

Primena PLC u elektroenergetici

■ Osnove programiranja PLC-a

■ Osnove programiranja PLC-a

Da bi PLC obavljao neku funkciju, definisanu od strane korisnika, u njega mora da se upiše odgovarajući program (kod).

Bez programa/koda/izvršnog koda PLC neće moći da obavlja bilo koju funkciju – beskoristan je.

U memoriju za program PLC-a, upisuju se naredbe koje su napisane mašinskim jezikom. Mašinski jezik (assembler) je izrazito nepraktičan za upotrebu i kreiranje bilo kakvog programa u assembleru je vrlo mukotrpan posao. Ranije su se svi kontroleri, pa i PLC, programirali u assembleru, ali to danas na sreću nije slučaj.

Za kreiranje programa za PLC danas se koriste dva osnovna pristupa – tekstualni i grafički.

Koji god pristup za kreiranje programa da se koristi, program se mora prevesti u skup naredbi na mašinskom jeziku (kompajlirati) kako bi bio upotrebljiv u kontroleru.

Dakle, proces uspostavljanja funkcionalnog PLC okruženja, pored instalacije hardvera, podrazumeva sledeće korake:

Definisanje problema – definisanje (logike) rešenja – kreiranje programa (grafički ili tekstualno) – kompajliranje – spuštanje kompajliranog programa na PLC – start programa.

Nakon ovih programa PLC i čitav pogon bi trebali automatski, ili barem poluautomatski da rade unedogled.

■ Osnove programiranja PLC-a

Za kreiranje programa za PLC danas se koriste dva osnovna pristupa – tekstualni i grafički.

Tekstualni - Structured control language (SLC)

Obezbeđuje fleksibilnost kod programiranja PLC-a i kod manipulisanja resursima PLC-a.

Podseća na C programski jezik.

Predstavlja značajno unapređenje u odnosu na asemblersko programiranje, ali je pisanje koda i dalje relativno složen posao.

Naravno, i dalje postoji mogućnost programiranja u assembleru, koji se takođe može svrstati u tekstualni pristup programiranja PLC-a.

Programming language	S7-1200	S7-1500
Ladder (LAD)	✓	✓
Function block diagram (FBD)	✓	✓
Structured control language (SCL)	✓	✓
Graph	✗	✓
Statement list (STL)	✗	✓

■ Osnove programiranja PLC-a

Za kreiranje programa za PLC danas se koriste dva osnovna pristupa – tekstualni i grafički.

Grafički – Ladder logic (LAD)

Obezbeđuje nešto manju fleksibilnost kod programiranja PLC-a i kod manipulisanja resursima PLC-a u odnosu na tekstualni pristup programiranja, ali razlike se sve više smanjuju u ovom smislu.

Sve (ili većina naredbi) se definiše grafički – pomoću postojećih, unapred definisanih funkcionalnih blokova (koji imaju grafički prikaz).

„Kod“ se „piše“ prevlačenjem blokova iz biblioteke (drag-and-drop) u radni prostor, podešavanjem blokova i njihovim međusobnih spajanjem.

Programiranje pomoću Ladder logic-a (leder dijagrama na srpskom) je jednostavno i intuitivno. Vrlo brzo se od definisanja programa može doći do njegove kreacije i PLC-a koji je programiran da radi određen posao.

Programming language	S7-1200	S7-1500
Ladder (LAD)	✓	✓
Function block diagram (FBD)	✓	✓
Structured control language (SCL)	✓	✓
Graph	✗	✓
Statement list (STL)	✗	✓

■ Osnove programiranja PLC-a

Za kreiranje programa za PLC danas se koriste dva osnovna pristupa – tekstualni i grafički.

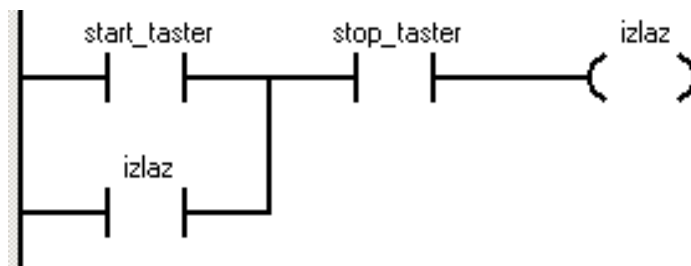
Grafički – Ladder logic (LAD)

Obezbeđuje nešto manju fleksibilnost kod programiranja PLC-a i kod manipulisanja resursima PLC-a u odnosu na tekstualni pristup programiranja, ali razlike se sve više smanjuju u ovom smislu.

Sve (ili većina naredbi) se definiše grafički – pomoću postojećih, unapred definisanih funkcionalnih blokova (koji imaju grafički prikaz).

„Kod“ se „piše“ prevlačenjem blokova iz biblioteke (drag-and-drop) u radni prostor, podešavanjem blokova i njihovim međusobnim spajanjem.

Programiranje pomoću Ladder logic-a (ladder dijagrama na srpskom) je jednostavno i intuitivno. Vrlo brzo se od definisanja programa može doći do njegove kreacije i PLC-a koji je programiran da radi određen posao.



Program se sastoji od uslova koji definišu da li će se neka instrukcija izvršiti !

■ Osnove programiranja PLC-a

Za kreiranje programa za PLC danas se koriste dva osnovna pristupa – tekstualni i grafički.

