СПЕЦИАЛИЗИРАНИ КОМПЮТЪРНИ СИСТЕМИ

Програмиране на програмируеми логически контролери

Съдържание

- Програмиране на програмируеми логически контролери
  - Операционен цикъл
  - Езици за програмиране
  - Видове данни
  - Аритметически и логически инструкции, инструкции за преход
  - Инструкции върху числа с плаваща запетая
  - Инструкции за обработка на таблици
  - Контакти и бобини

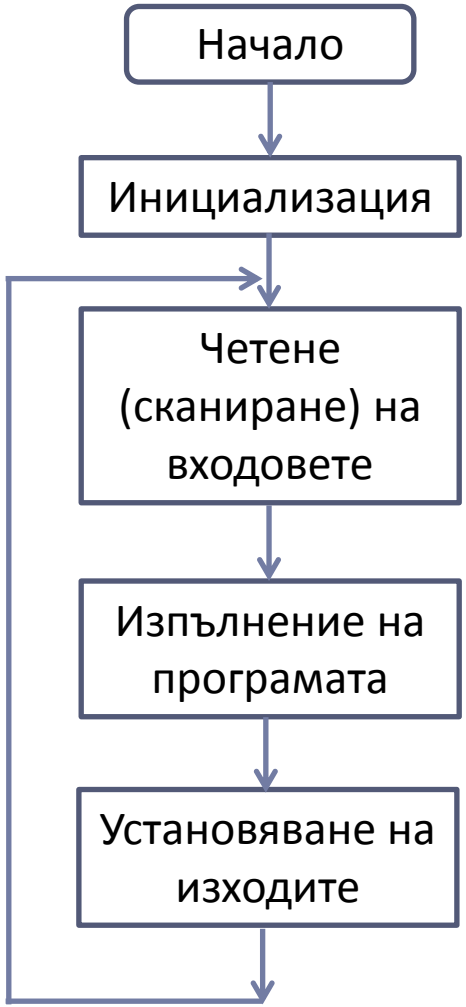
Операционен цикъл

- Работата на PLC се управлява от **операционна система (за реално време).**

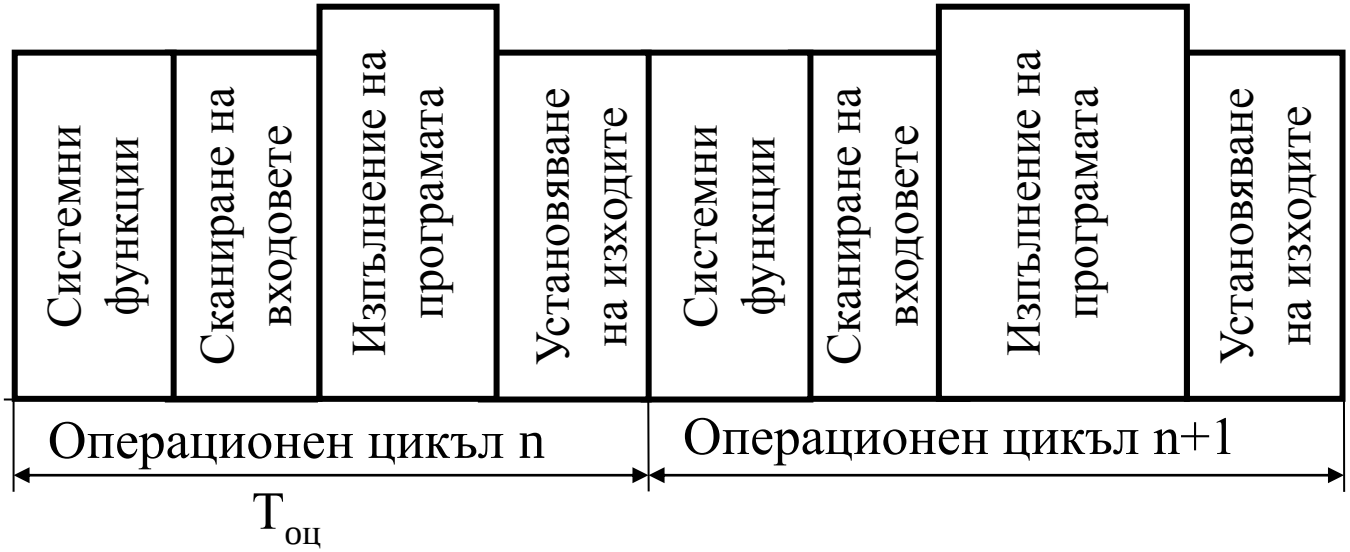
Тя осигурява изпълнението на следните процедури от CPU:

- сканиране състоянието на входовете и записване на това състояние в специално определена област на паметта;
- изпълнение на потребителската програма;
- записване състоянието на изходите в специално определена област на паметта и установяване на изходите в състояние, съответстващо на резултата от изпълнението на програмата;
- изпълнение на системни функции по контрол и управление на апаратни и програмно вградени функции и комуникация.

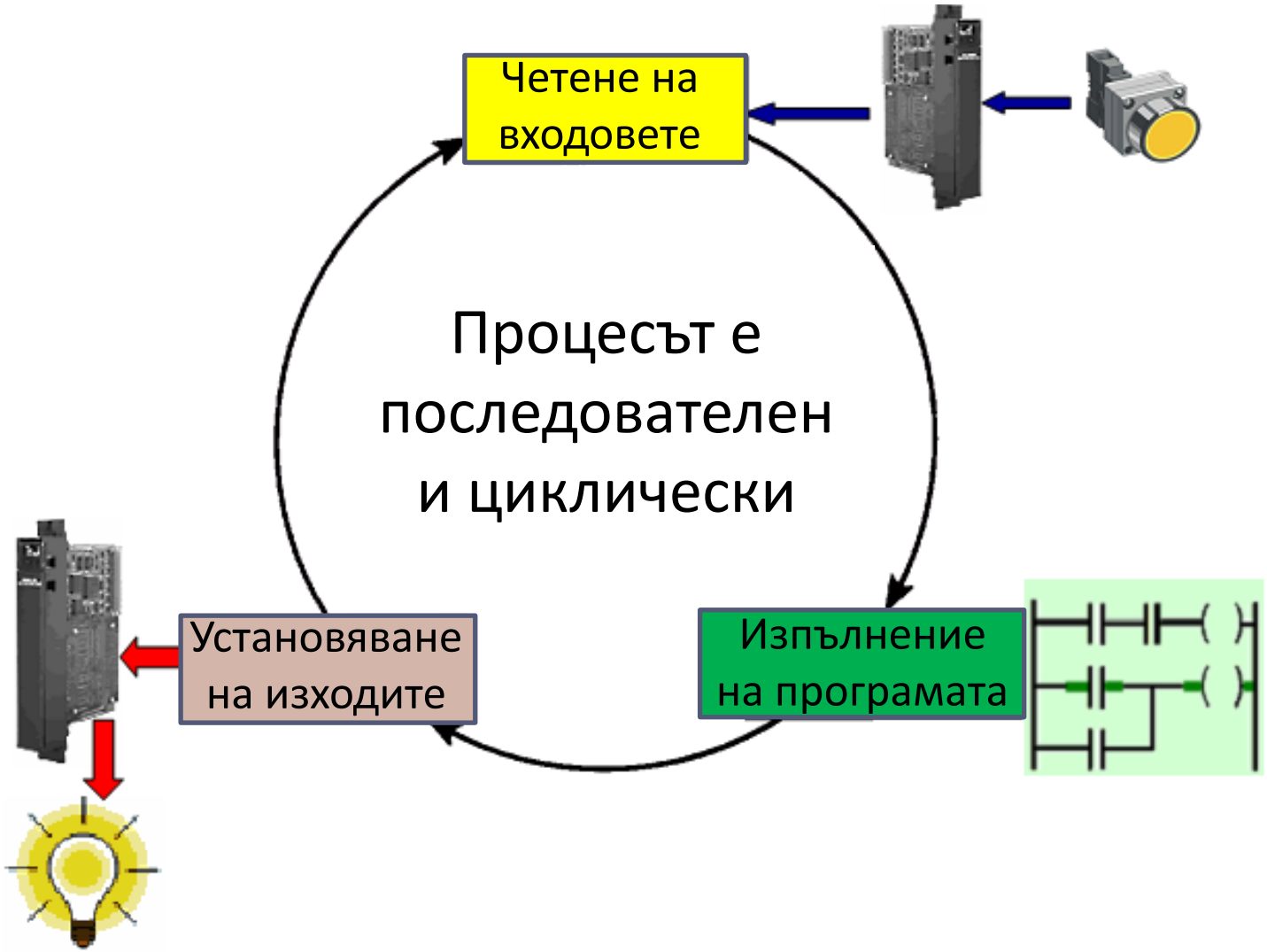
PLC – операционен цикъл



PLC – операционен цикъл



Процес на сканиране



Операционен цикъл

- **Продължителност на входните сигнали**

За да се регистрира промяната на входен сигнал, неговата продължителност трябва да бъде **по-голяма** от продължителността на операционния цикъл, т.е.

$$t_{вх.\ мин} = T_{оц} + T_{вх}$$

където:

$t_{вх.мин.}$ е минималната продължителност на входните сигнали;

$T_{оц}$ е продължителността на операционния цикъл;

$T_{вх}$ е продължителността на фазата за сканиране на входовете.

Операционен цикъл

- **Време за реакция**

Зависи от момента на промяна на входния сигнал спрямо фазите на операционния цикъл.

Максималното време, за което контролерът ще реагира на промяна на входен сигнал чрез съответната промяна на изходите в съответствие с програмата, е:

$$t_{реакц.макс.} = 2T_{оц} + T_{вх}$$

където:

$T_{оц}$ е продължителността на операционния цикъл;

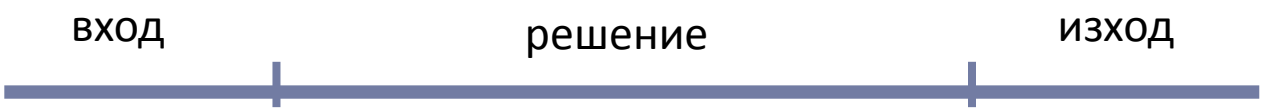
$T_{вх}$ е продължителността на фазата за сканиране на входовете.

Операционен цикъл

- Изпълнение на потребителската програма
  - ✓ Циклично

Може да има следните особености:

- **Първи вариант** – прочитат се всички входни сигнали в началото на цикъла, решават се логическите уравнения и в края на цикъла се активират необходимите изходи.

