KONTROLA POZICIJE S BIDIREKCIJSKIM POZICIONEROM KORISTEĆI TUSTINOVU APROKSIMACIJU KRIVULJE

Jakupak, Marin

Master's thesis / Diplomski rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Rijeka, Faculty of Engineering / Sveučilište u Rijeci, Tehnički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:190:868520

Rights / Prava: Attribution 4.0 International/Imenovanje 4.0 međunarodna

Download date / Datum preuzimanja: **2024-05-20**

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Engineering

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

KONTROLA POZICIJE S BIDIREKCIJSKIM POZICIONEROM KORISTEĆI TUSTINOVU APROKSIMACIJU KRIVULJE

Rijeka, studeni 2023.

Marin Jakupak

0069079399

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

KONTROLA POZICIJE S BIDIREKCIJSKIM POZICIONEROM KORISTEĆI TUSTINOVU APROKSIMACIJU KRIVULJE

Mentor: Prof. dr. sc. Dario Matika

Rijeka, studeni 2023.

Marin Jakupak

0069079399

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA DIPLOMSKE ISPITE

Rijeka, 18. travnja 2023.

Zavod: **Zavod za automatiku i elektroniku**
Predmet: **Automatizacija postrojenja i procesa**
Grana: **2.03.06 automatizacija i robotika**

ZADATAK ZA DIPLOMSKI RAD

Pristupnik: **Marin Jakupak (0069079399)**
Studij: Sveučilišni diplomski studij elektrotehnike
Modul: Automatika

Zadatak: **Kontrola pozicije s bidirekcijskim pozicionerom koristeći Tustinovu aproksimaciju krivulje/Position control with a bidirectional positioner using the Tustin curve approximation**

Opis zadatka:

Potrebno je projektirati i izraditi upravljački program za kontrolu pozicije primjenjiv na sustavima upravljanim hidrauličkim aktuatorima ili elektromotorim pogonima. Upravljački sustav treba omogućiti upravljanje brzinom aktuatora te kontrolu aktualne pozicije u odnosu na izračunatu poziciju dobivenu iz pozicionera. Projketiranje uključuje detaljan opis sustava, razvoj programske aplikacije upravljanja i vizualizacije u programskom okruženju Siemens Tia portal, te funkcionalno ispitivanje sustava korištenjem aplikacije za simulaciju na programabilnom logičkom kontroleru Siemens Tia Portal S7-1500.

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

Zadatak uručen pristupniku: 20. ožujka 2023.

Mentor:

Prof. dr. sc. Dario Matika

Dr. sc. Nikola Turk (komentor)

Predsjednik povjerenstva za diplomski ispit:

Prof. dr. sc. Dubravko Franković

IZJAVA

Izjavljujem da sam samostalno izradio diplomski rad, sukladno pravilniku o pisanju diplomskog rada prema zadatku preuzetog 20. ožujka 2023.

Rijeka, studeni 2023.

Marin Jakupak

ZAHVALA

Ovim putem bih se zahvalio mentoru prof. dr. sc. Dariu Matiki i komentoru dr. sc. Nikoli Turku na stručnoj podršci, savjetima te uloženom vremenu tijekom izrade ovog diplomskog rada.

Također hvala tvrtki Danieli Systec d.o.o. na pomoći oko odabira teme ovog rada i g. Toniju Tumpiću na vodstvu tijekom izrade diplomskog rada i izdvojenom vremenu.

Hvala i obitelji i prijateljima na pruženom strpljenju i razumijevanju tijekom studiranja.

SADRŽAJ

1 UVOD 1

2 REGULACIJA, REGULACIJSKI KRUG I ELEMENTI REGULACIJSKOG KRUGA 2

3 BIDIREKCIJSKI POZICIONER I MATEMATIČKI MODEL SUSTAVA 5

4 HARDVERSKO OKRUŽENJE 10

4.1 Programmable logic controller (PLC) 11

4.2 Frekvencijski pretvarač 13

4.3 Servo motor 13

4.4 Human machine interface (HMI) 14

4.5 Računalo – softverski alat 15

5 SOFTVERSKO OKRUŽENJE – TIA PORTAL 16

5.1 Uvod u glavne značajke 16

5.2 Konfiguracija uređaja i uspostava komunikacije 17

5.3 Programiranje PLC-a 19

5.4 Preuzimanje programa na PLC i *monitoring* način rada 21

5.5 Vizualizacija HMI-a 23

6 PROGRAMSKO RJEŠENJE 24

6.1 Upravljanje frekvencijskim pretvaračem 24

6.2 Izračun trenutne pozicije 32

6.3 Realizacija pozicionera 35

6.4 Projektiranje regulatora 48

7 IZRADA HMI VIZUALIZACIJE 54

7.1 Razmjena varijabli između PLC-a i HMI-a 54

7.2 Izrada početnog zaslona i predloška 55

7.3 Zaslon frekvencijskog pretvarača 56

7.4 Zaslon pozicionera 56

7.5 Zaslon grafičkog prikaza procesnih vrijednosti 57

7.6 Zaslon PI regulatora 58

8 TESTIRANJE PROGRAMSKOG ALGORITMA SUSTAVA 59

8.1 Idealan slučaj sa potpunim profilom brzine 59

8.2 Gibanje koristeći akceleraciju i deceleraciju 60

8.3 Gibanje koristeći profil akceleracija – deceleracija - niža konstantna brzina – deceleracija 61

8.4 Gibanje profilom akceleracija – niža konstantne brzina – deceleracija 62

8.5 Gibanje profilom brzine bez niže konstantne brzine 63
8.6 Testiranje regulatora utjecajem poremećajnih veličina 64
9 ZAKLJUČAK 69
LITERATURA 70
POPIS OZNAKA I KRATICA 71
SAŽETAK I KLJUČNE RIJEČI 72
ABSTRACT AND KEY WORDS 72
DODATAK A – GENERAL DB 73
DODATAK B – INVERTER IDB 74
DODATAK C – HMI COMMANDS DB 75
DODATAK D – POISTIONER DB 76

1 UVOD

Proces kojim se sa trenutnog stanja dolazi do željenog stanja upotrebom točno određenih i definiranih pravila naziva se upravljanje. Gledano iz tehničke perspektive, upravljanje se danas sve više oslanja na upotrebu tehničkih uređaja koji olakšavaju ili umanjuju ljudski rad što predstavlja proces automatizacije. Uređaji koji sudjeluju u ovom procesu nazivaju se automatski uređaji. Oni su zaduženi kako bi se kvalitetno, na ispravan način i određenim redoslijedom proces upravljanja mogao odvijati. [1]

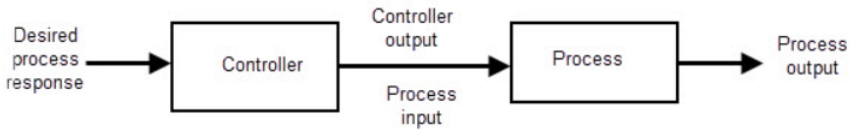
Tako će u ovom radu biti obrađen sustav koji izvršava kontrolu pozicije korištenjem pozicionera. Najjednostavnije prevedeno, sustav kojim se omogućuje pomicanje objekta iz trenutne pozicije u zadanu (konačnu) poziciju. Međutim, odlazi se korak dalje od upravljanja, a to je regulacija sustava. Iako su ovi pojmovi na prvi pogled slični i često se rabe u istom kontekstu, postoji razlika između njih te oni ne predstavljaju isti proces. Upravljanjem se misli na mogućnost da se stanje pojedinog sustava promjeni po određenim zakonitostima, dok se pod regulacijom smatra proces ili mehanizam koji omogućava da upravljanje sustavom bude izvršeno što točnije. Cilj regulacije je dakle, postizanje željenog stanja sustava ili praćenje njegove promjene na način da smanjimo (eliminiramo) utjecaj drugih veličina. [2]

Detaljniji princip rada regulacije bit će objašnjen u sljedećem poglavlju, gdje će se na primjeru sustava kontrole pozicije prikazati elementi regulacijskog kruga te nakon toga opisati i matematički model. U poglavlju nakon, bit će dan uvid u tehničke uređaje koji su korišteni. Sljedeće će poglavlje biti opis softverskog alata *TIA Portal* koji se koristi za generiranje algoritama upravljanja, testiranje i vizualizaciju procesa prije nego se implementira u stvarnom svijetu. U zadnja dva poglavlja će biti opisan način na koji se razvio algoritam upravljanja te odrađene simulacije testiranja sustava.

Krajnji je cilj ovog diplomskog rada napraviti sustav automatske regulacije pozicijom korištenjem potrebnih uređaja i softverskog alata te dobivene rezultate prikazati grafičkim putem. Na temelju tih rezultata izvući će se zaključci o ponašanju sustava te njegovim karakteristikama.

2 REGULACIJA, REGULACIJSKI KRUG I ELEMENTI REGULACIJSKOG KRUGA

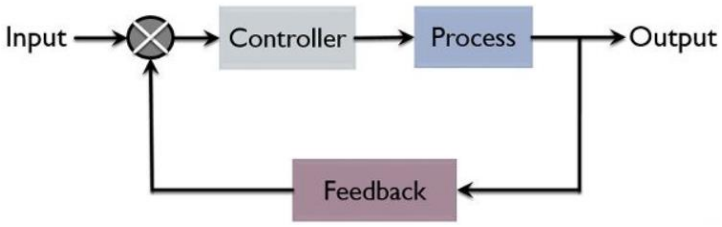
Kako bi se mogao razumjeti koncept regulacije, potrebno je krenuti od jednostavnijeg principa, preteče regulaciji, a to je upravljanje. Upravljanje se povezuje sa sustavom otvorenog kruga iz razloga što se izvršni član (koji je objekt upravljanja) kontrolira samo na temelju ulazne veličine bez prisutnosti povratnih informacija o utjecajima poremećajnih veličina ili ponašanju same izlazne veličine sustava. Na Slika 2.1. Blokovska shema otvorenog kruga [3] je prikazana generalna blokovska shema ovakvog sustava. Blokovska shema je prikaz sustava pomoću blokova koji predstavljaju glavne dijelove sustava, te linijama koje prikazuju odnose između njih. [1]



Slika 2.1. Blokovska shema otvorenog kruga [3]

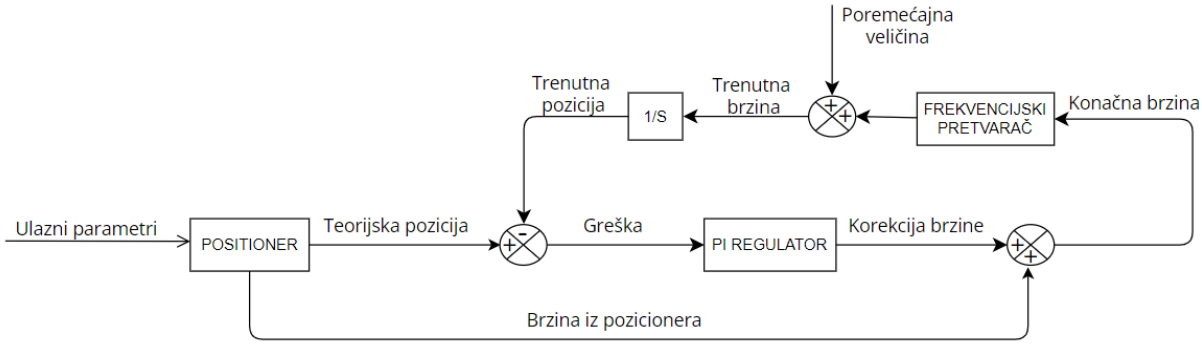
U ovom slučaju postoje dva bloka, koji predstavljaju upravljački element i proces kojim se upravlja. Putem ove sheme se može vidjeti raniji opis otvorenog kruga, gdje se ulazna veličina - željeni odziv procesa, postiže djelovanjem upravljačkog elementa na određenom procesu. Tada proces mijenja stanje te se to interpretira kao izlazna veličina sustava. Ovakvi sustavi su jednostavni za implementaciju, ali stoga često, uslijed djelovanja poremećaja, neprecizni pa imaju ograničenu upotrebu. [3]

S druge strane, ako se otvorenom sustavom doda povratna veza, preko koje se željena vrijednost može uspoređivati sa stvarnom vrijednošću sustava dobiva se zatvoreni regulacijski krug. Cilj ovakve modifikacije je mogućnost regulacije procesa na temelju poznavanja stanja izvršnog elementa. Slika prikazuje blokovsku shemu zatvorenog regulacijskog kruga.



Slika 2.2. Blokovska shema zatvorenog kruga [4]

U ovom se slučaju isto tako ulaznom veličinom zadaje željena vrijednost koju proces mora pratiti, ali također su dostupne povratne informacije u kojem je stanju proces nakon takve pobude. Stvarno stanje ne mora odgovarati zadanom stanju, iz razloga što se u realnim sustavima pojavljuju poremećajne veličine koje donose takozvano regulacijsko odstupanje. Konačna svrha regulacije je smanjiti odstupanje u granicama tolerancije, a element koji to omogućuje naziva se regulator. Princip rada regulatora biti će opisan zajedno sa blokovskom shemom u nastavku.



Slika 2.3. Blokovska shema sustava kontrole pozicije sa bidirekcijskim pozicionerom

Na Slika je prikazana blokovska shema kojom se bavi ovaj rad., odnosno sustava kontrole pozicije. Blokovska shema je nešto složenija, nego u prethodnim slučajevima koji su opisivali generalne sustave, zbog korištenja više elemenata koji će biti objašnjeni u ovom radu. Ulazna veličina podrazumijeva ulazne parametre koji su najčešće konačna pozicija gibanja te konfiguracijski parametri koji opisuju brzinu gibanja, odnosno na koji se način gibanje izvodi. Blok pozicioner predstavlja samu izvedbu pozicioniranja, pravila i ograničenja tijekom gibanja. Izlazne veličine iz tog bloka su brzina kojom se treba gibati kako bi se došlo do konačne pozicije te teorijska pozicija. Teorijska pozicija predstavlja poziciju u kojoj se treba nalaziti pozicioner ako se giba upravo spomenutom brzinom. Kako bi se dobile povratne informacije o brzini, odnosno poziciji, koristi se frekvencijski pretvarač. Detaljnije će biti opisan u 4. poglavlju, ali za sad je dovoljno znati da se prema njemu šalje konačna brzina kojom se želimo gibati te da se pomoću njega upravlja motorom. Isto tako frekvencijski pretvarač „igra ulogu“ mjernog uređaja pa time šalje povratne informacije o trenutnoj brzini motora. Ako nastupi neka poremećajna veličina, to će se odraziti na brzini motora. Pošto se u ovom sustavu izvršava kontrola pozicije, potrebno je dobiti poziciju iz poznate brzine. To radi blok integrator, na shemi oznake 1/S. Matematički je poznato da je pozicija integral brzine, te se stvarna vrijednost pozicije u kojoj će se pozicioner nalaziti dobiva integriranjem brzine vrtnje. Ovo čini povratnu vezu regulacijskog kruga, tako da je u bilo kojem trenutku poznata stvarna vrijednost pozicije u odnosu na onu teorijsku. Time se dolazi do uloge regulatora. Regulator je element koji je zadužen da, obično na

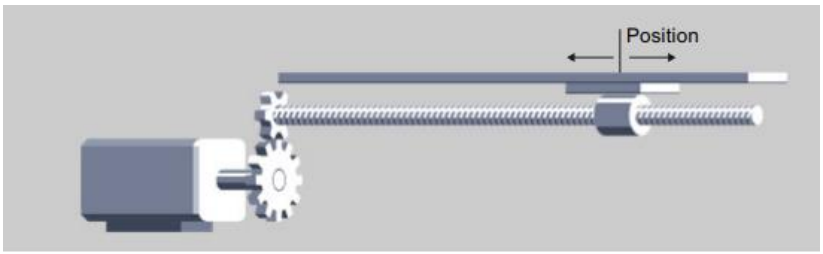
temelju dvije ulazne vrijednosti, ispravi ponašanje sustava. Ulazne vrijednosti su željena veličina i stvarna veličina, njihovom razlikom se dobiva greška (regulacijsko odstupanje) koja se putem regulacijskog djelovanja ispravlja (eliminira) mjenjanjem upravljačkog signala koji zapravo predstavlja izlaz iz regulatora. Regulacijsko djelovanje PID regulatora se sastoji od tri komponente - proporcionalno, integralno i diferencijalno djelovanje. Odabir tipa regulatora ovisi o specifičnim potrebama procesa koji se regulira, te se u ovo slučaju koristi PI regulator. Proporcionalno djelovanje proporcionalno mijenja izlaznu vrijednost u ovisnost o ulaznoj greški, dok integralno djelovanje integrira grešku po vremenu. Svako od njih donosi svoje prednosti te je ovaj tip regulatora dovoljan kako bi se uklonio utjecaj poremećajne veličine. Izlazna vrijednost iz regulatora je korekcijska vrijednost brzine koju je potrebno dodati na brzinu iz pozicionera kako bi se dobila konačna brzinu kojom se pozicioner mora gibati.

Može se primijetiti kako je glavni dio ovog prethodno opisanog sustava kontrole pozicije zapravo element pozicioner, koji uz ostale elemente prisutne u regulacijskom krugu omogućava funkcionalno rješenje. Iz tog će razloga u sljedećem poglavlju detaljnije biti opisan ovaj element te njegove specifikacije koje definiraju kontrolu pozicije.

3 BIDIREKCIJSKI POZICIONER I MATEMATIČKI MODEL SUSTAVA

Kontrola pozicije je u svijetu automatizacije jedan od najbitnijih elemenata kojim se postiže visoka preciznost i efikasnost u obavljanju zadataka pozicioniranja – drugim riječima, postiže se visoka efikasnost i preciznost u obavljanju zadaće pomicanje ili promjene pozicije objekta u odnosu na trenutni položaj. Sustav ili uređaj koji to omogućuje naziva se pozicioner. Bidirekcijski pozicioner omogućuje pozicioniranje u dva smjera po jednoj osi. Glavni dio je aktuator koji razvija potrebni moment za svladavanje određenog puta. Moment se generira pretvorbom nekog od oblika energije (električne, hidrauličke, pneumatičke) u mehanički rad. Električni AC/DC motori koriste električnu energiju, hidraulički aktuatori koriste fluid pod tlakom, dok pneumatski za dobivanje mehaničkog gibanja koriste komprimirani zrak. Aktuatorom se upravlja putem upravljačke jedinice koja generira upravljačke signale kako bismo ostvarili željeno gibanje. Upravljačka jedinica, aktuator (motor) i ostali uređaji bit će predstavljeni u sljedećem poglavlju.

Slika 3.1 prikazuje jednu od izvedba pozicionera. Aktuator putem redukcijskog prijenosa pomiče objekt koji se pozicionira u dva smjera. Redukcijski prijenos je izvršen unutar uređaja koji se naziva reduktor. Zadatak reduktora je smanjenje brzine vrtnje motora na brzinu prilagođenu procesu pozicioniranja. U ovom je slučaju izvršena pretvorba rotacijske brzine u linearnu brzinu.



Slika 3.1. Izvedba bidirekcijskog pozicionera [5]

Princip rada pozicionera te općenito sustava kontrole pozicije bit će opisan tijekom matematičkog modeliranja. Matematičko modeliranje je postupak kojim se problem iz stvarnog svijeta opisuje matematičkim izrazima, najčešće jednadžbama kako bi se sustav za koji je modeliranje rađeno mogao projektirati, analizirati i po potrebi optimirati. Model se sastoji od varijabli, koje mogu biti ulazne ili izlazne, te jednadžbi koje opisuje odnose između njih. U nastavku će biti opisan primjer modeliranja u slučaju sustava kontrole pozicije. [6]

Kao što je već prije bilo spomenuto, zadatak pozicionera je pomicanje do određene pozicije. Iz toga slijedi da izlazna varijabla sustava mora biti pozicija kao rezultat samog pozicioniranja. Ova pozicija je teorijska pozicija, koja je dobivena matematičkim izračunom i ne mora biti jednaka

stvarnoj poziciji. Međutim, motori, odnosno aktuatori koji se upravljaju izvršavaju mehanički rad rotacijskim gibanjem, što je ujedno brzina (okretaji u minuti). Iz tog je razloga brzina također izlazna varijabla sustava, ali je riječ o linearnoj brzini. Izraz (3.1) prikazuje izračun linearne brzine na temelju kružne brzine.

$$v_{lin} = \frac{\omega_{rot} \cdot d \cdot \pi}{r \cdot 60} \qquad (3.1)$$

Gdje je:

v_{lin} linearna brzina,

ω_{rot} broj okretaja u minuti,

d promjer bubnja,

r redukcijski faktor.

Linearno gibanje je izvedeno na temelju kružne brzine koja se preko redukcijskog faktora umanjuje za njegovu vrijednost. Osovina izlaza iz reduktora je spojena na bubanj koji služi za namatanje sajle. Odmatanjem sajle sa bubnja dolazi do povećanja pozicije, dok ponovnim namatanjem pozicija objekta spojenog na sajlu se smanjuje.

Pozicija se tada može dobiti kao integracija brzine, prema izrazu (3.2).

$$s(t) = \int v(t)\, dt \qquad (3.2)$$

gdje je:

$s(t)$ funkcija pozicije u vremenu,

$v(t)$ funkcija brzine u vremenu,

dt beskonačno mali interval u vremenu, diferencijal.

Međutim, računalni zapis ovakvog izraza nije praktičan za implementaciju na digitalno računalo (upravljačku jedinicu), pošto se radi o kontinuiranoj jednadžbi u vremenu, a kao što je poznato digitalna računala rade u diskretnoj domeni. Iz navedenog razloga, implementacija izraza (3.2) na upravljačku jedinicu ostvaruje se pomoću aproksimacijskih postupaka numeričke integracije. Tustinova aproksimacija krivulje jedan je aproksimacijski postupak koji zamjenjuje operaciju integrala sumom trapezoidalnih oblika uzetih u jednakim vremenskim razmacima. Ovakav postupak proizlazi iz činjenice da je integral površina ispod krivulje. Ako se Tustinova aproksimacija primjeni na izračun pozicije iz krivulje brzine, može se zapisati izrazom (3.3):

$$s = s_0 + 0.5 \cdot \Delta t \cdot \left(v_{prev} + v_{act}\right) \qquad (3.3)$$

gdje je:

s vrijednost trenutne pozicije

s_0 početna vrijednost pozicije

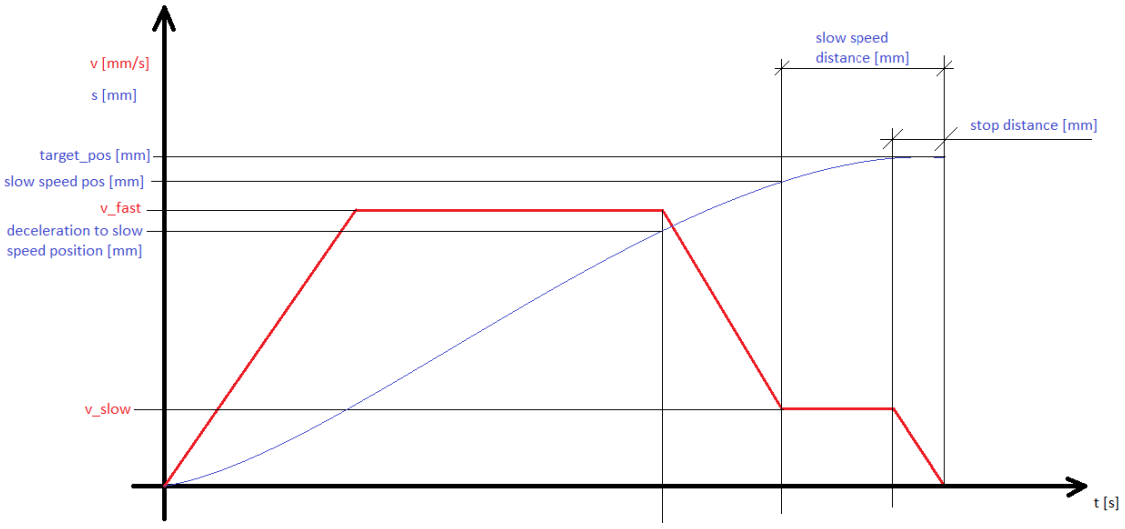
Δt interval očitavanja (vrijeme uzorkovanja)

v_{prev} vrijednost brzine prethodnog uzorka

v_{act} vrijednost trenutne brzine

Dakle, izračun se bazira na podjeli krivulje brzine na jednako razmaknute uzorke u vremenu i očitavanju vrijednosti brzine svakog uzorka. Kako bi aproksimacija izračuna pozicije bila što točnija potrebno je uzeti što više uzoraka u vremenu.

Ulazna varijabla pozicionera je trenutna pozicija, što označava stvarnu poziciju u kojoj se nalazimo. Isto tako u ulazne varijable se mogu svrstati i konfiguracijski parametri. To su parametri koje je moguće mijenjati po potrebi, kako bi se zadovoljile različite aplikacije pozicionera. Ovi parametri uključuju iznos akceleracije, deceleracije, više i niže konstantne brzine, te udaljenosti kretanja nižom konstantom brzinom. Njihovom se promjenom direktno utječe na promjenu brzine kojom se pozicioniramo te je time omogućeno korištenje pozicionera u različitim procesima. Ako se oni prikažu u ovisnosti o vremenu, dobije se profil brzine sa Slika 3.12 prikazan crvenom bojom.



Slika 3.1. Ovisnost brzine i pozicije u vremenu tijekom pozicioniranja sa označenim parametrima

Profil brzine predstavlja grafički prikaz promjene brzine u ovisnosti o vremenu, kako bi se iz trenutne pozicije stiglo do konačne pozicije. Idealni slučaj gibanja takvom putanjom je prikazan

na Slika 3.1 krivuljom crvene boje, gdje su upotrebljeni svi konfiguracijski parametri na način da vrijedi sljedeće:

Gibanje započinje ubrzavanjem sa trenutne pozicije sve do više konstantne brzine iz koje nakon određenog vremena usporavamo do niže konstantne brzine. Nižom se konstantnom brzinom gibamo sve do trenutka u kojem je potrebno usporavati kako bi stigli do konačne pozicije.

Ovakvim gibanjem se omogućuje sigurno, brzo i efikasno pozicioniranje iz razloga što se na ovaj način jednim dijelom putanje gibamo većom brzinom, što je ključno kako bi se pozicioniranje odradilo vremenski što brže. S druge strane, gibanje se drugim dijelom putanje nastavlja u nižoj brzini kako bi se pogon što preciznije pozicionirao u konačnu poziciju. Međutim, ovo ne mora vrijediti uvijek, gibanje može biti ograničeno samo korištenjem nekih od parametara te se time dobiva drugačiji profil brzine.

Kako bi se matematički opisalo gibanje, potrebno je upravo ovaj profil brzine podijeliti na faze, gdje je svaka od faza opisana određenim konfiguracijskim parametrima. Prva faza podrazumijeva akceleraciju, gdje brzina, na temelju poznatog parametra iznosa akceleracije, linearno raste tijekom vremena, što možemo zapisati izrazom (3.4).

$$v(t) = v_0 + acc * t \qquad (3.4)$$

Druga faza je faza više konstantne brzine. Viša konstantna brzina je parametar koji definira najveću moguću brzinu koja se postiže tijekom gibanja. Pošto je brzina konstantna, vrijedi izraz (3.5).

$$v(t) = konst. \qquad (3.5)$$

Treća faza je faza deceleracije sa više konstantne brzine na nižu konstantnu brzinu, gdje se brzina linearno smanjuje na osnovu parametra deceleracije. Brzina se može opisati izrazom (3.6) te se smanjuje sve do iznosa niže konstantne brzine.

$$v(t) = v_0 - dec * t \qquad (3.6)$$

Četvrta faza označava gibanje nižom konstantom brzinom. Ovu fazu karakteriziraju dva parametra, iznos niže konstantne brzine i udaljenost gibanja nižom konstantnom brzinom. Brzina je u ovoj fazi konstantna kao i u drugoj fazi, međutim drugačijeg je iznosa. Početak ove faze je definiran parametrom udaljenosti, koji definira put gibanja nižom konstantom brzinom i deceleracijom do konačne pozicije(prikazan na Slika 3.1 pod imenom *slow speed distance).*

Petom, ujedno i posljednjom fazom, se opisuje gibanje deceleracije do konačne pozicije, gdje vrijedi isti izraz promjene brzine kao i u trećoj fazi, međutim brzina se smanjuje sve do vrijednosti nula.

