Upravljanje industrijskim kontrolerom koristeći G-kod

Dominković, Marijan

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Marijan Dominković

Zagreb, 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentori:

Izv. prof. dr. sc. Tomislav Stipančić

Student:

Marijan Dominković

Zagreb, 2023.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se svome mentoru, izv. prof. dr. sc. Tomislavu Stipančiću, koji mi je uvelike pomogao s svojim stručnim savjetima pri izradi ovog rada. Također bih se zahvalio i asistentu, mag. ing. mech. Leonu Korenu, na pruženoj pomoći i praktičnim savjetima pomoću kojih je rad realiziran.

Zahvalio bih se i svojoj obitelji na pruženoj i velikoj podršci tijekom cjelokupnog obrazovanja, posebno tijekom studiranja. Hvala Vam, bez vas ne bih ostvario ovakve uspjehe.

Marijan Dominković

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne i diplomske ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa: **602 – 04 / 23 – 6 / 1**	
Ur.broj: **15 - 1703 - 23 -**	

ZAVRŠNI ZADATAK

Student: **Marijan Dominković** JMBAG: **0035215244**

Naslov rada na hrvatskom jeziku: **Upravljanje industrijskim kontrolerom koristeći G-kod**

Naslov rada na engleskom jeziku: **Industrial controller control using G-code**

Opis zadatka:

Razvoj tehnologije diktira uporabu sve novije opreme te kompatibilne programske podrške. Time se postojeća industrijska oprema unapređuje da bude u skladu s novim tehnologijama i konceptima, uključujući koncepte industrije 4.0.
U radu je potrebno izraditi programsko rješenje za Delta PLC upravljačko računalo koje upravlja starijim kartezijevim robotom koristeći G-kod (G-kod je smješten direktno na PLC-u čime je nestala potreba za dodatnim, vanjskim računalom). Rješenje treba uključivati sljedeće korake:

- proučiti dokumentaciju za Delta DVP15MC-06 PLC-e te se upoznati s mogućnostima implementacije G-koda,
- definirati G i M komande podržane od strane PLC-a,
- izraditi programsko rješenje u jeziku po izboru (Structured Text ili Ladder dijagram) koje će omogućiti pretvaranje unaprijed definiranog programskog koda u pomake robota,
- izrađeno programsko rješenje testirati na postavu u Laboratoriju za projektiranje izradbenih i montažnih sustava.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:	Datum predaje rada:	Predviđeni datumi obrane:
30. 11. 2022.	**1. rok:** 20. 2. 2023. **2. rok (izvanredni):** 10. 7. 2023. **3. rok:** 18. 9. 2023.	**1. rok:** 27. 2. – 3. 3. 2023. **2. rok (izvanredni):** 14. 7. 2023. **3. rok:** 25. 9. – 29. 9. 2023.

Zadatak zadao: Predsjednik Povjerenstva:

Doc. dr. sc. Tomislav Stipančić Prof. dr. sc. Branko Bauer

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

POPIS KRATICA

PLC	Programabilni logički upravljač (eng. Programmable Logic Controller)
CNC	Računalno numeričko upravljanje (eng. Computer Numerical Control)
CAD	Oblikovanje pomoću računala (eng. Computer Aided Design)
CAM	Proizvodnja pomoću računala (eng. Computer Aided Manufacturing)
IoT	Internet stvari (eng. Internet Of Things)
SCARA	(eng. Selective Compliant Articulated Robot for Assembly)
CAN	(eng. Controller Area Network)

POPIS OZNAKA

Oznaka	Jedinica	Opis
v	mm/s	Brzina
a	mm/s^2	Ubrzanje/Usporavanje
-	mm/s^3	Trzaj

SAŽETAK

U ovome radu izrađujemo programsko rješenje s kojim bi koristeći se G kodom upravljali starijim kartezijskim koordinatnim robotom. Samim robotom upravljamo pomoću programabilnog logičkog upravljača (skraćeno PLC) tvrtke Delta.

Kroz uvod upoznati ćemo se s osnovnim pojmovima ovog rada: roboti, PLC, G kod i industrija 4.0. Prvo ćemo predstaviti i sam kartezijski koordinatni robot, objasniti njegovu strukturu, prednosti i mane, te područja gdje se takva struktura upotrebljava. Taj se robot nalazi u sklopu Laboratorija za projektiranje izradbenih i montažnih sustava, te se prethodno koristio u industriji i kasnije u edukacijske svrhe. Zatim ćemo upoznati PLC pomoću kojega ćemo upravljati s robotom, njegovim mogućnostima i komunikacijskim.

Koristeći softver CANopen Builder za izradu programa i upravljanje PLC-om, izraditi ćemo program koristeći se ljestvičastim dijagramom i jednu od značajki softvera, CNC urednik, za kreiranje putanje s G kodom koju će robot pratiti.

U posljednja dva poglavlja pokazati ćemo detaljan opis programa s korištenim blokovima i koja je njihova funkcija, primjer korištene putanje G koda i kako sam program funkcionira.

Ključne riječi: programsko rješenje, G kod, kartezijski koordinatni robot, programablini logički upravljač, Delta, struktura, CANopen Builder, ljestvičasti dijagram, putanja

SUMMARY

In this paper the main task is to write a program solution that will use G code to operate an older cartesian coordinate robot. The robot itself is controlled by a programmable logic controller (or PLC for short) manufactured by the company Delta.

In the beginning of the paper we will introduce the basic concepts that relate to this theme: robots, PLCs, G code and Industry 4.0. We'll take a look at the cartesian coordinate robot, explain the configuration, it's benefits and downsides and lastly where this configuration is commonly used. The robot is located in the Laboratory for the design of manufacturing and assembly systems, with it being used in industries and later for educational purposes. After this we're going to talk about the PLC used to control the robot and the softver we use to achieve that.

Using the CANopen Builder softver used for programming and controlling PLCs, using one of it's methods, the ladder diagram, to create a program to control the robot and using it's other feature, the CNC program editor, to make a G code program and a path which the robot will follow.

In the last two chapters you will see a detailed description of the program and the function blocks that it uses, as well as a G code program created and how the program functions.

Key words: program, G code, cartesian coordinate robot, programmable logic controller, Delta, configuration, CANopen Builder, Ladder diagram, path

1. UVOD

1.1. Roboti

Roboti su automatizirani strojevi višestruke namjene koji se sastoje od upravljačkih uređaja, senzora i konstrukcije s pripadajućim pogonskim uređajima. Riječ robot potječe iz češkog jezika (češki *Robot,* prema *robota* što znači tlaka, kmetski rad), gdje ju prvi upotrebljava Karl Čapek 1920. godine u svojoj drami R.U.R. Prema ISO 8373 robot je automatski upravljani, programibilni, višenamjenski manipulator koji se može programirati u tri ili više osi, uz to može biti stacionaran ili mobilan.

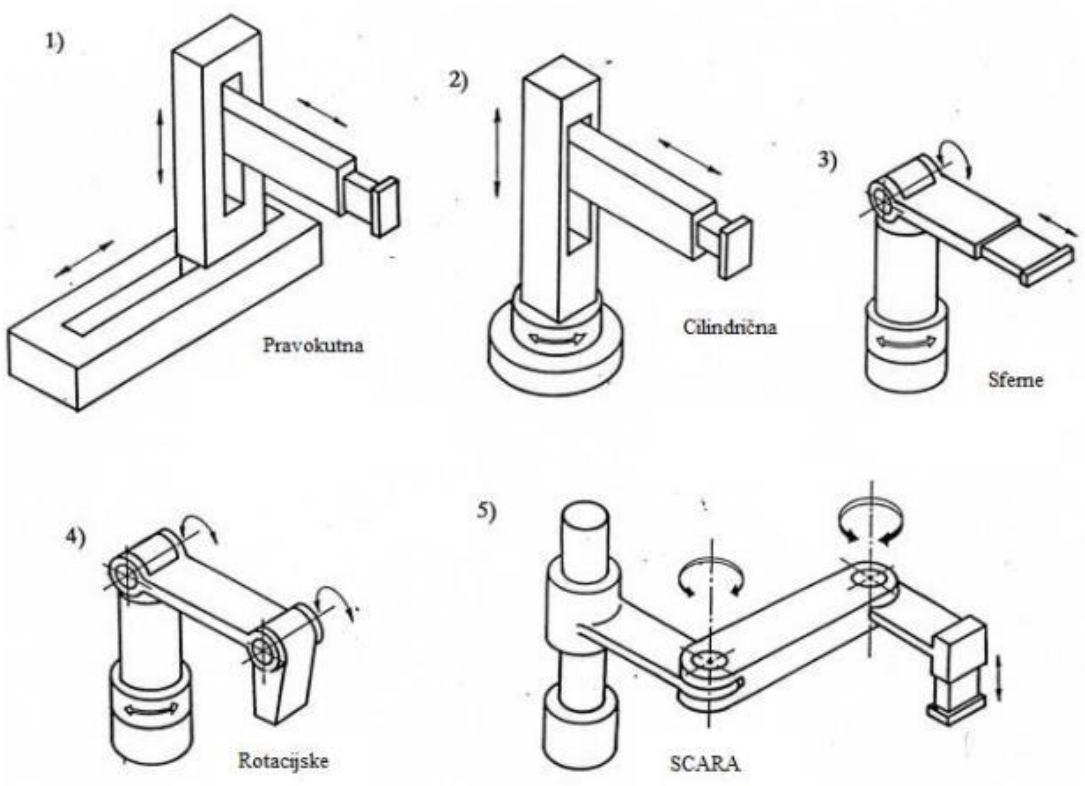
Roboti se dijele po stupnju pokretljivosti (statički i mobilni roboti), namjeni (industrijski, medicinski, edukacijski, podvodni, vojni roboti, osobni roboti, roboti za istraživanje), veličini (makroroboti, mikroroboti i nanoroboti) i strukturi konstrukcije (mehatronički, biotronički i bioroboti).

Interdisciplinarno znanstveno područje koje se bavi projektiranjem, konstruiranjem, upravljanjem i primjenom robota naziva se robotika. Zasnovano je na mehatronici i objedinjuje znanstvena područja poput strojarstva, elektrotehnike, elektronike, automatike, računarstva i umjetne inteligencije.

U današnje vrijeme najveću primjenu nalaze industrijski roboti. Industrijski robot je robotski sustav koji se koristi za proizvodnju, idealni su za poslove koji se smatraju teškim, monotonim i neprikladnim za ljude. Koriste se u procesima gdje se traži visoka i ujednačena kvaliteta, uz veliku preciznost. Tipično se primjenjuju kod zavarivanja, bojanja, sastavljanja, rastavljanja, pakiranja i označivanja, pregledu proizvoda i testiranja, te može pomoći pri rukovanju materijalom.

Postoji više struktura industrijskih robota koje se razlikuju po vrsti i položaju svake osi na robotu. Dvije vrste osi su translacijska (linearna) T i rotacijska R, te s kombinacijom tih osi dobijemo više različitih struktura:

- Kartezijska struktura – 3 translacijske osi (TTT)
- Cilindrična struktura – 2 translacijske i 1 rotacijska os (TTR)
- Kvazi cilindrična struktura – 1 translacijska i 2 rotacijske osi (RTR)
- Sferna struktura – 1 translacijska i 2 rotacijske osi (RRT)
- Rotacijska struktura – 3 rotacijske osi (RRR)
- SCARA struktura – 1 rotacijska i 3 rotacijske osi (RRRT)
- Zglobna struktura – 6 rotacijskih osi



Slika 1.1. – Tipovi robota

Kartezijski robot također zvan pravokutni i x-y-z robot biti će detaljnije obrađen u sljedećem poglavlju.

1.2. PLC upravljač

PLC (hrv. Programibilni logički upravljač) uređaji napravljeni su kako bi zamijenili stare i krute relejne upravljačke sustave čija bi prilagodba iziskivala puno vremena i velike troškove. Relejni upravljački sustavi imaju pokretne mehaničke dijelove, te su osjetljivi na nepovoljne utjecaje procesa. Pri promjeni proizvodnog programa relejni sklopovi bi se nanovo sastavljali uz promjenu ožičenja. PLC uređaji nemaju takvih nedostataka. Odlika im je velika fleksibilnost jer nije potrebno mijenjati sklopove nego se jednostavno reprogramira za novi proizvodni proces. Manji su od relejnih sklopova i otporni na nepovoljne utjecaje proizvodnje, npr. prašina, vlaga, vibracije, elektromagnetski utjecaji procesa u čijoj se neposrednoj blizini i nalazi što ih čini vrlo pouzdanima.

Programiranje PLC-a provodi se pomoću računala s odgovarajućim softverima poput CANopen Builder-a (detaljnije pojašnjeno u naknadnom poglavlju) i raznih drugih u jednom od tri programska jezika: Ladder diagram, Statement list i Function block. Sve tri metode programiranja podjednako su prihvaćene među inženjerima.

Postoje dva glavna tipa PLC uređaja:

- Fiksni PLC (eng. Fixed/Compact PLC) – također zvan Fixed I/O PLC je upravljač kojemu je broj I/O fiksan. Input/Output dijelovi su integrirani na sam mikroupravljač i njihov broj je određen od strane proizvođača.
- Modularni PLC (eng. Modular PLC) – upravljač koji ima mogućnost proširenja pomoću različitih modula kao što je I/O modul koji povećava broj ulaza/izlaza. Moduli su odvojeni od mikroupravljača, te se moraju fizički spojiti kako bi se stvorio PLC upravljački sustav.

S povećanjem broja ulazno/izlaznih stezaljki mora se povećati i snaga procesora, količina memorije, složenost uređaja i s time njegova cijena. Neki od PLC uređaja uz digitalne ulaze/izlaze imaju i analogne ulaze/izlaze, imaju mogućnost izvođenja matematičkih operacija nad realnim brojevima, PID regulaciju, mogućnost proširenja itd.

1.3. G kod

G kod ili NC kod najčešći je programski jezik korišten kod CNC uređaja (eng. Computer numerical control) u CAD/CAM programima (eng. Computer-aided design/manufacturing) i pri procesima OOČ (Obrada odvajanjem čestica) poput tokarenja i glodanja, te u aditivnim tehnologijama poput 3D ispisa. Prvi programski jezik za upravljanje NC stroja razvijen je na MIT-u tokom kasnih 1950-ih, dok je početkom 1960-ih standardiziran.