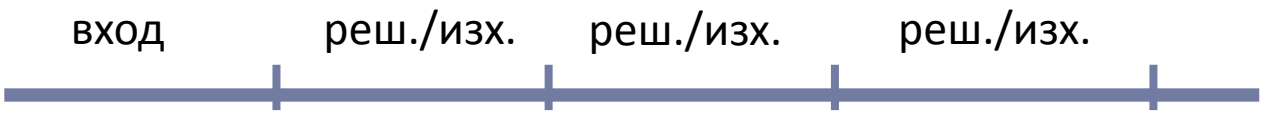


Операционен цикъл

- Изпълнение на потребителската програма
  - ✓ Циклично

Може да има следните особености:

- **Втори вариант** – прочитат се всички входни сигнали в началото на цикъла и след решаване на всяко логическо уравнение се активира съответният (необходимият) изход.

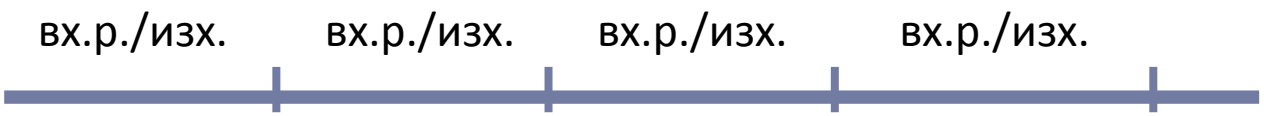


Операционен цикъл

- Изпълнение на потребителската програма
  - ✓ Циклично

Може да има следните особености:

- **Трети вариант** – входовете се четат на всеки n ms, независимо от продължителността на цикъла, а изходите се активират след решаване на поредното логическо уравнение.



Езици за програмиране

В практиката на използване на PLC най-голямо приложение са намерили следните езици за програмиране:

- Списък с инструкции (Instruction List - IL)
- Структуриран текст (Structured Text - ST)
- Релеен език (Ladder Diagram - LD)
- Функционална блокова диаграма (Function Block Diagram - FBD)

Езици за програмиране

А. Списъкът с инструкции (Instruction List - IL) е език от текстов тип и съдържа последователност от инструкции на отделен ред от програмата:

[етикет] оператор [операнд[,операнд…]] [коментари]

Пример:

…….

```
LD     %IX6.2      (* чете входен бит с адрес 6.2 *)
ST    BIT 1        (* записва резултата в BIT 1 *)
CAL FUNC 1         (* извиква функционален блок FUNC 1*)
```

……

Езици за програмиране

- **Б. Структуриран текст (Structured Text - ST)**

Основен елемент в ST е **изразът.**

Изразите представляват описания на операции върху операнди.

Операнди: променливи, литерали, функции или други изрази.

Типични операции: извикване на функция, степенуване, смяна на знака, инвертиране, умножение, делене, събиране, сравнение, конюнкция, дизюнкция, сума по модул 2 и др.

Езици за програмиране

- **В. Релеен език (Стълбовидна диаграма, Ladder Diagram - LD)** - графичен език

Елементи: променлива, нейно допълнение, начало и край на разклонение при паралелна верига, присвояване на резултата на изходна или междинна променлива.

“Захранващи" шини - съответстват на захранващите линии в оперативната верига на една релейна КС. Между тях се разполага мрежа от елементи, които са свързани както помежду си, така и със страничните шини. Тези елементи са:

Езици за програмиране

- **В. Релеен език (Стълбовидна диаграма, Ladder Diagram - LD)** - графичен език
  - *Връзки* - означават се с хоризонтални и вертикални линии. Състоянието на всяка връзка може да бъде "0" или "1", съответстващи на липса или наличие на напрежение.
  - *Контакти:* нормално отворени **-| |-** ; нормално затворени -**|/|**-; контакти, чувствителни по нарастващ (положителен) "0 → 1" или по падащ (отрицателен фронт) "1 → 0".

Езици за програмиране

- **В. Релеен език (Стълбовидна диаграма, Ladder Diagram - LD)** - графичен език
  - *Изходни елементи* -( )- съответстват на бобините на релетата или контакторите в релейните КС.

    Получават "захранване", само ако състоянието на елементите вляво от тях е "1".

    Изходните елементи са винаги двупозиционни и могат да бъдат: нормални; инверсни; с нулиране и запомняне; чувствителни по фронт и др.

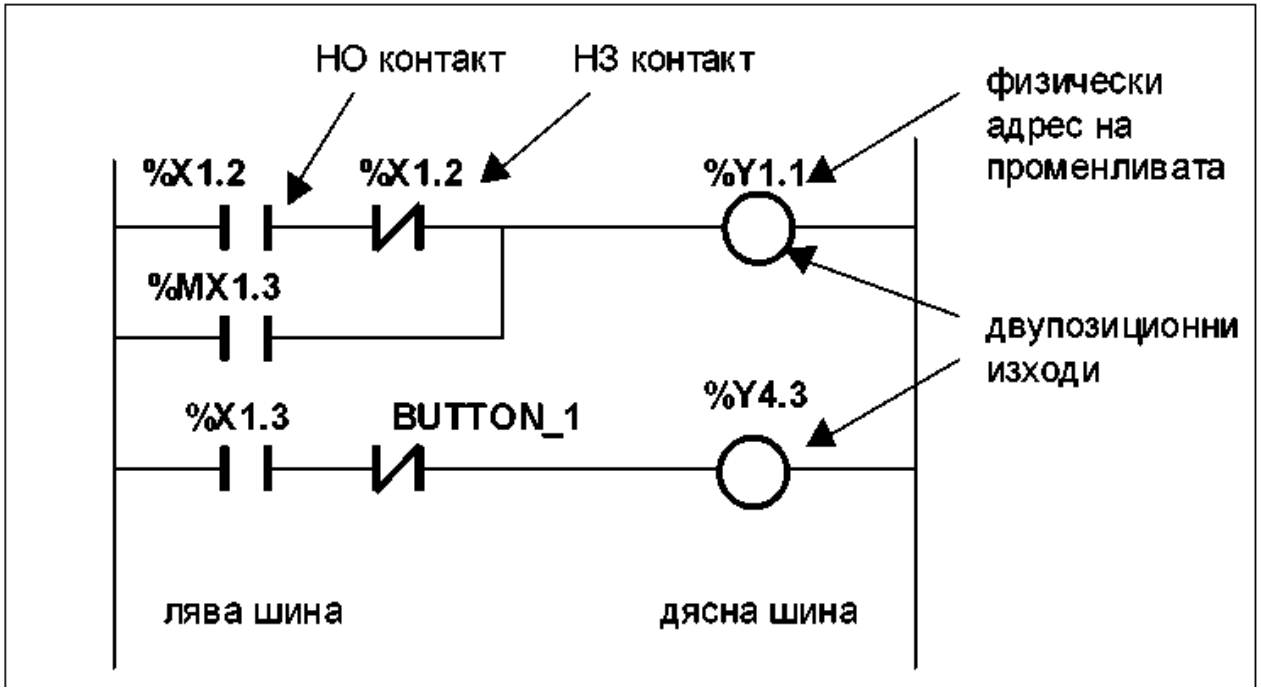
Езици за програмиране

- **В. Релеен език (Стълбовидна диаграма, Ladder Diagram - LD)** - графичен език
  - В LD програмите могат също да бъдат използвани функции и функционални блокове, по подобие на текстовите езици IL и ST.

Езици за програмиране

- **В. Релеен език (Стълбовидна диаграма, Ladder Diagram - LD)** - графичен език

Пример:



Езици за програмиране

- **Г. Функционално-блокова диаграма (Functional Block Diagram - FBD)**

Език за програмиране, основан на аналогия с алгоритъма за логическо управление на технологичния процес или с неговата схема на функциониране.

Функционалните блокове се означават с правоъгълници, при които входовете са отляво, а изходите - отдясно на символа.

Езици за програмиране

- **Фрагмент** от програма, написан на различни езици

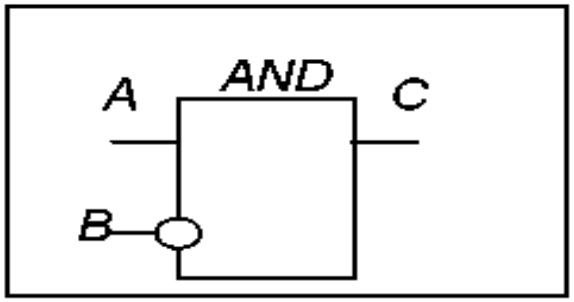
Instruction List (IL)

```
LD      A
ANDN    B
ST      C
```

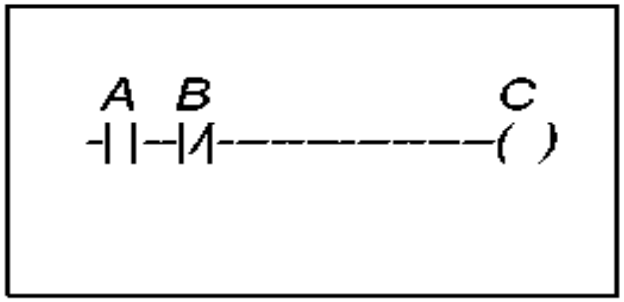
Structured Text (ST)

```
C:= A  AND  NOT B
```

Function Block Diagram (FBD)



Ladder Diagram (LD)



Обекти на езика за програмиране

- **Битови променливи**

Битовите променливи могат да приемат логическите стойности нула и единица.

Като битови променливи могат да се разглеждат:

- **цифровите входове** на контролера, означавани с ***%Ii.j*** (***i*** е номерът на модула на контролера, чийто вход се адресира, ***j*** е номерът на съответния вход);
- **цифровите изходи**, означавани с ***%Qi.j*** (***i*** и ***j*** имат същите значения както за входовете);
- **вътрешни** битови променливи - ***%Mi***;
- **вътрешни** битови **системни** променливи - ***%Si***;
- **битове от дума** - ***%SBRi.j;***
- **изходи на функционални блокове**, например - ***%TMi.Q***, ***%Ci.F***.

Обекти на езика за програмиране

- **Променливи от тип дума**

16-битови числа със знак (-32768 ÷ 32767) или без знак (0 ÷ 65535).

