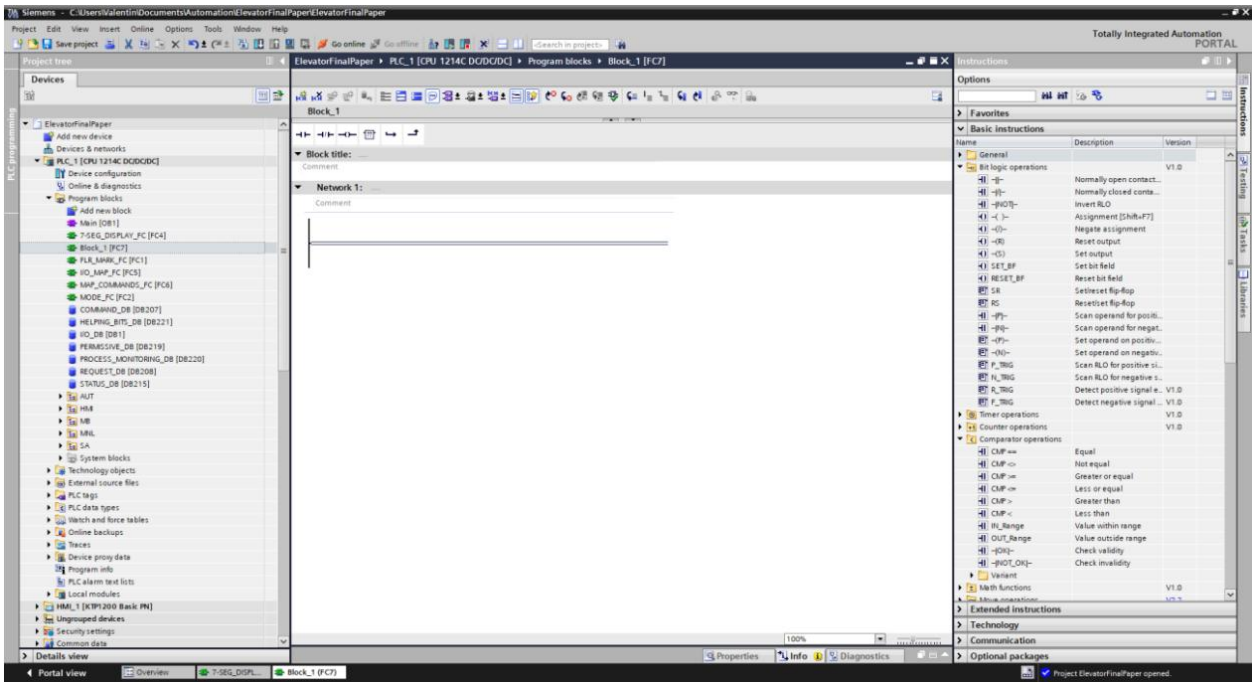


Slika 3.4.: WinCC sučelje

Projektni način rada, prikazan na slici 3.5., koristi se za razvoj programa. U njemu je moguće stvarati nove blokove te su prikazani svi napravljenih blokova. TIA Portal koristi nekoliko blokova.

- ✧ Organizacijski blok (OB) služi za strukturu programa, tj. kao poveznica korisničkog programa i operacijskog sustava.
- ✧ Funkcijski blok (FB) koristi parametre podatkovnog bloka te izvodi određene naredbe.
- ✧ Funkcija (FC) se najčešće koristi za izvođenje naredbi, ali za razliku od funkcijskog bloka bez spremanja podataka u memoriju.

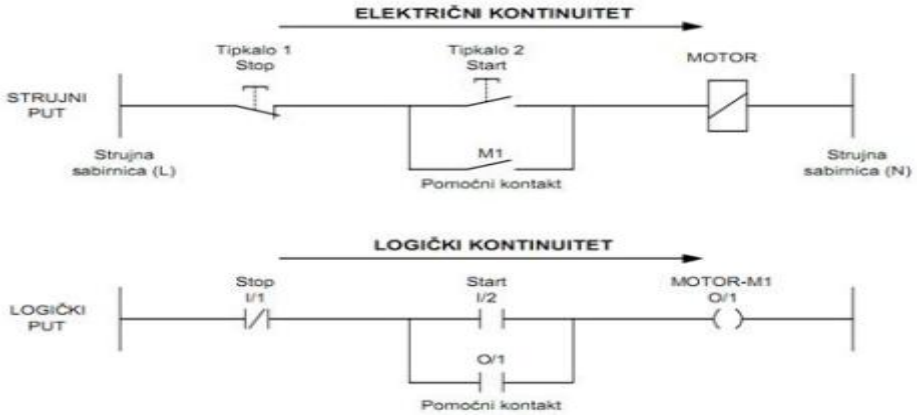
- ✧ Podatkovni blok (DB) služi za pohranu podataka za blokove koda. Globalnom podatkovnom bloku mogu pristupiti svi blokovi.
- ✧ U projektnom stablu se nalazi kategorija Tablica oznaka (eng. PLC tags). U nju se upisuju adrese ulaza i izlaza, njihove oznake te komentari.