Kod koristi različite naredbe koje počinju velikim slovima i završavaju brojevima kojim određuju čime funkcija upravlja. Svaka linija koda definira potrebnu brzinu, poziciju ili pomak kojom stroj mora izvršiti radnju. Programi za izradu jednostavnijih dijelova mogu sadržavati stotine, ako ne i tisuće linija koda koji svi moraju biti ispravni kako bi se postigao željeni oblik izratka. U pisanju koda prvo moramo definirati naredbe, zatim u istoj ili sljedećoj liniji pišemo koordinate položaja i vrijednosti brzine. Ukratko ćemo pojasniti neke naredbe korištene pri programiranju. Naredbe koje počinju velikim slovom:

- G – upravljaju kretanjem osi CNC stroja, koordinatnim sustavom, radnim točkama i geometrijom alata/obratka. Na primjer:
    - Naredba G01 služi za linearnu interpolaciju.
    - Naredba G42 uključuje desnu kompenzaciju promjera alata.
    - Naredbe G54 – G59 postavljaju radnu nultočku predmeta obrade.
- M – označavaju razne funkcije koje stroj može obavljati, ne upravljaju kretanjem osi, te su za različite strojeve drugačije. Na primjer:
    - Naredba M02 označava kraj programa.
    - Naredba M06 označava automatsku promjenu alata.
    - Naredba M08 označava uključivanje pumpe emulzije.
- T – označava redni broj alata.
- S – označava brzinu vrtnje svrdla.
- F – označava posmak ili posmičnu brzinu.
- N – služi za označavanje linija koda.

Programi poput CATIA-e i SolidWorks-a imaju mogućnost izrade CAM procesa. CAM je program u kojemu se pomoću dizajna u CAD-u izrađuje putanja alata za obradu. CAM može koristiti više alata i različitih putanja kako bi se optimizirala putanja alata i ostvario čim efektivniji proces obrade. Kada je izrađena optimalna putanja za obradu CAM može generirati

potrebni G kod. To je moguće pomoću postprocesora koji služi kao most između CAM-a i stroja. Zadatak mu je podatke iz CAM-a prevesti u G kod koji stroj razumije. No jednu stvar treba napomenuti, to je da putanja alata često nije najbrža ni najučinkovitija za proces posebno kod složenijih geometrija, te je potreban operater koji će ju urediti.

1.4. Industrija 4.0

Industrija 4.0 je trenutni trend automatizacije i razmjene podataka u proizvodnim tehnologijama. Pojam „Industrija 4.0“ zaživio je u 2011. godini na velesajmu u Hannoveru, najvećem svjetskom industrijskom velesajmu.

Industrija 4.0 revolucionira način proizvodnje, poboljšavanje i distribuciju proizvoda. Obuhvaća široki spektar novih tehnologija poput umjetne inteligencije, nanotehnologije, biotehnologije, 3D ispisa, robotike, Internet stvari (IoT, eng. Internet of Things), računarstvo u oblak (eng. Cloud Computing), Big Data, te Strojno učenje (eng. Machine learning) i integrira ih u proizvodne pogone. Time dolazi do stvaranja tzv. „Pametnih tvornica“. Unutar pametnih tvornica kibernetičko-fizički sustavi nadziru fizičke procese, stvaraju virtualnu kopiju fizičkog svijeta i čine decentralizirane odluke. Takvi sustavi komuniciraju međusobno i s ljudima u stvarnom vremenu putem Internet of Things.

Koncept uzima u obzir povećanu informatizaciju u proizvodnim industrijama, u kojima su fizički objekti jednostavno integrirani sa informacijskom mrežom. Kao rezultat toga proizvodni sustavi su okomito umreženi s poslovnim procesima unutar tvornice i poduzeća, te vodoravno povezani s mrežama vrijednosnog lanca kojima se može upravljati u realnom vremenu, od trenutka narudžbe sve do izlazne logistike.

Prema službenoj studiji Odbora za industriju, istraživanje i energiju Europske komisije iz 2016. godine postoji šest glavnih trendova:

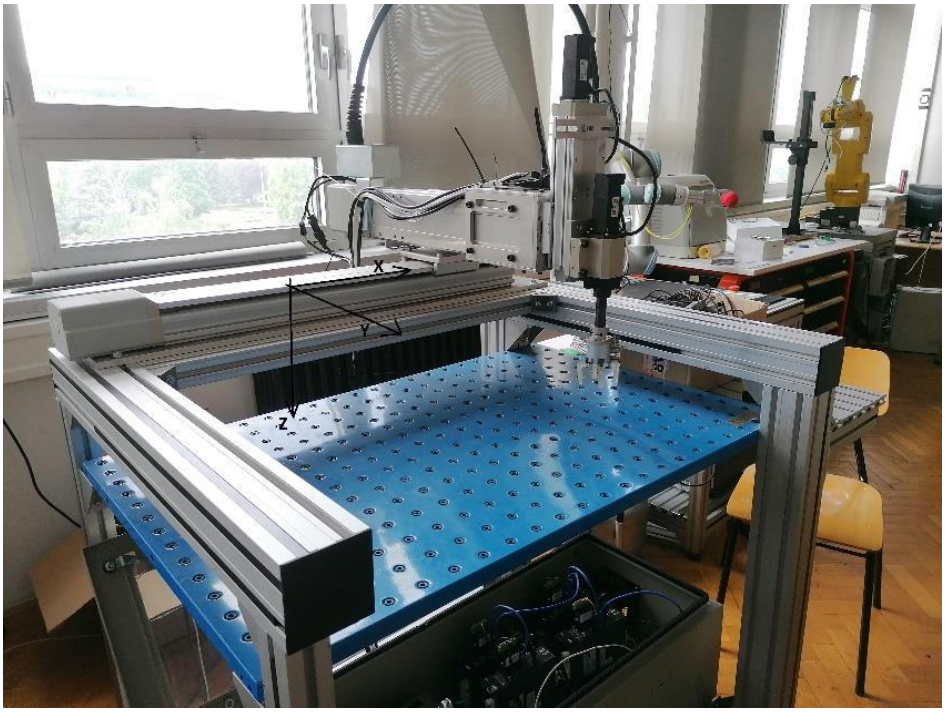
- Interoperabilnost: Kibernetsko-fizikalni proizvodni sustavi omogućuju ljudima i pametnim tvornicama da se povežu i komuniciraju jedni s drugima.
- Virtualizacija: virtualna kopija pametne tvornice kreirana je povezivanjem podataka senzora s virtualnim modelom tvornice i simulacijskog modela.
- Decentralizacija: sposobnost kibernetsko-fizičkog sustava za donošenje vlastitih odluka i za lokalnu proizvodnju zahvaljujući tehnologijama kao što su 3D ispis.
- Sposobnosti realnog vremena: sposobnost za prikupljanje i analizu podataka i neposredan uvid u iste.
- Orijentiranost na usluge

- Modularnost: fleksibilna prilagodba pametnih tvornica zahtijevanim promjenama zamjenom ili proširenjem pojedinačnih modula.

Koncepti i tehnologija Industrije 4.0 primjenjiva je u svim granama industrije poput diskretne proizvodnje, naftne i plinske industrije, rudarstva itd.

2. KARTEZIJSKI KOORDINATNI ROBOT

Kartezijski koordinatni robot, također zvan linearni robot ili x-y-z robot, industrijski je robot čije se glavne osi upravljanja linearno pomiču po trima osima kartezijskog koordinantog sustava (x, y i z) koje su međusobno okomite. U ovome slučaju kartezijski robot koji koristimo (Slika 2.1.) za izradu završnog rada sadrži dodatnu os, os (R), koja se nalazi na Z osi i služi za dodatno upravljanje hvataljkom robota.



Slika 2.1. – Kartezijski koordinantni robot

Kartezijski roboti popularna su opcija zbog fleksibilnosti konfiguracije što daje veliku mogućnost namještanja brzine, točnosti, posmaka i veličine robota. Uz to što su jedni od najčešćih industrijskih robota, kartezijska struktura često nalazi uporabu kod numerički upravljanih (CNC) strojeva i 3D pisača.

Kartezijski se roboti primjenjuju u istim područjima gdje se koriste SCARA i 6-osne tipove. No kartezijski roboti posjeduju određene prednosti u odnosu na navedene konfiguracije. Prva je veličina radnog prostora. Posjeduju pravokutni radni prostor gdje se značajan postotak tog prostora koristi kao aktivno radno područje robota. S druge strane SCARA i 6-osni tipovi robota posjeduju kružne ili ovalne radne prostore koji ostavljaju puno neiskorištenog (mrtvog) prostora posebno ako je potreban veliki radni hod robota. Kartezijski robot može se konstruirati pomoću bilo koje vrste linearnog pogona s mogućnošću različitih mehanizama pogona poput remena, kugličnog vijka, pneumatskog aktuatora ili linearnog motora. Uz to moguće je koristiti pogone

s zupčastom letvom i zupčanikom, ali se takav pogon češće koristi kod Gantry (portalnih) robota s vrlo dugim radnim hodom. Zato ovi tipovi robota posjeduju veliku točnost i ponovljivost čak i u odnosu na SCARA i 6-osne tipove.

Također bitna prednost im je lakoća programiranja zbog jednostavnije kinematike (tri linearne osi umjesto više rotacijskih osi). Zbog svoje krute strukture ovi roboti mogu manipulirati s relativno velikim teretima, pa se često koriste za aplikacije podizanja i spuštanja, utovar alatnih strojeva i slaganje dijelova u spremnike.

Glavni nedostatak kartezijskih robota je što im je potreban veliki volumen prostora za rad, iako se cijeli prostor ne koristi. Kartezijski robot zauzima isti volumen prostora koliko i X-os. Ako, na primjer, imate radni prostor koji zahtijeva 500 mm hoda, X-os će zauzeti taj prostor. Kartezijski roboti zauzimaju najveću potrebnu površinu od svih konfiguracija robota. Izložene površine za vođenje zahtijevaju pokrivanje u korozivnim ili prašnjavim okruženjima i ne mogu raditi pod vodom, kao što to mogu činiti SCARA roboti.

Popularne primjene kartezijske strukture su računalni numerički upravljački stroj (CNC stroj) i 3D ispis. Najjednostavnija primjena koristi se u strojevima za glodanje i crtačima gdje se alat, kao što je olovka, premješta preko X-Y ravnine i podiže i spušta na površinu kako bi se stvorio precizan dizajn. Eng. „Pick and place“ strojevi jedna su od primjena za robote s kartezijskim koordinatama. Na primjer, kartezijski roboti iznad glave primjenjuju se za kontinuirani utovar i istovar dijelova na proizvodnim linijama CNC tokarilica, izvodeći operacije skupljanja i postavljanja teških tereta u 3 osi (X, Y, Z) s velikom brzinom i velikom preciznošću pozicioniranja. Općenito, kartezijski roboti iznad glave prikladni su za mnoge sustave automatizacije.

3. Delta PLC i CANopen

3.1. DVP-15MC

Programibilni logički upravljač (PLC) korišten za upravljanje kartezijskim koordinatnim robotom u ovom završnom radu spada u seriju kontrolera DVP-MC tvrtke Delta. Ova serija PLC-a ima više ugrađenih komunikacijskih sučelja i može se jednostavno povezati s drugom opremom bez dodatnih komunikacijskih modula. Posjeduje komunikacijske priključke CANopen i Ethernet, te uz dodatne module za proširenje PLC se može povezati s raznim uređajima za industrijsku automatizaciju putem EtherCAT, DeviceNet, ProfiBus i RS-485 priključaka. Ethernet omogućuje komunikaciju između PLC-a i računala, dok se korištenjem priključka CANopen Motion putem komunikacijskog protokola CANopen DS402 omogućuje komunikacija između PLC-a i servo pogona.

Slika 3.1. – Programabilni logički upravljač DVP15MC11T-06

Dva su modela unutar serije DVP-15MC upravljača: DVP15MC11T i DVP15MC11T-06. Kapaciteti programa i komunikacijski priključci su jednaki dok je razlika u broju osi kojima mogu upravljati. Korišteni Delta PLC punog je naziva i oznake **DVP15MC11T-06.** Prikazan je na slici 3.1. Za napajanje koristi istosmjerni napon od 24 V, dok mu je izlaz tranzistorskog tipa. Ovaj model PLC-a može upravljati s 6 realnih osi (oznake od 1 do 16), s mogućnošću upravljanja do 16 virtualnih osi. Podržava instrukcije za jedno osno upravljanje pokreta kao što su brzina, pozicija, okretni moment (eng. Torque), vraćanje u početni položaj (eng. Homing) i instrukcije za više osno upravljanje kao što su elektronički zupčanik, E-Cam (eng. Electronic

Cam), rotacijski rez i posjeduje mogućnost čitanja i izvršavanja naredbi G-koda. Podržava knjižnicu standardnih instrukcija za upravljanje kretnjom definiranoj od strane međunarodnih organizacija što olakšava učenje i brži razvoj projekata.

PLC sadrži brzi procesor od 1 GHz, te ima mogućnost sinkroniziranja osi u vremenu 2 ms za 4 osi ili 4 ms za 8 osi. Kapacitet programa koji se može isprogramirati na PLC-u iznosi 20 MB i memorijski kapacitet za varijable iznosi jednako toliko. PLC sadrži 16 ulaznih i 8 izlaznih priključaka, dva priključka za Ethernet i utor za SD karticu. Veličina G-kod programa na PLC-u može biti do 256 kB, a istovremeno može biti i do 64 takvih G-kod programa. Računalno sučelje u kojemu se programira ovaj PLC je program *CANopen Builder*.

3.2. CANopen

CANopen je komunikacijski protokol i specifikacija profila uređaja za sustave koji se koriste u automatizaciji. CANopen standard sastoji se od sheme adresiranja, nekoliko malih komunikacijskih protokola i aplikacijskog sloja definiranog profilom uređaja. Komunikacijski protokoli imaju podršku za upravljanje mrežom, nadzor uređaja i komunikacija između čvorova, uključujući jednostavan prijenosni sloj za segmentaciju/desegmentaciju poruka. Protokol niže razine koji implementira podatkovnu vezu i fizičke slojeve obično je CAN (eng. Controller Area Network), iako uređaji koji koriste neka druga sredstva komunikacije kao npr. Ethernet Powerlink i EtherCAT također mogu implementirati CANopen profil uređaja. Komunikacija PLC-a i servo pogona izvodi se preko CANopen priključka za gibanje (eng. CANopen Motion) korištenjem protokola CANopen DS402.

4. PLC program za parsiranje G koda i upravljanja robotom

Programski kod PLC-a izrađen je u programskom softveru CANopen Builder. CANopen Builder proizvod je tvrtke Delta i softver koji je namijenjen implementiranju programskih kodova programabilnih logičkih upravljača (korišten kod mnogih serija Delta PLC-eva). Odabrali smo ovaj softver zbog njegove mogućnosti konfiguracije svakog servo pogona zasebno i mogućnost rada s njima kroz jedno sučelje. Svrha izrade programskog rješenja je kako bismo omogućili pretvaranje unaprijed definiranog programskog koda, što je u ovome radu G kod, u pomake robota. Programski kod unutar PLC-a izrađen je u ljestvičastom dijagramu (eng. Ladder djagram). Ljestvičasti dijagram se čita s lijeva na desno i odozgo prema dolje. U ovome programskom rješenju ljestvičasti dijagram je sastavljen kao sekvenca tj. svaki se dio, od samog početka (uključivanje pogona osi) do završnog dijela (parsiranje G koda), uključuje zasebno preko postavljenih ulaza i završetkom funkcije posjeduje odgovarajući izlaz koji signalizira da je funkcija uspješno provedena. Spajanje na robot direktno preko računala na PLC omogućena je pomoću mrežnog kabela.

