LG Industrial Systems

STEROWNIK PLC LG SERII MASTER-K

PODRĘCZNIK PROGRAMOWANIA

Opracowanie:

FOSTER
ul. JS Bacha 20
80-171 Gdańsk
tel. (58) 320 15 37
fax (58) 320 15 39
e-mail: foster@foster.pl
www.foster.pl

Spis treści

Rozdział 1 Program KGL WIN

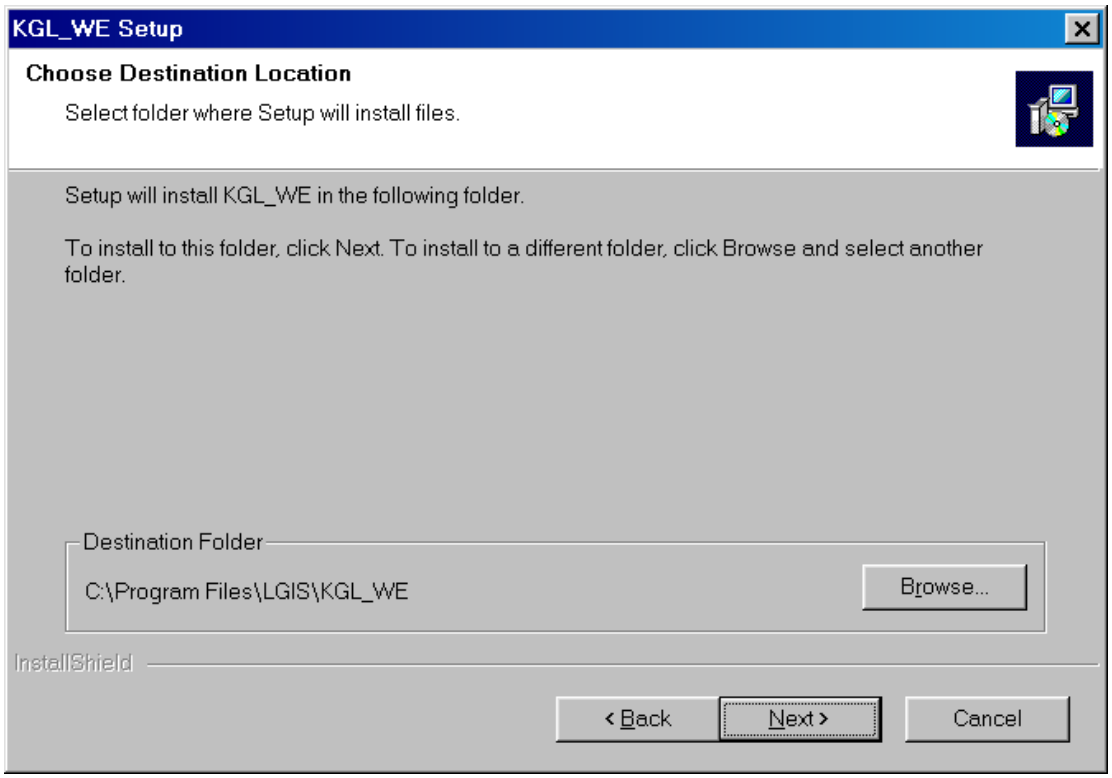
1.1 Wprowadzenie

Program KGL WIN jest narzędziem służącym do programowania wszystkich sterowników PLC LG serii Master K.
Aplikacja działająca w systemie Windows umożliwia programowanie za pomocą języka drabinek (Ladder - LD) lub listy instrukcji (Mnemonic - IL).

Program umozliwia m.in.:
- edycję programu w trybie off-line (bez obecności PLC)
- edycję on-line – przy działającym PLC
- odczyt programu ze sterownika
- debugowanie programu
- monitoring działającego programu
- konfigurację parametrów sterownika takich jak np. protokoły komunikacyjne, filtracja sygnałów wejściowych

1.1.1 Instalacja programu

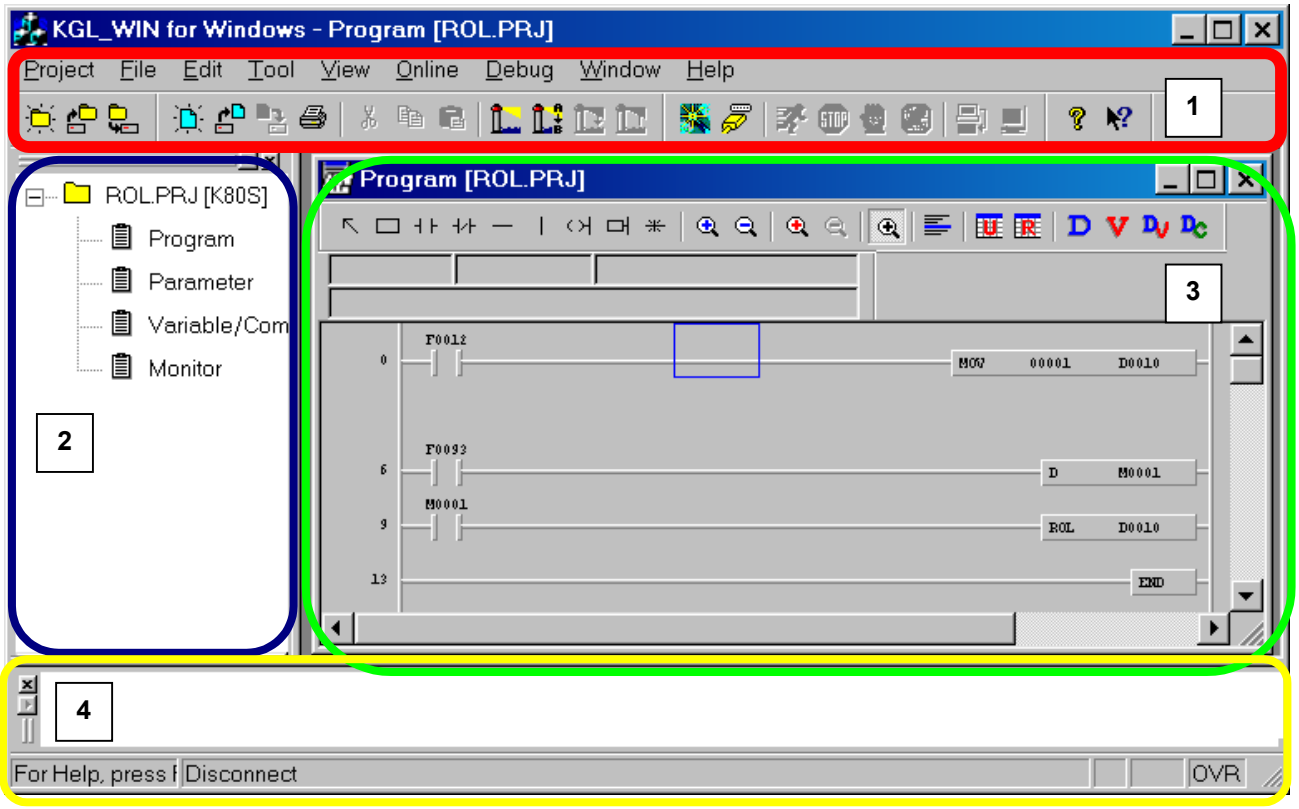
Wersja instalacyjna programu znajduje się w katalogu KGL_WIN na płycie CD-ROM. W podkatalogu DISK1 znajduje się program uruchamiający instalację (setup.exe).
Program instalacyjny prosi uzytkownika o podanie docelowej ścieżki instalacyjnej.



Po zainstalowaniu programu w menu **Start** pojawi się folder KGL_WE Application, a w nim pozycja KGL_WE uruchamiająca program.

1.1.2 Okno aplikacji KGL WIN

Okno aplikacji KGL WIN podzielone jest na cztery części.



1. Obszar Menu programu oraz paska narzędzi – umożliwia dostęp do większości funkcji programu. Przykładowo: tworzenie nowego projektu, zapis programu do PLC, drukowanie itd..

2. Obszar projektu umozliwia przechodzenie pomiędzy edycją programu, zmiennych, parametrów sterownika oraz podglądem stanu zmiennych.

3. Obszar edycji. W obszarze tym pojawiają się okna edycji programu, edycji zmiennych, edycji parametrów itp.

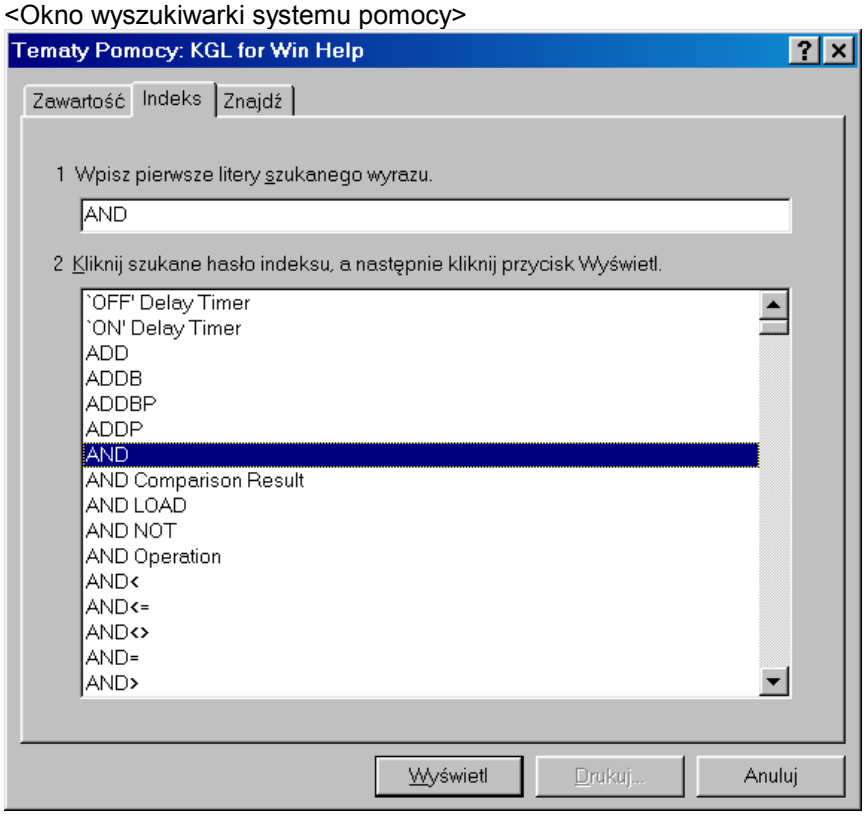
4. Obszar komunikatów – zawiera m.in. komunikaty o błędach i stanie połączenia z PLC .

1.1.3 System pomocy

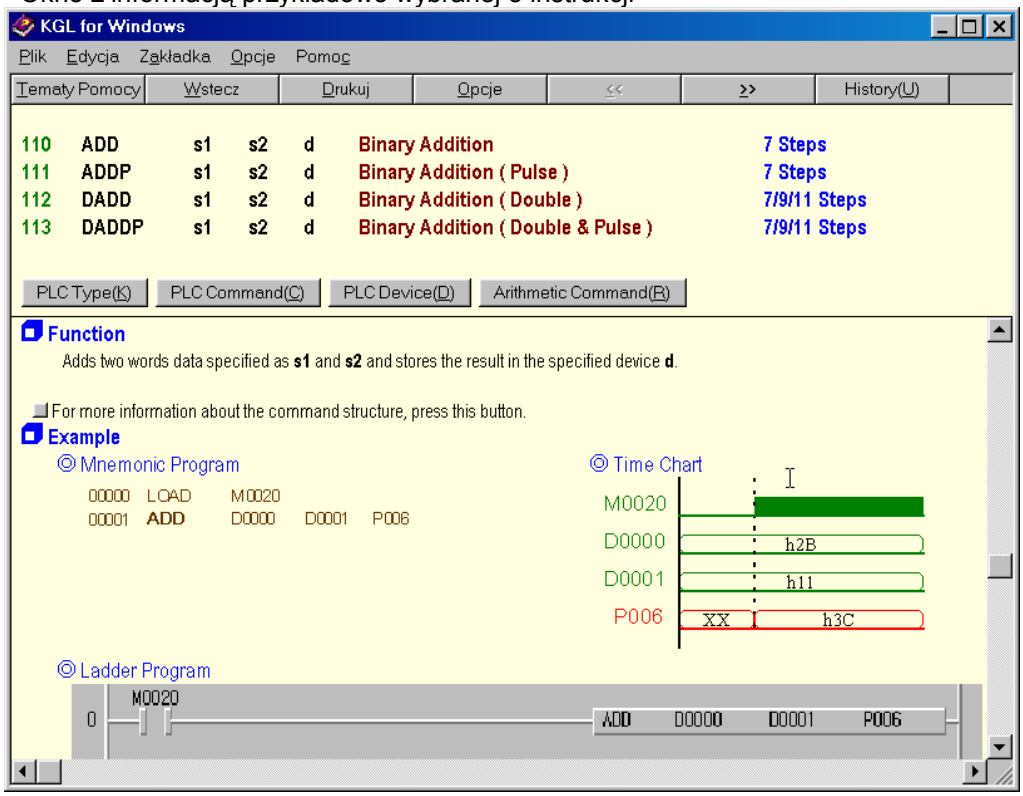
KGL WIN wyposażony jest w bogaty system pomocy zawierający wszystkie podstawe informacje dotyczące typów zmiennych, konfiguracji pamieci sterownika oraz instrukcji.

Dostęp do systemu pomocy możliwy jest z menu Help-KGLWIN Help lub z paska narzędzi  .

<Okno wyszukiwarki systemu pomocy>



<Okno z informacją przykładowo wybranej o instrukcji>

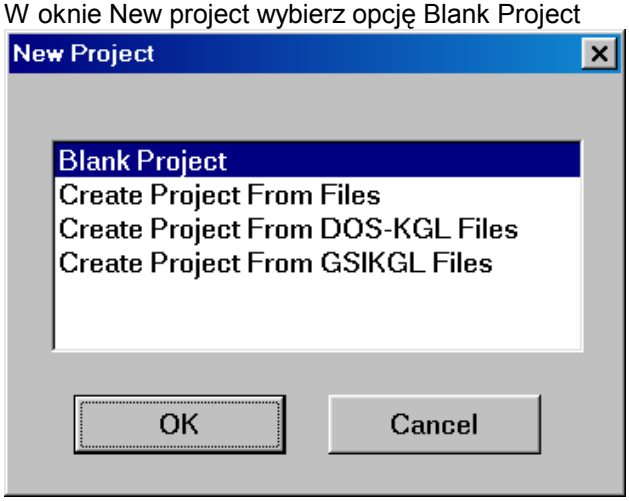


1.2 Tworzenie projektu

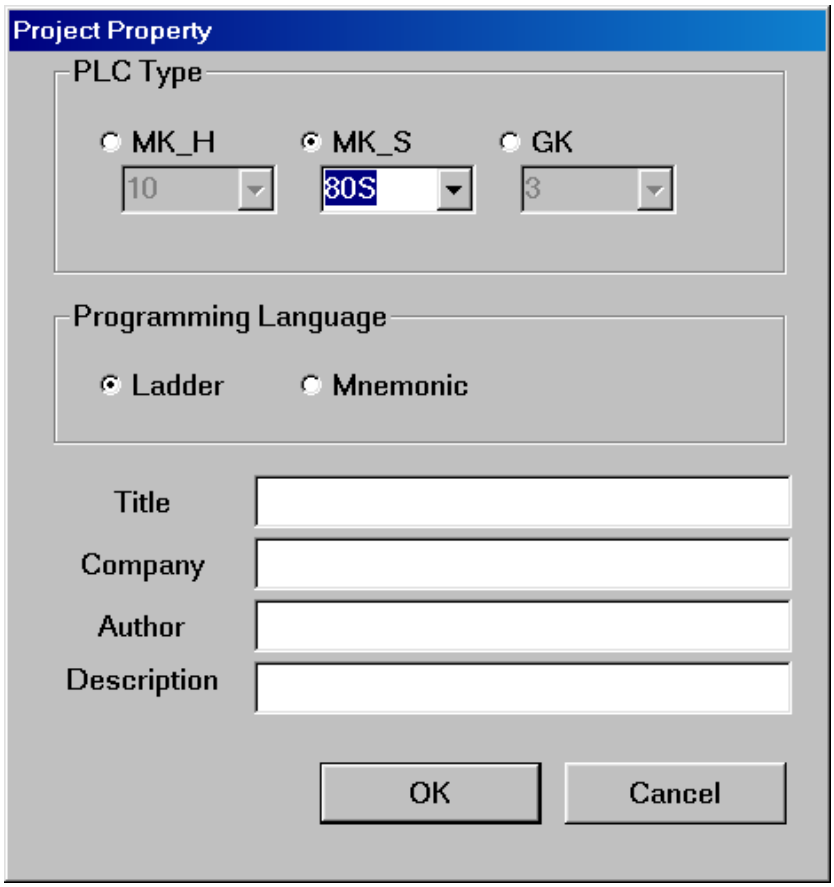
1.2.1 Tworzenie nowego projektu

Aby utworyć nowy projekt wybierz menu **Project-New project...**

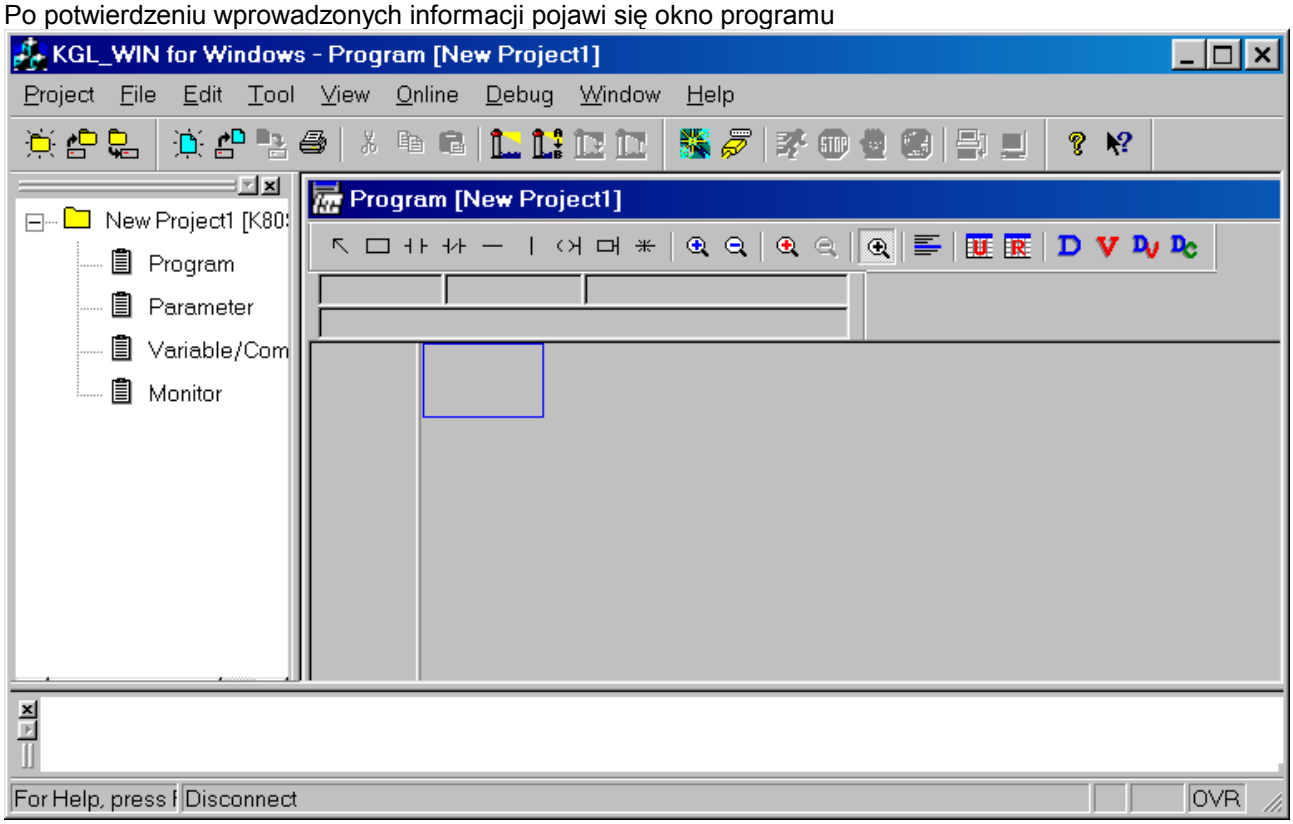
W oknie New project wybierz opcję Blank Project



Następne okno dialogowe umożliwia wybranie serii sterownika, dla którego będzie tworzony program, wybranie rodzaju edycji Ladder – drabinki, Mnemonic – lista instrukcji. Zmiany trybu edycji będzie można także dokonać w trakcie pisania programu.
Dodatkowe pola służą do wpisania informacji o autorze i programie – nie są obowiązkowe.

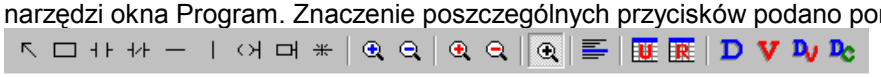


Po potwierdzeniu wprowadzonych informacji pojawi się okno programu



1.2.2 Edycja programu (język drabinkowy)

Program z języka drabinek tworzy się za pomocą kilku podstawowych „bloczków" widocznych na pasku narzędzi okna Program. Znaczenie poszczególnych przycisków podano poniżej.