Slika 3.5.: Projektni način rada (eng. Project View)

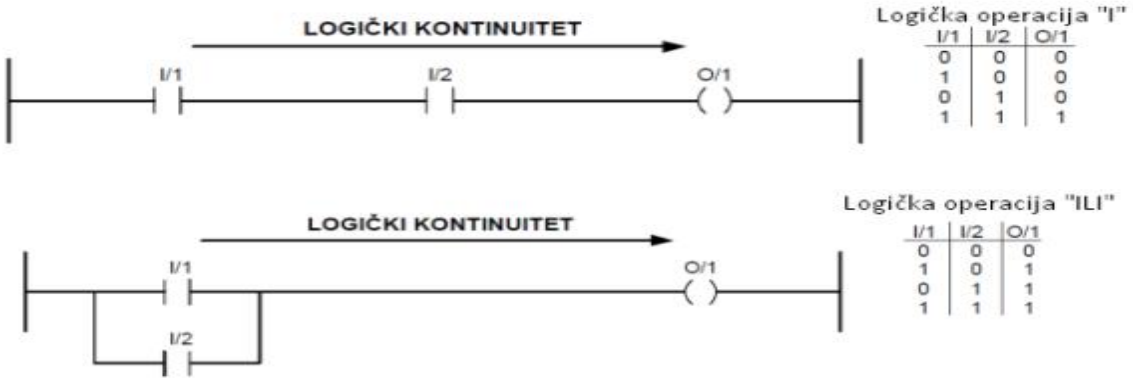
3.3. Ljestvičasti dijagram

Kao što je već navedeno, ljestvičasti dijagram nastao je kao zamjena za relejne sheme kako bi se omogućio što lakši prijelaz na novu tehnologiju, a naziv je dobio zbog svog izgleda, odnosno linija koje podsjećaju na ljestve.



Slika 3.6.: Usporedba strujne sheme i ljestvičastog dijagrama

Kao što se vidi na slici 3.6., usporedbom ljestvičastog dijagrama i strujne sheme vide se mnoge sličnosti. U shemi strujnog kruga simboli označavaju uređaje i njihovo stanje, a u ljestvičastom dijagramu se ispituje stanje tih uređaja. Programiranje ovim jezikom se može svesti na kombinacije logičkih naredbi "I" i "ILI". Kombinacijom ove dvije logičke operacije moguće je izvesti cijelu Boolovu algebru. Svaki logički put ima minimalno jednu izlaznu naredbu, a da bi se ona izvršila svi uvjeti prije nje moraju biti ispunjeni, neovisno u kojoj grani, kao što se vidi na slici 3.7., gdje se vidi primjer logičkih naredbi "I" i "ILI".



Slika 3.7.: Prikaz serijskog i paralelnog spoja u ljestvičastom dijagramu

Na slici 3.8. prikazani su osnovni simboli ljestvičastog dijagrama prema IEC 1131-3 standardu. Normalno otvoren ili radni kontakt(eng. NO contact ili normaly open) ne provodi struju kada je sklopnik otpušten, tj. u stanju je logičke "0", a kada se sklopnik zatvori, zatvara se strujni krug te prelazi u stanje logičke "1". Mirni ili normalno zatvoren kontant (eng. NC contact ili normaly closed) provodi struju dok je sklopnik otpušten, tj. u stanju je logičke "1" kada je strujni krug otvoren, a kada se strujni krug zatvori prelazi u stanje logičke "0". Zavojnica (eng. Coil) predstavlja bilo koji izlazni uređaj ili bit memorije koji se aktivira kada je ispunjena programska logika grane u kojoj se nalazi.

Slika 3.8.: Osnovni simboli ljestvičastog dijagrama

3.4. Maketa dizala

Maketa koja se nalazi na fakultetu. Na postolju makete se nalazi vertikalni stup te upravljačka ploča koja simulira upravljačku ploču pravog dizala. Maketa ima četiri kata, a svaki kat je označen induktivnim senzorima (S1, S2, S3, S4). Svaki kat ima vrata koja pokreće servomotor (V1, V2, V3, V4). Vrata se otvaraju na pet sekundi te se zatvaraju. Na jedan digitalni ulaz serijski su spojeni svi senzori vrata, a ulaz je u logičkoj jedinic ukoliko su sva vrata zatvorena. Pored svakih vrata nalaze se tipkala (T1, T2, T3, T4) s pripadajućom svjetlosnom diodom (D1, D2, D3, D4). 7-segmentni pokaznik na vrhu vertikalnog stupa je već kodiran te mu je samo potrebno predati u binarnom obliku na kojem se katu nalazi dizalo. Na upravljačkoj ploči nalaze se tipkala(T11, T22, T33, T44) i svjetlosne diode koje su serijski spojene sa svjetlosnim diodima na vertikalnom stupu. Na upravljačkoj ploči se nalazi i tipkalo za alarm i pripadajuća svjetlosna dioda. Svjetlosne diode za prikaz kata su zelene boje, a za alarm crvene boje.