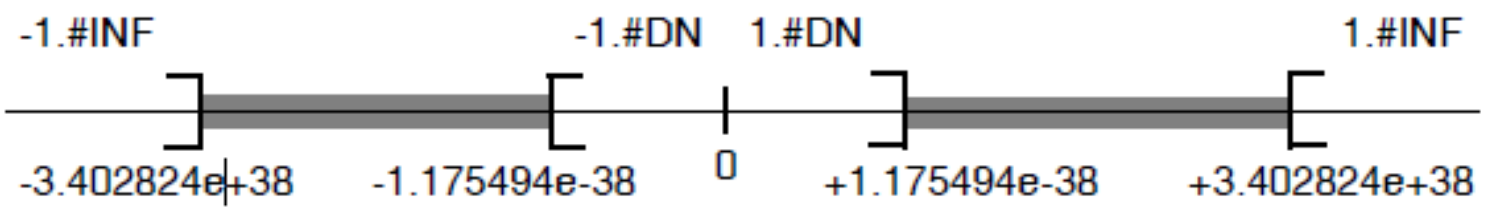
Примери за променливи от тип дума:

- Непосредствени стойности
- Вътрешни променливи (%MWi) (думи от паметта)
- Константи (%KWi)
- Думи за входно-изходен обмен (%IWi, %QWi%)
- Системни думи (%SWi)
- Конфигурационни или операционни думи от функционални блокове (%TM2.P, %Ci.P, ...)
- Стойности на аналогови входно-изходни сигнали (%IWAx.y.z, %QWAx.y.z)

Обекти на езика за програмиране

- **Променливи от тип двойна дума**
  - 32-битови числа със знак (-2147483648 ÷ +2147483647).
  - Числа с плаваща запетая

Bit 31	Bits {30...23}	Bits {22...0}
S	Exponent	Fractional part



Обекти на езика за програмиране

- **Числа с формат двойна дума**

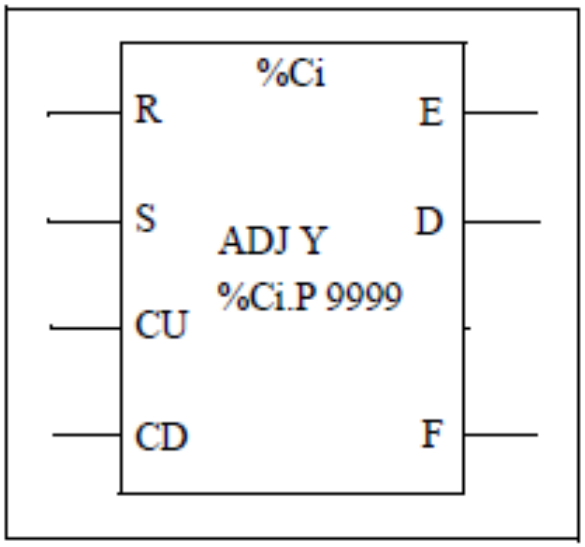
Примери:

- Непосредствени стойности
- Вътрешни променливи:
  - Числа с плаваща запетая - %MFi
  - Цели числа - %MDi
  - Константи - %KFi, %Kdi

Обекти на езика за програмиране

- **Функционални блокове**

Поддържат програмно достъпни битови обекти и специфични обекти от тип дума



Обекти на езика за програмиране

- **Структурирани обекти**

Комбинации от съседни, еднотипни обекти

Видове:

- Битови стрингове
- Таблици от думи
- Таблици от двойни думи
- Таблици от числа с плаваща запетая

Обекти на езика за програмиране

- **Структурирани обекти**
  - Битови стрингове

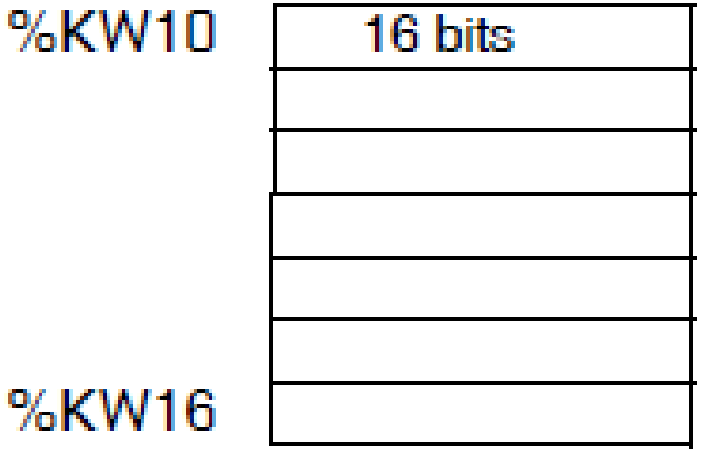
Пример: %M8:6

%M8	%M9	%M10	%M11	%M12	%M13

Обекти на езика за програмиране

- **Структурирани обекти**
  - Таблици от думи

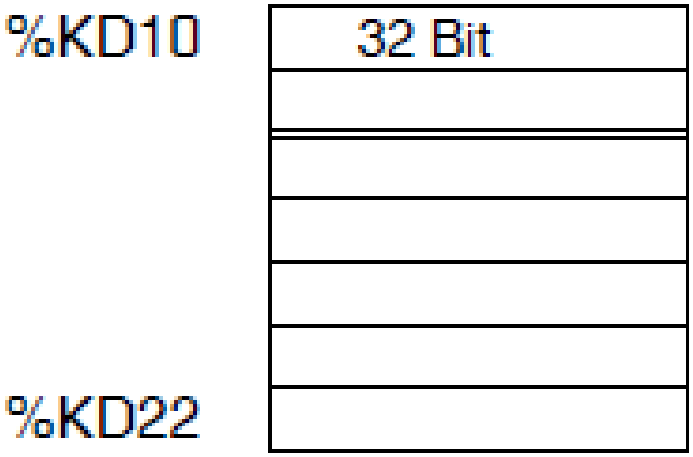
Пример: %KW10:7



Обекти на езика за програмиране

- **Структурирани обекти**
  - Таблици от двойни думи

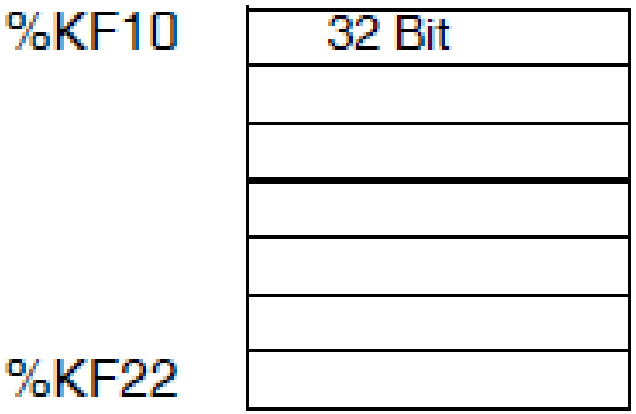
Пример: %KD10:7



Обекти на езика за програмиране

- **Структурирани обекти**
  - Таблици от числа с плаваща запетая

Пример: %KF10:7



Обекти на езика за програмиране

- **Индексирани обекти**

Индексирана дума е единична или двойна дума, или с плаваща запетая с индексиран обект.

Адресиране:

- Директно (пряко)
- Индексно

Пример: %MW108[%MW2] е дума с адрес, състоящ се от директен адрес108 плюс съдържанието на думата %MW2.

Ако думата %MW2 има съдържание 12, записът в %MW108[%MW2] е еквивалентен на записа в %MW120 (108 плюс 12).

Системни битове и системни думи

- **Системни битове (част от тях)**

%S0 – студен старт

Normally set to 0, it is set to 1 by:
- A power return with loss of data (battery fault),
- The user program or Animation Table Editor,
- Operations Display.

This bit is set to 1 during the first complete scan. It is reset to 0 by the system before the next scan.

%S1- топъл старт

Normally set to 0, it is set to 1 by:
- A power return with data backup,
- The user program or Animation Table Editor,
- Operations Display.

It is reset to 0 by the system at the end of the complete scan.

%S10 – входно-изходна грешка

Normally set to 1. This bit can be set to 0 by the system when an I/O fault is detected.

%S12 – ПЛК е в работен режим

This bit reflects the running state of the controller. The systems sets the bit to 1 when the controller is running. Or to 0 for stop, init, or any other state.

Системни битове и системни думи

Системни думи (част от тях)

%SW0 – период на сканиране

%SW6- състояние на контролера

Modifies controller scan period defined at configuration through the user program in the Animation Table Editor.

Controller Status:
0 = NO CONFIG
2 = STOPP
3 = RUN
4 = HALT

%SW30	Last scan time	Shows execution time of the last controller scan cycle (in ms). **Note:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle.
%SW31	Max scan time	Shows execution time of the longest controller scan cycle since the last cold start (in ms). **Note:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle.
%SW32	Min. scan time	Shows execution time of shortest controller scan cycle since the last cold start (in ms). **Note:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle.
%SW48	Number of events	Shows how many events have been executed since the last cold start. **Note:** Set to 0 (after application loading and cold start), increments on each event execution.

Системни битове и системни думи

- **Системни думи (част от тях)**

%SW49 %SW50 %SW51 %SW52 %SW53	Real-Time Clock (RTC)	RTC Functions: words containing current date and time values (in BCD): %SW49 — xN Day of the week (N=1 for Monday) %SW50 — 00SS Seconds %SW51 — HHMM Hour and minute %SW52 — MMDD Month and day %SW53 — CCYY Century and year These words are controlled by the system when bit **%S50** is at 0. These words can be written by the user program or by the terminal when bit **%S50** is at 1. On a falling edge of **%S50** the controller's internal RTC is updated from the values written in these words.

Основни операции

- **Контакти**
  - Контакти

- нормално отворен контакт
- нормално затворен контакт
- контакт за откриване на нарастващ фронт на входния сигнал
- контакт за откриване на падащ фронт на входния сигнал

Основни операции

- **Бобини**

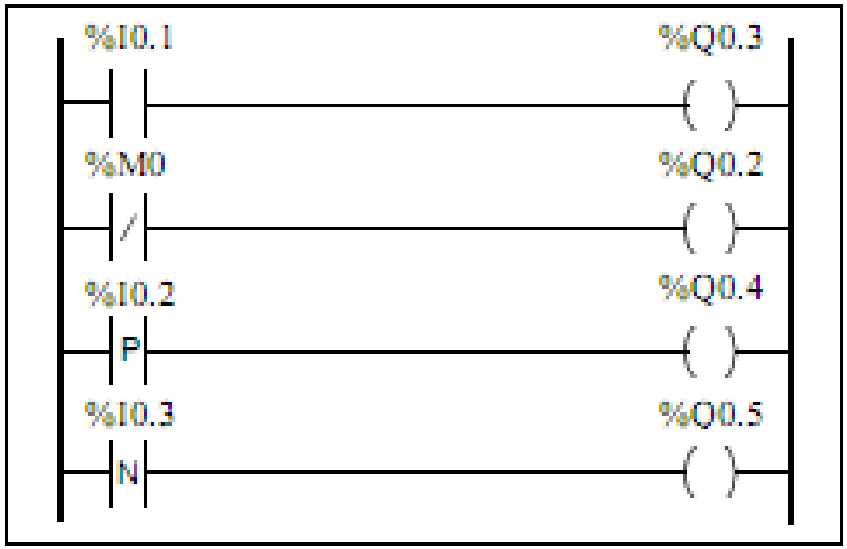
- установяване в активно ниво на посочения изход или битова променлива
- установяване в неактивно ниво на посочения изход или битова променлива
- трябва да бъде подаден активен сигнал на съответната променлива или изход и той да бъде запомнен
- на указания изход или битова променлива трябва да бъде установен неактивен сигнал (логическа нула) и той да бъде запомнен

Основни операции

- **Логически инструкции за зареждане и съхраняване**
  - Логическите инструкции за зареждане LD, LDN, LDR и LDF съответстват на отворен, затворен, регистриращ нарастващ фронт на сигнала и регистриращ падащ фронт на сигнала контакти.
  - Логическите инструкции за съхраняване са ST, STN, S и R.