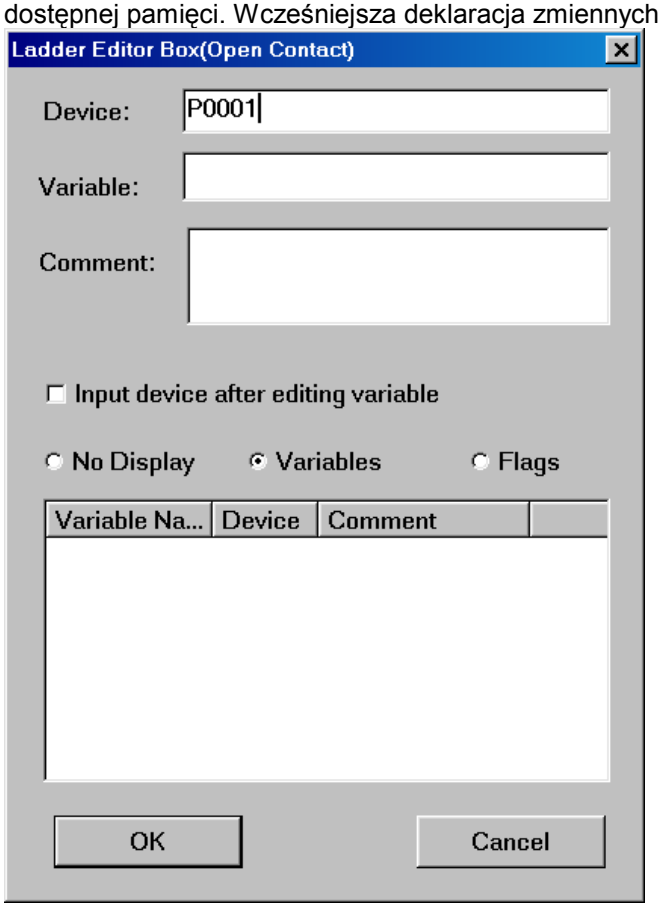


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

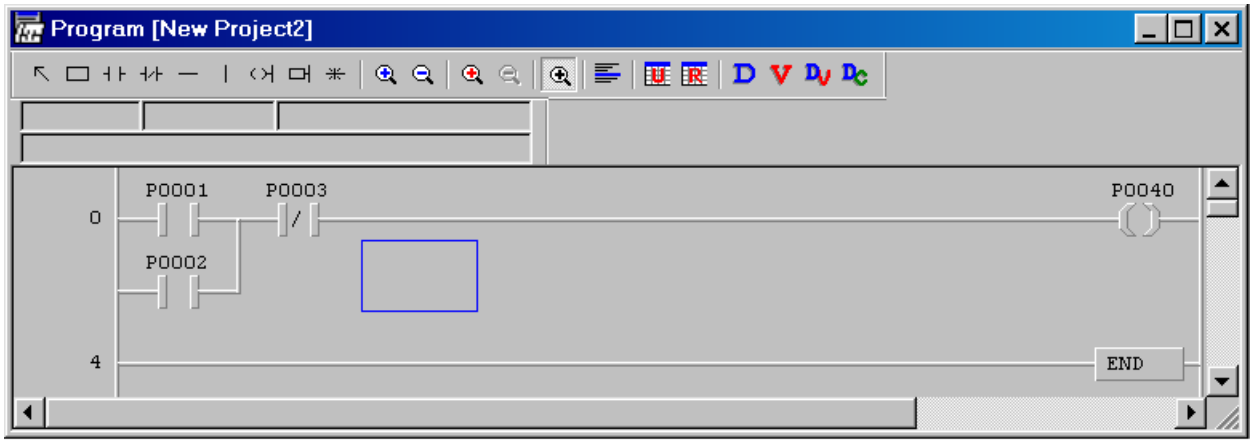
1. Podstawowy – kursor nie posiada żadnej specjalnej funkcji
2. Selekcja – umożliwia zaznaczenie fragmentu programu np. w celu kopiowania
3. Kontakt normalnie otwarty (NO)
4. Kontakt normalnie zamknięty (NC)
5. Linia pozioma – odpowiednik funkcji „and" lub przypisania
6. Linia pionowa – odpowiednik funkcji „or"
7. Cewka wyjściowa – odpowiednik zmiennej, której przypisujemy wyrażenie zapisane z lewej strony
8. Blok funkcyjny – umożliwia zapisanie za pomocą drabinek zaawansowanych funkcji np. liczników
9. Negacja
10. Powiększenie widoku programu
11. Pomniejszenie widoku programu
12. Dodanie kolumny
13. Usunięcie kolumny
14. Widok szczegółów instrukcji
15. Przejście do edycji programu w formie listy instrukcji
16. Widok użytych zmiennych
17. Widok operacji na użytych zmiennych
18. Pokazywane będą tylko adresy zmiennych
19. Pokazywane będą tylko nazwy zmiennych
20. Pokazywane będą adresy i nazwy
21. Pokazywane będą adresy i komentarze

Realizacja przykładowej funkcji logicznej: **P0040= (P0001 OR P0002) AND (NOT P0003)**

Kliknij na symbolu **-| |-** . Następnie kliknij w pierwszej kolumnie pola edycji (lub w miejscu, w którym chcesz umieścić symbol).
W oknie wprowadź adres bitu (P0001). ***UWAGA!*** KGL WIN umożliwia operację bezpośrednio na adresach dostępnej pamięci. Wcześniejsza deklaracja zmiennych nie jest obowiązkowa.



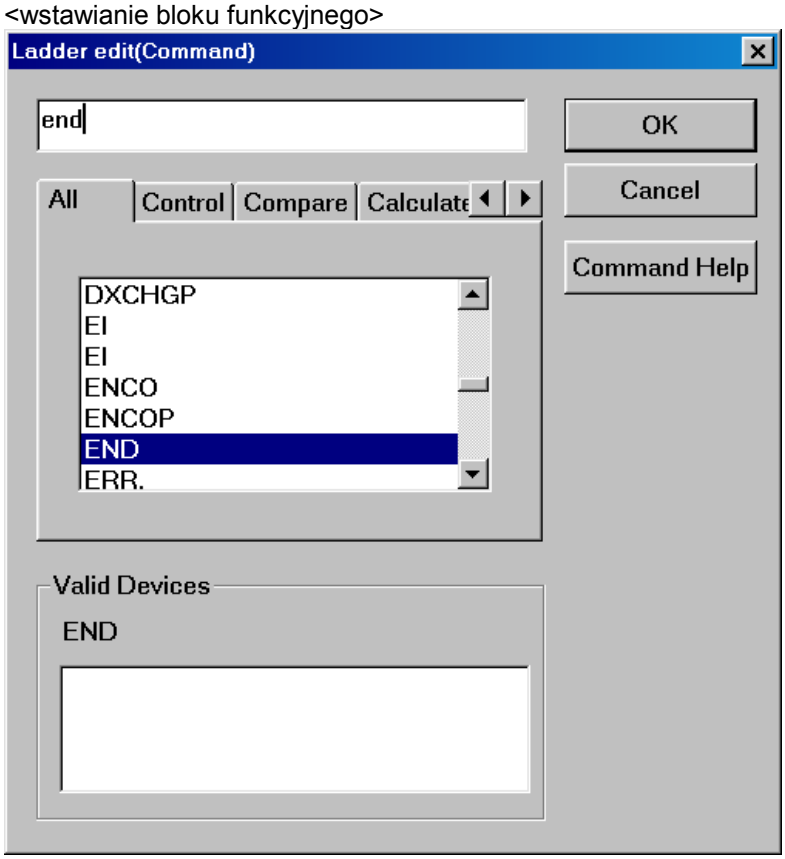
Tak samo postępuj w przypadku innych symboli przykładowo **-|/|-** oraz **–( )-**. Podana powyżej funkcja w postaci drabinek wygląda następująco:



UWAGA

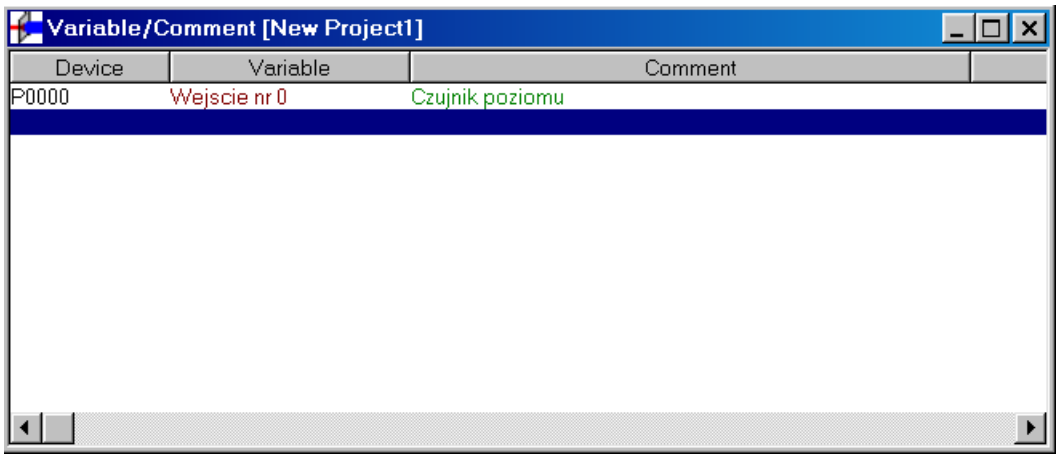
Program zawsze musi być zakończony instrukcją END. Aby wprowadzić instrukcję END kliknij na przycisk Blok funkcyjny i wpisz END.

<wstawianie bloku funkcyjnego>

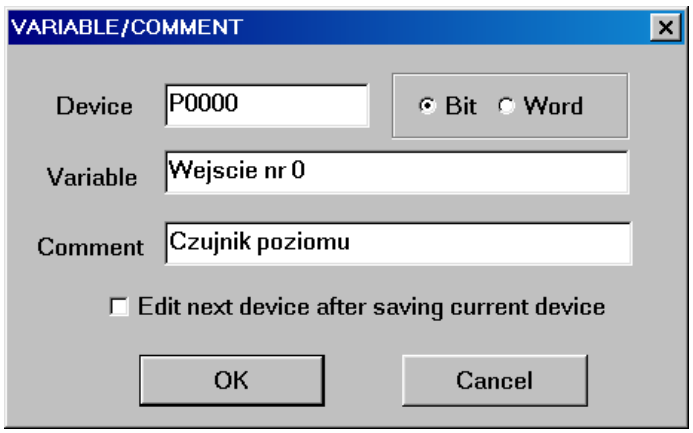


1.2.3 Edycja zmiennych

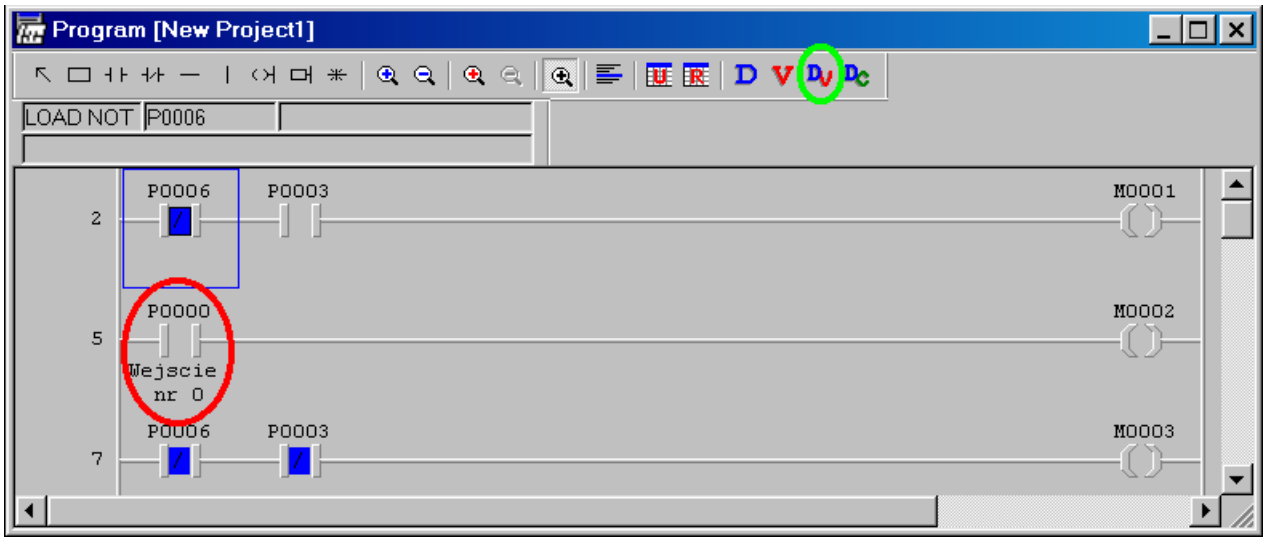
KGL WIN umożliwia edycję programu bez wcześniejszej definicji zmiennych – operacje zapisywane są z wykorzystaniem bezpośrednich adresó odpowiednich komórek pamięci. W celu uzyskania przejrzystości programu zaleca się zdefiniowanie listy występujących w nim zmiennych. Aby tego dokonać kliknij na gałęzi **Variable/Comment** drzewka projektu (z lewej strony okna aplikacji).



Po podwójnym kliknięciu na wolnej pozycji okna zawierającego listę zmiennych pojawi się okienko edycji zmeinnej.



Wprowadzone nazwy zmiennych mogą być widziane w podczes edycji programu po kliknięci przycisku **Device+Variable** na pasku narzędziowym okna edycji.

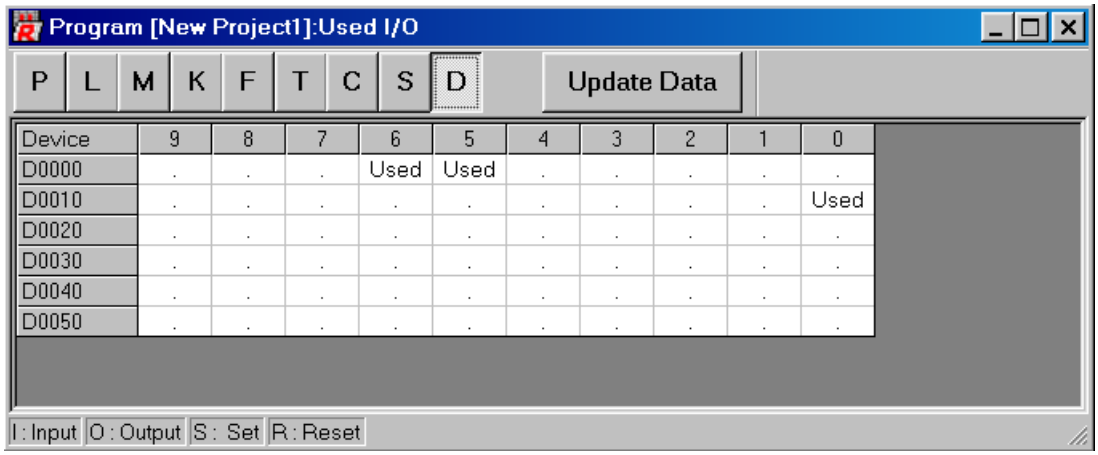


1.2.4 Lista użytych w programie komórek pamięci

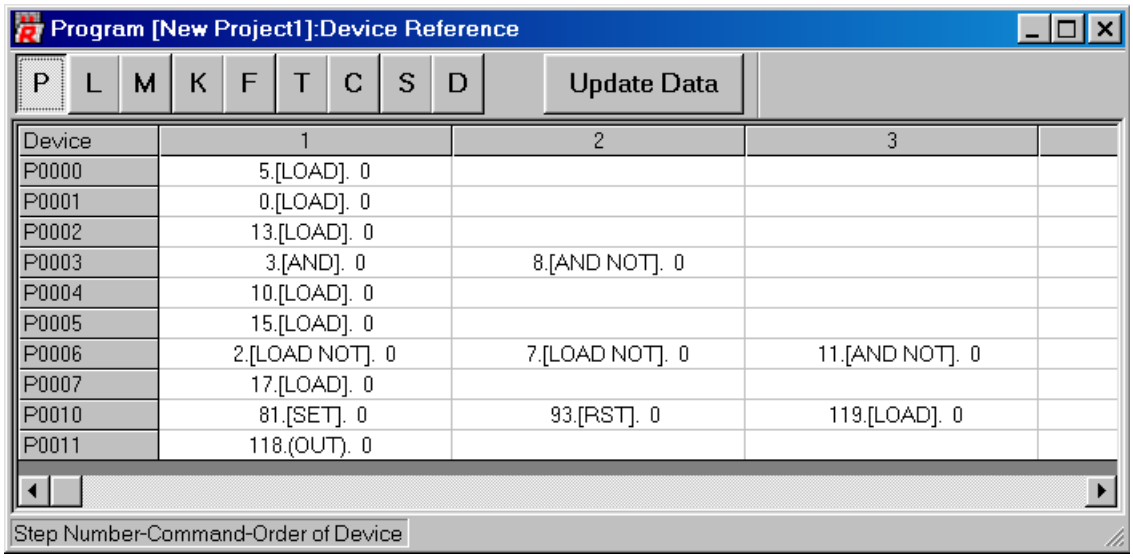
Bardzo pożytecznym narzędziem udostępnianym przez KGL WIN jest lista uzytych w programie komórek pamięci oraz lista operacji wykonanych na tych komórkach. Aby zobaczyć pierwszą z nich kliknij na przycisku **Used I/O** paska narzędzi okna edycji.

Druga z nich dostępna jest po kliknięciu na przycisk **Device Reference**.

W oknie listy użytych zmiennych wybieramy z paka narzędzi typ komórek, których wykorzystanie nas interesuje.

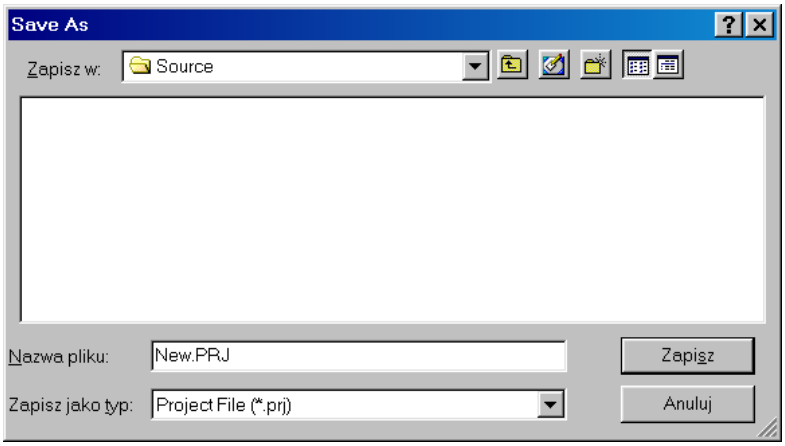


Podobnie postępujemy w oknie operacji z amiennych. Okno ta dostarcza nam informacji o tym, w którym kroku programu wystąpiła operacja, jaki był jej typ i którym operandem jest dana zmienna.



1.2.5 Zapis projektu na dysku

Aby zapisać projekt na dysku wybierz menu **Project- Save** lub **Project Save As...** i podaj odpowiednią nazwę.



1.3 Połączenie ze sterownikiem PLC

1.3.1 Kabel połączeniowy

Sterowniki LG łączone są z komputerem za pomocą złącza RS232 (COM1, COM2 w komputerze). Każdy sterownik posiada port loaderowy. Serie K80S, K120S, K200S, oraz K300S posiadają żeńskie gniazdo 9 pinowe, natomiast seria K10S1 gniazdo typu PS2.

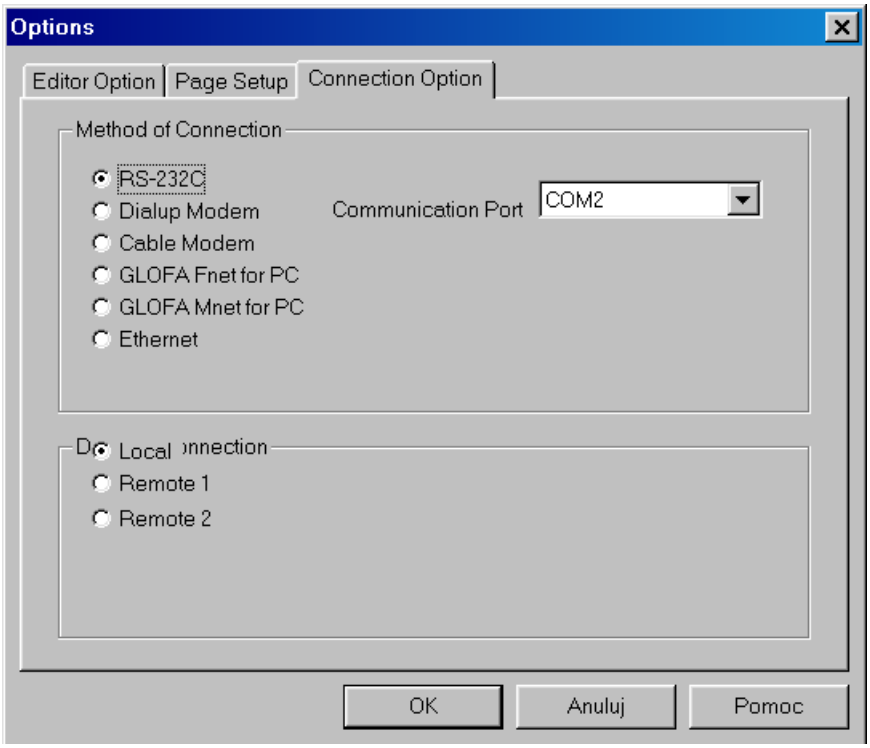
<Układ połączeń w kablu komputer PC- PLC LG (loader)>

kod	Komputer PC 9 pin	PLC – LG 9 pin (lub PS2)	kod
RxD	2	3	TxD
TxD	3	2	RxD
GND	5	5	GND

kod	Komputer PC 25 pin	PLC – LG 9 pin (lub PS2)	kod
RxD	3	3	TxD
TxD	2	2	RxD
GND	7	5	GND

1.3.2 Parametry połączenia

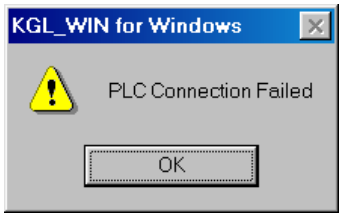
Aby ustalić parametry połączenia wybierz menu **Project- Options**. Na zakładnce **Connection Option** wybierz RS232C oraz podaj nazwę portu w twoim komputerze, którego używasz do połączenia ze sterownikiem.