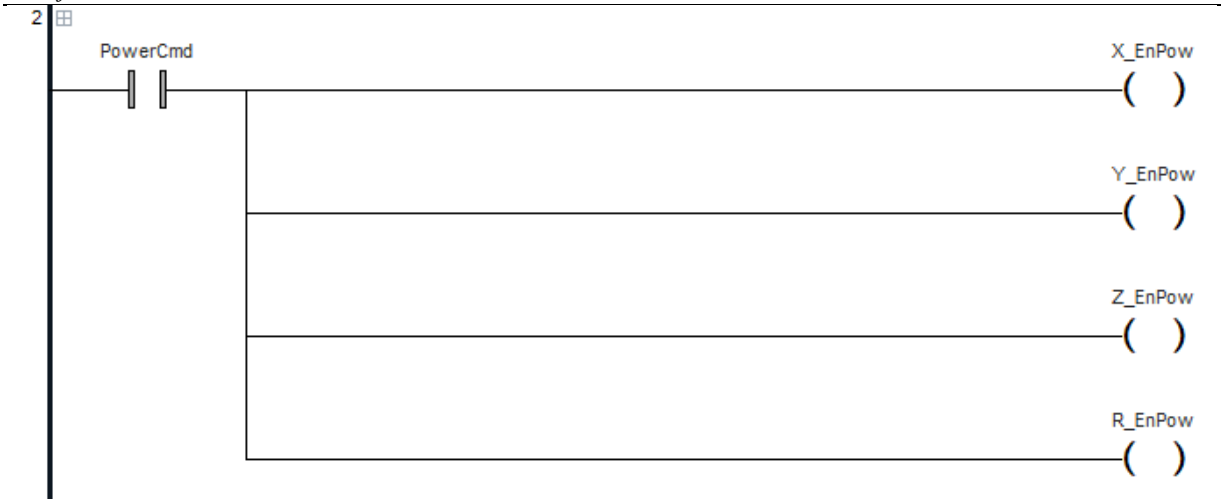
4.1. Programski kod PLC-a u Ladder dijagramu

Prethodno je navedeno da se ljestvičasti dijagram se izvršava sekvencijalno (liniju po liniju), stoga prva linija je ujedno i početak rada robota.

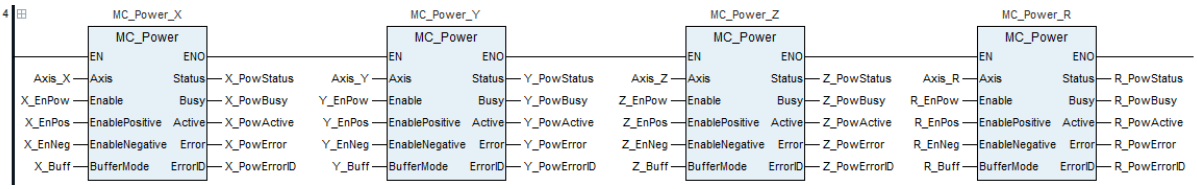
Slika 4.1. – Programski kod, Linija 1

Prekidač za hitne slučajeve, oznaka Emgcy_1 (Slika 4.1.) koji je postavljen kao obični radni kontakt, funkcionira drugačije u odnosu na normalne prekidače i ima bitniju ulogu. Dok eng. Emergency button nije pritisnut njegovi su kontakti spojeni i omogućuju propuštanje signala u Ladder dijagramu. Svrha mu je da u slučaju pritiskivanja, prekidač za hitne slučajeve prekida propuštanje signala, zaustavlja osi robota i onemogućava nastavak izvršavanja programa.

Pritiskom kontakta oznake Enable propuštamo signal na izlaz zvan PowerCmd (eng. Power on command). Uključivanjem PowerCmd dovodimo signal na ulaze pojedinih motora osi, čije oznake X(Y, Z i R)_PowEn, istovremeno (Slika 4.2.). Ti ulazi pokreću statusne blokove MC_Power (Slika 4.3.) koji pokreću servo pogone pojedinih osi. Na ulazu blokova MC_Power moramo definirati osi kojima upravljaju, ulazni signali Axis_X (Y, Z i R).

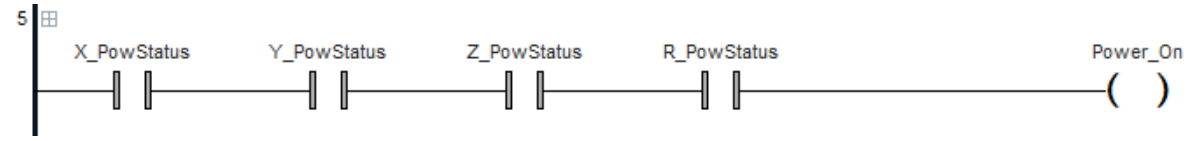


Slika 4.2. – Programski kod, Linija 2



Slika 4.3. – Programski kod, Linija 4 – MC_Power blokovi

Nakon što postavimo osi, moramo ulaze blokova oznake EnablePositive i EnableNegative postaviti na TRUE. Kada se svaka os upali na svom izlazu daje signal na X(Y,Z i R)_PowStatus. Kada se sve osi uključe i aktiviraju signali na PowStatus aktivira se izlaz Power_On (Slika4.4.).

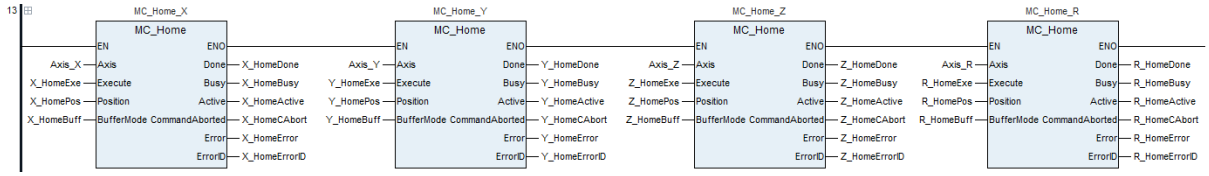


Slika 4.4. – Programski kod, Linija 5

Nakon inicijalnog paljenja robota i njegovih osi, potrebno je postaviti osi u početni položaj eng. Homing. Kako bismo izveli operaciju Homing koristimo statusne blokove MC_Home (Slika 4.6.). Pomoću kontakta Homing_Exe (Slika 4.5.) pokrećemo operaciju vraćanja u početni položaj i to na način da se blokovi uključuju sekvencijalno tj. izlaz jednog bloka npr. X_HomeDone propušta signal na ulaz sljedećeg bloka, koji je u ovom primjeru Y_HomeExe i taj se postupak ponavlja do R_HomeDone.

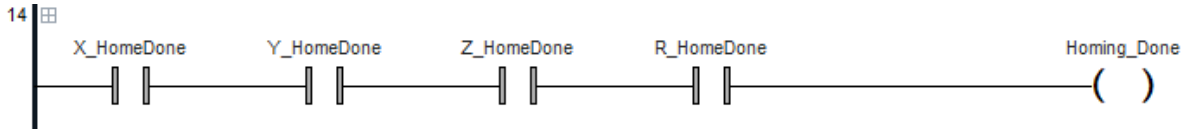
Slika 4.5. – Programski kod, Linija 6

Na ulazima blokova kao i u prethodnome primjeru, postavljamo os koju MC_Home postavlja u početni položaj s signalom na Axis_X(Y, Z i R), uključujemo blokove s signalom na HomeExe i signalom na X(Y, Z i R)_HomePos postavljamo poziciju na koju MC_Home blok postavlja osi u početnom položaju.



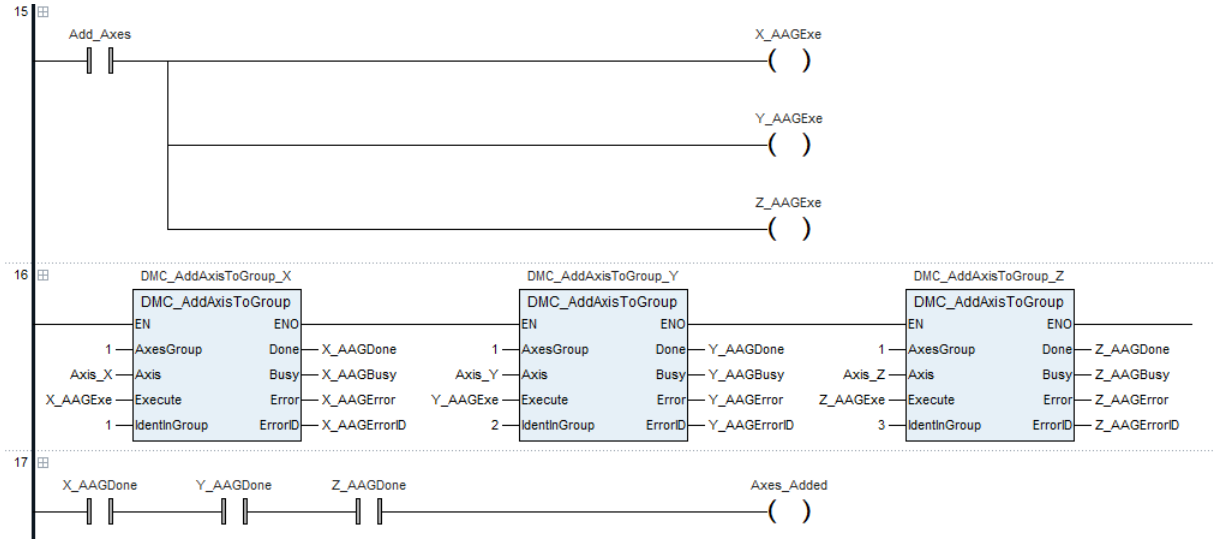
Slika 4.6. – Programski kod, Linija 13 – MC_Home blokovi

Nakon što su izlazi bloka X(Y, Z i R)_HomeDone uključeni, propušta se signal na zadani izlaz Homing_Done koji pokazuje da je robot uspješno postavio osi u njihov početni položaj.



Slika 4.7. – Programski kod, Linija 14

Nadalje potrebno je grupirati osi u jednu eng. AxesGroup. Robot će prema unaprijed definiranom G kodu izvoditi putanju gdje je potrebno više osi koje istovremeno mijenjaju položaj u odnosu na njihovu Home poziciju. Uz to funkcijski blokovi koji služe za postavljanje parametara, interpolaciju putanje i parsiranje G koda koriste zajedničku grupu osi. Grupiranje osi vršimo pomoću statusnog bloka DMC_AddAxisToGroup (Slika 4.8.).

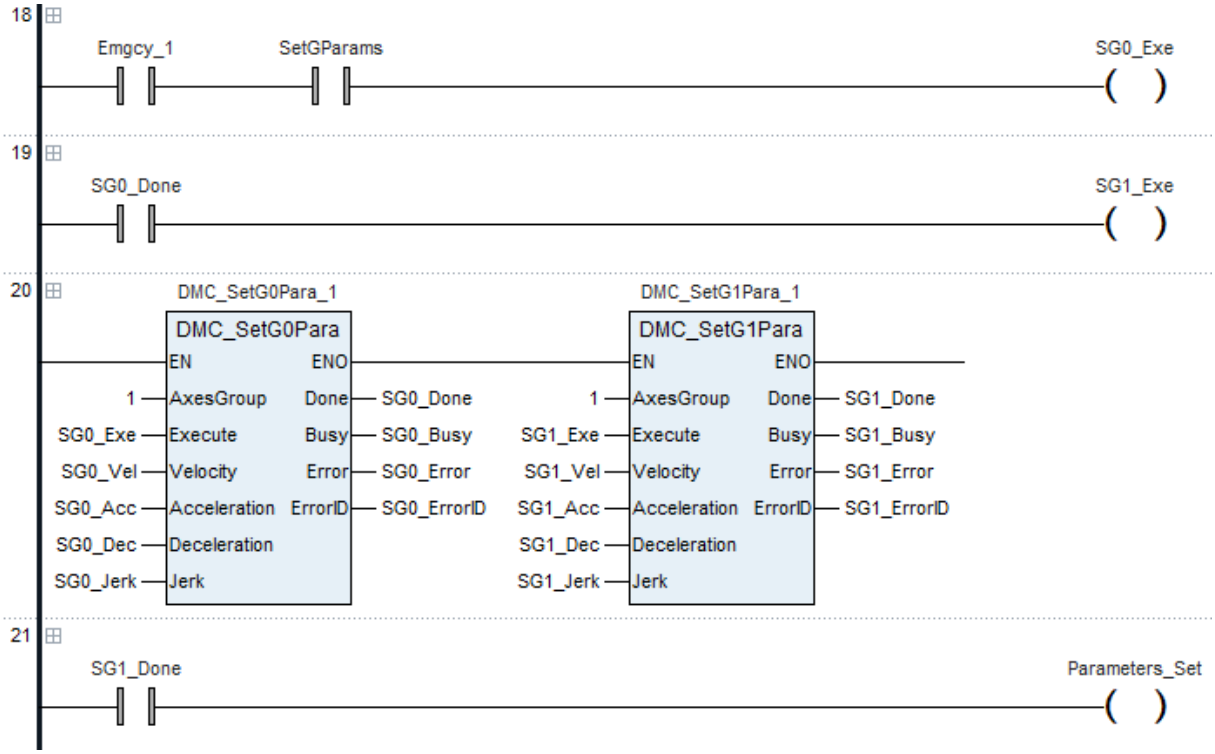


Slika 4.8. – Progamski kod, Linija 15, 16 i 17

Na ulazu bloka definiramo naziv zajedničke grupe osi, ovdje imenovana brojem 1. Kao i kod prethodnih blokova, definiramo pojedinu os koju će pojedini blok svrstati u zajedničku grupu. Pritiskom kontakta Add_Axes propuštamo signal do X(Y i Z)_AAGExe koji pokreće kreiranje

zajedničke grupe osi. Kada je izvršeno kreiranje grupe osi signal se dovodi do izlaza bloka X(Y i Z)_AAGDone koji potom uključuju izlaz Axes_Added.