Grafički – Function block diagram (FBD)

FBD je vrlo sličan LAD programiranju. Oba su grafički jezici i svode se na uzimanje elemenata iz biblioteke i njihovo prevlačenje u radni prostor.

Osnovna razlika je u tome što FBD ima karakteristiku da se određene operacije/naredbe mogu konciznije iskazati pomoću dostupnih elemenata.

Na primer, logičko I se u LAD realizuje pomoću više serijski povezanih elemenata, dok se ova operacija u FBD realizuje pomoću jednog bloka (bloka „logičko I“ koji ima više ulaza).

Na ovom kursu se analizira upotreba Ladder logic programiranja.

Programming language	S7-1200	S7-1500
Ladder (LAD)	✓	✓
Function block diagram (FBD)	✓	✓
Structured control language (SCL)	✓	✓
Graph	✗	✓
Statement list (STL)	✗	✓

■ Osnove programiranja PLC-a

Pre nego što zađemo u detalje Ladder logic programiranja, objasniti kako se izvestan program realizuje na samom PLC-u.

Sve operacije koje PLC izvršava se izvršavaju **ciklično** – u okviru jednog *sken ciklusa*, bez obzira da li operacije pripadale programu koji je razvio korisnik ili ne.



■ Osnove programiranja PLC-a

Pre nego što zađemo u detalje Ladder logic programiranja, objasniti kako se izvestan program realizuje na samom PLC-u.

Sve operacije koje PLC izvršava se izvršavaju **ciklično** – u okviru jednog *sken ciklusa*, bez obzira da li operacije pripadale programu koji je razvio korisnik ili ne.

1. Prihvat stanja (analognih i digitalnih) ulaza (senzor, prekidač...) u memoriju, u posebno područje koje predstavlja sliku ulaza.
2. Izvršenje programa - Ulazni operandi: memorija (slika ulaza). Rezultati: u memoriju (slika izlaza). Dakle, podaci: ne direktno sa ulaznih modula, niti se direktno iznose na izlazne module, već program razmenjuje podatke isključivo sa memorijom!
3. Ispis stanja izlaza - prenos podataka iz slike izlaza na izlazne linije (registre izlaznih modula) odnosno na izlazne digitalne i analogne stepene.
4. Komunikacija – prihvat ili slanje poruka putem nekog od protokola koje koristi PLC (PROFIBUS npr.).
5. Održavanje – ažuriranje internih časovnika i registara, upravljanje memorijom, kontrola ispravnosti PLC-a, kao i niz drugih poslova vezanih za održavanje sistema



■ Osnove programiranja PLC-a

Pre nego što zađemo u detalje Ladder logic programiranja, objasniti kako se izvestan program realizuje na samom PLC-u.

Sve operacije koje PLC izvršava se izvršavaju **ciklično** – u okviru jednog ***sken ciklusa***, bez obzira da li operacije pripadale programu koji je razvio korisnik ili ne.

Posle izvršenja petog koraka PLC se vraća na početak ciklusa i neprekidno ponavlja sve korake.

Osnovni sken ciklus može biti modifikovan pomoću zahteva za prekid ili nekih drugih specijalnih programskih naredbi.

Prihvat ulaza, obrada, ispis izlaza, komunikacija i održavanje se ciklički ponavljaju i u skladu sa unesenim programom, njihovo pojedinačno trajanje zavisi od broja ulaza i izlaza i složenosti algoritma i vrste primenjenog procesora. Ciklusi obrade obično traju od 0,1ms do nekoliko desetina ms.



■ Osnove programiranja PLC-a

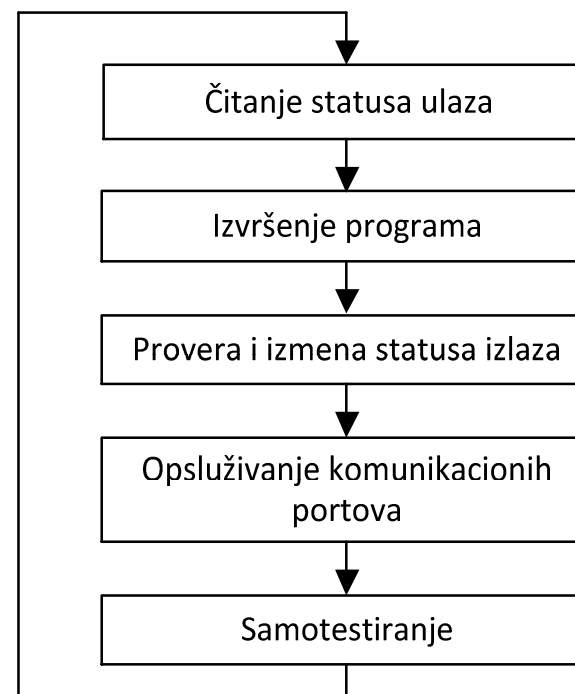
Pre nego što zađemo u detalje Ladder logic programiranja, objasniti kako se izvestan program realizuje na samom PLC-u.

Sve operacije koje PLC izvršava se izvršavaju **ciklično** – u okviru jednog *sken ciklusa*, bez obzira da li operacije pripadale programu koji je razvio korisnik ili ne (drugim rečima).

