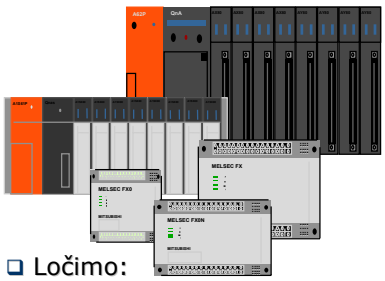
2. Programiranje logičnih krmilnikov/1



- Osnova programiranja PLK je programski jezik, precej podoben **zbirnemu** jeziku.
- Zaradi različnih pristopov in problemov, predvsem pri večjih aplikacijah ter različnih profilov programerjev so bili razviti še grafični programski jeziki (vmesniki).

- Ločimo:
  - **tekstovne programske jezike, ki so zelo podobni asemblerju (tekstovni urejevalniki)**
  - **grafične programske jezike (grafični urejevalniki)** (funkcije oz. ukaze vnašamo preko ekvivalentnih relejskih oz. kontaktnih shem ali funkcijskih bločnih shem)

shema se vedno prevede v tekstovno obliko (instructions)

2. Programiranje logičnih krmilnikov/2

2.1 Razlike v programiranju

- Med proizvajalci oz. ponudniki PLK-jev in programske opreme še vedno obstajajo razlike. Vsak proizvajalec ima svoj programski jezik.
- Različnost programiranja je še vedno prisotna, čeprav za to danes obstoji poseben standard, ki ureja in določa pravila pri programiranju.

Primer: Realiziraj naslednjo preklopno funkcijo: Izhod št. 0 naj se vključi, če je prižgan vhod št. 1 ali 2, pri tem pa vhod št. 3 ne sme biti prižgan.

SIEMENS (SIMATIC)		MITSUBISHI	IEC IL (STANDARDNI JEZIK)	
LD	*I0.1*	*LD X1*	*LD*	*%IX1*
O	*I0.2*	*OR X2*	*OR*	*%IX2*
AN	*I0.3*	*ANI X3*	*ANDN*	*%IX3*
=	*Q0.0*	*OUT Y0*	*ST*	*%QX0*

2. Programiranje logičnih krmilnikov/3

2.2 Standard IEC 1131 (International Electrotechnical Commission)

- Standard združuje številna določila v zvezi s PLK.
- Cilj:
  - poenotenje delovanja, uporabe, programiranja PLK in dokumentiranja
  - prenosljivost programov (žal to še vedno ni doseženo!!)
  - povezljivost v komunikacijska omrežja (še vedno težava)

2. Programiranje logičnih krmilnikov/4

2.2 Standard IEC 1131-3/1

- Standard ureja določila v zvezi s programiranjem PLK, programske jezike, tipe podatkov in vrste spremenljivk.
- Dejansko lahko izbiramo med štirimi programskimi jeziki:
  - dvema tekstovnima jezikoma (tekstovni urejevalnik):
    - IL (instruction list – seznam ukazov)
    - ST (structured text - strukturiran tekst)
  - dvem grafičnima jezikoma (grafični urejevalnik):
    - LD (ladder diagram - lestvični diagram),
    - FBD (function block diagram - funkcijski blokovni diagram)

2. Programiranje logičnih krmilnikov/5

2.2 Standard IEC 1131-3/2

- Strukturno programsko orodje (dodaten programski jezik, ki se uveljavlja, vendar še ni povsem standardiziran):
  - **SFC – sequential function chart (sekvenčni funkcijski diagram)**; programski jezik je namenjen strukturiranju opravil in programskih modulov (diagram poteka pri koračnih krmiljih, koračna veriga)

- Danes skorajda vsi proizvajalci (odvisno sicer od tipa krmilnika) omogočajo poleg svojih programskih jezikov tudi programiranje s standardnimi IEC-programskimi jeziki

2. Programiranje logičnih krmilnikov/6

2.2 Standard IEC 1131-3/3

- Podatkovni tipi in spremenljivke:
  - tipi podatkov
    - bitni:
      - Bool (0 ali 1)
    - numerični:
      - BYTE – dolžina 8 bitov (obseg števil: nepredznačeni 0 do 255, predznačeni –128 do +127)
      - WORD – dolžina 16 bitov (celoštevilski, obseg števil: 0 do 65535)
      - Integer (INT) – dolžina 16 bitov (predznačeni, obseg števil: –32768 do +32767)
      - Double Word (DWORD) – dolžina 32 bitov (celoštevilski, obseg števil: 0 do $2^{32}-1$)

2. Programiranje logičnih krmilnikov/7

2.2 Standard IEC 1131-3/4

- Podatkovni tipi in spremenljivke:
  - numerični
    - Double integer (DINT) – dolžina 32 bitov (predznačeni, obseg števil -2^{31} do $2^{31}-1$)
    - REAL – (floating point – zapis s plavajočo vejico, običajno 32-bitni)
  - ostali, kompleksni podatkovni tipi:
    - števci (CTU, CTD, CTUD)
    - časovniki (TON, TOF, TP)
    - FF (RS, SR)
- spremenljivke: notranje ali zunanje, globalne ali lokalne, konstante

