Sterowniki PLC

Pierwsze systemy sterowania budowane były z układów elektromechanicznych, tzn. z różnego typu przekaźników, układów czasowych i liczników.

Takie systemy zajmowały dużą objętość.

Budowane były duże szafy sterownicze.

Systemy charakteryzowały się dużą zawodnością.

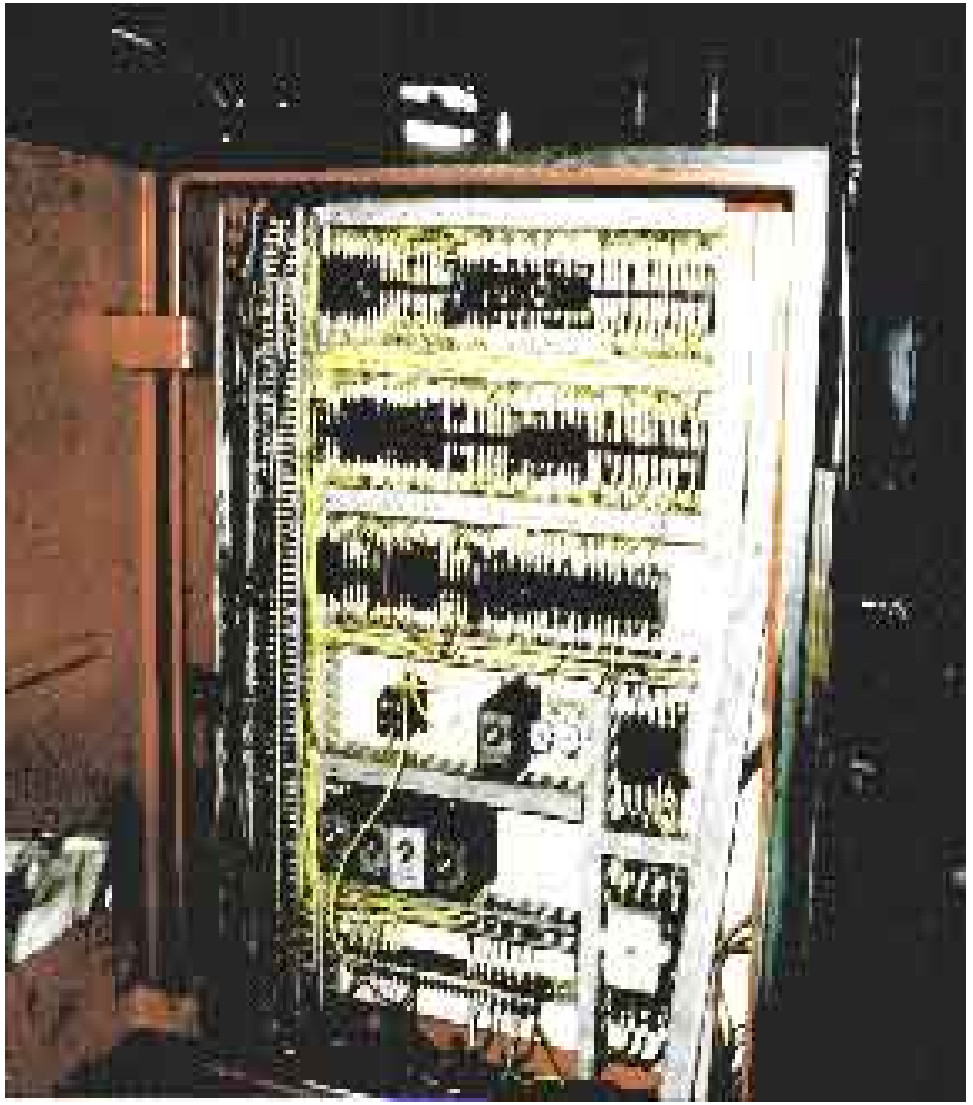
Sterowniki PLC

Postanowiono zastąpić elementy elektromechaniczne ich elektronicznymi analogami.

Uzyskano w rezultacie system sterowania znacznie bardziej elastyczny i niezawodny, prostszy w uruchamianiu i serwisowaniu, zajmujący zdecydowanie mniejszą objętość.

Ten kierunek zmian doprowadził do powstania tzw. sterowników programowalnych, w skrócie PLC (ang. Programmable Logic Controller).

Sterowniki PLC



Sterowniki PLC

1. Łatwość programowania i przeprogramowania, stosownie do zmieniających się warunków pracy.

2. Łatwość utrzymania w ruchu produkcyjnym, z możliwością napraw przez wymianę instalowanych modułów *(plug-in modules).*

3. Większa niezawodność w warunkach przemysłowych, przy mniejszych gabarytach niż sprzęt przekaźnikowy.

4. Koszty porównywalne ze stosowanymi panelami przekaźnikowymi i szafami sterowniczymi.

Sterowniki PLC

PLC (Programowalny Sterownik Logiczny) (ang. Programmable Logic Controller) – uniwersalne urządzenie mikroprocesorowe przeznaczone do sterowania pracą maszyny lub urządzenia technologicznego.

Sterownik PLC musi zostać dopasowany do określonego obiektu sterowania poprzez wprowadzenie do jego pamięci żądanego algorytmu działania obiektu.

Cechą charakterystyczną sterowników PLC odróżniającą ten sterownik od innych sterowników komputerowych jest cykliczny obieg pamięci programu.

Historia sterowników PLC

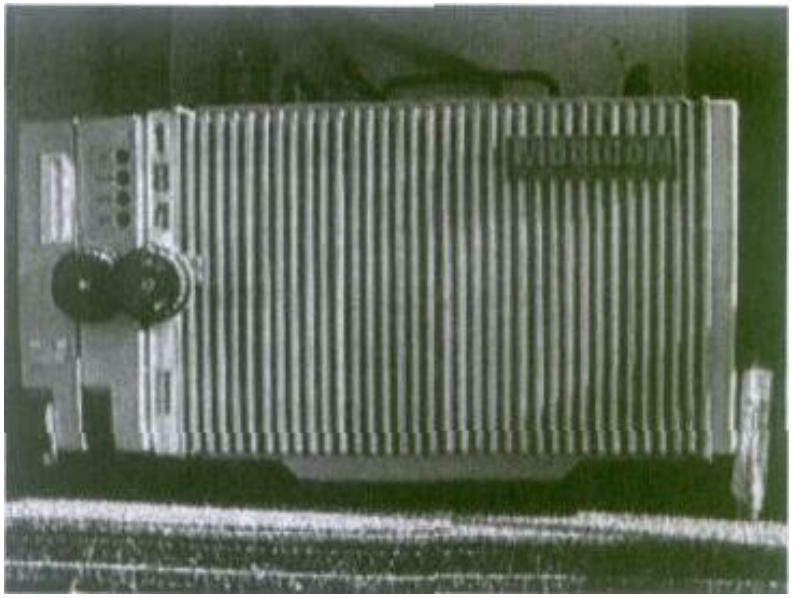
- W roku 1968 inżynierowie amerykańskiego przemysłu samochodowego wyszli z inicjatywą wprowadzenia sterowania nowego typu, w którym algorytm działania zapisywany byłby nie w "odrutowaniu" lecz w pamięci.

- W roku 1970 na wystawie obrabiarek w Chicago przedstawiono pierwszy system sterowania działający na zasadzie cyklicznego obiegu pamięci programu.