Основни операции

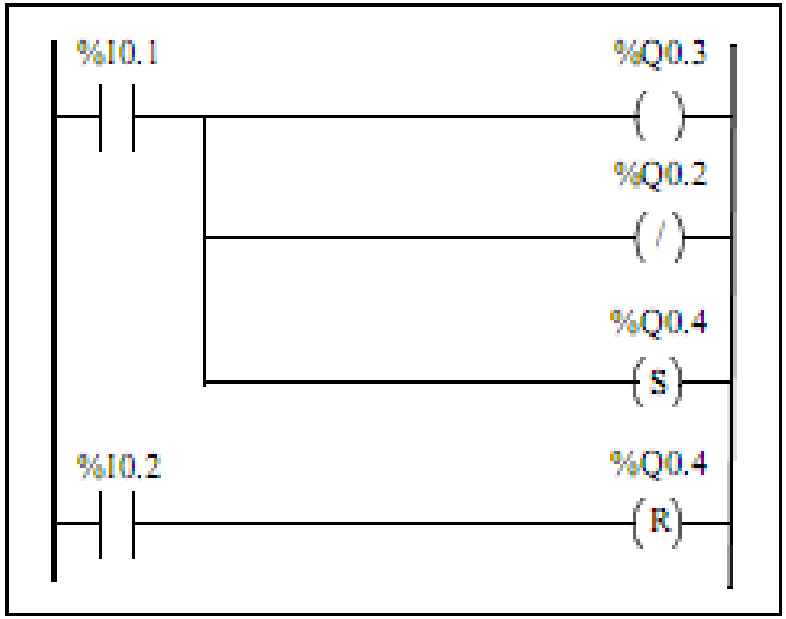
- **Съответствие между инструкции за зареждане и контакти**




```
LD    %I0.1
ST    %Q0.3
LDN   %M0
ST    %Q0.2
LDR   %I0.2
ST    %Q0.4
LDF   %I0.3
ST    %Q0.5
```

Основни операции

- **Съответствие между инструкции за съхраняване и бобини**



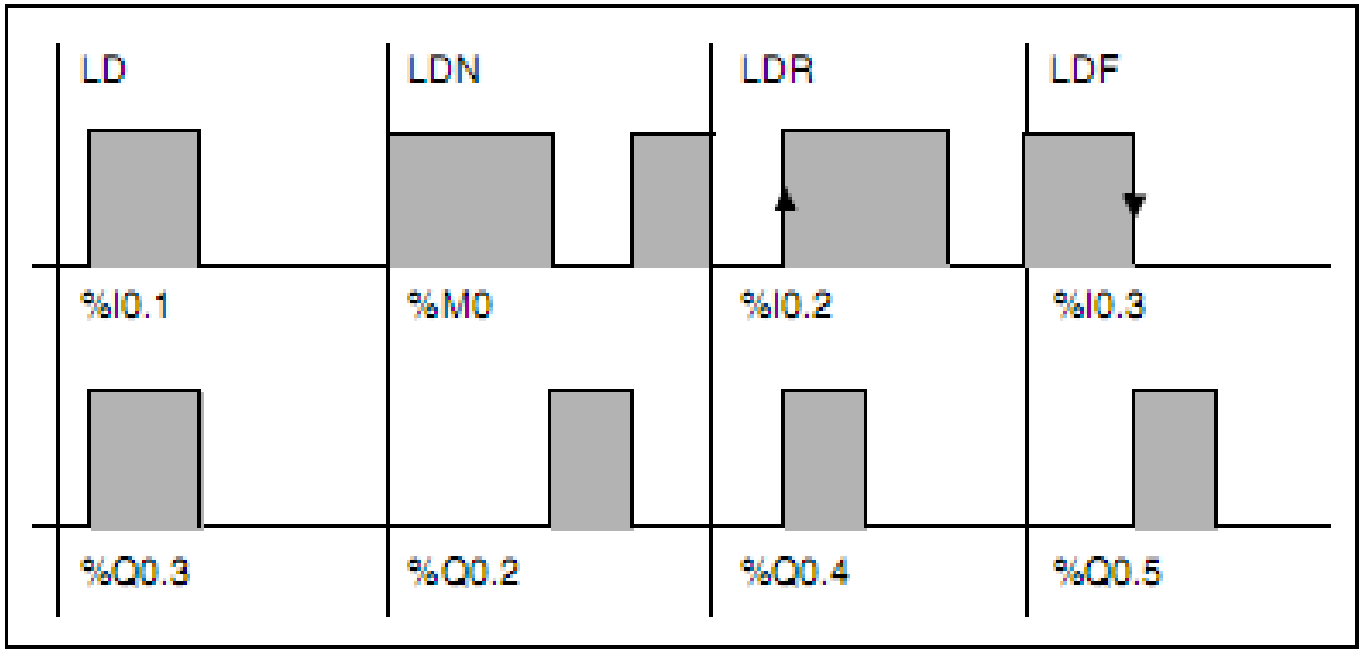

```
LD      %I0.1
ST      %Q0.3

STN     %Q0.2
S       %Q0.4

LD      %I0.2
R       %Q0.4
```