Slika 3.9.: Prikaz makete dizala

4. IZRADA PROGRAMA I VIZUALIZACIJA

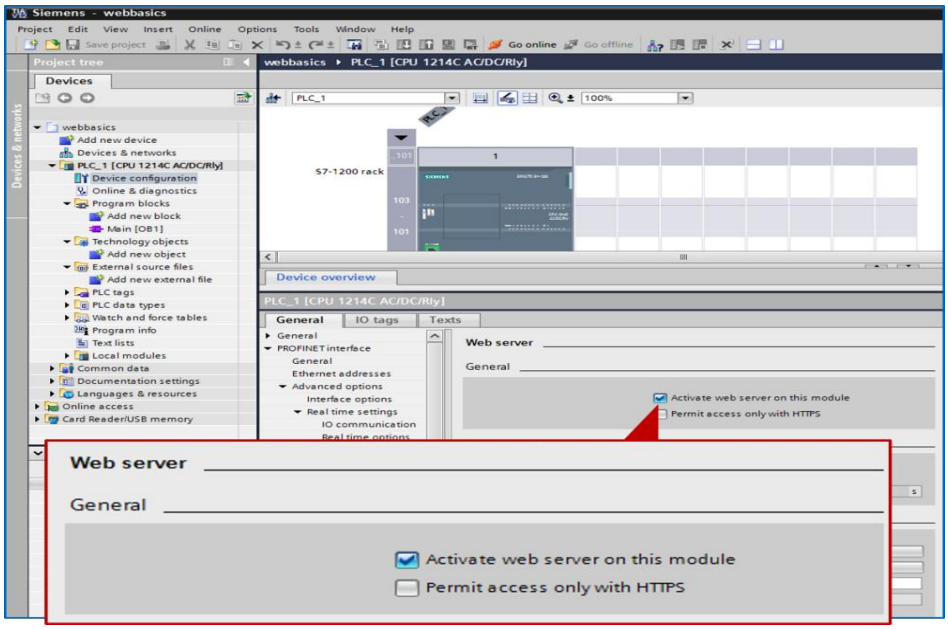
4.1. Opis zadatka

Tipkalo TA inicijalizira dizalo spuštanjem kabine dizala na prvi kat. Omogućiti pozivanje kabine dizala na bilo koji kat iz kabine, s upravljačke ploče ili preko web sučelja. LE diode odnosno promjena boje na web sučelju signaliziraju na koji kat je kabina pozvana. Dolaskom kabine na određeni kat vrata se otvaraju. Ukoliko se pritisne tipkalo alarma (TA) tijekom normalnog rada, potrebno je oglasiti alarm (DA), spustiti kabinu dizala na prvi kat te otvoriti vrata dizala. Ponovni rad dizala omogućiti tek kada se alarm poništi tipkalom TA.

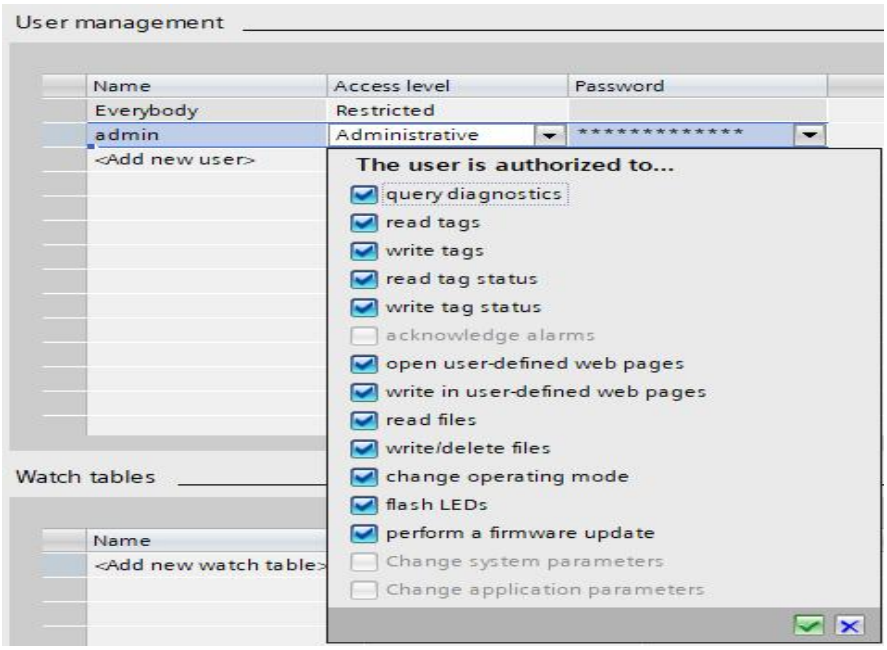
Pritiskom na tipkalo alarma (TA) tijekom normalnog rada potrebno je

4.2. Postavljanje web servera

Za korištenje web servera potrebno ga je prvo omogućiti u postavkama uređaja, što je prikazano na slici 4.1. Također je potrebno dodati korisničko ime za prijavu, omogućiti izmjene varijabli putem web preglednika, što je prikazano na slici 4.2., te zatim poslati konfiguraciju na PLC. Nakon toga se pomoću IP adrese uređaja može pristupiti Siemens PLC web serveru.