Osim parametara brzine, potrebno je poznavati i parametre pozicije koji će određivati teorijske vrijednosti pozicija u kojima je potrebno započeti s prelaskom iz jedne u drugu fazu. Na temelju konfiguracijskog parametra *slow speed distance* te ulaznog parametra konačna brzina moguće je izračunati poziciji početka gibanja nižom konstantom brzinom, što je zapisano izrazom (3.7).

$$slow\ speed\ position = target\ position - slow\ speed\ distance \qquad (3.7)$$

Sljedeća teorijska pozicija koju je potrebno izračunati je pozicija početka usporavanja sa više konstantne brzine u nižu konstantnu brzinu, označena na Slika 3.1 kao *deceleration to slow speed position.* Međutim, za izračun ove pozicije potrebno je znati udaljenost usporavanja sa više na nižu konstantnu vrijednost što je prikazano izrazom (3.8):

$$dec_{space} = \frac{\left(v_{fast} - v_{slow}\right)^2}{2 * dec} \qquad (3.8)$$

gdje je:

dec_{space}	udaljenost usporavanja sa više na nižu konstantnu brzinu
v_{fast}	vrijednost više konstantne brzine
v_{slow}	vrijednost niže konstantne brzine
dec	vrijednost deceleracije

Time se pozicija usporavanja sa više konstantne brzine na nižu konstantnu brzine računa izrazom (3.9):

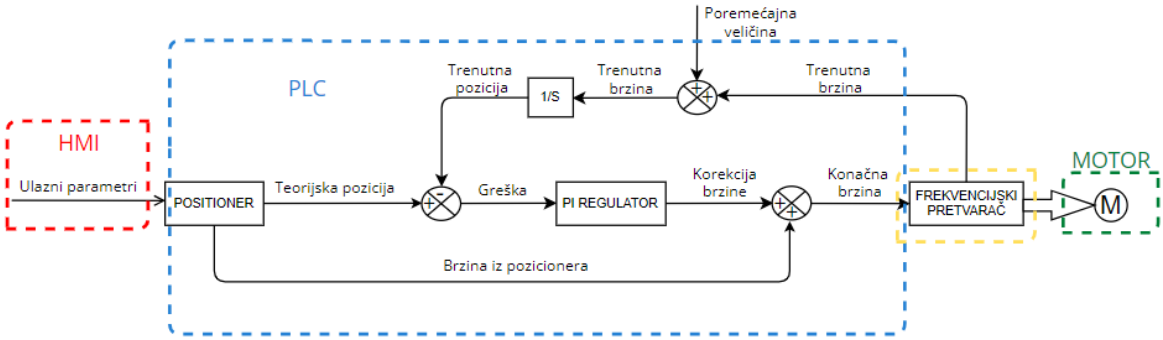
$$deceleration\ to\ slow\ speed = slow\ speed\ position - dec_{space} \qquad (3.9)$$

Preostao je izračun još jedne pozicije, pozicija početka usporavanja do konačne pozicije. Izraz (3.10) prikazuje izračun ove pozicije.

$$deceleration\ to\ zero = \frac{v_{slow}^2}{2 * dec} \qquad (3.10)$$

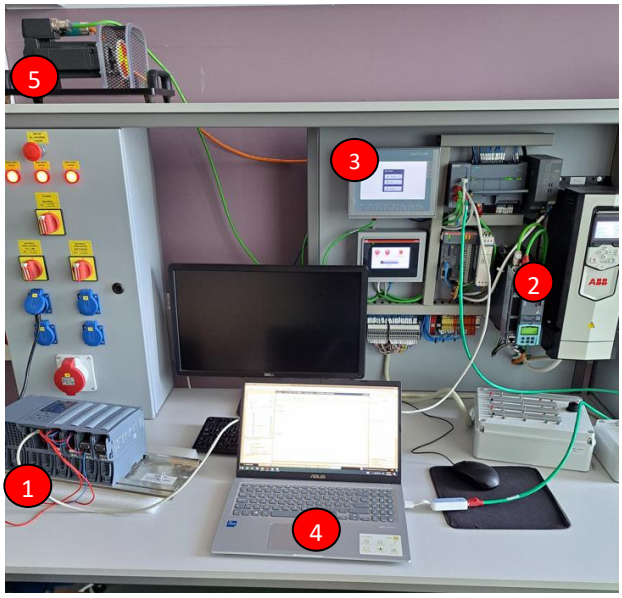
4 HARDVERSKO OKRUŽENJE

Kako bi se teorijski koncepti, opisani u prethodnim poglavljima mogli implementirati u realnom svijetu, neophodno se služiti automatskim uređajima koji imaju ulogu izvršavanja upravljanja. Neki od njih su već bili spomenuti kako bi se dobila slika potrebna za razumijevanje koncepta regulacije. Međutim u ovom će poglavlju biti predstavljeni preko uloge koju „nose“ u sustavu kontrole pozicije uz pojedinačni opis svakog od njih. Zapravo spomenuta uloga se već može vidjeti na **Error! Reference source not found.**.



Slika 4.1. Hardverska uloga u blokovskoj shemi sustava kontrole pozicije

Laboratorijski postav svih prisutnih uređaja koji su korišteni za izvršavanje sustava kontrole pozicije prikazani su na Slika 4.11.



Slika 4.11. Laboratorijski postav sustava kontrole pozicije

Numerirani oznake sa slike 4.2 predstavljaju sljedeće uređaje:

1. PLC 1516-3 PN/DP

2. SINAMICS S120 frekvencijski pretvarač

3. HMI KP700

4. Računalo

5. Motor

4.1 Programmable logic controller (PLC)

Programabilni logički kontroler (u nastavku PLC) je uređaj koji se koristi u industrijskim procesima kako bi se automatizirao neki proces, upravljalo mehanizacijom i olakšao pristup kompleksnom rješavanju problema. Prvobitna zadaća PLC-a je bila zamjena ogromne količine ožičenja relejne logike koja se trebala sprovesti kako bi se dobila određena logička funkcija, dok je u suvremeno doba PLC neizostavan uređaj u automatizaciji procesa. Princip rada zasnovan je na prikupljanju informacija o stanjima sa ulaznih varijabli, te izvršavanje logičkih operacija s ciljem kontroliranja izlaznih varijabli. Ulazne varijable su signali koji se dobivaju sa npr. tipkala, sklopke, senzora, dok se izlazni signali šalju prema relejima, aktuatorima, lampicama, frekvencijskim pretvaračima i slično. Signali koji se primaju ili šalju mogu biti diskretni ili analogni. Diskretni signali poprimaju vrijednosti nule (odsustvo signala) ili jedinice (prisutnost signala). Analogni signali prenose informacije preko naponskih (0-10V) ili strujnih (4-20mA) vrijednosti koji predstavljaju fizikalne veličine (temperatura, brzina, tlak, protok i sl.) u rasponu u kojem se mjere odgovarajućim uređajima. [7]

Dio PLC-a koji je zadužen za čitanje ulaznih vrijednosti, izvršavanje programa, raspodjelom memorije i upravljanje izlaznim uređajima je CPU. Brzina i snaga procesuiranja određuju performanse PLC-a. Na Slika 4.1.1 je prikazan Siemens PLC serije S7-1500 koji je korišten u ovom radu.

Slika 4.1.1. PLC 1516-3 PN/DP [8]

Točan model CPU-a je 1516-3 PN/DP. Dolazi sa većim LCD zaslonom koji omogućuje konfiguriranje. Ima dva PROFINET ulaza te jedan PROFIBUS, što omogućuje povezivanje više od jednog uređaja. Još bi bilo bitno spomenuti LED statusne lampice koje indiciraju različita stanja ovisno o načinu na koji svijetle. Tako postoje RUN/STOP LED, zeleno indicira *run mode* – način u kojem se izvršava program, dok žuto *stop mode* – način kada se program ne izvršava te je onemogućeno slanje izlaznih signala. ERROR LED indicira na pojavu neke greške kada se upali crveno svijetlo. Posljednja LED je MAINT koja žutom bojom signalizira da je potrebno napraviti neku vrstu održavanja.[9]

Kako bi CPU izvršio potrebnu komunikaciju sa željenim ulaznim i izlaznim uređajima potrebni su ulazni i izlazni moduli. Moduli mogu biti analogni i digitalni, tako postoje AI, AO, DI, DQ, DIQ moduli i sl. Jedna od glavnih odlika PLC-a je da su modularni, što znači da se mogu proširivati i prilagođavati dodavanjem modula prema potrebi.

Komunikacijski modul je još jedna vrsta modula koja je uvijek prisutna u svakom PLC-u. Uloga ovog modula je mogućnost povezivanja i vršenja razmjene podataka sa ostalim uređajima putem različitih vrsta portova. Portovi mogu biti tipa Ethernet (PROFINET/PROFIBUS protokoli), USB, RS-232 i slično. [7]

Raspodjela memorijom je isto tako jedna od zadaća CPU-a koju treba izvršiti. Memorija PLC-a je podijeljena u nekoliko vrsta. *Load memory* je trajna memorija za pohranu korisničkog programa, data blokova te hardver konfiguracije, a nalazi se na SIMATIC memorijskoj kartici. *Work memory* se koristi kada je CPU u radu te je nije moguće nadograditi za razliku od *Load memory*. Postoji još takozvana *retentive memory*, u koju se nakon gubitka napajanja pohranjuju bitne informacije za siguran nastavak rada.[10]

CPU i ostale komponente PLC-a zahtijevaju konstantnu i stabilnu električnu energiju. To je osigurano različitim vrstama napajanja, ovisno o primjeni PLC-a, koje pretvaraju 230V AC u 24V DC.

Programiranje PLC-a se uglavnom zasniva na računalnim softverima gdje se koriste programski jezici kako bi se ostvarila željena logička operacija. Programski jezik koji je prvobitno nastao na temelju relejnih shema naziva se ljestvičasti dijagram (eng. Ladder Diagram). Međutim, kasnije su se pojavili i drugi jezici što će detaljnije biti u sljedećem poglavlju gdje će se opisivati TIA Portal, standardni softverski alat za programiranje i konfiguraciju Siemens PLC-a. [7]

PLC u sustavu kontrole pozicije ima najznačajniju ulogu, kao što se može vidjeti sa Slika 4.1. Obuhvaća razvoj algoritma pozicionera gdje kao izlaznu vrijednost brzinu šalje prema frekvencijskom pretvaraču. Teorijsku poziciju dobivenu sa pozicionera uspoređuje sa trenutnom pozicijom te se na temelju toga programira regulacijsko djelovanje što predstavlja blok *PI*

Regulator. Isto tako će se provjeriti rad regulatora simuliranjem smetnji, odnosno poremećajnih veličina koje nisu neočekivane u realnim okolnostima.

4.2 Frekvencijski pretvarač

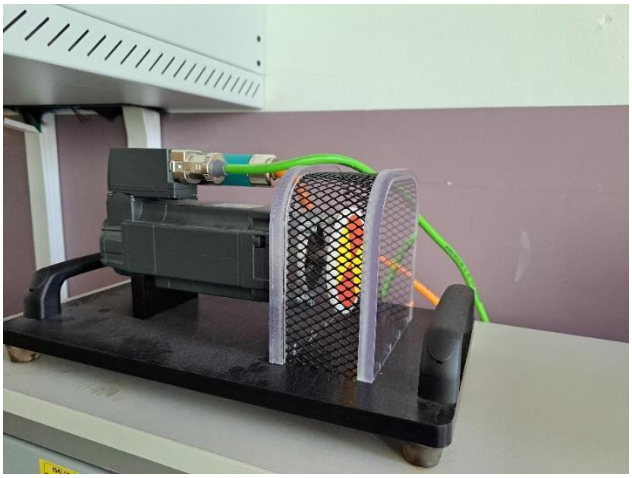
Frekvencijski se pretvarač koristi kada imamo zahtjev za promjenjivom brzinom motora. Radi na principu promjene napona i frekvencije ulaznog signala kako bi dobili promjenjivu brzinu. Upravljački dio pretvarača je SINAMICS DRIVE kojim se i uspostavlja komunikacija sa PLC-om te tako pruža razmjenu podataka. Podaci koji se razmjenjuju definirani su putem takozvanih telegrama. Telegram se sastoji od riječi za slanje i primanje podataka. Šalju se referenca brzine i naredbe za pokretanje i zaustavljanje motora, dok se primaju statusi od pretvarača te stvarna brzina motora. Telegram će biti detaljnije opisan kod konfiguracije komunikacije. Kako bi se omogućilo upravljanje brzinom u zatvorenoj petlji, pretvarač je opremljen enkoderom kojim se očitava trenutna brzina motora putem binarnog.
Korišteni model pretvarača je SINAMICS- S120, prikazan na Slika 4.2.1.

Slika 4.2.1. Frekvencijski pretvarač SINAMICS-S120

Jedna od uloge frekvencijskog pretvarača u kontroli pozicije je upravljanje motorom koji omogućuje na siguran i precizan način upravljanje promjenjivom brzinom vrtnje. Sljedeća uloga je djelovanje kao mjerni član u sustavu regulacije koji na temelju povratne informacije o trenutnoj brzini vrtnje daje uvid i o trenutnoj poziciji (dobivena integracijom brzine).

4.3 Servo motor

Servo motor oponaša ulogu aktuatora u sustavu kontrole pozicije sa bidirekcijskim pozicionerom te je povezan sa frekvencijskim pretvaračem.
Servo motor korišten u kontroli pozicije prikazan je na Slika 4.3.1.



Slika 4.3.1. Servo motor

4.4 Human machine interface (HMI)

HMI se koristi kao uređaj za interakciju između čovjeka i stroja pri upravljanju i nadzoru nekog procesa. Sastoji se od ekrana koji je osjetljiv na dodir i softverskog dijela kojim se ostvaruje komunikacija sa ostalim uređajima. Na zaslonu su grafički prikazana tipkala, sklopke, lampice, animacije, grafovi i slično. Operater ima mogućnost upravljanja procesom, a isto tako prima povratne informacije koje se ključne pri nadzoru nekog procesa – dojava alarmnog stanja, stanja o greškama (eng. Fault), itd. Model HMI-a koji se koristio prikazan je na Slika 4.4.1, a ima oznaku KP700 Basic PN, 7“ zaslon i razlučivost 800x480.

Slika 4.4.1. HMI KP700 Basic PN [11]

Uloga HMI-a u sustavu kontrole pozicije odnosi se na unosu ulaznih parametara od strane operatera, koji se onda putem komunikacije šalju na PLC. Isto tako jedna od uloga je i nadzor procesa koji podrazumijeva prikaz svi statusnih i procesnih veličina ključnih u izvođenju. Neke od njih su informacije o frekvencijskom pretvaraču, informacije o pozicioneru (pozicija, brzina, run/stop stanje), grafički prikaz brzine i pozicije u vremenu.

4.5 Računalo – softverski alat

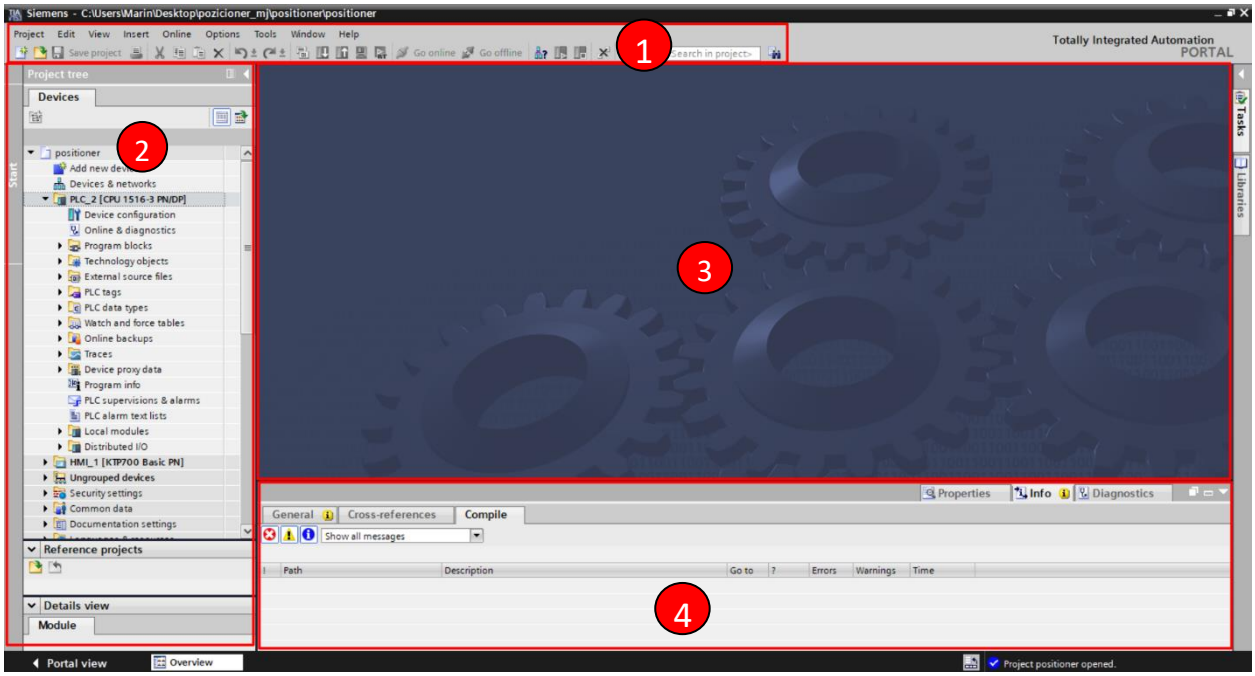
Posljednji, ali isto tako neizostavan element ovog sustava je računalo. Iako ne sudjeluje direktno u upravljanju procesom, veoma je bitno u početnoj fazi razvoja. Razvoj sustava upravljanja podrazumijeva programiranje i konfiguraciju PLC-a, izradu vizualizacije za HMI te uspostavu komunikacije između PLC-a, HMI-a i frekvencijskog pretvarača. U nekim se slučajevima koristi za nadzor i dijagnosticiranje problema rada ostalih komponenata. Spomenute radnje dostupne su unutar specifičnih softverskih alata koji se koriste u svijetu automatizacije. U ovom se radu služilo prije već spomenutim TIA Portalom koji će biti tema sljedećeg poglavlja.

5 SOFTVERSKO OKRUŽENJE – TIA PORTAL

TIA (Totally Integrated Automation) Portal je softverski alat koji omogućuje programiranje PLC uređaja u različitim programskim jezicima, dijagnosticiranje problema rada praćenjem izvršavanja koda u stvarnom vremenu, konfiguraciju i komunikaciju sa ostalim uređajima spojenih na mrežu, izradu vizualizacije, itd.

5.1 Uvod u glavne značajke

Prikaz projektnog zaslona TIA Portal prikazan je na Slika 5.1.1.



Slika 5.1.1. Prikaz projektnog zaslona TIA Portal-a

Numerirane sekcije sa slike označuju sljedeće:

1. Glavna traka izbornika te alatna traka
2. Projektno stablo (eng. Project tree)
3. Radni prostor
4. *Inspector window*

Glavna traka izbornika sadrži sve funkcije koje su potrebne u radu kao što je kreiranje i spremanje projekta, opcije uređivanja projekta, mogućnosti upravljanja prikazom unutar programa, online/offline interakcija sa PLC-om, podrška i slično.

Alatna traka sadrži ikone za često korištene funkcije, npr. skidanje programa na PLC, *compile* programa, online/offline način prikaza, pokretanje simulacijskog načina rada, run/stop način rada CPU-a, itd.

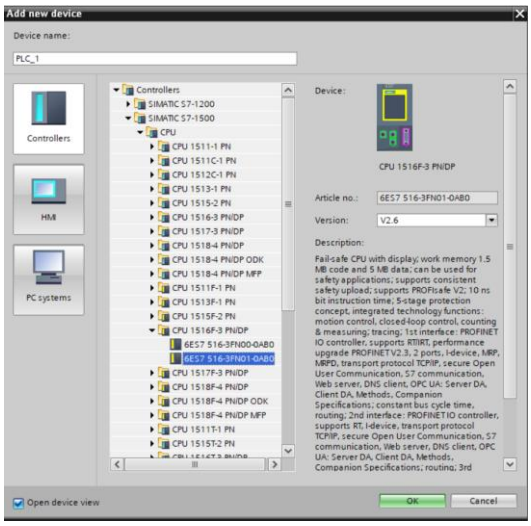
Projektno stablo sadrži sve informacije koje se tiču uređaja korištenih u projektu, komunikaciju između njih, dokumentaciju projekta, sigurnosne postavke itd... Informacije uređaja se odnose na njihovu konfiguraciju, programske blokove, tagove, HMI zaslone i slično.

Radni prostor služi za prikazivanje otvorenih naredbi i funkcija koje se izvode.

Inspector window pruža dodatne informacije i postavke o odabranim objektima, informacijski prozor izvršenim radnjama te dijagnostički prozor. [5]

5.2 Konfiguracija uređaja i uspostava komunikacije

Uređaje koji će se koristiti u projektu moguće je dodati iz dostupnog kataloga putem navigacijskog stabla. Postupak odabira uređaja prikazan je na Slika 5.2.1 na primjeru PLC-a korištenog u ovom radu.



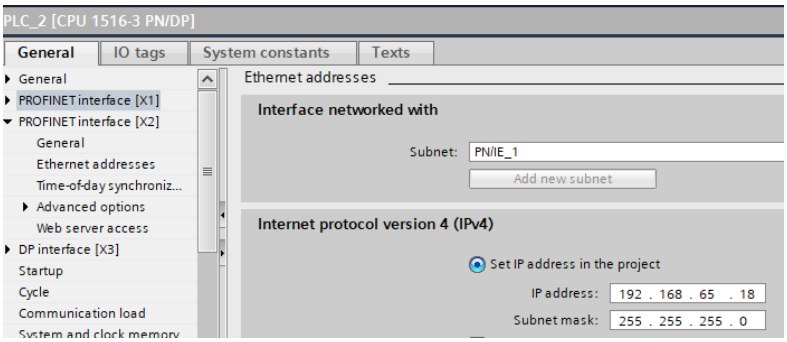
Slika 5.2.1. Katalog za odabir potrebnog PLC uređaja

Isti princip vrijedi i za HMI uređaje i ostale uređaje koji su dostupni u katalogu, u ovom će se slučaju koristiti još frekvencijski pretvarač.

Pristup konfiguraciji uređaja moguće je isto tako putem navigacijskog stabla preko polja „Devices and Networks“. U ovoj se sekciji otvara prikaz sa svim prisutnim uređajima u projektu. Kako bi se pristupilo konfiguraciji uređaja, odabere se uređaj uz prikaz dodatnih postavki u *Inspector window-u.*

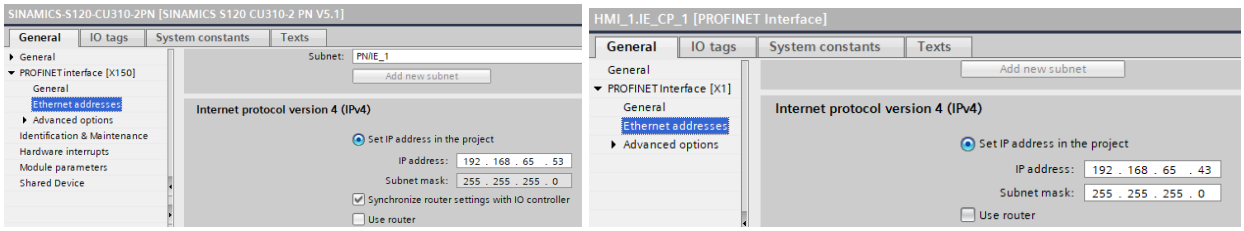
U slučaju PLC-a ovdje je moguće konfigurirati razne postavke portova koji su prisutni, postavke sigurnosti podataka, pristup web serveru, podešavanje cikličkog izvršavanja programa te generalni uvid u sve postavke koje su dostupne za konfiguraciju. Uređaji upotrebljeni u ovom

radu povezani su preko mreže, u kojoj svaki uređaj ima jedinstvenu IP adresu. Podešavanje IP adrese prikazano je na Slika 5.2.2 u slučaju PLC-a.



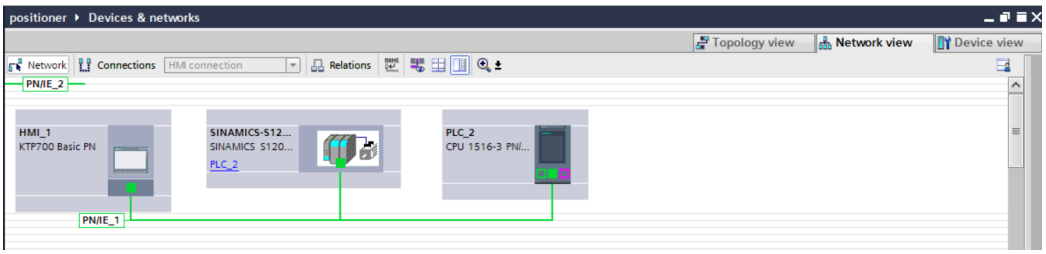
Slika 5.2.2. Podešavanje IP adrese PLC-a

Isti stvar vrijedi za postavljanje IP adrese HMI-a i pretvarača što je prikazano na Slika 5.2.3.



Slika 5.2.3. Podešavanje IP adrese HMI-a i frekvencijskog pretvarača

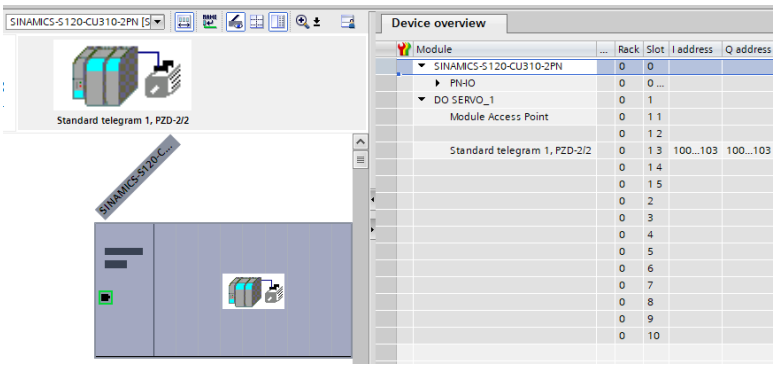
Kada su IP adrese uređaja podešene, kako bi se ostvarila komunikacija između njih, potrebno ih je povezati (*„drag and drop“* metoda) u *Network* prikazu sekcije „Devices and Networks“. Uspješno uspostavljena komunikacija prikazana je na Slika 5.2.4.



Slika 5.2.4. Uspješna komunikacija između uređaja u Network view-u

Nakon što je omogućena komunikacija između PLC-a i frekvencijskog pretvarača, potrebno je definirati koja vrsta podataka će se razmjenjivati. Sljedeće je izvedivo ako se dodaju posebni moduli u konfiguraciji pretvarača. Potrebno je odabrati uređaj SINAMICS-S120 te zatim *Device view*. Ovaj prikaz omogućuje dodavanje novih modula po potrebi te upravljanje njihovim adresama. Moduli sadrže različite vrste telegrama, a telegram je struktura koja razmjenjuje

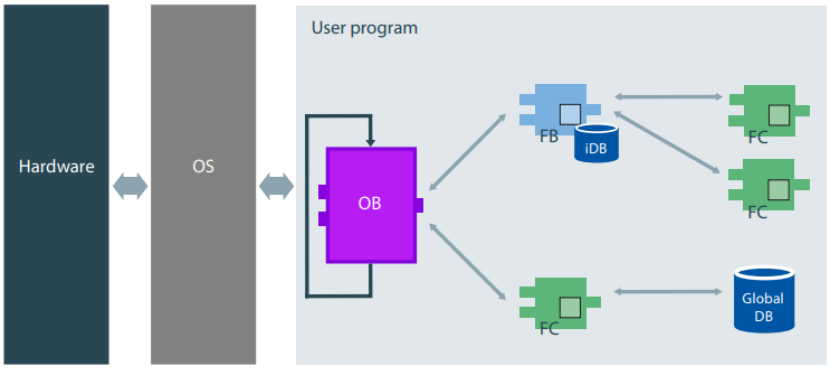
parametre bitne za upravljanje između PLC-a i pretvarača. Razmjena se odvija putem PROFINET protokola čija je konfiguracija u prethodnim koracima postavljena. Parametre koji su potrebni za prijenos, odnosno telegram koji se koristi za razmjenu, odabire se na temelju aplikacije za koju se pretvarač koristi. U ovom radu pretvarač upravlja brzinom vrtnje motora, tako da je potreban telegram koji će sadržavati parametre o brzini. Telegram pod nazivom „Standard telegram 1“ upravo razmjenjuje sa PLC-om referentnu brzinu, trenutnu brzinu, te kontrolnu i statusnu riječ. Detaljna struktura telegrama će biti opisana u sljedećem poglavlju. Na Slika 5.2.5 se može vidjeti konfiguriran pretvarač sa potrebnim modulima u vidu komunikacije sa PLC-om.



Slika 5.2.5. Moduli i podmoduli korišteni kod pretvarača

5.3 Programiranje PLC-a

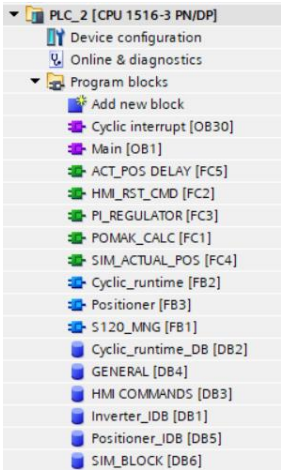
Programiranje PLC-a odnosi se na međusobnu interakciju operacijskog sustava (OS), korisničkog programa i hardvera. Njihov odnos je prikazan na Slika 5.3.1.



