

# SELECONTROL softver

# CAP1131

## Implementacija standarda IEC 1131

**K**ao što je u tekstu iz prethodnog broja rečeno, IEC 1131 je međunarodni standard koji definiše osnovne postavke za sva buduća rešenja iz oblasti automatizacije. Standard čine pet delova, kojih ćemo se ovom prilikom ukratko podsetiti:

- Prvi deo:** opšti pregled koji definiše osnovne pojmove iz oblasti automatizacije
- Drugi deo:** hardver - interna hardverska struktura kontrolerskog sistema i njegovog okruženja
- Treći deo:** definicije sintakse i semantike korišćenih programskih jezika
- Četvrti deo:** korisničke informacije - uputstva za korišćenje opreme, za programiranje, itd.
- Peti deo:** definicije internih i eksternih komunikacija, odnosno bus sistema

Postojeći sistemi automatizacije, iako sastavljeni od standardnih elemenata: PLC, komunikaciona magistrala (bus), aplikativni softver, u velikoj meri se razlikuju jedni od drugih - naime svaki proizvođač je promovisao neko svoje hardversko rešenje, softver, te se za korisnika koji bi prelazio sa jednog sistema na drugi javljao problem adaptacije na novo okruženje. Iz uporednog pregleda koji sledi jasno je zašto je bilo neophodno uvesti neka opšta pravila u oblasti automatizacije, odnosno uspostaviti standarde:

Nestandardizovana rešenja:	Standardizovana rešenja:
Različita hardverska okruženja	Strukturalno programiranje
Različiti programski jezici (svaki proizvođač ima svoj softver)	Portabilni softver
Glomazni i složeni projekti	Lako uvođenje "slabih" tačaka u projektu
Obuka i odgođavanje - šta se desi kada programer napusti firmu	

**Softverski model standarda IEC 1131 - programski jezici** Treći deo standarda IEC 1131 specificira sintaksu i semantiku programskih jezika. Postoje sledeće grupe programskih jezika:

- Tekstualni:** Instruction List (IL)  
Structured Text (ST)
- Grafički:** Functional Block Diagram (FBD)  
Ladder diagram (LD)  
Sequential Function Chart (SFC)  
Instrukcijska lista (IL)

IL je linijski orijentisan jezik, što znači da je jedna instrukcija PLC-u sadržana u jednoj liniji (slično assembleru). Jedna instrukcija IL je sastavljena od sledećih elemenata:

**labela: Operator Operand (\* Komentar\*)**

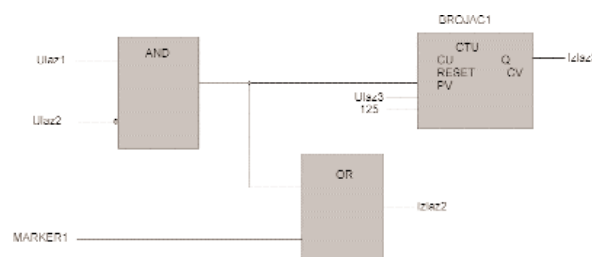
Korak1: LD brojac NE 0 JMPK  
Korak2: .....

Rezultat svake operacije (instrukcije) se odmah smešta u **akumulator**, te on sadrži aktuelni rezultat. Akumulator je memorijska lokacija čija se širina dinamički menja, zavisno od tipa promenljive (Boolean, Integer, Real ...) koja je smeštena u njemu

- LD Ulaz1 (\* Unesi bulovu promenljivu - dig. ulaz u akumulator \*)
- AND Ulaz2 (\* Akumulatorska vrednost AND stanje na dig. ulazu broj 2, rezultat smesti u akumulator \*)
- OR Ulaz3 (\* Akumulatorska vrednost OR stanje na dig. ulazu broj 3, rezultat smesti u akumulator \*)
- ST Izlaz1 (\* Vrednost akumulatora pošalji na digitalni izlaz \*)

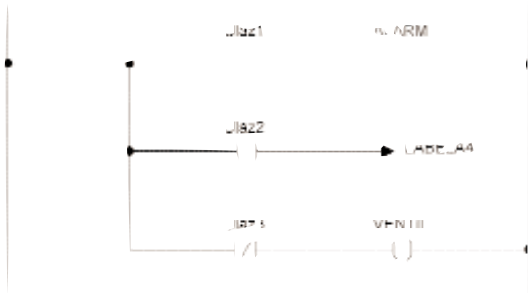
Promenljive Ulaz1, Ulaz2, Ulaz3 i Izlaz1 su tipa Boolean, a odgovaraju fizičkim adresama digitalnih ulaza i izlaza. O načinu pridruženja simboličkih imena fizičkim adresama biće više reči kasnije u tekstu.