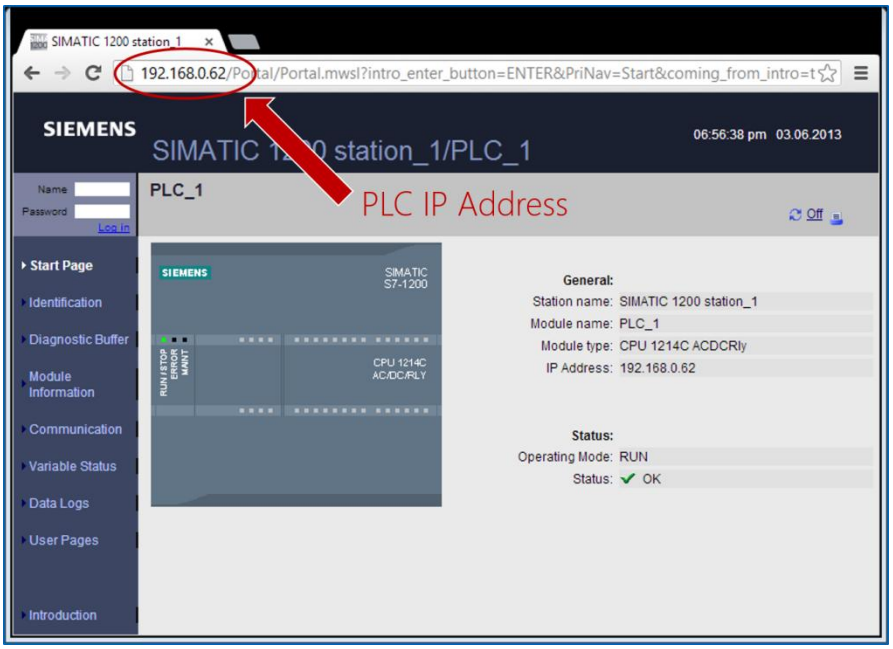


Slika 4.1.: Aktivacija web servera



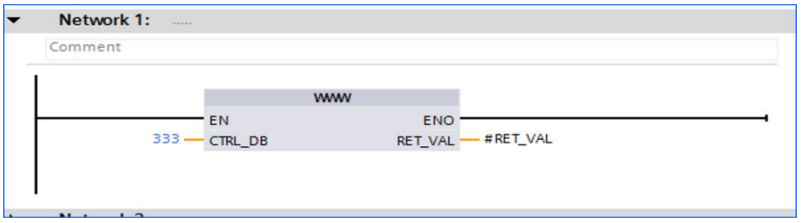
Slika 4.2.: Dodavanje korisnika

Na stranici se mogu dobiti razni korisni podaci, kao na primjer u kartici dijagnostički spremnik(eng. Diagnostic Buffer) gdje se nalaze podaci o radu PLC-a. Na stranici je također moguće nadgledati promjenu vrijednosti ili mijenjati PLC tagove.



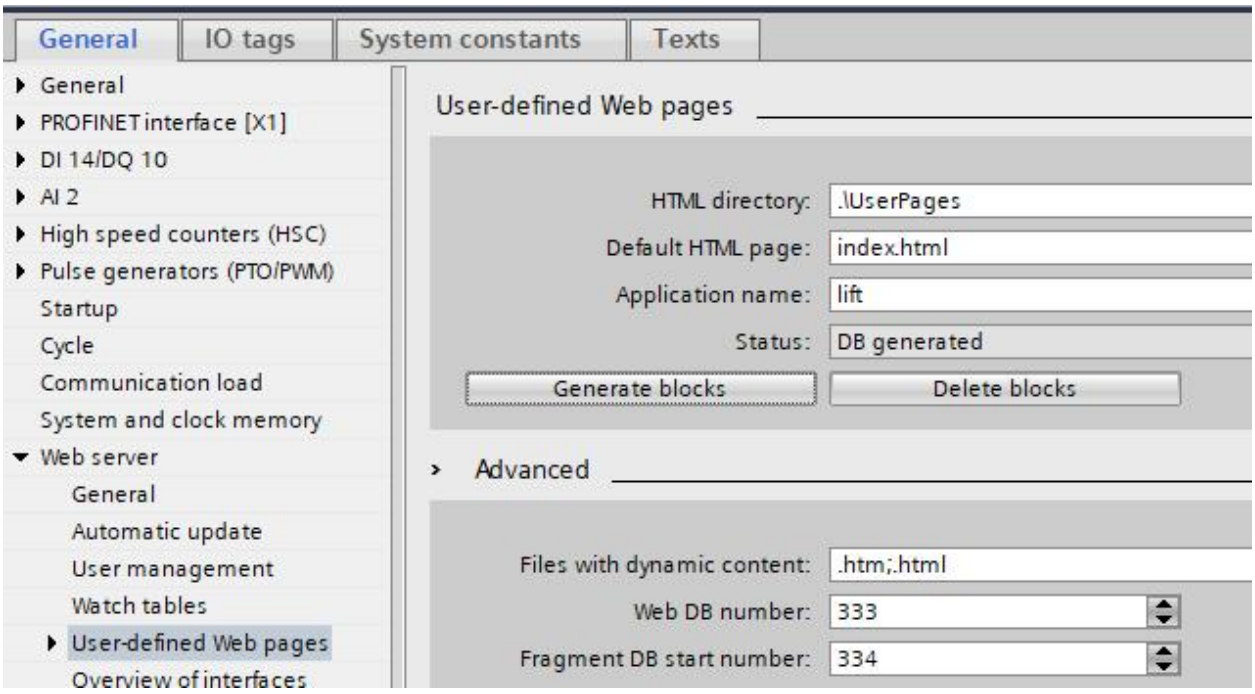
Slika 4.3.: Početna stranica web servera

Za izradu korisničkih stranica prvo je potrebno dodati blok WWW, koji je prikazan na slici. Ovaj blok obrađuje zahtjeve s web preglednika i dohvaća tražene podatke.



Slika 4.4.: WWW blok

Zatim je potrebno u postavkama postaviti zadati HTML direktorij i zadanu HTML stranicu te generirati DB-ove. Postavljanje se vrši tako da kliknemo "*Device configuration* → *Web server* → *User-defined Web pages*" te odaberemo željene postavke, kao što je prikazano na slici 4.5. Generiranje DB-ova je potrebno napraviti nakon svake promjene na .html dokumentima.