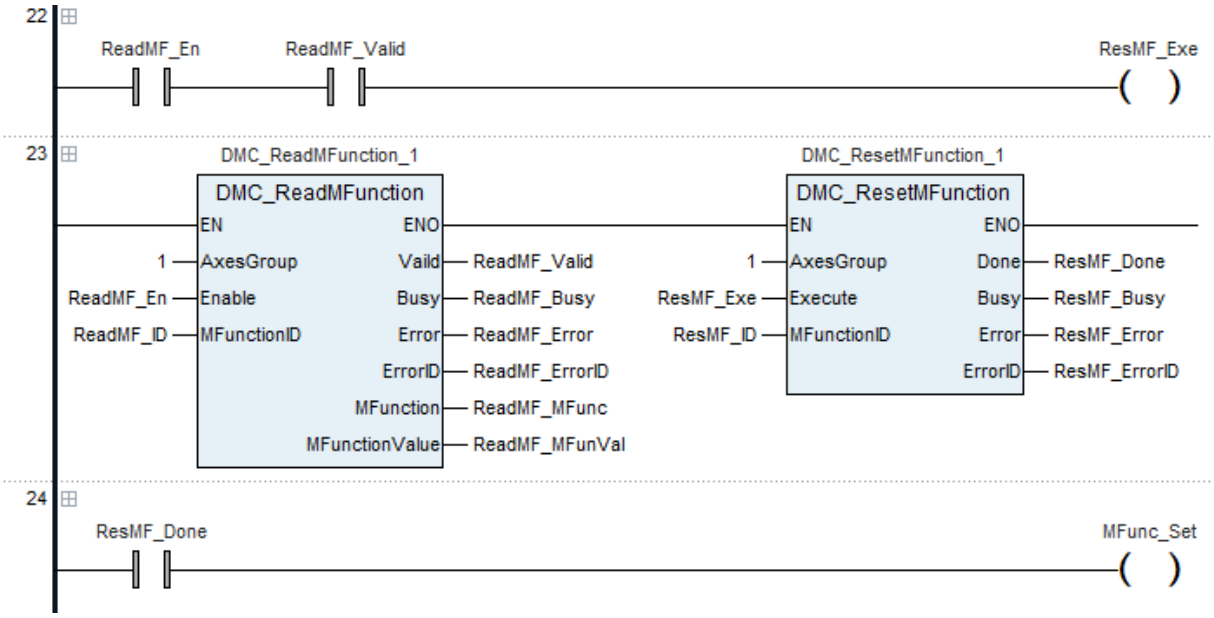
Nakon uključivanja robota, paljenja pogona osi, postavljanja početnog položaja i kreiranja zajedničke grupe osi, sljedeći korak je postaviti parametre gibanja prema kojima će se osi gibati. Parametre postavljamo s statusnim blokovima DMC_SetG0Para i DMC_SetG1Para (Slika 4.9.), koji služe za postavljanje parametara brzine (eng. Velocity), ubrzanja (eng. Acceleration), usporavanja (eng. Deceleration) i trzaja (eng. Jerk) za naredbe G koda: G00, G01, G02 i G03. Te naredbe upravljaju gibanjem stroja. U naknadnom potpoglavlju pojasniti ćemo značenje tih naredbi.



Slika 4.9. – Programski kod, Linije 18, 19, 20 i 21

Pritiskom kontakta SetGParams Propuštamo signal na SG0_Exe koji uključuje prvi blok DMC_SetG0Para_1 koji prethodno navedene parametre postavlja za naredbu G00. Treba uočiti da umjesto pojedinog bloka za pojedinu os, sada imamo jedan blok za jednu zajedničku grupu osi koju smo prethodno kreirali. Izvršavanjem bloka, izlaz SG0_Done će propustiti signal na ulaz bloka DMC_SetG1Para_1 pod nazivom SG1_Exe koji pokreće postavljanje parametara za naredbe G01, G02 i G03. Nakon što su parametri blokova definirani uključuje se izlaz Parameters_Set.

U uvodu rada pojasnili smo što je to G kod i od kakvih se naredbi sastoji, te također spomenuli kako PLC s kojim upravljamo robotom u ovome radu podržava G kod. Uz G funkcije, često se u praksi koriste i brojne M funkcije koda. Kako bi program pri izvođenju putanje zadanom G kodom učitao M funkciju koja se nalazi u njemu potrebni su nam blokovi DMC_ReadMFunction i DMC_ResetMFunction (Slika 4.10.).



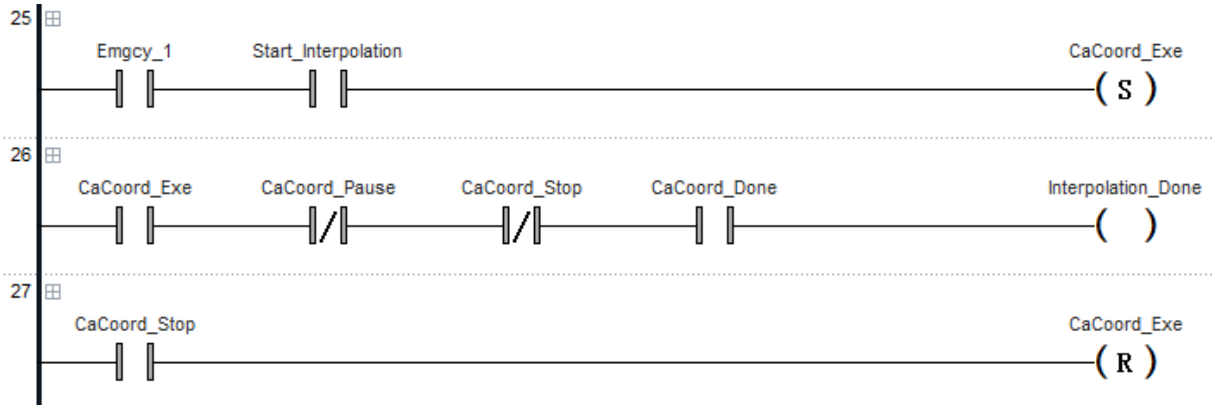
Slika 4.10. – Programski kod, Linije 22, 23 i 24

Kontaktom ReadMF_En uključujemo blok DMC_ReadMFunction_1 čija je svrha učitavanje M funkcije koja bi se nalazila unutar G koda. Nakon što se funkcija učita uključuje se izlaz ReadMF_Valid i izlaz ReadMF_MFunVal koji prikazuje učitanu funkciju. Prema slici(4.10.) uočljivo je da se uključivanjem ReadMF_Valid potom uključuje ResMF_Exe. Taj signal na ulazu pokreće blok DMC_ResetMFunction_1 koji resetira vrijednost M funkcije, te se nastavlja izvršavanje G koda. S uključivanjem izlaza ResMF_Done, uključuje se i izlaz MFunc_Set. Ako imamo više M funkcija u kodu, potrebno je za svaku dodati po još jedan par blokova.

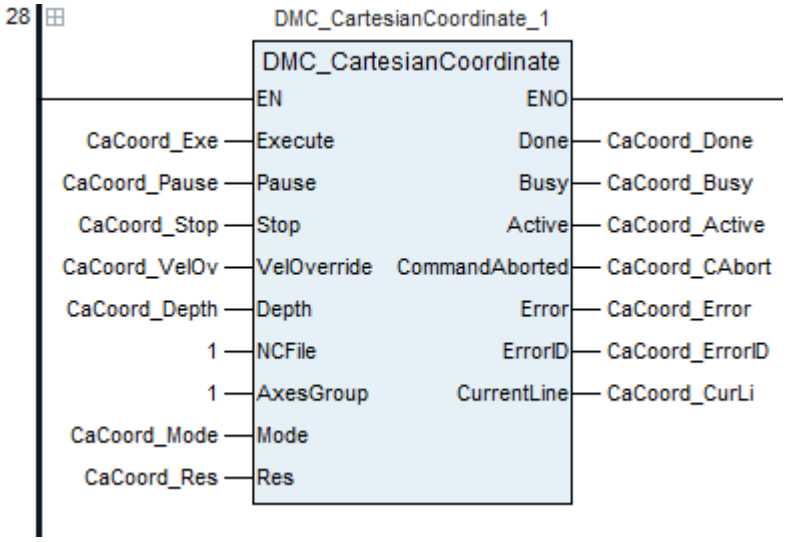
Nakon postavljanja svih bitnih funkcija slijedi najvažniji dio rada koji je upravljanje robota prema putanji zadanoj G kodom. Tu funkciju obavlja blok DMC_CartesianCoordinate (Slika 4.11.) koji vrši interpolaciju prema putanji zadanoj u G kodu i upravlja s kartezijskim koordinatnim robotom.

Pritiskom kontakta Start_Interpolation (Slika 4.12.) propuštamo signal na CaCoord_Exe koji pokreće blok. Blok prolazi kroz G kod i pomiče robota prema putanji zadanoj u G kodu. Izlaz bloka pod nazivom CaCoord_CurLi prikazuje koju liniju G koda robot trenutno izvršava. Ukoliko želimo privremeno zaustaviti izvršavanje pritisnemo kontakt CaCoord_Pause, a želimo li prekinuti izvršavanje koda pritisnemo kontakt CaCoord_Stop koji prekida izvršavanje

G koda i kao što je prikazano na liniji 27 na slici 4.12. resetira ulaz bloka CaCoord_Exe na FALSE.



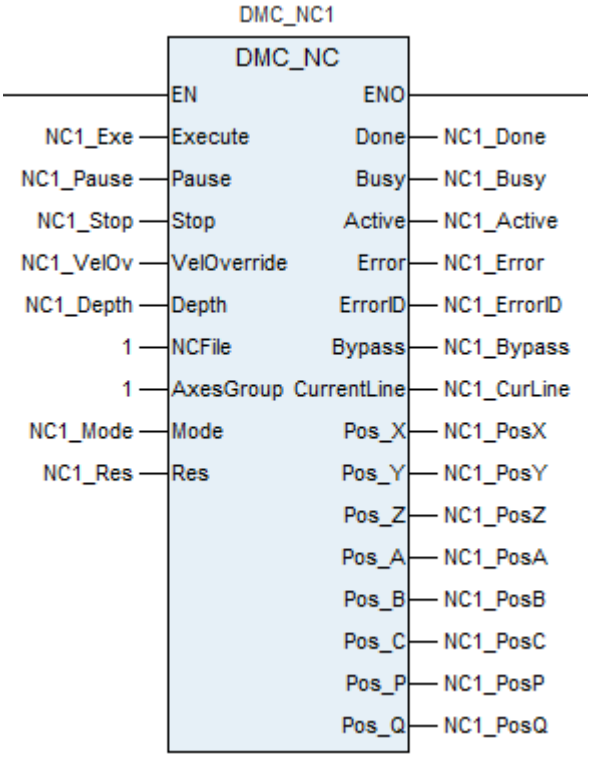
Slika 4.12. – Programski kod, Linije 25, 26 i 27



Slika 4.11 – Programski kod, Linija 28 – blok DMC_CartesianCoordinate

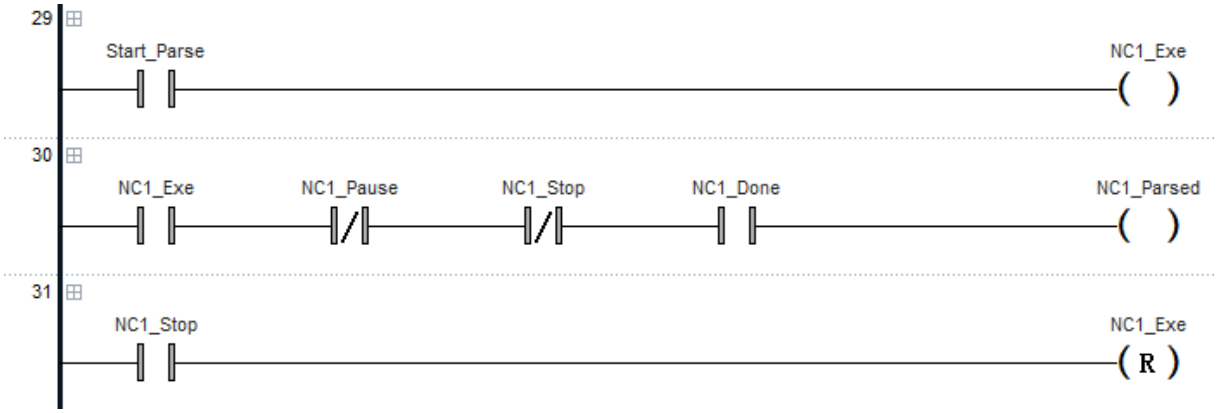
Nakon što blok završi s provedbom i kada robot izvrši zadanu putanju uključuje se CaCoord_Done koji propušta signal do izlaza Interpolation_Done.

Nakon interpolacije potrebno je još parsirati G kod. Tu funkciju izvršava blok DMC_NC.



Slika 4.13. – Programski kod, Linija 28 – blok DMC_NC

Blok DMC_NC (Slika 4.13.) analizira položaje pojedinih osi i na izlazu prikazuje njihove položaju pri prolasku po linijama G koda. Prikazane pozicije i promjena njihovih vrijednosti je apsolutna tj. vrijednosti prikazane na izlazima, u ovome radu su to NC1_PosX(Y i Z), je položaj robota gledano od njegovog početnog položaja (eng. Home). Ti izlazi ne upravljaju gibanjem osi, samo prikazuju položaj osi po liniji G koda koja se parsira.



Slika 4.14. – Programski kod, Linije 29, 30 i 31

Parsiranje započinjemo pritiskom na kontakt Start_Parse (Slika 4.14.) koji će propustiti signal na ulaz NC1_Exe i time pokrenuti blok DMC_NC1. Izlaz bloka NC1_CurLine prikazuje redni broj linije koja se parsira, te na izlazima NC1_PosX(Y i Z) vidimo položaj osi po trenutnoj

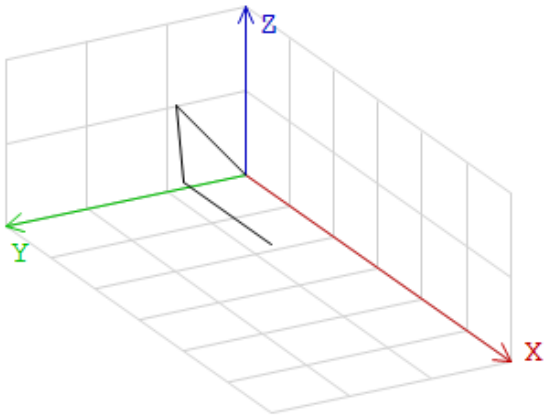
putanji. Slično kao kod prethodnog bloka, DMC_CartesianCoordinate, možemo privremeno zaustaviti parsiranje koda pritiskom na kontakt NC1_Pause, a potpuno prekinuti taj proces s kontaktom NC1_Stop koji će potom resetirati ulaz NC1_Exe u stanje FALSE. Kada je funkcija uspješno obavljena uključuje se izlaz NC1_Done, NC1_PosX(Y i Z) prikazuju krajnji položaj osi i propušta se signal i uključuje se konačni izlaz NC1_Parsed. (Slika 4.14.)