Dane techniczne

	S5-95U	**S7-314IFM**
Język programowania	STEP 5	STEP7
Pamięć	16 kB	24 kB
Czas obróbki instrukcji binarnej	2µs	1µs
Znaczniki	2048	2048
Timery	128 0,01 ÷ 9990s	128 0,01 ÷ 9990s
Liczniki	128 0 ÷ 999	128 0 ÷ 999
Wejścia cyfrowe (wbudowane)	16	16 + 4
Wyjścia cyfrowe (wbudowane)	16	16
Wejścia analogowe (wbudowane)	8 0 ÷ 10 V	4 0 ÷ 10 V
Wyjścia analogowe (wbudowane)	1 0 ÷ 10 V ; 0 ÷20 mA	1 0 ÷ 10 V ; 0 ÷20 mA
Możliwości rozbudowy	32 moduły	32 moduły

Historia sterowników PLC



Za pierwszy sterownik jest uznawany sterownik zaprojektowany przez Dicka Morleya w 1969 r., firma Modicon

W 1976 roku wprowadzono sterowniki PLC wyposażone w kasety sterowania zdalnego, które umożliwiły monitorowanie i uaktualnianie dużej liczby punktów wejść i wyjść *(I/O, Input/Output)* za pomocą połączeń komunikacyjnych, przy odległościach nawet do kilkuset metrów od jednostki centralnej sterownika.

Historia sterowników PLC

W 1977 roku firma Allen-Bradley jako pierwsza zastosowała w sterownikach PLC mikroprocesor 8080 z wykorzystaniem dodatkowego koprocesora dla operacji bitowych.

Z początkiem lat osiemdziesiątych **XX** w. zaczęto w sterownikach wprowadzać moduły inteligentne, które - wyposażone we własne procesory – mogły realizować znacznie bardziej złożone funkcje obliczeniowe.

Sterowniki PLC zaczęły także zastępować nie tylko przekaźnikowe układy sterowania logicznego, lecz także regulatory analogowe, a nawet mikrokomputery.

Zalety stosowania sterowników PLC

- łatwość programowania z użyciem języka schematów drabinkowych, podobnego do klasycznych schematów stykowo-przekaźnikowych,

- zwiększenie niezawodności komputerów przemysłowych na tyle, aby mogły działać w zanieczyszczonym środowisku,

- wprowadzenie programowej kontroli obwodów wejściowych i wyjściowych oraz innych możliwości diagnostyki systemowej i obiektowej,

- zapewnienie komunikacji z gniazdami przemysłowymi, panelami operatorskimi, wyświetlaczami, komputerami osobistymi oraz innymi urządzeniami stanowiącymi łącze operatora *(MMI, Man Machine Interface).*

Zasada działania sterowników PLC

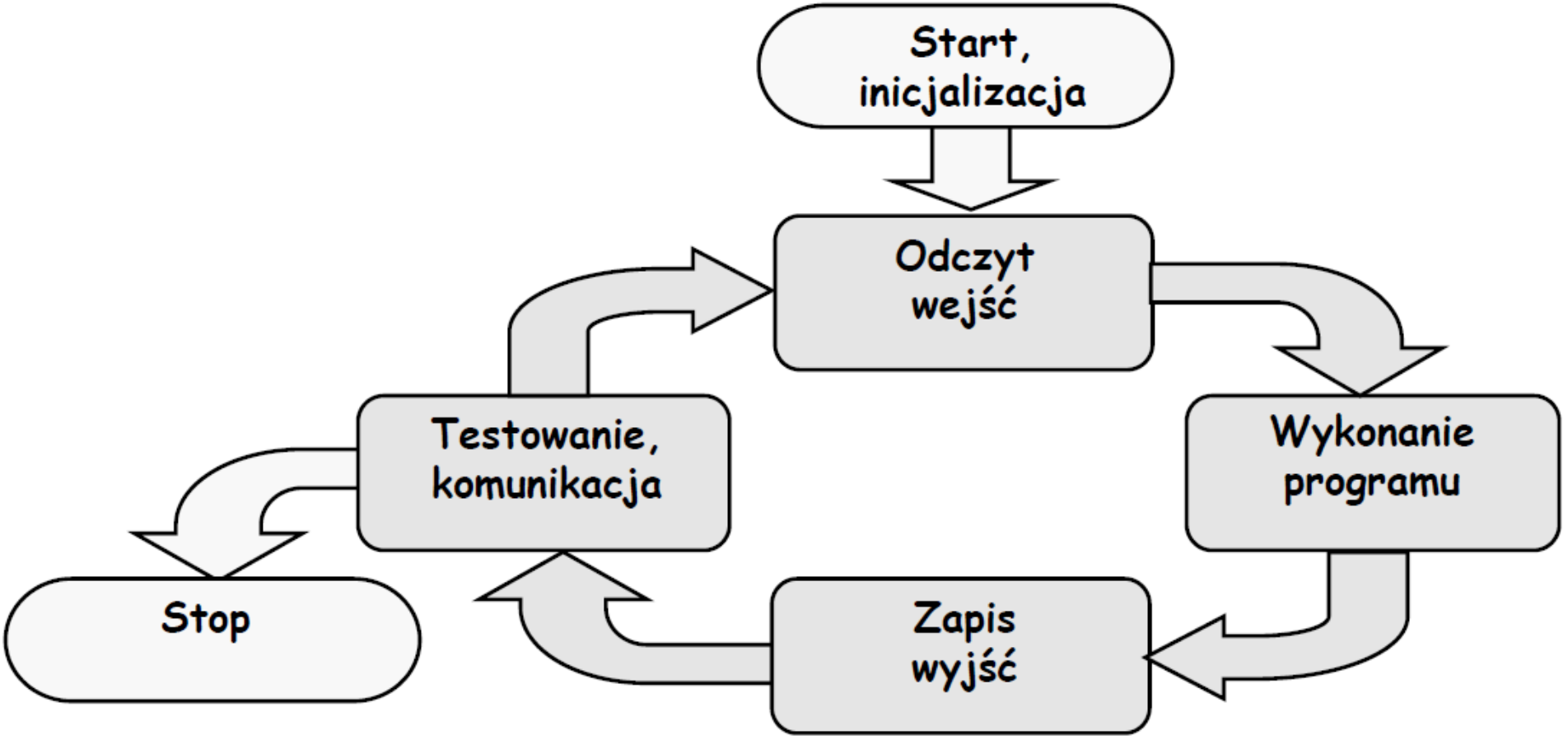
Sterownik wykonuje kolejno po sobie pojedyncze rozkazy programu w takiej kolejności, w jakiej są one zapisane w programie. Na początku każdego cyklu program odczytuje "obraz" stanu wejść sterownika i zapisuje ich stany (obraz wejść procesu).

Po wykonaniu wszystkich rozkazów i określeniu (wyliczeniu) aktualnego dla danej sytuacji stanu wyjść, sterownik wpisuje stany wyjść do pamięci będącej obrazem wyjść procesu a system operacyjny wysterowuje odpowiednie wyjścia sterujące elementami wykonawczymi.

Cykl pracy sterownika

- Autodiagnostyka
- Odczyt wejść
- Wykonanie programu
- Zadania komunikacyjne
- Ustawienia wyjść

Cykl pracy sterownika



Cykl pracy sterownika PLC

Cykl pracy sterownika

Po starcie sterownik przechodzi do podstawowego cyklu pracy. Cykl (skan) rozpoczyna się od równoległego odczytu wejść.