1.3.3 Ustanawianie połączenia ze sterownikiem

Aby połączyć się ze sterownikiem wybierz z menu **Online-Connect** lub naciśnij przycisk **Connect** na pasku narzędzi.

Jeżeli parametry połaczenia oraz fizyczne połączenie (np. kabel) są poprawne w obszarze komunikatów aplikacji pojawi się informacja **Connected With PLC**. Przycisk Connect na pasku narzędzi otrzyma nowe znaczenie – Disconnect (Rozłącz) W przeciwym przypadku pojawi się monit:



1.3.4 Rozłączanie

Aby zamknąć połączenie ze sterownikiem wybierz z menu **Online-Disconnect** lub naciśnij przycisk **Disconnect** na pasku narzędzi.

W obszarze komunikatów aplikacji pojawi się informacja **Disconnected With PLC**.

1.3.5 Zapis programu w PLC

Po połączeniu z PLC wybierz menu **Online-Download** lub naciśnij przycisk **Download** na pasku narzędzi.

1.3.6 Uruchomienie programu

Aby sterownik rozpoczął wykonywanie programu niezbędne jest przestawienie go tryb RUN. W tym celu Naciśnij przycisk RUN na pasku narzędzi.

1.3.7 Skrócona procedura wgrywania programu

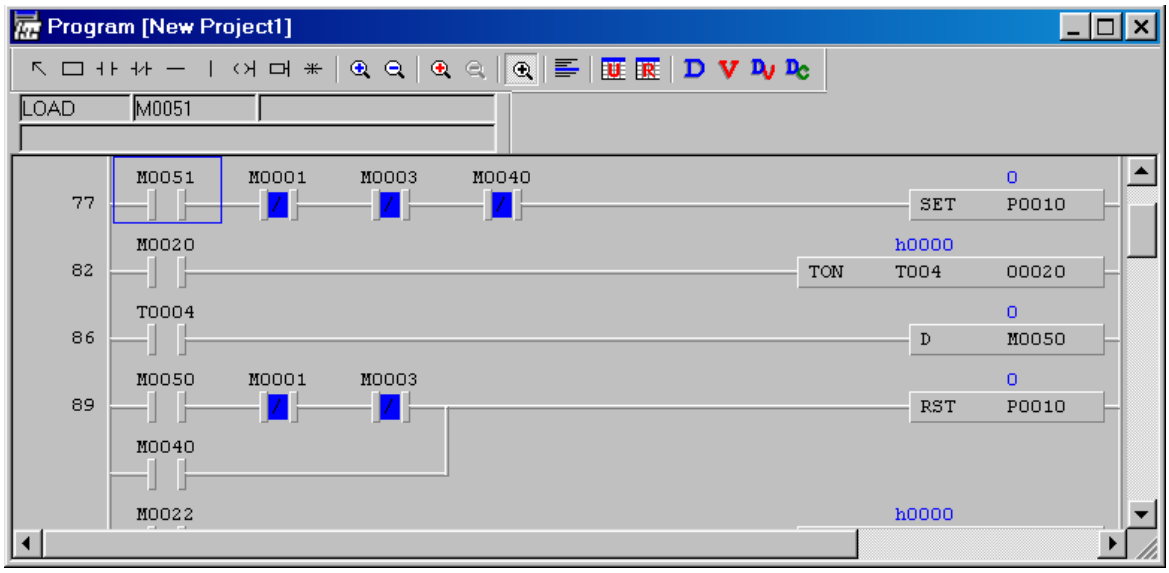
KGL WIN udostępnia skróconą procedurę wgrywania programu do PLC. Służy do tego polecenie **Connect+Download+Run+Monitor** z menu **Online** lub przycisk **Connect+Download+Run+Monitor** z paska narzędzi.

Po wybraniu tego polecenia KGL WIN nawiązuje połącznie z PLC, zapisuje w jego pamięci program, ustawia PLC w tryb RUN i włącza monitorowanie stanu zmiennych w PLC.

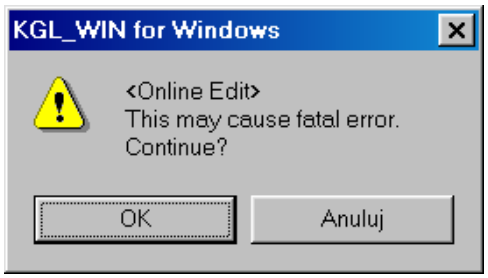
1.4 Praca online

1.4.1 Monitor

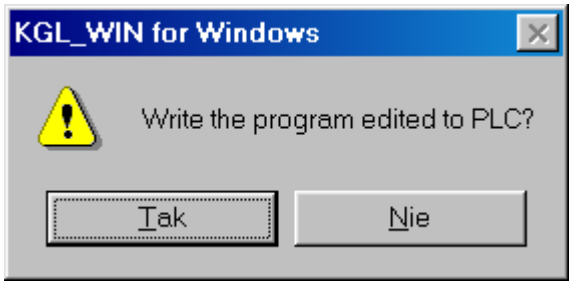
Aby przejśc do pracy w trybie monitor połącz się z PLC i wybierz opcję Monitor z menu Online.
W trybie tym KGL WIN umożliwia podgląd stanu zmiennych programu, a także ich modyfikację.



W trybie tym możliwa jest także modyfikacja programu. Przy próbie wprowadzenia zmian do programu w trybie Online pojawi się ostrzeżenie o mozliwości wywołania poważnego błędu w działającym układzie automatyki.

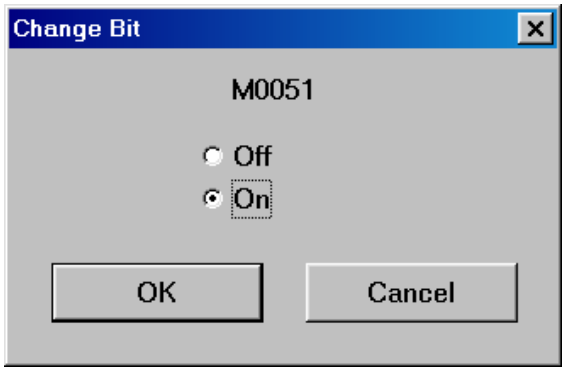


Przed zapisaniem zmodyfikowanego programu do PLC KGL WIN zapyta raz jeszcze czy zapisać zmodyfikowany program w pamięci sterownika:



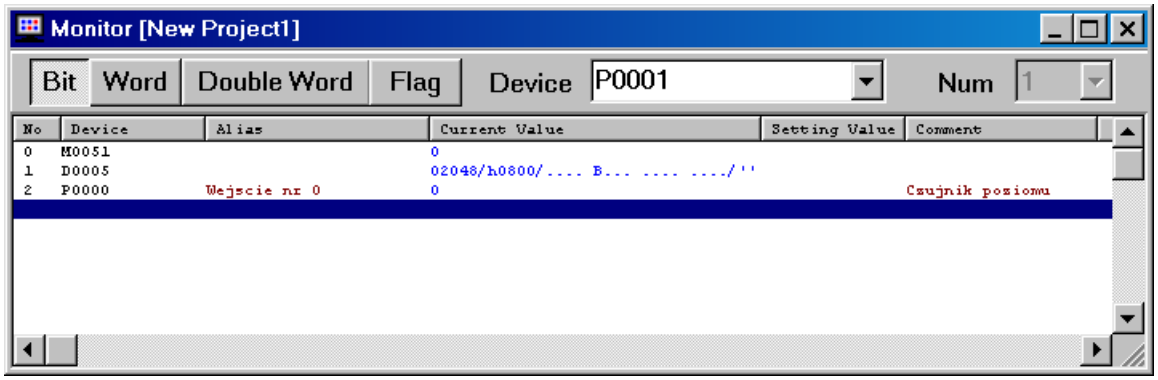
1.4.2 Zmiana wartości zmiennej

Zmiany wartości wybranej zmiennej dokonujemy kilkając prawym klawiszem myszy na symbolu odnoszącym się do tejże zmiennej. Z podręcznego menu wybieramy opcję **Change Current I/O**. W oknie dialogowym wprowadzamy nową wartość.



1.4.3 Montorowanie wartości wielu zmiennych – okno Monitor

Kliknij na gałęzi Monitor drzewka projektu. W pasku okna monitora wybierz typ zmiennej, którą chcesz dodać do listy. W polu edycji (Device) wpisz jej adres i potwierdź klawiszem ENTER.



Rozdział 2 Specyfikacja

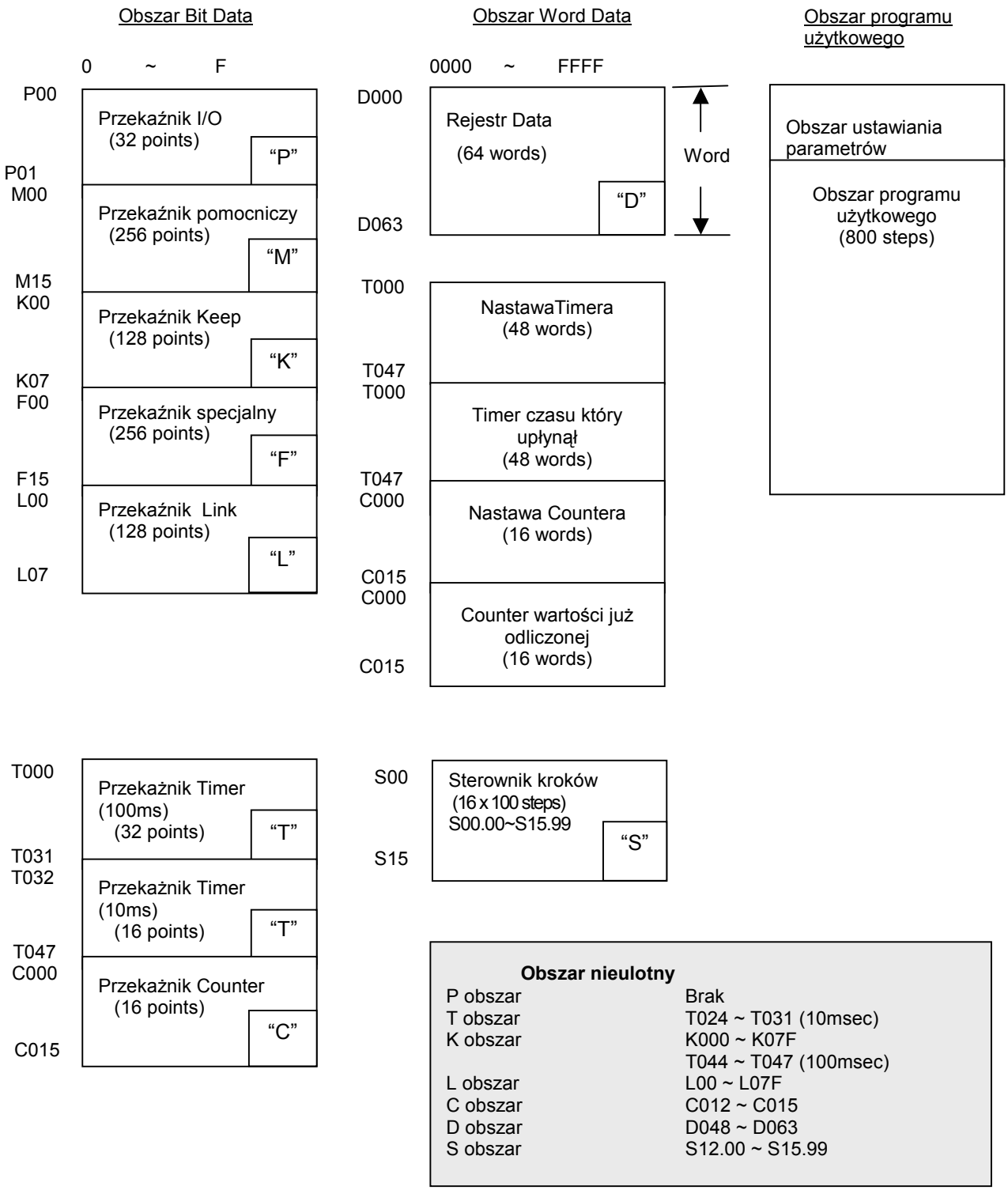
2.1 Dane techniczne

K10S1 / K80S / K200S / K300S

Pozycja		K10S1	K80S	K200S	K300S
Sposób wykonywania programu		Cykliczne wykonywanie programu			
Sposób sterowania I/O		Bezpośredni (Odświeżanie)			
Liczba instrukcji	Podstawowe	30			
	Application	154	218		
Max liczba wejść/wyjść		14	10-80	384	5121
Szybkość procesora		3.2 ~ 7.6μs/krok	0.5 μs/krok		0.2 μs/krok
Liczba kroków programu		800	7000		15 000
P (I/O relay)		P0000 ~ P001F (32 points)	P0000 ~ P015F (256 points)		P0000 ~ P035F (512 points)
M (Auxiliary relay)		M0000 ~ M015F (256 points)	M0000 ~ M191F (3072 points)		
K (Keep relay)		K0000 ~ K007F (128 points)	K0000 ~ K031F (512 points)		
L (Link relay)		L0000 ~ L007F (128 points)	L0000 ~ L063F (1024 points)		
F (Special relay)		F0000 ~ F015F (256 points)	F0000 ~ K063F (1024 points)		
T (Timer relay)	100ms	T000 ~ T031 (32 points)	T000 ~ T191 (192 points)		
	10ms	T032 ~ T047 (16 points)	T192 ~ T255 (64 points)		
C (Counter relay)		C000 ~ C015 (16 points)	C000 ~ C255 (255 points)		
S (Step controller)		S00.00 ~ S15.99 (16×100 steps)	S00.00~ S99.99 (100×100 steps)		
D (Data register)		D0000 ~ D0063 (64 words)	D0000 ~ D4999 (5000 words)		
The range of integer		16 bit : – 32768 ~ 32767 32 bit : – 2147483648 ~ 2147483647			
Typy timerów		Opóźnienie On, Opóźnienie Off, Akumulacyjny, Monostabilny, Powtarzalny (5 typów)			
Typy liczników		Up, Down, Up-down, Ring counter (4 typy)			
Język programowania		Mnemonic, Ladder diagram			
Funkcje specjalne		Real time clock (Zegar czasu rzeczywistego), High speed counter(Szybki licznik), RS-485 communication			

2.2 Mapa konfiguracji pamięci

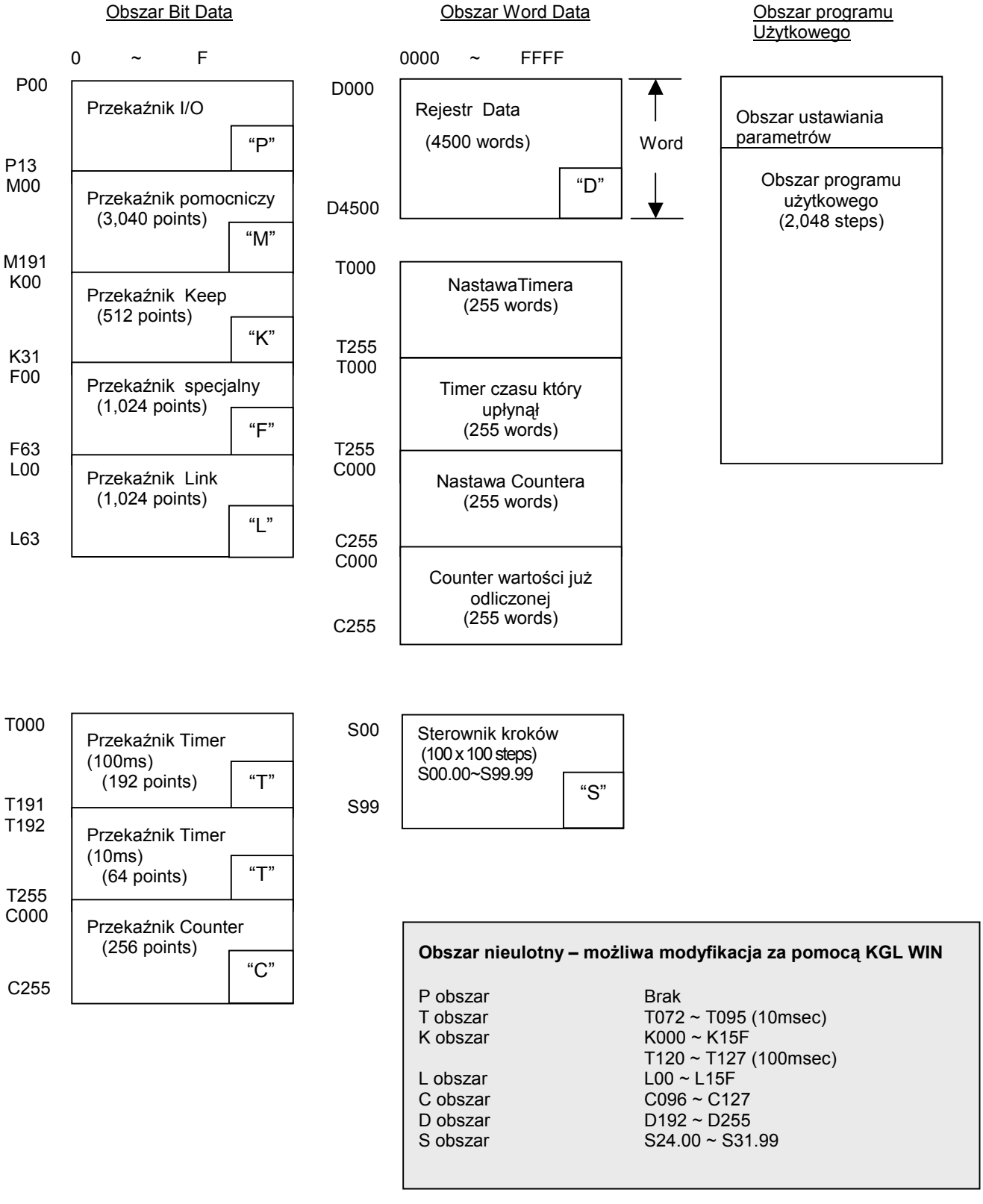
2.2.1 K10S1



Obszar nieulotny

P obszar	Brak
T obszar	T024 ~ T031 (10msec)
K obszar	K000 ~ K07F
	T044 ~ T047 (100msec)
L obszar	L00 ~ L07F
C obszar	C012 ~ C015
D obszar	D048 ~ D063
S obszar	S12.00 ~ S15.99

2.2.2 K80S



Obszar nieulotny – możliwa modyfikacja za pomocą KGL WIN

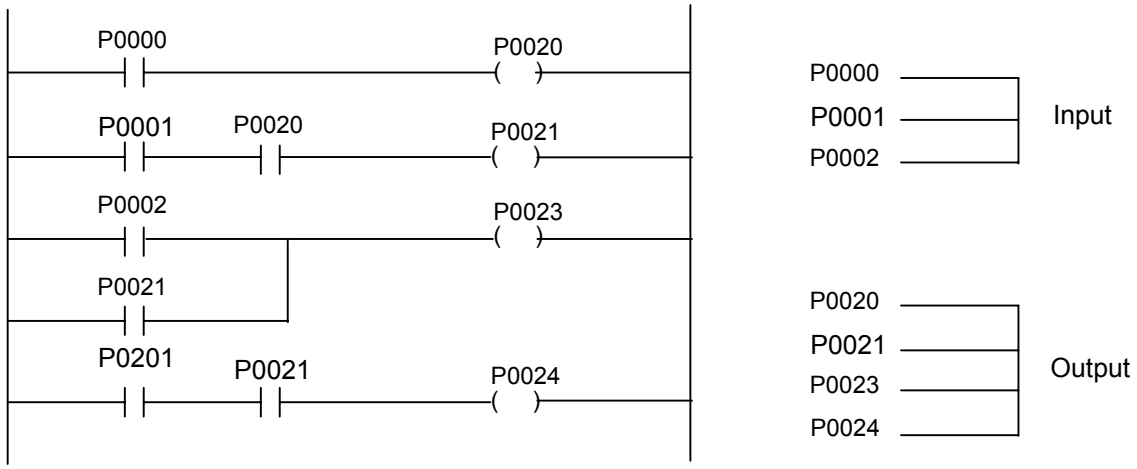
P obszar	Brak
T obszar	T072 ~ T095 (10msec)
K obszar	K000 ~ K15F
	T120 ~ T127 (100msec)
L obszar	L00 ~ L15F
C obszar	C096 ~ C127
D obszar	D192 ~ D255
S obszar	S24.00 ~ S31.99

2.3 Obiekty pamięci serii MASTER-K

2.3.1 Obszar input / output : P

Obiekty P są używane do przemieszczania danych pomiędzy PLC CPU a urządzeniami zewnętrznymi.
Obiekty input zachowują dane ON/OFF przychodzące z urządzeń zewnętrznych (np przyciski, przełączniki wyboru, krańcówki, przełączniki cyfrowe, etc.) na moduł input. Dane wejściowe są używane przez program jako styki (NO[1] i NC) oraz jako dane źródłowe dla instrukcji basic(podstawowych) i application.
Obiekty output są używane do wystawiania na module output wyników operacji programu do urządzeń zewnętrznych (np cewki, styczniki, lampki sygnałowe, wskaźniki cyfrowe). Dla obiektów output dostępne są tylko styki typu NO.
Nadmiarowe obiekty P które nie są podłączone do urządzeń zewnętrrznych, mogą być użyte w taki sam sposób jak przekaźniki pomocnicze M.