Dodatna dva koraka koje program u modernim PLC izvršava ciklično

Korak 4.
opsluživanje serijskih
portova

Korak 5.
samotestiranje

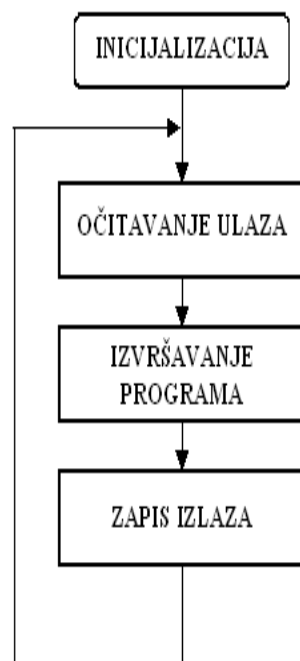


■ Osnove programiranja PLC-a

Pre nego što zađemo u detalje Ladder logic programiranja, objasniti kako se izvestan program realizuje na samom PLC-u.

Sve operacije koje PLC izvršava se izvršavaju **ciklično** – u okviru jednog ***sken ciklusa***, bez obzira da li operacije pripadale programu koji je razvio korisnik ili ne (trećim rečima).

Tri osnovna koraka (faze) koja se ponavljaju:



Korak 1.

Testiranje ulaza. Koji ima status ON ili status OFF. Svi ovi ulazi se upisuju u odgovarajuće memorijske lokacije. Te vrednosti se koriste tokom celog sledećeg koraka.

Korak 2.

Izvršenje programa, instrukciju po instrukciju. Na osnovu programa, i na osnovu stanja u programu korišćenih ulaza odgovarajuća akcija se vrši. Akcija može biti definisana i na osnovu rezultata sačuvanog od ranije u internoj memoriji.

Korak 3.

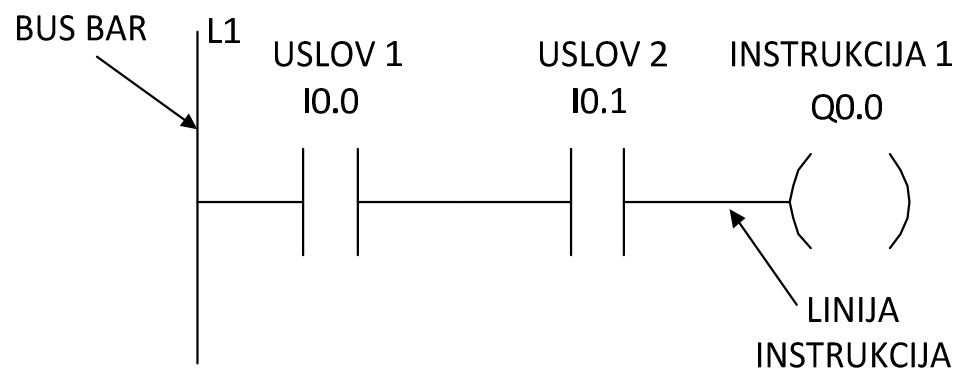
Provera i korekcija izlaza. Izlazi dobijaju stanje u zavisnosti od stanja ulaza iz koraka 1 i izvršenog programa iz stanja 2.

Nakon koraka 3 PLC počinje da izvršava ponovo korak 1.

Vreme skeniranja zavisi od programa i od tipa PLC.

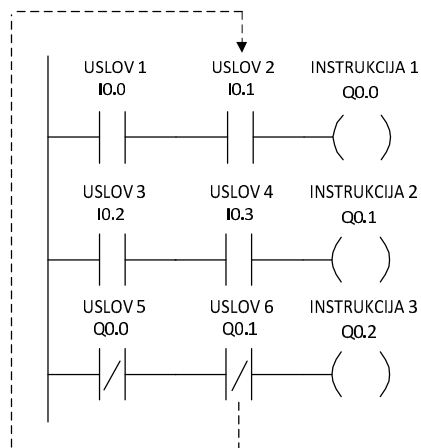
■ Osnove programiranja PLC-a

Ladder logic programiranje



Ako su zadovoljeni USLOV1 i USLOV2
INSTRUKCIJA 1 će se izvršiti.

U datom primeru ako su IO.0 i IO.1 =1 izlaz
Q0.0 će biti 1 , tj. 24V.



Instrukcije se izvršavaju redom.
Nakon tačno određenog vremena ,
scan time, ciklus se ponavlja.

Operacije se izvršavaju s desna
na levo i od gore ka dole.

■ Osnove programiranja PLC-a

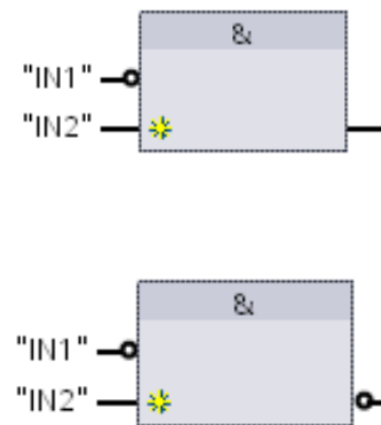
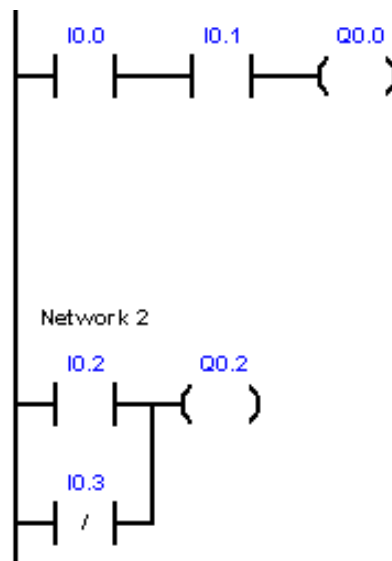
Ladder logic programiranje

Uslovi mogu biti kontrolisani raznim bitovima, ulaznim I0.0, izlaznim bitovima Q0.0, memorijskim bitovima M0.0, tajmerskim T37, brojačkim C30...