Slika 4.5.: Postavljanje početne HTML stranice i zadanog direktorija

4.2.1. Čitanje vrijednosti

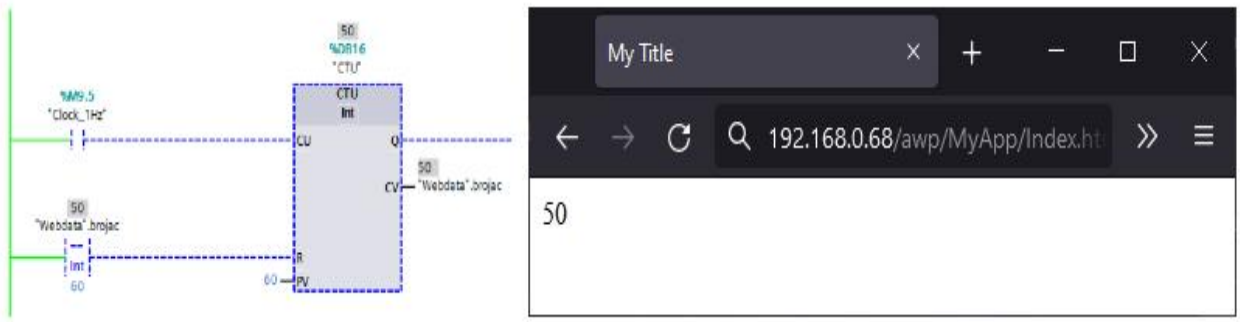
Korisnička stranica omogućuje ispisivanje te mijenjanje vrijednosti podataka preko web preglednika. Za prikaz vrijednosti potrebno je prvo kreirati DB iz kojeg ćemo čitati vrijednost.

Kao primjer kreiran je DB “webdata” u kojem se nalaze varijabla “brojac”. U glavnom programu koristimo brojač prema gore(eng. Count up, CTU) s ulazom 1Hz koji se sam resetira kada mu vrijednost dosegne 60s. Varijabla “brojac” iz DB-a “webdata” je postavljena na CV izlaz CTU-a Zatim je potrebno kreirati “index.html” kao što je prikazano na slici 4.6.

```
<!DOCTYPE html>
  <html>
    <head>
      <meta charset="utf-8">
      <title>My Title</title>
    </head>
  <body>
      :="webdata".brojac:
  </body>
</html>
```

Slika 4.6.: Index.html za čitanje vrijednosti varijable

Pomoću oznaka “:=” i “:” se vrši čitanje varijabli iz DB-a te se linija “:="webdata".brojac:” prikazuje kao vrijednost spremljena u DB-u, što se vidi na slici 4.7. U samoj naredbi “:="webdata".brojac:”, webdata je DB, a brojac je element DB-a. Ukoliko želimo koristiti varijable iz popisa PLC tagova, tada je dovoljno napisati samo naziv varijable. Tada u “index.html” pisemo “:=brojac:” umjesto “:="webdata".brojac:”. Nakon toga potrebno je generirati blokove u postavkama, prebaciti program na PLC te otvoriti korisničku stranicu koristeći web preglednik.



Slika 4.7.: Prikaz programa i web sučelja

4.2.2. Mijenjanje vrijednosti

Za mijenjanje vrijednosti potrebno je izmijeniti “index.html”. Na samom početku je potrebno dodati AWP komentar koji govori koja varijabla će se mijenjati. Ukoliko ne dodamo ovaj komentar, mijenjanje vrijednosti nije moguće. U glavnom programu, odnosno DB-u je dodana nova varijabla “broj” te će se njezina vrijednost mijenjati preko web preglednika. Na slici 4.8. je prikazan “index.html” za mijenjanje vrijednosti varijable. AWP komentar omogućava mijenjanje varijable, a :=”webdata”.broj: ispisuje broj na stranici. Unutar elementa <form/> su definirani elementi i atributi za unos i slanje upisanih vrijednosti u DB na PLC-u.

```
<!-- AWP_In_Variable Name='"webdata".broj' -->
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>My Title</title>
    </head>
    <body>
     :="webdata".broj:
     <form method="post">
       <input name='"webdata".broj' type="text" />
       <button type="submit">Save</button>
     </form>
   </body>
</html>
```

Slika 4.8.: Index.html za mijenjanje vrijednosti varijable

4.3. Izrada programa

Program je rađen u ljestvičastom dijagramu. Struktura programa sastoji se od jednog organizacijskog bloka u kojem se poziva WWW blok koji služi za povezivanje korisničke stranice s programom PLC-a te jednog funkcijskog bloka, u kojem se nalazi 8 mreža za realizaciju rada dizala.

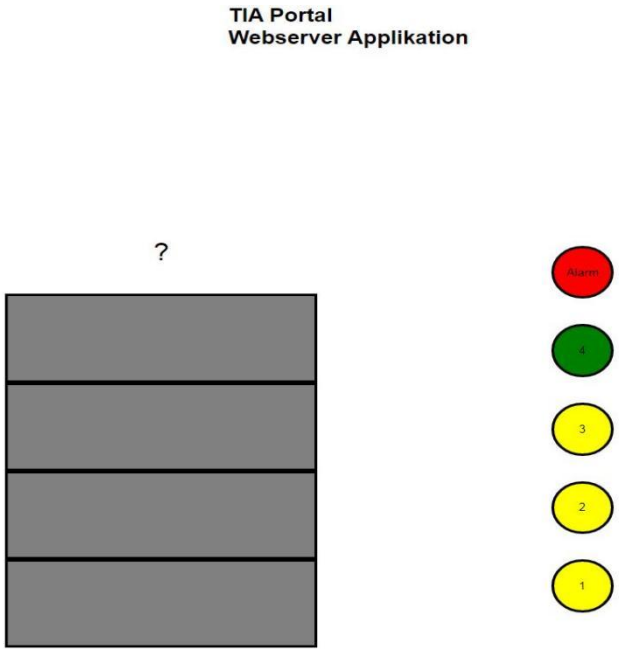
Kod pokretanja dizala potrebno je napraviti inicijalizaciju na način da se kabina postavi na prvi kat ako se već ne nalazi tamo. Ukoliko se ne odradi inicijalizacija nije moguće koristiti dizalo. Pritiskom na tipkalo TA vršimo inicijalizaciju, kojim ujedno radimo i aktivaciju i