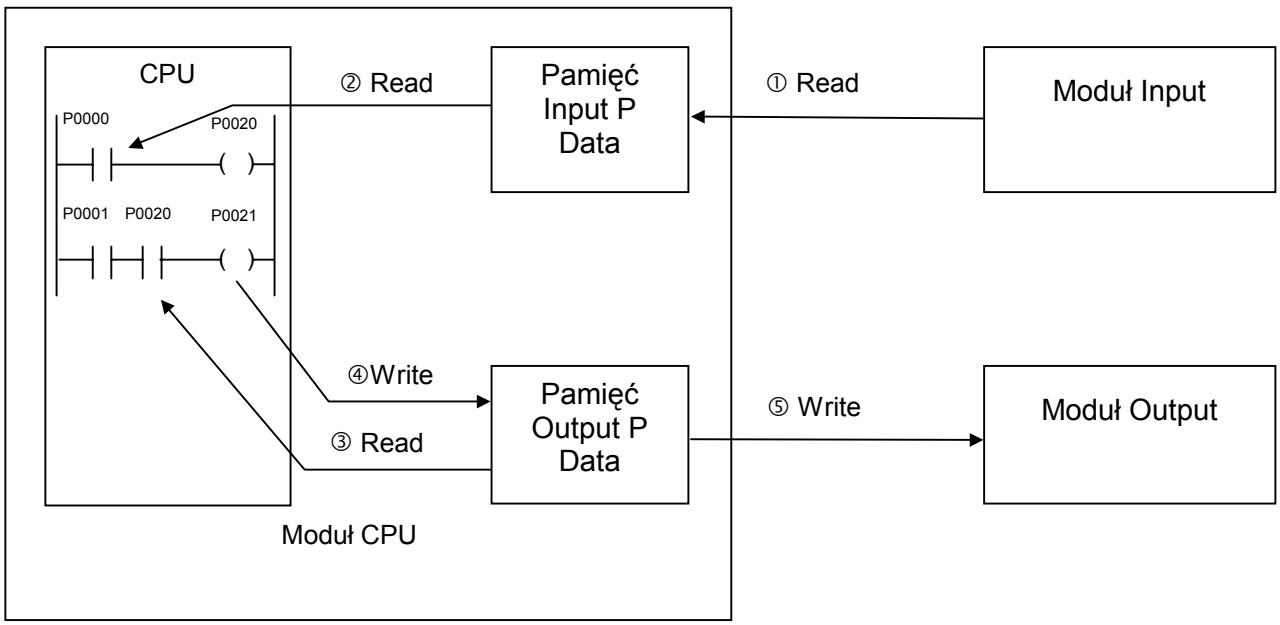
< Rys. 1. Przykład konfiguracji input/output >



Sygnały wejściowe są zapamietywane w pamięci input data w momencie poprzedzającym wykonanie każdego scan'u. Pamięć input data jest używana do wykonywania sekwencyjnie operacji programu. Wyniki operacji są sukcesywnie wystawiane do pamięci output data. Dane z pamięci output data są wystawiane na moduł output po wykonaniu instrukcji END. Proszę się upewnić, że nie ma konfliktu pomiędzy input a output w programie użytkowym, ponieważ seria MASTER-K używa obszaru P wspólnie dla input i output.

[1] NO : **N**ormally **O**pen contact, NC : **N**ormally **C**losed contact

< Rys. 2. Przepływ danych input / output w modzie odświeżania >



- Odświeżenie Input
  Dane wejściowe są czytane (①) z modułu input module przed wykonaniem step 0 i zapamiętane w pamięci input data.
- Gdy jest wykonywany rozkaz input contact(wprowadzenie styku):
  Dane wejściowe są czytane (②) z pamięci input data i użyte do wykonania sekwencji programu.
- Gdy jest wykonywany rozkaz output contact(wyprowadzenie styku):
  Dane wyjściowe są czytane (③) z pamięci output data i użyte do wykonania sekwencji programu.
- Gdy wykonywane jest wystawianie instrukcji OUT:
  Wynik operacji (④) jest zapamiętywany w pamięci output data.
- Odświeżenie Output
  Dane (⑤) z pamięci output data są wystawiane do modułu output po wykonaniu instrukcji END.

2.3.2 Przekaźnik pomocniczy : M

Obszar M jest wewnętrznym przekaźnikiem używanym przez PLC CPU i nie może on być bezpośrednio połączony z urządzeniami zewnętrznymi. Cały obszar M oprócz obszaru latched(zatrzaskujący się) będzie wyczyszczony do 0, gdy PLC zostanie włączony lub wejdzie w mod RUN. W K80S/K200S / K300S / K1000S, użytkownik może zmienić obszary latched przez zmianę parametrów(parameter setting).

2.3.3 Przekaźnik Keep : K

Obszar K funkcjonuje tak samo jak obszar M. Jednakże wyniki operacji są zachowane, przez zmianę parametrów(parameter setting).Obszar K może być wyczyszczony przy użyciu następujących metod;

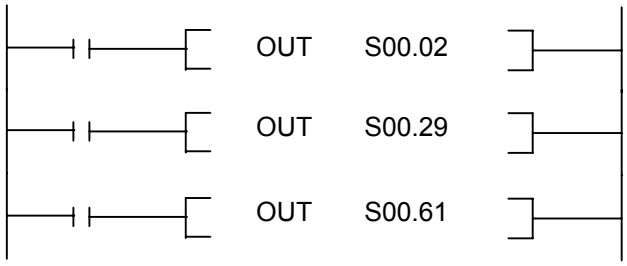
- Wstaw procedurę inicjalizacji do sekwencji programu.
- Uruchom funkcję wyczyszczenia danych przez hand-held loader (KLD-150S)
- Uruchom funkcję wyczyszczenia danych przez PC graphic loader (KGL-WIN)

2.3.4 Przekaźnik Link : L

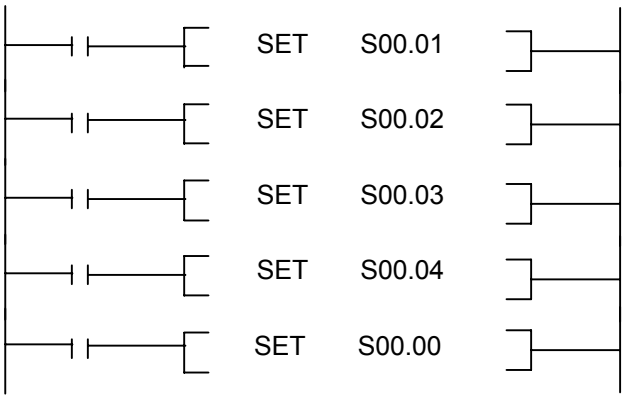
Obszar L jest wewnętrzną pamięcią używaną do przechowywania danych lub do komputerowych systemów link. Jeżeli w systemie PLC nie ma zamontowanych modułów link to obszar ten może być używany tak samo jak obszar M. W K200S / K300S / K1000S, istnieje możliwość zmiany zakresu obszaru link przez zmianę parametrów(parameter setting). Szczegóły użycia obszaru L, patrz lista przekaźników link w appendix oraz
w podręczniku computer link.

2.3.5 Sterownik kroków : S

Obszar S może być użyty dla dwóch rodzajów sterownika kroków, zgodnie z instrukcjami - OUT lub SET.
Jeżeli użyto instrukcji OUT, obszar S funkcjonuje w priorytecie „ostatnio użyty”. W przeciwnym przypadku funkcjonuje on jako sterownik sekwencyjny. (Szczegóły użycia - patrz rozdział 4)
Gdy PLC zostanie włączony lub wejdzie w mod RUN, to obszar S zostanie zainicjowany na krok (Sxx.00) oprócz obszaru latch wyznaczonego przez nastawę parametrów(parameter setting).



W tej samej grupie, włączony ostatnio ma proiorytet.



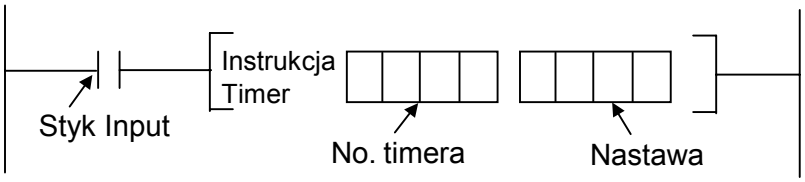
Sterowanie sekwencyjne oznacza, że dany proces może być wykonany tylko wtedy gdy poprzedni został już zakończony.

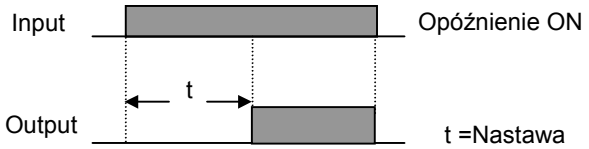
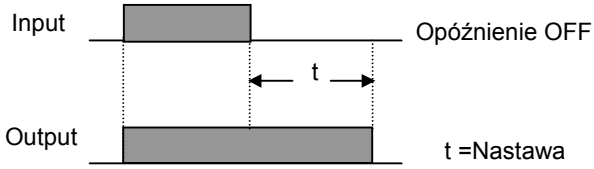
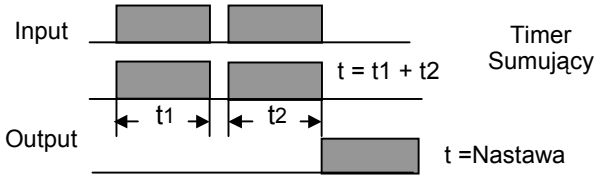
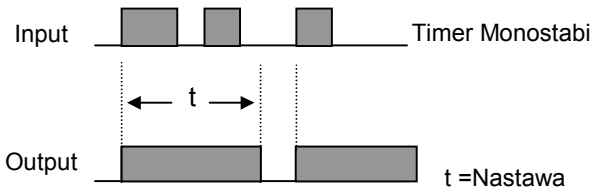
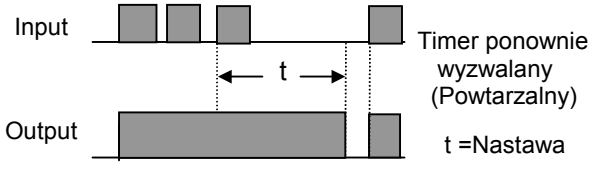
Warunek wyczyszczenia (Sxx.00) może być wykonany w każdym czasie, sekwencji procesu.

2.3.6 Przekaźnik Timer : T

Seria MASTER-K posiada timery 100msec i 10msec (K120S także 1msec). Działanie timerów jest różne w zależności instrukcji timera (TON, TOFF, TMR, TMON, TRTG). Maksymalna nastawa timera jest hFFFF hexadecymalnie lub 65535 dziesiętnie. Rysunki poniżej pokazują działanie timerów w zależności od ich instrukcji.

< Rys. 3. Typy timerów i ich działanie >

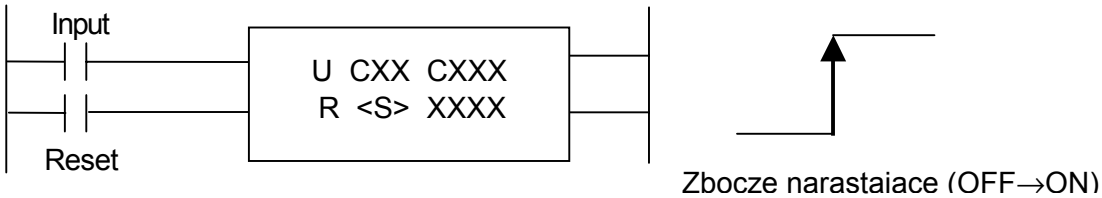


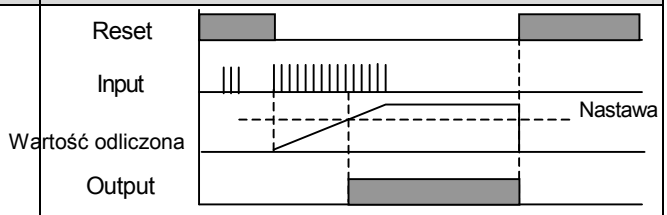
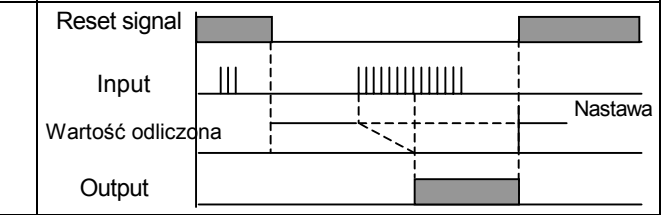
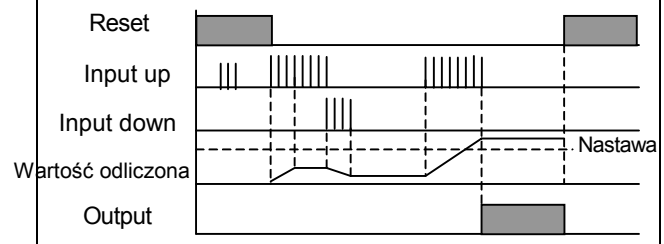
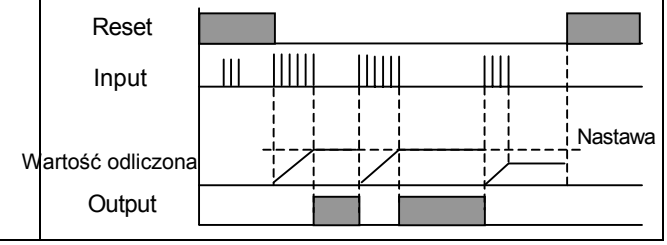
Instrukcja timera	Opis	Działanie	Przebiegi czasowe
TON	ON Delay	Narastająco	
TOFF	OFF Delay	Opadająco	
TMR	Accumulation ON Delay	Narastająco	
TMON	Monostable	Opadająco	
TRTG	Retriggerable	Opadająco	

2.3.7 Przekaźnik Counter : C

Licznik liczy narastające zbocza impulsów. Zliczanie odbywa się w momencie zmiany na input z OFF na ON. Seria MASTER-K posiada 4 instrukcje licznika: CTU, CTD, CTUD i CTR. Maksymalna wartość nastawy licznika wynosi hFFFF (= 65535). Praca liczników została pokazana poniżej.

< Rys. 4. Typy i działanie instrukcji liczników >



Symbol instrukcji	Typ	Sposób liczenia	Sygnał input	Przebiegi czasowe
CTU	Up Counter	Increment	1	
CTD	Down counter	Decrement	1	
CTUD	Up/Down Counter	Increment / Decrement	2	
CTR	Ring counter	Increment	1	

2.3.8 Rejestr Data D

Obszar D jest używany do zapamiętywania danych numerycznych. Każdy rejestr danych zawiera 16 bitów (1 słowo) które jest podstawową jednostką do zapisu i odczytu.
Dla instrukcji o podwójnej długości słowa, numer rejestru danych przeznaczony jest dla 16 bitów młodszych, natomiast numer rejestru danych +1, przeznaczony jest dla 16 bitów starszych.

Przykład)

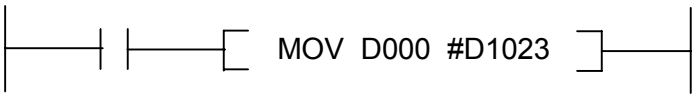
High 16 bits	Lower 16 bits
D51	D50
h1234	h5678

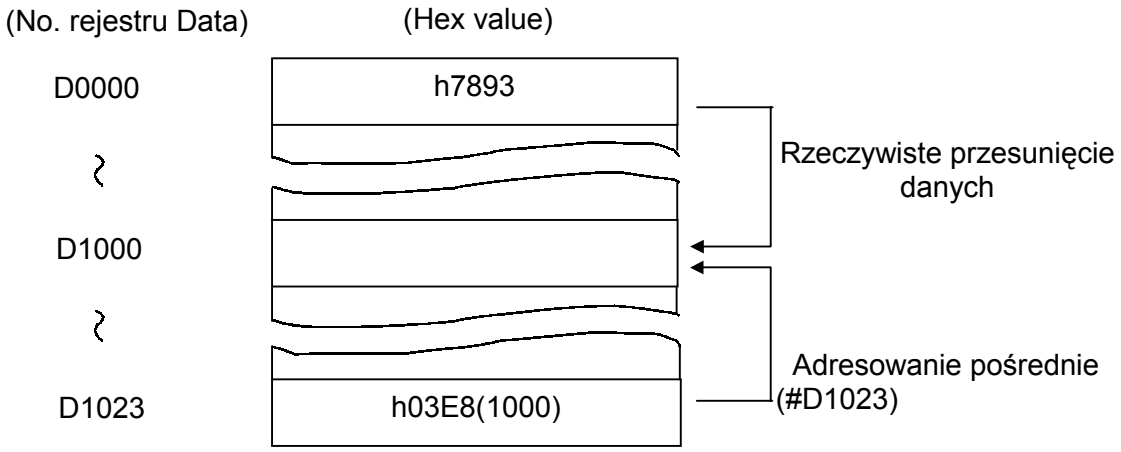
Obszar D oprócz obszaru latch wyznaczonego przez nastawę parametrów, zostanie wyczyszczony na 0, gdy PLC zostanie włączony lub wejdzie w mod RUN.

2.3.9 Pośrednie wyznaczenie rejestru data : #D

#D jest używany do pośredniego adresowania obszaru D. Zawartość rejestru data oznaczonego symbolem '#' wskazuje rzeczywisty adres rejestru data pod którym zapamiętywany jest wynik operacji. Jeżeli #D jest używana w związku z instrukcją o podwójnym słowie, numer rejestru danych #D przeznaczony jest dla 16 bitów młodszych, natomiast numer rejestru danych +1, przeznaczony jest dla 16 bitów starszych.

Przykład)





Uwagi

Jeżeli wartość danych w rejestrze # D przekroczy zakres adresu obszaru D, to zostanie ustawiona flaga błędu(F110), a wykonywana instrukcja zostanie zignorowana.

2.3.10 Przekaźnik specjalny : F

Obszar F jest przeznaczony wyłącznie do odczytu i użytkownik nie może zmienić zawartości tego obszaru.
Szczegóły – patrz tabela przekaźnik Fw rozdziale appendix.

2.3.11 Przekaźnik specjalny M / L : M / L

Niektóre przekaźniki M lub L są zarezerwowane do specjalnych zastosowań. Patrz lista przekaźników specjalnych w rozdziale appendix i bądź ostrożny w używaniu obszarów M lub L w programie.

2.3.12 Rejestr Special data : D

Niektóre rejestry data są zarezerwowane do specjalnych zastosowań. Rejestry te różnią się w zależności od typu CPU. Patrz lista rejestrów specjalnych w rozdziale appendix i bądź ostrożny w używaniu tych rejestrów w programie.
Dla sterownika Master K80S obszar rejestrów specjalnych zawiera komórki D4500 - D4999.

2.4 Ustawianie parametrów

2.4.1 Nastawa Watchdog timera

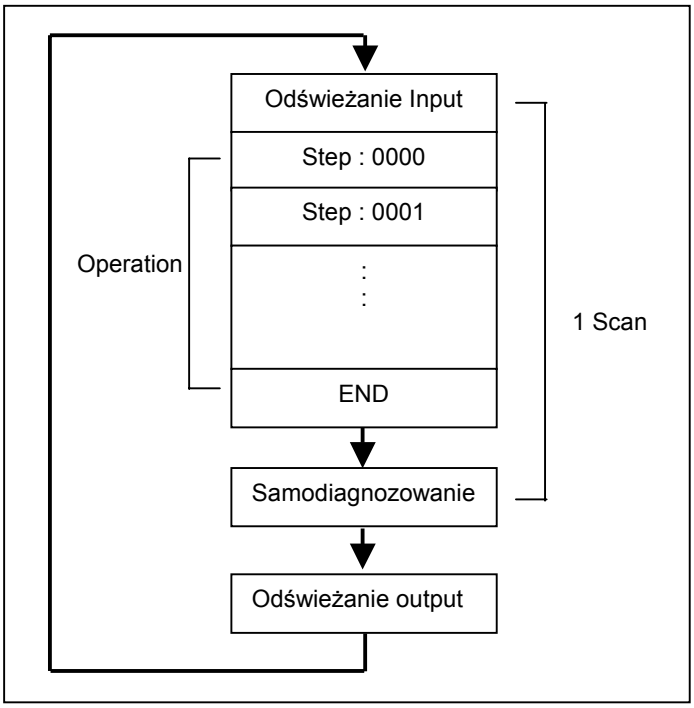
(Tylko dla K200S / K300S / K1000S)
Zakres nastawy : 10msec ~ 6000msec
Jednostki nastawy : 10msec
Wartość pierwotna watch dog timera wynosi 200msek. Watch dog timer sterowników K10S1, K10S, K30S, oraz K60S jest ustawiony na stałe na 200msek.

2.5 Praca CPU

2.5.1 Praca cykliczna

W pracy cyklicznej jest powtarzana seria operacji. CPU powtarza operacje jak pokazano poniżej.

Rys. 2-3 Praca CPU



CPU odświeża input data, następnie wykonuje sekwencję programu przechowywaną w pamięci wewenętrznej, poczynając od step 0 do instrukcji END. Po wykonaniu instrukcji END, CPU wykonuje samodiagnozowanie i odświeża output data, następnie powraca do odświeżenia input.

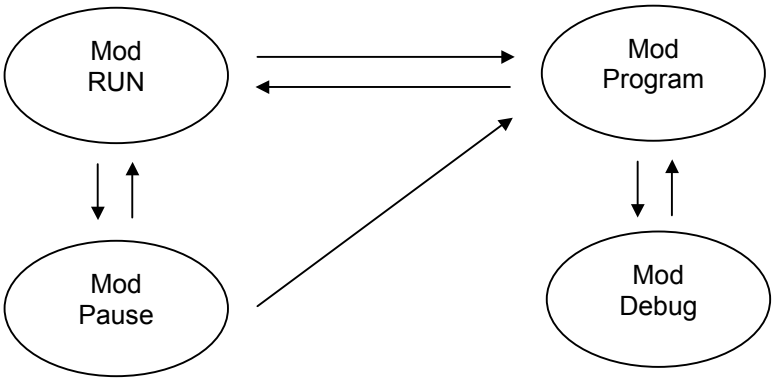
Uwagi

Scan : Seria kroków od step 0 do następnego step 0 nazywana jest scan'em. Czas scanu jednostki CPU jest liczony jako całkowity czas wykonania sekwencji programu (step 0 do END) plus czas wewnętrznego procesu (samodiagnoza i odświeżanie I/O) jednostki CPU.