Na osnovu trenutnog stanja bitova
Koji ih kontrolišu izabrana kombinacija
uslova definiše da li će instrukcija
biti omogućena ili ne.

$Q0.0 = I0.0 \text{ i } I0.1$

$Q0.2 = I0.2 \text{ ili } I0.3$



Ladder logic programiranje - kratko podsećanje na Bulovu algebru.

1. Zatvorenost: za svaki element x i y iz skupa B važi
 - (i) $x + y$ je element B ,
 - (ii) $x \cdot y$ je element B .
2. Postojanje neutralnih elemenata za operacije $+$ i $\cdot$
 - (i) postoji element 0 iz B takav da za svako x iz B važi $0 + x = x + 0 = x$,
 - (ii) postoji element 1 iz B takav da za svako x iz B važi $1 \cdot x = x \cdot 1 = x$.
$$\exists 0, 1 \in B \therefore \forall x \in B \Rightarrow (i) \wedge (ii).$$
3. Komutativnost: za sve elemente x i y u skupu B važi
 - (i) $x + y = y + x$,
 - (ii) $x \cdot y = y \cdot x$.
4. Distributivnost: za sve elemente x, y i z u skupu B važi
 - (i) $x \cdot (y + z) = x \cdot y + x \cdot z$,
 - (ii) $x + (y \cdot z) = (x + y) \cdot (x + z)$.
5. Postojanje inverznog elementa: Za svaki element x iz skupa B postoji u B element $\bar{x}$, takav da važi
 - (i) $x + \bar{x} = 1$,
 - (ii) $x \cdot \bar{x} = 0$.
6. U skupu B postoje najmanje dva različita elementa, tj. $0 \neq 1$.

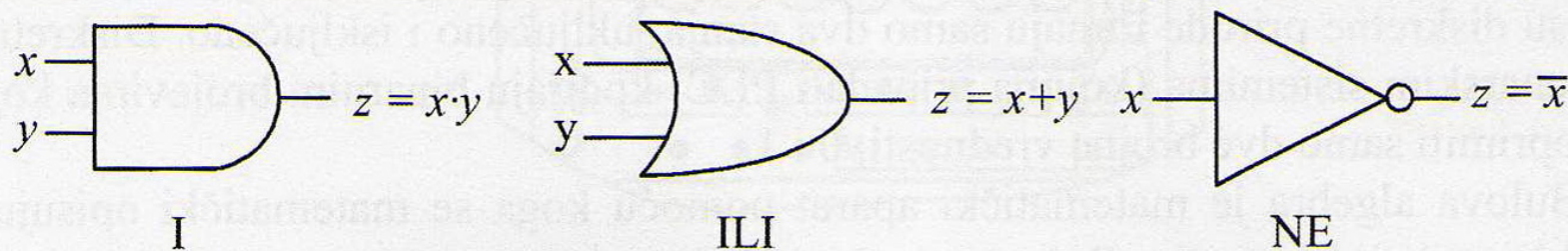
■ Osnove programiranja PLC-a

Ladder logic programiranje - kratko podsećanje na Bulovu algebru.

		I (AND)	ILI (OR)	NE (NO)
x	y	$z = x \cdot y$	$z = x + y$	$z = \bar{x}$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Tabela 5-1 Tabelarni prikaza definicija logičkih operacija

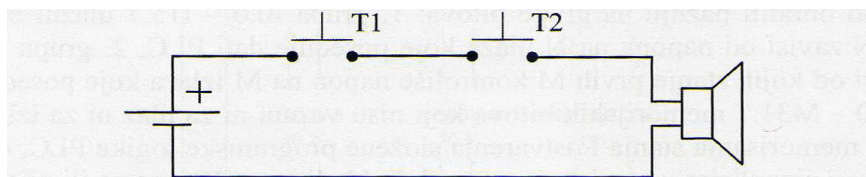
Na slici 5.1 prikazani su simboli logičkih kola za realizaciju Bulovih funkcija.



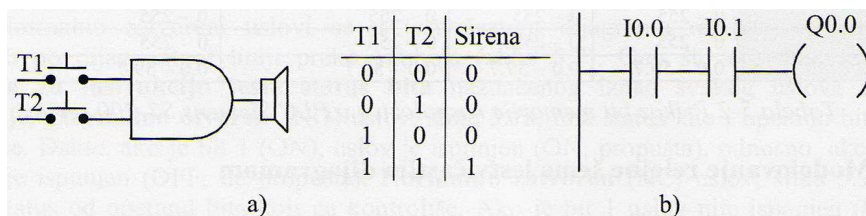
Slika 5.1 Simboli logičkih kola I, ILI i NE

■ Osnove programiranja PLC-a

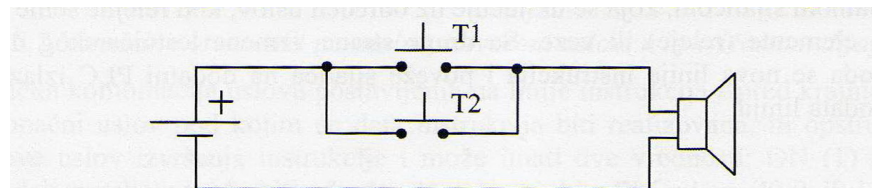
Ladder logic programiranje - kratko podsećanje na Bulovu algebru.