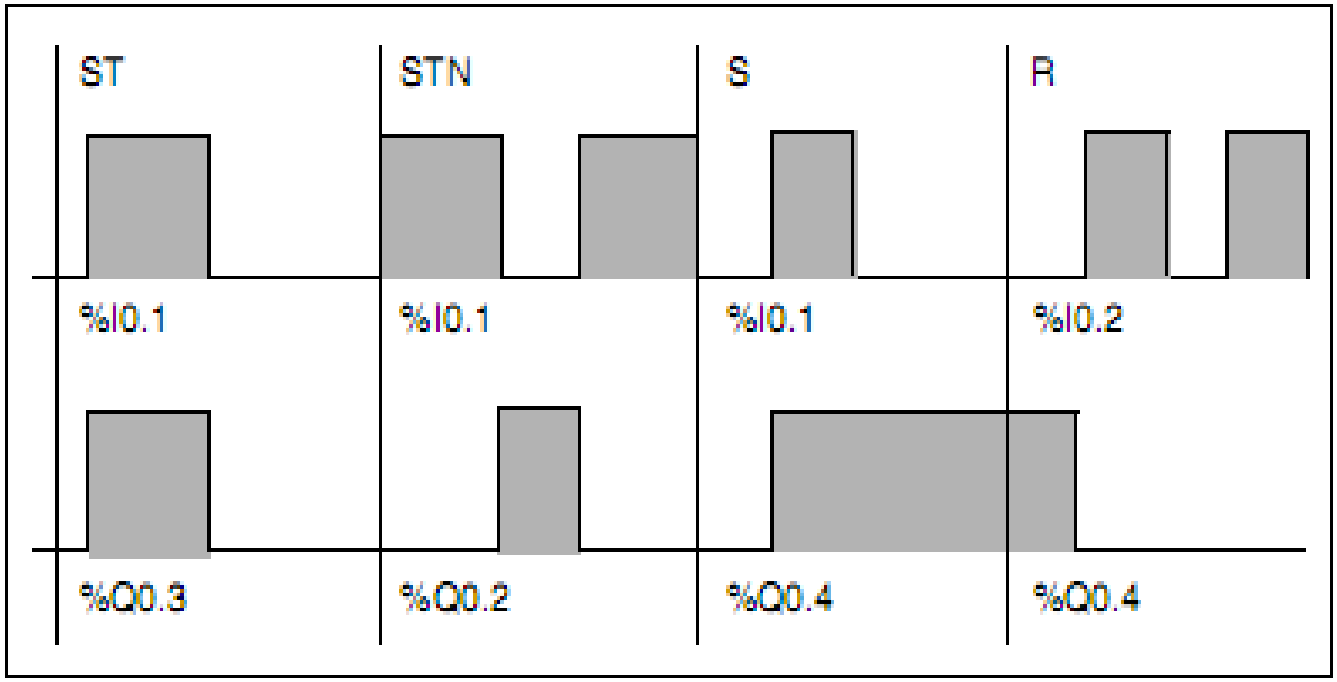
Основни операции

- **Времедиаграми на инструкции за зареждане**



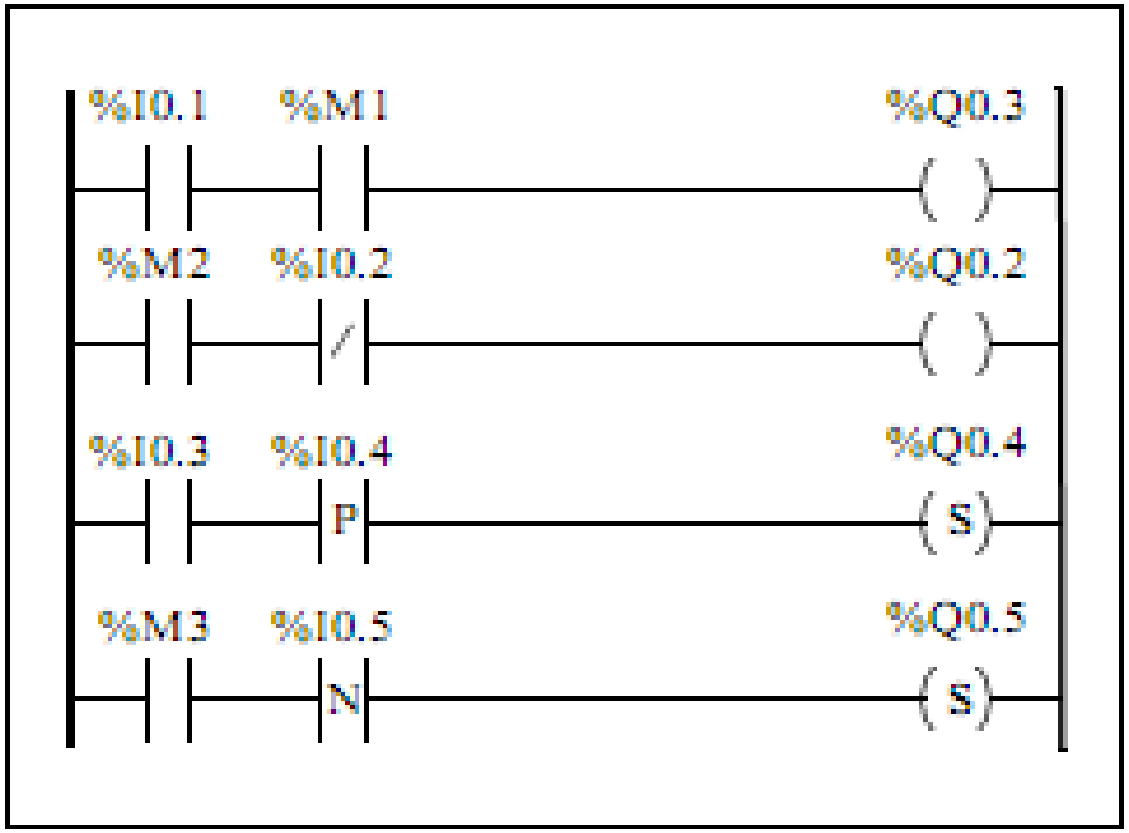
Основни операции

- **Времедиаграми на инструкции за съхраняване**



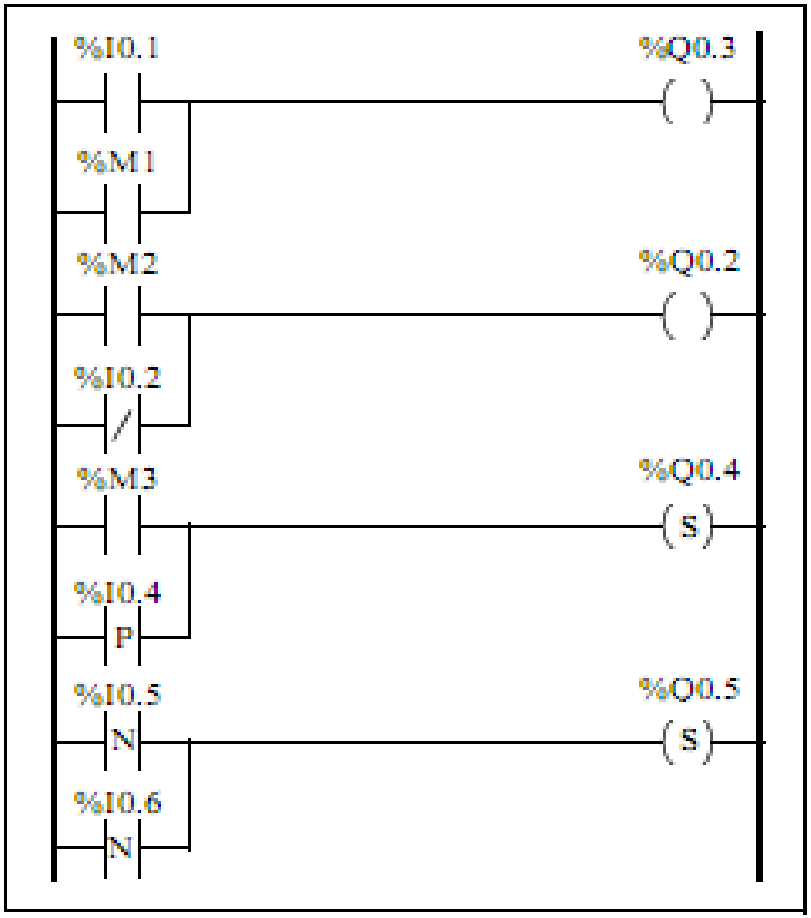
Основни операции

- Логически операции AND, ANDN, ANDR и ANDF



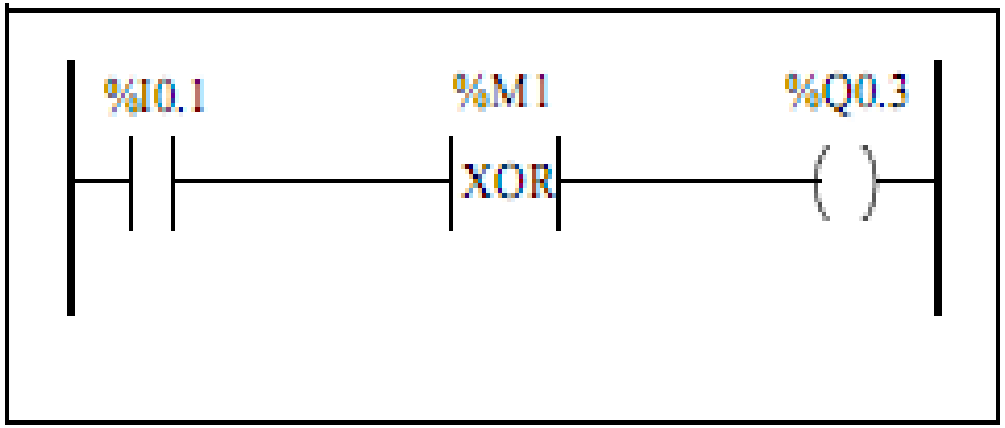
Основни операции

- Логически операции OR, NOR, ORR и ORF



Основни операции

- Логически операции XOR, XORN, XORR, XORF



Основни операции

- Специални инструкции:
  - OPEN - създава прекъсване в Ladder рънг, независимо от резултатите от последните логически операции;
  - SHORT - позволява да се премине през рънг, независимо от резултатите от последните логически операции.

Основни операции

- Инструкция за присвояване

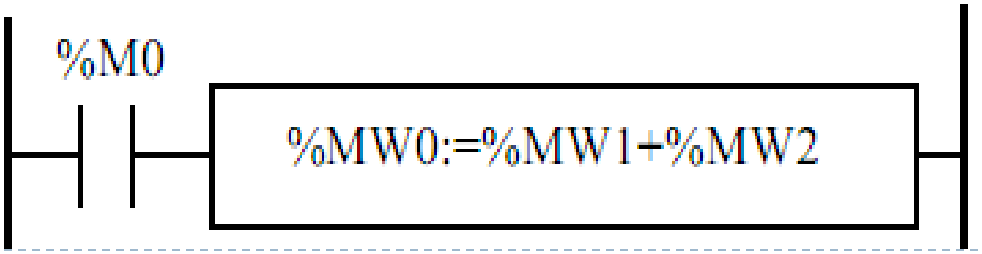
Op1 := Op2

Инструкцията се записва в операторен блок.

Основни операции

- Инструкции за аритметични операции

Инструкция/ Оператор	Функция	Синтаксис
+	Събиране на два операнда	Op1:=Op2 + Op3
-	Изваждане на два операнда	Op1:=Op2 - Op3
*	Умножение на два операнда	Op1:=Op2 * Op3
/	Деление на два операнда	Op1:=Op2 / Op3
REM	Остатък от деленето на два операнда	Op1:=Op2 REM Op3
SQRT	Корен квадратен от един операнд	Op1:=SQRT(Op2)
INC	Увеличава с единица един операнд	INC Op1
DEC	Намалява с единица един операнд	DEC Op1



Основни операции

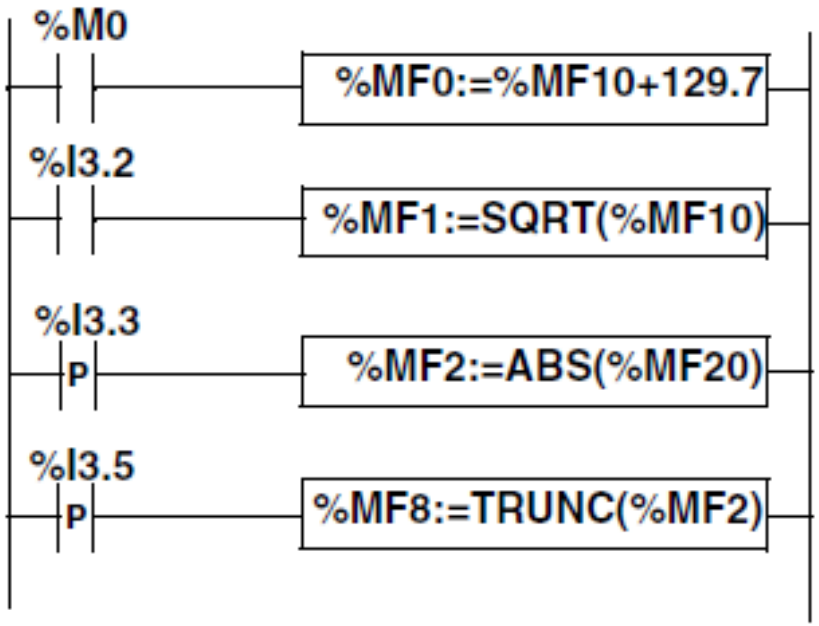
- Операции върху числа с плаваща запетая

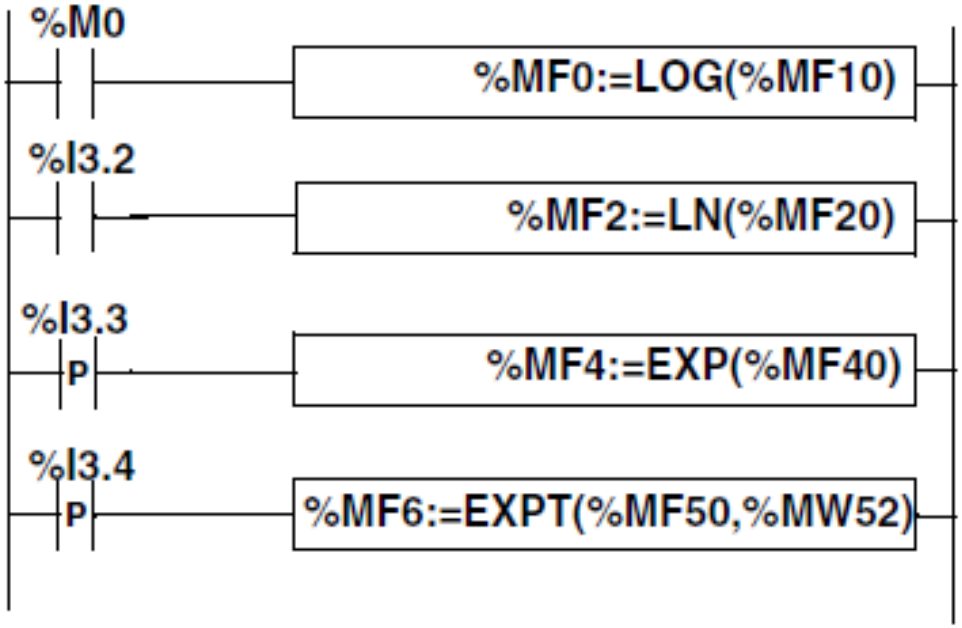
Означение	Действие	Означение	Действие
+	Събиране	SQRT	Корен квадратен
-	Изваждане	ABS	Абсолютна стойност
*	Умножение	TRUNK	Цялата част на реално число
/	Деление	EXP	Експонента e^x
LOG	Десетичен логаритъм	EXPT	Цяло число, повдигнато на степен, която е реално число
LN	Натурален логаритъм		