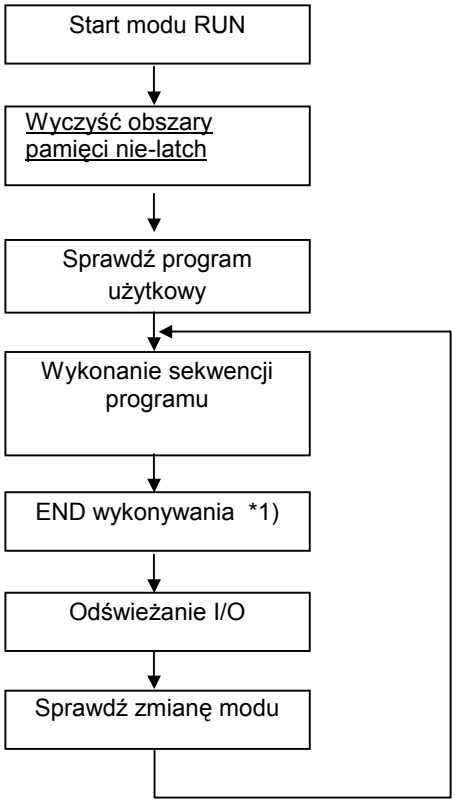
2.5.2 Mod pracy CPU

Seria MASTER-K posiada 4 mody pracy jak poniżej. Strzałki pokazują jakie zmiany modów pracy są możliwe.

<Rys. 2-4 Mody pracy serii MASTER-K >

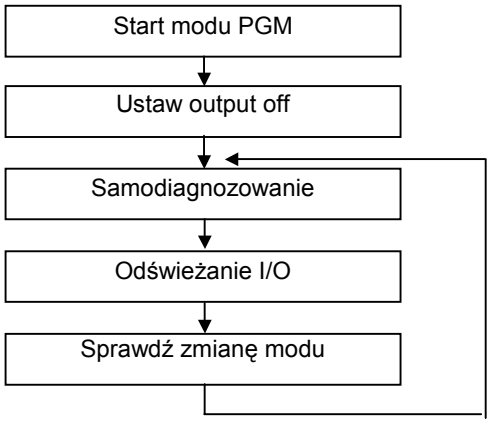


1) Diagram modu RUN



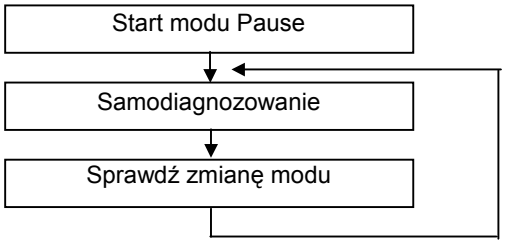
*1) END(Koniec) wykonywania : Samodiagnozowanie, uaktualnienie Timer / Counter

2) Diagram modu Program (PGM)



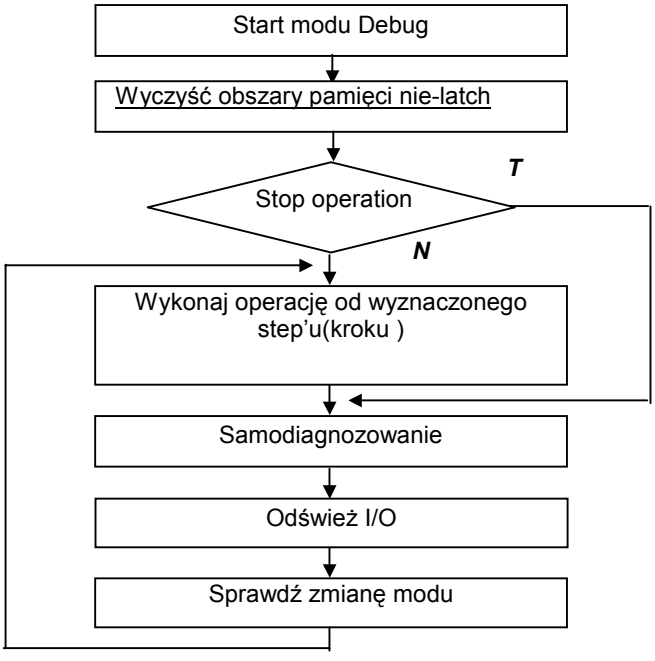
- Program czytanie / pisanie / monitorowanie może być wykonywany w modzie program. Zewnętrzne połączenia przewodów mogą być sprawdzane przez funkcję wymuszenia stanów on/off na I/O.

3) Diagram modu Pause



- Stopuje pracę CPU, lecz utrzymuje status output oraz pamięci wewnętrznej.

4)Diagram modu Debug



2.6 Funkcje specjalne serii MASTER-K

2.6.1 Funkcja RTC (Real Time Clock) (Zegar czasu rzeczywistego)

Ponieważ funkcja RTC jest opcjonalna, nie cała seria MASTER-K obsługuje tę funkcję. Patrz katalog oraz podręczik CPU dla stosowanego modelu.
Zegar funkcji RTC funkcjonuje w oparciu o baterię lub super kondensator, nawet gdy CPU jest odłączona od zasilania.

1) Dane zegara
Dane zegara zawierają rok, miesiąc, dzień, godzinę, minuty, sekundy i datę.

Nazwa danych	Opis	
Rok	Dwie najmłodsze cyfry Ery Chrześcijańskiej	
Miesiąc	1 do 12	
Dzień	1 do 31 (Rok przestępny jest odróżniany automatycznie)	
Godzina	0 do 23 (24 godziny)	
Minuta	0 do 59	
Sekunda	0 do 59	
Data	0	Niedziela
	1	Poniedziałek
	2	Wtorek
	3	Środa
	4	Czwartek
	5	Piątek
	6	Sobota

2) Dokładność
1,728 sekund na dzień (głównie temperatura)

3) K10S / K30S / K60S
a) Czytaj RTC data
RTC data są zapamiętane jak pokazuje tabela poniżej.

Obszar pamięci (Word)	Description		Przykładowe dane (BCD format)
	Wyższy byte	Niższy byte	
L012	Rok	-	h98xx
L013	Dzień	Miesiąc	h2212
L014	Godzina	Data	h1902
L015	Sekunda	Minuta	h4637

Przykład *: 1998. 12. 22. 19:37:46, Wtorek*

b) Zapisz RTC data

Są dwa sposoby zapisu nowych RTC data do CPU.
Pierwszy to użycie handy loader (KLD-150S)(Programator ręczny) lub graphic loader (KGL-WIN)(Program na PC). Szczegółowe informacje patrz podręcznik user's manual KLD-150S lub KGL-WIN.
Drugi sposób to użycie programu piszącego. Przez ustawienie specjalnych bitów użytkownik może zamienić bieżące RTC data na ustawione w wyspecyfikowanym obszarze pamięci. Adresy pamięci obszaru ustawionych danych oraz przykładowy program, podane są poniżej.

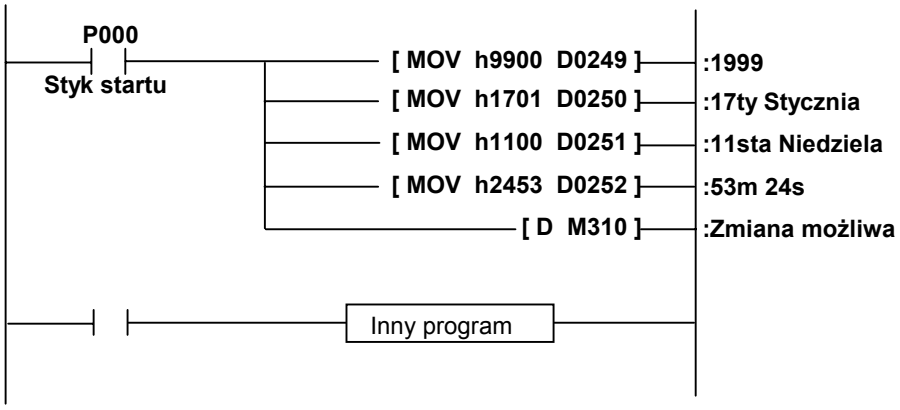
Ustawione RTC data są zapamiętywane jako następująca tabela.

Obszar pamięci (Word)	Opis		Przykładowe dane (BCD format)
	Wyższy byte	Niższy byte	
D249	Rok	-	h99xx
D250	Dzień	Miesiąc	h1701
D251	Godzina	Data	h1100
D252	Sekunda	Minuta	h2453

Przykład : 1999. 1. 17. 11:53:24, Niedziela

M310 (bit zmiany RTC data) : Gdy bit M310 zostanie ustawiony, nowe dane z D249 ~ D252 zostaną przesunięte do L12 ~ L15. Po tej operacji, M310 musi zostać natychmiast wyzerowany, ponieważ bieżące dane będą odświeżane z każdym scan'em gdy bit M310 jest ustawiony.

<Przykładowy program>



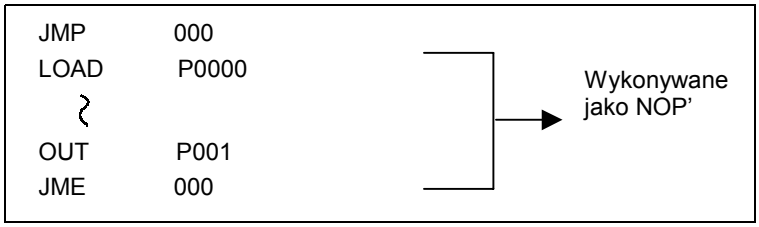
Uwagi

a) RTC data nie są ustawiane fabrycznie. Przed użyciem funkcji RTC , zapisz poprawne RTC data do modułu CPU lub RTC.
b) Jeżeli nierozsądne RTC data są wpisane do CPU, to operacja RTC nie może być wykonywana normalnie.
Przykład : 13 (miesiąc) 32 (dni)

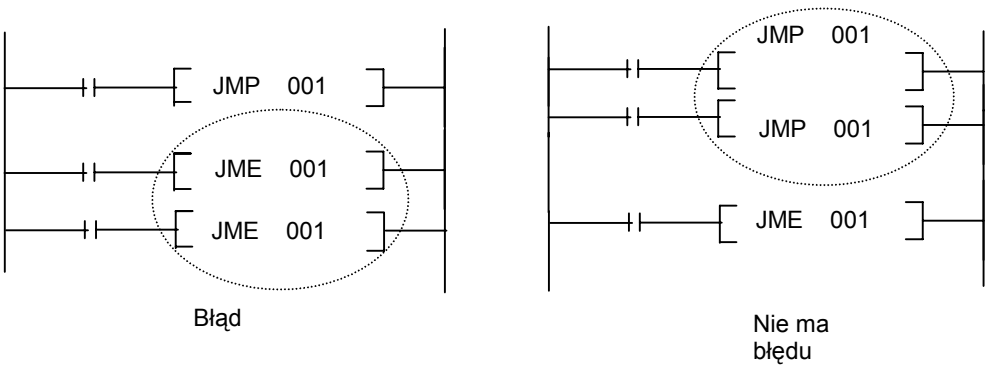
2.7 Sprawdzanie programu

2.7.1 JMP – JME

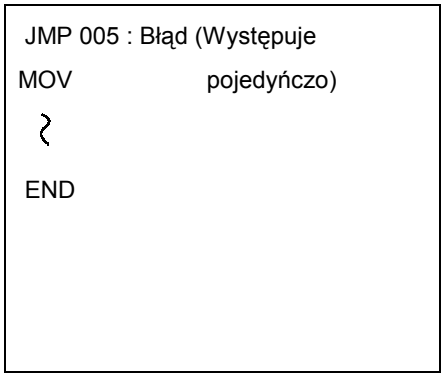
1) Jeżeli warunek input instrukcji JMP n zostanie ustawiony, CPU pomija wszystkie instrukcje aż do napotkania JME n. Pominięte instrukcje są traktowane jako instrukcje NOP. Maksymalnie może być użytych 128 JMP-JME. (JMP 0 ~ JMP 127, JME 0 ~ JME 127)

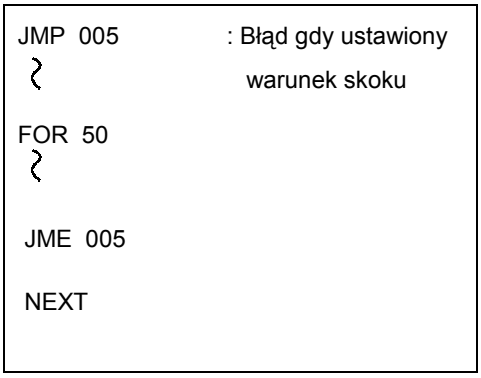


2) Instrukcja JMP n powinna współpracować tylko z jedną instrukcją JME n. W programie może występować tylko jedna instrukcja JME n. Jednakże instrukcja JMP n może występować wielokrotnie.



3) Instrukcja JMP n bez odpowiadającej jej instrukcji JME n (JMP n bez odpowiednika), spowoduje błąd programu. Jeżeli JME lub JMP znajduje się wewnątrz pęli (podprogram, FOR~NEXT blok lub procedura interrupt), to wystąpi błąd operacji w momencie ustawienia warunku skoku JMP.





2.7.2 CALL , SBRT / RET

1) CALL n, CALLP n :
   Instrukcja CALL(P) wykonuje podprogram wyspecyfikowany przez wskaźnik ‘n’. Dozwolony wielokrotny poziom zagłębiania instrukcji CALL(P).
2) SBRT / RET
   Instrukcja SBRT wskazuje start podprogramu, a RET wskazuje koniec. Obie te instrukcje powinny występować parami.

```
LOAD    P000
  ≀
SBRT    40          : Błąd (SBRT przed END)
  ≀
END
RET                 : Błąd (Występuje pojedyńczo)
```

```
LOAD    P042
  ≀
CALL    30          : Błąd (Brak SBRT)
  ≀
END
```

```
LOAD    P010
  ≀
CALL    30
  ≀
END
SBRT    30  : Błąd (Brak RET)
```

```
LOAD    P03F
  ≀
END
SBRT    20  : Błąd (Brak CALL)
  ≀
RET
```

2.7.3 MCS – MCSCLR

Instrukcja MCS n startuje sekwencję master control. Po każdej instrukcji MCS występuje liczba (n), która wskazuje jaki jest priorytet sekwencji master control. Zakres (n) jest 0 ~ 7.

```
MCS   0 : High(Wysoki)
  ≀   ↕
MCS   7 : LOW(Niski)
```

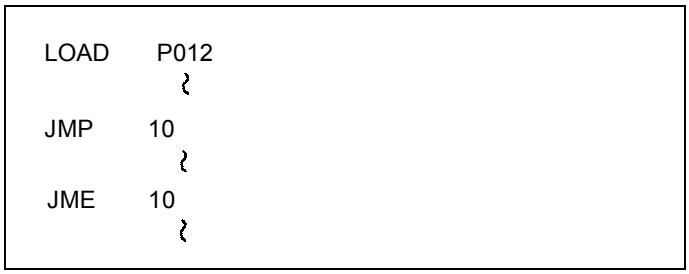
Instrukcja MCSCLR n kończy sekwencję master control. Gdy instrukcja MCSCLR jest wykonywana, to wszystkie master control które mają niższy priorytet, są automatycznie zerowane.

```
MCS       0
MCS       1
  ≀       ↕
MCSCLR    0 : (MCS 1 jest zerowana automatycznie)
MCSCLR    1 : Błąd (Niepoprawny rozkaz MCSCLR)
```

Gdy używana jest master control, to powinna startować od najwyższego priorytetu a kończyć od najniższego priorytetu. Instrukcje MCS n i MCSCLR n powinny występować parami. W przeciwnym wypadku wystąpi błąd.

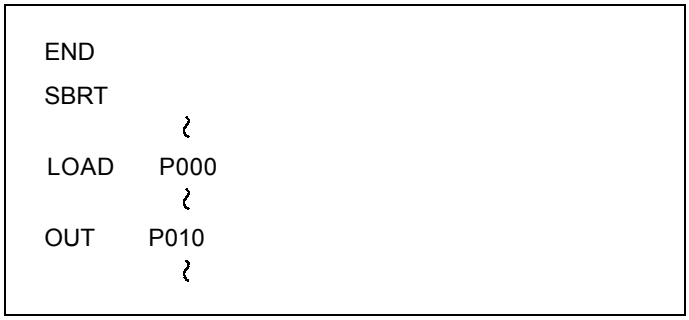
2.7.5 END / RET

1) Jeżeli w sekwencji programu brakuje END , wystąpi błąd i CPU przerwie pracę.



: Brakujący END

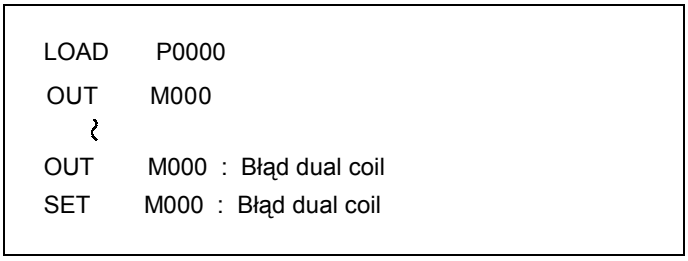
2) Jeżeli w sekwencji podprogramu brakuje RET , wystąpi błąd i CPU przerwie pracę.



: Brakujący RET

2.7.6 Dual coil(Podwójna cewka)

Jeżeli memory device(obiekt pamięci) jest używany jako output operacji dwa lub więcej razy, to wystąpi błąd
dual coil. Ponieważ nie jest to poważny błąd, nie nastąpi zatrzymanie pracy CPU.



2.8 Obsługa błędów

2.8.1 Błędy podczas RUN / STOP

Gdy wystąpi błąd operacji (błąd adresowania pośredniego, błąd operacji BCD, etc) , jednostaka CPU decyduje
kontynuować operację lub nie bazuje na nastawach parametrów.
Patrz rozdział 2.4.4 .

2.8.2 Flagi błędów (F110 / F115)

Gdy wystąpi błąd CPU w modzie RUN, to zapalą się dwie flagi błędów (F110 i F115).
F110 jest odświeżana po wykonaniu każdej instrukcji. Wykonywana instrukcja nie jest związana z jakimkolwiek błędem (jak na przykład instrukcja LOAD), jest to tylko konsekwencja poprzedniej wartości. Natomiast flaga F115 jest ustawiana natychmiast po wystąpieniu błędu. Zerowanie flagi F115 wykonuje instrukcja CLE. Tabela poniżej pokazuje przykłady działania F110 i F115.

Program	Wystąpił błąd?	F110	F115	Uwagi
ADD D0 10 M20	Nie	OFF	OFF	
MOV D0 #D10	Tak	ON	ON	D10 = hFFFF
LOAD P0000	N/A	ON	ON	
INC D0	Nie	OFF	ON	
LOAD P0001	N/A	OFF	ON	
WAND P01 M10 #D400	Tak	ON	ON	D400 = hFF00
LOAD P0002	N/A	ON	ON	
WAND P01 M10 D300	Nie	OFF	ON	
CLE	N/A	OFF	OFF	Zeruj F115
LOAD P0003	N/A	OFF	OFF	
WAND P01 M10 D500	Nie	OFF	OFF	
BCD hFFFF D20	Tak	ON	ON	

2.8.3 Wskazania LEDów

1)K10S1 / K10S / K30S / K60S

Nazwa LED	Status pracy	Wskazanie LED
ERR	· Błąd poważny	Miganie z okresem 1sek
	· Błąd Light	
	· Błąd programu lub parametru	
RUN	· CPU jest w modzie RUN	ON w sposób ciągły
	CPU jest w modzie Stop lub wystąpił błąd	OFF w sposób ciągły

2.8.4 Lista kodów błędów (Error code)

Typ Error	Komunikat	Kod (F006)	CPU	Opis	Korekcja
Błąd wewnętrzny systemu	System error	h0001	Stop	Uszkodzenie w ROM systemowym lub w H/W .	Kontakt z serwisem LG
Błąd OS ROM	OS ROM error	h0002	Stop	Uszkodzony wewnętrzny ROM.	Kontakt z serwisem LG
Błąd OS RAM	OS RAM error	h0003	Stop	Uszkodzony wewnętrzny RAM.	Kontakt z serwisem LG
Błąd Data RAM	Data RAM error	h0004	Stop	Uszkodzony RAM data.	Kontakt z serwisem LG
Błąd Program RAM	Error	h0005	Stop	Uszkodzony RAM program.	Kontakt z serwisem LG
Błąd Gate array	G/A error	h0006	Stop	Uszkodzona matryca CPU.	Kontakt z serwisem LG
Błąd Sub rack power down	Sub power error	h0007	Stop	Brak zasilania szyny rozszerzeń lub jej uszkodzenie.	Sprawdź zasilanie szyny rozszerzeń.
Błąd OS WDT time out (przekroczenie czasu)	OS WDT error	h0008	Stop	Za długi czas operacji CPU (nie chodzi tu o czas scan'u).	Włącz/wyłącz CPU.Jeżeli błąd jest nadal, skontaktuj się z serwisem LG.
Błąd wspólnej RAM	Common RAM error	h0009	Stop	Błąd wspólnej RAM	Kontakt z serwisem LG
Przerwa w bezpieczniku	I/O fuse error	h000A	Run (Stop)	Przepalony bezpiecznik	Wymień bezpiecznik
Błąd kodu instrukcji	OP code error	h000B	Stop	CPU podczas wykonywania napotkał instrukcję której nie może zdekodować.	Kontakt z serwisem LG
Błąd Flash Memory	User memory error	H000C	Stop	CPU nie ma dostępu do wewnętrznej pamięcil flash.	Sprawdź pamięć flash i w razie potrzeby wymień.
Błąd I/O slot	I/O slot error	h0010	Stop	① Montaż lub demontaż modułu gdy PLC jest zasilany.Czynność niepoprawna. ② Uszkodzony moduł I/O lub kabel rozszerzający.	① Wyłącz zasilanie – Zamontuj moduł – Włącz zasilanie ② Wymień moduł I/O lub kabel rozszerzający.
Przekroczenie liczby I/O	Max I/O over	h0011	Stop	Liczba punktów I/O przekroczyła limit. (Błąd nadmiernej ilości zamontowanych Fmm , ···)	Wymień moduł I/O.
Błąd karty specjalnej I/F	Special I/F error	h0012	Stop	Błąd wystąpił podczas dostępu do karty specjalnej.	Kontakt z serwisem LG
Błąd Fmm 0 I/F	Fmm 0 I/F error	h0013	Stop	Błąd Fmm 0 I/F	Kontakt z serwisem LG
Błąd Fmm 1 I/F	Fmm 1 I/F error	h0014	Stop	Błąd Fmm 1 I/F	Kontakt z serwisem LG
Błąd Fmm 2 I/F	Fmm 2 I/F error	h0015	Stop	Błąd Fmm 2 I/F	Kontakt z serwisem LG
Błąd Fmm 3 I/F	Fmm 3 I/F error	h0016	Stop	Błąd Fmm 3 I/F	Kontakt z serwisem LG
Błąd parameteru	Parameter error	h0020	Stop	Zły parametr lub ma on niepoprawną sumę kontrolną	Zmień parametry nastaw.