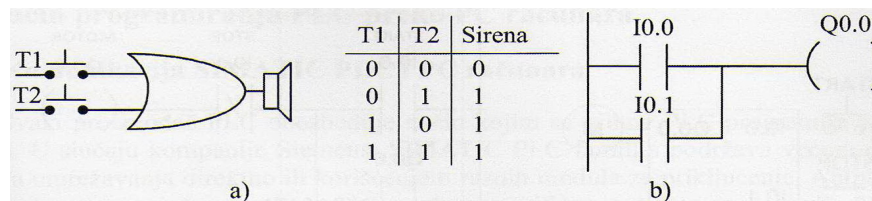
Slika 5.9 Električno kolo sa dva redno vezana tastera i zvučnikom



Slika 5.10 Za kolo na slici 5.9: a) binarna logika b) lestvičarsti dijagram



Slika 5.11 Električno kolo sa dva paralelno vezana tastera i zvučnikom



Slika 5.12 Za kolo na slici 5.11: a) binarna logika b) lestvičasti dijagram

■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

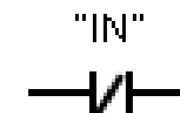
NO

simbolički, energija teče kroz kontakt kada je zatvoren. NO je zatvoren kada je statusni bit koji kontroliše taj kontakt 1 (npr. kada je spoljni prekidač zatvoren, tj. ima se 24V na ulazu PLC-a)



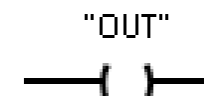
NC

obratno: provodan kada je statusni bit 0 (tj. kada je prekidač otvoren)



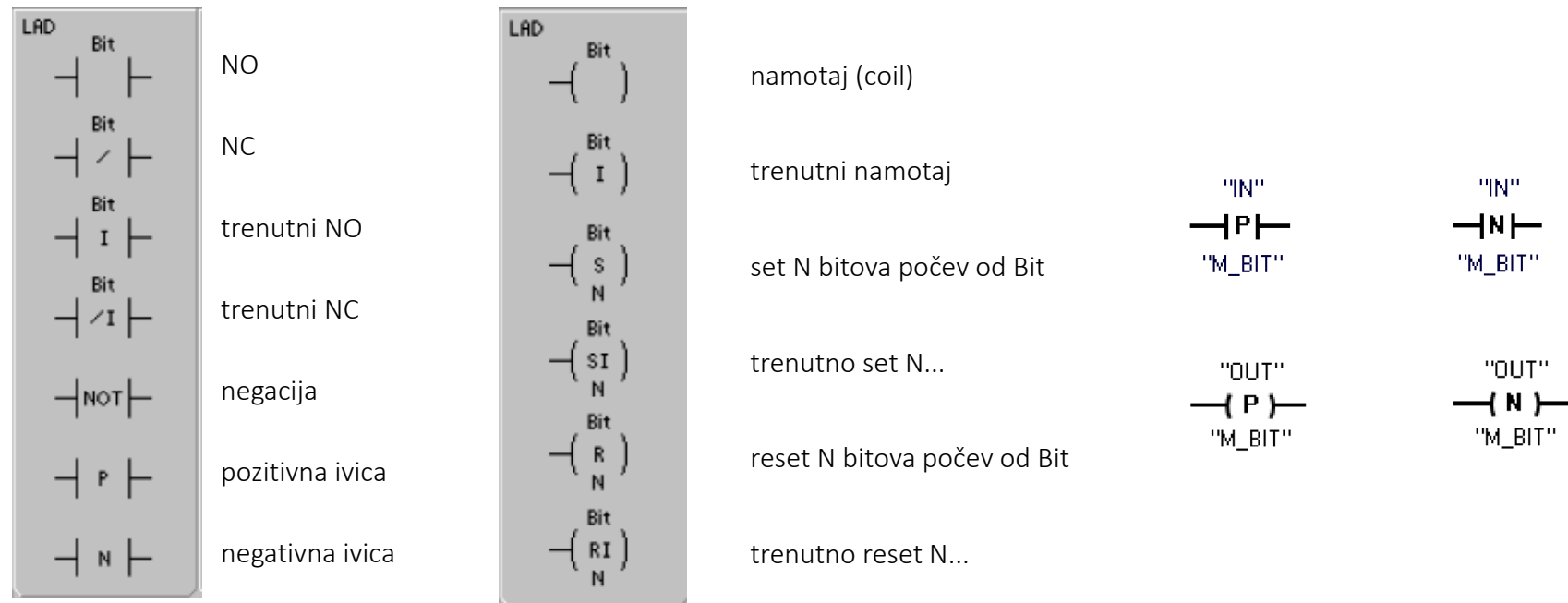
Namotaj (Coil)

simbolički predstavlja relej koji je pobuđen (energized) kada energija teče prema njemu. Kada je pobuđen, on uključuje statusni bit odgovarajućeg **izlaza**. Isti statusni bit može biti korišćen da kontroliše NO i NC kontakte **bilo gde u programu**



■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.