**Funkcijski blok dijagram (FBD)** FBD je programski jezik koji po svojoj strukturi odgovara logici obrade signala u elektronskim kolima. Funkcijski blokovi su grafički elementi sa određenim brojem ulaza i izlaza, nalaze se u bibliotekama softvera, i mogu se povezivati sa ulazima i izlazima drugih funkcionalnih blokova; takođe, na njihove ulaze mogu se dovoditi i nezavisne konstante i promenljive (konstante i promenljive koje ne zavise od drugih funkcionalnih blokova programa). Za lakše razumevanje primera, CTU je funkcijski blok koji odgovara "UP" brojaču, čiji su ulazi i izlazi:



**CU** - dovođenjem impulsa na ovaj ulaz inkrementuje se brojač  
**RESET** - pojavom signala na ovom ulazu resetuje se brojač  
**PV** - konstantna vrednost na ovom ulazu definiše Preset vrednost brojača  
**Q** - tokom brojanja na ovom izlazu se pojavljuje Integer vrednost trenutne vrednosti brojanja  
**Q** - promenljiva tipa Boolean, kada se pojavi na ovom izlazu, indicira stanje CV=PV

**Ladder dijagram (LD)** "Praistorijska" rešenja automatizacije (iako pouzdana, da ne zamere starije kolege in enjeri) bila su zasnovana na relejnoj logici - određena kombinacija kontakta relea aktivirala je određeni aktuator. LD je svoj "uzor" našao u

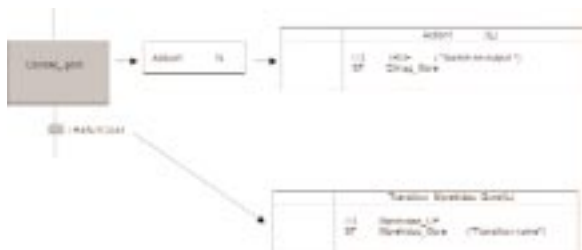


tom sistemu, i svojom strukturom opisuje tok energije kroz mre u softverskih "kontakta" i "kalema". Ovaj programski jezik radi samo sa promenljivama tipa Boolean, i u grafičkom prikazu postoje dve vertikalne linije koje označavaju linije "napajanja" između kojih se nalaze određene kombinacije kontakta i kalema "relea". Leva vertikalna linija ima uvek vrednost TRUE. Promenljive Ulaz1 i Ulaz2 odgovaraju "radnim kontaktima", a Ulaz3 "mirnom kontaktu" - to su promenljive tipa Boolean, i mogu predstavljati digitalne ulaze, ili softverske markere (korisnik im definiše funkciju). Simboli za Alarm i Ventil označavaju izlazne promenljive u vidu kalema relea, i te promenljive mogu setovati digitalne izlaze, ili softverske flegove (zavisno čemu su pridru ene).

**Sekvencijalni funkcionalni dijagram (SFC)** SFC i nije programski jezik u klasičnom smislu tog pojma; to je metoda koja omogućava da se slo eni softverski projekat "razbije" na pregledne celine, od kojih svaka obavlja tačno određenu funkciju, i između kojih postoji kontrolisan tok informacija. Osnovni pojmovi SFC-a su:

**Step (korak)** Označava jednu funkcionalnu celinu softverskog projekta, sastavljenu od jedne ili više akcija. Na korisniku je da osmisli način na koji će da struktuiru kompletan projekat, koliko će koraka da ima, i koje će akcije pojedini koraci izvršavati

**Akcija** Jedan korak može biti izvršavati jednu ili više akcija, a svakoj od njih može biti pridru en jedan od "atributa": N - po napuštanju obrade koraka akcija koja je u njemu definisana nije više aktivna S - akcija definisana u određenom koraku ostaje aktivna i po "napuštanju" obrade tog koraka R - akcija definisana u nekom od prethodnih koraka se, na početku obrade novog koraka sa ovako označenom akcijom, prekida P - akcija se aktivira samo u momentu početka obrade određenog koraka, ili po napuštanju obrade.