Основни операции

- Операции върху числа с плаваща запетая

Примери:





Основни операции

- Операции върху числа с плаваща запетая

Operators	Syntax
+, - *, /	Op1:=Op2 Operator Op3
SQRT, ABS, TRUNC, LOG, EXP, LN	Op1:=Operator(Op2)
EXPT	Op1:=Operator (Op2,Op3)

Основни операции

- Операции за изчисляване на тригонометрични функции

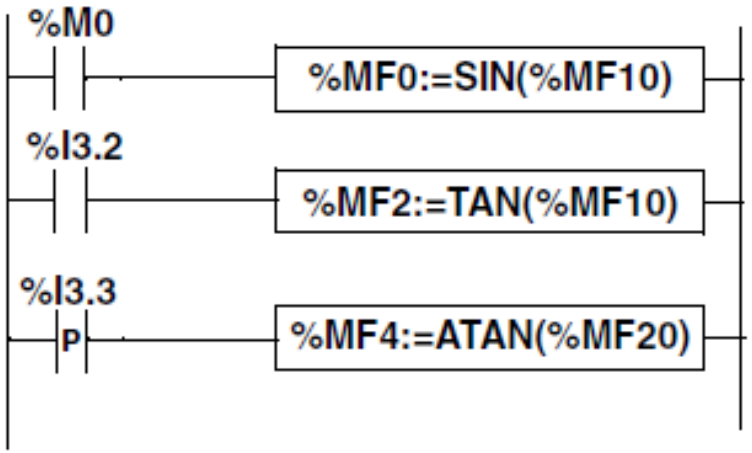
Синтаксис:

SIN	sine of an angle expressed as a radian,	**ASIN**	arc sine (result within $-\frac{\pi}{2}$ and $\frac{\pi}{2}$)
COS	cosine of an angle expressed as a radian,	**ACOS**	arc cosine (result within 0 and π)
TAN	tangent of an angle expressed as a radian,	**ATAN**	arc tangent (result within $-\frac{\pi}{2}$ and $\frac{\pi}{2}$)

Основни операции

- Операции за изчисляване на тригонометрични функции

Примери:



Синтаксис:

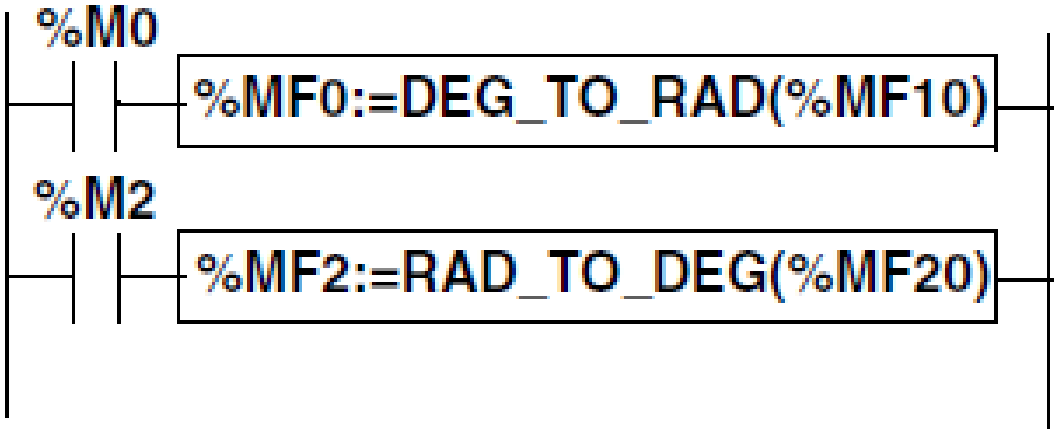
Operators	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
SIN, COS, TAN, ASIN, ACOS, ATAN	Op1:=Operator(Op2)	%MFi	%MFi, %KFi

Основни операции

- Инструкции за преобразуване

DEG_TO_RAD – преобразува градуси в радиани, резултатът е стойност на ъгъла между 0 и 2 π

RAD_TO_DEG - преобразува радиани в градуси, резултатът е стойност на ъгъла между 0 и 360°

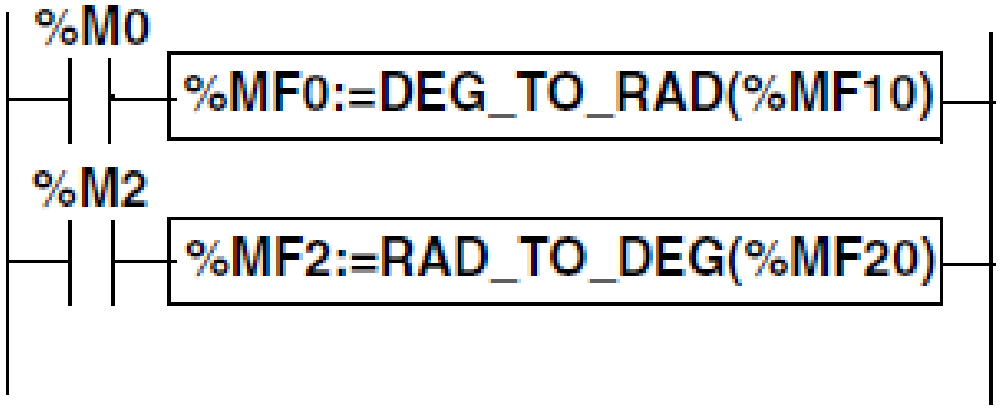


Основни операции

- **Инструкции за преобразуване**

DEG_TO_RAD – преобразува градуси в радиани, резултатът е стойност на ъгъла между 0 и 2 π

RAD_TO_DEG - преобразува радиани в градуси, резултатът е стойност на ъгъла между 0 и 360°



Синтаксис:

Operators	Syntax	Operand 1 (Op1)	Operand 2 (Op2)
DEG_TO_RAD RAD_TO_DEG	Op1:=Operator(Op2)	%MFi	%MFi, %KFi

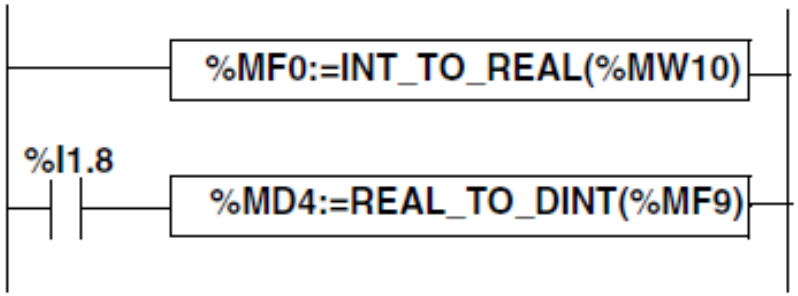
Основни операции

- **Инструкции за преобразуване**

цяло ↔ плаваща запетая

INT_TO_REAL	conversion of an integer word --> floating
DINT_TO_REAL	double conversion of integer word --> floating
REAL_TO_INT	floating conversion --> integer word (the result is the nearest algebraic value)
REAL_TO_DINT	floating conversion --> double integer word (the result is the nearest algebraic value)

Примери:



Основни операции

- **Инструкции за преобразуване**

цяло ↔ плаваща запетая

Синтаксис:

Operators	Syntax
INT_TO_REAL	Op1=INT_TO_REAL(Op2)

Operand 1 (Op1)	Operand 2 (Op2)
%MFi	%MWi,%KWi

Operators	Syntax
DINT_TO_REAL	Op1=DINT_TO_REAL(Op2)

Operand 1 (Op1)	Operand 2 (Op2)
%MFi	%MDi,%KDi

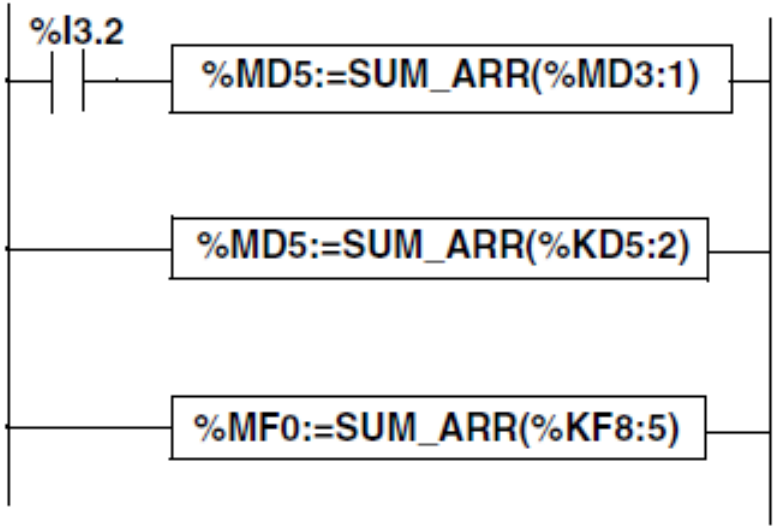
Operators	Syntax
REAL_TO_INT	Op1=Operator(Op2)
REAL_TO_DINT	

Type	Operand 1 (Op1)	Operand 2 (Op2)
Words	%MWi	%MFi, %KFi
Double words	%MDi	%MFi, %KFi

Основни операции

- **Инструкции за работа с таблици**

SUM_ARR – събира всички елементи от таблицата

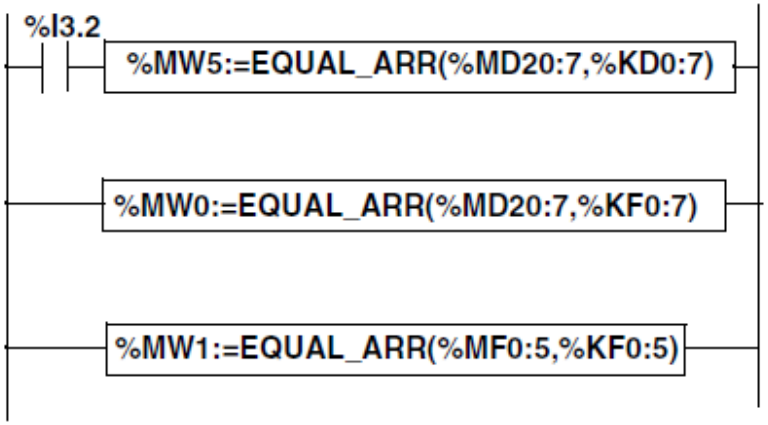


Основни операции

- **Инструкции за работа с таблици**

EQUAL _ARR – сравнява две таблици елемент по елемент

Ако има разлика, то първият различен елемент се връща под формата на дума, иначе се връща -1.