Obraz stanu wejść zapamiętany zostaje w pamięci wewnętrznej sterownika. Instrukcje programu wykonywane są w naturalnej kolejności aż do instrukcji kończącej program – END.

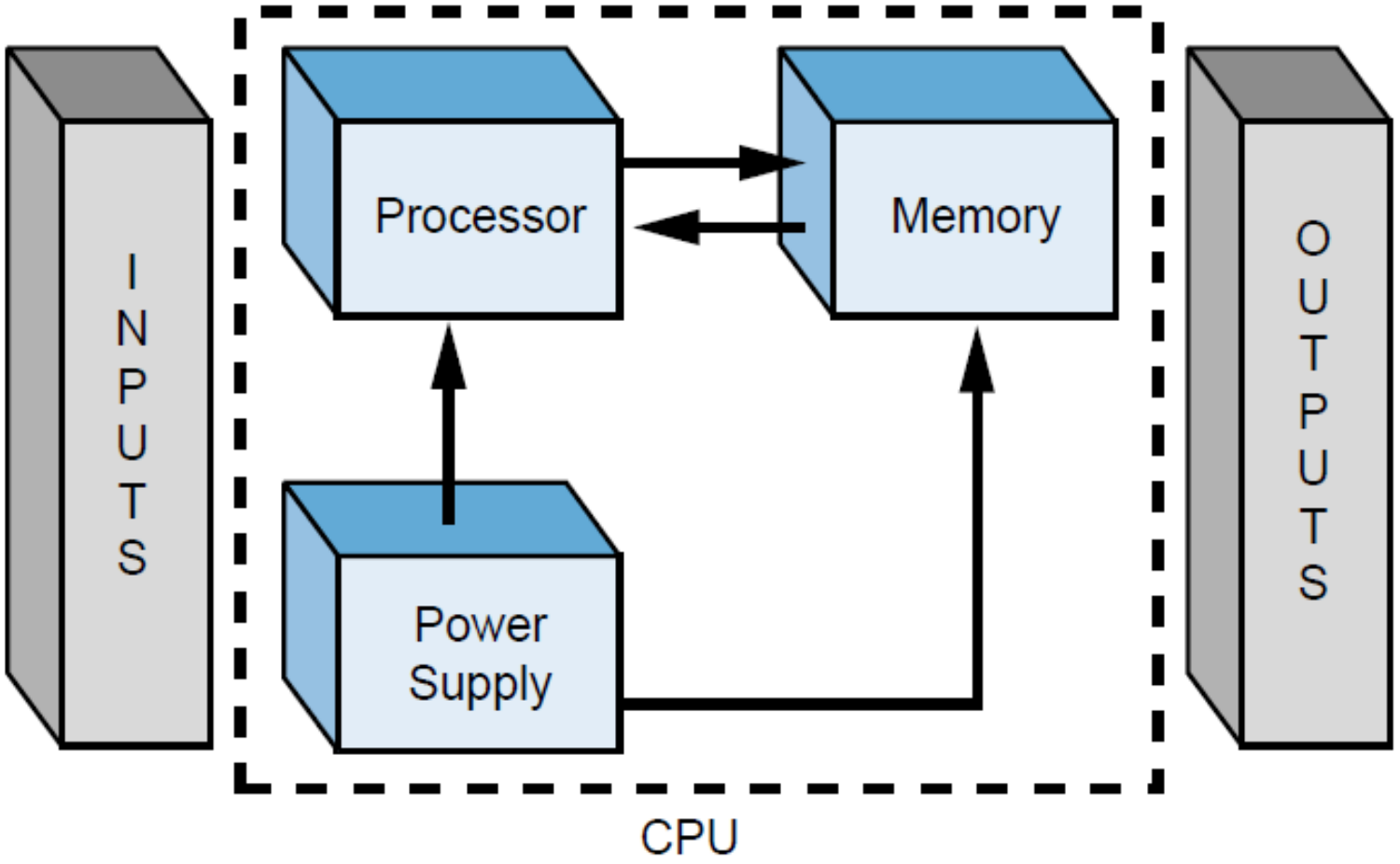
Jeżeli w trakcie wykonywania programu potrzebna jest informacja o stanie wejść to pobierana jest z pamięci obrazu. Podobnie aktualny stan wyjść zapisywany jest do pamięci wewnętrznej.

Cykl pracy sterownika cd.

Po instrukcji END sterownik przechodzi do kolejnej fazy cyklu – stan wyjść z pamięci wewnętrznej zostaje przepisany do bufora wyjść fizycznych i jest podtrzymany przez kolejny cykl pracy sterownika.

Kolejna faza zamykająca pojedynczy cykl sterownika przeznaczona jest dla wewnętrznych operacji diagnostycznych oraz dla potrzeb komunikacji z urządzeniami zewnętrznymi (w tym z programatorem).

Struktura sterownika cd.




POWER
RUN
ERROR
COM1
RUN
STOP
IN
OUT
DVP-12SA2
COM1
R


DVP28SV
POWER
RUN
ERROR
BAT.LOW
RS-232
RS-485
IN
OUT
RUN
STOP
TR


SIMOS

Rodziny sterowników

- mogą być programowane w tym samym języku i z użyciem tego samego pakietu programowego,

- mają takie same zmienne programowe oraz taką samą strukturę modułów *I/O* (moduły, płyty łączeniowe, drajwery, kable łączeniowe itp.),

- istnieje możliwość przenoszenia programów między modelami oraz korzystania z tych samych opcji w każdym modelu.

• komunikacja z urządzeniami obiektowymi, takimi jak sterowniki, regulatory,

• zbieranie i przetwarzanie zmiennych procesowych pochodzących z urządzeń obiektowych oraz ich archiwizacji w bazie danych,

• interfejs operatora *(MMI)* służącego do wizualizacji procesu i jego obsługi (sterowanie ręczne, zmiany wielkości zadanych),

• wizualizacji wartości zmiennych procesowych (aktualnych i historycznych) w różnych formach graficznych,

• opracowania raportów dotyczących bieżącego stanu procesu, zużycia materiałów oraz stanu pracy maszyn i urządzeń,

Rozwiązania redundancyjne

Polegają na wprowadzeniu w warstwie obiektowej rezerwowych czujników oraz elementów sterowania. W warstwie komunikacji stosuje się również rezerwowe moduły komunikacyjne, uzupełnione odpowiednim oprogramowaniem, zapewniające ciągłe utrzymywanie w sieci połączenia - podstawowego, a w razie awarii – rezerwowego.

W warstwie sterownikowej pracują co najmniej dwie jednostki centralne, między którymi zapewnia się wymianę danych w każdym cyklu programowym. W systemach z gorącą rezerwą jednostki centralnej *(Hot Standby CPU Redundancy)* zapewnia się wymianę i porównanie obrazu procesu a następnie porównanie wyników obliczeń i sprawdzenie ich zgodności, po czym przekazuje się obliczone sterowania przez moduły komunikacyjne i moduły wyjściowe do organów wykonawczych.