4.2. Softverski podržane naredbe G koda

Spomenuli smo u prethodnim poglavljima što je to G kod, od kakvih se naredbi sastoji i koja im je funkcija. PLC koji koristimo podržava čitanje i izvršavanje G koda. No softver CANopen Builder posjeduje CNC urednik gdje se uređuje program, te je također moguće ubaciti G kod iz drugih softvera u taj urednik. Prateći pravila iz poglavlja „UVOD“ vezana za pisanje G koda možemo sami osmisliti primjer putanje ili možemo ubaciti gotovi kod unutra. Još jedna značajka u CANopen Builder-ovom CNC uredniku je ta da za napisani G kod u prozoru ispod pojaviti će se trodimenzionalni prikaz putanje stroja. Na slikama ispod (Slika(4.15.) i Slika(4.16.)) imati ćemo prikazan primjer G koda i izgled generirane putanje.

```
1   N0000 G0 X100 Y100 Z100
2   N0001 G1 X200 Y150
3   N0002 G1 X300 Y150
```

Slika 4.16 – Primjer G koda



Slika 4.15 – Prikaz putanje

Generirani G kod potrebno je pozvati unutar programa upravljača što izvodimo pomoću blokova poput DMC_CartesianCoordinate, koji upravlja robotom i interpolira putanju prema G kodu, te s DMC_NC koji analizira poziciju pojedine osi po liniji G koda.

U sljedećoj tablici prikazane su naredbe G koda podržane u CANopen Builderu. Napomena, potpuni popis podržanih M funkcija nalazi se u prilogu na kraju rada.

G kod	Funkcija
G00	Brzi hod alata
G01	Linearna interpolacija
G02	Kružni hod alata u smjeru kazaljke na satu
G03	Kružni hod alata u smjeru suprotno kazaljke na satu
G04	Zaustavljanje
G17	XY radna ravnina
G18	XZ radna ravnina
G19	YZ radna ravnina
G50	Isključivanje skaliranja
G51	Uključivanje skaliranja
G52	Postavljanje radnog koordinatnog sustava
G90	Apsolutni koordinatni sustav
G91	Inkrementalni koordinatni sustav
M0-M99	M funkcije

Tablica 4.1 – Podržane funkcije G koda

5. Upravljanje robotom i primjer G koda

U ovom poglavlju pokazati ćemo upravljanje robotom i parsiranje G koda pomoću programskog rješenja prikazanog unutar poglavlja 4 (puni izgled programskog koda nalazi se u prilogu), te uz primjer G koda kojim smo kreirali putanju koju robot prati.

Kao što je u prethodnim poglavljima spomenuto i prikazano, programsko rješenje je napisano u obliku ljestvičastog dijagrama i s njime možemo direktno upravljati s pojedinim osima. Napisani kod stavljamo na PLC, koji je povezan s računalom putem ethernet kabela, pa s njime možemo upravljati robotom. Unutar softvera CANopen Builder postavili smo zasebne ulaze/izlaze za pokretanje/zaustavljanje programa tj. kao signali da je određena funkcija, npr. Homing, uspješno obavljena. Razlog tomu je zato što nemamo fizičke tipke pomoću kojih bi pokrenuli program niti fizičke indikatore da je dio programa uspješno izvršen. Kako bismo olakšali praćenje na sljedećim slikama (izuzev Slike 5.1.) određeni kontakti zaokruženi su obojanim elipsama. Crvena elipsa označava ulaz kojim pokrećemo određenu operaciju, dok ljubičasta elipsa označava izlaz koji pokazuje da je pokrenuta operacija izvršena.

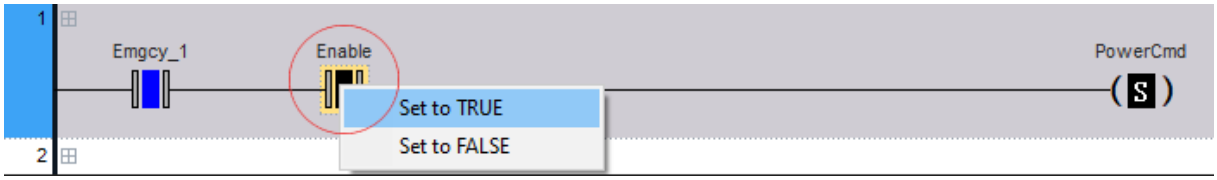
Korisnik preko računala unutar CANopen Buildera može pokrenuti i/ili promijeniti sljedeće funkcije/parametre:

- Uključivanje/isključivanje pogona osi.
- Određivanje početnog položaja osi.
- Inicijalizacija postavljanja/vraćanja osi u početni položaj.
- Stvaranje grupe osi.
- Postavljanje parametara: brzine (eng. Velocity), ubrzanja (eng. Acceleration), usporavanja (eng. Deceleration) i trzaja (eng. Jerk) za naredbe G00, G01, G02, G03.
- Učitavanje M funkcija.
- Pokretanje/privremeno zaustavljanje/prekid interpolacije putanje G koda.
- Pokretanje/privremeno zaustavljanje/prekid parsiranja G koda.

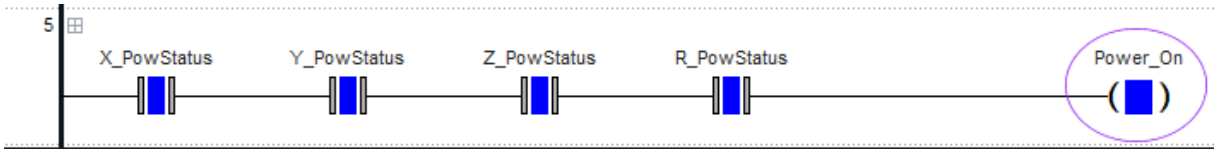
Kako bi korisnik pokrenuo program, mora ga prvo učitati na PLC (tipka Download, crveni krug na slici ispod). Nakon uspješnog preuzimanja programa, potrebno je program prebaciti u online mode (zeleni krug na slici ispod) i pokrenuti PLC (ljubičasti krug na slici ispod).

Slika 5.1. – Učitavanje programa i pokretanje PLC-a

Kada je sve uključeno, korisnik pokreće osi pritiskom na tipku Enable (stanje prelazi u TRUE), uključuju se sva četiri servo pogona istovremeno što je prikazano na zajedničkom izlazu Power_On. Također ulazi blokova EnablePositive i EnableNegative moraju biti TRUE.

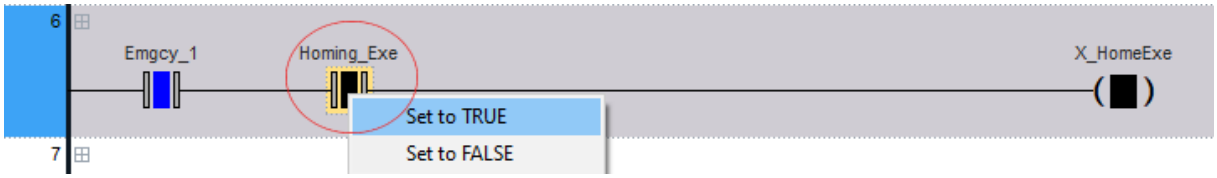


Slika 5.2. – Pokretanje servo pogona

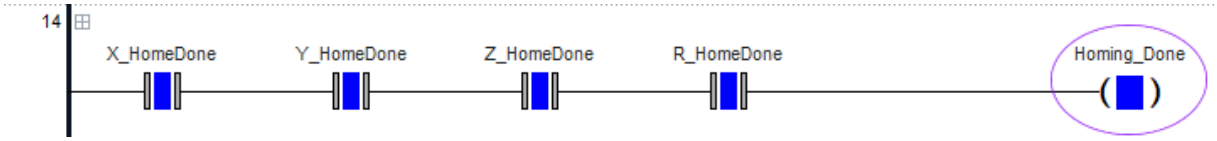


Slika 5.3. – Servo pogoni su uspješno uključeni

Nakon što je uključio servo pogone osi korisnik mora postaviti osima početni položaj. Na jednom od ulaza bloka MC_Home, pod imenom X(Y, Z i R)Home_Pos, mora unijeti poziciju koju smatra početnim položajem osi, u ovome radu to je (0, 0, 0, 0).



Slika 5.4. – Pokretanje operacije Homing

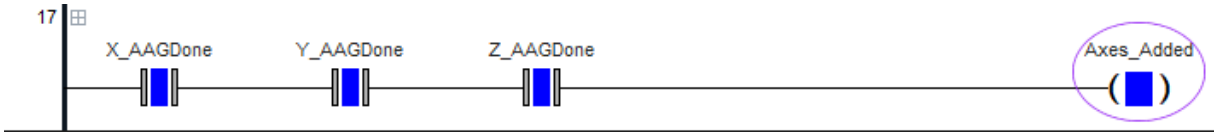


Slika 5.5. – Operacija Homing izvršena

Nakon što je unio poziciju, pritiskom postavljenog ulaza Home_Exe pokreće se Homing osi, pri čemu robot pomiče sve usi u zadani početni položaj i tamo se zaustavlja, pri čemu se uključuje izlaz Homing_Done.

Sljedeća operacija koju korisnik mora pokrenuti je kreiranje grupe osi. Prije kreiranja grupe korisnik mora unijeti osi koje želi dodati u grupu, naznačiti koji je redni broj osi i naziv grupe koja će se koristiti u daljnjim blokovima.

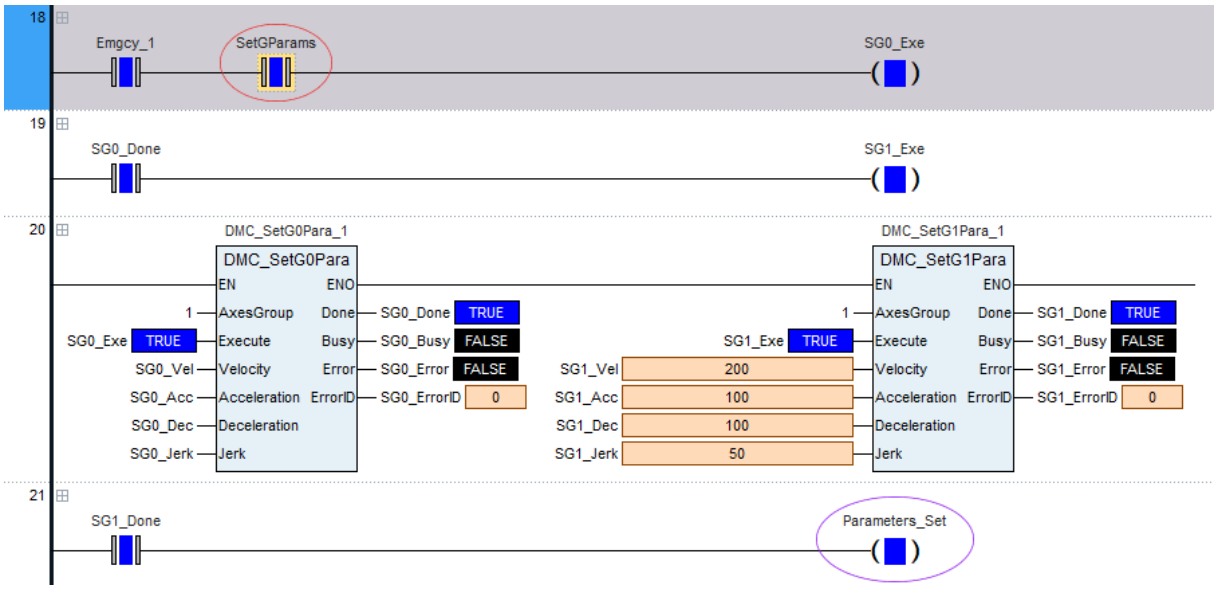
Slika 5.6. – Pokretanje operacije kreiranja grupe osi



Slika 5.7. – Operacija kreiranja grupe osi izvršena

Pritiskom na kontakt Add_Axes pokreću se blokovi MC_AddAxisToGroup pri čemu se kreira grupa i signal dolazi na zajednički izlaz Axes_Added. Ovaj korak se ne smije preskočiti jer za pokretanje sljedećih blokova potrebna je grupa osi.

Korisnik postavlja parametre koji opisuju kinematiku gibanja osi; brzina, ubrzanje, usporavanje i trzaj pomoću blokova DMC_SetG0Para i DMC_SetG1Para.



Slika 5.8. – Postavljanje parametara za gibanje osi

Vrijednosti tih parametara na oba bloka upisuje na ulaze Velocity, Acceleration, Deceleration i Jerk. U tablici 5.1. napisane su vrijednosti istih parametara za oba bloka. Pritiskom na kontakt SetGParams pokrećemo blokove čime se te vrijednosti postavljaju za naredbe G koda G00, G01, G02 i G03. kada završi postavljanje na oba bloka uključuje se izlaz Parameters_Set.

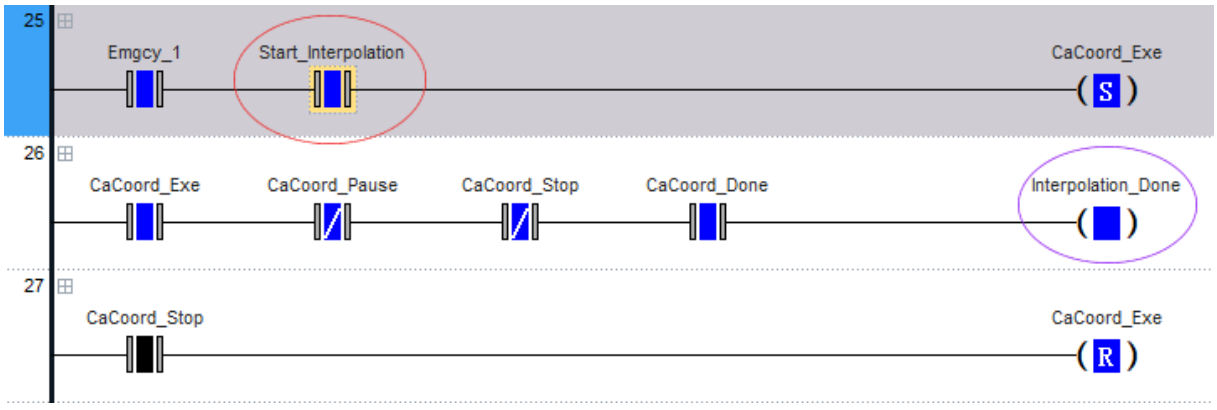
Parametar	Veličina
SG0_Vel	200 mm/s
SG0_Acc	100 mm/s^2
SG0_Dec	100 mm/s^2
SG0_Jerk	50 mm/s^3
SG1_Vel	200 mm/s
SG1_Acc	100 mm/s^2
SG1_Dec	100 mm/s^2
SG1_Jerk	50 mm/s^3

Tablica 5.1. – Vrijednosti postavljenih parametara

Prateći tablicu 4.1. iz poglavlja 4.2. vidimo da CANopen Builder podržava M funkcije od M00 do M99, stoga kako bi program mogao te naredbe provesti koristimo dva bloka u paru. To su blokovi DMC_ReadMFunction i DMC_ResetMFunction.