Slika 5.3.1. Odnos hardvera, OS i korisničkog programa [5]

Uloga operacijskog sustava je povezivanje korisničkog programa sa hardverom te se brine o upravljanju memorijom, obradi grešaka te organizacijom i pozivanjem svih funkcija. Korisnički program se sastoji od blokova u kojima se koristeći neki od programskih jezika izrađuje logičko rješenje problema.

Prije nego se objasni svaki od programskih blokova, ukratko će biti opisani programski jezici. Postoje nekoliko opcija kada je u pitanju programiranje kontrolera. Odabir programskog jezika ovisi o aplikaciji i preferenci programera. Ladder dijagram (LD) je grafički programski jezik kod kojeg se programiranje vrši povezivanjem kontakata (eng. contact) i zavojnica (eng. coil) što podsjeća na relejne sheme koje su se koristile prije računalnog programiranja. Structured Control Language (SCL) se koristi za nešto kompleksnije zadatake koje je potrebno realizirati, korištenjem funkcija, petlji i slično. Nalikuje na C programski jezik. Structured Text (STL) isto je tako tekstualnog oblika namijenjen za složenije matematičke operacije ili stvaranje posebnih uvjeta. Function Block Diagram (FBD) je još jedan grafički programski jezik koji je sastavljen od blokova koji predstavljaju neku funkciju ili operaciju, te ih je moguće povezati međusobno kako bi se stvorila neka vrsta logike. Prednost koju ima serija 1500 PLC-a je optimizacija strojnog koda, kojim se postiže direktno prevođenje bilo kojeg programskog jezika u strojni kod. Ovim postupkom nije potrebno izvršavati dodatne pretvorbe između jezika kako bi se dobio strojni kod te tako postoji jednak stupanj izvođenja svakog od programskih jezika. [5]
Već smo spomenuli kako se prilikom izrade programa služimo blokovima, kojih ima nekoliko vrsta kako bi program bio strukturiran i pregledan za čitanje ili detektiranje pogrešaka. Na Slika 5.3.2 su prikazani svi programski blokovi koji se koriste u ovom projektu.



Slika 5.3.2. Programski blokovi projekta

Organizacijski blokovi (OB) su poveznica između korisničkog programa i operacijskog sustava. Postoje različiti tipovi ovih blokova, najčešći je „Program cycle“ koji se ciklički izvršava svo vrijeme (OB1 na Slika 5.3.2). Koristi se za klasične instrukcije i naredbe kod kojih nisu potrebni posebni uvjeti. S druge strane ako želimo pozvati neki program u određenim vremenskim intervalima to je moguće putem „Cyclic interrupt“ bloka (OB30 na Slika 5.3.2). Postoji još i

„Startup“ blok koji se poziva samo jednom kod pokretanja PLC iz *stop* u *run* stanje, blokovi koji se pozivaju kada se pojave neke vrste pogreške i slično.

Global data blokovi (DB) spremaju sve potrebne varijable koje se mogu koristiti u cijelom korisničkom programu, odnosno u drugim programskim blokovima.

Funkcije (FC) su blokovi koji nemaju svoju memoriju u smislu *data* bloka. Sastoje se od ulaznih vrijednosti i programskog koda na temelju kojih vraćaju izlazne vrijednosti. Koriste se kada se neka funckija više puta ponavlja na različitim mjestima u programu te ih je potrebno pozvati u OB bloku.

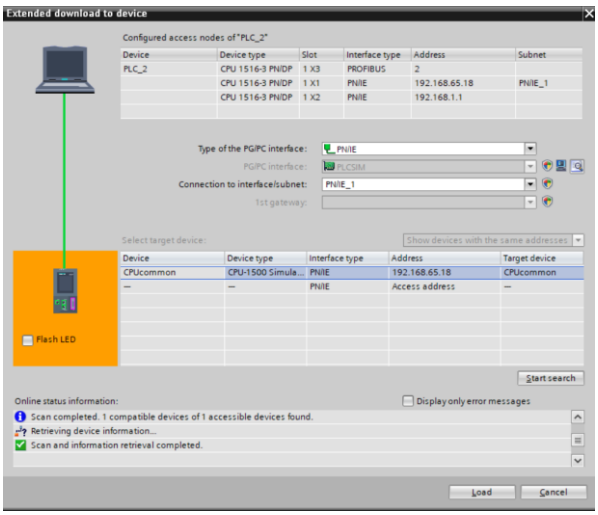
Funkcijski blokovi (FB) za razliku od FC imaju svoju memoriju pod nazivom Instance DB (iDB). IDB služi za spremanje vrijednosti korištenih varijabli (input, output, inout, static) i nakon pozivanje bloka. Ako se žele koristiti u programu, potrebno ih je, kao i FC, pozvati u OB bloku.

Svaka varijabla korištena u DB zapisana je u obliku memorije koja predstavlja veličinu te varijable i sadržaj koji se zapisuje. Time se na osnovu veličine memorije koja se koristi mogu definirati tipovi podataka. Tip podatka koji koristi samo jedan bit memorije naziva se BOOL. Poprima vrijednosti 0 ili 1, gdje je 0=false, a 1=true. Sljedeći tip podataka temelji se na skupu osam bitova, a naziva se BYTE. Nadalje, skup od 2 BYTE podatka, odnosno 16 bitova naziva se WORD. Integer (INT) je iste memorijske veličine kao i WORD, međutim razlika je što INT predstavlja zapis vrijednosti sa predznakom. Vrijedi još spomenuti tip podataka REAL, koji je skup 32 bita, a karakterizira ga svojstvo kojim je moguć zapisa brojeva sa decimalnom točkom i predznakom. [5]

Detaljan postupak programiranja PLC-a bit će prikazan u sljedećem poglavlju koje se odnosi na razvoj programskog rješenja sustava kontrole pozicije.

5.4 Preuzimanje programa na PLC i *monitoring* način rada

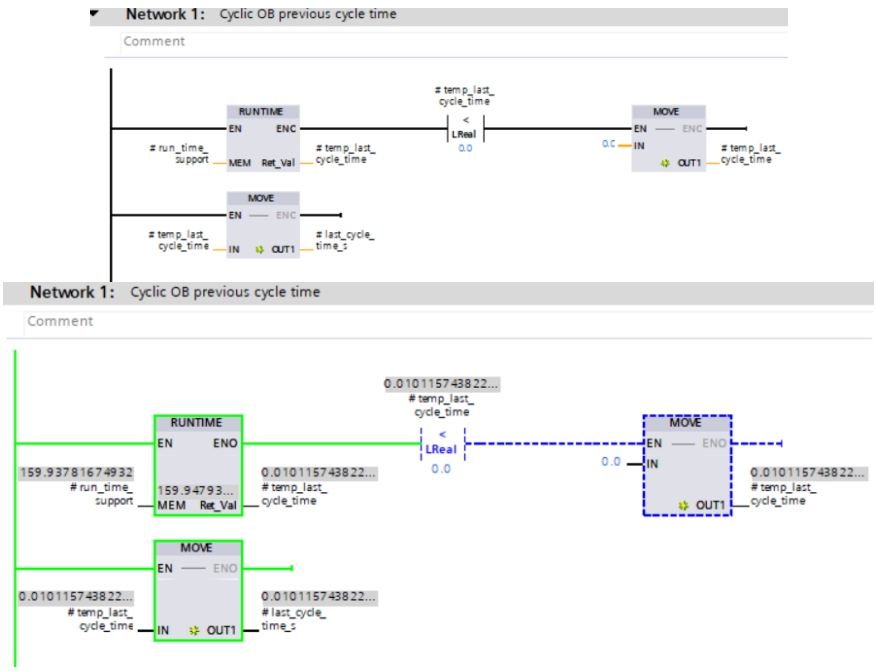
Verziju programa koja je napisana u projektu potrebno je preuzeti na PLC kako bi se napravljene promjene ostvarile. *Download to device* ikona alatne trake služi za navedeno te se tada otvara prozor prikazan na Slika 5.4.1.



Slika 5.4.1. Preuzimanje programa na PLC

U prozoru sa Slika 5.4.1 se odabire PLC na koji se žele preuzeti podaci projekta. Preuzimanje programa na PLC izvršava se uvijek kada je prisutna modifikacija projekta u odnosu na onu verziju projekta koji je zadnji put preuzet na PLC.

Monitoring način rada podrazumijeva promatranje izvođenja programa u stvarnom vremenu. Ovaj način rada je predviđen samo za čitanje te je potrebno da se verzija programa u TIA Portalu i ona preuzeta na PLC-u podudaraju. *Monitoring* je pogodan za dijagnostiku te nadzor nad stanjima varijabli tijekom izvršavanja. Na sljedećoj Slika 5.4.2 je prikazana usporedba jednog segmenta koda (napisanog u LD) u offline i online/monitoring načinu rada.



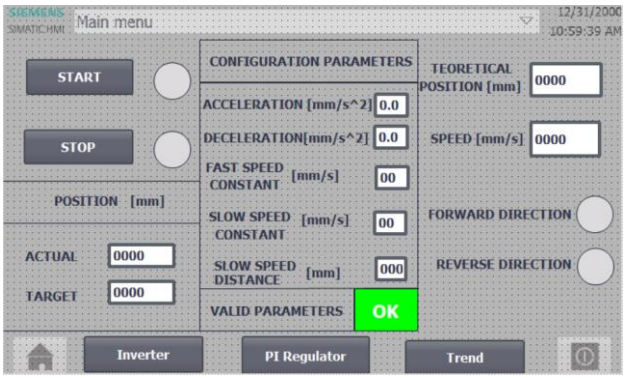
Slika 5.4.2. Usporedba offline i monitoring načina rada

Može se primijetiti kako su zelenom bojom prikazani elementi i linije koje se izvršavaju, dok su plavom isprekidanom prikazani neaktivni elementi. Također prisutne su i vrijednosti svake varijable, za razliku od offline način rada.

5.5 Vizualizacija HMI-a

Kreiranje i uređivanje zaslona s ciljem kontrole te nadzora procesa može se smatrati izradom vizualizacije HMI-a. Nakon dodavanja HMI uređaja iz kataloga u projekt, pojavljuje se sekcija u navigacijskom stablu sa svim funkcijama i objektima koji se koriste prilikom izrade vizalizacije. U funkciji *Screens* moguće je dodavati i pregledavati zaslone. Primjer jednog zaslona iz ovog rada je prikazan na Slika 5.5.1. Sastoji se od tekstualnog zapisa, *button*-a (start i stop), raznih geometrijskih oblika, polja za unos ili polja za prikaz, padajućeg izbornika između zaslona, datuma i vremena.

Osim navedenih elemenata sa Slika 5.5.1, dodaju se često i trendovi koji prikazuju promjene neke procesne veličine u vremenu.



Slika 5.5.1. Primjer uređivanja zaslona

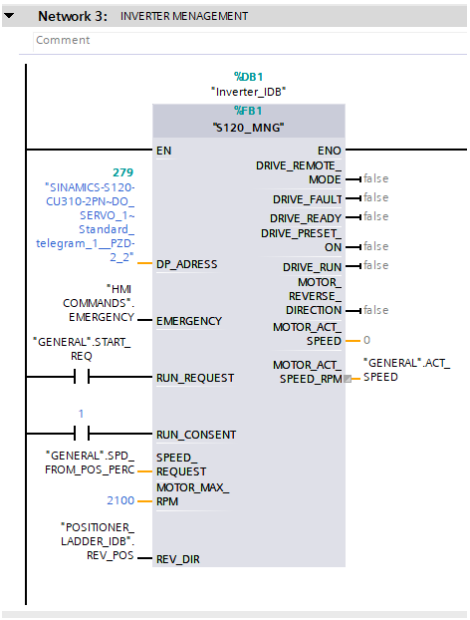
Svaki ovaj element obično se definira ili upravlja nekim HMI tagom. HMI tagovi mogu biti vanjski ili unutarnji. Unutarnji se odnose samo na prijenos podataka unutar HMI dijela. S druge strane, vanjski tagovi koriste se u komunikaciji sa PLC-om. Takav se tag tada služi memorijom varijable sa kojom je povezan u PLC-u. HMI tagovi se dodjeljuju u dodatnim postavkama svakog od elementa, gdje se još omogućuje velik broj opcija uređivanja, animacija, upravljanja prikazom i slično.

6 PROGRAMSKO RJEŠENJE

Programsko rješenje odnosi se na razvoj algoritma u TIA Portalu s ciljem ostvarivanja sustava kontrole pozicije. Ono obuhvaća programiranje PLC-a te izradu vizualizacije HMI-a. Tako će u ovom poglavlju programiranje biti podijeljeno u cjeline od kojih se svaka odnosi na jedan element upravljanja.

6.1 Upravljanje frekvencijskim pretvaračem

Upravljanje frekvencijskim pretvaračem izvršeno je pomoću FB bloka, sa Slika 6.1.1. Ulazne varijable predstavljaju naredbe koje se šalju prema pretvaraču, dok izlazne predstavljaju statusne i procesne vrijednosti koje se primaju sa pretvarača.



Slika 6.1.1. Funkcijski blok upravljanja frekvencijskog pretvarača

Kreiranje FB bloka, stvara Instance DB koji služi kao unutarnja memorija ovom bloku u koji se zapisuju sve korištene varijable. Prve varijable koje je potrebno zapisati su one koje predstavljaju telegram, odnosno koje služe u komunikaciji između PLC-a i pretvarača. Telegram koji se koristi, spomenut kod uspostave komunikacije, je Standard telegram 1 PZD-2/2. PZD je izraz koji se koristi za procesne podatke koji se razmjenjuju u komunikaciji. Oznaka 2/2 označava količinu podataka koji se šalju u obliku *word*, u ovom slučaju se šalju i primaju dva *word* tipa podatka. Dakle, telegram je struktura koja opisuje koji se podaci razmjenjuju s pretvaračem u obliku *word* tipa podatka, a moguće ju je pronaći u priručniku pretvarača. [12] Slanje podataka definirano je prvim *word*-om kojim se šalje referentna brzina vrtnje, dok je drugi *word* podijeljen

u 16 bitova te se naziva komandna riječ. Ove podatke potrebno je zapisati u iDB, što je prikazano na Slika 6.1.2.

Inverter_IDB

	Name	Data ...	Start value	Retain	Accessible f...	Writa...	Visible in ...	Se..	Sup...	Comment
19	Static									
20	PLC->DRIVE	Struct								
21	DriveControl0	Struct								
22	08_RESERVED	Bool	false							
23	09_RESERVED_1	Bool	false							
24	10_CNTRL_BY_PLC	Bool	false							control by PLC/no control by PLC
25	11_SETPNT_INV	Bool	false							setpoint inversion 1=reverse direction 0=forward direction
26	12_RESERVED_2	Bool	false							
27	13_M_POTENT_STP_RAISE	Bool	false							motorized potentiometar setpoint raise
28	14_M_POTENT_LWR_STP	Bool	false							motorized potentiometar lower setpoint
29	15_RESERVED_3	Bool	false							
30	00_ON_OFF	Bool	false							ON/OFF (OFF1)
31	01_NO_COAST_DWN	Bool	false							no coast-down/ coast-down (OFF2)
32	02_NO_QUICK_STP	Bool	false							no quick stop/ quick stop(OFF3)
33	03_ENBL_OP	Bool	false							enable operation/inhibit operation
34	04_ENBL_RAMP	Bool	false							enable ramp-function generator/inhibit ramp-function gener
35	05_CONTINUE_RAMP	Bool	false							continue ramp-function generator/freeze ramp-function gen
36	06_ENBL_SETPOINT	Bool	false							enable setpoint/inhibit setpoint
37	07_ACKNOW_FLT	Bool	false							acknowledge faults
38	DriveControl1	Struct								
39	MAIN_STPT	Int	0							

Slika 6.1.2. IDB zapisanih varijabli za komunikaciji u smjeru PLC-> pretvarač

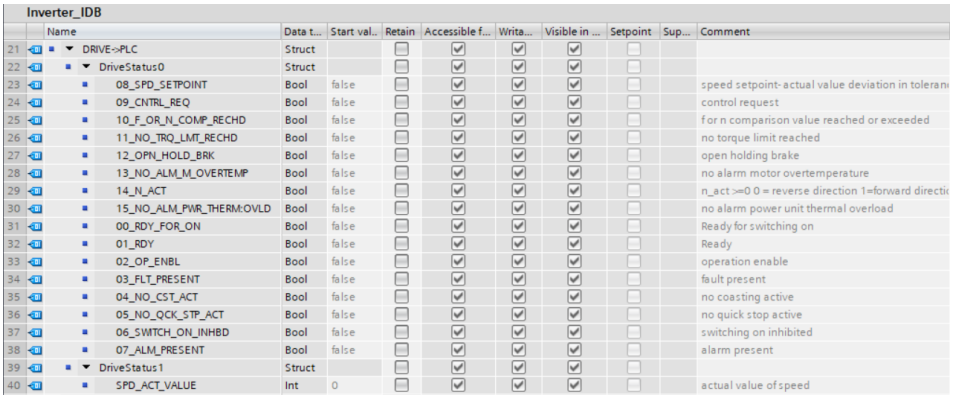
Opis svake varijable zasebno dan je u Tablica 6.1.1.

Tablica 6.1.1. Opis varijabli u PLC -> pretvarač komunikaciji [12]

STW1		
Bit	**Opis**	**Pojašnjenje**
0	ON/OFF (OFF1)	Uključivanje motora- ready stanje uz bit 3/Isključivanje motora putem rampe
1	No coast-down / coast-down (OFF2)	Isključivanje motora momentalno/ Omogućeno uključivanje motora
2	No Quick Stop / Quick Stop (OFF3)	Isključivanje motora uz rampu / Omogućeno uključivanje motora
3	Enable/inhibit operation	Isključivanje motora(otkazivanje impulsa) / Uključivanje motora (omogućivanje impulsa)
4	Enable/inhibit ramp function generator	Omogućivanje ili onemogućivanje RFG
5	Continue/freeze ramp function generator	Izlaz iz RFG-a prati referencu ili staje na trenutnoj vrijednosti
6	Enable/inhibit setpoint	Omogućavanje reference/zaustavljanje motora
7	Acknowledge faults	Slanje greške
8	Reserved	-
9	Reserved	-
10	Control by PLC/no control by PLC	Upravljanje inverterom putem PLC-a/kontrola putem PLC-a zanemarena
11	Setpoint inversion	Vrtnja motora u suprotnom smjeru
12	Reserved	-
13	Motorized potentiometer setpoint raise	Povećanje reference u motoriziranom potenciometru

14	Motorized potentiometer lower setpoint	Smanjenje reference u motoriziranom potenciometru
15	Reserved	-
NSOLL_A		
INT	Speed setpoint	Brzina vrtnje motora

S druge strane, podaci koji se primaju sa pretvarača zapisani su također u istom iDB-u te prikazani na Slika 6.1.3.


Inverter_IDB

	Name	Data t...	Start val...	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Sup...	Comment
21	DRIVE->PLC	Struct			✓	✓	✓			
22	DriveStatus0	Struct			✓	✓	✓			
23	08_SPD_SETPOINT	Bool	false		✓	✓	✓			speed setpoint- actual value deviation in toleran
24	09_CNTRL_REQ	Bool	false		✓	✓	✓			control request
25	10_F_OR_N_COMP_RECHD	Bool	false		✓	✓	✓			f or n comparison value reached or exceeded
26	11_NO_TRQ_LMT_RECHD	Bool	false		✓	✓	✓			no torque limit reached
27	12_OPN_HOLD_BRK	Bool	false		✓	✓	✓			open holding brake
28	13_NO_ALM_M_OVERTEMP	Bool	false		✓	✓	✓			no alarm motor overtemperature
29	14_N_ACT	Bool	false		✓	✓	✓			n_act >=0 0 = reverse direction 1=forward directi
30	15_NO_ALM_PWR_THERM:OVLD	Bool	false		✓	✓	✓			no alarm power unit thermal overload
31	00_RDY_FOR_ON	Bool	false		✓	✓	✓			Ready for switching on
32	01_RDY	Bool	false		✓	✓	✓			Ready
33	02_OP_ENBL	Bool	false		✓	✓	✓			operation enable
34	03_FLT_PRESENT	Bool	false		✓	✓	✓			fault present
35	04_NO_CST_ACT	Bool	false		✓	✓	✓			no coasting active
36	05_NO_QCK_STP_ACT	Bool	false		✓	✓	✓			no quick stop active
37	06_SWITCH_ON_INHBD	Bool	false		✓	✓	✓			switching on inhibited
38	07_ALM_PRESENT	Bool	false		✓	✓	✓			alarm present
39	DriveStatus1	Struct			✓	✓	✓			
40	SPD_ACT_VALUE	Int	0		✓	✓	✓			actual value of speed

Slika 6.1.3. IDB zapisanih varijabli za komunikaciji u smjeru pretvarač-> PLC

Tablica 6.1.2 predstavlja opis svake varijable koji se odnosi na komunikaciju od pretvarača prema PLC-u.

Tablica 6.1.2. Opis varijabli u DRIVE-> PLC komunikaciji [12]

ZSW1		
Bit	**Opis**	**Pojašnjenje**
0	Ready for switching on	Prisutnost napajanja
1	Ready	OFF1=1, motor uključen
2	Operation enabled	Motor prima referencu te može pokrenuti
3	Fault present	Prisutano *fault* stanje
4	No coasting active	Nije omogućen *cost-down* do stanja mirovanja
5	No quick stop active	Nije aktivano brzo zaustavljanje
6	Switching on inhibited	Moguće je jedino pokrenuti motor promjenom stanja OFF1 u ON
7	Alarm present	Pojava alarmnog stanja
8	Speed setpoint – actual value deviation	Tolerancija devijacije brzine u zadanom rasponu
9	Control request	Zahtjev za kontrolom nad inverterom

10	F or n comparison value reached or exceeded	Brzina veća ili jednaka od maksimalne brzine
11	Torque limit reached	Vrijednost maksimalnog momenta ili struje je postignuta
12	Open holding brake	Otvaranje ili zatvaranje kočnice
13	No alarm motor overtemperature	-
14	Reverse or Forward direction	1=vrtnja u smjeru kazaljke, 0=vrtnja u suprotnom smjeru od kazaljke na satu
15	No alarm power unit thermal overload	-
NIST_A		
INT	Speed actual value	Trenutna brzina vrtnje motora

Nakon upisivanje svih potrebnih varijabli komunikacije sa pretvaračem, definiraju se ostale varijable koje služe kod programiranja ovog bloka, a prikazane su u Dodatku A u iDB-u. Pošto ulazne i izlazne varijable, takoreći opisuju ovaj blok, radi lakšeg razumijevanja njihovo pojašnjenje je navedeno u Tablica 6.1.3.

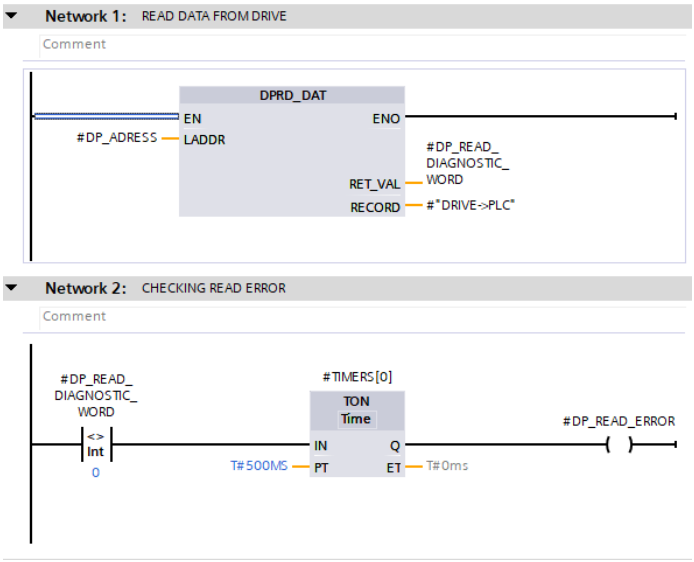
Tablica 6.1.3. Opis ulaznih i izlaznih varijabli pretvarača

Ulazne varijable	
Varijabla	**Pojašnjenje**
DP_ADRESS	Adresa hardvera koja predstavlja podmodul (telegram) sa kojeg se razmjenjuju podaci
EMERGENCY	Isklop motora u slučaju opasnosti
RUN_REQUEST	Zahtjev za pokretanjem motora
RUN_CONSENT	Dozvola za pokretanjem
SPD_FROM_POS_PERC	Referenca brzine u postotcima
MOTOR_MAX_RPM	Maksimalna brzina vrtnje u o/min
REV_DIR_CMD	Komanda za suprotni smjer vrtnje
Izlazne varijable	
DRIVE_REMOTE_MODE	Lokalna kontrola nad pretvaračem (*remote*)
DRIVE_FAULT	Prisustvo *fault* stanja
DRIVE_READY	Pretvarač je spreman za uključivanje
DRIVE_PRESET_ON	Motor je spreman za pokretanje
DRIVE_RUN	Motor je pokrenut
MOTOR_REVERSE_DIR	Suprotan smjer vrtnje prisutan
MOTOR_ACT_SPEED	Trenutna brzina vrtnje u %
MOTOR_ACT_SPEED_RPM	Trenutna brzina vrtnje u o/min

Nadalje će biti opisana struktura bloka pretvarača, odnosno način programiranja logičkih operacija u LD-u koristeći varijable komunikacije te varijable iz Dodatka A.

Programski blokovi su sastavljeni od *network*-a koji mogu sadržavati jednu ili više linija pomoću kojih je kod zapisan. Prvi *network* se odnosi na funkciju koja omogućava čitanje podataka sa

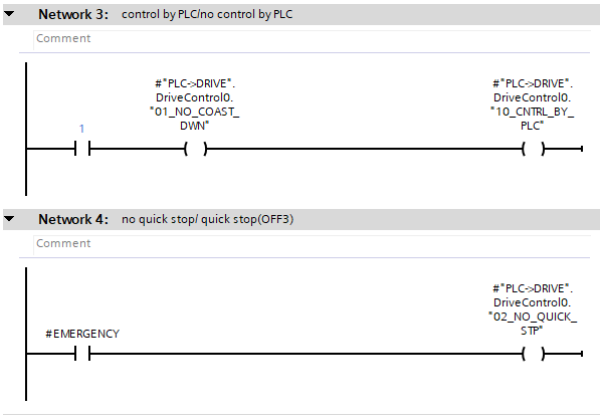
pretvarača. Potrebno je priložiti DP_ADRESS kako bi funkcija imala pristup hardver komponenti telegrama te strukturu DB-a u koju će se podaci sa pretvarača upisivati. Ako se pojavi pogreška u komunikaciji, funkcija je zapisuju pomoću izlaza RET_VAL. Drugi *network* se odnosi upravo na zapisivanje te pogreške. Zapisana je pomoću funkcije TON, koja predstavlja *timer*. *Timeri* su funkcije koje stvaraju vremensku odgodu u liniji u kojoj se koriste, a na temelju odgode kakvu stvaraju podijeljenu su u dvije vrste. TON *timer* aktivira izlaz nakon prolaska vremenske odgode (odgoda uključivanja), dok TOF *timer* odmah aktivira izlaz, ali ga drži aktivnim i kada ulazni signal više nije aktivan (odgoda isključivanja). Vrijeme odgode zapisano je u PT ulazu *timera*. Ako se vratimo na network 2, greška će se smatrati za valjanom te biti zapisana u pomoćnu varijablu DP_READ_ERROR ako je greška koju javlja funkcija DPRD_DAT aktivna 500ms. Prvi i drugi network prikazani su na Slika 6.1.4.



Slika 6.1.4. Instrukcija za čitanje podataka sa pretvarača te zapis pogreške u komunikaciji

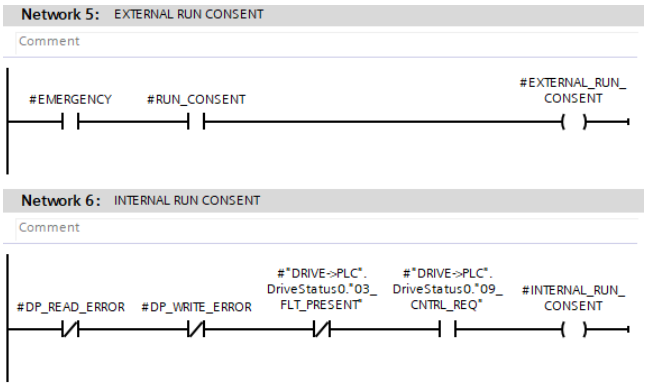
Putem sljedećih nekoliko network-a aktivirat će se bitovi iz komandne riječi (varijable strukture PLC->DRIVE u iDB-u).

Network 3 prikazuje aktiviranje bita 1 i bita 10. Ova dva bita su aktivna uvijek iz razloga što u ovoj primjeni pretvarača uvjet nije potreban. Network 4 prikazuje aktiviranje bita 2 uz prisutnu *EMERGENCY* varijablu. Network 3 i 4 su prikazani na Slika 6.1.5.