**Tranzicija** Prelazak sa jednog koraka na drugi, odnosno napuštanje obrade jednog koraka i početak obrade sledećeg uslovljen je ispunjenjem uslova tranzicije.

I akcija i tranzicija se programiraju u jednom od pomenutih programskih jezika (IL, ST, FBD, LD). Treba podsetiti da se softverski taskovi izvršavaju ciklično (vidi članak iz prethodnog broja), i u svakom ciklusu se izvršava **samo jedan korak**, odnosno akcije definisane u njemu. Kada se ispuni uslov tranzicije prelazi se na sledeći korak, te se u narednim ciklusima samo on obrađuje (dok se na ispuni sledeći uslov tranzicije), itd.

**Način definisanja i deklarisanja promenljivih**

Promenljive koje koristi program mogu biti definisane direktno, preko svoje fizičke adrese, ili posredno, kao simbolička imena. Ako se promenljiva predstavlja direktno, koristi se isti način označavanja koji koristi i sam softver CAP1131; naime, kada korisnik definiše hardverske resurse koje će njegov program koristiti (CPU, I/O module, itd.), tzv. **hardverski konfigurator** (koji je ugrađen u CAP1131) automatski uspostavlja pomenute direktne oznake ulaza, izlaza, internih registara CPU-a (markera).

U nastavku su dati primeri kako hardverski konfigurator definiše oznake za neke promenljive:

- %QX1.1.0.7** digitalni (X) izlaz (Q) na CAN magistrali (kanal 1), na čvoru 1 CAN-a, na modulu 0 čvora, kanal 7 modula
- %IB1.2.1.0** port (B) digitalnih ulaza (I) koji se nalazi na CAN magistrali (kanal 1), na čvoru 2 CAN-a, na modulu 1 čvora, port 0 (kanali 0 - 7)
- %MD0.12.0** sistemski (0) marker-interni registar CPU-a (M) veličine Double Word, obuhvata bitove na adresi 12, i to bitove 12.0 do 12.31

Još jednom treba podsetiti da ove oznake CAP1131 automatski formira, i da su na raspolaganju korisniku tokom programiranja. Međutim, za programera je prikladnije da sve promenljive (uključujući i digitalne/analogne ulaze ili izlaze) označava nekim svojim nazivima, koji će njegov program činiti preglednijim i čitljivijim. Pridru ivanje naziva promenljivama vrši se u tzv. zaglavlju programa (headers), a način na koji se to radi biće demonstriran u praktičnom primeru koji sledi. Osim definisanja, svaku korišćenu promenljivu treba i deklarirati, pomoću jedne od sledećih CAP1131 oznaka:

- VAR** Označava promenljivu kao lokalnu
- VAR\_GLOBAL** Označava promenljivu kao globalnu
- VAR\_EXTERNAL** deklarise promenljivu koja predstavlja ulazni ili izlazni kanal, ili promenljivu koja je već negde deklarirana kao globalna
- VAR\_INPUT, VAR\_OUTPUT** Kada korisnik kreira sopstvene funkcije ili funkcionalne blokove (podprogramme), pomoću ovih specifikatora deklarise promenljivu kao ulazni argument ili izlaznu vrednost funkcije
- VAR\_RETAIN** Označava lokalnu promenljivu koja ne gubi svoju vrednost u slučaju nestanka napajanja CPU-a
- VAR\_GLOBAL\_RETAIN** Označava globalnu promenljivu koja ne gubi svoju vrednost u slučaju nestanka napajanja CPU-a

Mogućnost izbora svakog od pomenutih specifikatora, pregled raspoloživih ulaza i izlaza, itd. CAP1131 korisniku nudi interaktivno preko menija, čime ga oslobađa obaveze pamaćenja i pisanja po papiru. Na kraju ovog dela treba još napomenuti da Selecontrol PLC, za razliku od konvencionalnih PLC-a, ne ograničava broj korišćenih promenljivih programa; jedino ograničenje predstavlja količina raspoložive memorije PLC-a.