deaktivaciju alarma. Nakon što je inicijalizacija obavljena, moguće je pozivanje dizala na željeni kat. Za poziv na bilo koji kat potrebno je pritisnuti tipkalo u kabini, na upravljačkoj ploči ili pak tipkalo na web sučelju. Uvjet da dizalo krene prema željenom katu je da motor ne ide gore ili dolje, da vrata nisu otvorena te da nije upaljen alarm. Pozivom dizala na određeni kat pale se LE diode u kabini i na upravljačkoj ploči te se mijenja boja tipkala na web sučelju. Prolaskom kabine pored senzora na željenom katu se deaktivira poziv te se otvaraju vrata na odabranom katu. Također, aktivacijom senzora se na web sučelju mijenja boja kata na kojem se nalazi kabina te se na taj način prikazuje gdje se kabina nalazi. Na maketi se nalazi i 7 segmentni pokaznik na kojem se prikazuje na kojem se katu nalazi dizalo. Web sučelju brojčano prikazuje na kojem se katu nalazi kabina.

4.4. Vizualizacija pomoću web sučelja

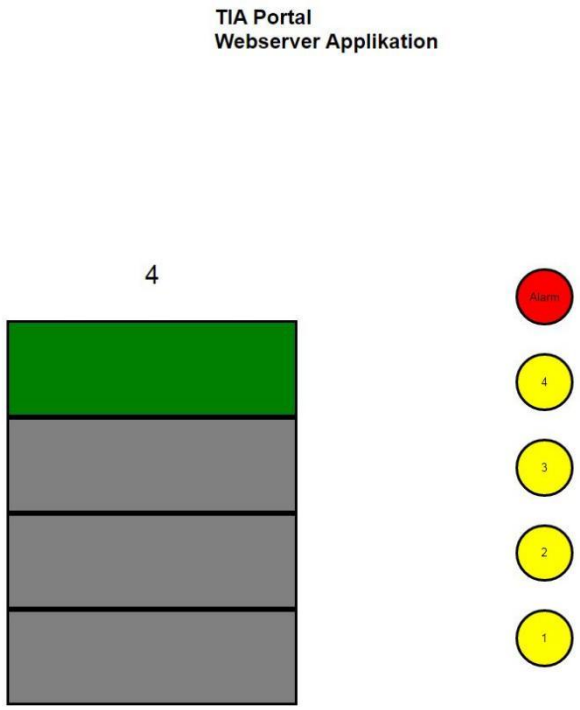
Budući su veličina i jačina ugrađenog web servera ograničavajući faktori, samo web sučelje ne podržava kompleksne stranice. Siemens na svojim stranicama navodi da je moguće koristiti HTML, CSS te JavaScript, dok je WEB API moguće koristiti na S7-1500.

Samo sučelje je jednostavno, sastoji se od prikaza zgrade od četiri kata te od upravljačke ploče koja je identična onoj na maketi. Pritiskom na bilo koju tipku poziva se dizalo na odabrani kat, a tipkalu se mijenja boja iz žute u zelenu dokle god dizalo ne stigne na odabrani kat. Na slici 4.9. je prikazan poziv dizala na 4. kat.



Slika 4.9.: Pozivanje dizala

Prolaskom dizala pored senzora koji se nalaze na svakom katu mijenja se boja kata zgrade na sučelju, što se može vidjeti na slici 4.10. Iznad dizala je prikazano na kojem se katu dizalo nalazi. U „IO.html“ datoteci nalaze se svi ulazi i izlazi korišteni u TIA Portalu. Svakih pola sekunde čitamo stanje ulaza i izlaza te na osnovu izlaza se mijenja boja katova, a boja tipkala na osnovu stanja ulaza. Za mijenjanje stanja inputa koristi se „POST request“ u Ajaxu, koji bi prilikom pritiska na tipkalo postavio u stanje 1. Boja tipkala i katova se automatski vraća na zadanu boju, bez interakcije korisnika, iz razloga što se vrijednosti čitaju iz TIA portala, te kad se vrijednosti promjene na 0 i boja će se promijeniti. Korištenjem Ajaxa za dohvaćanje i ažuriranja varijabli skraćeno je vrijeme učitavanja, ali nakon svakog slanja zahtjeva dolazi do treperenja stranice te se ona u tom trenutku postavlja na prvobitne postavke, što je vidljivo na slici 4.9, gdje se umjesto prikaza trenutnog kata nalazi “?”. .



Slika 4.10.: Dizalo na 4. katu

5. ZAKLJUČAK

S povećanjem obujma proizvodnje rasla je i potreba za sustavom upravljanja kojim će se moći lakše upravljati te kojeg će biti lakše modificirati. Iz te potrebe nastali su programibilni logički kontroleri. U vrlo kratkom vremenu su zbog svoje efikasnosti, jednostavnosti i niske cijene našli primjenu u raznim industrijama, ali i u nekomercijalnim djelatnostima, kao što je upravljanje radom dizala. Upravljanje dizalom u ovom radu je napravljeno pomoću web sučelja na kojem se moglo pratiti na kojem katu se dizalo nalazi te odabrati željeni kat. Za realizaciju je korišten ugrađen web server koji omogućava izradu jednostavnih web sučelja. Za samu izradu sučelja moguće je koristiti HTML, CSS i JavaScript. Pritiskom na tipkalo TA ili alarm vrši se inicijalizacija bez koje dizalo neće raditi. Nakon što je obavljena inicijalizacija moguće je korištenjem fizičkih tipki na maketi ili tipki na web sučelju pozvati dizalo na željeni kat. Pritiskom na tipku se mijenja boja kako bi se vidjelo na koji kat je pozvano dizalo, a dolaskom na određeni kat se mijenja boja kata na stranici iz sive u zelenu te se ispisuje broj kata na pokazniku iznad prikaza zgrade. Zbog ograničenja u kreiranju stranice dolazi do treperenja zbog stalnog učitavanja vrijednosti varijabli iz TIA Portala.