2. Programiranje logičnih krmilnikov/8

2.3 Programski jeziki/1

2.3.1 Seznam ukazov (IL-instruction list)

- osnovni programski jezik, podoben zbirniku
- sintakso določa standard (*programski jeziki pri posameznih proizvajalcih še vedno odstopajo od standarda*)
- primernejši za izkušene programerje
- omogoča rešitev nekaterih kompleksnejših problemov, ki jih s pomočjo grafičnih programskih jezikov rešimo mnogo težje ali pa sploh ne
- omogoča uporabo simboličnih imen in komentarjev:

```
LD    %IX1          LD    VHOD1   (* Naloži VHOD1 *)
OR    %IX2          OR    VHOD2
ANDN  %IX3          ANDN  VHOD3
ST    %QX0          ST    IZHOD0  (* Zapiše na IZHOD0 *)
```

2. Programiranje logičnih krmilnikov/9

2.3 Programski jeziki/2

2.3.2 Strukturiran tekst (ST-structured text)

- višjenivojski programski jezik (le nekateri PLK-ji)
- sintaksa je podobna Pascalu:

```
%QX0:= (%IX1 OR %IX2) AND NOT %IX3;
```

- deklaracija spremenljivk:

```
VAR VHOD1: INT
```

- omogoča zapisovanje sestavljenih stavkov:

```
IF ... THEN ... ELSE ... END_IF;
FOR ... DO ... END_FOR;
WHILE ... DO ... END_WHILE;
```


2. Programiranje logičnih krmilnikov/10

2.3 Programski jeziki/3

2.3.3 Lestvični diagram (LD-ladder diagram)/1

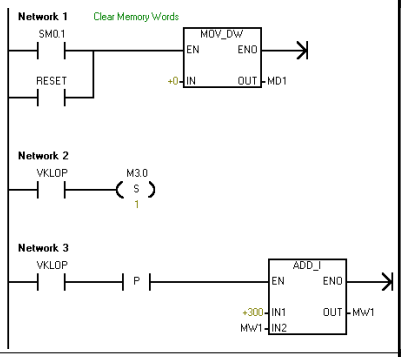
- jezik je primeren za začetnike in je najbolj priljubljen med neprofesionalnimi programerji
- simbolična in pregledna predstavitev vezja relejskih krmilij (osnova so relejske sheme ali krmilni načrti, zamenjava fizičnih simbolov s simboli programskih elementov)
- gradniki:
  - kontakti (vhodi: stikala, tipke, gumbi..., notranji vhodni pogoji)
  - tuljave (izhodi: luči, motorji, releji..., notranji pogoji)
  - funkcijski bloki (časovniki, števci, primerjalniki, računske in ostale funkcije)

2. Programiranje logičnih krmilnikov/11

2.3 Programski jeziki/4

2.3.3 Lestvični diagram (LD-ladder diagram)/2

- simbole zlagamo v "prečke"
- prečke so levo in desno omejene z napajalnima vodnikoma (navpični črti, leva predstavlja vir napajanja, desna ponor in se praviloma opušča)
- ni primeren za kompleksne naloge



2. Programiranje logičnih krmilnikov/12

2.3 Programski jeziki/5

2.3.3 Lestvični diagram (LD-ladder diagram)/3

Statični kontakti		
Št.	Simbol	Opis
1	*** --| |--	**Normalno odprt oz. delovni kontakt**
2	*** --|/|--	**Normalno zaprt oz. mirovni kontakt**

Trenutno delujoče tuljave		
St.	Simbol	Opis
1	*** --( )--	**Tuljava**
2	*** --(/)--	**Negirana oz. inverzno delujoča tuljava**

V (ST) pomeni:

```
IF VHOD THEN
   IZHOD:=1
ELSE
   IZHOD:=0
END_IF;
```

2. Programiranje logičnih krmilnikov/13

2.3 Programski jeziki/6

2.3.3 Lestvični diagram (LD-ladder diagram)/4

Tuljave z zadržanim delovanjem		
St.	Simbol	Opis
1	*** --(S)--	**Tuljava SET**
2	*** --(R)--	**Tuljava RESET**

V (ST) za S pomeni:

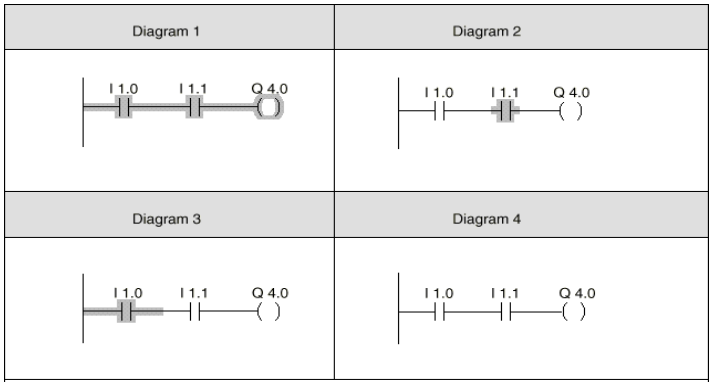
```
IF VHOD THEN
   IZHOD:=1
END_IF;
```