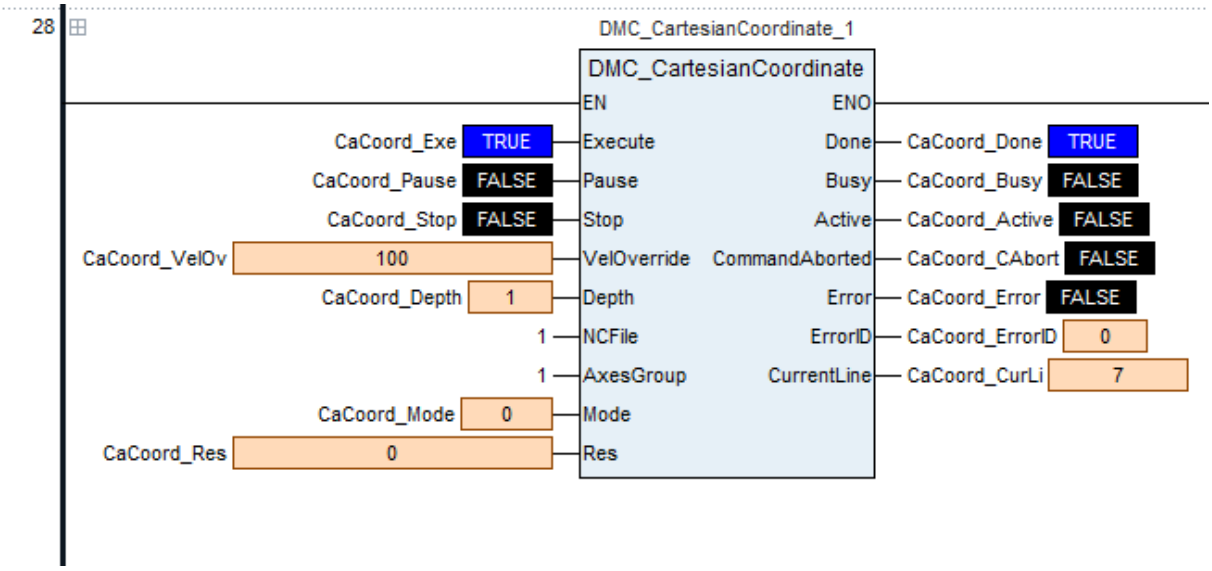
Korisnik mora unijeti koja je M naredba koju navodi u kodu na ulaz MFunctionID oba bloka. Recimo da unutar G koda upiše M naredbu M02 koja označava kraj programa. Ona se zapisuje u zadnjoj liniji koda, te ju korisnik mora na ulazu MFunctionID upisati. Kada provedba G koda dođe to te linije korisnik mora kontaktom ReadMF_En pokrenuti prvi blok koji učitava M02 naredbu. Zatim se uključuje kontakt ResMF_Exe koji resetira vrijednost M funkcije. Zatim se uključuje izlaz MFunc_Set. Ukoliko kod nema M funkciju, kao što je slučaj u ovome radu, ovaj se korak može preskočiti jer ne utječe na ostatak programa.

Kada je korisnik postavio sve potrebne parametre, ostaje mu izvršavanje G koda. Na bloku DMC_CartesianCoordinate unosi ID broj NC datoteke (što je u ovome radu broj 1), grupu osi, zatim pokreće interpolaciju putanje pritiskom na kontakt Start_Interpolation.



Slika 5.9. – Pokretanje interpolacije putanje i signal uspješno izvršene operacije

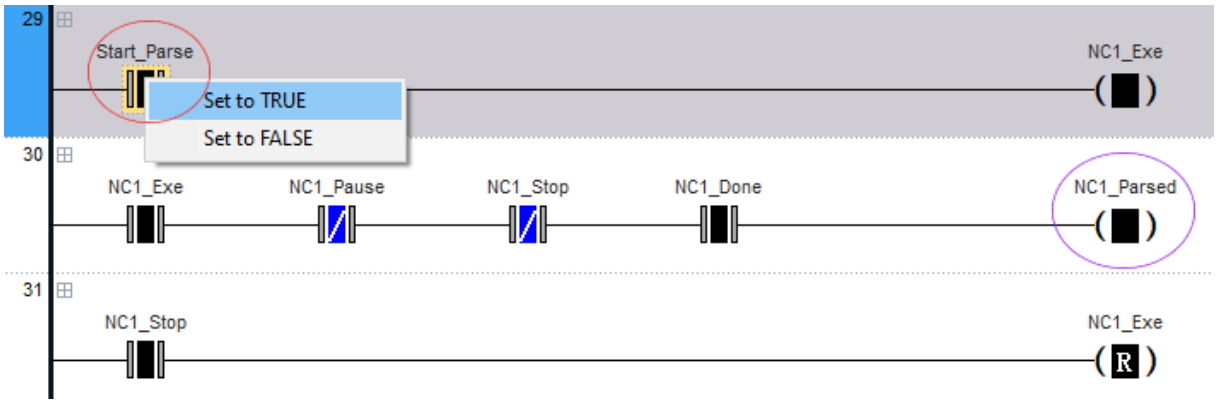
Blok će izvršavati kod liniju po liniju što će biti prikazano na njegovom izlazu, ovdje imenovan CaCoord_CurLi kao i u samim kretnjama robota. Ukoliko je korisnik učinio grešku, kao npr. postavio je krivi alat, može privremeno zaustaviti izvršavanje interpolacije pritiskom na CaCoord_Pause, pri čemu se robot zaustavlja i korisnik može ispraviti pogrešku i nastaviti s radom robota. No ako poželi potpuni prekid izvršavanja, pritisne CaCoord_Stop, pri čemu se blok resetira i potrebno je zaustaviti PLC i ponovno ga pokrenuti. Nakon ponovnog pokretanja PLC-a korisnik mora pokrenuti servo pogone osi i vratiti osi u zadani početni položaj. Kada robot uspješno izvrši putanju uključuje se izlaz Interpolation_Done.



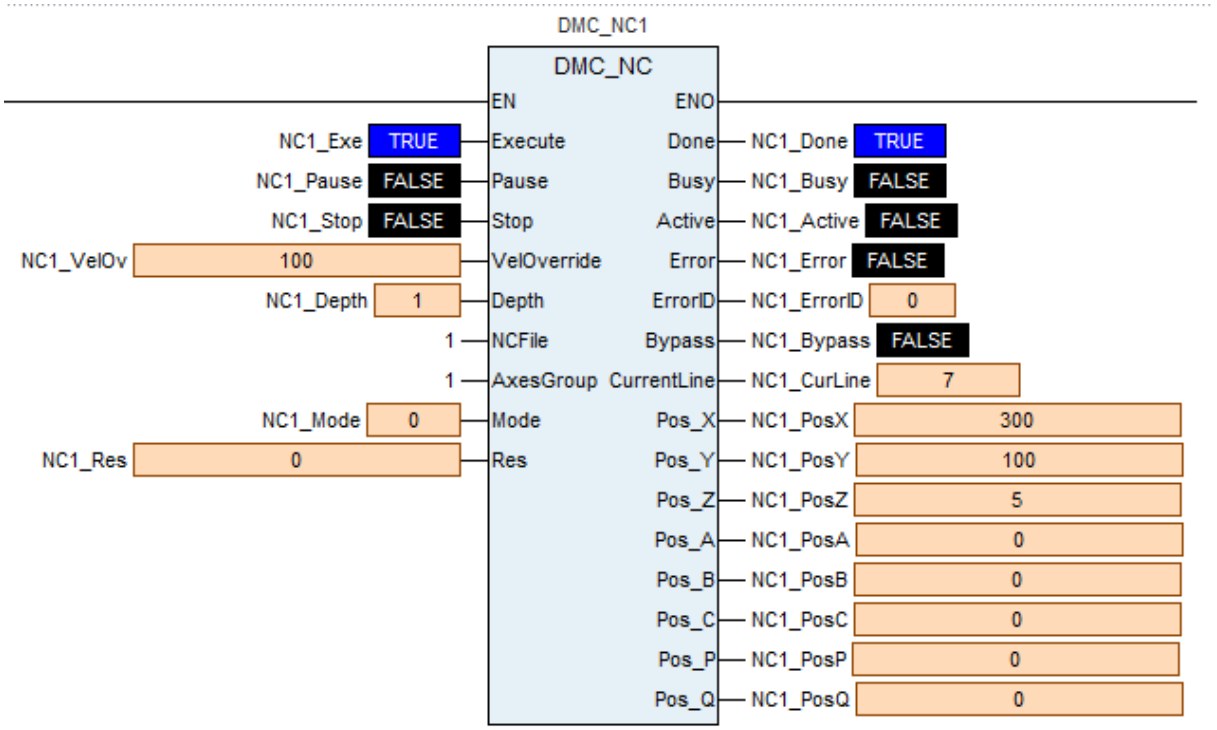
Slika 5.10. – Blok DMC_CartesianCoordinate nakon izvršavanja interpolacije

Završetkom interpolacije korisnik uključuje blok DMC_NC1. Na bloku također unosi grupu osi i ID broj NC datoteke. Pritiskom na kontakt Start_Parse pokrenuti će se operacija parsiranja tj. prevoditi će linije G koda i prikazati ih kao pomake pojedine osi na izlazima NC1_PosX(Y i Z).

Kada je operacija izvršena uključuje se izlaz NC1_Parsed.



Slika 5.11. – Pokretanje parsiranja G koda

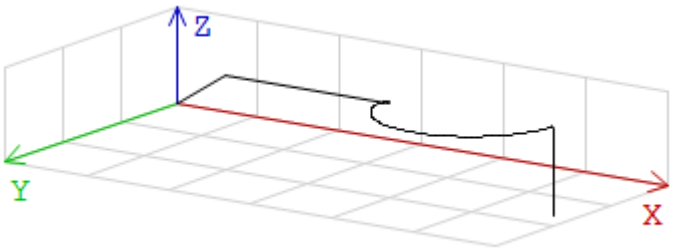


Slika 5.12. – Blok DMC_NC1 nakon izvršene operacije parsiranja

Rezultat cijelog programa je pomicanje robota po putanji zadanoj G kodom s parametrima koji upravljaju kinematikom gibanja osi. Na sljedećim slikama prikazani su primjer G koda korišten unutar rada za proces i kreirana putanju robota.

```
1  N0000 G91
2  N0001 G0 X0 Y0 Z0
3  N0002 G1 X100 Y100 Z50
4  N0003 G1 X100
5  N0004 G2 X100 R50
6  N0005 G1 Z-45
```

Slika 5.13. – Primjer korištenog G koda



Slika 5.14. – Prikaz putanje generirane unutar CNC urednika

6. ZAKLJUČAK

Cilj ovog rada bio je izraditi programsko rješenje kako bismo omogućili da kartezijski koordinatni robot izvrši radnu operaciju obrade prateći putanju zadanu G kodom bez direktnog upravljanja korisnika. Kako bismo to uspješno izveli morali smo se upoznati s svrhom zbog koje je rad napravljen, koja je dio veće slike, da se stariji robot ponovno upotrijebi u industriji i dovede do standarda Industrije 4.0. Morali smo se upoznati s opremom kojom ćemo taj zadatak obaviti, to su već prethodno spomenuti kartezijski koordinatni robot, PLC DVP15MC11T-06 i programsko sučelje CANopen Builder u kojemu je programsko rješenje napravljeno.

Koristeći se metodom Ladder dijagrama složili smo program, koristeći blokove s funkcijama potrebnim za izvršavanje tog zadatka. Za pokretanje servo pogona pojedinih osi koristili smo blok MC_Power, te koristeći se kontaktom Enable omogućili njihovo istovremeno uključivanje. S blokom MC_Home za pojedinu os postavili smo poziciju koja je početni položaj robota prije izvođenja bilo kakvog gibanja. Za korištenje blokova koji se odnose na G kod, morali smo kreirati grupu osi. To smo učinili koristeći blok DMC_AddAxisToGroup.

Nakon kreiranja grupe, odlučili smo postaviti parametre gibanja osi: brzina, ubrzanje, usporavanje i trzaj. Postavili smo ih putem blokova DMC_SetG0Para i DMC_SetG1Para. Njihova je svrha te navedene parametre postaviti za naredbe unutar G koda što upravljaju gibanjem: G0, G1, G2 i G3. Kako bismo omogućili korištenje M funkcija G koda, koristili smo DMC_ReadMFunction i DMC_ResetMFunction blokove koji rade u paru. Prvi učitava korištenu M funkciju prilikom izvršavanja G koda, zatim se pomoću drugog bloka ta funkcija resetira i izvršavanje G koda se nastavlja. Blok čija je funkcija upravljanje robotom je DMC_CartesianCoordinate koji vrši interpolaciju prema G kodu, te s izvršavanjem pojedine linije koda robot se giba po generiranoj putanji. Na kraju potrebno je bilo taj kod parsirati. Tu je funkciju obavio blok DMC_NC koji, čitajući G kod, na svom izlazu prikazao promjenu pozicija osi prema toku izvršavanja koda. Završetkom parsiranja završava i program.

Program je uspješno obavio zadatak. Rješenje je moguće oblikovati na više načina koristeći postojeće blokove i donijeti poboljšanja programu kako bi izvršio zadatak učinkovitije.

LITERATURA

[1] Krešimir Major – Rekonstrukcija robota kartezijske strukture, Diplomski rad 2022.

[2] Goran Malčić – Programljivi logički kontroleri, Tehničko Veleučilište Zagreb

[3] Nedeljko Matejak – Industrija 4.0, Diplomski rad 2017.

[4] Delta Electronics Inc. – DVP-15MC Series Motion Controller Operation Manual 2020.

[5] https://www.wwdmag.com/home/article/21016258/what-is-a-programmable-logic-controller-p (14.1.2023.)

[6] https://www.ibm.com/topics/industry-4-0 (15.1.2023.)

[7] https://www.linearmotiontips.com/what-is-a-cartesian-robot/ (17.1.2023.)

[8] https://ammc.com/the-benefits-and-applications-of-cartesian-robots/ (17.1.2023.)

[9] https://www.steckermachine.com/blog/g-code-and-m-code-programming (18.1.2023.)

[10] https://howtomechatronics.com/tutorials/g-code-explained-list-of-most-important-g-code-commands/ (18.1.2023.)

[11] https://blog.tormach.com/what-is-cad-cam-gcode (18.1.2023.)

[12] https://www.starrapid.com/blog/what-is-g-code-and-why-is-it-important-for-your-parts/ (18.1.2023.)

[13] https://cnc.com.hr/g-kod/ (31.1.2023.)

[14] https://www.academia.edu/11379212/Tablica_G_Funkcija (31.1.2023.)