Синтаксис:

Syntax of table comparison instruction:

```
Res:=EQUAL_ARR(Tab1,Tab2)
```

Parameters of table comparison instructions:

Type	Result (Res)	Tables (Tab1 and Tab2)
Double word tables	%MWi	%MDi:L,%KDi:L
Floating word tables	%MWi	%MFi:L,%KFi:L

Основни операции

- **Инструкции за работа с таблици**

EQUAL _ARR – сравнява две таблици елемент по елемент

Ако има разлика, то първият различен елемент се връща под формата на дума, иначе се връща -1.

Пример:

```
%MW5:=EQUAL_ARR(%MD30:4,%KD0:4)
```

Comparison of 2 tables:

Rank	Word Table	Constant word tables	Difference
0	%MD30=10	%KD0=10	=
1	%MD31=20	%KD1=20	=
2	%MD32=30	%KD2=60	Different
3	%MD33=40	%KD3=40	=

The value of the word %MW5 is 2 (different first rank)

Основни операции

- **Инструкции за работа с таблици**

Функции за търсене

FIND_EQR: претърсва за първия елемент, който е равен на зададена стойност. Връща позицията му.

FIND_GTR: претърсва за първия елемент, който е по-голям от зададена стойност. Връща позицията му.

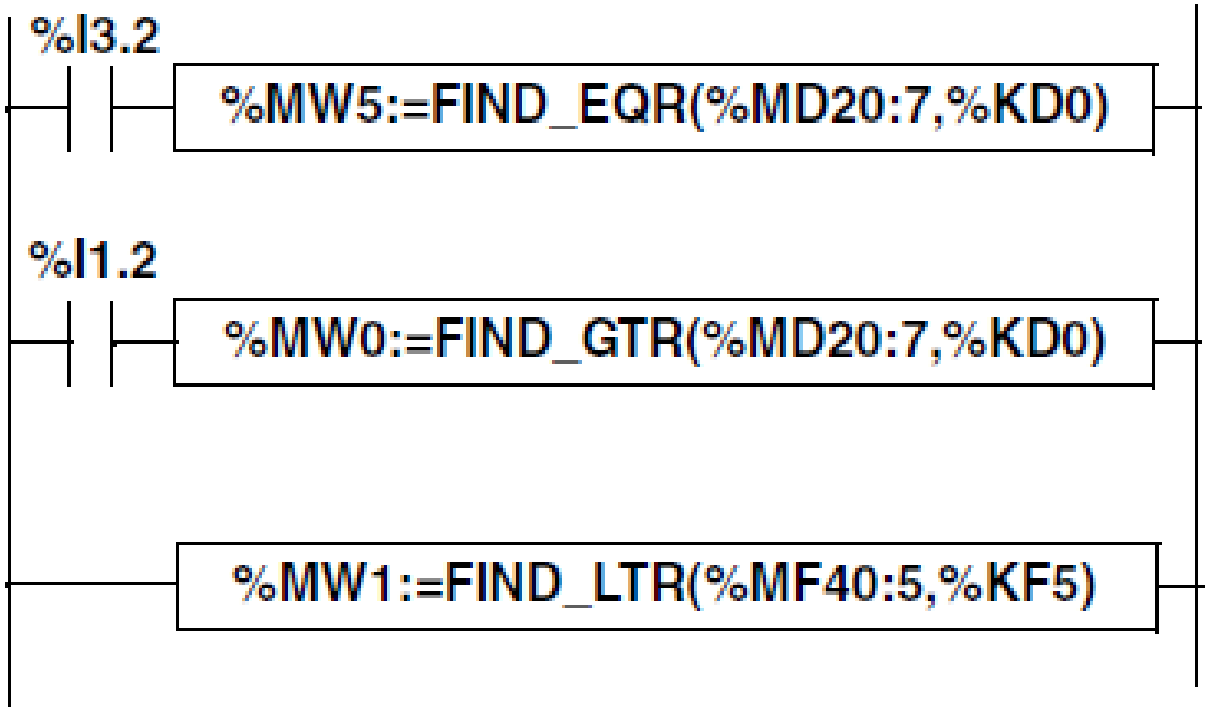
FIND_LTR: претърсва за първия елемент, който е по-малък от зададена стойност. Връща позицията му.

Ако не открие такъв елемент, връща 0.

Основни операции

- **Инструкции за работа с таблици**

Примери:



Основни операции

- **Инструкции за работа с таблици**

Синтаксис:

Syntax of table search instructions:

Function	Syntax
FIND_EQR	Res:=Function(Tab,Val)
FIND_GTR	
FIND_LTR	

Parameters of floating word and double word table search instructions:

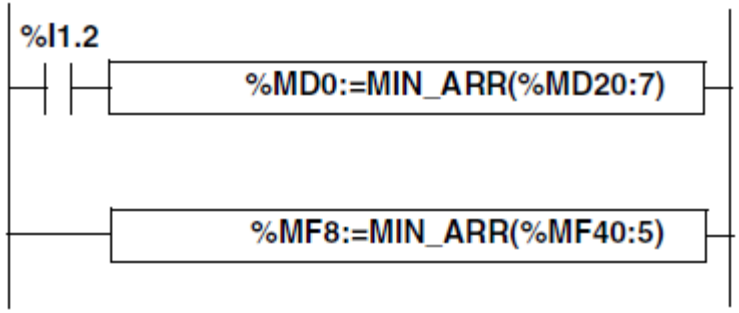
Type	Result (Res)	Table (Tab)	Value (val)
Floating word tables	%MWi	%MFi:L,%KFi:L	%MFi,%KFi
Double word tables	%MWi	%MDi:L,%KDi:L	%MDi,%KDi

Основни операции

- **Инструкции за работа с таблици**

MAX_ARR: търси максималния елемент от таблица

MIN_ARR: търси минималния елемент от таблица



Syntax of table search instructions for max and min values:

Function	Syntax
MAX_ARR	Res:=Function(Tab)
MIN_ARR	

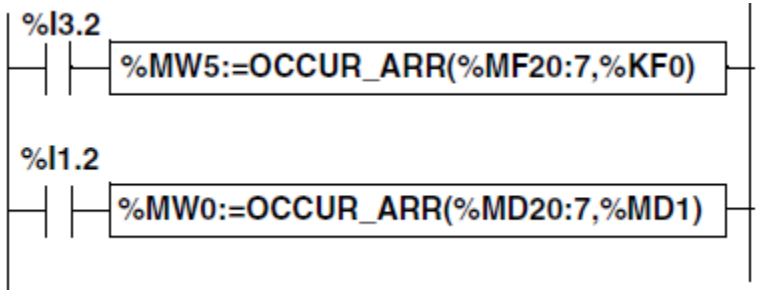
Parameters of table search instructions for max and min values:

Type	Result (Res)	Table (Tab)
Double word tables	%MDi	%MDi:L,%KDi:L
Floating word tables	%MFi	%MFi:L,%KFi:L

Основни операции

- **Инструкции за работа с таблици**

OCCUR_ARR: търси броя на елементите от таблица, които са равни на зададена стойност



Syntax of table search instructions for max and min values:

Function	Syntax
OCCUR_ARR	Res:=Function(Tab,Val)

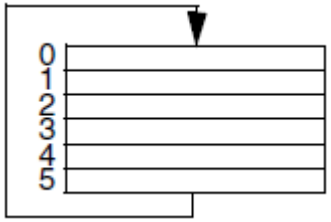
Parameters of table search instructions for max and min values:

Type	Result (Res)	Table (Tab)	Value (Val)
Double word tables	%MWi	%MDi:L,%KDi:L	%MDi,%KDi
Floating word tables	%MFi	%MFi:L,%KFi:L	%MFi,%KFi

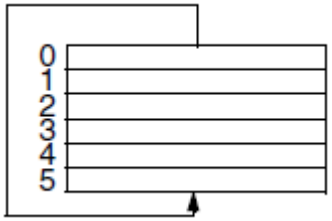
Основни операции

- **Инструкции за работа с таблици**

ROL_ARR: извършва ротация на n позиции на елементите на таблица в посока отгоре надолу



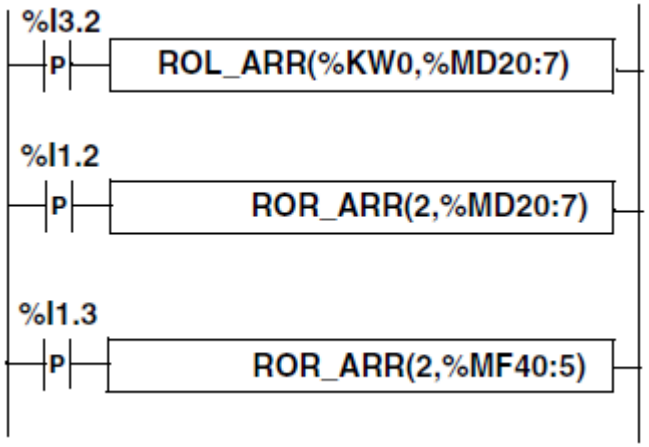
ROR_ARR: извършва ротация на n позиции на елементите на таблица в посока отдолу нагоре



Основни операции

- **Инструкции за работа с таблици**

ROL_ARR; ROR_ARR



Function	Syntax
ROL_ARR	Function(n,Tab)
ROR_ARR	

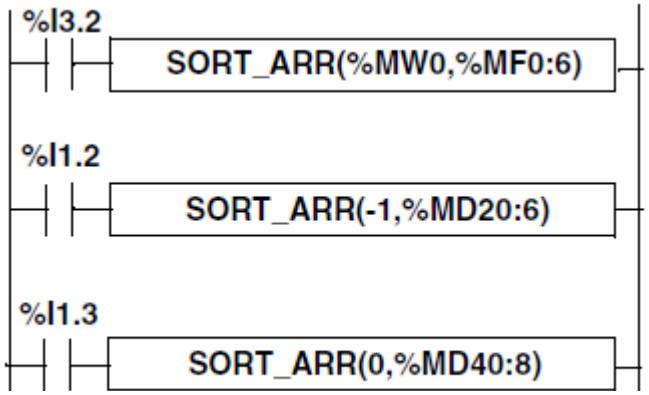
Parameters of rotate shift instructions for floating word tables: **ROL_ARR** and **ROR_ARR**:

Type	Number of positions (n)	Table (Tab)
Floating word tables	%MWi, immediate value	%MFi:L
Double word tables	%MWi, immediate value	%MDi:L

Основни операции

- **Инструкции за работа с таблици**

SORT_ARR: пренарежда елементите по нарастваща или намаляваща стойност



Syntax of table sort functions:

Function	Syntax
SORT_ARR	Function(direction,Tab)

- the "direction" parameter gives the order of the sort: direction > 0 the sort is done in ascending order; direction < 0, the sort is done in descending order, direction = 0 no sort is performed.
- the result (sorted table) is returned in the Tab parameter (table to sort).