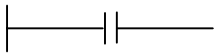
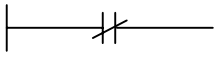
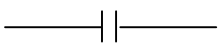
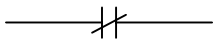
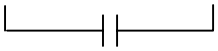
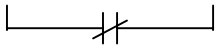
Error Code(Kody błędów) (Kontynuacja)

Typ Error	Komunikat	Kod (F006)	CPU	Opis	Korekcja
Błąd parameteru I/O	I/O parameter error	h0021	Stop	Gdy CPU zostanie zasilona lub ustawiona w mod RUN, a moduły I/O nie są zamontowane tak jak w nastawach parametrów I/O.	Zmień parametry nastaw lub przestaw moduły I/O
Błąd Maximum I/O	I/O parameter error	h0022	Stop	Wartość nastawy parametrów I/O lub ilość zamontowanych punktów I/O przekracza maksymalną liczbę punktów I/O modułu CPU	Zmień parametry nastaw
Błąd parametru Fmm 0	Fmm 0 parameter error	h0023	Run	Błąd parametru Fmm 0	Zmień parametry nastaw
Błąd parametru Fmm 1	Fmm 1 parameter error	h0024	Run	Błąd parametru Fmm 1	Zmień parametry nastaw
Błąd parametru Fmm 2	Fmm 2 parameter error	h0025	Run	Błąd parametru Fmm 2	Zmień parametry nastaw
Błąd parametru Fmm 3	Fmm 3 parameter error	h0026	Run	Błąd parametru Fmm 3	Zmień parametry nastaw
Błąd Operacji	Operation error	h0030	Stop (Run)	· Błąd operacji BCD · Błąd operandu	Sprawdź program
Błąd WDT	WDT over error	h0031	Stop	Czas scan przekroczył wartość nastawy parametru “watch dog” timera.	Zmień parametry nastaw lub wstaw instrukcję WDT.
Błąd zmiany programu	PGM Change error	h0032	Stop	Błąd wystąpił w czasie edycji programu w modzie RUN. (Zmiana nie została zakończona.)	-
Błąd zmiany programu	PGM Change error	h0033	Run	Błąd wystąpił w czasie edycji programu w modzie RUN.	-
Błąd sprawdzaniakodu	Code chack error	h0040	Stop	Natrafiono na instrukcję która nie może być zdekodowana w programie.	Sprawdź program
Błąd braku instrukcji END	Missing END instruction	h0041	Stop	Brak w programie instrukcji END.	Wstaw instrukcję END na końcu programu.
Błąd braku RET	Missing RET instruction	h0042	Stop	Brak instrukcji RET w podprogramie.	Wstaw instrukcję RET na końcu podprogramu.
Błąd braku SBRT	Missing SBRT instruction	h0043	Stop	Przywoływany jest podprogram przez instrukcję CALL, lecz takiego podprogramu nie ma.	Napisz podprogram
Błąd instrukcji JMP~JME	JMP/JME error	h0044	Stop	Instrukcje JMP~JME zostały użyte niepoprawnie.	Sprawdź program
Błąd instrukcji FOR~NEXT	FOR~NEXT error	h0045	Stop	Instrukcje FOR~NEXT zostały użyte niepoprawnie.	Sprawdź program
Błąd MCS ~ MCSCLR	MCS ~ MCSCLR error	h0046	Stop	Instrukcje MCS ~MCSCLR zostały użyte niepoprawnie.	Sprawdź program
Błąd MPUSH ~ MPOP	MPUSH ~ MPOP error	h0047	Stop	Instrukcje MPUSH ~ MPOP zostały użyte niepoprawnie.	Sprawdź program
Błąd Dual Coil	Dual Coil error	h0048	Stop	Device został użyty jako output operacji więcej niż jeden raz.	Sprawdź program
Błąd Syntax	Syntax error	h0049	Stop	Zły warunek input lub za dużo instrukcji LOAD, etc.	Sprawdź program
Błąd Baterii	Battery error	h0050	Run	Za niskie napięcie baterii podtrzymującej.	Zmień na nową,

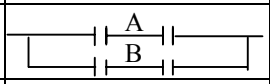
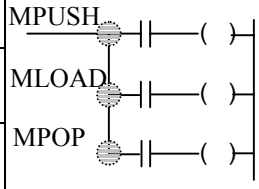
Rozdział 3 Instrukcje

3.1 Instrukcje podstawowe

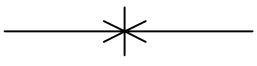
3.1.1 Instrukcje Contact

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
LOAD	-		-	NO styk operacja start	○	4- 1
LOAD NOT	-		-	NC styk operacja start	○	4- 1
AND	-		-	NO styk połączenie szeregowe	○	4- 3
AND NOT	-		-	NC styk połączenie szeregowe	○	4- 3
OR	-		-	NO styk połączenie równoległe	○	4- 4
OR NOT	-		-	NC styk połączenie równoległe	○	4- 4

3.1.2 Instrukcje Connection

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
AND LOAD	-		-	Połączenie szeregowe bloków	○	4- 6
OR LOAD	-		-	Połączenie równoległe bloków	○	4- 8
MPUSH	005		-	Zapamiętuje wynik operacji	○	4- 10
MLOAD	006		-	Czyta wynik operacji z MPUSH	○	4- 10
MPOP	007		-	Czyta wynik operacji z MPUSH i zeruje wynik	○	4- 10

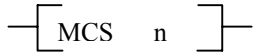
3.1.3 Instrukcja negacji

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
NOT	-		-	Neguje wynik operacji	○	4- 12

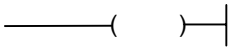
Uwagi

Aplikacja w typie CPU : ○ = Wszystkie CPUs ; □ = K10S1 / K10S / K30S / K60S ; * = K200S / K300S / K1000S

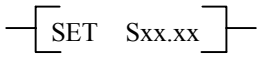
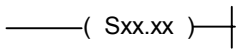
3.1.4 Instrukcje Master control

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
MCS	010		-	Startuje master control	○	4- 13
MCSCLR	011		-	Kończy master control	○	4 – 13

3.1.5 Instrukcje Output

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
D	017		-	Narastające zbocze sygnału wejściowego ustawia device na czas trwania jednego scan'u.	○	4- 16
D NOT	018		-	Opadające zbocze sygnału wejściowego ustawia device na czas trwania jednego scan'u.	○	4 – 18
SET	-		-	Ustawia obiekt	○	4 – 19
RST	-		-	Resetuje obiekt	○	4 – 20
OUT	-		-	Wystawia obiekt na wyjściu	○	

3.1.6 Instrukcje Step controller (Sterowania krokami)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
SET S	-		-	Sterowanie sekwencyjne	○	4- 22
OUT S	-		-	Ostatnio użyty ma priorytet	○	4 – 24

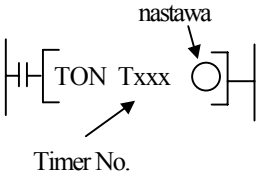
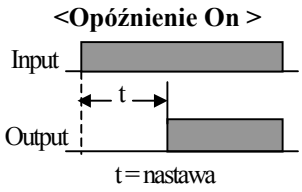
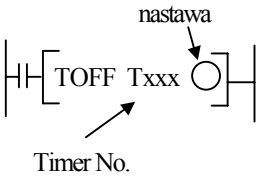
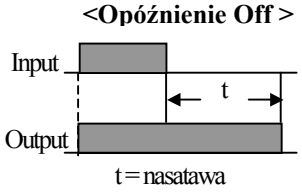
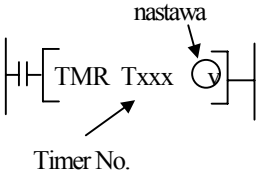
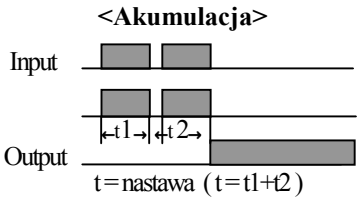
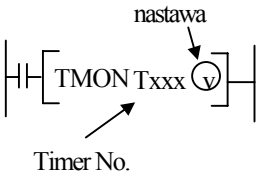
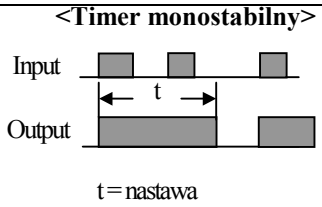
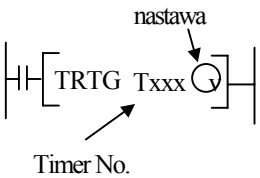
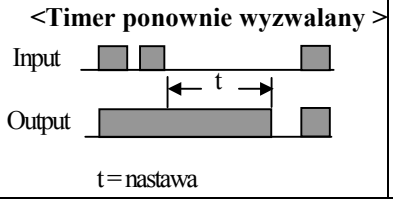
3.1.7 Instrukcja END

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
END	001		-	Kończy sekwencję programu	○	4- 25

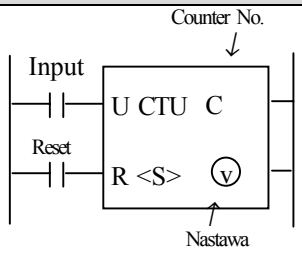
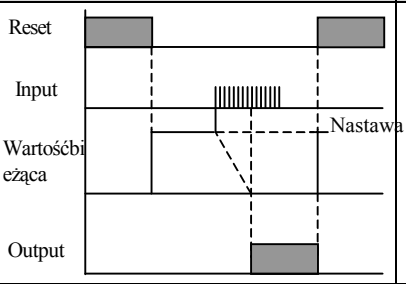
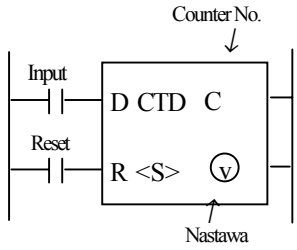
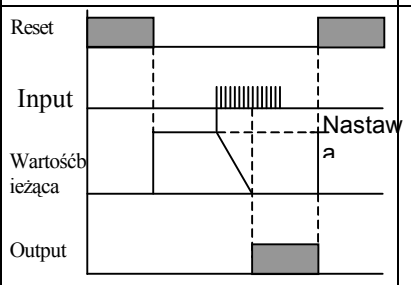
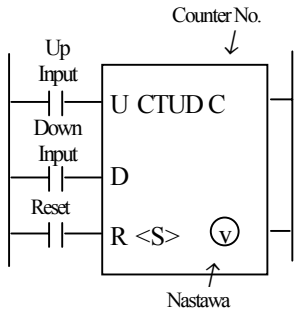
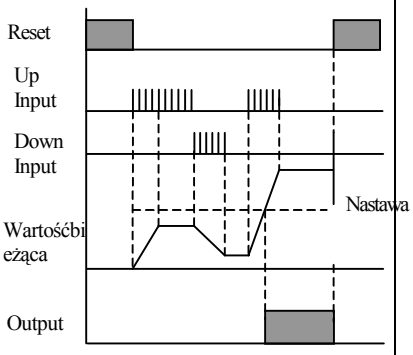
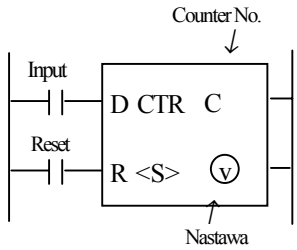
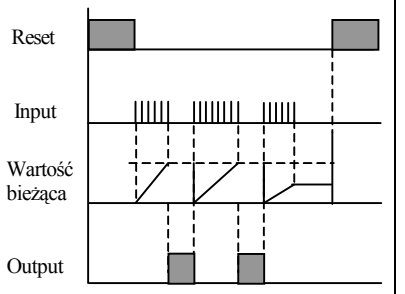
3.1.8 Instrukcja „nic nie rób" No operation

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
NOP	000	Brak symbolu	-	Nie rób nic (zajmuje 1 krok)	○	4- 26

3.1.9 Instrukcje liczników czasu - Timer

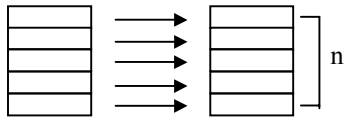
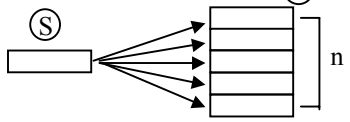
Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
TON	-		-		○	4- 27
TOFF	-		-		○	4 – 29
TMR	-		-		○	4 – 31
TMON	-		-		○	4 – 33
TRTG	-		-		○	4 – 35

3.1.10 Instrukcje liczników- Counter

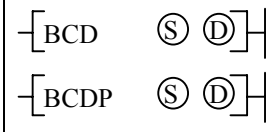
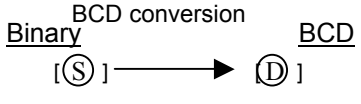
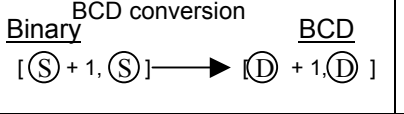
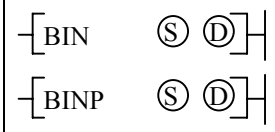
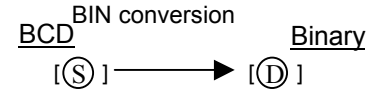
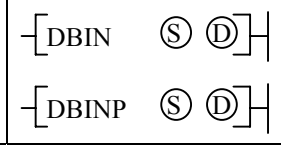
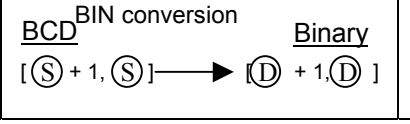
Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
CTU	-		-		○	4- 37
CTD	-		-		○	4 – 38
CTUD	-				○	4 – 39
CTR	-				○	4 – 41

3.2 Instrukcje aplikacyjne

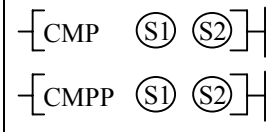
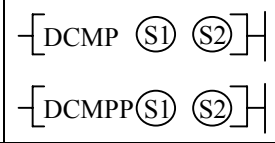
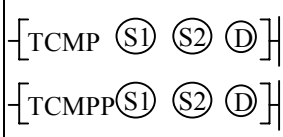
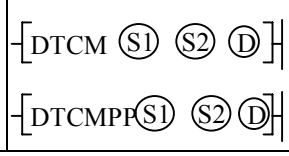
3.2.1 Instrukcje Data transfer

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
MOV MOVP	080 081	MOV Ⓢ Ⓓ MOVP Ⓢ Ⓓ	16 bits	Move data (Przesuń dane) [Ⓢ] ⟶ [Ⓓ]	○	5-1
DMOV DMOVP	082 083	DMOV Ⓢ Ⓓ DMOV Ⓢ Ⓓ	32 bits	Move data (Przesuń dane) [Ⓢ + 1, Ⓢ] ⟶ [Ⓓ + 1, Ⓓ]	○	5-1
CMOV CMOVP	084 085	CMOV Ⓢ Ⓓ CMOV Ⓢ Ⓓ	16 bits	Rewersyjne przesunięcie danych $[\overline{Ⓢ}]$ ⟶ [Ⓓ]	○	5-3
DCMOV DCMOVP	086 087	DCMO Ⓢ Ⓓ DCMOVP Ⓢ Ⓓ	32 bits	Rewersyjne przesunięcie danych $[\overline{Ⓢ + 1}, \overline{Ⓢ}]$ ⟶ [Ⓓ + 1, Ⓓ]	○	5-3
GMOV GMOVP	090 091	GMOV Ⓢ Ⓓ n GMOVP Ⓢ Ⓓ n	16 bits	Ⓢ Przesunięcie grupowe Ⓓ 	○	5-5
FMOV FMOVP	092 093	FMOV Ⓢ Ⓓ n FMOVP Ⓢ Ⓓ n	16 bits	Przesunięcie wypełniające Ⓓ 	○	5-7
BMOV BMOVP	100 101	BMOV Ⓢ Ⓓ Cw BMOVP Ⓢ Ⓓ Cw	n bit	Przesuń Bit	○	5-9

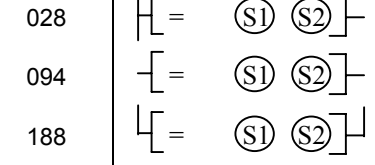
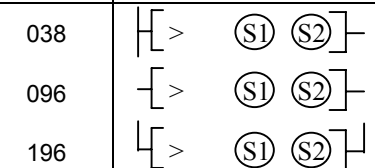
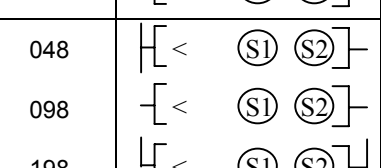
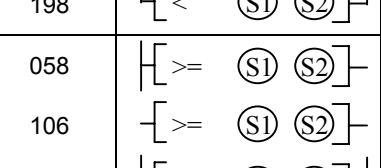
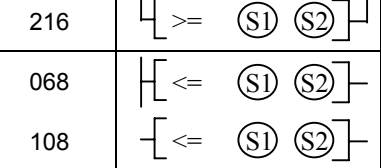
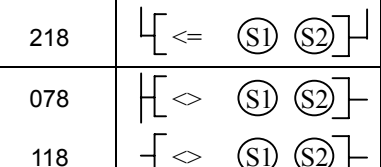
3.2.2 Instrukcje Conversion (Konwersji)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
BCD BCDP	060 061		16 bits		○	5-11
DBCD DBCDP	062 063		32 bits		○	5-11
BIN BINP	064 065		16 bits		○	5-14
DIND DBINP	066 067		32 bits		○	5-14

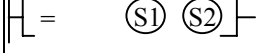
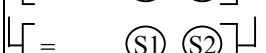
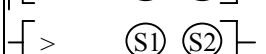
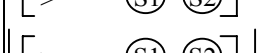
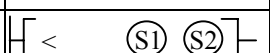
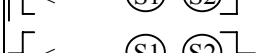
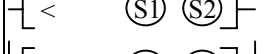
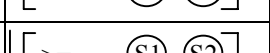
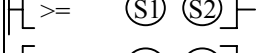
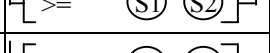
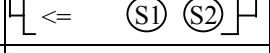
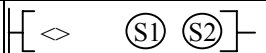
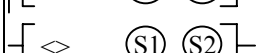
3.2.3 Instrukcje Compare (Porównania)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
CMP CMPP	050 051		16 bits	Porównaj S1 i S2. Wynik w F120 ~ F125	○	5-16
DCMP DCMPP	052 053		32 bits	Porównaj [S1+1, S1] i [S2+1, S2] Wynik w F120 ~ F125	○	5-16
TCMP TCMPP	054 055		16 bits	Porównaj S1 i 16 słów z S2 Wynik (16bitów) zapamiętany w D	○	5-19
DTCMP DTCMPP	056 057		32 bits	Porównaj [S1+1, S1] i 32 slowa z S2 Wynik (32bity) zapamiętany w [D+1, D]	○	5-19

Instrukcje Porównania (Kontynuacja)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
LOAD=	028		16 bits	Warunek Input jest ON gdy [S1] = [S2]	*	5-21
AND=	094					5-22
OR=	188					5-23
LOAD>	038		16 bits	Warunek Input jest ON gdy [S1] > [S2] (Uwzględniając znak)	*	5-21
AND>	096					5-22
OR>	196					5-23
LOAD<	048		16 bits	Warunek Input jest ON gdy [S1] < [S2] (Uwzględniając znak)	*	5-21
AND<	098					5-22
OR<	198					5-23
LOAD>=	058		16 bits	Warunek Input jest ON gdy [S1] >= [S2] (Uwzględniając znak)	*	5-21
AND>=	106					5-22
OR>=	216					5-23
LOAD<=	068		16 bits	Warunek Input jest ON gdy [S1] <= [S2] (Uwzględniając znak)	*	5-21
AND<=	108					5-22
OR<=	218					5-23
LOAD<>	078		16 bits	Warunek Input jest ON gdy [S1] <= [S2] (Uwzględniając znak)	*	5-21
AND<>	118					5-22
OR<>	228					5-23

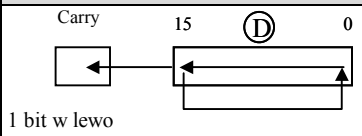
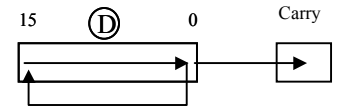
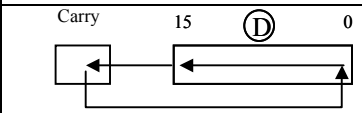
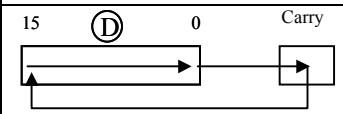
Instrukcje Porównania (Kontynuacja)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
LOADD=	029		32 bits	Warunek Input jest ON gdy [S1+1, S1] = [S2+1, S2]	*	5-21
ANDD=	095					5-22
ORD=	189					5-23
LOADD>	039		32 bits	Warunek Input jest ON gdy [S1+1, S1] > [S2+1, S2] (Uwzględniając znak)	*	5-21
ANDD>	097					5-22
ORD>	197					5-23
LOADD<	049		32 bits	Warunek Input jest ON gdy [S1+1, S1] < [S2+1, S2] (Uwzględniając znak)	*	5-21
ANDD<	099					5-22
ORD<	199					5-23
LOADD>=	059		32 bits	Warunek Input jest ON gdy [S1+1, S1] >= [S2+1, S2] (Uwzględniając znak)	*	5-21
ANDD>=	107					5-22
ORD>=	217					5-23
LOADD<=	069		32 bits	Warunek Input jest ON gdy [S1+1, S1] <= [S2+1, S2] (Uwzględniając znak)	*	5-21
ANDD<=	109					5-22
ORD<=	219					5-23
LOADD<>	079		32 bits	Warunek Input jest ON gdy [S1+1, S1] <> [S2+1, S2]	*	5-21
ANDD<>	119					5-22
ORD<>	229					5-23