2. Programiranje logičnih krmilnikov/14

2.3 Programski jeziki/7

2.3.3 Lestvični diagram (LD-ladder diagram)/5

Zaporedna vezava, IN-funkcija

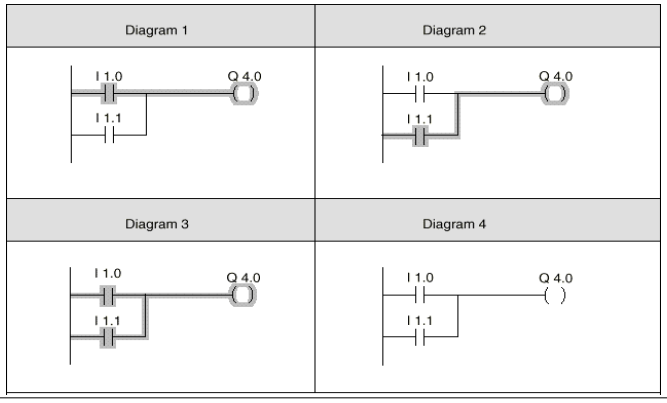


2. Programiranje logičnih krmilnikov/15

2.3 Programski jeziki/8

2.3.3 Lestvični diagram (LD-ladder diagram)/6

Vzporedna vezava, ALI-funkcija



2. Programiranje logičnih krmilnikov/16

2.3 Programski jeziki/9

2.3.4 Funkcijski blokovni diagram (FBD-function block diagram)/1

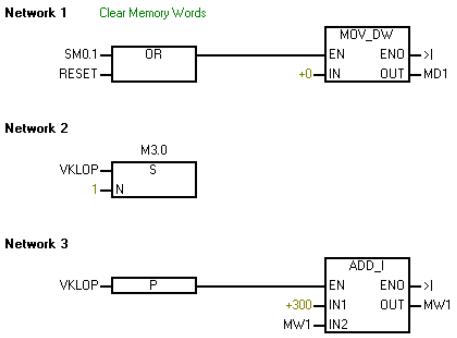
- krmilje predstavimo kot digitalno vezje (funkcijska shema)
- gradniki so funkcijski bloki (ni kontaktov in tuljav kot pri LD)
- vhodni signali vedno v blok vstopajo z leve
- izhodni signali vedno izstopajo iz bloka z desne

2. Programiranje logičnih krmilnikov/17

2.3 Programski jeziki/10

2.3.4 Funkcijski blokovni diagram (FBD-function block diagram)/2

- nazorno prikazuje delovanje programa oz. delovanje funkcijske sheme (vezja)



2. Programiranje logičnih krmilnikov/18

2.3 Programski jeziki/11

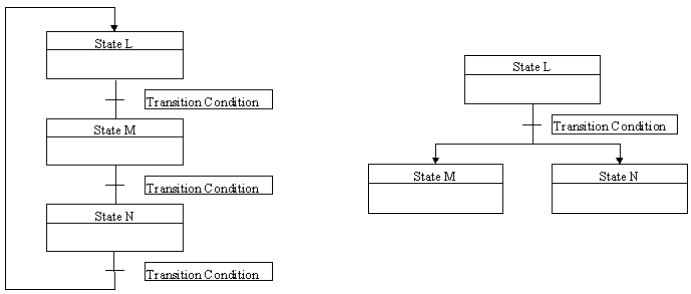
2.3.5 Sekvenčni funkcijski diagram (SFC-sequental function chart)/1

- jezik je namenjen programiranju koračnih krmilij, krmilje predstavimo v koračni verigi
- diagram predstavlja stanja sistema in prehode med njimi
- običajno ga uporabljamo v povezavi z ostalimi jeziki
- gradniki: elementi koračnega člena (korak, akcija, prehod, vrsta signala)

2. Programiranje logičnih krmilnikov/19

2.3 Programski jeziki/12

2.3.5 Sekvenčni funkcijski diagram (SFC-sequental function chart)/2



2. Programiranje logičnih krmilnikov/20

2.3.6 Naslavljanje oz. adresiranje

- **Sintaksa za IEC-naslavljanje:**
  - ⇨ **%** **Predpona za vsak IEC naslov**
  - ⇨ **I, Q, M** **I= Input, Q=Output, M=Memory**
  - ⇨ **X, W, D** **X=Bool, W=Word, D=Integer**
  - ⇨ **števec** **direktni naslov**
- **Posamezni proizvajalci imajo zaradi svojih programskih jezikov svoja pravila pri naslavljanju posameznih enot:**

Primer: adresiranje vhodne enote

Siemens:	Mitsubishi	IEC-naslov
I0.3	*X3*	*%IX3*

- **Območje naslovov in operandov je odvisno od tipa krmilnika.**