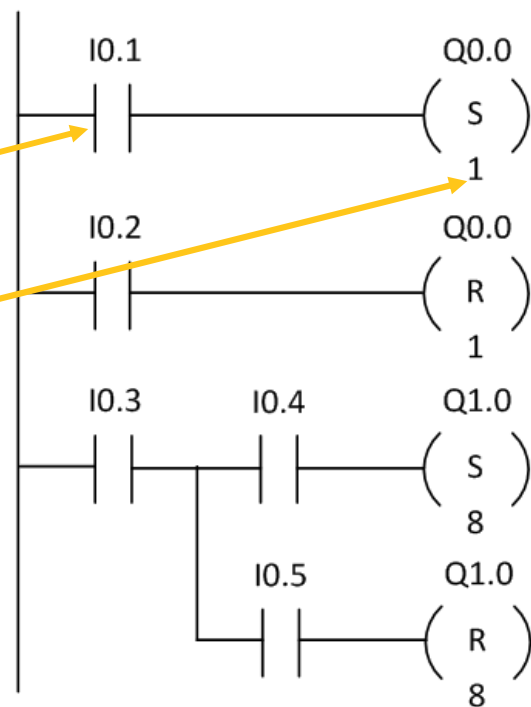
■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

SET (S) i RESET (R)

instrukcija vrše setovanje (uključenje; postavljanje na nivo 1), odnosno resetovanje (isključenje; spuštanje na nivo 0) određenog broja pozicija (do 255), počevši od lokacije određenog adresnog bita. Ipak, tipično se setuje ili resetuje jedna pozicija, tj. jedan bit.

Aktivacija ulaza I0.1 u prvoj liniji instrukcija omogućava izvršenje SET instrukcije koja setuje jednu bit lokaciju, tj. izlaz Q0.0. Aktivacija ulaza I0.2 u drugoj liniji instrukcija omogućava izvršenje RESET instrukcije koja isti taj izlaz resetuje. Ukoliko ulazi I0.1 i I0.2 nisu više aktivni izlaz Q0.0 pamti svoje prethodno stanje i ostaje u stanju set ili reset.

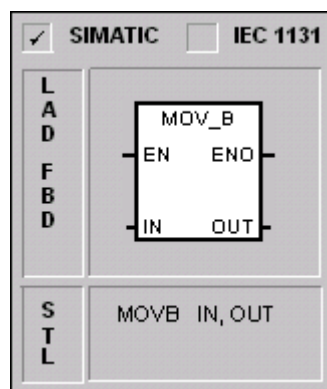


■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

DODELA VREDNOSTI

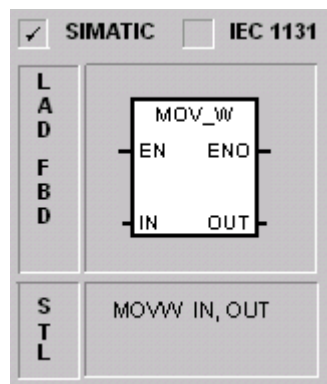
MOV instrukcija



Move Byte (MOVB) instrukcija kopira broj u ulaznom bajtu (IN) u izlazni bajt (OUT) bez promene vrednosti.



Ovaj primer upisuje broj 102 u lokaciju VB10



Move Word (MOVW) instrukcija kopira broj u ulaznom reči (IN) u izlaznu reč (OUT) bez promene vrednosti.
Word = 16 bitova = 2 bajta

Ove dve instrukcije se često za podešavanje parametara dodatnih modula. Na primer, podešavanje nekog komunikacionog modula zahteva 9600 brzinu prenosa u lokaciji X koja definiše baudrate. Tada, u registar X preko MOVEW postavimo 9600 tako što IN povežemo sa 9600, a na OUT adresu registra X.



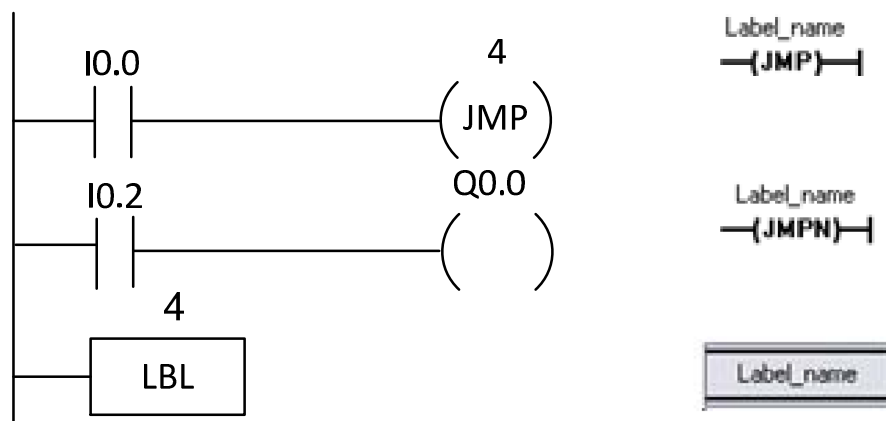
■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

SKOK

Kontrola toka programa, JMP and LBL

Jump to label (JMP) instrukcija predstavlja grananje (skok) ka određenoj liniji tj. labeli (LBL) unutar programa. Ukoliko je aktivan NO uslov kontrolisan bitom IO.0, dakle ako je na ulazu IO.0 napon od 24V, omogućava se izvršenje *Jump to Label* instrukcije. U datom primeru omogućena JMP instrukcija izvršava grananje programa ka 4-toj labeli (poziciji) u programu. Isti broj 4 se nalazi iznad instrukcije JMP i njoj odgovarajuće labele. Dakle, ako je IO.0=1, u datom PLC ciklusu se preskače druga instrukcijska linija programa i ona se neće izvršiti.