6. LITERATURA

[1] T., Šurina, Automatska regulacija, Školska knjiga, Zagreb, 1987.

[2] N., Perić, Automatsko upravljanje – predavanja, FER, Zagreb, 2004.

[3] D., Slišković, R., Cupec, Osnove automatskog upravljanja, materijali s predavanja, FERIT

[4] 60 years SIMATIC | Specials | Siemens Global, https://new.siemens.com/global/en/company/about/history/specials/60-years-of-simatic.html, 24.04.2022

[5] Product Details - Industry Mall - Siemens WW, https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7214-1HE30-0XB0, 25.04.2022

[6] Product Details - Industry Mall - Siemens WW, https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7223-1PL32-0XB0, 25.04.2022

[7] https://us.profinet.com/the-difference-between-profibus-and-profinet/, 29.04.2022

[8] W., Bolton, Programmable Logic Controllers, Elsevier Newnes, 2006.

[9] F.D. Petruzella, Programmable Logic Controllers, New York, Glencoe/McGraw-Hill, 1998.

[10] Build Responsive Websites with HTML5 and CSS3 | Udemy, 22.01.2022

[11] Learn Html Css And Javascript Online - Coursera, 25.01.2022

[12] https://us.profinet.com/what-is-the-difference-between-profibus-dp-and-pa/, 30.06.2022

[13] https://www.w3schools.com/css/css_intro.asp, 30.06.2022

[14] https://www.w3schools.com/html/default.asp, 30.06.2022

Sažetak

U radu je opisano upravljanje dizalom pomoću programirljivog logičkog kontrolera. Radom programiljivog logičkog kontrolera (PLC) će se upravljati preko web sučelja. Opisana je uloga PLC-a u automatizaciji postrojenja, kao i pregled razvoja. Napravljen je pregled programskih jezika korištenih za programiranje PLC-a te jezika za razvoj web stranica. Opisano je kreiranje web sučelja za čitanje i unos podataka. Napravljen je program za rad makete dizala i web sučelje za upravljanje. Web sučelje se sastoji od prikaza zgrade, pokaznika i kontrolnog panela. Prilikom poziva dizala tipka mijenja boju, a dolaskom dizala na željeni kat mijenja se boja kata zgrade i na pokazniku se ispisuje trenutni kat. Za izradu je korišten PLC Siemens S7-1200 s dodatnim signalnim modulom i napajanjem.

Ključne riječi: dizalo, HTML, PLC, Siemens, TIA Portal, web server

Abstract

Title: Developing a web application for managing PLC

The paper describes the operation of the elevator using a programmable logic controller. The operation of the programmable logic controller (PLC) will be managed through the web interface. The role of PLC in plant automation is described, as well as an overview of development. An overview of the programming languages used for the programming of PLC and the language for the development of websites has been made. It is described to create a web interface for reading and entering data. A program has been created for the operation of the elevator model and a web interface for control. The web interface consists of a representation of a building, a display and a control panel. When calling the elevator, the button changes color, and with the arrival of the elevator to the desired floor, the color of the floor of the building changes and the current floor is printed on the demonstration. The PLC Siemens S7-1200 with an additional signal module and power supply was used for construction.

Keywords: elevator, HTML, PLC, Siemens, TIA Portal, web server

Životopis

Josip Jasnić rođen je 14.02.1996. u Osijeku. U Valpovu pohađa osnovnu školu Matije Petra Katančića te zatim upisuje Medicinsku školu u Osijeku, smjer fizioterapeutski tehničar. Nakon završetka srednje škole odrađuje pripravnički staž na Odjelu za fizikalnu medicinu i rehabilitaciju KBC-a Osijek. Nakon odrađenog staža upisuje stručni studij elektrotehnike, smjer automatika, na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