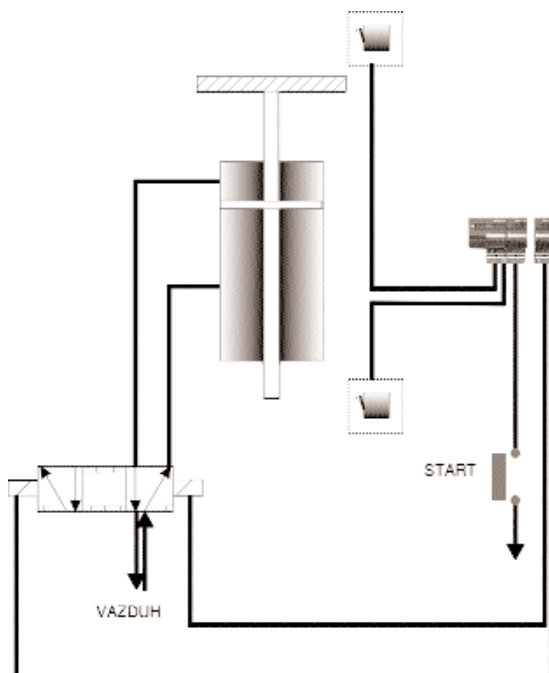
### Organizacija softverskog projekta u CAP1131 programu Selecontrol

CAP1131 program predstavlja, po svojim karakteristikama, implementaciju standarda IEC 1131-3, i to svih njegovih (već pomenutih) delova. Njegov interaktivni deo, **Project Navigator**, omogućuje korisniku definisanje kompletnog projekta, počev od hardverske konfiguracije, preko definisanja korišćenih promenljivih programa, definisanja softverskih modula projekta (POU - Program Organization Units), do definisanja tzv. taskova koji objedinjuju sve ove celine u jedan monolitan softverski projekat.

**Project Navigator** softvera Selecontrol CAP1131 omogućava:

- pregled aktuelne strukture softverskog projekta
- pregled svih elemenata definisanih u projektu
- direktan pristup svakom objektu

Sve pomenute mogućnosti biće demonstrirane na praktičnom primeru kontrole rada jednog elektropneumatskog sistema namenjenog za podizanje platformi. Taj elektropneumatski sistem čine: elektromagnetni pneumatski razvodnik 5/3 (normalno zatvoren), pneumatski cilindar dvosmernog dejstva, i PLC Selecontrol CPU 715. Principijelna šema ovog sistema je data na slici iznad.

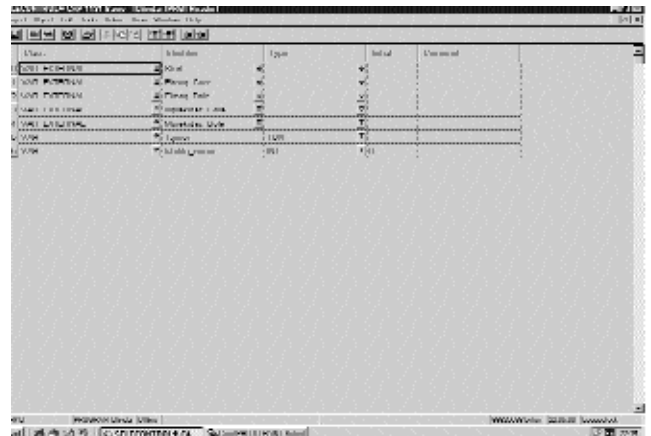


Pritisak na taster START aktivira elektromagnet razvodnika; ovaj zauzme prvo stabilno stanje pri kome se pušta vazduh u donju komoru cilindra; dok traje impuls na elektromagnetu, klip cilindra se pomera na gore, platforma se podiže. Kada se dostigne gornji položaj, aktivira se mikroprekidač, prekine se slanje impulsa na elektromagnet i startuje se vremenska pauza od 20 sekundi - naime toliko se platforma zadržava u gornjem položaju. Po

isteku tih 20 sekundi aktivira se drugi elektromagnet razvodnika, koji uslovljava da razvodnik zauzme drugo stabilno stanje, čime se vazduh pušta u gornju komoru cilindra, klip cilindra se pomera na dole, platforma se spušta.

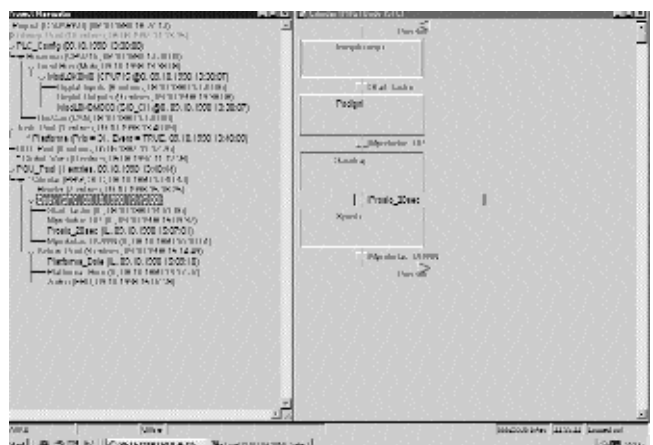
Programiranje CPU-a Selecontrol 715 se odvija na sledeći način:

U Project Navigator-u se definišu hardverski resursi (CPU714), naziv aplikativnog programa i vrsta jezika (u ovom slučaju to je Cilindar (SFC), zatim task kome pripada taj program (u ovom slučaju task Platforma), i definišu se promenljive programa u tzv. "zaglavlju" programa (POU Header)

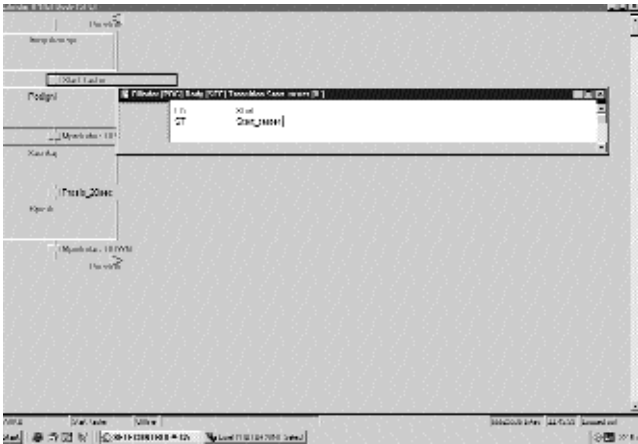


Može se uočiti da su sve promenljive koje odgovaraju ulazno/izlaznim kanalima deklarirane kao VAR\_EXTERNAL; takođe, promenljiva Tajmer je tzv. pseudonim za funkcionalni blok TON, koji ima funkciju merenja vremena od trenutka kada se pozove u programu, do pojave Boolean signala na ulazu IN

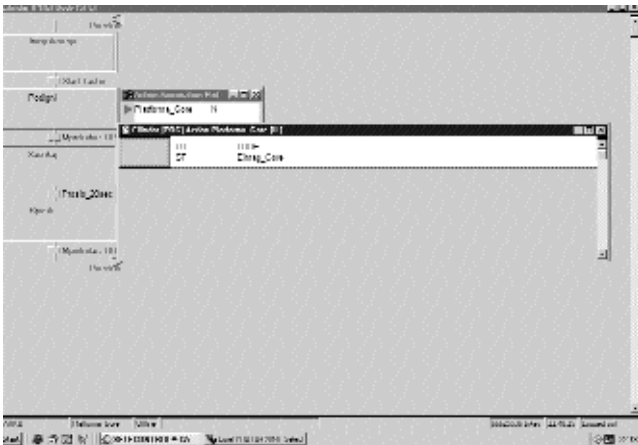
Iz Project Navigator-a se aktivira opcija Body programa, i otvara se prozor u kome se kreira naš program. Na slici koja sledi dat je prikaz već kreirane aplikacije, s tim što će svaki korak i tranzicija biti posebno objašnjeni



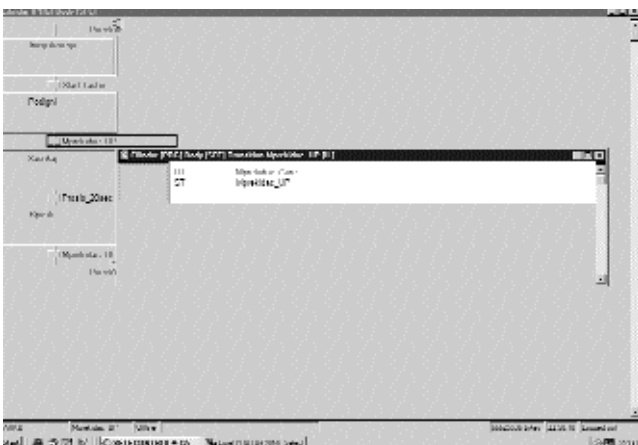
Početni korak, Inicijalizacija, je obavezan u svakom SFC programu; on služi i kao "wait" stanje dok se ne ispuni uslov prve tranzicije. Pritisak na taster START, koji predstavlja uslov prve tranzicije (Start\_taster) aktivira sledeći korak.