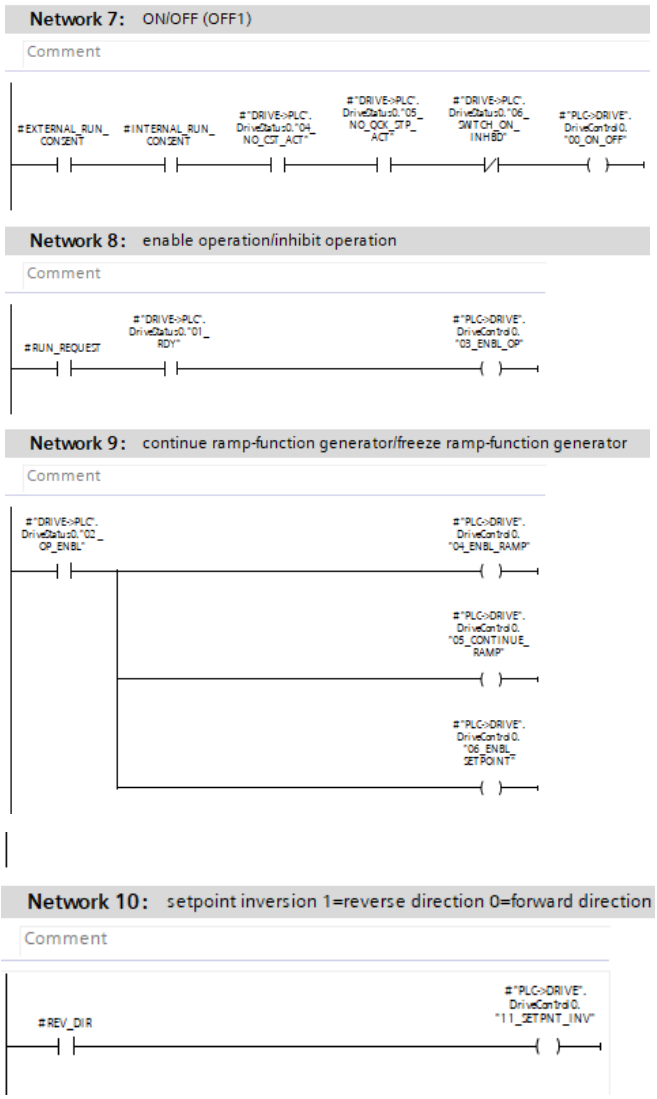
Slika 6.1.5. Uključivanje bitova za upravljanje pretvaračem

Nadalje definiramo varijable za vanjsko i unutarnje odobravanje pokretanja motora. EXTERNAL_RUN_CONSENT je aktivan ako je prisutna komanda za odobravanje te EMERGENCY. INTERNAL_RUN_CONSENT ovisi o neprisustvu grešaka u komunikaciji, neprisustvu *fault* stanja pretvarača te aktiviranog bita za kontrolu nad inverterom, što je prikazano na Slika 6.1.6.



Slika 6.1.6. Definiranje vanjskih i unutarnjih uvjeta odobravanja pokretanja

Kada su vanjski i unutarnji uvjeti za pokretanje zadovoljeni te pretvarač šalje da su svi potrebni bitovi uključeni, tek tada je moguće aktivirati glavni bit 0, za uključenje motora. Bit 3, koji omogućuje izvršavanje operacije (promjena brzina do reference) je uključen ako imamo ulaznu komandu za pokretanje aktivnu, te ako pretvarač vraća status *ready*. Tom aktiviranom komandom uključujemo i preostale bitove koji su potrebni za ispravno funkcioniranje, bit 4, 5 i 6. Ako je potrebno gibanje u suprotnom smjeru, uključuje se bit 11. Navedene linije koda prikazane su na Slika 6.1.7.



Slika 6.1.7. Uključivanje bita za pokretanje pretvaračem, bita za mogućnost slanje reference, preostalih bitova za ispravan način rada te uvjet za uključivanje bita suprotnog smjera

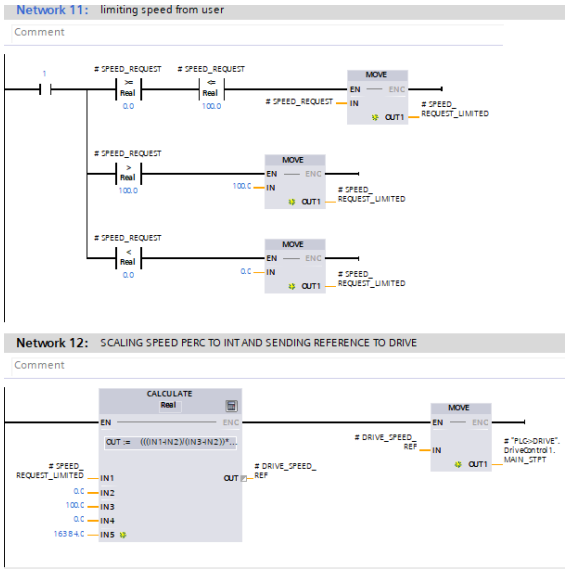
Kako bi se, zbog sigurnosti, brzina koja dolazi sa pozicionera zadržala u granicama od 0 do 100% potrebno je istu ograničiti, što je prikazano na Slika 6.1.8. Ako je brzina manja od 0, zapisuje se vrijednost 0, ako je veća od 100, zapisat će se vrijednost 100. Može se primijetiti kako se brzina koja se šalje prema pretvaraču, šalje samo u pozitivnim vrijednostima. Ako je potrebno gibati se u suprotnom smjeru (negativni o/min) uključuje se bit za suprotan smjer, uz pozitivnu referencu brzine. Međutim, brzina se direktno ne šalje prema pretvaraču u postotcima, nego kao INT tip podatka što se može vidjeti u iDB-u. Stoga se brzina zapisana u postotcima (0-100%) mora skalirati u INT (0-16384). Instrukcija koja omogućuje zapisivanje matematičkih operacija naziva se *Calculate*. Pomoću ove instrukcije se zapiše izraz 6.1.1 kojim se izvršava spomenuto skaliranje.

$$Drive\ speed\ INT = \left(\frac{act\ val - perc\ low}{perc\ high - perc\ low} * (int\ high - int\ low)\right) + int\ low \qquad (6.1.1),$$

gdje je:

drive speed INT	trenutna vrijednost brzine u INT,
act val	trenutnu vrijednost brzine u %,
perc low	minimalna vrijednost u %,
perc high	maksimalna vrijednost u %,
int low	minimalna vrijednost u INT
int high	maksimalna vrijednost u INT.

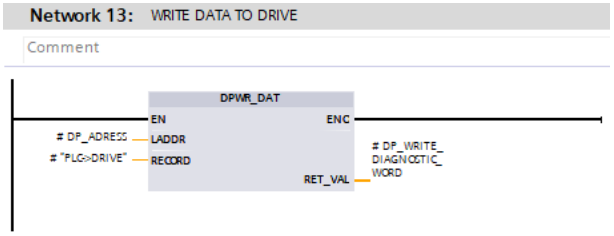
Kada je izvršeno skaliranje izlazna vrijednost iz instrukcije *Calculate* se preslikava u kontrolnu riječ reference brzine prema pretvaraču. Prethodno opisano je prikazano isto tako na Slika 6.1.8.



Slika 6.1.8. Ograničavanje, skaliranje i mapiranje brzine koja se šalje prema pretvaraču

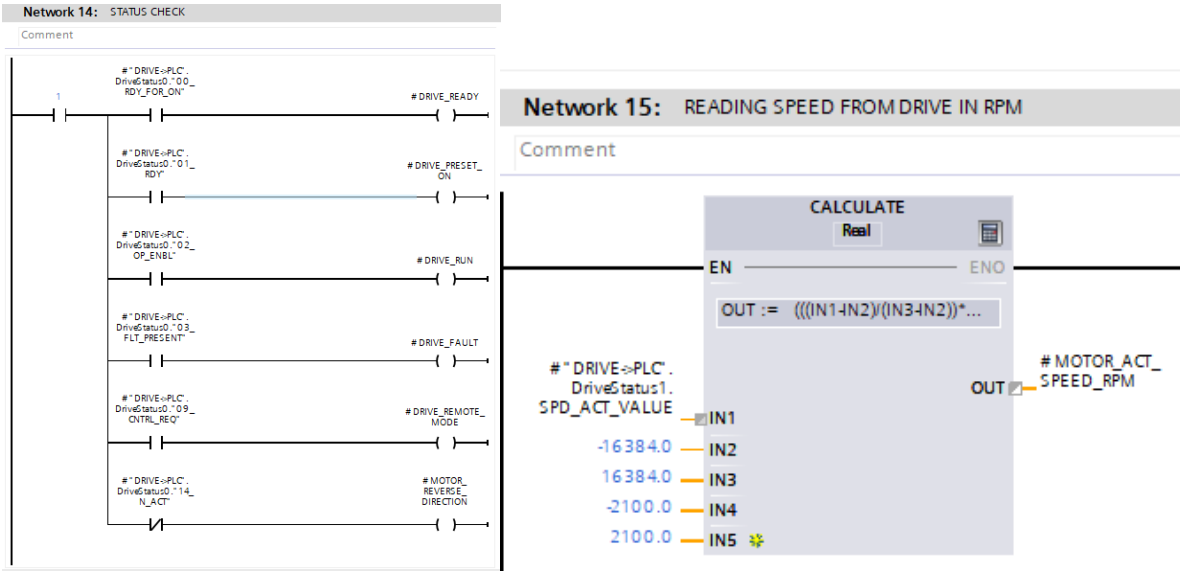
Ovakav raspored uključivanja bitova, odnosno slanja varijabli prema pretvaraču naziva se sekvencom upravljanja te je definirana u priručniku pretvarača. [12]

Nakon što se završilo sa zapisivanjem vrijednosti prema pretvaraču, potrebna je funkcija koja će te podatke poslati putem komunikacije, odnosno pomoću telegrama. Slično kao i sa primanjem podataka sa pretvarača, uvodi se funkcija DPWR_DAT kojoj je potrebno definirati hardver adresu na kojoj se nalazi telegram te struktura iBD-a u kojem su zapisane varijable u smjeru komunikacije PLC->pretvarač, što je prikazano na Slika 6.1.9.



Slika 6.1.9. Instrukcija za upisivanje varijabli koje se šalju prema pretvaraču

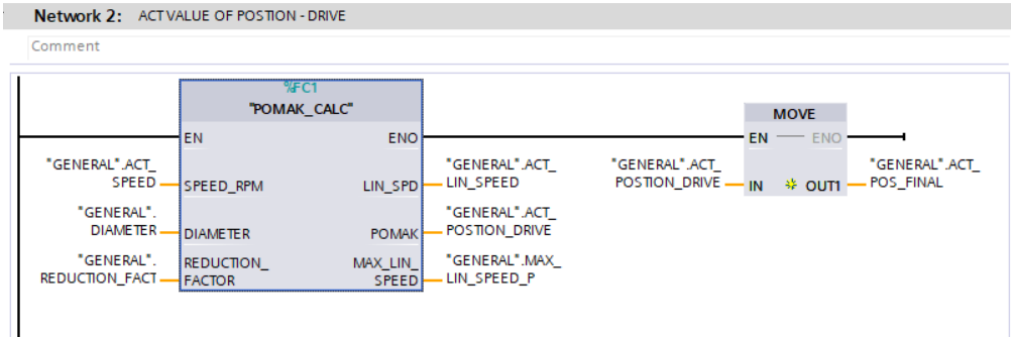
Nadalje, varijablama koje predstavljaju bitove prisutnih stanja pretvarača (statusna riječ) aktiviramo izlazne varijable iz FB-a. Tako je na Slika 6.1.10 prikazano dobivanje statusa *ready, preset, run, fault, remote mode* te *reverese direction*. Isto tako kako bi se dobila vrijednost trenutne brzine vrtnje koristi se statusna riječ SPD_ACT_VALUE, koja je INT tip podatka te je skaliramo po maksimalnom broju okretaja, odnosno od -2100 do 2100 o/min. Jednadžba kojom se skalira je već prikazana izrazom. Network u kojem se odvija spomenuto skaliranje je prikazan isto tako na Slika 6.1.10.



Slika 6.1.10. Čitanje statusnih bitova pretvarača te trenutne brzine

6.2 Izračun trenutne pozicije

Sljedeći programski blok ovog projekta, u kojem je izvršen izračun trenutne pozicije naziva se POMAK_CLC. Prikazan je na Slika 6.2.1 kao FC koji je pozvan u Cyclic interrupt (OB30).

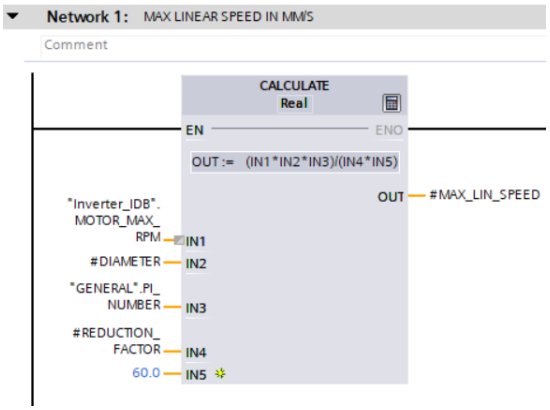


Slika 6.2.1. Funkcija izračuna trenutne pozicije iz brzine

Trenutna pozicija se dobiva na temelju trenutne brzine sa pretvarača. Međutim, kako je u poglavlju 3 opisano, potrebno je brzinu sa pretvarača, koja je u o/min, pretvoriti u linearnu brzinu, u mm/s, odnosno brzinu kojom se pozicioner giba. Ulazne varijable su time, brzina sa pretvarača, promjer bubnja te redukcijski faktor. Izlazne varijable su trenutna brzina kojom se pozicioner giba (mm/s), trenutna pozicija te vrijednost maksimalne linearne brzine koju pozicioner dostiže.

Sve navedene varijable koje se koriste za ulaz i izlaz iz bloka su zapisane u *General DB* koji je prikazan u Dodatku B.

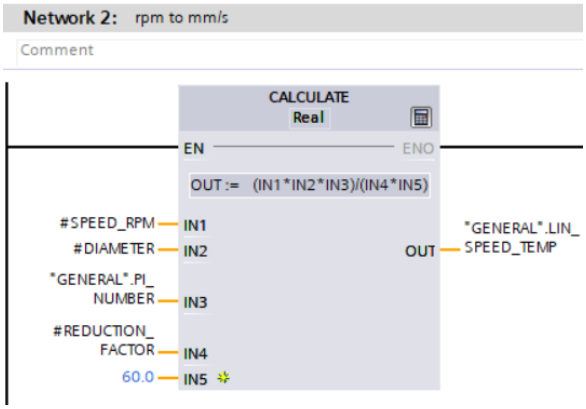
Prvi network funkcije POMAK_CLC je izračun maksimalne linearne brzine, Slika 6.2.2.



Slika 6.2.2. Izračun maksimalne postignute linearne brzine

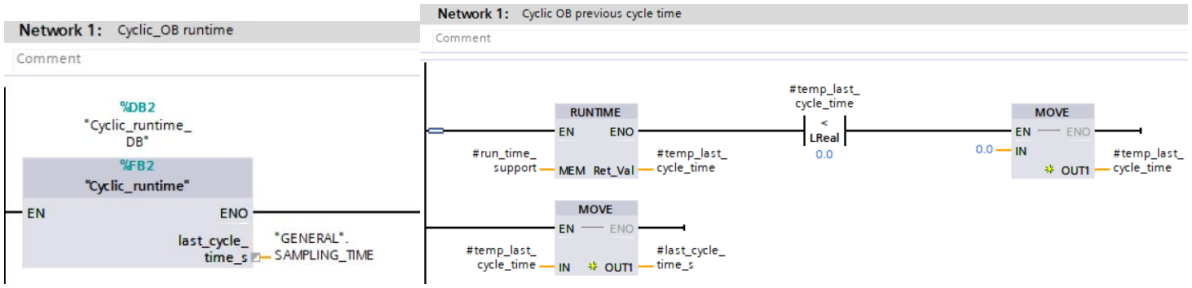
Temelji se na izračunu korištenjem instrukcije *Calculate* kojom zapisuje izraz (3.1) putem zamjene linearne i kružne trenutne vrijednosti brzine njihovim maksimalnim vrijednostima.

Istim izrazom (3.1) je izvršeno računanje trenutne linearne brzine u sljedećem networku, Slika 6.2.3.



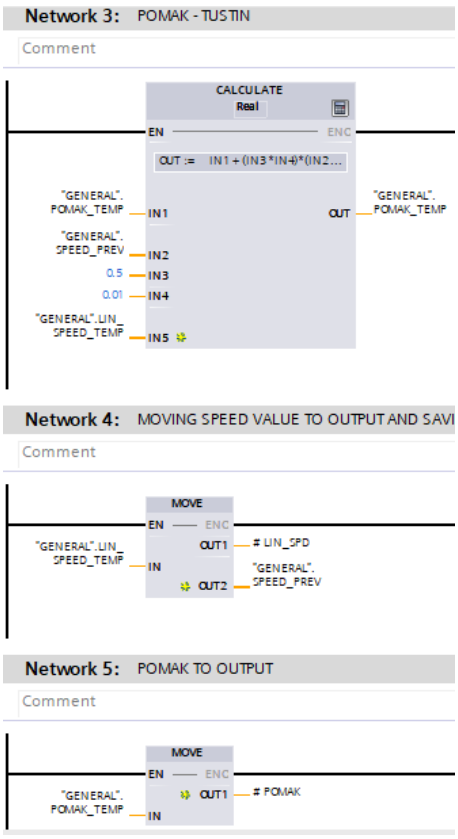
Slika 6.2.3. Izračun trenutne linearne brzine

Kako bi se izračunala pozicija, primjenom Tustinove aproksimacije, potrebno je osim trenutne brzine poznavati i vrijednost brzine iz prethodnog ciklusa. Vrijeme trajanja ciklusa određeno je korištenjem Cyclic interrupt OB-a. Ono se postavlja kod kreiranja bloka, međutim postoji i funkcija koja računa trajanje ciklusa te je prikazana na Slika 6.2.4 sa svojom unutrašnjom strukturom. *Cyclic runtime* funkcija se poziva na početku OB-a.



Slika 6.2.4. Funkcijski blok izračuna trajanja ciklusa

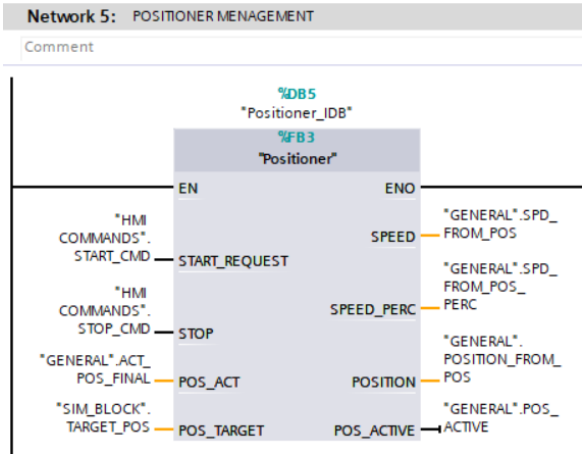
Ukratko, na temelju instrukcije RUNTIME, zapisuje se trajanje ciklusa i dovodi na izlaz funkcijr *Cyclic runtime*. Ta je vrijednost pomoću varijable SAMPLING_TIME korištena za izračun pozicije iz brzine. Instrukcijom Calculate izračunava se trenutna pozicija upotrebom formule Tustinove apoksimacije, te je potrebno nakon te linije koda, spremiti trenutnu brzinu u posebnu varijablu SPEED_PREV, kojom će se u idućem ciklusu označavati brzina trenutnog ciklusa. Također je izvršeno mapiranje trenutne brzine i pozicije na izlaz iz bloka. Navedeno je prikazano na Slika 6.2.5.



Slika 6.2.5. Izračun trenutne pozicije korištenjem Tustinove aproksimacije

6.3 Realizacija pozicionera

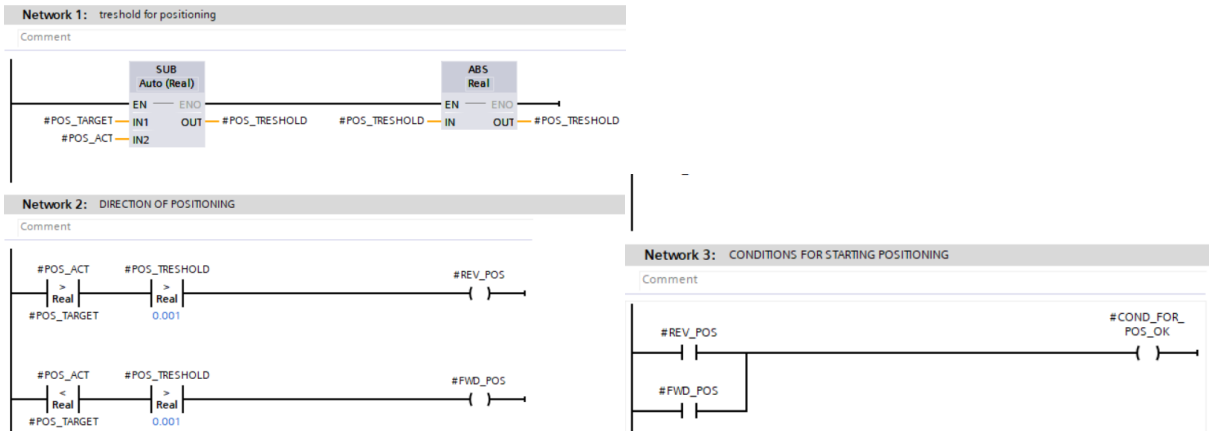
Pozicioner je FB koji na temelju trenutne i konačne pozicije te konfiguracijskih parametara izračunava brzinu potrebnu za gibanje tijekom te putanje. Na temelju te brzine, računa se i teorijska pozicija koja se postiže. Princip rada pozicionera je opisan u matematičkom modelu, ali će ovdje biti još uključeni posebni slučajevi zapisani programskim kodom. Ulazne varijable u blok su komande pokretanja i zaustavljanja pozicionera. One su spremljene u HMI COMMANDS DB-u prikazanom u Dodatku C. Aktualna pozicija, koja dolazi sa bloka POMAK_CLC, je također ulazni parametar kao i konačna pozicija. Izlaz iz bloka su brzina i teoretska pozicija te stanje POS_ACTIVE koje označuje aktivnost pozicioniranja. FB pozicionera je prikazan na Slika 6.3.1, pozvan je u Cyclic interrupt OB-u nakon bloka POMAK_CALC.



Slika 6.3.1. Funkcijski blok upravljanja pozicionerom

Iz razloga što je blok pozicionera FB, ima svoj DB, odnosno iDB koji je prikazan u Dodatku D. Vrijedi spomenuti da su konfiguracijski parametri zapisani u iDB-u kao *Static* varijable te će one biti poslane u komunikaciji sa HMI-a.

Struktura bloka *Positioner* započinje sa definiranjem smjera pozicioniranja. Ako je trenutna pozicija veća od konačne pozicije te izvan graničnih (treshold) vrijednosti tada se gibanje odvija u negativnom smjeru. Treshold vrijednost se dobiva na temelju razlike trenutne i konačne pozicije koja ne prelazi 0.001mm. Dakle, pozicioniranje će se odvijati kada je razlika u poziciji veća od treshold vrijednosti. S druge strane, pozicioniranje u pozitivnom smjeru podrazumijeva konačnu poziciju veću od trenutne. Nadalje, ako su uvjeti za smjer pozicioniranja aktivni, aktivira se pomoćna varijable koja označava da su uvjeti za pozicioniranje zadovoljeni. Prethodno opisano prikazuje Slika 6.3.2 u obliku prva tri network-a.



Slika 6.3.2. Uvjeti definiranja smjera kod pozicioniranja

Sljedeća dva network-a napisana su u SCL programskom jeziku iz razloga što je lakše i pregledniji zapisati petlje te se jednostavnije služiti matematičkim operacijama. Oba network-a

se odnose na proračun teoretskih pozicija za različite slučajeve gibanja, pri čemu prvi network služi za pozitivan smjer, dok drugi za negativan smjer pozicioniranja. Iz razloga što su identične matematičke operacije samo zapisane drugačijim predznacima, ovdje će biti opisan samo segment koji se odnosi na pozitivan smjer. Prikazan je na Slika 6.3.3.

Network 4: CALCULATING PARAMETERS FWD

Comment

```
IF #FWD_POS= TRUE THEN
    IF #PHASE.FIRST_PHASE = TRUE THEN
        #DEC_SPACE := (SQR(#SPEED) - SQR(#SLW_SPEED_CONST)) / (2 * #DEC);
        #SLW_SPD_POS := #POS_TARGET - #SLW_SPEED_DIST;
        #DEC_TO_SLW_SPD_POS := #SLW_SPD_POS - #DEC_SPACE;
        IF #SPEED < #SLW_SPEED_CONST OR #NO_SLW_SPEED THEN
            #DEC_TO_0 := #POS_TARGET - SQR(#SPEED) / (2 * #DEC);
        END_IF;
        IF #SLW_SPEED_DIST = 0 THEN
            #DEC_SPACE := SQR(#SPEED) / (2 * #DEC);
            #DEC_TO_SLW_SPD_POS := #POS_TARGET - #DEC_SPACE;
            #DEC_TO_0 := #DEC_TO_SLW_SPD_POS;
        END_IF;
    ELSE
         #DEC_SPACE := (SQR(#FST_SPEED_CONST) - SQR(#SLW_SPEED_CONST)) / (2 * #DEC);
         #SLW_SPD_POS := #POS_TARGET - #SLW_SPEED_DIST;
         #DEC_TO_SLW_SPD_POS := #SLW_SPD_POS - #DEC_SPACE;
         #DEC_TO_0 := #POS_TARGET - SQR(#SLW_SPEED_CONST) / (2 * #DEC);
         IF #SLW_SPEED_DIST = 0 THEN
             #DEC_SPACE := SQR(#FST_SPEED_CONST) / (2 * #DEC);
             #DEC_TO_SLW_SPD_POS := #POS_TARGET - #DEC_SPACE;
             #DEC_TO_0 := #DEC_TO_SLW_SPD_POS;
         END_IF;
    END_IF;
END_IF;
```

Slika 6.3.3. SCL zapis izračuna teorijskih parametara

Teorijske pozicije koje je potrebno izračunati označuju prijelomne točke u kojima se gibanje nastavlja nekom drugom fazom (brzinom). Faze gibanja su akceleracija, viša konstantna brzina, deceleracija sa više na nižu konstantnu brzinu, niža konstantna brzina te deceleracije do konačne pozicije. Pošto ih ima 5, nazvane su u programu kao prva, druga, treća, četvrta i peta faza te će se ovako i referirati u radu. Teorijske pozicije koje je potrebno odrediti zapisane su u Tablica 6.3.1 sa njihovim prikladnim varijablama korištenim u programu.

Tablica 6.3.1. Opis teorijskih pozicija na temelju varijabli

Varijabla	Teorijska pozicija
DEC_TO_SLW_DWN_POS	početak deceleracije sa više na nižu konstantnu brzinu
SLW_SPD_POS	Početak niže konstantne brzine
DEC_TO_0	Početak deceleracije do konačne pozicije
DEC_SPACE	Duljina puta deceleracije sa više na nižu konstantnu brzinu

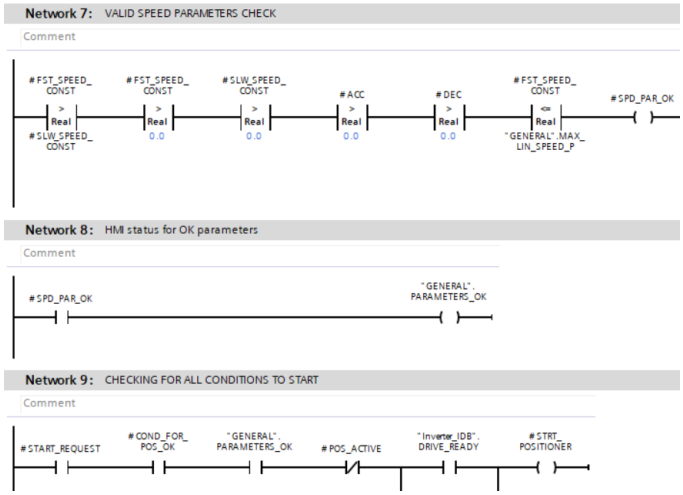
Ako se vratimo na Slika 6.3.3. SCL zapis izračuna teorijskih parametara, prvi uvjet izračuna parametara je prisutnost varijable smjera pozicioniranja. Nadalje, početak svakog gibanja započinje akceleracijom, odnosno prvom fazom. Kada se gibamo prvom fazom za izračun DEC_TO_SLW_DWN_POS i DEC_TO_0 koristimo trenutnu brzinu iz razloga što ove pozicije ne moraju biti određene višom, odnosnom nižom brzinom.

Uvjet koji se još mora naglasiti kako bi se definirala DEC_TO_0 pozicija je, da nije prisutno gibanje nižom konstantom brzinom, odnosno trenutna brzina nije postigla vrijednost niže konstantne brzine. Ovaj izračun samo vrijedi ako se gibanje izvršava u prvoj fazi. Time će se u ovom slučaju dobiti gibanje koje sadržava samo akceleraciju i deceleraciju.