Moduły inteligentne

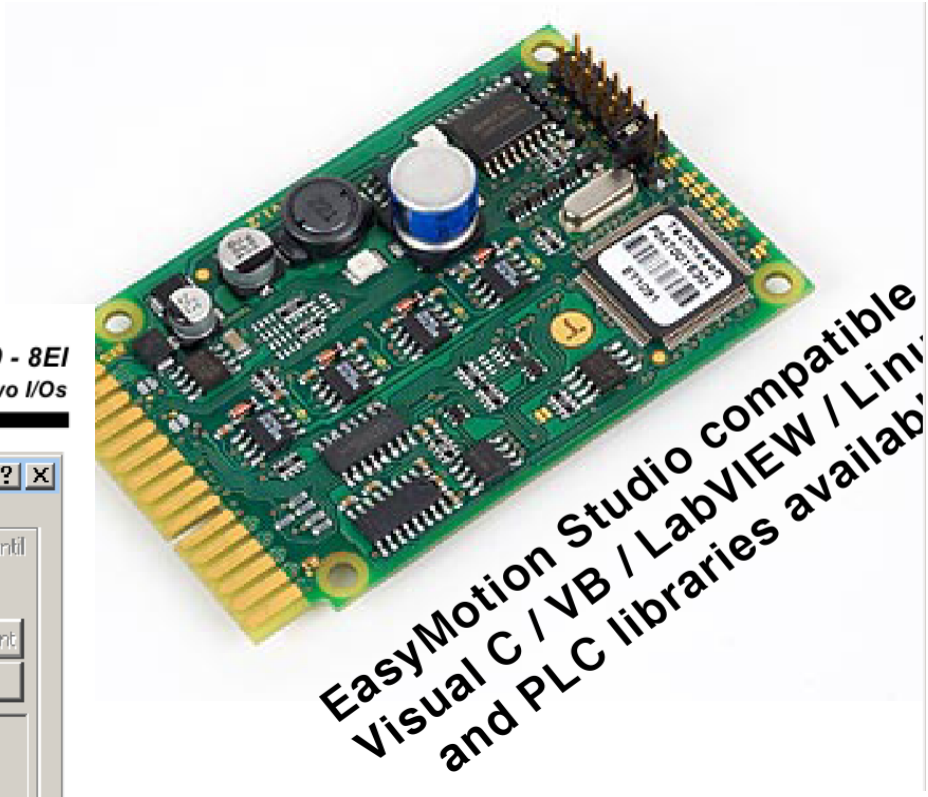
Moduły inteligentne (hardware) są to złożone, dedykowane układy sterowania, które jednak w prosty sposób można włączyć do magistrali sterowników lub do ich sieci komunikacyjnych. Mogą to być moduły systemów ważenia i dozowania, mierników elektrycznych z rozbudowanymi algorytmami obróbki sygnałów pomiarowych itp.

W latach dziewięćdziesiątych XX w. nastąpiło znaczne powiększenie asortymentu modułów inteligentnych (Intelligent Modules), wyposażonych we własne układy mikroprocesorowe, które realizują nieraz bardzo złożone algorytmy przetwarzania sygnałów i procedury sterowania oraz komunikacji.

Moduły inteligentne

Najszerzej są tu stosowane moduły szybkich liczników *(HSC, High Speed Counter)* oraz moduły pozycjonowania osi napędów *(APM, Axis Positioning Module),* które są przystosowane do współpracy ze sprzężonymi z tymi osiami przetwornikami obrotowo-impulsowymi (tzw. *enkoderami)* i układami sterowania napędów przez przetwornice częstotliwości źródeł zasilania (tzw. *falowniki*).

Moduły inteligentne cd.



APPLICATION NOTE 048.051.IDM.102
12/23/05

IDM240 - 5EI / IDM640 - 8EI
Start/Stop motion using two I/Os

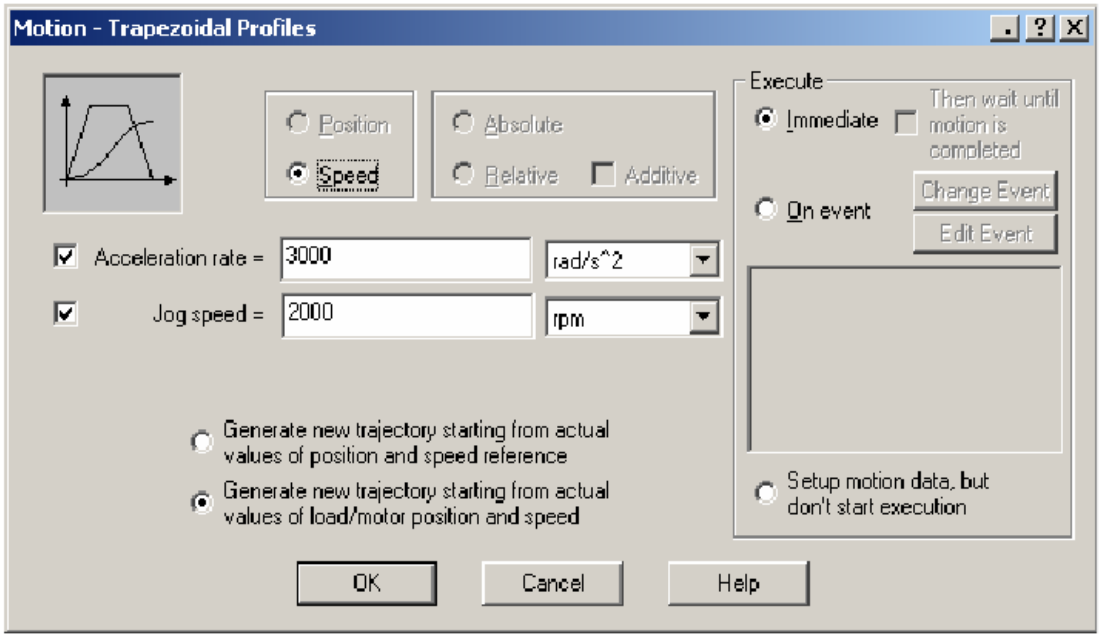


Figure 6. *How to configure and start motion using a speed profile (double-clicked 3)*

Moduły inteligentne cd.

Daje się też zauważyć dążenie do integracji w jednym systemie sterowania funkcji typowych dla sterownika PLC, panelu operatorskiego z jednoczesnym uniezależnieniem się od warstwy sprzętowej.

Korzystając z uniwersalnego, wspólnego dla różnych platform sprzętowych oprogramowania narzędziowego, można łatwo przenieść aplikację z jednego systemu do innego.
W ten sposób, w zależności od potrzeb, użytkownik może szybko rozbudowywać lub zmieniać swój system sterowania.

Komunikacja

Obecnie największy nacisk jest położony na zagadnienia komunikacyjne.

Stąd pojawianie się nowych modułów komunikacyjnych, umożliwiających przesyłanie danych na bardzo duże odległości między sterownikami PLC, czy to za *pomocą poczty elektronicznej,* czy też *SMS.*

Tego typu komunikacja umożliwia przede wszystkim powiadamianie obsługi procesu o nieprawidłowościach występujących w procesie i przesyłanie poleceń do sterownika.

Część 8. Wytyczne do stosowania i implementacji języków programowania

W tej części normy przedstawiono informacje uzupełniające dotyczące stosowania języków programowania zdefiniowanych w części 3 oraz ogólne wymagania dotyczące sprzętu i oprogramowania konieczne do rozwijania i konserwacji programów użytkownika.