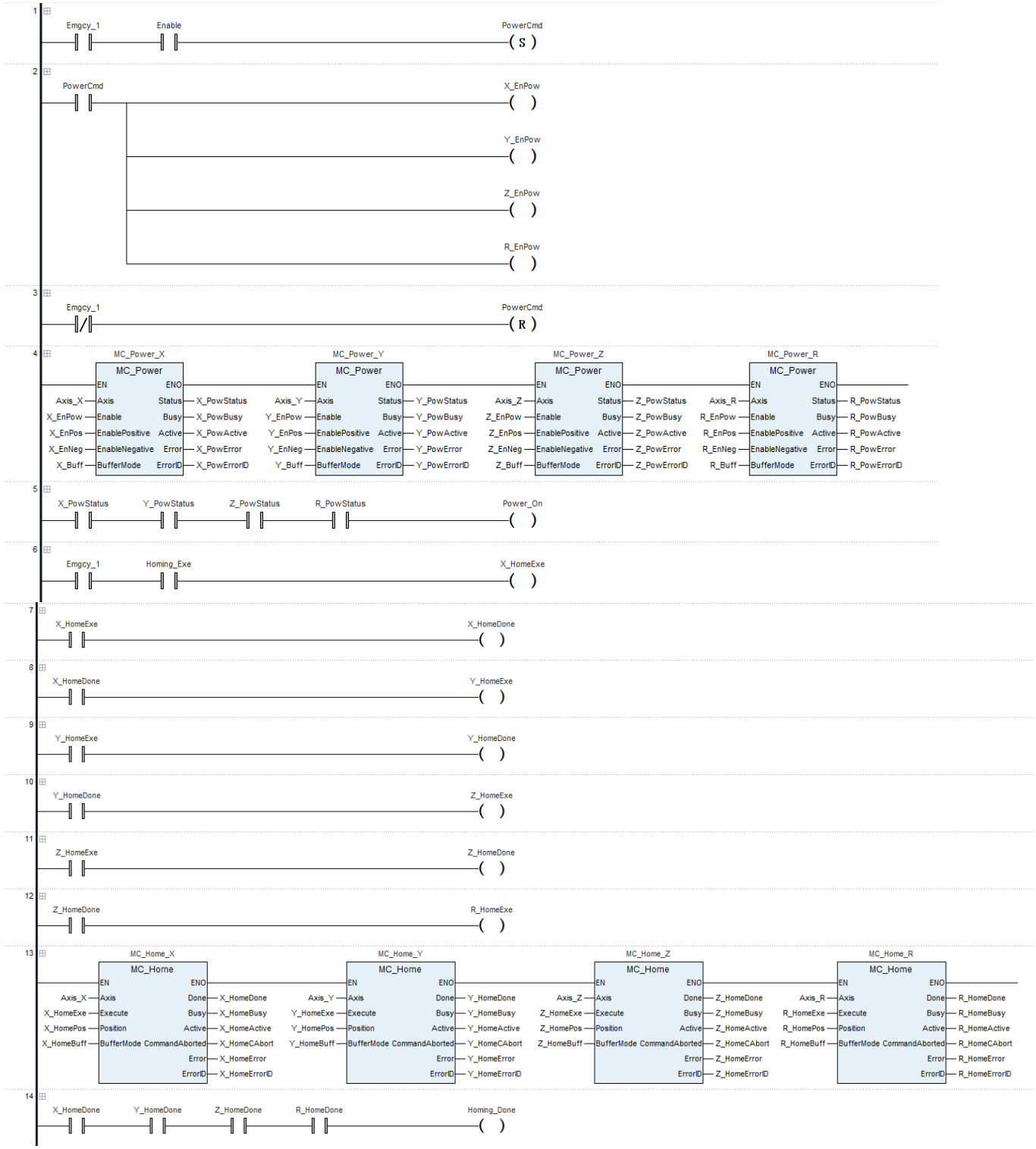
PRILOZI

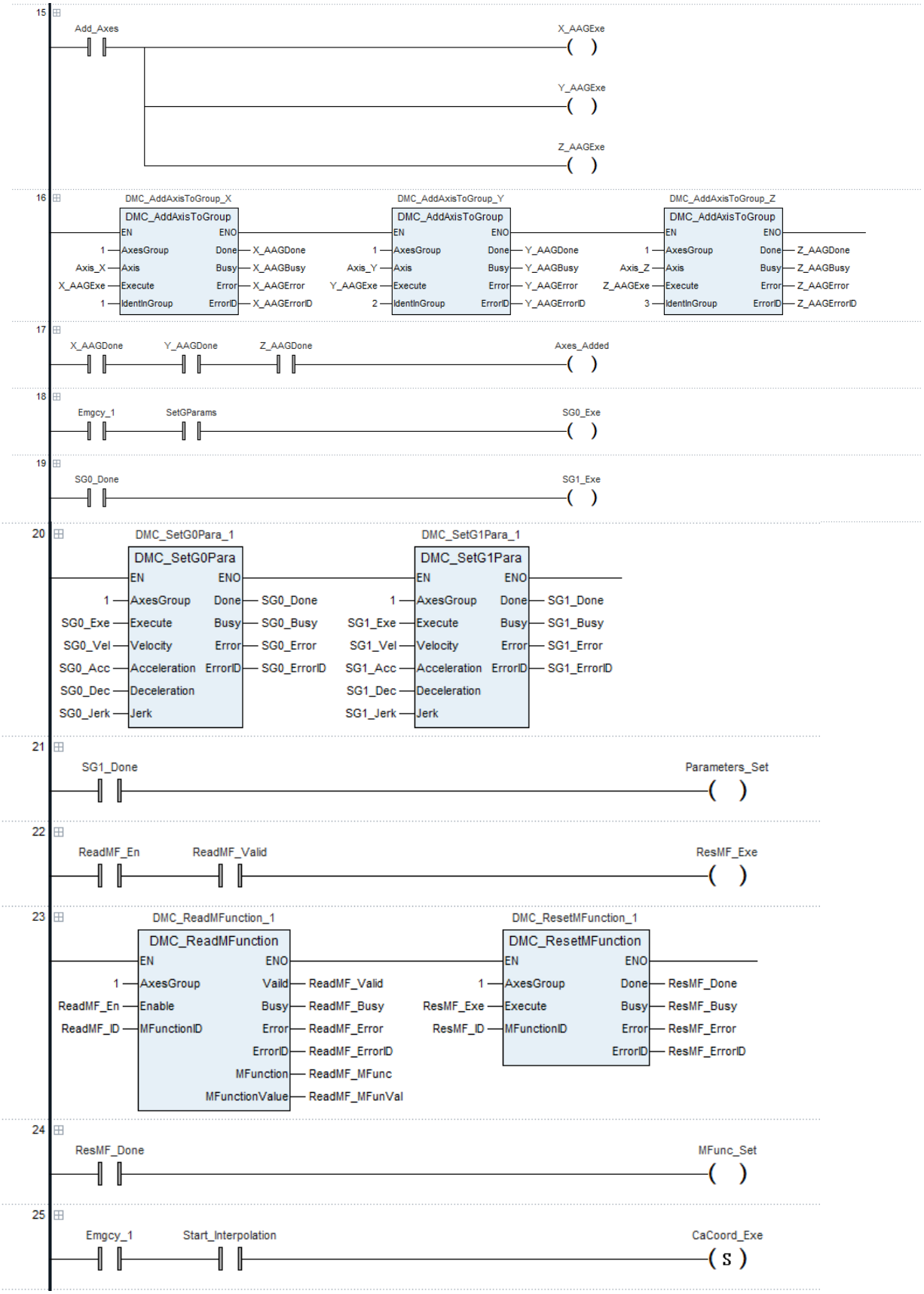
I. Programski kod PLC-a (Ladder diagram)

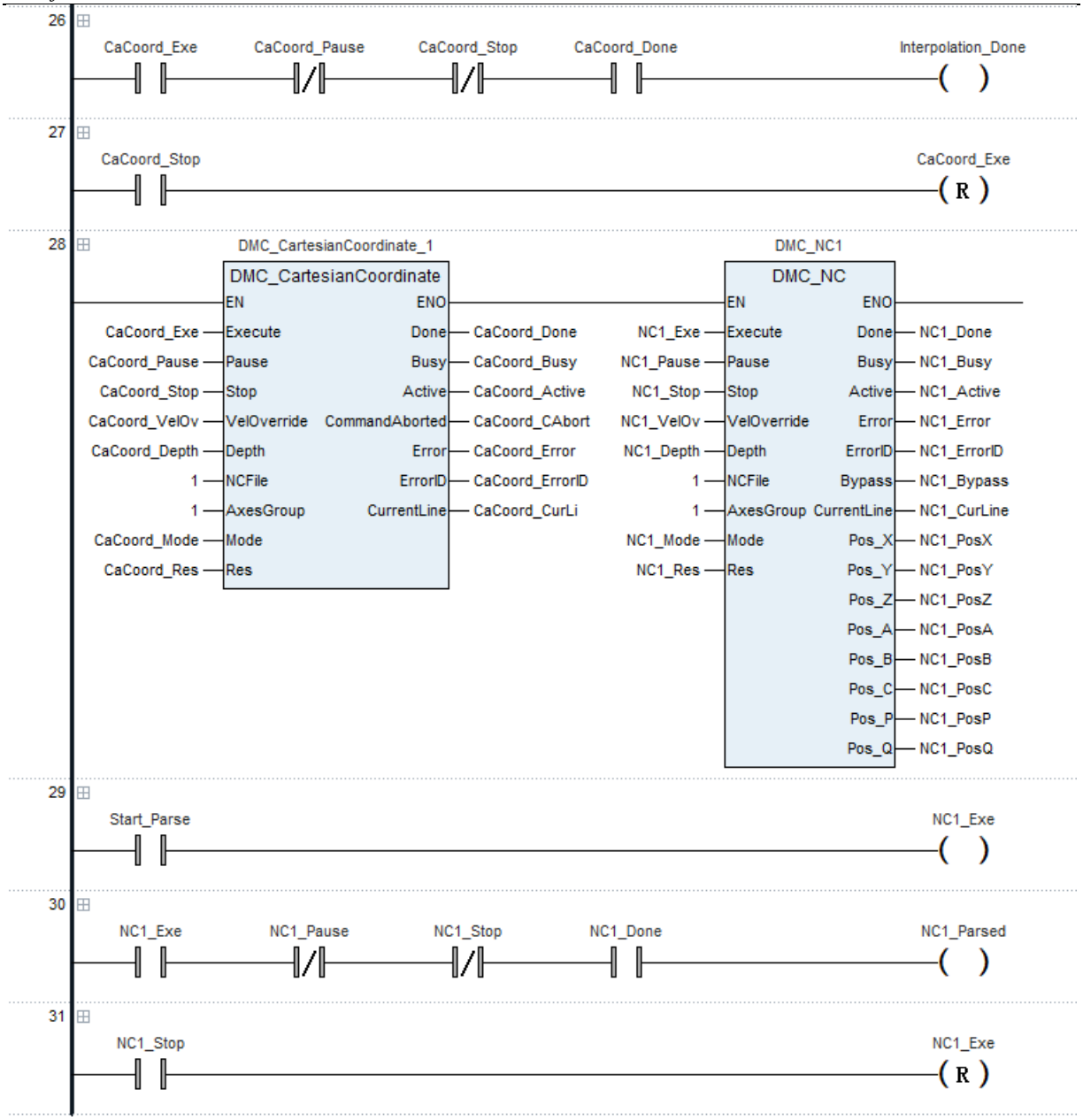
II. Popis signala

III. Tablica M funkcija

I. Programski kod PLC-a (Ladder diagram)




15
Add_Axes
X_AAGExe
Y_AAGExe
Z_AAGExe
16
DMC_AddAxisToGroup_X
DMC_AddAxisToGroup
EN
ENO
1
AxesGroup
Done
X_AAGDone
Axis_X
Axis
Busy
X_AAGBusy
X_AAGExe
Execute
Error
X_AAGError
1
IdentInGroup
ErrorID
X_AAGErrorID
DMC_AddAxisToGroup_Y
1
AxesGroup
Done
Y_AAGDone
Axis_Y
Axis
Busy
Y_AAGBusy
Y_AAGExe
Execute
Error
Y_AAGError
2
IdentInGroup
ErrorID
Y_AAGErrorID
DMC_AddAxisToGroup_Z
1
AxesGroup
Done
Z_AAGDone
Axis_Z
Axis
Busy
Z_AAGBusy
Z_AAGExe
Execute
Error
Z_AAGError
3
IdentInGroup
ErrorID
Z_AAGErrorID
17
X_AAGDone
Y_AAGDone
Z_AAGDone
Axes_Added
18
Emgcy_1
SetGParams
SG0_Exe
19
SG0_Done
SG1_Exe
20
DMC_SetG0Para_1
DMC_SetG0Para
EN
ENO
1
AxesGroup
Done
SG0_Done
SG0_Exe
Execute
Busy
SG0_Busy
SG0_Vel
Velocity
Error
SG0_Error
SG0_Acc
Acceleration
ErrorID
SG0_ErrorID
SG0_Dec
Deceleration
SG0_Jerk
Jerk
DMC_SetG1Para_1
DMC_SetG1Para
1
AxesGroup
Done
SG1_Done
SG1_Exe
Execute
Busy
SG1_Busy
SG1_Vel
Velocity
Error
SG1_Error
SG1_Acc
Acceleration
ErrorID
SG1_ErrorID
SG1_Dec
Deceleration
SG1_Jerk
Jerk
21
SG1_Done
Parameters_Set
22
ReadMF_En
ReadMF_Valid
ResMF_Exe
23
DMC_ReadMFunction_1
DMC_ReadMFunction
EN
ENO
1
AxesGroup
Vaild
ReadMF_Valid
ReadMF_En
Enable
Busy
ReadMF_Busy
ReadMF_ID
MFunctionID
Error
ReadMF_Error
ErrorID
ReadMF_ErrorID
MFunction
ReadMF_MFunc
MFunctionValue
ReadMF_MFunVal
DMC_ResetMFunction_1
DMC_ResetMFunction
EN
ENO
1
AxesGroup
Done
ResMF_Done
ResMF_Exe
Execute
Busy
ResMF_Busy
ResMF_ID
MFunctionID
Error
ResMF_Error
ErrorID
ResMF_ErrorID
24
ResMF_Done
MFunc_Set
25
Emgcy_1
Start_Interpolation
CaCoord_Exe
S


26
CaCoord_Exe
CaCoord_Pause
CaCoord_Stop
CaCoord_Done
Interpolation_Done
27
CaCoord_Stop
CaCoord_Exe
R
28
DMC_CartesianCoordinate_1
DMC_CartesianCoordinate
EN
ENO
CaCoord_Exe
Execute
Done
CaCoord_Done
CaCoord_Pause
Pause
Busy
CaCoord_Busy
CaCoord_Stop
Stop
Active
CaCoord_Active
CaCoord_VelOv
VelOverride
CommandAborted
CaCoord_CAbort
CaCoord_Depth
Depth
Error
CaCoord_Error
1
NCFile
ErrorID
CaCoord_ErrorID
1
AxesGroup
CurrentLine
CaCoord_CurLi
CaCoord_Mode
Mode
CaCoord_Res
Res
DMC_NC1
DMC_NC
EN
ENO
NC1_Exe
Execute
Done
NC1_Done
NC1_Pause
Pause
Busy
NC1_Busy
NC1_Stop
Stop
Active
NC1_Active
NC1_VelOv
VelOverride
Error
NC1_Error
NC1_Depth
Depth
ErrorID
NC1_ErrorID
1
NCFile
Bypass
NC1_Bypass
1
AxesGroup
CurrentLine
NC1_CurLine
NC1_Mode
Mode
Pos_X
NC1_PosX
NC1_Res
Res
Pos_Y
NC1_PosY
Pos_Z
NC1_PosZ
Pos_A
NC1_PosA
Pos_B
NC1_PosB
Pos_C
NC1_PosC
Pos_P
NC1_PosP
Pos_Q
NC1_PosQ
29
Start_Parse
NC1_Exe
30
NC1_Exe
NC1_Pause
NC1_Stop
NC1_Done
NC1_Parsed
31
NC1_Stop
NC1_Exe
R