Još jedan uvjet koji je prisutan uz gibanje u prvoj fazi je provjera vrijednosti varijable SLW_SPEED_DIST, koja označava je li definirano gibanje nižom konstantom brzinom. Ako nije, odnosno ako je vrijednost ove udaljenosti nula, potrebno je definirati da se usporavanje u konačnu poziciju izvršava sa najviše postignute brzine. Formule koje se koriste za izračun ovih parametara su opisane u poglavlju 3.

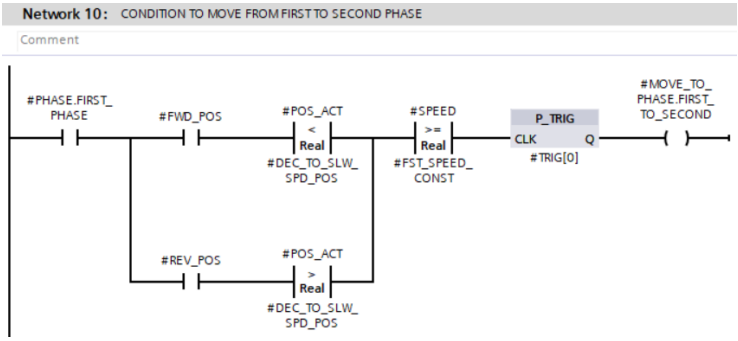
Sljedeće što je u kodu definirano je izračun parametara dok se gibanje ne odvija u prvoj fazi. To znači da će se teorijske pozicije izračunati na temelju viših i nižih konstantnih brzina iz razloga što upravo one definiraju preostale faze. Isto je tako i u ovom slučaju potrebno staviti uvjet kada nema gibanja nižom konstantom brzinom da se usporavanje do konačne pozicije odvije sa više konstantne brzine.

Preostali program se nastavlja u LD-u. Slika 6.3.4 prikazuje sljedećih tri network-a. Kreće se sa provjerom valjanosti definiranih konfiguracijskih parametara. Ako su oni zadovoljeni preko varijable PARAMETERS_OK se šalju na HMI, kao statusna potvrda. Zahtjev za pokretanjem pozicionera se odvija putem HMI komande, koja se dovodi na ulaz bloka. Međutim mora se zadovoljiti nekoliko uvjeta prije nego se on stvarno pokrene, odnosno izvrši pozicioniranje. Uvjet za smjer pozicioniranja mora biti aktivan, aktivan uvjet za valjanost parametara, potvrda sa pretvarača da je spreman za pokretanje motora te ne smije biti već aktivno pozicioniranje u toku.



Slika 6.3.4. Provjera ispravnosti parametara te pokretanje pozicioniranja

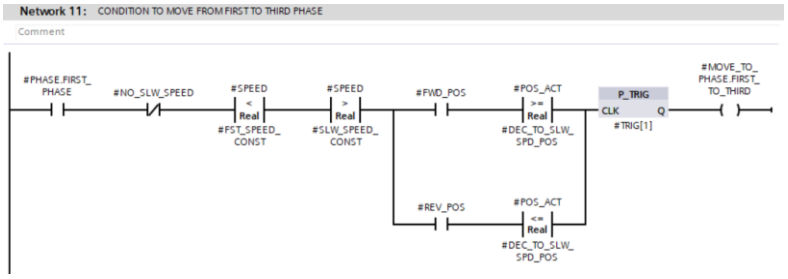
Sljedeći dio programa pozicionera se odnosi na definiranje faza i prijelaza između faza na osnovu trenutne pozicije u kojoj se nalazimo te teorijske pozicije koje označavaju upravo taj prelazak iz faze u fazu. Isto tako nekad će uvjet za prelazak biti i trenutna brzina u odnosu na konstante višu i nižu brzinu. Slika 6.3.5 prikazuje uvjete za prijelaz iz prve u drugu fazu.



Slika 6.3.5. Uvjet za prelazak iz prve u drugu fazu

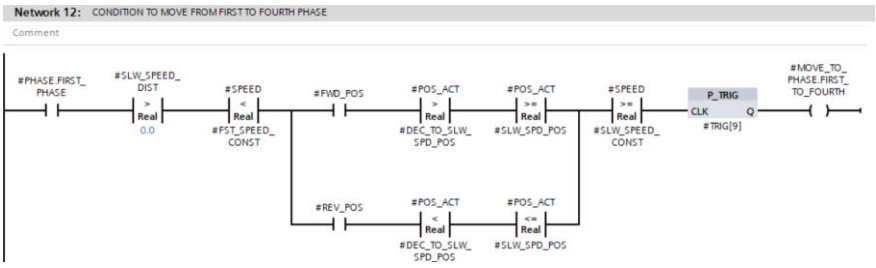
Kako bi se kreirali uvjeti, koriste se instrukcija P_TRIG, odnosno okidač pozitivnog brida.
P_TRIG aktivira izlaz samo za jedan ciklus kada se na ulaz dovede pozitivan brid. Pozitivan brid označava promjenu stanja sa 0 u 1, gdje 0 predstavlja neaktivnost, a 1 aktivnost neke varijable ili linije koda.
Jedan od uvjeta, koji mora postojati kako bi se dogodio prijelaz iz prve u drugu fazu je već gibanje u prvoj fazi. Ako je to slučaj, uspoređuje se trenutna pozicija sa onom teorijskom koja označava prijelaz u treću fazu. Dakle, ako je trenutna pozicija manja od pozicije usporavanja sa više u nižu konstantnu brzinu, može se prijeći u drugu fazu kada trenutna brzina dosegne vrijednost više konstantne brzine. U negativnom smjeru situacija je obratna, prelazak iz prve u drugu fazu događa se ako je trenutna pozicija veća od pozicije usporavanja sa više u nižu

konstantnu brzinu kada trenutna brzina dostigne vrijednost više konstantne brzine. Trenutna brzina se računa na kraju programskog bloka te će njen izračun biti kasnije prikazan. Kada ovaj uvjet za prelazak postane aktivan, on će služiti kao uvjet za definiranje gibanja u drugoj fazi.
Sljedeći network, na Slika 6.3.6, prikazuje uvjet koji će se ostvariti ako se prethodni uvjet nije ostvario, a to je prelazak iz prve u treću fazu.



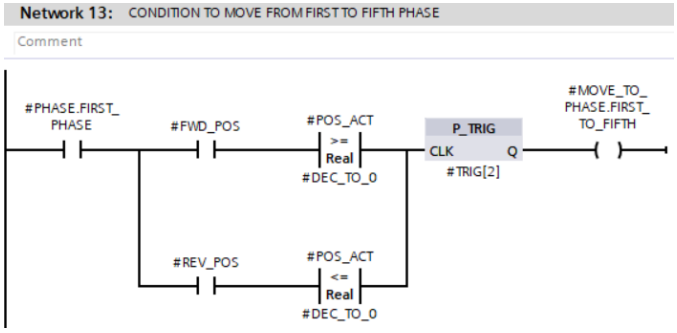
Slika 6.3.6. Uvjet za prelazak iz prve u treću fazu

Prijelaz sa prve u treću fazu događa se kada je trenutna pozicija prešla teorijsku poziciju usporavanja sa više konstantne brzine u nižu, ali trenutna brzina nije uspjela postići vrijednost više konstantne brzine te je naše gibanje primorano „zaobići“ drugu fazu.
Prijelaz iz prve u četvrtu fazu, prikazan je na Slika 6.3.7. Dogodit će se kada je trenutna pozicija veća i od pozicije usporavanja sa više konstantne brzine na nižu i od pozicije početka gibanja nižom konstantnom brzinom.Također, dodatni uvjet je još da brzina tada poprimi vrijednost niže konstantne brzine.



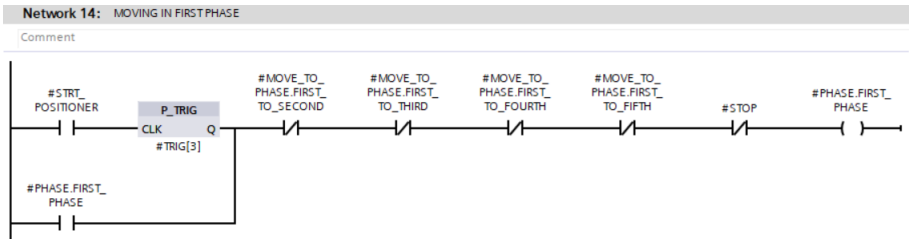
Slika 6.3.7. Uvjet za prelazak iz prve u četvrtu fazu

Posljednji prijelaz iz prve faze koji je još moguć je direktan prijelaz iz prve u petu fazu, odnosno u usporavanje do konačne pozicije. Prikazan je na Slika 6.3.8. Događa se kada je trenutna pozicija tijekom gibanja u prvoj fazi veća od pozicije usporavanja do konačne pozicije u pozitivnom smjeru gibanja.



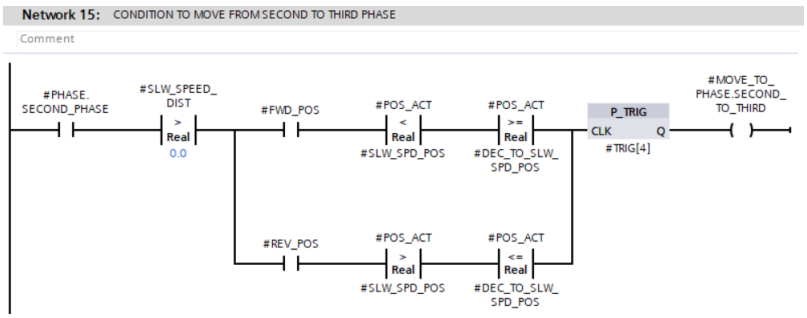
Slika 6.3.8. Uvjet za prelazak iz prve u petu fazu

Nakon uvjeta koji se dogode pri prelasku iz prve faze u neku sljedeću, definira se gibanje prve faze na Slika 6.3.9. Ono je aktivno kada se pokrene pozicioner i samoodržavanjem je izvedeno da ostaje aktivno sve dok se ne aktivira neki od uvjeta koji prekida gibanje u prvoj fazi. Samoodržavanje je postupak koji se primjenjuje kada je potrebno izlaznu varijablu držati aktivnom nakon odsustva ulazne varijable koja je aktivira. Drži se aktivnom upravo sama sobom te se zato naziva samoodržavanje. Ulazna varijabla je u ovom slučaju STRT_POSITIONER te on samo za jedan ciklus aktivira izlaznu varijablu FIRST_PHASE koja već u istom ciklusu drži sama sebe aktivnom jer u idućem ciklusu ulazna varijabla više neće biti aktivna. Tako će uvjeti koji jedino mogu prekinuti gibanje u prvoj fazi biti prelasci u neku od preostalih faza ili komanda za zaustavljanje pozicioniranja.



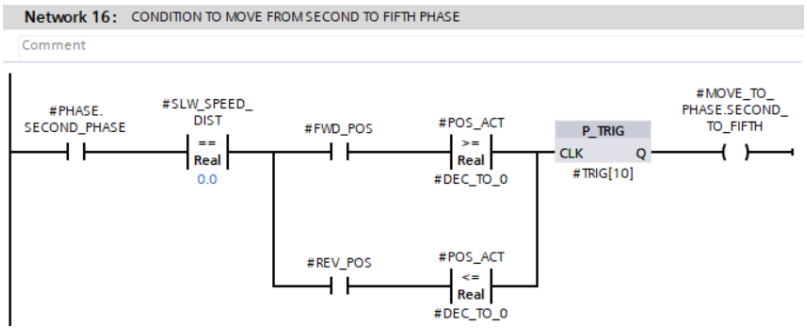
Slika 6.3.9. Gibanje u prvoj fazi

Nadalje se opisuju uvjeti prelaska koji prethode gibanju u drugoj fazi. Kada je gibanje u drugoj fazi aktivno, u treću se fazi prelazi ako je trenutna pozicija veća od pozicije usporavanja sa više konstantne brzine u nižu, ali mora biti manja od pozicije početka gibanja nižom konstantnom brzinom. Ovo je prikazano na Slika 6.3.10.



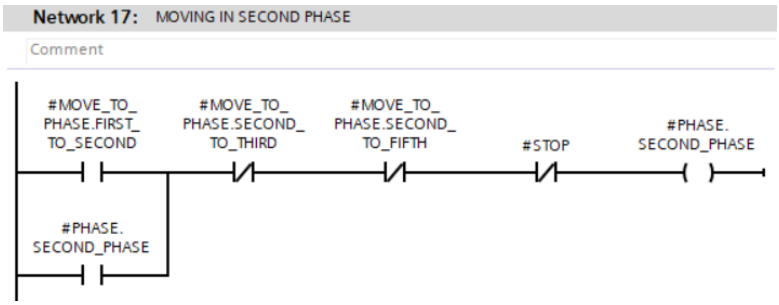
Slika 6.3.10. Uvjet za prelazak iz druge u treću fazu

Sa druge faze moguće je također prelazak u petu fazu. Ovakav se prelazak odnosi kada nema gibanja nižom konstantnom brzinom, odnosno SLW_SPEED_DIST je jednak nula. Time se dobiva usporavanje do konačne pozicije sa više konstantne brzine direktno. Slika 6.3.11 prikazuje uvjet prelaska sa druge u petu fazu.



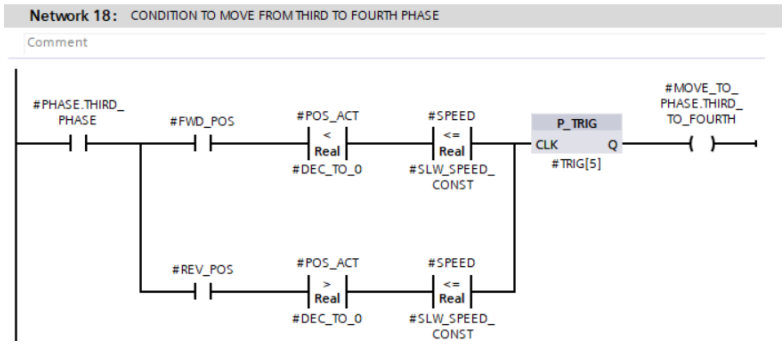
Slika 6.3.11. Uvjet za prelazak iz druge u petu fazu

Sljedeća linija koda odnosi se na definiranje gibanja drugom fazom, prikazana je na Slika 6.3.12. Gibanje drugom fazom moguće je aktivirati uvjetom koji označuje prelazak sa prve u drugu fazu. Tada se pomoću samoodržavanja gibanje nastavlja odvijati u drugoj fazi sve dok ga jedan od uvjeta ne prekine. To su uvjeti prelaska s druge u treću fazu ili sa druge u petu fazu ili komanda za zaustavljanje pozicioniranja.



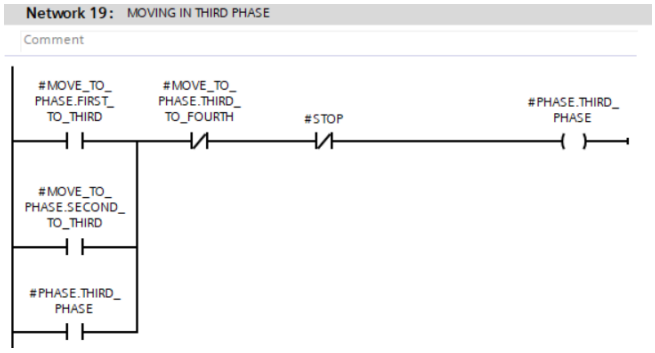
Slika 6.3.12. Gibanje u drugoj fazi

Nadalje, kada se gibanje odvija u fazi deceleracije sa više na nižu konstantnu brzinu te se brzina spusti do niže konstantne brzine, aktivira se prijelaz u četvrtu fazu. Uvjet je samo da trenutna pozicija nije veća od pozicije usporavanja do konačne pozicije. Opisano je prikazano na Slika 6.3.13 pomoću koda.



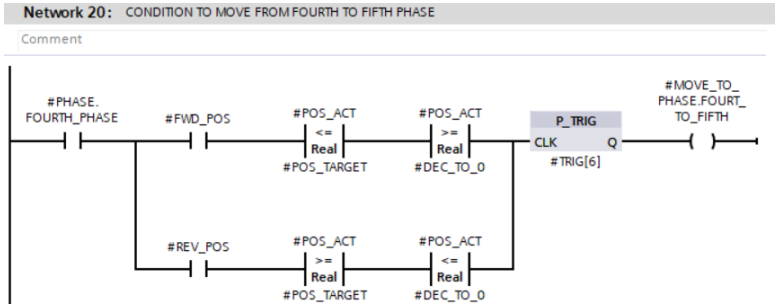
Slika 6.3.13. Uvjet za prelazak iz treće u četvrtu fazu

Nakon ovog uvjeta može se definirati gibanje u trećoj fazi, prikazano na Slika 6.3.14. Gibanje u trećoj fazi moguće je aktiviranjem uvjeta prelaska s prve ili druge faze u treću fazu. Kada je gibanje u trećoj fazi aktivno, uvjet prelaska s treće u četvrtu ili stop komanda ga prekida.



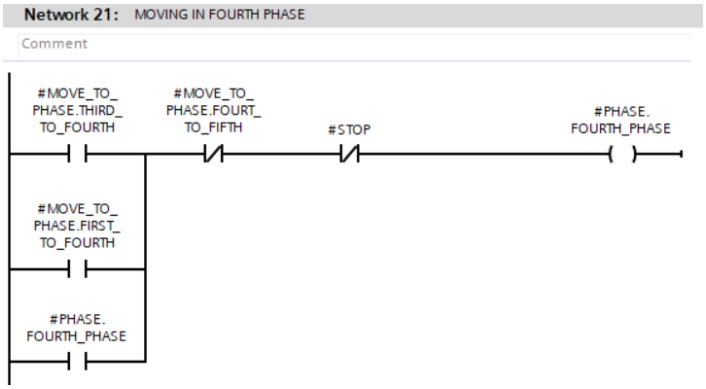
Slika 6.3.14. Gibanje u trećoj fazi

Sljedeći uvjet prelaska koji je moguće izvršiti je sa četvrte u petu fazu kojeg prikazuje Slika 6.3.15. Događa se kada trenutna pozicija prelazi teorijsku točku usporavanja do konačne pozicije.



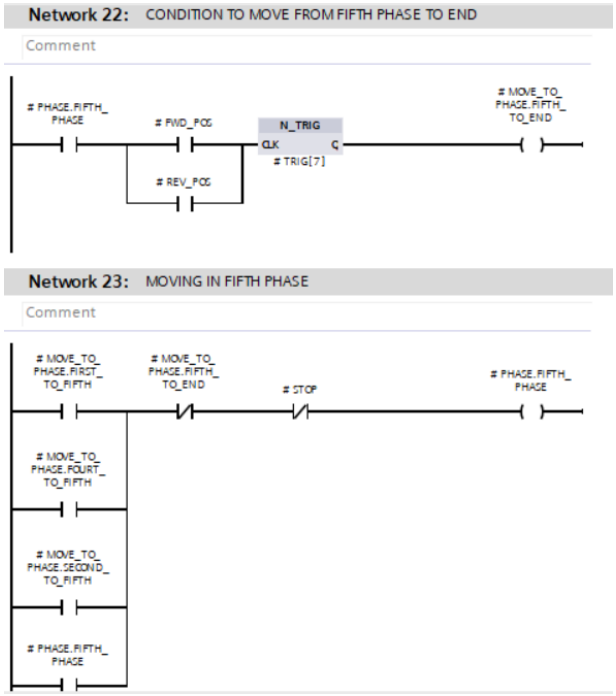
Slika 6.3.15. Uvjet za prelazak iz četvrte u petu fazu

Nakon toga se može definirati gibanje četvrte faze, prikazano na Slika 6.3.16. Po istom principu kao i za prošle faze, gibanje četvrtom fazom aktivira se prilikom prelaska ili s treće ili s prve faze u četvrtu, a prekida prelaskom s četvrte u petu fazu ili komandom za zaustavljanje.



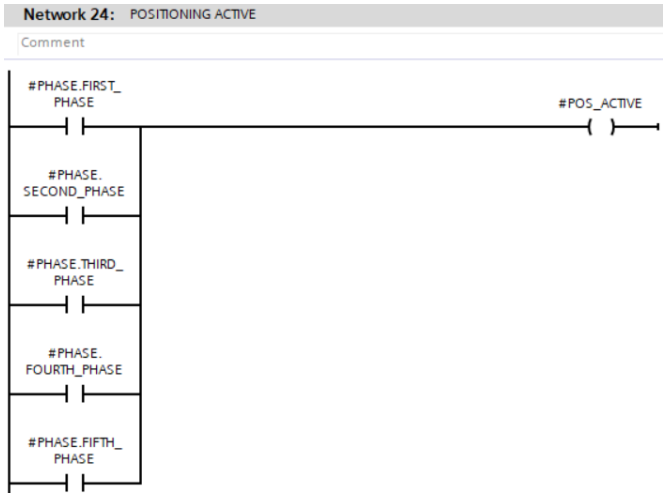
Slika 6.3.16. Gibanje u četvrtoj fazi

Preostao je još uvjet koji će označit završetak gibanja, prikazan na Slika 6.3.17. Za to će se iskoristiti N_TRIG instrukcija koja aktivira izlaz dolaskom negativnog brida, odnosno promjenom stanja sa 1 na 0. Time varijable koje označuju smjer pozicioniranja prestanu biti aktivne kada su unutar zadanih graničnih vrijednosti, tada se aktivira i varijabla koja označava završetak pozicioniranja. Na Slika 6.3.17 je isto tako prikazano definiranje gibanja petom fazom. Aktivira se uvjetom prelaska sa prve, druge ili četvrte faze u petu, a prekida se uvjetom završetka pozicioniranja ili komandom stop.



Slika 6.3.17. Uvjet za završetak gibanja te gibanje u petoj fazi

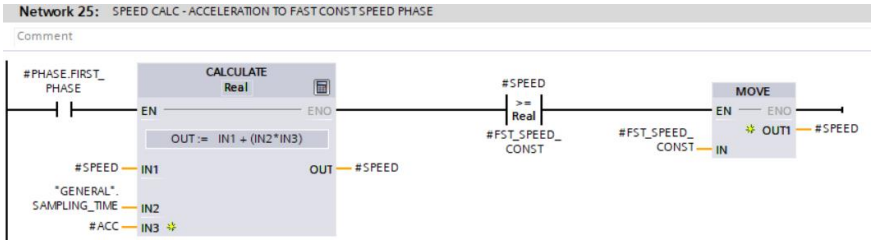
Ovime su definirani svi prijelazi između faza i opisana gibanja svake od faza. Sljedeći network prikazuje status aktivnog pozicioniranja, a aktivan je ako je gibanje u bilo kojoj od faza aktivno. Slika 6.3.18 prikazuje status POS_ACTIVE koji se izvodi iz FB-a te predstavlja aktivnost pozicioniranja.



Slika 6.3.18. Aktivno pozicioniranje

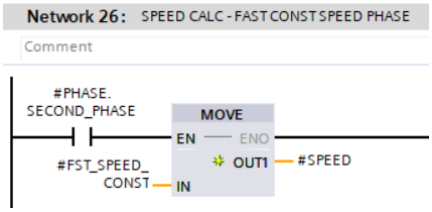
Sljedeći korak koji se treba izvršiti je izračun trenutne brzine u ovisnosti o fazi u kojoj se gibanje odvija. Slika 6.3.19 prikazuje izračun brzine prve faze korištenjem instrukcije *Calculate*. Već je poznato kako je prva faza faza akceleracije. Proračun brzine se temelji na umnošku akceleracije

(konfiguracijski parametar) i vremena uzorkovanja. Vrijeme uzorkovanja je varijabla koja označuju vrijeme trajanja ciklusa te se time svakog ciklusa dobivena brzina zbraja sa brzinom prethodnog ciklusa. U ovoj fazi brzina linearno raste.



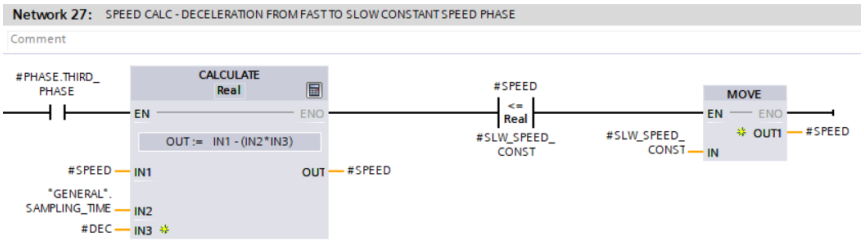
Slika 6.3.19. Izračun brzine prve faze

Kada se gibanje odvija u drugoj fazi, fazi konstantne više brzine, brzina se svaki ciklus samo zapisuje kao brzina više konstantne brzine. Slika 6.3.20 prikazuje izračun brzine u drugoj fazi.



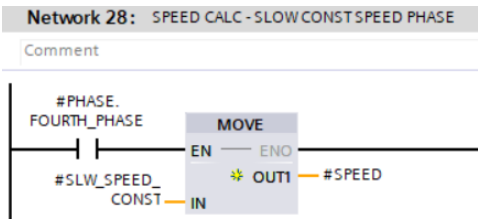
Slika 6.3.20. Izračun brzine druge faze

Izračun brzine u trećoj fazi temelji se na deceleraciji. Brzina se na temelju prethodne brzine oduzima za umnožak deceleracije i vremena uzorkovanja, prikazano na Slika 6.3.21.



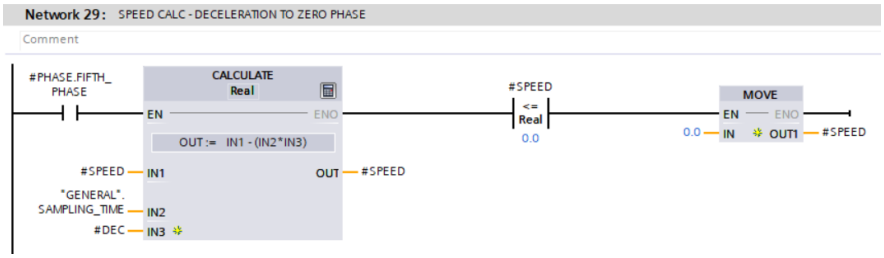
Slika 6.3.21. Izračun brzine treće faze

Četvrta faza je faza niže konstantne brzine. Slično kao i u drugoj fazi, brzina se samo zapisuje kao vrijednost niže konstantne brzine što je prikazano na Slika 6.3.22.



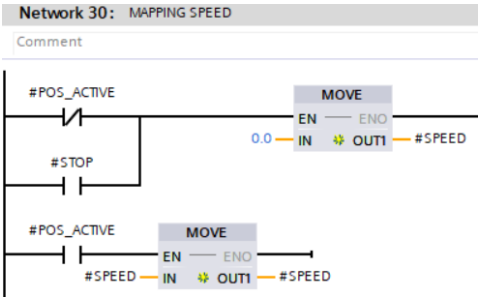
Slika 6.3.22. Izračun brzine četvrte faze

U posljednjoj fazi, fazi deceleracije do konačne pozicije, brzina se računa kao i u trećoj fazi, što je prikazano na Slika 6.3.23.



Slika 6.3.23. Izračun brzine pete faze

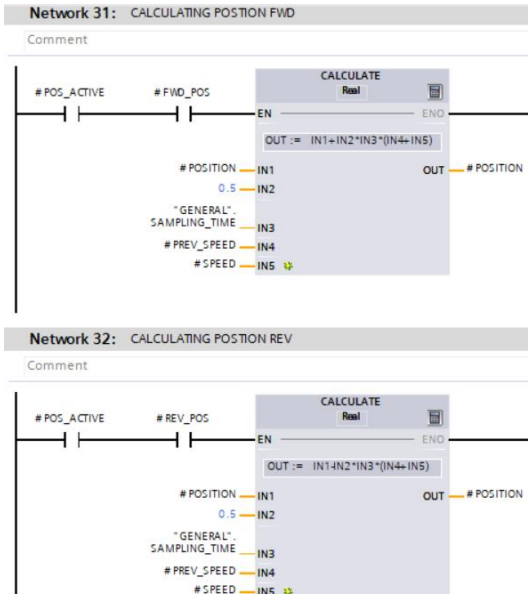
Kada nije aktivna niti jedna od faza, odnosno pozicioniranje se ne izvršava, trenutna brzina kojom se potrebno gibati je nula. Isto tako, ako se uključi stop komanda, potrebno je zaustaviti pozicioner, odnosno spustiti brzinu na 0. Ovo je zapisano u network-u sa Slika 6.3.24.