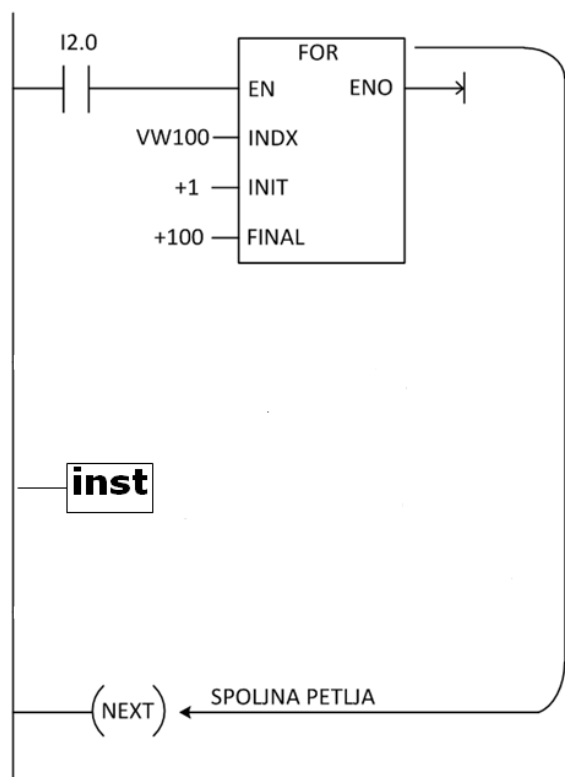


■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

PETLJA

Kontrola toka programa, FOR and NEXT



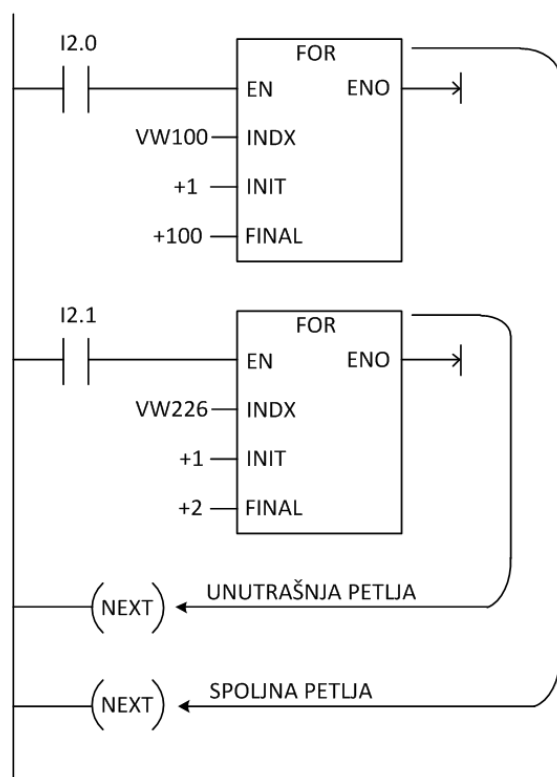
FOR naredba izvršava instrukcije koje se nalaze između reči FOR i NEXT. Za ovu instrukciju je potrebno zadati indeksnu vrednost (u primeru mem lokacija VW100), početnu (1) i krajnju vrednost (100). Primer će 100 puta izvršiti spoljnu petlju a u svakoj spoljnoj i dva puta unutrašnju, dakle ukupno 200 prelazaka i instrukcija inst bi se izvršila 200 puta u svakom PLC ciklusu.

■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

PETLJA

Kontrola toka programa, FOR and NEXT



FOR naredba izvršava instrukcije koje se nalaze između reči FOR i NEXT. Za ovu instrukciju je potrebno zadati indeksnu vrednost (u primeru mem lokacija VW100), početnu (1) i krajnju vrednost (100). Primer će 100 puta izvršiti spoljnu petlju a u svakoj spoljnoj i dva puta unutrašnju, dakle ukupno 200 prelazaka i instrukcija inst bi se izvršila 200 puta u svakom PLC ciklusu.