Prilog

Prilog 1. Index.html

```
<!DOCTYPE html>
<html>

<head>
<title>Userdefined Website - Application Start</title>
<meta charset="UTF-8" >
<meta http-equiv="refresh" content="0.5" >
</head>

<style>
body {
font-family : Arial, Helvetica, sans-serif;
margin : 0px;
text-align : left;
text-decoration : none;
color : #000000;
overflow : auto;
}
#header {
display: flex;
justify-content: space-evenly;
align-items: center;
}
#floor-number {
font-size: 30px;
}
h2 {
width: fit-content;
}
.main-wrapper {
width: 100vw;
height: 100vh;
position: relative;
display: flex;
justify-content: center;
align-items: center;
}
.building {
display: flex;
flex-direction: column;
background-color: grey;
width: 20%;
```

```
height: 60%;
margin-right: 15%;
}
.building > * {
border: 3px solid black;
height: 25%;
}
.building > #floor-number {
background-color: white;
display: flex;
justify-content: center;
align-items: center;
border: none;
}
.alarm {
background-color: red;
}
.buttons {
width: 10%;
display: flex;
flex-direction: column;
gap: 2rem;
}
form > input {
border-radius: 50%;
width: 5em;
height: 5em;
border: 3px solid black;
}
.button-input {
background-color: yellow;
}
</style>

<body>

<div id="header">
<h2>TIA Portal<br>Webserver Applikation</h2>

</div>

<div class="main-wrapper">
<div class="building">
<div id="floor-number">?</div>
<div id="four"></div>
<div id="three"></div>
<div id="two"></div>
<div id="one"></div>
</div>
```

```
<div class="buttons">
<form method="post" action="">
<input type="submit" id="alarm-button" class="alarm" value="Alarm">
<input type="hidden" name='"webdata".alarm' size="80" value="1">
</form>
<form method="post" action="">
<input type="submit" id="T44" class="button-input" value="4">
<input type="hidden" name='"webdata".T44' size="80" value="1">
</form>
<form method="post" action="">
<input type="submit" id="T33" class="button-input" value="3">
<input type="hidden" name='"webdata".T33' size="80" value="1">
</form>
<form method="post" action="">
<input type="submit" id="T22" class="button-input" value="2">
<input type="hidden" name='"webdata".T22' size="80" value="1">
</form>
<form method="post" action="">
<input type="submit" id="T11" class="button-input" value="1">
<input type="hidden" name='"webdata".T11' size="80" value="1">
</form>

</div>
<div style="display: none;">
<h1 id="S1">:="webdata".S1:</h1>
<h1 id="S2">:="webdata".S2:</h1>
<h1 id="S3">:="webdata".S3:</h1>
<h1 id="S4">:="webdata".S4:</h1>
<h1 id="T1">:="webdata".T11:</h1>
<h1 id="T2">:="webdata".T22:</h1>
<h1 id="T3">:="webdata".T33:</h1>
<h1 id="T4">:="webdata".T44:</h1>
<h1 id="alarm">:="webdata".alarm:</h1>
</div>
<script src="jquery-2.0.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(document).ready(function(){
$.ajaxSetup({ cache: false });
$("input[id=T11]").click(function(){
url="IO.htm";
name='"webdata".T11';
val='1';
sdata=escape(name)+'='+val;
$.post(url,sdata,function(result){});
});
$("input[id=T22]").click(function(){
url="IO.htm";
name='"webdata".T22';
val='1';
sdata=escape(name)+'='+val;
```

```
$.post(url,sdata,function(result){});
});
$("input[id=T33]").click(function(){
url="IO.htm";
name='"webdata".T33';
val='1';
sdata=escape(name)+'='+val;
$.post(url,sdata,function(result){});
});
$("input[id=T44]").click(function(){
url="IO.htm";
name='"webdata".T44';
val='1';
sdata=escape(name)+'='+val;
$.post(url,sdata,function(result){});
});
$("input[id=alarm]").click(function(){
url="IO.htm";
name='"webdata".alarm';
val='1';
sdata=escape(name)+'='+val;
$.post(url,sdata,function(result){});
});

});

const firstFloor = document.getElementById('S1').innerText;
const secondFloor = document.getElementById('S2').innerText;
const thirdFloor = document.getElementById('S3').innerText;
const fourthFloor = document.getElementById('S4').innerText;
const firstDiv = document.getElementById('one');
const secondDiv = document.getElementById('two');
const thirdDiv = document.getElementById('three');
const fourthDiv = document.getElementById('four');
const display= document.getElementById('floor-number');
firstFloor  === '1' ? firstDiv.style.backgroundColor = 'green' : firstDiv.style.backgroundColor =
'grey';
secondFloor === '1' ? secondDiv.style.backgroundColor = 'green' :
secondDiv.style.backgroundColor = 'grey';
thirdFloor  === '1' ? thirdDiv.style.backgroundColor = 'green' : thirdDiv.style.backgroundColor
= 'grey';
fourthFloor   === '1' ? fourthDiv.style.backgroundColor = 'green' :
fourthDiv.style.backgroundColor = 'grey';
firstFloor  === '1' ? display.innerText = '1' : ''
secondFloor === '1' ? display.innerText = '2' : ''
thirdFloor  === '1' ? display.innerText = '3' : ''
fourthFloor   === '1' ? display.innerText = '4' : ''
```

```
const firstButton = document.getElementById('T1').innerText;
const secondButton = document.getElementById('T2').innerText;
const thirdButton = document.getElementById('T3').innerText;
const fourthButton = document.getElementById('T4').innerText;
const alarm = document.getElementById('alarm-button').innerText;
const firstButtonDiv = document.getElementById('T11');
const secondButtonDiv = document.getElementById('T22');
const thirdButtonDiv = document.getElementById('T33');
const fourthButtonDiv = document.getElementById('T44');
const alarmDiv = document.getElementById('alarm');

firstButton === '1' ? firstButtonDiv.style.backgroundColor = 'green' :
firstButtonDiv.style.backgroundColor = 'yellow';
secondButton === '1' ? secondButtonDiv.style.backgroundColor = 'green' :
secondButtonDiv.style.backgroundColor = 'yellow';
thirdButton === '1' ? thirdButtonDiv.style.backgroundColor = 'green' :
thirdButtonDiv.style.backgroundColor = 'yellow';
fourthButton === '1' ? fourthButtonDiv.style.backgroundColor = 'green' :
fourthButtonDiv.style.backgroundColor = 'yellow';

alarm === '1' ? alarmDiv.style.backgroundColor = 'blue' : alarmDiv.style.backgroundColor =
'red';

</script>
</div>
</body>
</html>
```

Prilog 2. IO.html

```
<!-- AWP_In_Variable Name='"webdata".T11' -->
<!-- AWP_In_Variable Name='"webdata".T22' -->
<!-- AWP_In_Variable Name='"webdata".T33' -->
<!-- AWP_In_Variable Name='"webdata".T44' -->
<!-- AWP_In_Variable Name='"webdata".alarm' -->
:="webdata".T11:
:="webdata".T22:
:="webdata".T33:
:="webdata".T44:
:="webdata".S1:
:="webdata".S2:
:="webdata".S3:
:="webdata".S4:
:="webdata".alarm:
```