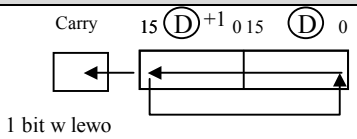
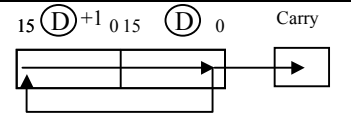
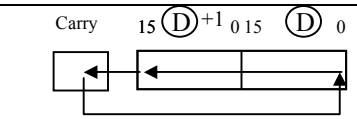
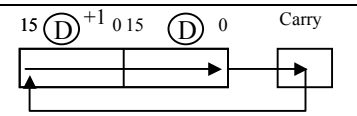
3.2.4 Instrukcje Increment / Decrement (Zwiekszania/ Zmniejszania)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
INC INCP	020 021	INC Ⓓ INCP Ⓓ	16 bits	Increment (Zwiekszanie) [Ⓓ] +1 ⟶ [Ⓓ]	○	5-24
DINC DINCP	022 023	DINC Ⓓ DINCP Ⓓ	32 bits	Increment (Zwiekszanie) [Ⓓ+1,Ⓓ] +1 ⟶ [Ⓓ+1Ⓓ]	○	5-24
DEC DECP	024 025	DEC Ⓓ DECP Ⓓ	16 bits	Decrement (Zmniejszanie) [Ⓓ] -1 ⟶ [Ⓓ]	○	5-26
DDEC DDECP	026 027	DDEC Ⓓ DDECP Ⓓ	32 bits	Decrement (Zmniejszanie) [Ⓓ+1,Ⓓ] -1 ⟶ [Ⓓ+1Ⓓ]	○	5-26

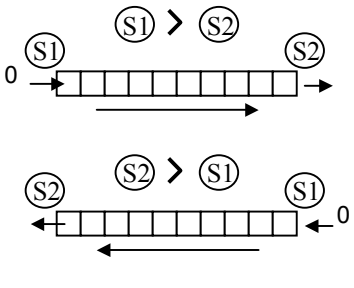
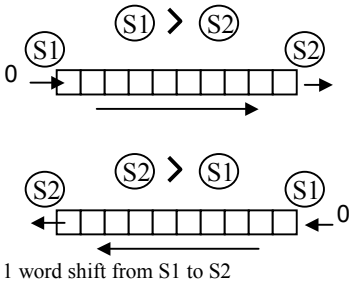
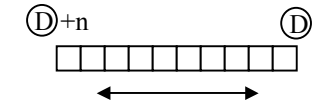
3.2.5 Instrukcje Rotation (Rotacji)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
ROL ROLP	020 021	ROL Ⓓ ROLP Ⓓ	16 bits	 1 bit w lewo	○	5-28
ROR RORP	034 035	ROR Ⓓ RORP Ⓓ	16 bits	 1 bit w prawo	○	5-30
RCL RCLP	040 041	RCL Ⓓ RCLP Ⓓ	16 bits	 1 bit w lewo uwzględniając carry	○	5-32
RCR RCRP	044 045	RCR Ⓓ RCRP Ⓓ	16 bits	 1 bit w prawo uwzględniając carry	○	5-34

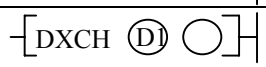
Instrukcje Rotation (Kontynuacja)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
DROL DROLP	022 023	DROL D DROLP D	32 bits	 1 bit w lewo	○	5-28
DROR DRORP	036 037	DROR D DRORP D	32 bits	 1 bit w prawo	○	5-30
DRCL DRCLP	042 043	DRCL D DRCLP D	32 bits	 1 bit w lewo uwzględniając carry	○	5-32
DRCR DRCRP	046 047	DRCR D DRCRP D	32 bits	 1 bit w prawo uwzględniając carry	○	5-34

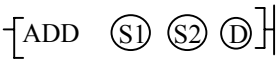
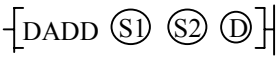
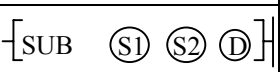
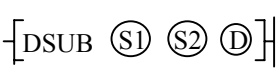
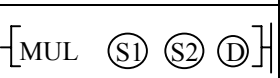
3.2.6 Instrukcje Shift

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
BSFT BSFTP	074 075	BSFT S1 S2 BSFTP S1 S2	S1-S2 bits	 1 bit shift from S1 to S2	○	5-36
WSFT WSFTP	070 071	WSFT S1 S2 WSFTP S1 S2	S1-S2 words	 1 word shift from S1 to S2	○	5-38
SR	237	SR D n	16 bits	 Bit shift (See 4. For details)	*	5-40

3.2.7 Instrukcje Exchange (Wymiany)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
XCH	102		16 bits	[D1] ⟷ [D2]	○	5-42
XCHP	103					
DXCH	104		32 bits	[D1+1, D1] ⟷ [D2+1, D2]	○	5-42
DXCHP	105					

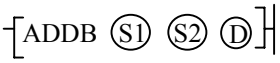
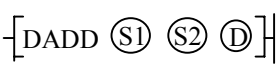
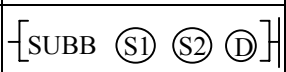
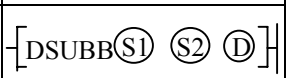
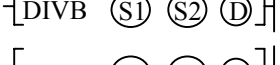
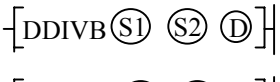
3.2.8 Instrukcje BIN arithmetic

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
ADD	110		16 bits	[S1] + [S2] ⟶ [D]	○	5-44
ADDP	111					
DADD	112		32 bits	[S1+1, S1] + [S2+1, S2] ⟶ [D+1, D]	○	5-44
DADDP	113					
SUB	114		16 bits	[S1] - [S2] ⟶ [D]	○	5-46
SUBP	115					
DSUB	116		32 bits	[S1+1, S1] - [S2+1, S2] ⟶ [D+1, D]	○	5-46
DSUBP	117					
MUL	120		16 bits	[S1] × [S2] ⟶ [D+1, D] [D+1] : High word, [D] : Low word	○	5-48
MULP	121					
DMUL	122		32 bits	[S1+1, S1]×[S2+1, S2] ⟶ [D+3,D+2,D+1,D] [D+3,D+2] = Higher 2 words [D+1, D] = Lower 2 words	○	5-48
DMULP	123					

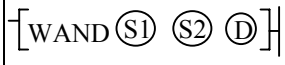
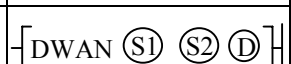
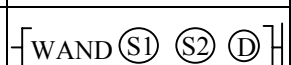
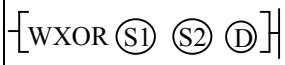
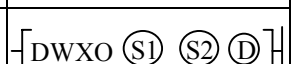
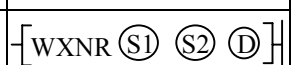
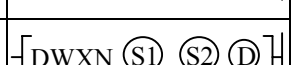
Instrukcje BIN arithmetic (Kontynuacja)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
DIV DIVP	124 125	DIV S1 S2 D DIVP S1 S2 D	16 bits	[S1]÷[S2] ⟶ [D] [D+1] = Remainder (Reszta) [D] = Quotient (Wynik)	○	5-50
DDIV DDIVP	126 127	DDIV S1 S2 D DDIVP S1 S2 D	32 bits	[S1+1, S1]÷[S2+1, S2] ⟶ [D+3, D+2, D+1, D] [D+3, D+2] = Remainder (Reszta) [D+1, D] = Quotient (Wynik)	○	5-50
MULS MULSP	072 073	MULS S1 S2 D MULSP S1 S2 D	16 bits	Mnozenie (Uwzgledniajac znak) [S1] × [S2] ⟶ [D+1, D] [D+1] : High word, [D] : Low word	○	
DMULS DMULSP	076 077	DMUL S1 S2 D DMULSP S1 S2 D	32 bits	Mnozenie (Uwzgledniajac znak) [S1+1, S1]×[S2+1, S2] ⟶ [D+3,D+2,D+1,D] [D+3,D+2] = Higher 2 words [D+1, D] = Lower 2 words	○	
DIVS DIVSP	088 089	DIVS S1 S2 D DIVSP S1 S2 D	16 bits	Dzielenie (Uwzgledniajac znak) [S1]÷[S2] ⟶ [D] [D+1] = Remainder (Reszta) [D] = Quotient (Wynik)	○	
DDIVS DDIVSP	128 129	DDIVS S1 S2 D DDIVSP S1 S2 D	32 bits	Dzielenie (Uwzgledniajac znak) [S1+1, S1]÷[S2+1, S2] ⟶ [D+3, D+2, D+1, D] [D+3, D+2] = Remainder (Reszta) [D+1, D] = Quotient (Wynik)	○	

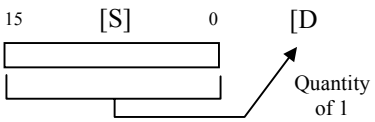
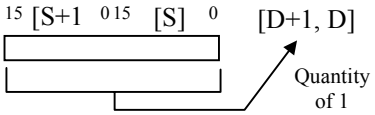
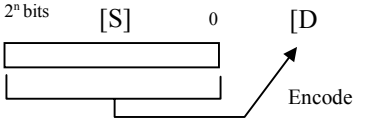
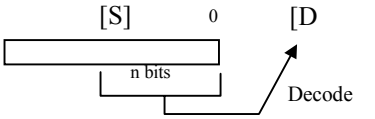
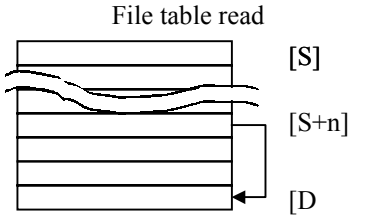
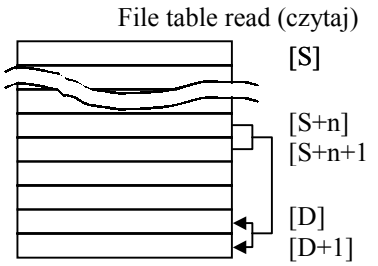
3.2.9 Instrukcje BCD arithmetic

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
ADDB ADDBP	130 131		16 bits	BCD addition (Dodawanie) [S1] + [S2] ⟶ [D]	○	5-52
DADDB DADDBP	132 133		32 bits	BCD addition (Dodawanie) [S1+1,S1]+[S2+1,S2] ⟶ [D+1,D]	○	5-52
SUBB SUBBP	134 135		16 bits	BCD subtraction (Odejmowanie) [S1] - [S2] ⟶ [D]	○	5-54
DSUBB DSUBBP	136 137		32 bits	BCD subtraction (Odejmowanie) [S1+1,S1]-[S2+1,S2] ⟶ [D+1,D]	○	5-54
MULB MULBP	140 141		16 bits	BCD multiplication (Mniozenie) [S1] × [S2] ⟶ [D+1, D] [D+1] : High word, [D] : Low word	○	5-56
DMULB DMULBP	142 143		32 bits	BCD multiplication (Mniozenie) [S1+1, S1]×[S2+1, S2] ⟶ [D+3,D+2,D+1,D] [D+3,D+2] = Higher 2 words [D+1, D] = Lower 2 words	○	5-56
DIVB DIVBP	144 145		16 bits	BCD division(Dzielenie) [S1]÷[S2] ⟶ [D] [D+1] = Remainder (Reszta) [D] = Quotient (Wynik)	○	5-58
DDIVB DDIVBP	146 147		32 bits	BCD division(Dzielenie) [S1+1, S1]÷[S2+1, S2] ⟶ [D+3, D+2, D+1, D] [D+3, D+2] = Remainder (Reszta) [D+1, D] = Quotient (Wynik)	○	5-58

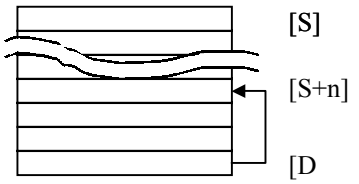
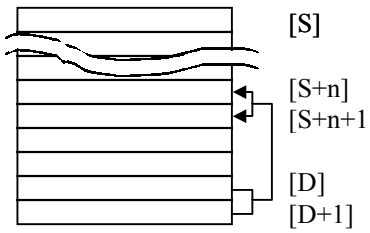
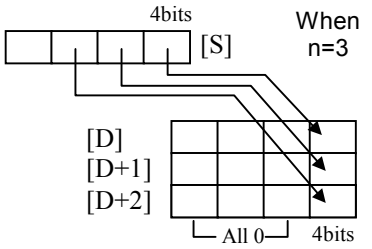
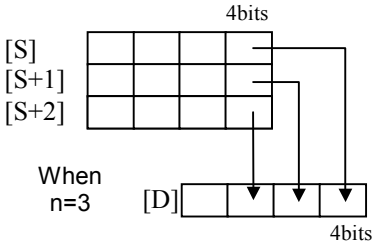
3.2.10 Instrukcje Logical opration

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
WAND WANDP	130 131		16 bits	[S1] AND [S2] ⟶ [D]	○	5-60
DWAND DWANDP	132 133		32 bits	[S1+1,S1] AND [S2+1,S2] ⟶ [D+1,D]	○	5-60
WOR WORP	154 155		16 bits	[S1] OR [S2] ⟶ [D]	○	5-62
DWOR DWORP	156 157		32 bits	[S1+1,S1] OR [S2+1,S2] ⟶ [D+1,D]	○	5-62
WXOR WXORP	160 161		16 bits	[S1] XOR [S2] ⟶ [D]	○	5-64
DWXOR DWXORP	162 163		32 bits	[S1+1,S1] XOR [S2+1,S2] ⟶ [D+1,D]	○	5-64
WXNR WXNRP	164 165			[S1] XNR [S2] ⟶ [D]	○	5-66
DWXNR DWXNRP	166 167			[S1+1,S1] XNR [S2+1,S2] ⟶ [D+1,D]	○	5-66

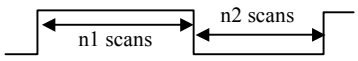
3.2.11 Instrukcje Data processing

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
SEG SEGP	174 175	SEG Ⓢ Ⓓ Cw SEGP Ⓢ Ⓓ Cw	16 bits	7 Segment decode [S] → dekoduje → [D]	○	5-68
ASC ASCP	190 191	ASC Ⓢ Ⓓ Cw ASCP Ⓢ Ⓓ Cw		Zamienia data [S] na ASCII code format i zapamietuje w [D].	○	5-71
BSUM BSUMP	170 171	DBIN Ⓢ Ⓓ DBINP Ⓢ Ⓓ	16 bits		○	5-73
DBSUM DBSUMP	172 173	DBIN Ⓢ Ⓓ DBINP Ⓢ Ⓓ	32 bits		○	5-73
ENCO ENCOP	176 177	ENCO Ⓢ Ⓓ n ENCOP Ⓢ Ⓓ n	2^n bits		○	5-75
DECO DECOP	178 179	DECO Ⓢ Ⓓ n DECOP Ⓢ Ⓓ n	n bits		○	5-77
FILR FILRP	180 181	FILR Ⓢ Ⓓ n FILRP Ⓢ Ⓓ n	16 bits		○	5-79
DFILR DFILRP	182 183	DFILR Ⓢ Ⓓ n DFILRP Ⓢ Ⓓ n	32 bits		○	5-79

Instrukcje Data processing (Kontynuacja)

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
FILW FILWP	184 185	FILW Ⓢ Ⓓ n FILWP Ⓢ Ⓓ n	16 bits	Zapisz File table  	○	5-81
DFILW DFILWP	186 187	DFILW Ⓢ Ⓓ n DFILWP Ⓢ Ⓓ n	32 bits	Zapisz File table  	○	5-81
DIS DISP	194 195	DIS Ⓢ Ⓓ n DISP Ⓢ Ⓓ n	16 bits	 	○	5-83
UNI UNIP	192 193	UNI Ⓢ Ⓓ n UNIP Ⓢ Ⓓ n	32 bits	 	○	5-85
IORF IORFP	200 201	IORF Ⓓ1 ◯ IORFP Ⓓ1 Ⓓ2	16 bits	Odświeża obszar pamięci od [D1] do [D2] ([D1] < [D2])	*	5-87

3.1.12 Instrukcje Systemowe

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
FALS	204	FALS n		Zapamietuje n do wyspecyfikowanego obszaru F	*	5-89
DUTY	205	DUTY Ⓓ n1 n2		Generuje impulsy jak poniżej 	○	5-90
WDT WDTP	202 203	WDT WDTP		Zeruje watch dog timer	*	5-92
OUTOFF	208	OUTOFF		Ustawia wszystkie outputs na Off	○	5-94
STOP	008	STOP		Stopuje pracę CPU	*	5-95

3.2.13 Instrukcje skoków - Branch

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
JMP JME	012 013	JMP n JME n		Jump (Skok) Jump end (Koniec skoku)	○	5-96
CALL CALLP	014 015	CALL n CALLP n		Przywołaj subroutine	○	5-98
SBRT RET	016 004	SBRT n RET		Startuj subroutine (podprogram) End(Kończ) subroutine	○	

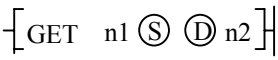
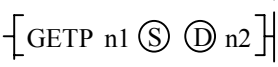
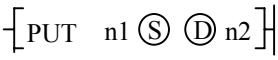
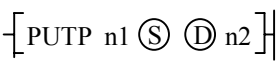
3.2.14 Instrukcje pętli Loop

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
FOR NEXT	206 207	FOR n NEXT		Wykonuje n razy sekwencję programu zawartą pomiędzy FOR i NEXT	*	5-100
BREAK	220	BREAK		Escape from FOR/NEXT loop (Wyjdź z pętli FOR/NEXT)	*	5-101

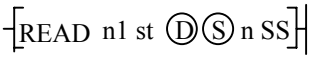
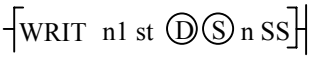
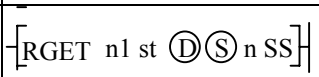
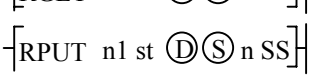
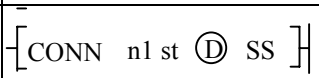
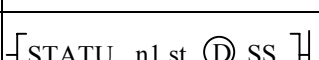
3.2.15 Instrukcje Flag

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
STC	002			Set the carry flag (Ustaw carry)	○	5-102
CLC	003			Clear the carry flag (Zeruj carry)		
CLE	009			Clear the error flag (Zeruj carry)	*	5-103

3.2.16 Instrukcje Special module

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
GET	230			Czytaj data ze wspólnej pamięci RAM modułu specjalnego	*	5-104
GETP	231					
PUT	234			Pisz data do wspólnej pamięci RAM modułu specjalnego	*	5-106
PUTP	235					

3.2.17 Instrukcje Data link

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
READ	244			Czytaj/Pisz data zdalnej station	*	5-108
WRITE	245					5-111
RGET	232			Czytaj/Pisz data wspólnej pamięci RAM modułu specjalnego	*	5-113
RPUT	233					5-116
CONN	246			Ustanów kanał komunikacyjny	*	
STATUS	247			Czytaj data zdalnej station	*	5-118

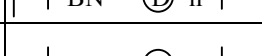
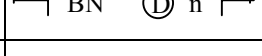
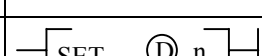
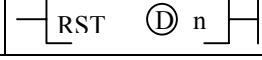
3.2.18 Instrukcje Interrupt

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
EI	236	EI n		Umożliwij interrupt	*	5-119
DI	239	DI n		Uniemożliwij interrupt		
EI	221	EI		Umożliwij wszystkie interrupts	*	5-119
DI	222	DI		Uniemożliwij wszystkie interrupts		
TDINT	226	TDINT n		Start TDI routine	*	5-120
INT	227	INT n		Start PDI routine		5-121
IRET	225	IRET		End interrupt(przerwanie) routine		

3.2.19 Instrukcje Sign inversion

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
NEG	240	NEG Ⓓ		Zmień znak [D]	*	5-122
NEGP	241	NEGP Ⓓ				
DNEG	242	DNEG Ⓓ		Zmień znak [D+1, D]	*	5-122
DNEGP	243	DNEGP Ⓓ				

3.2.20 Instrukcje Bit contact

Symbol instrukcji	Funkcja No	Symbol drabinkowy	Jedn	Wykonywane operacje	CPU	Str.
BLD	248		-	Styk NO startuje bitem n słowa [D]	*	5-124
BLDN	249			Styk NC startuje bitem n słowa [D]	*	5-124
BAND	250			Styk NO łączy szeregowo bitem n słowa [D]	*	5-125
BANDN	251			Styk NC łączy szeregowo bitem n słowa [D]	*	5-125
BOR	252			Styk NO łączy równolegle bitem n słowa [D]	*	5-126
BORN	253			Styk NC łączy równolegle bitem n słowa [D]	*	5-126
BOUT	236			Wystawia wynik operacji do bitu n słowa [D]	*	5-127
BSET	223			Ustaw bit n słowa [D]	*	5-128
BRST	224			Zeruj bit n słowa [D]	*	5-128

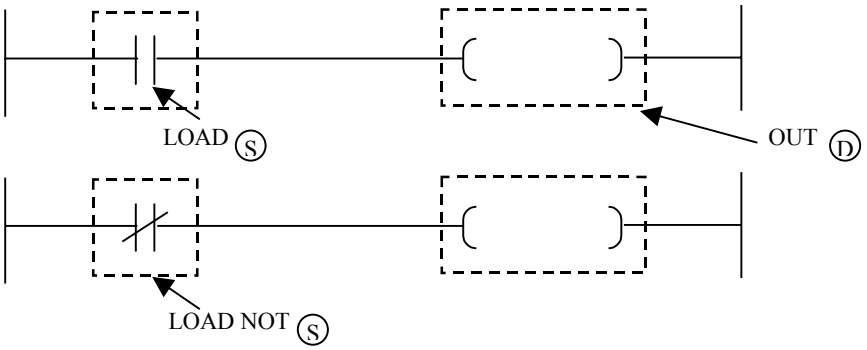
4. Instrukcje podstawowe

4.1 Instrukcje stykowe

4.1.1 LOAD, LOAD NOT, OUT

LOAD LOAD NOT OUT	

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Int		Error (F110)	Zero (F111)	Carry (F112)
LOAD LOAD NOT	Ⓢ	O	O	O	O	O	O	O	O				1			
OUT	Ⓓ	O	O	O	O*				O							



* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1)LOAD Ⓢ
  a)Funkcje
    - Start styku NO.
    - Pobiera on/off data ze wskazanego obiektu (Ⓢ) i używa dane jako wynik operacji.