Funkcje przetwarzania sygnałów

Przetwarzanie sygnałów jest realizowane przez *system operacyjny* oraz wykonywany *program użytkownika,* przy wykorzystaniu pamięci programu użytkownika i pamięci danych. Funkcja ta obejmuje przetwarzanie sygnałów pochodzących z czujników i wewnętrznej pamięci danych oraz wypracowanie sygnałów dla elementów wykonawczych i wewnętrznej pamięci danych, zgodnie z programem użytkownika.

Funkcje *systemu operacyjnego* są przeznaczone do zarządzania wewnętrznymi funkcjami sterownika PLC, takimi jak: kontrola konfiguracji, diagnostyka, zarządzanie pamięcią, zarządzanie wykonywaniem programu użytkownika, komunikacja
z urządzeniami peryferyjnymi i funkcjami interfejsu z czujnikami i elementami wykonawczymi.

Program i pamięci użytkownika

Program użytkownika może składać się z pewnej liczby zadań. Wykonywanie każdego zadania jest realizowane sekwencyjnie, a każda funkcja programowalna jest realizowana w czasie aż do zakończenia zadania. Rozpoczęcie zadania i jego realizacja (periodycznie lub sterowane przerwaniami) są wykonywane pod kontrolą systemu operacyjnego.

Pamięć programu użytkownika służy do zapewnienia miejsca w pamięci do przechowywania rozkazów, których wykonywanie (cykliczne lub sterowane przerwaniami) określa sposób działania sterowanego procesu lub maszyny.

Pamięć danych użytkownika służy do zapewnienia miejsca w pamięci do przechowywania danych operacyjnych wejść/wyjść *(I/O, Input/Output) oraz* innych danych (np. wartości nastaw przekaźników czasowych, liczników, warunków alarmów, parametrów dotyczących maszyny lub procesu), wymaganych podczas wykonywania programu użytkownika.

Całkowity czas odpowiedzi sterownika

Całkowity czas odpowiedzi sterownika (TRT, Total system Response Time) wykonującego program użytkownika jest sumą wszystkich czasów cząstkowych przejścia sygnału (informacji) przez system, poczynając od przyłączy elementów wejściowych, a kończąc na przyłączach wyjściowych. W przypadku periodycznego wykonywania zadania czas ten wynosi:

$$TRT = TID + TIT + rTSC + TUT + TQT + TQD$$

gdzie: *TID* - czas opóźnienia wejścia cyfrowego, *TIT*- czas przesyłania wejścia cyfrowego, *TSC* - czas próbkowania (czas cyklu sterownika), *TUT* - czas wykonania zadania użytkownika, *TQT'* - czas przesyłania wyjścia cyfrowego, *TQD - czas* opóźnienia wyjścia cyfrowego, *r* - parametr, którego wartość zależy od tego, czy informacja wejściowa jest dostępna do przetwarzania tuż przed rozpoczęciem zadania *(r* = r_{min}), czy tuż po rozpoczęciu zadania *(r* = r_{max}*)*- Wartości r_{min} i r_{max} zależą od organizacji programów użytkownika.

Funkcje interfejsu z czujnikami i elementami wykonawczymi

Informacje o stanie i danych z procesu lub maszyny są przenoszone do systemu I/O sterownika za pomocą sygnałów binarnych, cyfrowych, przyrostowych lub analogowych. Podobnie wyniki wypracowane przez funkcje przetwarzania są przenoszone do procesu lub maszyny za pomocą odpowiednich sygnałów binarnych, cyfrowych, przyrostowych lub analogowych. Duża różnorodność używanych czujników i elementów wykonawczych wymaga stosowania szerokiego zakresu sygnałów wejściowych i wyjściowych.

W systemach I/O są stosowane różne metody przetwarzania, konwersji i izolacji sygnałów. Na ogół systemy te charakteryzują się modułowością, pozwalającą na stosowanie konfiguracji zgodnie z potrzebami procesu lub maszyny, a także na późniejsze jej rozszerzanie.

Oddalone stanowiska I/O

Systemy I/O mogą być umieszczone w bezpośrednim sąsiedztwie funkcji przetwarzania sygnału (jednostki centralnej), ale mogą być też montowane w sąsiedztwie czujników lub elementów wykonawczych, z dala od funkcji przetwarzania sygnałów.

Szczególne rozwiązanie stanowią tu oddalone stanowiska I/O *(RIOS, Remote Input Output Station),* które umożliwiają obsługiwanie interfejsu wejścia/wyjścia pod nadzorem jednostki centralnej, a więc multipleksowanie/demultipleksowanie wejść/wyjść oraz wstępne i końcowe przetwarzanie danych.

Funkcje testujące

Podczas pisania, uruchamiania i sprawdzania programu użytkownik może być wspomagany przez następujące funkcje testujące:

• sprawdzanie stanów wejść i wyjść oraz funkcji wewnętrznych (zegary, liczniki),

• sprawdzanie sekwencji programu, np. praca krokowa, zmiany czasu cyklu programu, polecenia zatrzymania,

• symulacja funkcji interfejsów lub informacji wymienianej między zadaniami lub modułami wewnętrznymi sterownika.

Dokumentacja roli sterownika

Dokumentacja powinna zawierać pełny opis systemu sterownika i jego zastosowania, w tym:

- opis konfiguracji sprzętu,

- dokumentację programu użytkownika, zawierającą wydruk programu z nazwami mnemonicznymi sygnałów i danych procesowych, tabelę wzajemnych odwołań wszystkich danych procesowych, komentarze i opis modyfikacji,

- dokumentację serwisową.

Funkcje komunikacyjne

Funkcje komunikacyjne obsługują wymianę programów i danych między urządzeniami zewnętrznymi a jednostką centralną sterownika PLC.

Oprócz przesyłania programów i zbiorów danych umożliwiają także monitorowanie i diagnostykę systemu. Zwykle komunikacja taka jest realizowana za pomocą transmisji szeregowej w sieci lokalnej *(LAN, Local Area Network)* lub w systemie sprzężeń punkt do punktu *(point-to-point).*

Funkcje zasilania

Do funkcji zasilania należy wytwarzanie napięć niezbędnych do działania systemu PLC, a także dostarczanie sygnałów kontrolnych w celu poprawnej synchronizacji startu i stopu elementów wyposażenia.