■ Osnove programiranja PLC-a

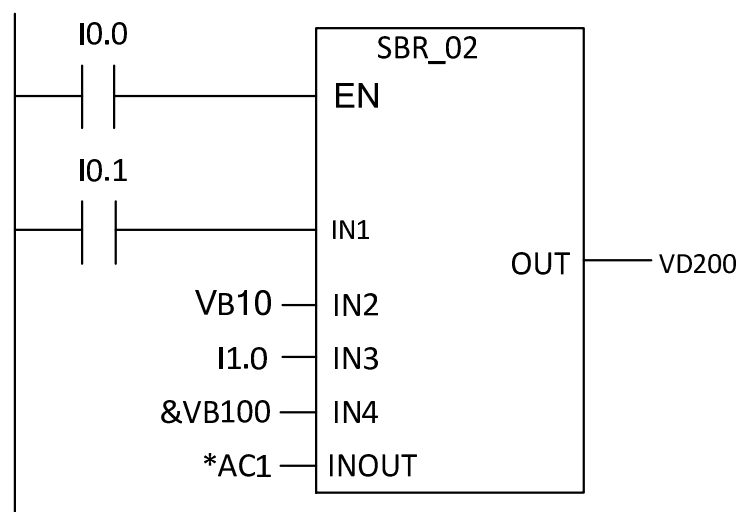
Ladder logic programiranje – osnovni elementi i operacije.

POTPROGRAM

Kontrola toka programa

Glavni PLC program može da inicira izvršenje potprograma. Potprogram se izvršava do kraja i nakon toga predaje kontrolu glavnom programu nad mrežom sa koje je potprogram bio pozvan.

Potprogram se često koristi pri radu sa komunikacionim ili nekim drugim modulima. PLC programer samo preuzme potprogramski blok koji inicijalizuje ili kontroliše neki od serijskih portova i na taj način, bez potrebe za razumevanjem svih detalja, kontroliše rad tih portova, šalje ili prima podatke, itd.



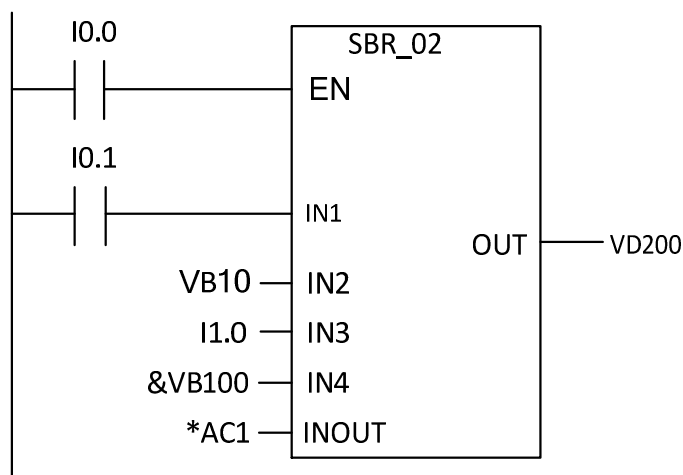
■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

POTPROGRAM

Kontrola toka programa

Tip parametra	Objašnjenje
IN	Parametri potprograma. Ako upravlja adresom (kao što je VB10), vrednost bita na određenoj lokaciji se uvodi u potprogram. Ako je podatak konstanta (16#1234), ili neka adresa (&VB100), vrednost konstante ili adrese je provedena u potprogram.
IN_OUT	Provodi vrednost sa određene lokacije parametra i rezultujuću vrednost iz potprograma vraća na istu lokaciju. Konstante (kao što je 16#1234) i adrese (kao što je &VB100) ne mogu biti ulazno/izlazni parametri.
OUT	Rezultujuću vrednost iz potprograma vraća na lokaciju određenog parametra.
TEMP	Bilo koja lokalna memorija koja nije korišćena za prosleđivanje parametara može biti upotrebljena za privremeno skladištenje unutar potprograma.



■ Osnove programiranja PLC-a

Ladder logic programiranje – osnovni elementi i operacije.

Floating point matematika

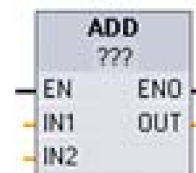
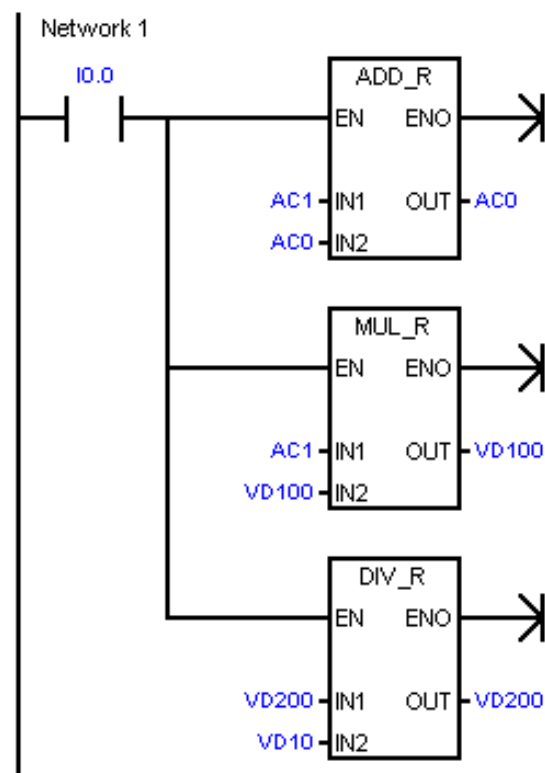
$IN1 + IN2 = OUT$

$IN1 - IN2 = OUT$
(nije prikazano)

$IN1 * IN2 = OUT$

$IN1 / IN2 = OUT$

Itd..



Pored osnovnih matematičkih operacija, nad varijablama se mogu izvršiti i MOD (računanje modula), dizanje na stepen, itd.