Slika 6.3.24. Mapiranje brzine

Može se primijetiti kako je brzina uvijek pozitivna, kao što je već bilo spomenuto u ovom radu. Trenutna brzina koja je prethodno bila izračunata se šalje prema pretvaraču u pozitivnim vrijednosti, dok se negativne vrijednosti (suprotan smjer vrtnje) dobiva aktiviranjem varijable za promjenu smjera. Nakon što je izračunata brzina, iz nje se može dobiti pozicija integriranjem. Ta pozicija je teorijska, odnosno ona pozicija u kojom bi se trebali nalaziti ako se pozicioner giba zadanom brzinom te nisu prisutne poremećajne veličine u sustavu. Izračun pozicije za pozitivan i

negativan smjer gibanja prikazan je na Slika 6.3.25 te se temelji na zapisu formule Tustinove aproksimacije, izraz (3.3) putem instrukcije *Calculate*.

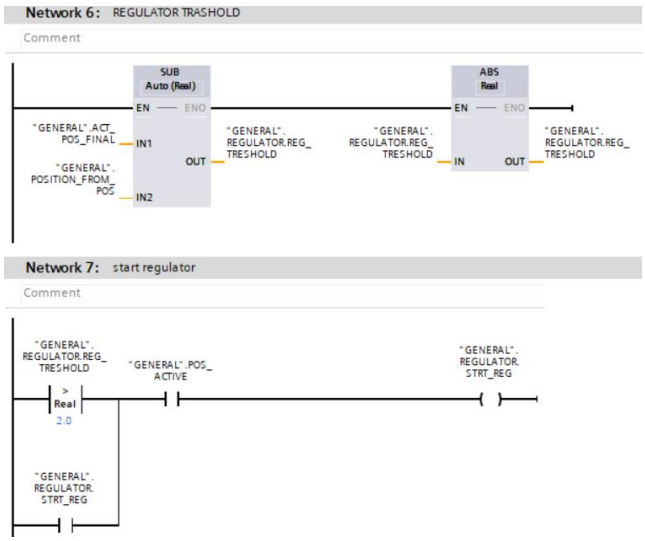


Slika 6.3.25. Izračun teorijske pozicije

6.4 Projektiranje regulatora

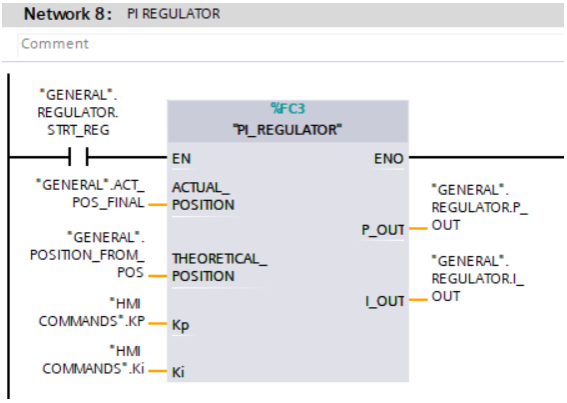
Nakon što se izradio blok upravljanja pretvarača te je izrađen algoritam upravljanja pozicionerom kreće se na projektiranje regulatora. Svrha regulatora je smanjenje greške koje nastaje između trenutne pozicije sa pretvarača i teorijske pozicije iz pozicionera. Greška u idealnom slučaju ne nastaje, međutim u realnim slučajevima se može pojaviti zbog inercije koja sprečava teorijsko gibanje, trenja i slično. Međutim, u ovom radu će se ovakve situacije zamijeniti generiranjem poremećajnih veličina kako bi se ispitao rad regulatora.

Projektiranje regulatora se nastavlja u Cyclic interrupt OB-u odmah nakon network-a gdje se pozvao blok pozicionera. Započinje se sa postavljanjem treshold vrijednosti regulatora što je prikazano na Slika 6.4.1.



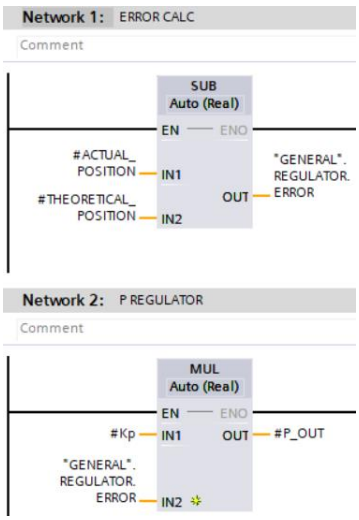
Slika 6.4.1. Izračun treshold vrijednosti za pokretanje regulatora

Granična vrijednost se odnosi na razliku trenutne i teorijske pozicije. Ako je ta razlika veća od 2mm, pokreće se regulator te je aktivan dokle je god aktivno pozicioniranje.
Kada je pokretanje regulatora zadovoljeno, poziva se funkcija PI_REGULATOR, prikazana na Slika 6.4.2. Ovom je funkcijom opisan rad regulator na principu PI djelovanja.



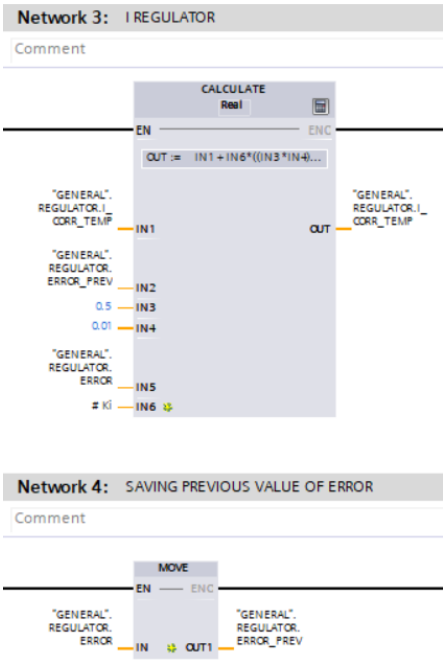
Slika 6.4.2. Funkcijski blok PI regulatora

Funkcija regulatora kao ulazne veličine ima teorijsku i trenutnu poziciju te pojačanja regulacijskih djelovanja Kp i Ki. Pojačanja su varijable koja se postavljaju putem HMI-a. Izlazi iz regulatora su P i I korekcijsko djelovanje.
Sadržaj strukture bloka temelji se na izračunu varijable greške preko koje se definira P korekcijsko djelovanje prikazano na Slika 6.4.3. Greška je razlika trenutne i teorijske pozicije. P djelovanje predstavlja umnožak te greške sa pojačanjem Kp.



Slika 6.4.3. Izračun pogreške pozicije te P regulacijskog djelovanja

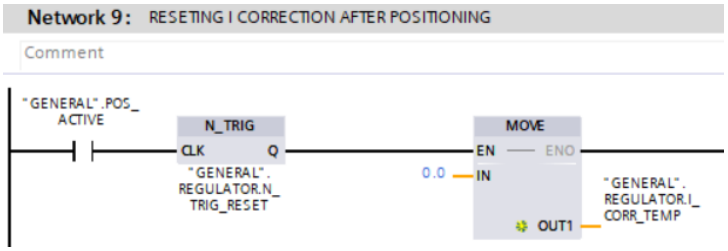
Na Slika 6.4.4 je prikazan izračun I korekcijskog djelovanja. Poznato je kako izlaz iz I regulatora predstavlja integral greške. Tako je moguće pomoću Tustinove aproksimacije izračunati površinu ispod greške na temelju vrijednosti greške trenutnog i prošlog ciklusa, vremena uzorkovanja te pojačanja, Ki.



Slika 6.4.4. Izračun I regulacijskog djelovanja

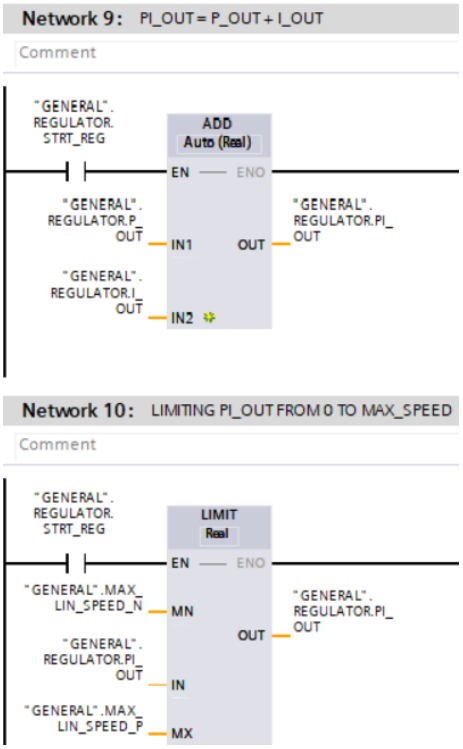
Iz razloga što nakon pozicioniranja uvijek ostane neka mala vrijednost greške, vrijednost I korekcijskog djelovanja će se povećavati izračunom površine ispod greške (integrala) iako se pozicioniranje ne izvršava. Time se u network-u ispod pozivanja regulatora u Cyclic interrupt-u I djelovanje mora resetirati na nulu, što prikazuje Slika 6.4.5. Kada pozicioniranje postane

neaktivno, zapisano pomoću N_TRIG instrukcije, vrijednost nula se zapisuje u I korekcijsko djelovanje.



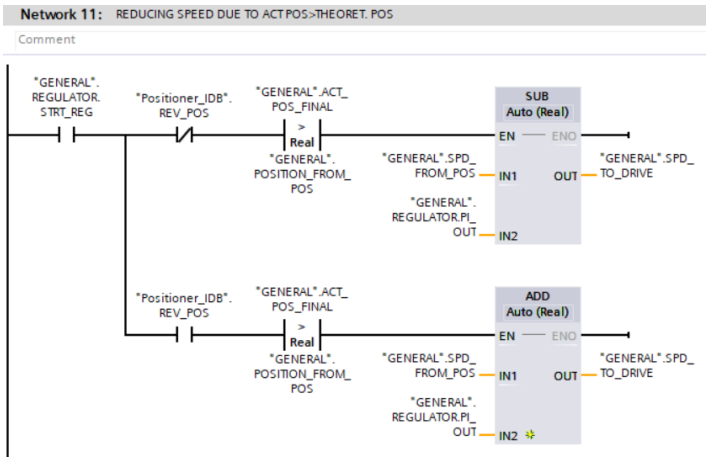
Slika 6.4.5. Resetiranje I regulacijskog djelovanja

Nakon dobivanja pojedinačnih korekcijskih djelovanja regulatora, potrebno ih je zbrojiti kako bi se dobilo PI djelovanje. Na Slika 6.4.6 je prikazano dobivanje PI djelovanja te isto tako ograničavanje njegove vrijednosti. PI korekcijsko djelovanje predstavlja vrijednost brzine koju je potrebno dodati ili oduzeti od brzine koja se šalje pretvaraču. Iz tog se ta vrijednost mora ograničiti u granicama u kojim se brzina šalje prema pretvaraču, a to je od nula do maksimalne moguće.



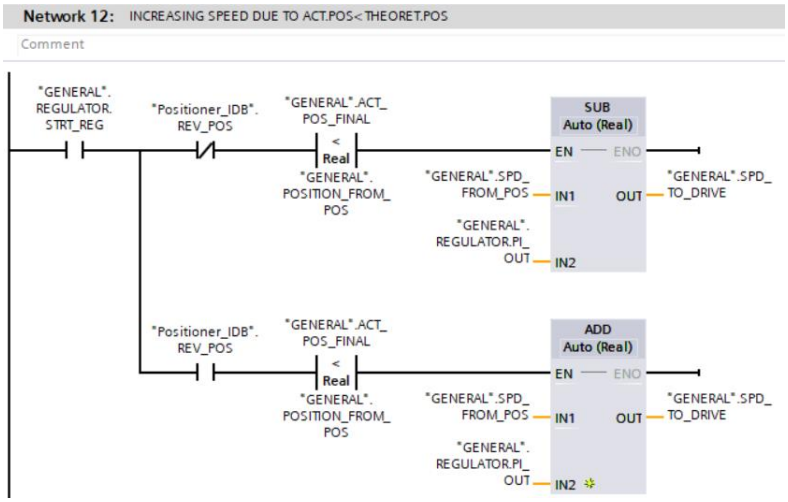
Slika 6.4.6. Dobivanje PI regulacijskog djelovanja te ograničavanje s obzirom na brzinu

Sljedeće je potrebno odrediti kako će se regulacijsko djelovanje iskoristiti na način da ispravlja prisustvo poremećajnih veličina. Poremećajne veličine mogu djelovati na dva načina na trenutnu poziciju. Prvi način je da povećaju brzinu te se time nalazimo u većoj trenutnoj poziciji u odnosu na teorijsku. Kako bi se u tom slučaju putem regulacijskog djelovanja ispravila pogreška potrebno je smanjiti trenutnu brzinu jer se pozicioner nalazi „ispred“ teorijske putanje. Brzina koja se šalje pretvaraču smanjuje se na način da se brzina iz pozicionera (koja se prije direktno slala pretvaraču) umanji za vrijednost korekcijskog djelovanja, što je opisano na Slika 6.4.7.



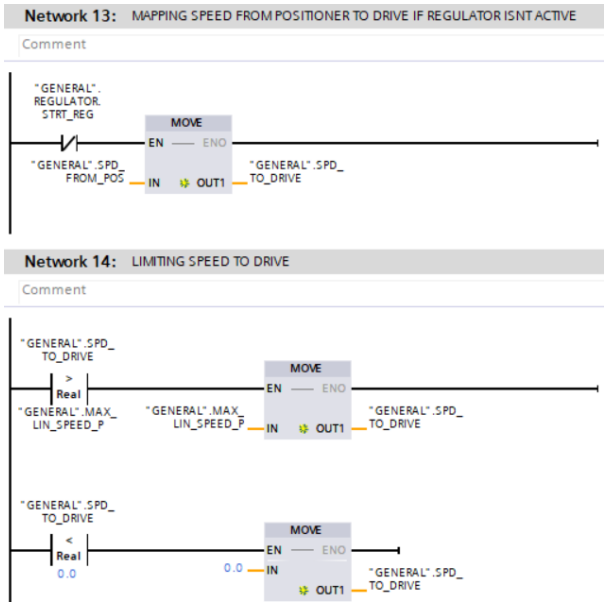
Slika 6.4.7. Smanjivanje konačne brzine prema pretvaraču

Drugi slučaj, kako poremećajne veličine mogu djelovati na trenutnu poziciji, je umanjiti brzinu te se tako trenutna pozicija očitava kao manja od teorijske. Tada je potrebno korekcijskim djelovanjem regulatora povećati brzinu koja se šalje prema pretvaraču. Povećanje brzine prema pretvaraču dobiva se u slučaju kada se od brzina iz pozicionera oduzme korekcijsko djelovanje PI regulatora iz razloga što je ono negativno u ovom slučaju, što je prikazano naSlika 6.4.8.



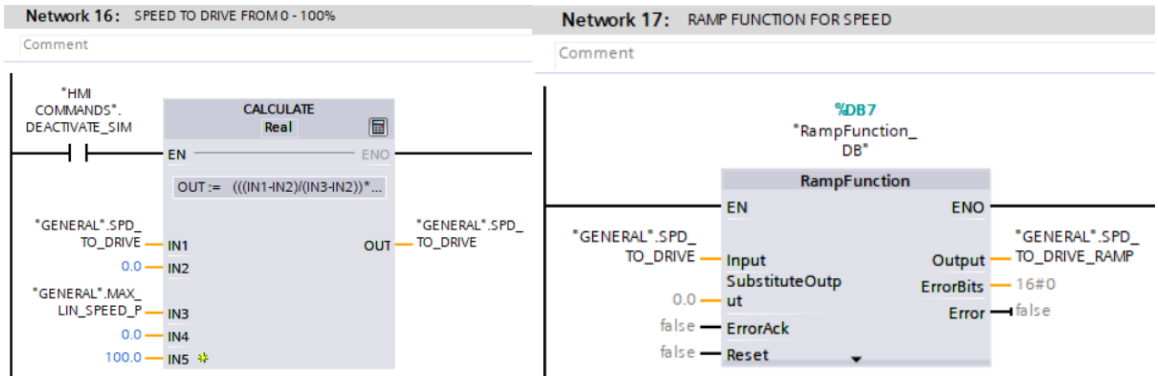
Slika 6.4.8. Povećanje konačne brzine prema pretvaraču

U sljedećem se network-u zapisuje brzina koja se šalje pretvaraču u slučaju kada regulator nije uključen. To je brzina iz pozicionera. Nadalje se limitira brzina koja se šalje prema pretvaraču, ako je ona veća od maksimalne, šalje se maksimalna vrijednost, ako je manja od nule, šalje se nula. Sljedeće je prikazano u dva network-a na Slika 6.4.9.



Slika 6.4.9. Mapiranje i limitiranje brzine prema pretvaraču

Posljednja što još treba napraviti kako bi se brzina poslala u ispravnim jedinicama prema pretvaraču je skalirati. Brzina sa kojom se do sada radilo je linearna brzina u mm/s, pošto je pretvarač izveden tako da prihvaća brzinu u postotcima, linearna brzina (0- maksimalna linearna) je skalirana po brzini u postotcima (0-100%). Isto tako je na Slika 6.4.10 prikazano korištenje instrukcije *RampFunction*. Ona se koristi kada su prisutni skokovi koje je potrebno rampirati. Skokovi u ovom slučaju mogu nastati korištenjem regulatora kada korekcijsko djelovanje momentalno pojavom greške poprima neku vrijednost ispravljanja.



Slika 6.4.10. Skaliranje brzine prema pretvaraču te korištenje Ramp funkcije

7 IZRADA HMI VIZUALIZACIJE

Kako bi se izvedeno programsko rješenje moglo iskoristiti u praktične svrhe u realnom svijetu potrebno je osigurati mogućnost kontrole i nadzora nad sustavom. Više puta je spomenuto kako se ulazni parametri definiraju u sustavu od strane operatera na osnovu primjene pozicionera te karakteristika procesa koji se automatizira. Time će se primjerice, u jednom procesu koristiti visoke vrijednosti akceleracije zbog potrebe brze reakcije sustava, dok će u nekom drugom odgovarati i postepeno ubrzavanje. Slično je i s pojačanjima regulatora koja su različita za svaki proces u ovisnosti o tipu poremećajne veličine. Time mora postojati mogućnost promjene ulaznih parametara, uz praćenje ponašanja sustava, a to je omogućeno HMI uređajima. Izrada vizualizacije HMI zaslona odnosi se upravo na postavljanje različitih elemenata kojima je moguće izvršiti naredbe, promjenu parametara, te nadzor nad procesnim i statusnim veličinama u grafičkom ili brojčanom obliku. Prije nego će se opisivati zasloni koji se koriste u ovakvom sustavu, bit će predstavljen dio komunikacije varijabli koju je potrebno izvršiti između PLC i HMI dijela.

7.1 Razmjena varijabli između PLC-a i HMI-a

Tablica *HMI tags* u sekciji HMI uređaja navigacijskog stabla prikazana je na Slika 7.1.1. Njome su definirane sve varijable koje će sudjelovati u komunikaciji sa PLC-om.

HMI tags

Name	Tag table	Data type	Connection	PLC name	PLC tag	Address	Access mode	Acqu...	Source comment
ACC	Default tag ...	Real	HMI_Conne...	PLC_2	Positioner_IDB.ACC		<symbolic access>	1 s	acc - confg. paramter
ACT_SPEED_RPM	Default tag table	Real	HMI_Connectio...	PLC_2	GENERAL.ACT_SPEED		<symbolic access>	1 s	actual speed from drive [%]
DEACTIVATE_SIM	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".DEACTIVATE_SIM		<symbolic access>	1 s	deactivate simulation
DEC	Default tag table	Real	HMI_Connectio...	PLC_2	Positioner_IDB.DEC		<symbolic access>	1 s	dec - confg. paramter
DECREASE_RED_FAC	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".DECREASE_RED_...		<symbolic access>	1 s	deacrease reduction factor
DRIVE_FAULT	Default tag table	Bool	HMI_Connectio...	PLC_2	Inverter_IDB.DRIVE_FAULT		<symbolic access>	1 s	
DRIVE_PRESET	Default tag table	Bool	HMI_Connectio...	PLC_2	Inverter_IDB.DRIVE_PRESET_ON		<symbolic access>	1 s	
DRIVE_READY	Default tag table	Bool	HMI_Connectio...	PLC_2	Inverter_IDB.DRIVE_READY		<symbolic access>	1 s	
DRIVE_RUN	Default tag table	Bool	HMI_Connectio...	PLC_2	Inverter_IDB.DRIVE_RUN		<symbolic access>	1 s	
EMERGENCY	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".EMERGENCY		<symbolic access>	1 s	emergency cmd for inverter
FST_SPEED_CONST	Default tag table	Real	HMI_Connectio...	PLC_2	Positioner_IDB.FST_SPEED_CONST		<symbolic access>	1 s	fast const. speed - confg. paramter
INCREASE_RED_FAC	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".INCREASE_RED_F...		<symbolic access>	1 s	increase reduction factor
Ki	Default tag table	Real	HMI_Connectio...	PLC_2	"HMI COMMANDS".Ki		<symbolic access>	1 s	I gain
KP	Default tag table	Real	HMI_Connectio...	PLC_2	"HMI COMMANDS".KP		<symbolic access>	1 s	P gain
MOTOR_DIRECTION	Default tag table	Bool	HMI_Connectio...	PLC_2	Inverter_IDB.MOTOR_REVERSE_DIRE...		<symbolic access>	1 s	
PARAMETERS_OK	Default tag table	Bool	HMI_Connectio...	PLC_2	GENERAL.PARAMETERS_OK		<symbolic access>	1 s	HMI status for OK parameters
POSITION_ACTUAL	Default tag table	Real	HMI_Connectio...	PLC_2	GENERAL.ACT_POS_FINAL		<symbolic access>	1 s	actual position to HMI
POSITION_FROM_POS	Default tag table	Real	HMI_Connectio...	PLC_2	GENERAL.POSITION_FROM_POS		<symbolic access>	1 s	position from positioner [mm]
REDUCTION_FACTOR	Default tag table	Real	HMI_Connectio...	PLC_2	GENERAL.REDUCTION_FACT		<symbolic access>	1 s	reduction factor
SLW_SPEED_CONST	Default tag table	Real	HMI_Connectio...	PLC_2	Positioner_IDB.SLW_SPEED_CONST		<symbolic access>	1 s	slow const.speed - confg. paramter
SLW_SPEED_DIST	Default tag table	Real	HMI_Connectio...	PLC_2	Positioner_IDB.SLW_SPEED_DIST		<symbolic access>	1 s	slow speed distance - confg. para...
SPEED_FROM_POS	Default tag table	Real	HMI_Connectio...	PLC_2	GENERAL.SPD_FROM_POS		<symbolic access>	1 s	speed from postioner [mm/s]
START_CMD	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".START_CMD		<symbolic access>	1 s	start command
STOP_CMD	Default tag table	Bool	HMI_Connectio...	PLC_2	"HMI COMMANDS".STOP_CMD		<symbolic access>	1 s	stop command
STRT_POSITIONER	Default tag table	Bool	HMI_Connectio...	PLC_2	GENERAL.POS_ACTIVE		<symbolic access>	1 s	positioning active
Tag_ScreenNumber	Default tag table	Int	<Internal tag>		<Undefined>			1 s	
TARGET_POS	Default tag table	Real	HMI_Connectio...	PLC_2	SIM_BLOCK.TARGET_POS		<symbolic access>	1 s	target position

Slika 7.1.1. HMI Tags tablica

Komunikacija vanjskih tagova se odvija na temelju simboličkog pristupa. Temelji se na odabiru potrebne varijable sa PLC DB-a za koju se namjerava napraviti HMI tag. Postoji jedna varijabla

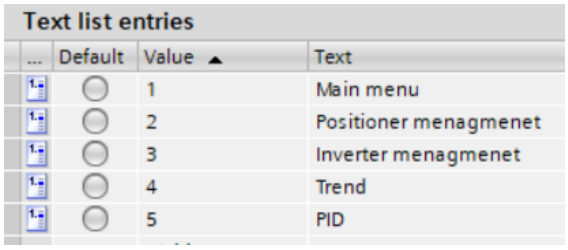
koja je unutarnji tag, Tag_ScreenNumber. Ona predstavlja broj zaslona pomoću koje će se izraditi padajući izbornik za prebacivanje između zaslona.

7.2 Izrada početnog zaslona i predloška

Predložak koji će se koristiti na svakom zaslonu prikazan je na Slika 7.2.1. Sadrži logo Siemens tvrtke, datum i vrijeme, gumb za početni zaslon, gumb za isključivanje HMI uređaja te padajući izbornik.

Slika 7.2.1. Predložak zaslona

Padajući izbornik služi kao navigacija između zaslona. Potrebno je napraviti tekstualnu listu te je učitati u dodatnim postavkama izbornika. Tekstualna lista je prikazana na Slika 7.2.2, a sastoji se od svih korištenih zaslona uz dodijeljene brojčane oznake.



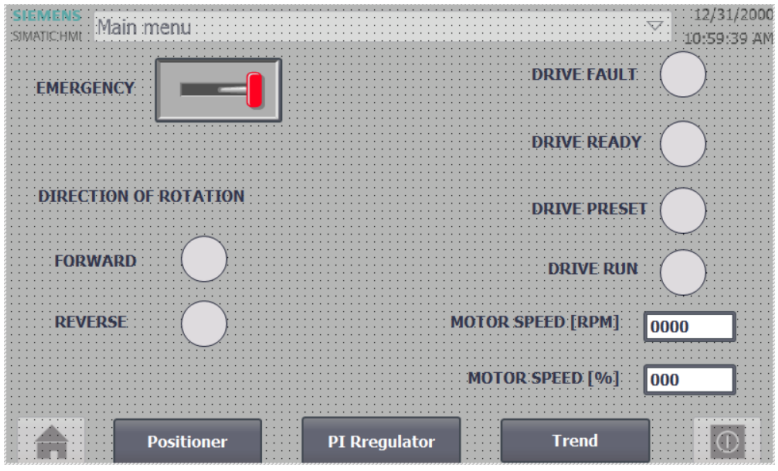
Slika 7.2.2. Tekstualna lista

Početni zaslon je prikazan na Slika 7.2.3. Prikazuje temu rada, te također gumbe kojima je moguć prelazak na druge zaslone, pored padajućeg izbornika.

Slika 7.2.3. Početni zaslon

7.3 Zaslon frekvencijskog pretvarača

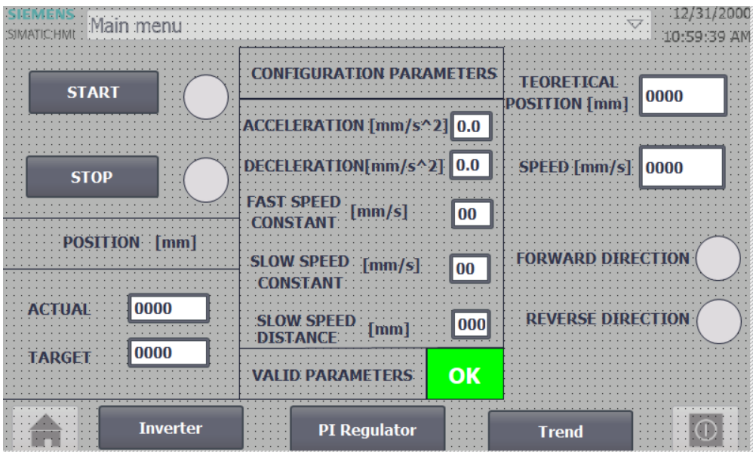
Na Slika 7.3.1 je prikazan zaslon koji omogućuje nadzor nad radom frekvencijskog pretvarača. Emergency je komanda koja treba biti aktivna tijekom ispravnog načina rada. Isto tako je prikazan trenutni smjer okretaja motora. Statusne varijable načina rada pretvarača su *run, ready, preset i fault*. Također ovdje se prati brzina vrtnje motora u okretajima u minuti te postotcima.



Slika 7.3.1. Zaslon frekvencijskog pretvarača

7.4 Zaslon pozicionera

Zaslon u kojem se unose ulazni parametri pozicionera te nadziru njegove izlazne veličine, prikazan je na Slika 7.4.1.

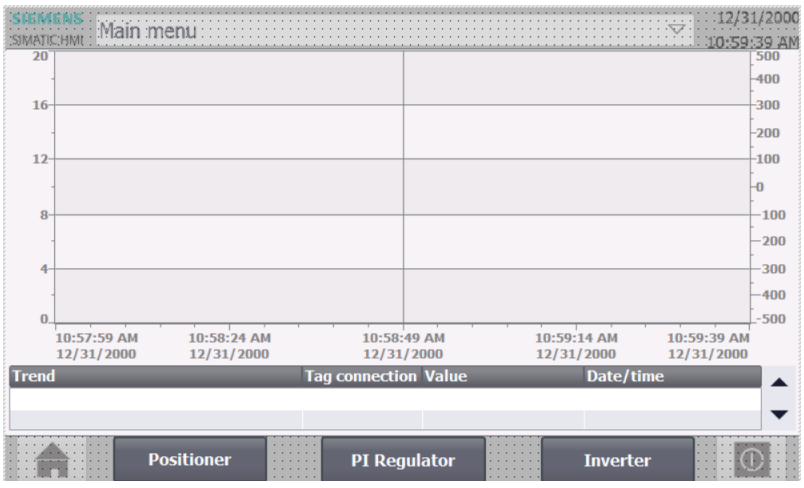


Slika 7.4.1. Zaslon pozicionera

Stoga moguće je primijetiti komande za pokretanje i zaustavljanje pozicionera, polja za unos konfiguracijskih parametara te konačne pozicije. Izlazne veličine koje se promatraju su trenutna i teorijska pozicija u milimetrima, brzina pozicionera u mm/s te smjer izvršavanja pozicioniranja. Isto tako prije nego se kreće u rad sa postavljenim parametrima, polje „Valid Parameters“ obavještava jesu li uneseni parametri ispravni.