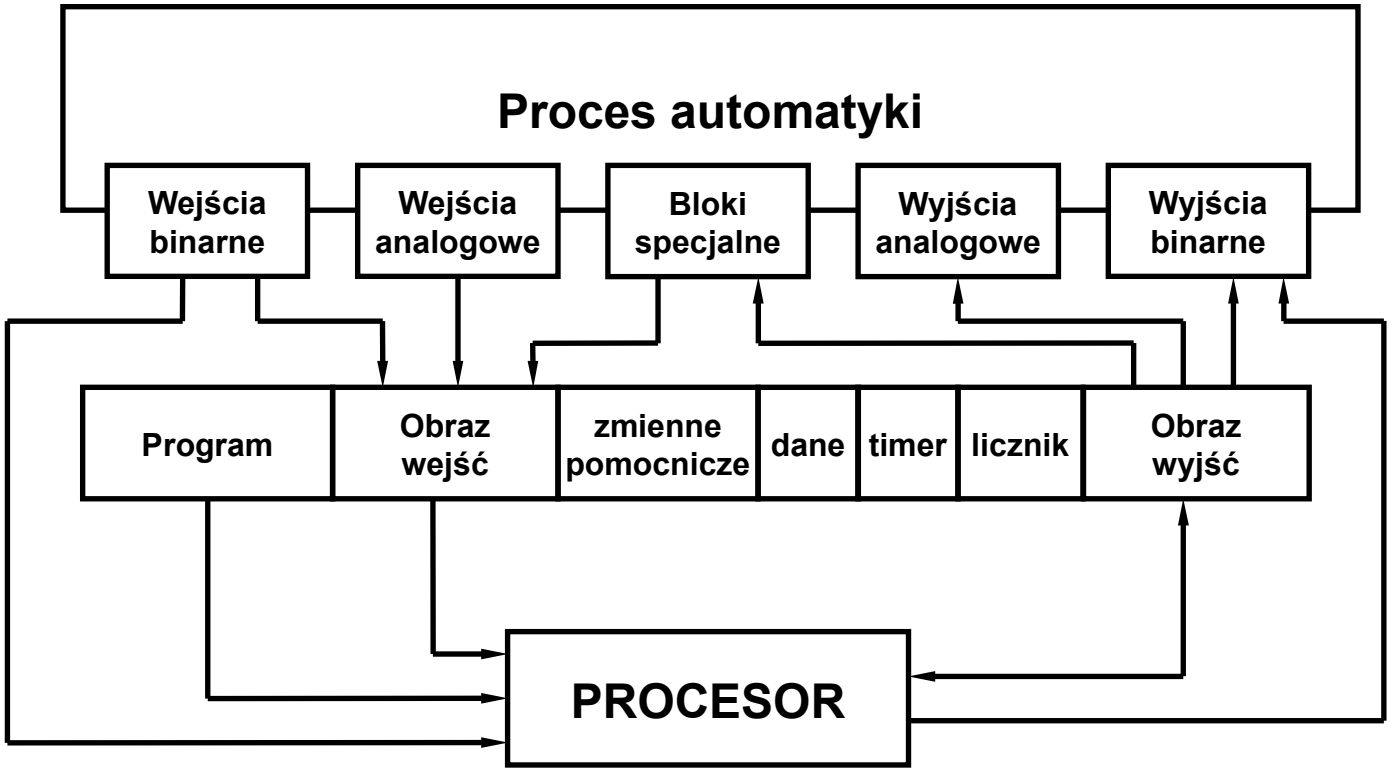
W zależności od wymaganych napięć, poboru mocy, wymagań bezprzerwowego działania itp. Stosuje się różne typy zasilaczy. Najczęściej obwody wejść i wyjść sterownika są zasilane napięciem stałym 24 V, a sam sterownik albo napięciem stałym 24 V, albo napięciem przemiennym 240 V.

Jednym z podstawowych czynników zwiększających dyspozycyjność systemu PLC jest stosowanie redundancji zasilania.

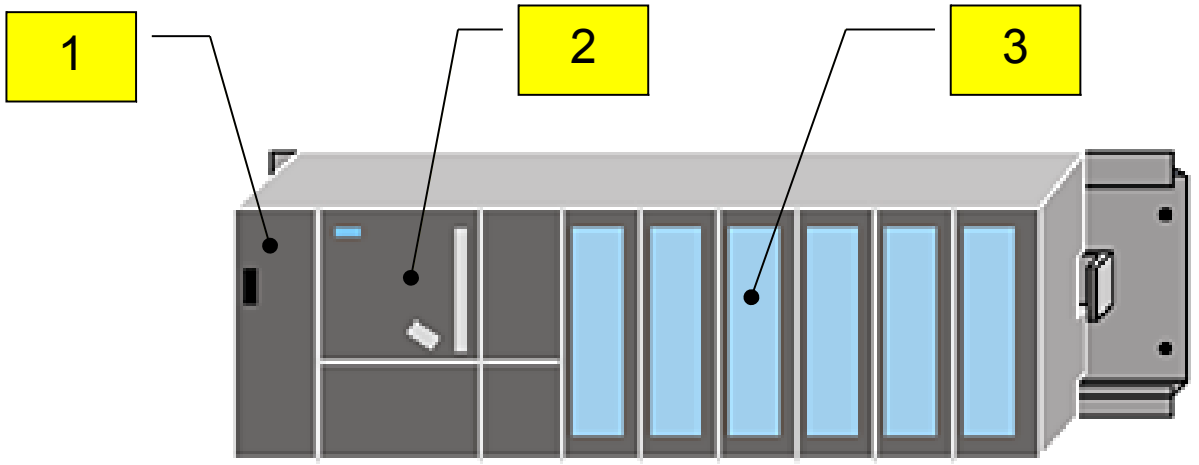
Języki programowania

- LD (Ladder Diagram) logika drabinkowa - schemat zbliżony do klasycznego rysunku technicznego elektrycznego;
- FBD (Function Block Diagram) - diagram bloków funkcyjnych, sekwencja linii zawierających bloki funkcyjne;
- ST (Structured Text) tekst strukturalny - język zbliżony do Pascala;
- IL (Instruction List) lista instrukcji - rodzaj asemblera;
- SFC (Sequential Function Chart) sekwencyjny ciąg bloków - sekwencja bloków programowych z warunkami przejścia.

Struktura sterowników



Montaż sterowników

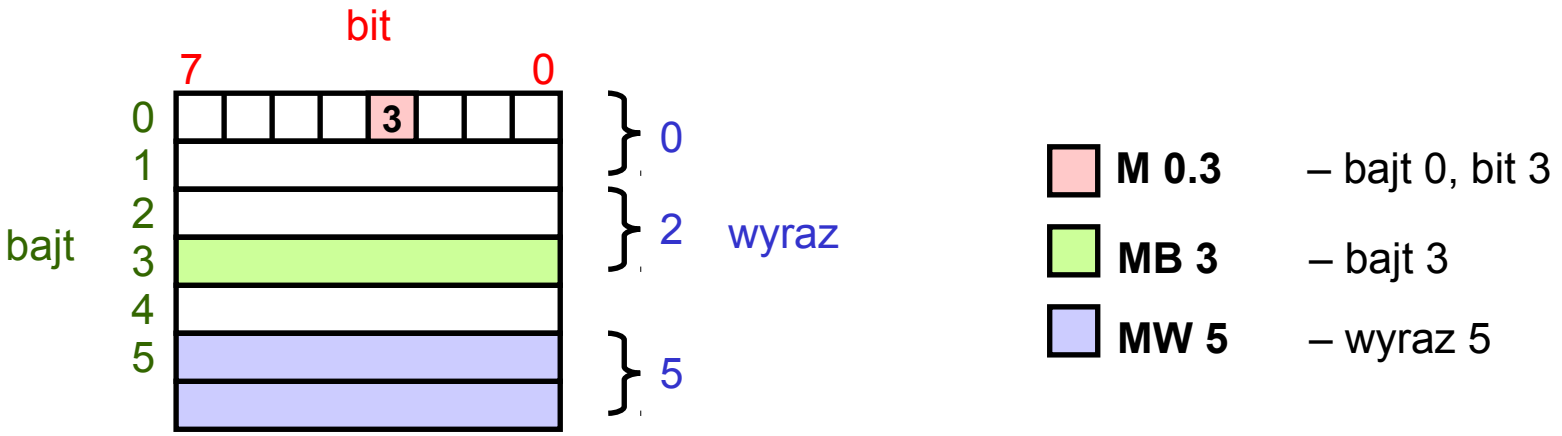


Sterowniki PLC są produkowane w postaci modułów montowanych na szynie montażowej w następującej kolejności:

1. Zasilacz.
2. Jednostka sterująca.
3. Moduły I/O (wejścia i wyjścia)

Adresowanie pamięci

W pamięci sterownika wyodrębniona jest pewna ilość miejsc do przechowywania chwilowych wyników operacji. W sterownikach PLC rozróżniamy 4 tryby adresowania: bitowo, bajtowo, wyrazowo oraz przy pomocy dwóch słów. Adresując słownie operujemy na 16-tu bitach i przy pomocy dwóch słów na 32-ch bitach.