Vidi se kako je jednostavno programirati tranziciju: promenljiva Start, koja predstavlja logičko stanje na dig. ulazu, upisuje se u promenljivu koja ima isto ime kao i tranzicija (Start\_taster). Kada na tom dig. ulazu bude "1", uslov tranzicije biće ispunjen i biće moguć prelaz na korak Podigni.

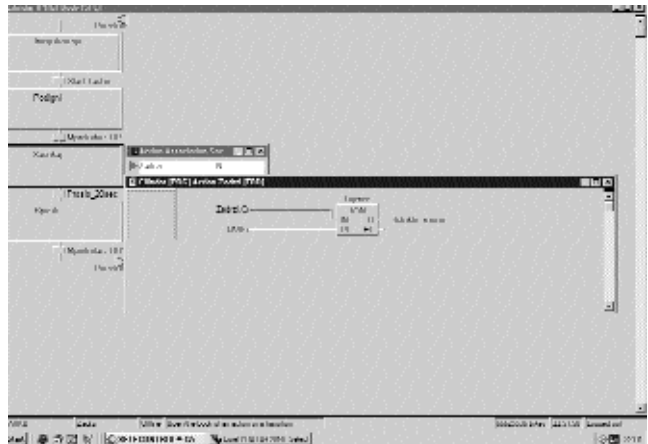


Tom koraku pridružena je akcija Platforma\_Gore, koja će se izvršavati samo dok korak Podigni bude aktivan (dok se ispunio uslov naredne tranzicije Mprekidac\_UP), o čemu govori atribut akcije N. Programiranje te akcije je jednostavno: konstanta TRUE (logička "1") se upiše u promenljivu Elmag\_Gore, koja (vidi sliku zaglavlja) odgovara dig. izlazu koji aktivira elektromagnet za podizanje cilindra. Ovaj korak, odnosno akcija koja je u njemu, će



se izvršavati u više uzastopnih ciklusa taska sve dok se ne ispunio uslov tranzicije Mprekidac\_UP, posle čega se prelazi na sledeći korak:

Logika programiranja ove tranzicije je slična prethodnoj tranziciji: promenljiva Mprekidac\_Gore, koja predstavlja stanje na dig. ulazu koga aktivira mikroprekidač gornjeg polo aja, upisuje se u promenljivu čiji naziv odgovara nazivu tranzicije Mprekidac\_UP. Sledeći korak (Sačekaj) sadrži akciju za zadržavanje cilindra u gornjem položaju (akcija Zadrži), i ta je akcija programirana u jeziku FBD-Funkcionalni Blok Dijagram;



Ovo je primer grafičkog programiranja: blok Tajmer (koji je pseudonim za bibliotetski blok-podprogram TON) ima sledeće ulazno-izlazne promenljive:

- IN na ovaj ulaz se dovede Boolean promenljiva Zadrzi.Q. Koja je to promenljiva, s obzirom da je nema u zaglavlju programa? Svaka akcija ima svoj "fleg", koji se setuje kada je akcija završena. Pomoću ove svojevrsne povratne veze obezbeđeno je da blok Tajmer sam sebe resetuje po isteku 20 sec
- PT na ovaj ulaz upiše se promenljiva tipa Time (tipovi su spomenuti u prethodnom broju)
- Q ovo je izlaz tipa Boolean, koji se setuje kada tajmer odmeri 20 sec.
- ET na ovom izlazu može se, u svakom trenutku, očitati proteklo vreme

Tranzicije i akcije koje su preostale u programu, a koje se odnose na fazu spuštanja platforme, su po svojoj logici slične prethodno opisanim, te ih čitaoci na osnovu opisanog algoritma rada elektropneumatske instalacije, zaglavlja programa i prethodnih objašnjenja mogu razumeti i sami. Namerna autora bila je da čitaoci upoznaju sa programiranjem u standardu IEC 1131, koji danas u svetu predstavlja opšte prihvaćeni model, te se, pored Selectron-a i mnogi drugi proizvođači opredeljuju za tu vrstu programiranja. Firma Delta Electronics, iz Trstenika, koja je zvanični predstavnik firme Selectron Lyss Ltd. za Jugoslaviju, stoji Vam na raspolaganju za sva pitanja iz ove oblasti, kao i iz oblasti procesne

**Kontakt adresa:**

**DELTA ELECTRONICS**

PO Box 00  
K. Milice 19 57240 Trstenik  
Tel: +381-31 / 113-484  
Fax: +381-31 / 113-484