7.5 Zaslon grafičkog prikaza procesnih vrijednosti

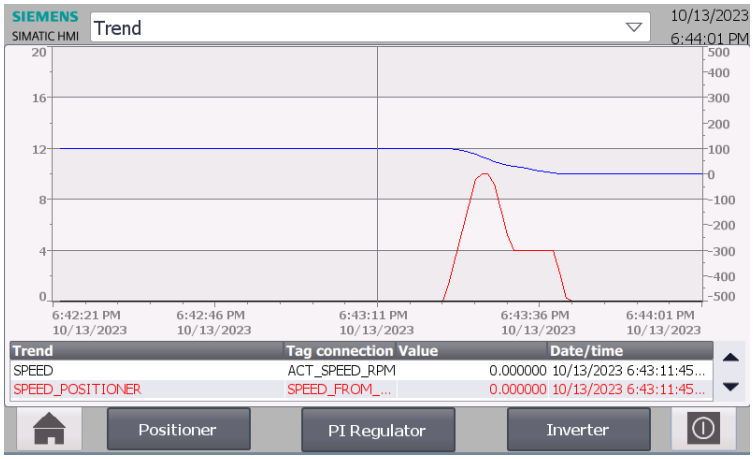
Zaslon koji služi za nadzor procesnih vrijednosti putem grafičkog prikaza naziva se Trend, a prikazan je na Slika 7.5.1. Trend označava element unutar vizualizacije kojim je moguće postići praćenje neke varijable promjenom u vremenu.



Slika 7.5.1. Zaslon grafičkog prikaza procesnih vrijednosti

U dodatnim se postavkama dodaju tagovi koji predstavljaju varijable PLC-a koje se žele prikazivati. U ovom se slučaju graf odnosi na prikaz trenutne pozicije te brzine koja se šalje pretvaraču. Ordinata funkcije brzine je smještena na lijevoj strani trenda (mm/s), dok je ordinata

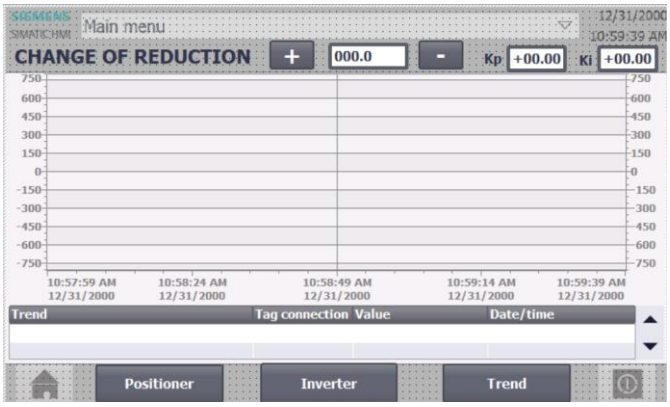
funkcije pozicije smještena na desnoj strani(mm). U ovom se radu testiranje sustava (u idućem poglavlju) neće raditi pomoću ovakvih trendova sa HMI-a, ali na Slika 7.5.2 je prikazana promjena brzine i pozicije u vremenu prilikom jednog testiranja.



Slika 7.5.2. Prikaz trenda pri testiranju programa

7.6 Zaslon PI regulatora

Zaslon koji se odnosi na upravljanje regulatorom prikazan je na Slika 7.6.1. Sastoji se od polja za unos pojačanja Kp i Ki te trenda. Trend prikazuje odnos trenutne i teorijske pozicije zato što se na temelju tog odnosa i regulator upotrebljava.

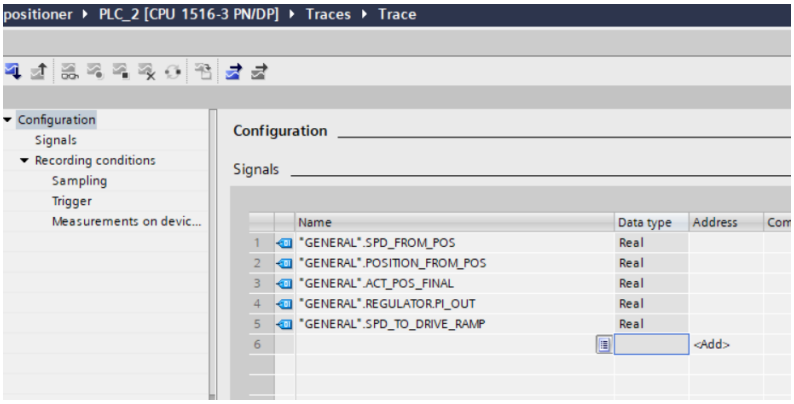


Slika 7.6.1. Zaslon PI regulatora

8 TESTIRANJE PROGRAMSKOG ALGORITMA SUSTAVA

Testiranje sustava temelji se na prikazu rezultata dobivenih izvedbom programskog rješenja u poglavlju 6 korištenjem laboratorijskog postava. Rezultati će se prikazati pomoću tzv. Trace-a, a odnosit će se na prikaz različitih karakteristika sustava koje su bile prezentirane putem programskog koda. Karakteristike sustava se odnose na različite slučajeve kojima se postiže gibanje od trenutne do konačne pozicije, kako promjena konfiguracijskih parametara utječe na gibanje, utjecaj poremećajnih veličina, odnosno ispravljanje regulatora i slično. Trace je funkcija unutar PLC sekcije navigacijskog stabla koja omogućuje snimanje varijabli u vremenu, slično kao element Trend u HMI vizualizaciji, međutim prikladnija je za prikaz rezultata u smislu opcija koje pruža za očitavanja vrijednosti.

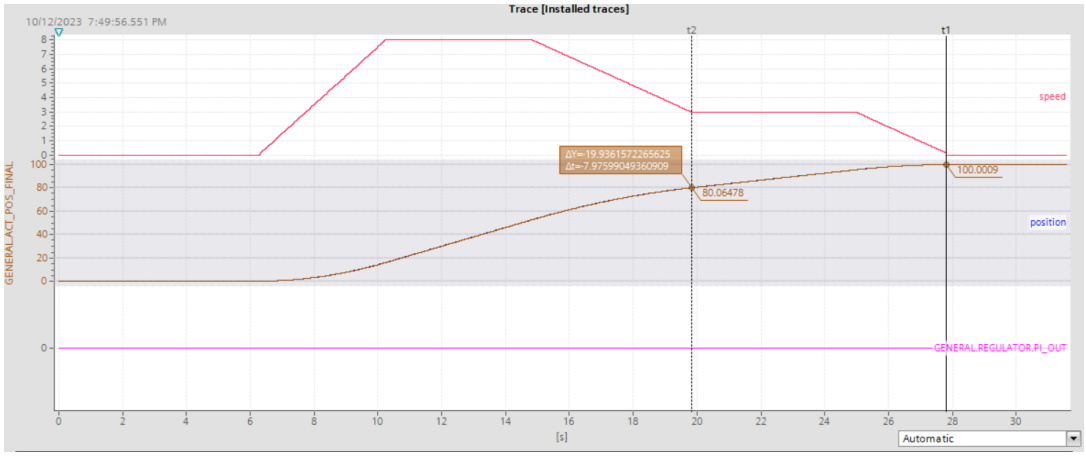
Prvi korak je odrediti varijable koje će se koristiti u Trace-u, što je prikazano na Slika 7.6.1. Varijable koje se koriste su trenutna i teorijska pozicija, brzina iz pozicionera, rampirana brzina koja se šalje pretvaraču te izlaz iz PI regulatora.



Slika 7.6.1. Konfiguracija Trace-a

8.1 Idealan slučaj sa potpunim profilom brzine

Slučaj kada je gibanje obuhvaćenom potpunim profilom brzine te nema poremećajnih veličina je prikazan na Slika 8.1.1. Odgovara Slika 3.1 u poglavlju 3 preko koje je postavljen matematički model sustava. Parametri korišteni za ovakav slučaj su prikazani u Tablica 8.1.1. Iz Slika 8.1.1. Pozicioniranje potpunim profilom brzine je moguće vidjeti kako potpuni profil brzine uključuje fazu akceleracije, fazu konstantne više brzine, fazu deceleracije sa više na nižu konstantnu brzinu, fazu niže konstantne brzine te fazu deceleracije do konačne pozicije. Postavljeni su vertikalni kursori kako bi se prikazala udaljenost gibanja nižom konstantom brzinom. Ovaj slučaj je testiran bez prisustva poremećajnih veličina te se time trenutna i teorijska brzina poklapaju. Iz tog je razloga izlaz iz regulatora nula, odnosno nije došlo do njegovog uključivanja.



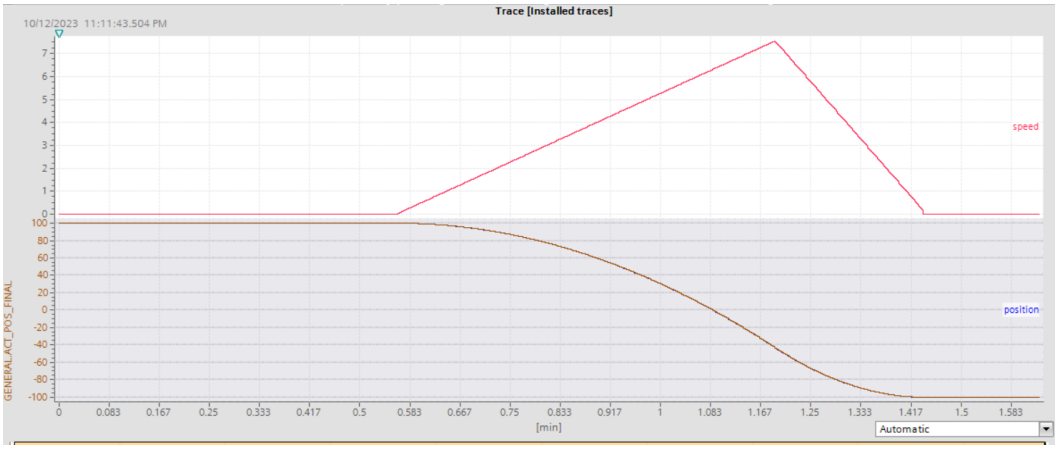
Slika 8.1.1. Pozicioniranje potpunim profilom brzine

Tablica 8.1.1. Ulazni i konfiguracijski parametri pozicioniranja potpunim profilom brzine

Naziv parametra	**Vrijednost parametra**
Trenutna pozicija	0 mm
Konačna pozicija	100 mm
Akceleracija	2 mm/s^2
Deceleracija	1 mm/s^2
Udaljenost gibanja nižom konst. brzinom	20 mm
Viša konstantna brzina	8 mm/s
Niža konstantna brzina	3 mm/s

8.2 Gibanje koristeći akceleraciju i deceleraciju

Sljedeće pozicioniranje koje je demonstrirano je slučaj kada se tijekom ubrzavanja ne stiže do vrijednosti više/niže konstantne brzine pa nastupa deceleracija do konačne pozicije, što je prikazano na Slika 8.2.1. Parametri koji su korišteni za dobivanje ovog slučaja prikazani su u Tablica 8.2.1. Može se primijetiti da je niža konstantna brzina 10 mm/s, a najveća moguća brzina koja se postiže kako bi se deceleracijom od 0.5 mm/s^2 stiglo do konačne pozicije je nešto viša od 7mm/s. Također ovo je slučaj kada se pozicioniranje izvršava u negativnom smjeru, koji je definiran kada je trenutna pozicija veća od konačne pozicije.



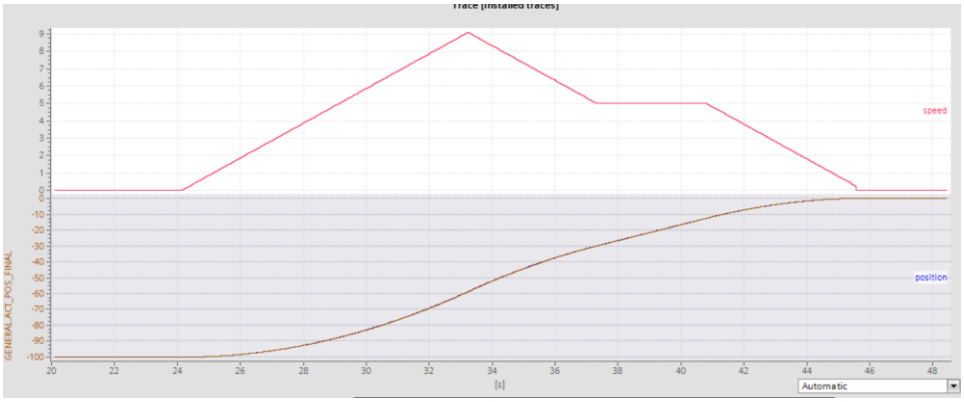
Slika 8.2.1. Pozicioniranje akceleracijom i deceleracijom

Tablica 8.2.1. Ulazni i konfiguracijski parametri pozicioniranja akceleracijom i deceleracijom

Naziv parametra	**Vrijednost parametra**
Trenutna pozicija	100 mm
Konačna pozicija	-100 mm
Akceleracija	0.2 mm/s^2
Deceleracija	0.5 mm/s^2
Udaljenost gibanja nižom konstantnom brzinom	50 mm
Viša konstantna brzina	18 mm/s
Niža konstantna brzina	10 mm/s

8.3 Gibanje koristeći profil akceleracija – deceleracija - niža konstantna brzina – deceleracija

Pozicioniranje kod kojeg se ne doseže vrijednost više konstantne brzine te se zbog toga nastavlja u fazi deceleracije sa više u nižu konstantnu brzine prikazano je na Slika 8.3.1. Parametri korišteni kod ovakvog pozicioniranja prikazani su u Tablica 8.3.1.



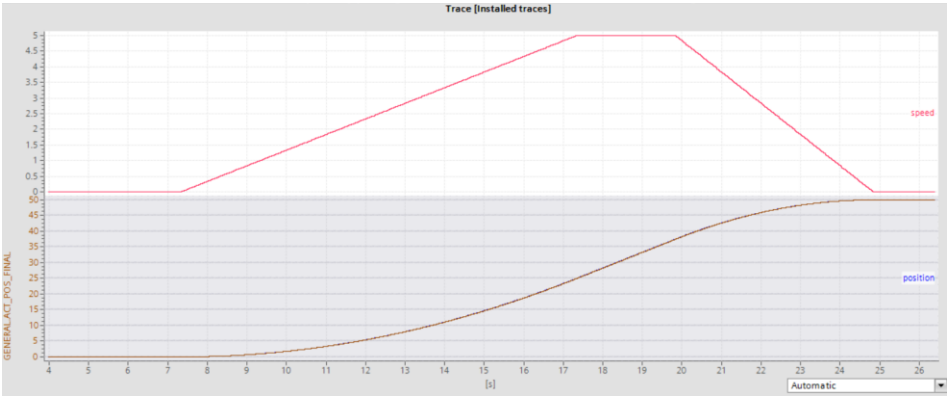
Slika 8.3.1. Pozicioniranje akceleracija - deceleracija - niža konstantna brzina- deceleracija

Tablica 8.3.1. Ulazni i konfiguracijski parametri pozicioniranja akceleracija - deceleracija- niža konstantna brzina - deceleracija

Naziv parametra	**Vrijednost parametra**
Trenutna pozicija	-100 mm
Konačna pozicija	0 mm
Akceleracija	1 mm/s^2
Deceleracija	1 mm/s^2
Udaljenost gibanja nižom konstantnom brzinom	30 mm
Viša konstantna brzina	10 mm/s
Niža konstantna brzina	5 mm/s

8.4 Gibanje profilom akceleracija – niža konstantne brzina – deceleracija

Nešto drugačija situacija nego u prošlom slučaju, pozicija u kojoj se treba početi gibati nižom konstantnom brzinom je prijeđena, ali brzina nije poprimila nižu konstantnu vrijednost te se ubrzavanje nastavlja. Kada brzina poprima nižu konstantnu vrijednost, gibanje se nastavlja do pozicije deceleracije u konačnu poziciju. Slika 8.4.1 prikazuje navedeni profil brzine tijekom pozicioniranja. Ulazni i konfiguracijski parametri ovog slučaja su prikazani u Tablica 8.4.1.



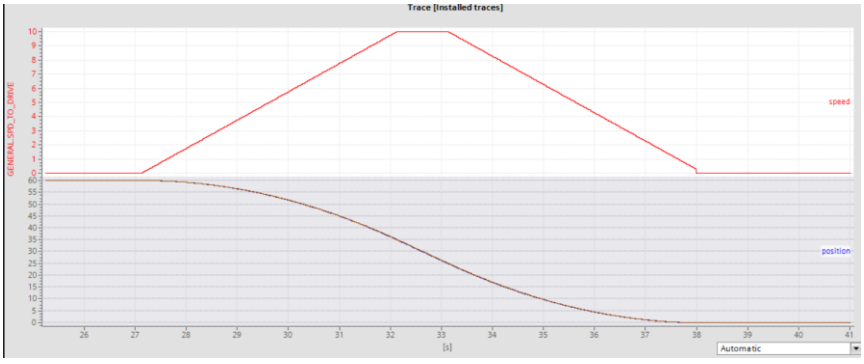
Slika 8.4.1. Pozicioniranje bez više konstantne brzine

Tablica 8.4.1. Ulazni i konfiguracijski parametri pozicioniranja bez više konstantne brzine

Naziv parametra	**Vrijednost parametra**
Trenutna pozicija	0 mm
Konačna pozicija	50 mm
Akceleracija	0.5 mm/s^2
Deceleracija	1 mm/s^2
Udaljenost gibanja nižom konstantnom brzinom	30 mm
Viša konstantna brzina	10 mm/s
Niža konstantna brzina	5 mm/s

8.5 Gibanje profilom brzine bez niže konstantne brzine

Kada je slučaj da se pozicioniranje ne želi izvršavati putem faze niže konstantne brzine, tada se obično parametar za udaljenost gibanja nižom konstantnom brzinom postavlja u vrijednost nula. Slučaj kada se gibanje izvršava profilom brzine akceleracija - viša konstantna brzina – deceleracija je prikazan na Slika 8.5.1.



Slika 8.5.1. Pozicioniranje bez niže konstantne brzine

Tablica 8.5.1 prikazuje parametre korištene kod slučaja bez niže konstantne brzine. Može se primijetiti kako je maksimalna postignuta brzina upravo viša konstantna brzina.

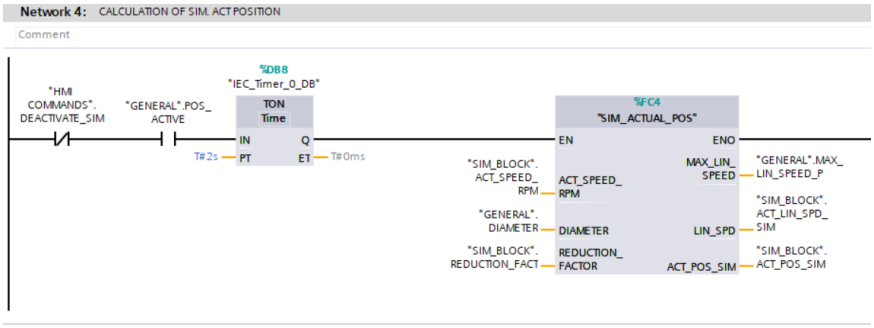
Tablica 8.5.1. Ulazni i konfiguracijski parametri pozicioniranja bez niže konstantne brzine

Naziv parametra	Vrijednost parametra
Trenutna pozicija	0 mm
Konačna pozicija	10 mm
Akceleracija	9 mm/s^2
Deceleracija	9 mm/s^2
Udaljenost gibanja nižom konstantnom brzinom	0 mm
Viša konstantna brzina	10 mm/s
Niža konstantna brzina	5 mm/s

8.6 Testiranje regulatora utjecajem poremećajnih veličina

Kao što je već bilo spomenuto regulator dolazi u stanje rada kada pogreška trenutne i teorijske pozicije bude veća od 2 mm. Kako bi se ostvario taj slučaj u simulacijskom okruženju, potrebno je izraditi blokove u programu koji će (jednim dijelom) na neki način spriječiti točan izračun pozicije.

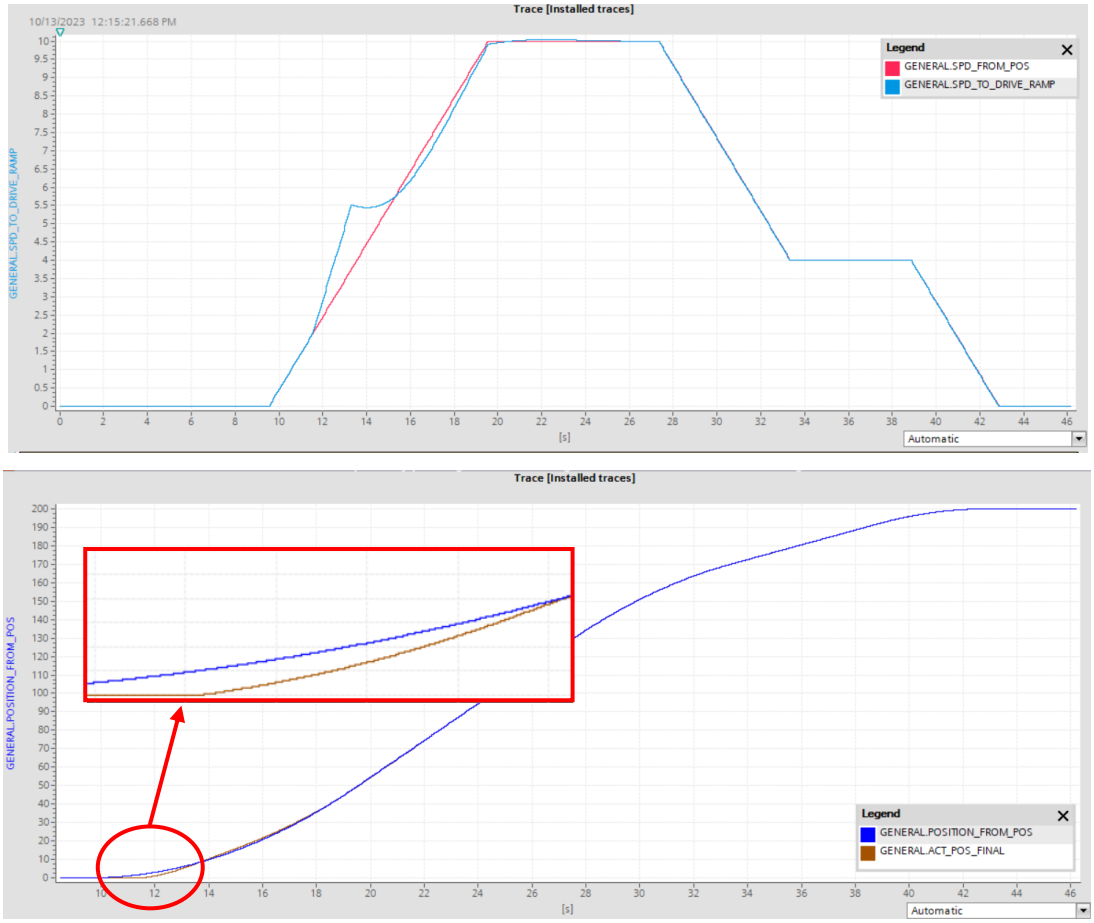
Prva poremećajna veličina je odgođen izračun trenutne pozicije. Kako bi se ona mogla simulirati potrebno je dodati TON timer zbog kojeg će se trenutna pozicija računati dvije sekunde nakon pokrenutog simuliranja. Slika 8.6.1 prikazuje TON timer prije bloka izračuna trenutne pozicije.



Slika 8.6.1. Izvedba poremećajne veličine odgođenog izračuna trenutne pozicije

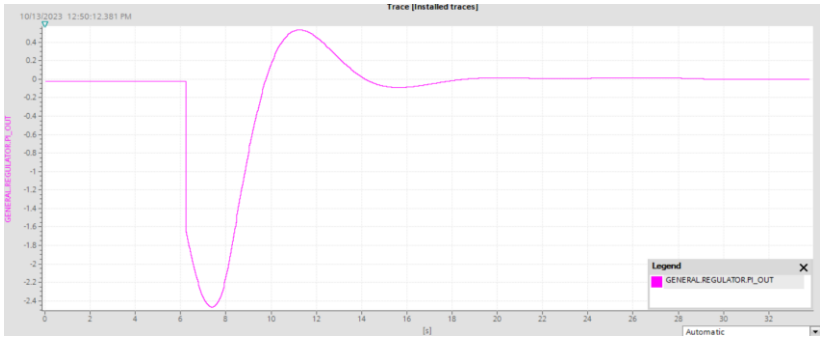
Slika 8.6.2 prikazuje razliku brzine iz pozicionera te one konačne koja se šalje pretvaraču zbog razlike trenutne pozicije u odnosu na teorijsku. U približenom prikazu vidljivo je kako trenutna pozicija zaostaje za teorijskom, odnosno poprima manju vrijednost te se na temelju toga brzina

sa pozicionera mora povećati za vrijednost korekcijskog djelovanja PI regulatora. Rezultat toga je konačna brzina označena plavom bojom koja se šalje pretvaraču.



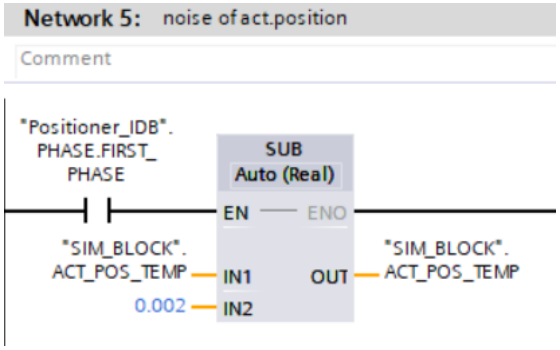
Slika 8.6.2. Prikaz brzine i pozicije tijekom djelovanja regulatora pri utjecaju poremećajne veličine

Slika 8.6.3 prikazuje regulacijsko djelovanje PI regulatora. Skok koji je prisutan na početku je rezultat uključenja regulatora i parametara regulatora koji su odabrani. Konačna brzina se rampira kako se ovaj skok ne bi prenio na brzinu prema pretvaraču. Nakon tog skoka, regulator na temelju svog djelovanja smanjuje razliku teorijske i trenutne brzine. Pojačanja regulatora korištena u ovom slučaju su Kp = 0.8 te Ki = 0.65. Dobivena su uzastopnom provjerom rezultata na temelju najbržeg smanjivanja greške.



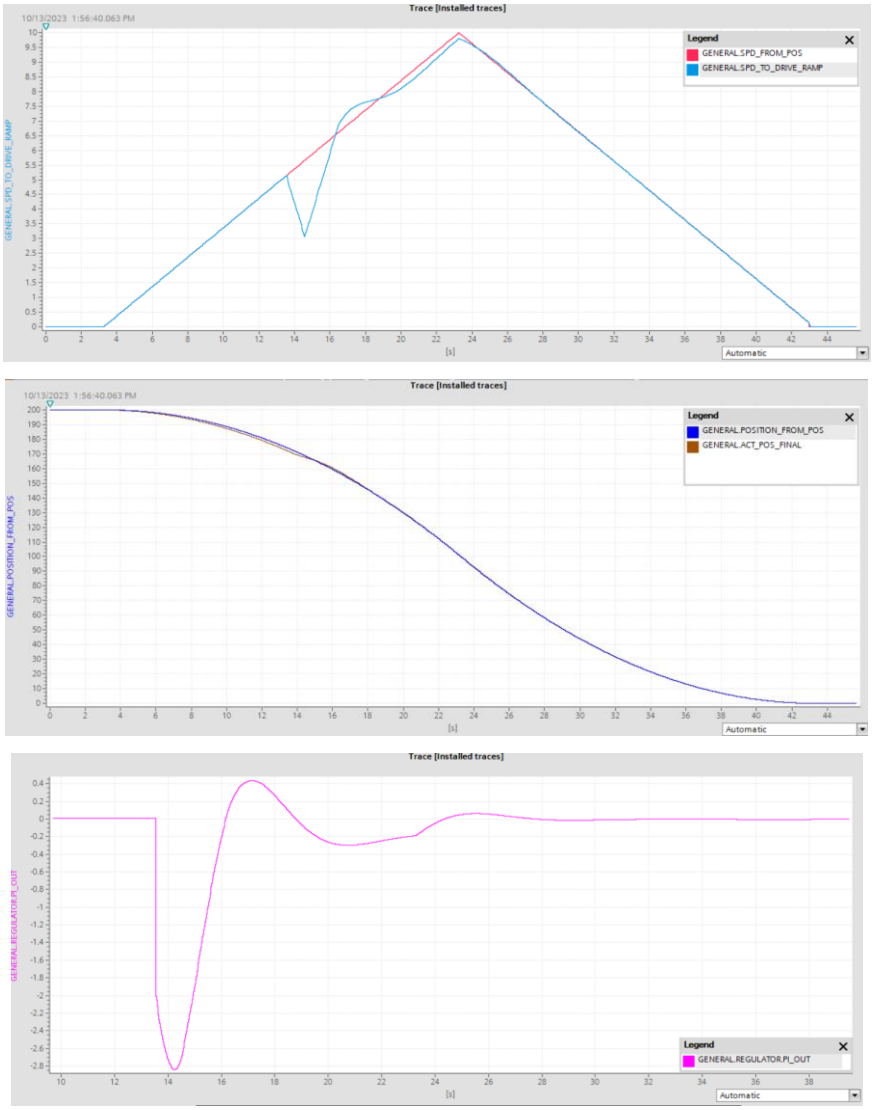
Slika 8.6.3. Korekcijsko djelovanje regulatora

Druga poremećajna veličina djeluje na sustav na način da se trenutna pozicija oduzima tijekom faze akceleracije. Ovakav poremećaj se kreira dodavanjem linije koda sa Slika 8.6.4 nakon računanja trenutne pozicije iz brzine u funkciji POMAK. Slika 8.6.4 prikazuje oduzimanje pozicije za vrijednost od 0.002 tijekom svakog ciklusa kada se gibanje izvršava u prvoj fazi, fazi akceleracije.