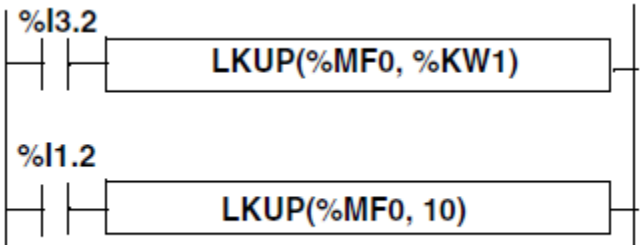
Parameters of table sort functions:

Type	Sort direction	Table (Tab)
Double word tables	%MWi, immediate value	%MDi:L
Floating word tables	%MWi, immediate value	%MFi:L

Основни операции

- **Инструкции за работа с таблици**

LKUP: графично представяне на данните, записани в таблицата, чрез линейна интерполация между точките



The interpolation algorithm applied to the floating points in the table is as follows:

$Y = Y_i + ((Y_{i+1} - Y_i)/(X_{i+1} - X_i))*(X - X_i)$

for $X_i \leq X \leq X_{i+1}$ where i = 1 ... (m-1)

Syntaxe of the function:

Function	Syntax
LKUP	Result=LKUP(Tab, Nb)

Operands of the function:

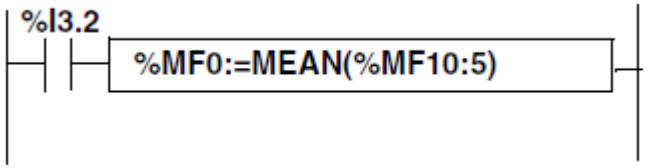
Table (Tab)	Result (Res)	Nb
%MFi	%MWi	immediate value, %MWi, %KWi

The operand "Nb" represents thr number of floating points in the table to be taken into consideration. It must always be an even number.

Основни операции

- **Инструкции за работа с таблици**

MEAN : определяне на средно аритметична стойност между указан брой елементи на таблицата



Syntax of the floating point table mean calculation function:

Function	Syntax
MEAN	Result=Function(Op1)

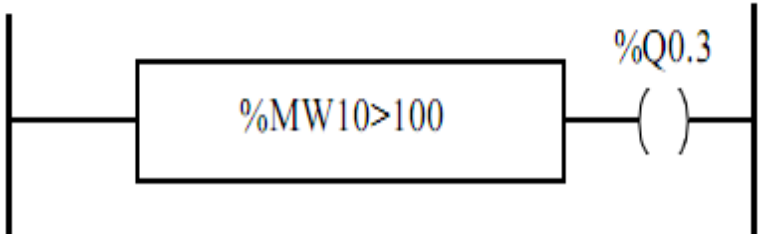
Parameters of the calculation function for a given number L of values from a floating point table:

Operand (Op1)	Result (Res)
%MFi:L, %KFi:L	%MFi

Основни операции

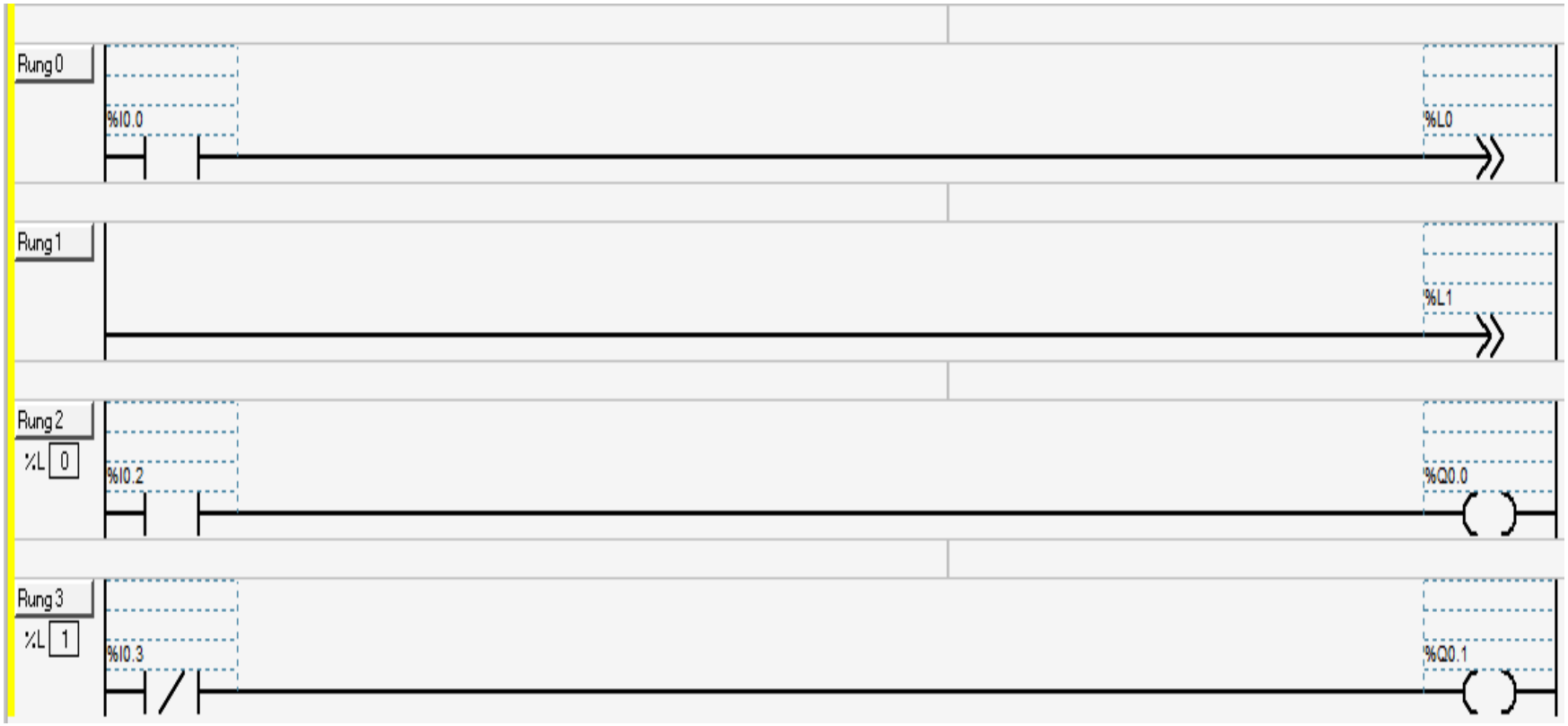
- Инструкции за аритметическо сравнение

Инструкция/ Оператор	Функция	Синтаксис
>	Проверява дали оператор 1 е по-голям от оператор 2	Op1 > Op2
>=	Проверява дали оператор 1 е по-голям или равен от оператор 2	Op1 >= Op2
<	Проверява дали оператор 1 е по-малък от оператор 2	Op1 < Op2
<=	Проверява дали оператор 1 е по-малък или равен от оператор 2	Op1 <= Op2
=	Проверява дали оператор 1 е равен на оператор 2	Op1 = Op2
<>	Проверява дали оператор 1 е различен от оператор 2	Op1 <> Op2



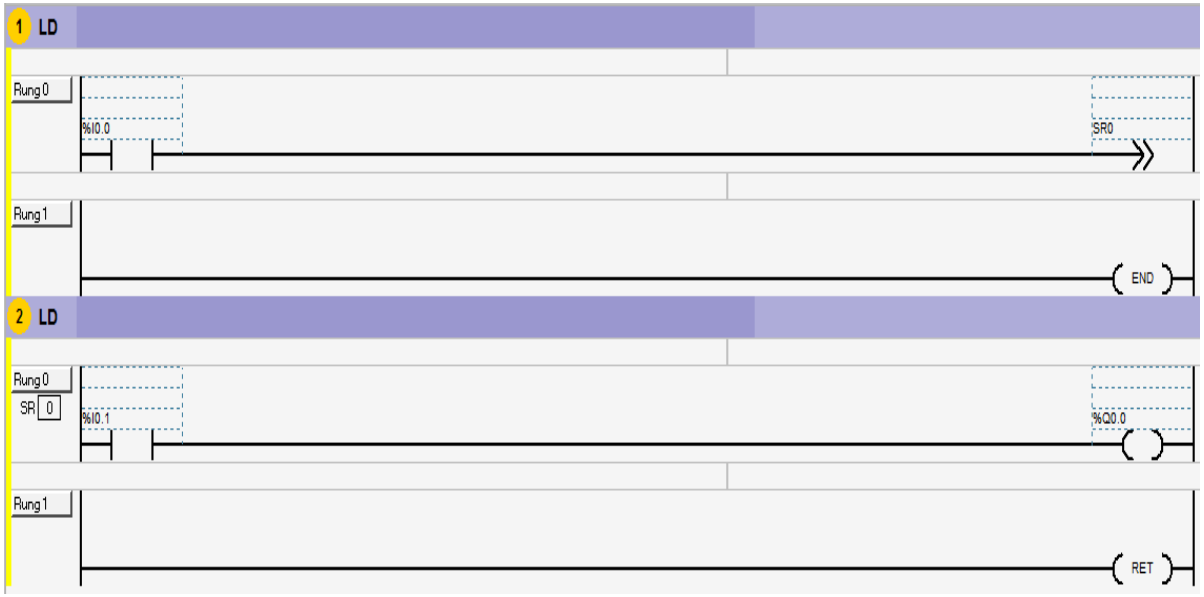
Основни операции

- Инструкции за условен и безусловен преход



Основни операции

- Инструкции за преход към подпрограма



Twido допуска до 64 подпрограми, които НЕ МОГАТ да се влагат!!!

Литература

TwidoSW.pdf