Języki programowania

Istnieją 3 podstawowe języki programowania sterowników PLC:

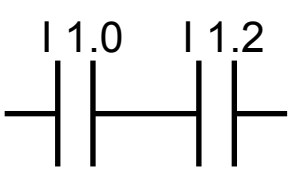
- **LAD** – jest to język oparty na rysowaniu schematu zwanego drabinkowym, bardzo wygodny do układania programu mając dany układ przekaźnikowy mający działać automatycznie,
- **CSF** (FBD) – stosowany do programowania sterownika, kiedy dysponujemy układem zbudowanym z bramek logicznych,
- **STL** – będący językiem mnemonicznym, o strukturze podobnej do wewnętrznego języka mikroprocesorów (asemblera).

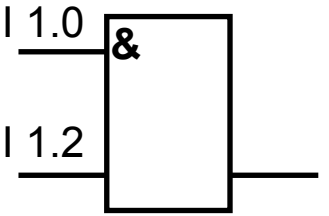
Biorąc pod uwagę funkcje jakie posiadają poszczególne języki, język STL oferuje największe możliwości, gdyż pozwala na użycie funkcji i instrukcji niedostępnych w dwóch pozostałych. Przekształcenie programu z LAD na CSF i odwrotnie jak również z LAD lub SCF na STL jest możliwe. Konwersja z STL na LAD lub CSF nie jest możliwa w każdym przypadku.

Struktura języków programowania

LAD **CSF** **STL**

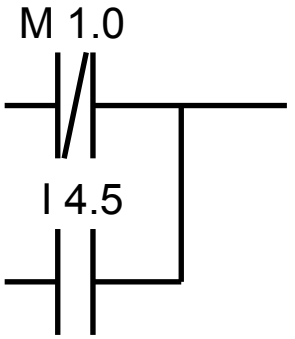
- iloczyn logiczny (AND)

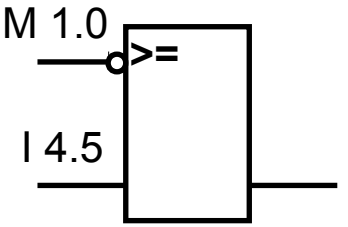




A I 1.0
A I 1.2

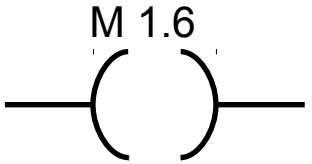
- suma logiczna (OR)

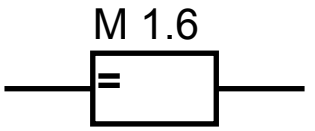




ON M 1.0
O I 4.5

- wynik operacji





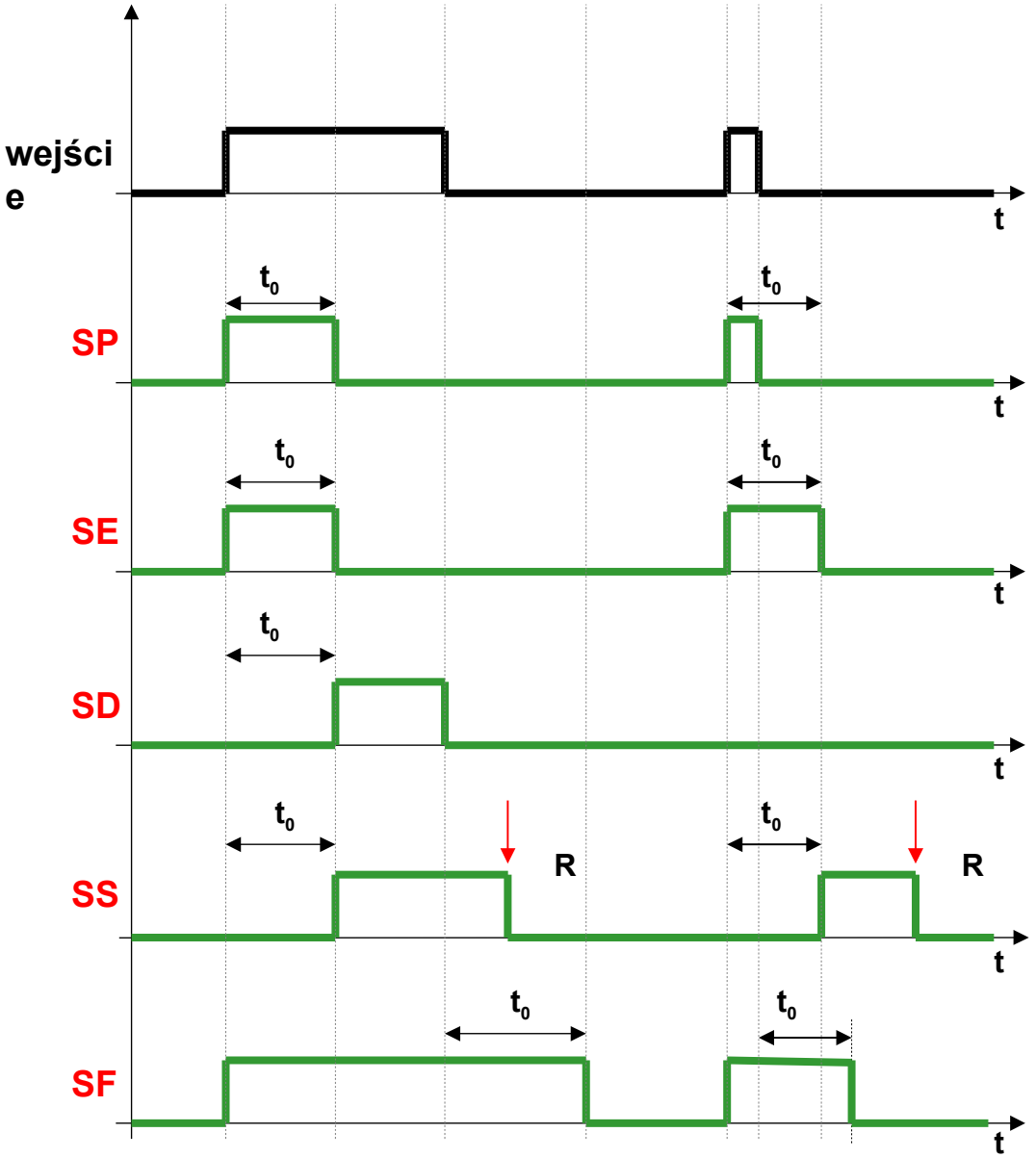
= M 1.6

Moduł czasowy (timer)