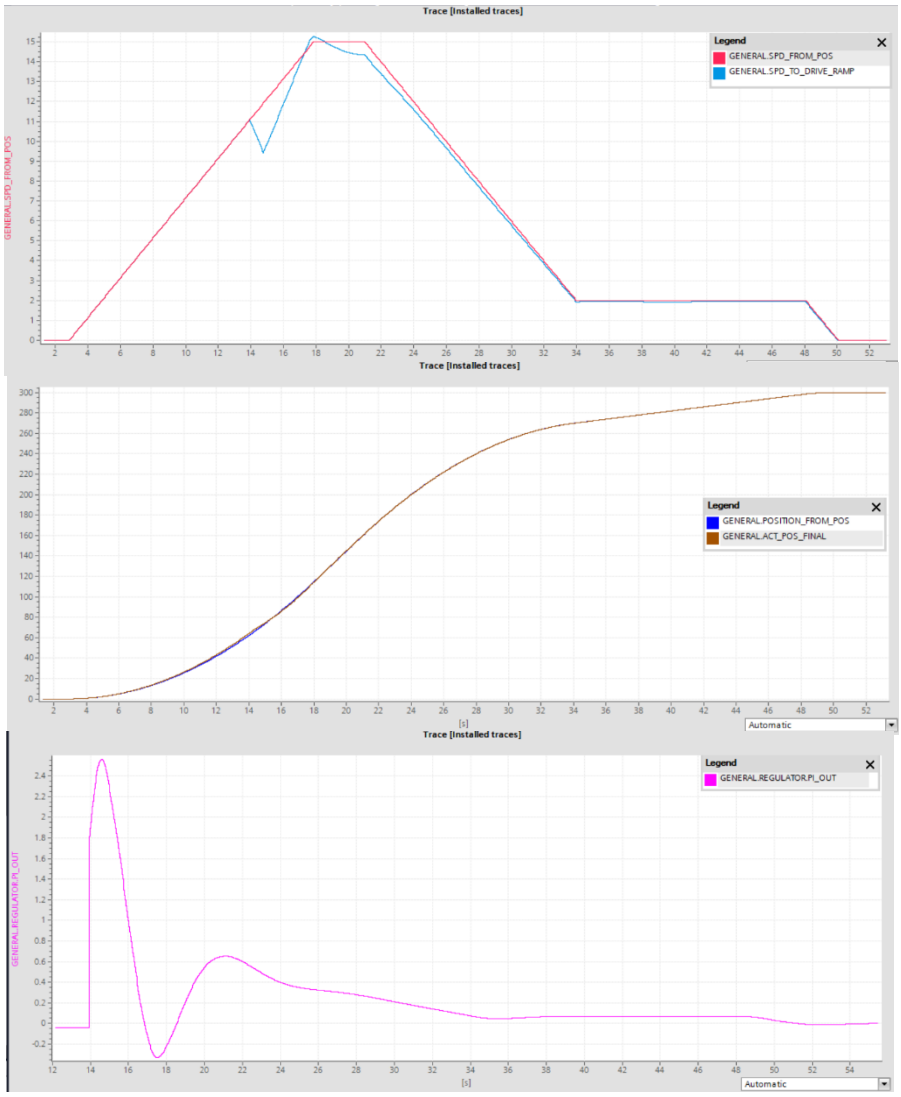
Slika 8.6.4. Izvedba poremećajne veličine oduzimanja pozicije tijekom akceleracije

Na Slika 8.6.5 je prikazana grafička ovisnost brzine i pozicije u vremenu uz priloženo regulacijsko djelovanje. Pozicioniranje se u ovom slučaju izvršava u negativnom smjeru. Time se zbog veće trenutne od teorijske pozicije, brzina prema pretvaraču treba smanjiti. Parametri regulatora korišteni za ovaj slučaj su Kp = 1 i Ki =0.9.



Slika 8.6.5. Prikaz brzine, pozicije i korekcijskog djelovanja pri utjecaju poremećajne veličine oduzimanja trenutne pozicije tijekom prve faze

Posljednji slučaj koji će biti prikazan u ovom radu što se tiče poremećajnih veličina je promjena redukcijskog faktora. Njegovom se promjenom mijenja brzina kojom se pozicioner giba, a time i izračun trenutne pozicije. Slika 8.6.6 prikazuje brzinu, poziciju i korekciju regulatora. Promjena redukcijskog faktora odrađena je sa vrijednosti 300 na vrijednost 290. Smanjenje redukcijskog faktora dovodi do povećanja brzine koja se očituje sa pretvarača te se time i trenutna pozicija povećava. Kako bi se uklonila pogreška nastala u odnosu na teorijsku poziciju, potrebno je brzinu sa pozicionera povećati za iznos korekcijskog djelovanja. Parametri korišteni za ovaj regulator su Kp = Ki = 0.9.



Slika 8.6.6. Prikaz brzine, pozicije i korekcijskog djelovanja pri utjecaju poremećajne veličine promjene redukcijskog faktora

9 ZAKLJUČAK

Kontrola pozicije je u svijetu automatizacije jedan od učestalijih elemenata kojim se ostvaruje visoka kvaliteta rada i povećanje efikasnosti procesa. Bazira se na principu uspoređivanju trenutne pozicije u kojoj se pozicioner nalazi sa onom teorijskom pozicijom, koja je dobivena matematičkim (teorijskim) putem. Izračun pozicija dobiven je upotrebom Tustinovog aproksimativnog postupka koji olakšava primjenu integrala brzine računalnim putem. Uloga regulatora u ovakvom sustavu je ispravljanje pogreške između trenutne i teorijske pozicije koja nastaje zbog poremećajnih veličina u realnom svijetu.

Glavni je cilj u ovom radu bio razviti algoritam upravljanja pozicionerom u programskom alatu *TIA Portal* te izvršiti razna testiranja njegovog rada. Detaljno je prikazan postupak programiranja PLC-a te vizualizacije HMI-a. Isto tako, prikazani su rezultati grafičkim putem na temelju tog programskog rješenja. Moguće je vidjeti kako se pozicioniranje izvršava od svega nekoliko centimetara do nekoliko desetina centimetara, iako ograničenja udaljenosti ne postoje. U ovisnosti o ulaznim i konfiguracijskim parametrima koji se koriste, moguće je napraviti nekoliko različitih profila brzine od kojih je svaki prikazan. Poremećajne veličine su se simulirale kako bi se provjerio rad regulatora te se može vidjeti da ovisno o vrsti poremećajne veličine, regulator nešto brže ili nešto sporije uspijeva ispraviti pogrešku.

LITERATURA

[1] Kuljača, Lj.; Vukić Z.: „Automatsko upravljanje sistemima“, Školska knjiga, Zagreb, 1985.

[2] Herbert M. Sauro: „Control and regulation of pathways via negative feedback“, s Interneta, https://royalsocietypublishing.org/doi/10.1098/rsif.2016.0848, 15. listopada 2023.

[3] Teja R.: „Open Loop System“, s Interneta, https://www.electronicshub.org/open-loop-system/, 15. listopada 2023.

[4] Electronic coach: „Closed-Loop Control System“, s Interneta, https://electronicscoach.com/closed-loop-control-system.html, 15. listopada 2023.

[5] Siemens: „STEP 7 and WinCC Engineering V17 System Manual“, s Interneta, https://cache.industry.siemens.com/dl/files/671/109798671/att_1071920/v1/STEP_7_WinCC_V17_enUS_en-US.pdf, 15. listopada 2023.

[6] Matika D.: „ Automatsko upravljanje: 2. predavanje“, 2020.

[7] Wikipedia: „Programmable logic controller“, s Interneta, https://en.wikipedia.org/wiki/Programmable_logic_controller, 15. listopada 2023.

[8] „Siemens SIMATIC S7-1500 CPU 1516-3 PN/DP“, s Interneta, https://www.quicktimeonline.com/6es7516-3an02-0ab0, 15. listopada 2023.

[9]Siemens: „SIMATIC S7-1500 CPU 1516-3 PN/DP Manual“, s Interneta, https://cache.industry.siemens.com/dl/files/914/59191914/att_915307/v1/s71500_cpu1516_3_pn_dp_manual_en-US_en-US.pdf, 15. listopada 2023.

[10] Siemens: „SIMATIC S7-1500, S7-1500R/H, SIMATIC Drive controller, ET 200SP, ET 200pro Structure and use of CPU memory, Function manual“ CPU 1516-3 PN/DP Manual“, s Interneta, https://cache.industry.siemens.com/dl/files/101/59193101/att_897341/v1/s71500_structure_and_use_of_the_PLC_memory_function_manual_en-US_en-US.pdf, 15. listopada 2023.

[11]Siemens: „SIMATIC HMI KTP700 Basic“, s Interneta, https://mall.industry.siemens.com/mall/en/us/Catalog/Product/6AV21232GB030AX0, 15. listopada 2023.

[12] Siemens: „SINAMICS S120/S150 List manual“, s Interneta, https://cache.industry.siemens.com/dl/files/911/99682911/att_863921/v1/S120_S150_List_Manual_LH1_0414_eng.pdf, 15. listopada 2023.

POPIS OZNAKA I KRATICA

PLC – Programmable logic controller

HMI – Human machine interface

LED – Light-emitting diode

CPU – Central processing unit

AC – Alternating current

DC – Direct current

TIA – Totally integrated automation

LD – Ladder diagram

OB – Organization block

FB – Function block

FC - Function

DB – Data block

PI - Proportional and integral

o/min – okretaji u minuti

SAŽETAK I KLJUČNE RIJEČI

U ovom radu cilj je izraditi sustav kontrole pozicije koji omogućuje pomicanje nekog objekta u dva smjera po jednoj osi. Najprije se upoznajemo sa osnovnim konceptima regulacije i regulacijskih krugova. Nadalje se opisuje bidirekcijski pozicioner na temelju matematičkog modela te uređaji koji se koriste za dobivanje ovakvog sustava. Dan je opis softverskog alata *TIA Portal* preko kojeg je izvršeno programiranje PLC-a i vizualizacija HMI-a. Programskim rješenjem predstavljen je detaljan uvid u programski kod korišten pri razvoj algoritma upravljanja, također su prikazani zasloni HMI uređaja. Na kraju su odrađena testiranja sustava kontrole pozicije kako bi se utvrdila njegova ispravnost i točnost.

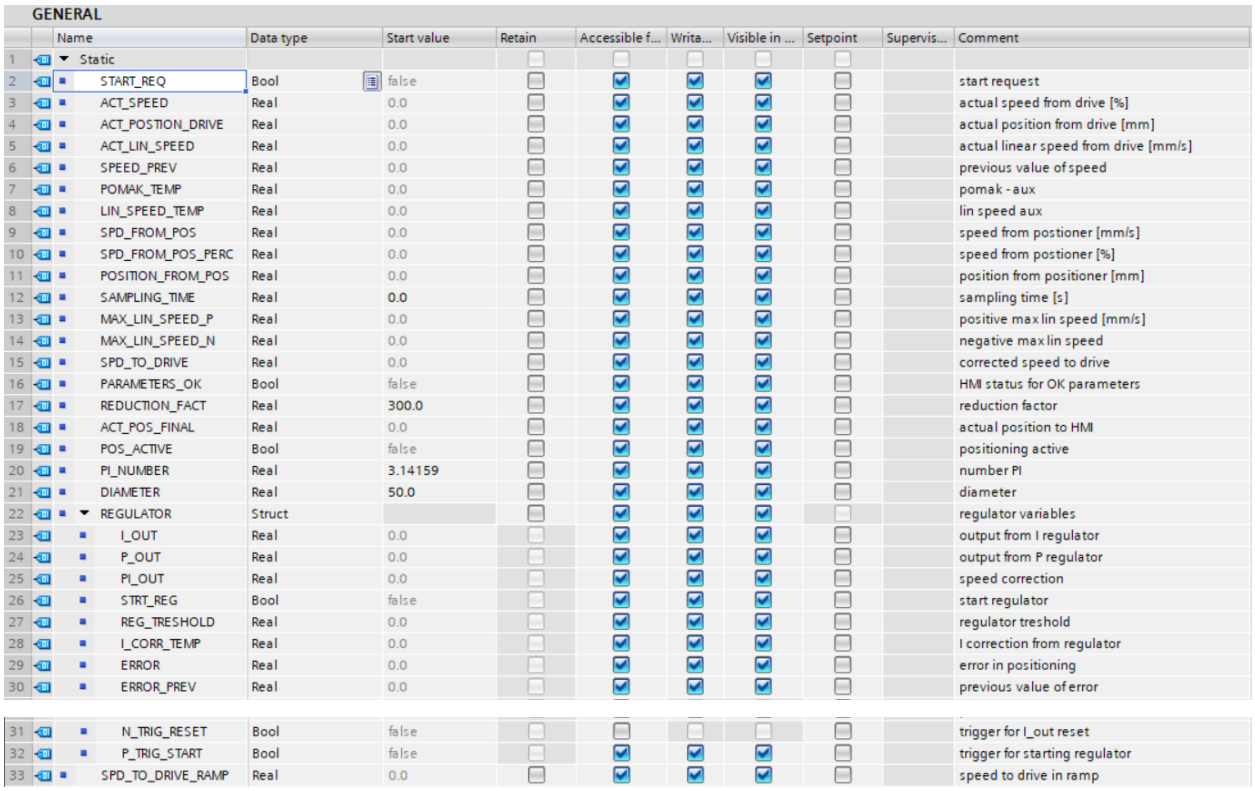
Ključne riječi : kontrola pozicije, regulacija, upravljanje, regulacijski krug, PI regulator, PLC, HMI, frekvencijski pretvarač, motor, Siemens, TIA Portal

ABSTRACT AND KEY WORDS

In this master's thesis, the goal is to create a position control system that allows moving an object in two directions along one axis. First, we get acquainted with the basic concepts of control and control systems. Furthermore, the bidirectional positioner based on a mathematical model and the devices used to obtain such a system are described. A description is given of the *TIA Portal* software tool through which PLC programming and HMI visualization were performed. The program solution presents a detailed insight into the program code used in the development of the control algorithm, and the screens of the HMI devices are also presented. At the end, the position control system was tested to determine its correctness and accuracy.

Keywords: position control, control, open loop system, closed loop system, PI controller, PLC, HMI, frequency converter, motor, Siemens, TIA Portal

DODATAK A – GENERAL DB



GENERAL

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Static									
2	START_REQ	Bool	false		☑	☑	☑			start request
3	ACT_SPEED	Real	0.0		☑	☑	☑			actual speed from drive [%]
4	ACT_POSTION_DRIVE	Real	0.0		☑	☑	☑			actual position from drive [mm]
5	ACT_LIN_SPEED	Real	0.0		☑	☑	☑			actual linear speed from drive [mm/s]
6	SPEED_PREV	Real	0.0		☑	☑	☑			previous value of speed
7	POMAK_TEMP	Real	0.0		☑	☑	☑			pomak - aux
8	LIN_SPEED_TEMP	Real	0.0		☑	☑	☑			lin speed aux
9	SPD_FROM_POS	Real	0.0		☑	☑	☑			speed from postioner [mm/s]
10	SPD_FROM_POS_PERC	Real	0.0		☑	☑	☑			speed from postioner [%]
11	POSITION_FROM_POS	Real	0.0		☑	☑	☑			position from positioner [mm]
12	SAMPLING_TIME	Real	0.0		☑	☑	☑			sampling time [s]
13	MAX_LIN_SPEED_P	Real	0.0		☑	☑	☑			positive max lin speed [mm/s]
14	MAX_LIN_SPEED_N	Real	0.0		☑	☑	☑			negative max lin speed
15	SPD_TO_DRIVE	Real	0.0		☑	☑	☑			corrected speed to drive
16	PARAMETERS_OK	Bool	false		☑	☑	☑			HMI status for OK parameters
17	REDUCTION_FACT	Real	300.0		☑	☑	☑			reduction factor
18	ACT_POS_FINAL	Real	0.0		☑	☑	☑			actual position to HMI
19	POS_ACTIVE	Bool	false		☑	☑	☑			positioning active
20	PI_NUMBER	Real	3.14159		☑	☑	☑			number PI
21	DIAMETER	Real	50.0		☑	☑	☑			diameter
22	REGULATOR	Struct			☑	☑	☑			regulator variables
23	I_OUT	Real	0.0		☑	☑	☑			output from I regulator
24	P_OUT	Real	0.0		☑	☑	☑			output from P regulator
25	PI_OUT	Real	0.0		☑	☑	☑			speed correction
26	STRT_REG	Bool	false		☑	☑	☑			start regulator
27	REG_TRESHOLD	Real	0.0		☑	☑	☑			regulator treshold
28	I_CORR_TEMP	Real	0.0		☑	☑	☑			I correction from regulator
29	ERROR	Real	0.0		☑	☑	☑			error in positioning
30	ERROR_PREV	Real	0.0		☑	☑	☑			previous value of error
31	N_TRIG_RESET	Bool	false		☐	☐	☐			trigger for I_out reset
32	P_TRIG_START	Bool	false		☑	☑	☑			trigger for starting regulator
33	SPD_TO_DRIVE_RAMP	Real	0.0		☑	☑	☑			speed to drive in ramp

DODATAK B – INVERTER IDB

Inverter_IDB

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Input									
2	DP_ADRESS	HW_SUBMODULE	0		☑	☑	☑			
3	EMERGENCY	Bool	false		☑	☑	☑			
4	RUN_REQUEST	Bool	false		☑	☑	☑			
5	RUN_CONSENT	Bool	false		☑	☑	☑			
6	SPEED_REQUEST	Real	0.0		☑	☑	☑			
7	MOTOR_MAX_RPM	Int	0		☑	☑	☑			
8	REV_DIR	Bool	false		☑	☑	☑			
9	Output									
10	DRIVE_REMOTE_MODE	Bool	false		☑	☑	☑			
11	DRIVE_FAULT	Bool	false		☑	☑	☑			
12	DRIVE_READY	Bool	false		☑	☑	☑			
13	DRIVE_PRESET_ON	Bool	false		☑	☑	☑			
14	DRIVE_RUN	Bool	false		☑	☑	☑			
15	MOTOR_REVERSE_DIR...	Bool	false		☑	☑	☑			
16	MOTOR_ACT_SPEED	Int	0		☑	☑	☑			
17	MOTOR_ACT_SPEED_R...	Int	0		☑	☑	☑			
18	InOut									
19	Static									
20	PLC->DRIVE	Struct		☐	☑	☑	☑			
21	DriveControl0	Struct		☐	☑	☑	☑			
22	08_RESERVED	Bool	false	☐	☑	☑	☑			
23	09_RESERVED_1	Bool	false	☐	☑	☑	☑			
24	10_CNTRL_BY_...	Bool	false	☐	☑	☑	☑			control by PLC/no control by PLC
25	11_SETPNT_INV	Bool	false	☐	☑	☑	☑			setpoint inversion 1=reverse direction 0=forwa
26	12_RESERVED_2	Bool	false	☐	☑	☑	☑			
27	13_M_POTENT...	Bool	false	☐	☑	☑	☑			motorized potentiometar setpoint raise
28	14_M_POTENT...	Bool	false	☐	☑	☑	☑			motorized potentiometar lower setpoint
29	15_RESERVED_3	Bool	false	☐	☑	☑	☑			
30	00_ON_OFF	Bool	false	☐	☑	☑	☑			ON/OFF (OFF1)
31	01_NO_COAST...	Bool	false	☐	☑	☑	☑			no coast-down/ coast-down (OFF2)
32	02_NO_QUICK...	Bool	false	☐	☑	☑	☑			no quick stop/ quick stop(OFF3)
33	03_ENBL_OP	Bool	false	☐	☑	☑	☑			enable operation/inhibit operation
34	04_ENBL_RAMP	Bool	false	☐	☑	☑	☑			enable ramp-function generator/inhibit ramp-fu
35	05_CONTINUE_...	Bool	false	☐	☑	☑	☑			continue ramp-function generator/freeze ramp-
36	06_ENBL_SETP...	Bool	false	☐	☑	☑	☑			enable setpoint/inhibit setpoint
37	07_ACKNOW_F...	Bool	false	☐	☑	☑	☑			acknowledge faults
38	DriveControl1	Struct		☐	☑	☑	☑			
39	MAIN_STPT	Int	0	☐	☑	☑	☑			
40	DRIVE->PLC	Struct		☐	☑	☑	☑			
41	DriveStatus0	Struct		☐	☑	☑	☑			
42	08_SPD_SETPO...	Bool	false	☐	☑	☑	☑			speed setpoint- actual value deviation in tolera
43	09_CNTRL_REQ	Bool	false	☐	☑	☑	☑			control request
44	10_F_OR_N_C...	Bool	false	☐	☑	☑	☑			f or n comparison value reached or exceeded
45	11_NO_TRQ_L...	Bool	false	☐	☑	☑	☑			no torque limit reached
46	12_OPN_HOLD...	Bool	false	☐	☑	☑	☑			open holding brake
47	13_NO_ALM_...	Bool	false	☐	☑	☑	☑			no alarm motor overtemperature
48	14_N_ACT	Bool	false	☐	☑	☑	☑			n_act >=0 0 = reverse direction 1=forward direc
49	15_NO_ALM_P...	Bool	false	☐	☑	☑	☑			no alarm power unit thermal overload
50	00_RDY_FOR_...	Bool	false	☐	☑	☑	☑			Ready for switching on
51	01_RDY	Bool	false	☐	☑	☑	☑			Ready
52	02_OP_ENBL	Bool	false	☐	☑	☑	☑			operation enable
53	03_FLT_PRESENT	Bool	false	☐	☑	☑	☑			fault present
54	04_NO_CST_ACT	Bool	false	☐	☑	☑	☑			no coasting active
55	05_NO_QCK_S...	Bool	false	☐	☑	☑	☑			no quick stop active
56	06_SWITCH_O...	Bool	false	☐	☑	☑	☑			switching on inhibited
57	07_ALM_PRESE...	Bool	false	☐	☑	☑	☑			alarm present
58	DriveStatus1	Struct		☐	☑	☑	☑			
59	SPD_ACT_VALUE	Int	0	☐	☑	☑	☑			actual value of speed
60	DP_READ_DIAGNOSTI...	Int	0		☑	☑	☑			
61	DP_WRITE_DIAGNOSTI...	Int	0		☑	☑	☑			
62	TIMERS	Array[0..10] of IEC_...			☑	☑	☑			
63	DP_READ_ERROR	Bool	false		☑	☑	☑	☑		
64	DP_WRITE_ERROR	Bool	false		☑	☑	☑			
65	EXTERNAL_RUN_CONS...	Bool	false		☑	☑	☑			
66	INTERNAL_RUN_CONS...	Bool	false		☑	☑	☑			
67	SPEED_REQUEST_LIMI...	Real	0.0		☑	☑	☑			
68	DRIVE_SPEED_REF	Int	0		☑	☑	☑			
69	MOTOR_ACT_SPD_ABS	Int	0		☑	☑	☑			

DODATAK C – HMI COMMANDS DB

HMI COMMANDS

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Static									
2	START_CMD	Bool	false		☑	☑	☑			start command
3	STOP_CMD	Bool	false		☑	☑	☑			stop command
4	KP	Real	0.0		☑	☑	☑			P gain
5	Ki	Real	0.0		☑	☑	☑			I gain
6	EMERGENCY	Bool	false		☑	☑	☑			emergency cmd for inverter
7	INCREASE_RED_FAC	Bool	false		☑	☑	☑			increase reduction factor
8	DECREASE_RED_FAC	Bool	false		☑	☑	☑			deacrease reduction factor

DODATAK D – POISTIONER DB

Positioner_IDB

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Input									
2	START_REQUEST	Bool	false		☑	☑	☑			start command
3	STOP	Bool	false		☑	☑	☑			stop command
4	POS_ACT	Real	0.0		☑	☑	☑			actual postion from drive
5	POS_TARGET	Real	0.0		☑	☑	☑			target position
6	Output									
7	SPEED	Real	0.0		☑	☑	☑			speed from positioner
8	SPEED_PERC	Real	0.0		☑	☑	☑			speed from positioner [%]
9	POSITION	Real	0.0		☑	☑	☑			position from positioner
10	POS_ACTIVE	Bool	false		☑	☑	☑			
11	InOut									
12	Static									
13	ACC	Real	0.0		☑	☑	☑			acc - confg. paramter
14	DEC	Real	0.0		☑	☑	☑			dec - confg. paramter
15	FST_SPEED_CONST	Real	0.0		☑	☑	☑			fast const. speed - confg. paramter
16	SLW_SPEED_CONST	Real	0.0		☑	☑	☑			slow const.speed - confg. paramter
17	SLW_SPEED_DIST	Real	0.0		☑	☑	☑			slow speed distance - confg. paramter
18	DISTANCE	Real	0.0		☑	☑	☑			distance to postion
19	T_ACC	Real	0.0		☑	☑	☑			acceleration time
20	T_DEC_0	Real	0.0		☑	☑	☑			deceleration time to 0
21	DISTANCE_ACC	Real	0.0		☑	☑	☑			acceleration distance
22	DISTANCE_DEC_0	Real	0.0		☑	☑	☑			deceleration distance to 0
23	SLW_SPD_POS	Real	0.0		☑	☑	☑			postion when slow speed start
24	FST_SPD_POS	Real	0.0		☑	☑	☑			
25	DEC_SPACE	Real	0.0		☑	☑	☑			space for deceleration from fast to slow speed
26	DEC_TO_SLW_SPD_POS	Real	0.0		☑	☑	☑			position when deceleration to slow speed begi
27	DEC_TO_0	Real	0.0		☑	☑	☑			position when deceleration to 0 begin
28	PREV_SPEED	Real	0.0		☑	☑	☑			previous speed
29	STRT_POSITIONER	Bool	false		☑	☑	☑			START CALCULATE
30	POS_PAR_OK	Bool	false		☑	☑	☑			position parameters ok
31	SPD_PAR_OK	Bool	false		☑	☑	☑			speed parameters ok
32	FWD_POS	Bool	false		☑	☑	☑			condition for FWD positioning
33	REV_POS	Bool	false		☑	☑	☑			condition for REV positioning
34	PHASE	Struct			☑	☑	☑			phases of moving
35	FIRST_PHASE	Bool	false		☑	☑	☑			
36	SECOND_PHASE	Bool	false		☑	☑	☑			
37	THIRD_PHASE	Bool	false		☑	☑	☑			
38	FOURTH_PHASE	Bool	false		☑	☑	☑			
39	FIFTH_PHASE	Bool	false		☑	☑	☑			
40	MOVE_TO_PHASE	Struct			☑	☑	☑			conditions for moving from one phase to anoth
41	FIRST_TO_SECOND	Bool	false		☑	☑	☑			
42	FIRST_TO_THIRD	Bool	false		☑	☑	☑			
43	FIRST_TO_FOURTH	Bool	false		☑	☑	☑			
44	FIRST_TO_FIFTH	Bool	false		☑	☑	☑			
45	SECOND_TO_THIRD	Bool	false		☑	☑	☑			
46	SECOND_TO_FIFTH	Bool	false		☑	☑	☑			
47	THIRD_TO_FOURTH	Bool	false		☑	☑	☑			
48	FOURT_TO_FIFTH	Bool	false		☑	☑	☑			
49	FIFTH_TO_END	Bool	false		☑	☑	☑			
50	THIRD_TO_END	Bool	false		☑	☑	☑			
51	TRIG	Array[0..10] of Bool			☑	☑	☑			array of bool for triggers
52	COND_FOR_POS_OK	Bool	false		☑	☑	☑			conditions for positioning OK
53	POS_TRESHOLD	Real	0.0		☑	☑	☑			treshold
54	NO_SLW_SPEED	Bool	false		☑	☑	☑			