II. Popis signala

Scope	Name	Address	Data Type	Initial Value	Comment
VAR	Emgcy_1	%IX0.0	BOOL		Input Emergency button 1
VAR	Enable	%MX337.0	BOOL		Input Enable program start
VAR	Axis_X		USINT	4	
VAR	X_EnPos	%MX0.1	BOOL	TRUE	
VAR	X_EnNeg	%MX0.2	BOOL	TRUE	
VAR	X_Buff		MC_Buffer_Mode		
VAR	X_PowBusy	%MX0.3	BOOL		
VAR	X_PowActive	%MX0.4	BOOL		
VAR	X_PowError	%MX0.5	BOOL		
VAR	X_PowErrorID	%MW1	WORD		
VAR	Axis_Y		USINT	3	
VAR	Y_EnPos	%MX4.1	BOOL	TRUE	
VAR	Y_EnNeg	%MX4.2	BOOL	TRUE	
VAR	Y_Buff		MC_Buffer_Mode		
VAR	Y_PowBusy	%MX4.3	BOOL		
VAR	Y_PowActive	%MX4.4	BOOL		
VAR	Y_PowError	%MX4.5	BOOL		
VAR	Y_PowErrorID	%MW3	WORD		
VAR	Axis_Z		USINT	5	
VAR	Z_EnPos	%MX8.1	BOOL	TRUE	
VAR	Z_EnNeg	%MX8.2	BOOL	TRUE	
VAR	Z_Buff		MC_Buffer_Mode		
VAR	Z_PowBusy	%MX8.3	BOOL		
VAR	Z_PowActive	%MX8.4	BOOL		
VAR	Z_PowError	%MX8.5	BOOL		
VAR	Z_PowErrorID	%MW5	WORD		
VAR	Axis_R		USINT	2	
VAR	R_EnPos	%MX12.1	BOOL	TRUE	
VAR	R_EnNeg	%MX12.2	BOOL	TRUE	
VAR	R_Buff		MC_Buffer_Mode		
VAR	R_PowBusy	%MX12.3	BOOL		
VAR	R_PowActive	%MX12.4	BOOL		
VAR	R_PowError	%MX12.5	BOOL		
VAR	R_PowErrorID	%MW7	WORD		
VAR	MC_Power_X		MC_Power		
VAR	MC_Power_Y		MC_Power		
VAR	MC_Power_Z		MC_Power		
VAR	MC_Power_R		MC_Power		
VAR	PowerCmd	%QX0.7	BOOL	FALSE	Power on Command
VAR	DMC_AddAxisToGroup_X		DMC_AddAxisToGroup		
VAR	DMC_AddAxisToGroup_Y		DMC_AddAxisToGroup		
VAR	DMC_AddAxisToGroup_Z		DMC_AddAxisToGroup		
VAR	DMC_ControlAxisByPos_X		DMC_ControlAxisByPos		
VAR	DMC_ControlAxisByPos_Y		DMC_ControlAxisByPos		
VAR	DMC_ControlAxisByPos_Z		DMC_ControlAxisByPos		
VAR	DMC_CartesianCoordinate_1		DMC_CartesianCoordinate		
VAR	DMC_SetG0Para_1		DMC_SetG0Para		
VAR	DMC_ReadMFunction_1		DMC_ReadMFunction		
VAR	DMC_ResetMFunction_1		DMC_ResetMFunction		
VAR	MC_Home_X		MC_Home		
VAR	MC_Home_Y		MC_Home		
VAR	MC_Home_Z		MC_Home		
VAR	MC_Home_R		MC_Home		

VAR	X_EnPow	%MX0.0	BOOL		Enable power for X
VAR	Y_EnPow	%MX4.0	BOOL		Enable power for Y
VAR	Z_EnPow	%MX8.0	BOOL		Enable power for Z
VAR	R_EnPow	%MX12.0	BOOL		Enable power for R
VAR	X_PowStatus	%MX0.6	BOOL	FALSE	
VAR	Y_PowStatus	%MX4.6	BOOL	FALSE	
VAR	Z_PowStatus	%MX8.6	BOOL	FALSE	
VAR	R_PowStatus	%MX12.6	BOOL	FALSE	
VAR	Power_On	%QX0.0	BOOL		
VAR	X_HomeExe	%MX16.0	BOOL	FALSE	
VAR	X_HomeDone	%MX16.1	BOOL	FALSE	
VAR	Y_HomeExe	%MX20.0	BOOL	FALSE	
VAR	Y_HomeDone	%MX20.1	BOOL	FALSE	
VAR	Z_HomeExe	%MX40.0	BOOL	FALSE	
VAR	Z_HomeDone	%MX40.1	BOOL	FALSE	
VAR	R_HomeExe	%MX44.0	BOOL	FALSE	
VAR	R_HomeDone	%MX44.1	BOOL	FALSE	
VAR	X_HomePos	%ML3	LREAL	0	
VAR	X_HomeBuff		MC_Buffer_Mode		
VAR	X_HomeBusy	%MX16.2	BOOL		
VAR	X_HomeActive	%MX16.3	BOOL		
VAR	X_HomeCAbort	%MX16.4	BOOL		
VAR	X_HomeError	%MX16.5	BOOL		
VAR	X_HomeErrorID	%MW9	WORD		
VAR	Y_HomePos	%ML4	LREAL	0	
VAR	Y_HomeBuff		MC_Buffer_Mode		
VAR	Y_HomeBusy	%MX20.2	BOOL		
VAR	Y_HomeActive	%MX20.3	BOOL		
VAR	Y_HomeCAbort	%MX20.4	BOOL		
VAR	Y_HomeError	%MX20.5	BOOL		
VAR	Y_HomeErrorID	%MW11	WORD		
VAR	Z_HomePos	%ML6	LREAL	0	
VAR	Z_HomeBuff		MC_Buffer_Mode		
VAR	Z_HomeBusy	%MX40.2	BOOL		
VAR	Z_HomeActive	%MX40.3	BOOL		
VAR	Z_HomeCAbort	%MX40.4	BOOL		
VAR	Z_HomeError	%MX40.5	BOOL		
VAR	Z_HomeErrorID	%MW21	WORD		
VAR	R_HomePos	%ML7	LREAL	0	
VAR	R_HomeBuff		MC_Buffer_Mode		
VAR	R_HomeBusy	%MX44.2	BOOL		
VAR	R_HomeActive	%MX44.3	BOOL		
VAR	R_HomeCAbort	%MX44.4	BOOL		
VAR	R_HomeError	%MX44.5	BOOL		
VAR	R_HomeErrorID	%MW23	WORD		
VAR	Homing_Done	%QX0.1	BOOL		Output Homing is done
VAR	X_AAGExe	%MX64.0	BOOL	FALSE	
VAR	Y_AAGExe	%MX68.0	BOOL	FALSE	
VAR	Z_AAGExe	%MX72.0	BOOL	FALSE	
VAR	X_AAGDone	%MX64.1	BOOL	FALSE	
VAR	Y_AAGDone	%MX68.1	BOOL	FALSE	
VAR	Z_AAGDone		BOOL	FALSE	
VAR	DMC_SetG1Para_1		DMC_SetG1Para		
VAR	X_AAGBusy	%MX64.2	BOOL		
VAR	X_AAGError	%MX64.3	BOOL		

VAR	X_AAGErrorID	%MW33	WORD		
VAR	Y_AAGBusy	%MX68.2	BOOL		
VAR	Y_AAGError	%MX68.3	BOOL		
VAR	Y_AAGErrorID	%MW35	WORD		
VAR	Z_AAGBusy	%MX72.1	BOOL		
VAR	Z_AAGError	%MX72.2	BOOL		
VAR	Z_AAGErrorID	%MW37	WORD		
VAR	SG0_Exe	%MX76.0	BOOL	FALSE	
VAR	SG0_Vel	%ML10	ARRAY [1..8] OF LREAL	[200,200,200]	
VAR	SG0_Acc	%ML18	ARRAY [1..8] OF LREAL	[100,100,100]	
VAR	SG0_Dec	%ML26	ARRAY [1..8] OF LREAL	[100,100,100]	
VAR	SG0_Jerk	%ML34	ARRAY [1..8] OF LREAL	[50,50,50]	
VAR	SG0_Done	%MX76.1	BOOL		
VAR	SG0_Busy	%MX76.2	BOOL		
VAR	SG0_Error	%MX76.3	BOOL		
VAR	SG0_ErrorID	%MW39	WORD		
VAR	SG1_Exe	%MX336.0	BOOL	FALSE	
VAR	SG1_Vel	%ML43	LREAL	200	
VAR	SG1_Acc	%ML44	LREAL	100	
VAR	SG1_Dec	%ML45	LREAL	100	
VAR	SG1_Jerk	%ML46	LREAL	50	
VAR	SG1_Done	%MX336.1	BOOL		
VAR	SG1_Busy	%MX336.2	BOOL		
VAR	SG1_Error	%MX336.3	BOOL		
VAR	SG1_ErrorID	%MW169	WORD		
VAR	CaCoord_Exe		BOOL	FALSE	
VAR	CaCoord_Pause		BOOL	FALSE	
VAR	CaCoord_Stop		BOOL	FALSE	
VAR	CaCoord_VelOv		LREAL	100	
VAR	CaCoord_Depth		UINT	1	
VAR	CaCoord_Mode		INT	0	
VAR	CaCoord_Res		LREAL		
VAR	CaCoord_Done		BOOL	FALSE	
VAR	CaCoord_Busy		BOOL	FALSE	
VAR	CaCoord_Active		BOOL	FALSE	
VAR	CaCoord_CAbort		BOOL	FALSE	
VAR	CaCoord_Error		BOOL	FALSE	
VAR	CaCoord_ErrorID		WORD	0	
VAR	CaCoord_CurLi		UDINT		
VAR	ReadMF_En		BOOL	FALSE	Input Read M Function
VAR	ReadMF_ID		USINT		
VAR	ReadMF_Valid		BOOL	FALSE	
VAR	ReadMF_Busy		BOOL		
VAR	ReadMF_Error		BOOL		
VAR	ReadMF_ErrorID		WORD		
VAR	ReadMF_MFunc		BOOL		
VAR	ReadMF_MFunVal		LREAL		
VAR	ResMF_Exe		BOOL	FALSE	
VAR	ResMF_ID		USINT		
VAR	ResMF_Done		BOOL		
VAR	ResMF_Busy		BOOL		
VAR	ResMF_Error		BOOL		
VAR	ResMF_ErrorID		WORD		
VAR	NC1_Exe		BOOL	FALSE	Execute NC code parsing

VAR	NC1_Pause		BOOL	FALSE	
VAR	NC1_Stop		BOOL	FALSE	
VAR	NC1_VelOv		LREAL	100	
VAR	NC1_Depth		UINT	1	
VAR	NC1_Mode		INT	0	
VAR	NC1_Res		LREAL	0	
VAR	NC1_Done		BOOL	FALSE	
VAR	NC1_Busy		BOOL	FALSE	
VAR	NC1_Active		BOOL	FALSE	
VAR	NC1_Error		BOOL	FALSE	
VAR	NC1_ErrorID		WORD	0	
VAR	NC1_Bypass		BOOL	FALSE	
VAR	NC1_CurLine		UDINT	0	
VAR	NC1_PosX		LREAL	0	
VAR	NC1_PosY		LREAL	0	
VAR	NC1_PosZ		LREAL	0	
VAR	NC1_PosA		LREAL	0	
VAR	NC1_PosB		LREAL	0	
VAR	NC1_PosC		LREAL	0	
VAR	NC1_PosP		LREAL	0	
VAR	NC1_PosQ		LREAL	0	
VAR	NC1_Parsed	%QX0.6	BOOL	FALSE	Output NC code is parsed
VAR	Axes_Added	%QX0.2	BOOL		Output Axes added to group
VAR	Parameters_Set	%QX0.3	BOOL		Output Parameters are set
VAR	MFunc_Set	%QX0.4	BOOL		Output Mfunction is set
VAR	Homing_Exe	%MX337.2	BOOL		Input Homing Execute
VAR	Add_Axes	%MX337.3	BOOL		Input Add axes to group
VAR	SetGParams	%MX337.4	BOOL		Input Set parameters
VAR	DMC_NC1		DMC_NC		
VAR	Interpolation_Done	%QX0.5	BOOL		Output Interpolation done
VAR	Start_Parse	%MX337.5	BOOL		Input Start with parsing
VAR	Start_Interpolation	%MX337.6	BOOL		Input Start with interpolation

III. Tablica M funkcija

M Funkcija	Opis
M00	Obavezni programski stop
M01	Uvjetni programski stop
M02	Kraj programa
M03	Okretanje vretena u desno
M04	Okretanje vretena u lijevo
M05	Zaustavljanje vretena
M06	Promjena alata
M08	Uključivanje hlađenja
M09	Isključivanje hlađenja
M10	Blokiranje 4. osi
M11	Deblokiranje 4. osi
M12	Blokiranje 5. osi
M13	Deblokiranje 5. osi
M16	Promjena alata
M19	Orijentacija vretena
M21 - M28	Optimalne korisničke funkcije
M29	Postavljanje izlaznih releja s M funkcijom
M30	Kraj i resetiranje glavnog programa
M31	Transporter strugotine prema naprijed
M32	Transporter strugotine prema nazad
M33	Zaustavljanje transportera strugotine
M34	Povećanje rashladne tekućine
M35	Smanjenje rahladne tekućine
M36	Dio za paletu spreman
M39	Zakretanje revolvera s alatima
M41	Niski stupanj prijenosa
M42	Visoki stupanj prijenosa
M50	Promjena palete
M51 - M58	Optimalne korisničke M funkcije
M59	Postavljanje izlaznih releja
M61 - M68	Čišćenje optimalnih korisničkih M funkcija
M69	Čišćenje izlaznih releja
M70 - M71	Stezanje i otpušanje stezaljki
M76	Monitor isključen
M77	Monitor uključen
M78	Generira alarm
M79	Generira alarm
M80	Automatsko otvaranje vrata
M81	Automatsko zatvaranje vrata
M82	Otpuštanje alata
M83	Uključivanje pištolja za zrak
M84	Isključivanje pištolja za zrak
M86	Stezanje alata
M88	Hlađenje kroz vreteno uključeno
M89	Hlađenje kroz vreteno isključeno
M90	Ulaz za stezaljku uključen
M91	Ulaz za stezaljku isključen
M95	Sleep mode
M96	Skok ako nema unosa
M97	Pozivanje lokalnog podprograma
M98	Pozivanje eksternog podprograma
M99	Kraj podprograma