Działanie modułu czasowego odpowiada sposobowi działania przekaźnika czasowego z opóźnionym załączaniem lub wyłączaniem. Maksymalnie można zaprogramować 128 modułów czasowych oznaczonych instrukcją T0 do T127. W sterownikach Simatic możemy korzystać s 5-ciu różnie działających układów czasowych:

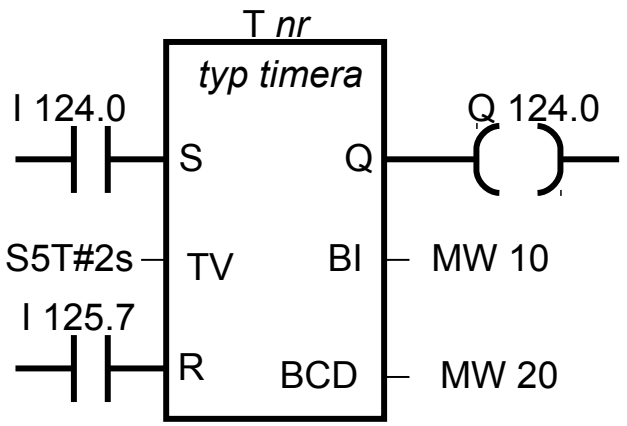
SP – **Pulse Timer**: Daje na wyjściu sygnał o określonej długości tylko przy aktywnym sygnale START.
SE – **Extended Pulse Timer**: Daje na wyjściu sygnał o określonej długości przy krótkiej aktywacji sygnału START.
SD – **On-Delay Timer**: Ustawienie timera jako timer z opóźnionym załączaniem.
SS – **Retentiv On-Delay Timer**: Uaktywnia się przez krótką aktywację sygnału START. Kasowanie jest możliwe tylko wejściem kasującym.
SF – **Off-Delay Timer**: Ustawienie timera jako timer z opóźnionym wyłączaniem.

Wykresy czasowe



Wykorzystanie timera

LAD



STL

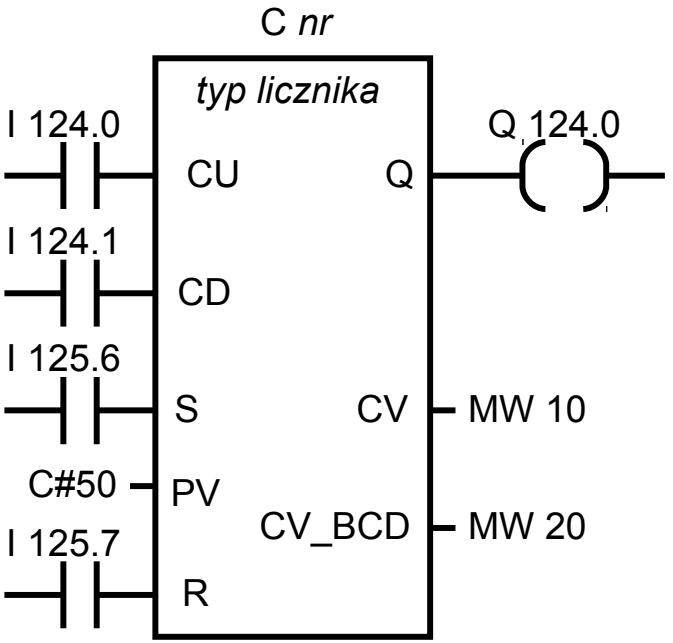
```
A     I  124.0
L     S5T#2s
SP    T1
A     I  125.7
R     T1
L     T1
T     MW  10
LC    T1
T     MW  20
A     T1
=     Q124.0
```

Typ timera	STEP 5	STEP 7
SP	1_-_	S_PULSE
SE	1_-_V	S_PEXT
SD	T!_!0	S_ODT
SS	T!_!S	S_ODTS
SF	0!_!T	S_OFFDT

Liczniki

Licznik może zliczać sygnały zarówno w przód jak i do tyłu. Zakres liczenia zawiera się w przedziale od 0 do 999. Maksymalnie można zaprogramować 128 liczników.

LAD



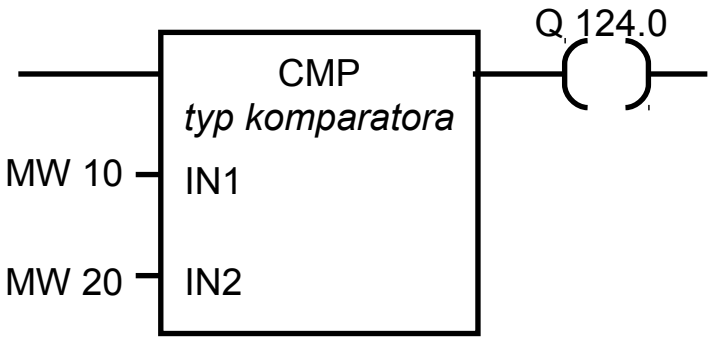
STL

```
A    I  124.0
CU   C1
A    I  124.1
CD   C1
A    I  125.6
L    C# 50
S    C1
A    I  125.7
R    C1
L    C1
T    MW10
LC   C1
T    MW20
A    C1
=    Q124.0
```

Komparatory

Komparator służy do porównywania ze sobą dwóch wartości 16-bitowych lub 32-bitowych.

LAD



STL

```
L     MW10
L     MW20
==I
=     Q124.0
```

typ komparatora	STEP 5 16-bit	STEP 7 16-bit	STEP 7 32-bit	STEP 7 rzeczywiste
równy	!=F	==I	==D	==R
różny	><F	<>I	<>D	<>R
większy	>F	>I	>D	>R
mniejszy	<F	<I	<D	<R
większy lub równy	>=F	>=I	>=D	>=R
mniejszy lub równy	<=F	<=I	<=D	<=R

Przykład

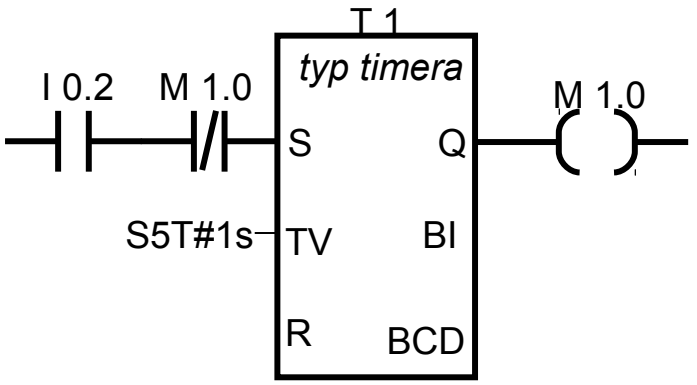
Programowanie i działanie sterownika PLC najlepiej zobrazować na przykładzie.

Problem:
Przy załączonym wejściu I 0.2 na wyjściu Q 2.5 ma być generowany sygnał taktujący o stałym i równym czasie trwania impulsu i pauzy.

Rozwiązanie:
Zadanie można rozwiązać programując dwie gałęzie:
1. Timer typu SD generujący w pamięci impuls co 1 sekundę w czasie gdy naciśnięty jest przycisk startujący (I 0.2).
2. Układ przełączający stan lampki (Q 2.5) w momencie wystąpienia impulsu.

Rozwiązanie

1



2