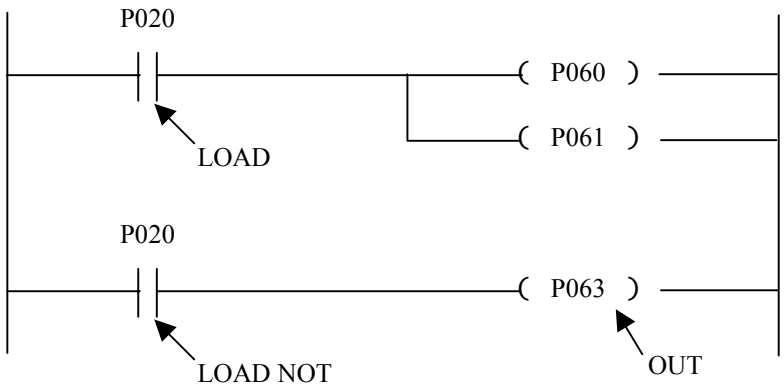
2)LOAD NOT Ⓢ
  a)Funkcje
    - Start styku NC.
    - Pobiera on/off data ze wskazanego obiektu (Ⓢ) i używa dane jako wynik operacji.

3)OUT Ⓓ
  a)Funkcje
    - Wydaje wynik operacji do wskazanego obiektu (Ⓓ).
    - Można użyć kilka instrukcji OUT równolegle z tym samym wynikiem operacji.

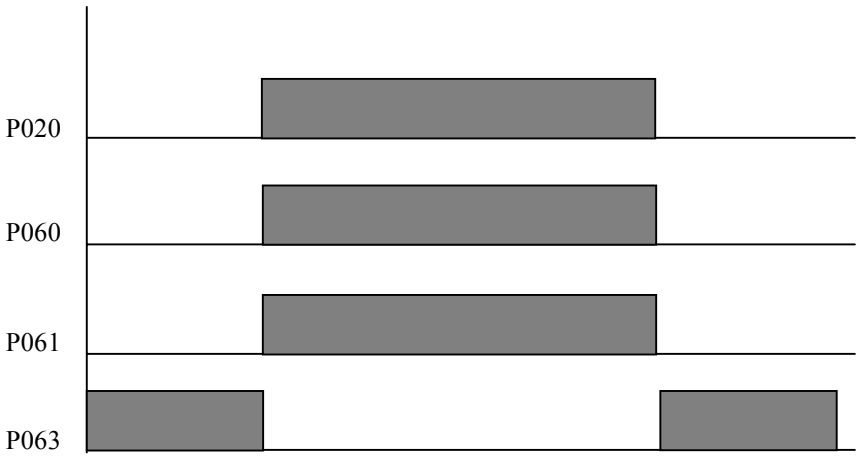
4) Przykład programu

- Gdy warunek input (P020) jest ON, to P060 i P061 zostanie ustawiony ON a P062
- zostanie wyłączony OFF.

[Program]



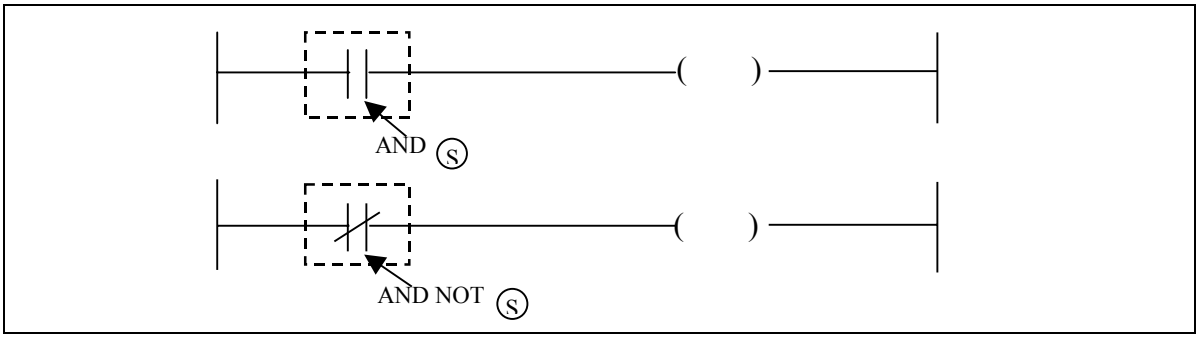
[Diagram czasowy]



4.1.2 AND, AND NOT

AND AND NOT	

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Int		Error (F110)	Zero (F111)	Carry (F112)
AND AND NOT	Ⓢ	O	O	O	O	O	O	O	O				1			



1) AND

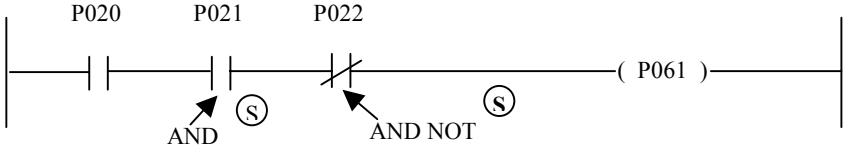
a)Funkcje
- Połączenie szeregowe styku NO
- Czyta on/off data ze wskazanego (Ⓢ)ektu ( ), wykonuje operację AND tych danych i wyniku poprzedniej operacji, czego następnie używa jako wynik nowej operacji.

2) AND NOT

a)Funkcje
- Połączenie szeregowe styku NC
- Czyta on/off data ze wskazanego (Ⓢ)ektu ( ), wykonuje operację AND tych danych i wyniku poprzedniej operacji, czego następnie używa jako wynik nowej operacji.

3) Przykład programu

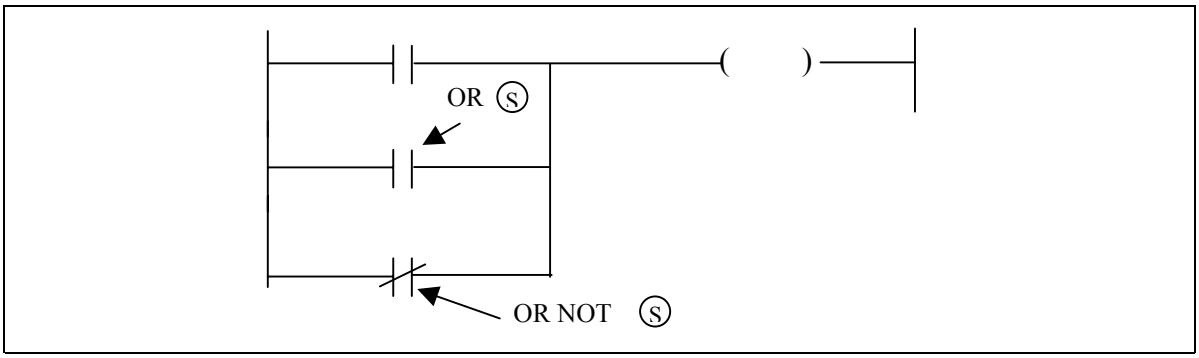
Styk P061 zostanie ustawiony ON, gdy P020 i P021 są ON a P022 jest OFF.



OR, OR NOT

OR OR NOT	

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OR OR NOT	Ⓢ	O	O	O	O	O	O	O	O				1			



1)OR
a)Funkcje
- Połączenie równoległe styku NO
- Czyta on/off data ze wskazanego (Ⓢ)ektu ( ), wykonuje operację OR tych danych i wyniku poprzedniej operacji, czego następnie używa jako wynik nowej operacji.

2)OR NOT
a)Funkcje
- Połączenie równoległe styku NC
- Czyta on/off data ze wskazanego (Ⓢ)ektu ( ), wykonuje operację OR tych danych i wyniku poprzedniej operacji, czego następnie używa jako wynik nowej operacji.

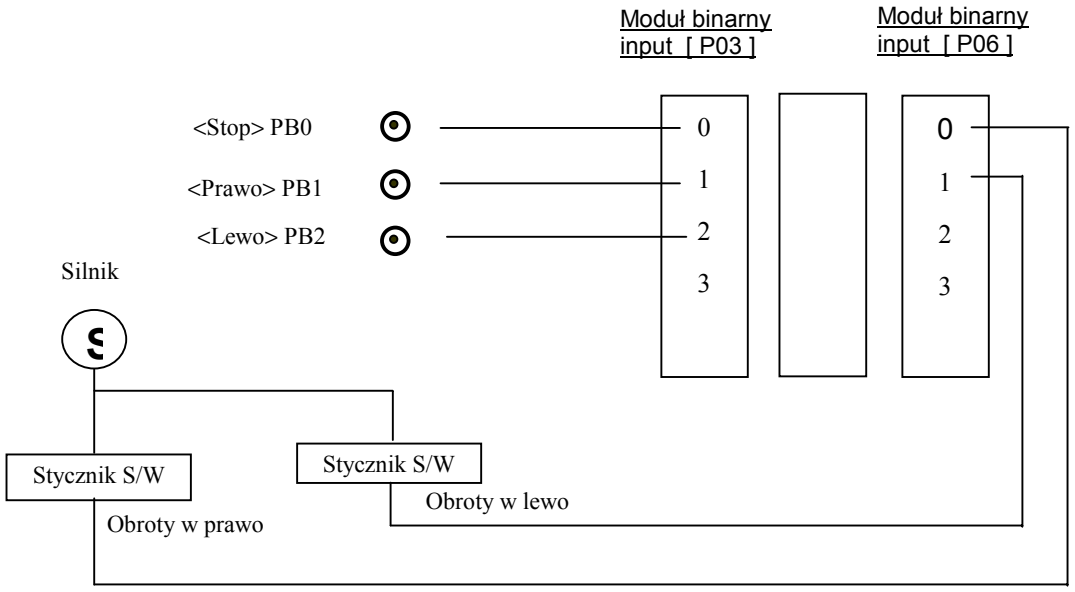
3)Przykład programu
Styk P061 zostanie ustawiony ON, gdy P020 i P021 jest ON.

Praca silnika (Przykład stosowania instrukcji LOAD, AND, OR, OUT)

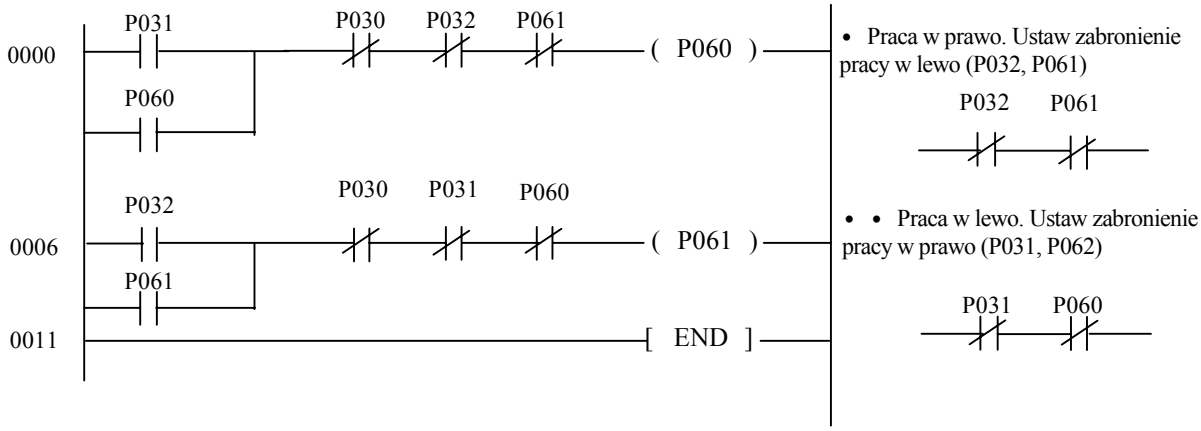
1. Działanie

Są trzy przyciski - PB0, PB1 i PB2. Gdy PB1 jest przyciśnięty, silnik startuje i obraca się w prawo. Startuje w przeciwnym kierunku (w lewo), gdy zostanie przyciśnięty przycisk PB2. PB0 jest przyciskiem bezpieczeństwa i silnik zatrzymuje się, gdy PB0 zostaje przyciśnięty.

2. Struktura systemu

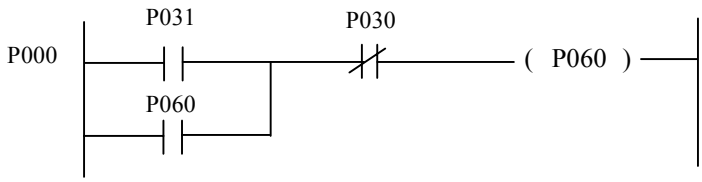


3. Program



Uwagi **[Obwód samopodtrzymujący się]**

Gdy tylko P031 jest ON, P060 ustawi się na ON i będzie tak długo w tej pozycji aż P030 ustawi się na ON.

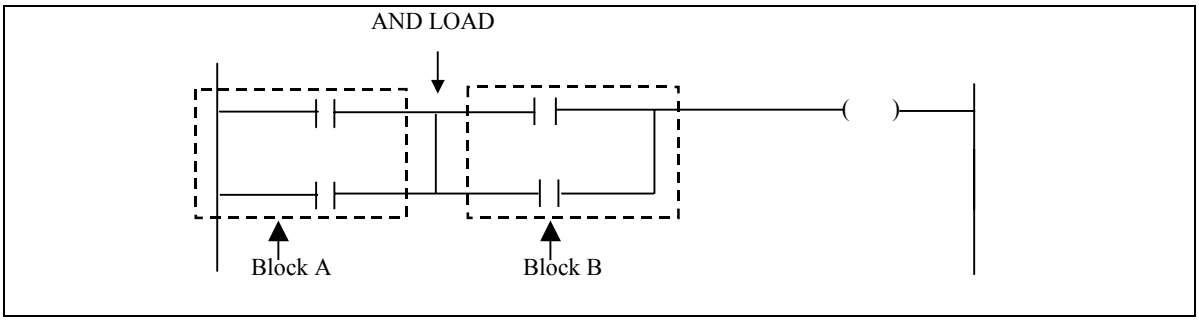


4.2 Instrukcje Connection

4.1.1 AND LOAD

AND LOAD	

Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
AND LOAD												1			

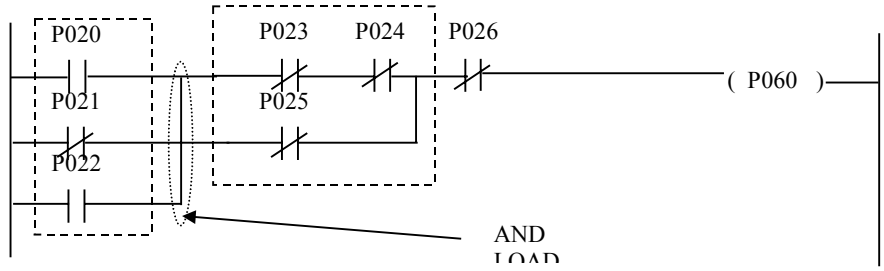


1)Funkcje

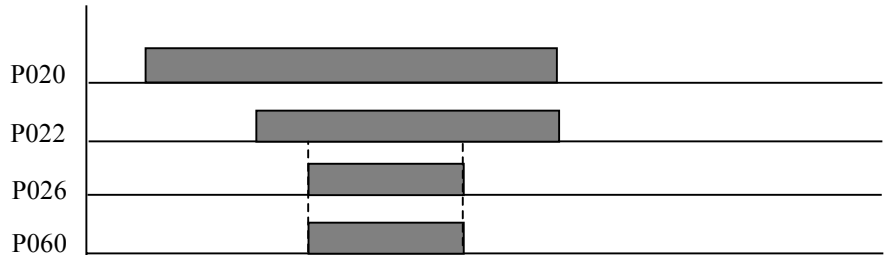
- Wykonuje operację AND bloku A i bloku B, i używa jako wyniku operacji.
- Instrukcja AND LOAD może być napisana kolejno do 7 razy.

2)Przykład programu

[Program]



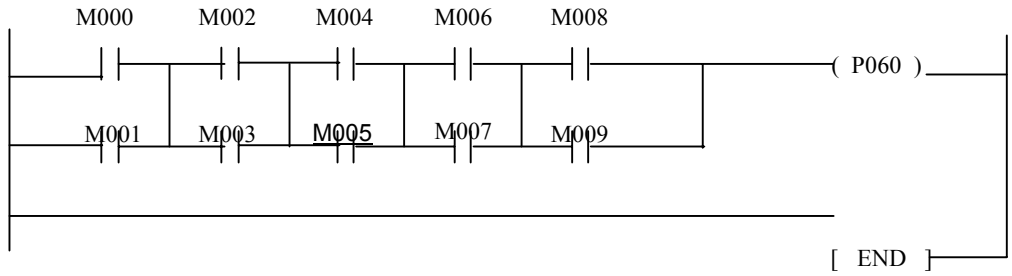
[Diagram czasowy]



[UWAGA] Kolejne użycie instrukcji AND LOAD

Są dwie metody łączenia kilku bloków szeregowo. Patrz przykład poniżej.

[Program drabinkowy]



[Program mnemoniczny]

[A] Nie używaj instrukcji AND LOAD kolejno	
LOAD	M000
OR	M001
LOAD	M002
OR	M003
AND LOAD	
LOAD	M004
OR	M005
AND LOAD	
LOAD	M006
OR	M007
AND LOAD	
LOAD	M008
OR	M009
AND LOAD	
OUT	P060

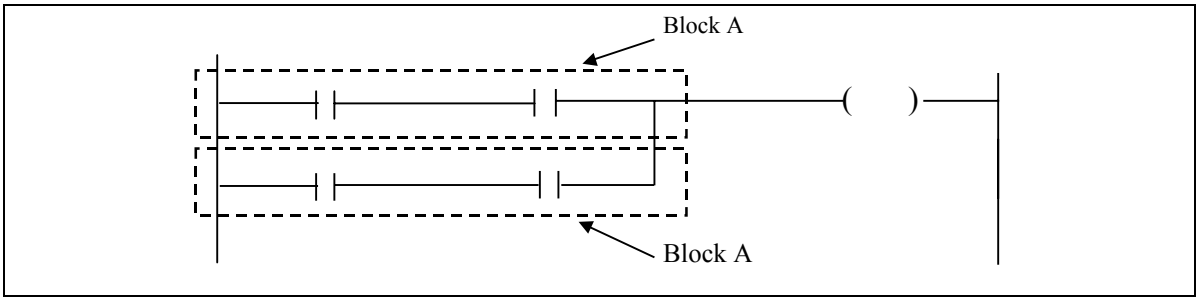
[B] Używaj instrukcji AND LOAD kolejno	
LOAD	M000
OR	M001
LOAD	M002
OR	M003
LOAD	M004
OR	M005
LOAD	M006
OR	M007
LOAD	M008
OR	M009
AND LOAD	
AND LOAD	
AND LOAD	
AND LOAD	
OUT	P060

Instrukcja AND LOAD może być użyta kojeno 7 razy (8 bloków). Gdy łączysz więcej niż 8 bloków, pisz program mnemonicznie jak pokazuje przykład [A]. Jeżeli używasz programu KGL-WIN i piszesz program drabinkowy, to KGL-WIN software dokonuje konwersji programu drabinkowego na mnemoniczny [A] automatycznie.

4.2.2 OR LOAD

OR LOAD	

Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OR LOAD												1			

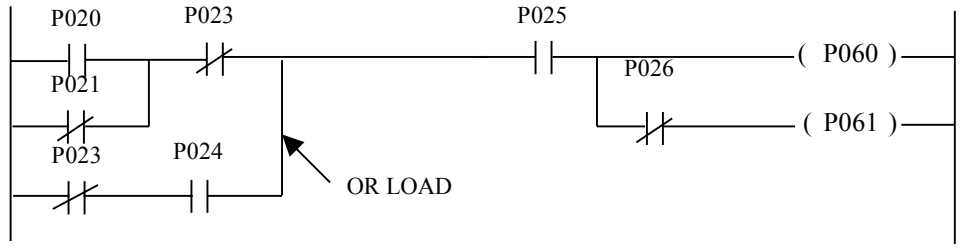


1)Funkcje

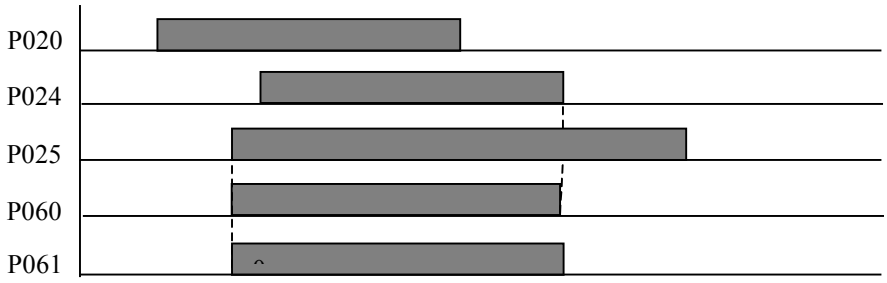
- Wykonuje operację OR na bloku A i bloku B, i używa jako wyniku operacji.
- Instrukcja OR LOAD może być napisana kolejno do 7 razy.

2)Przykład programu

[Program]



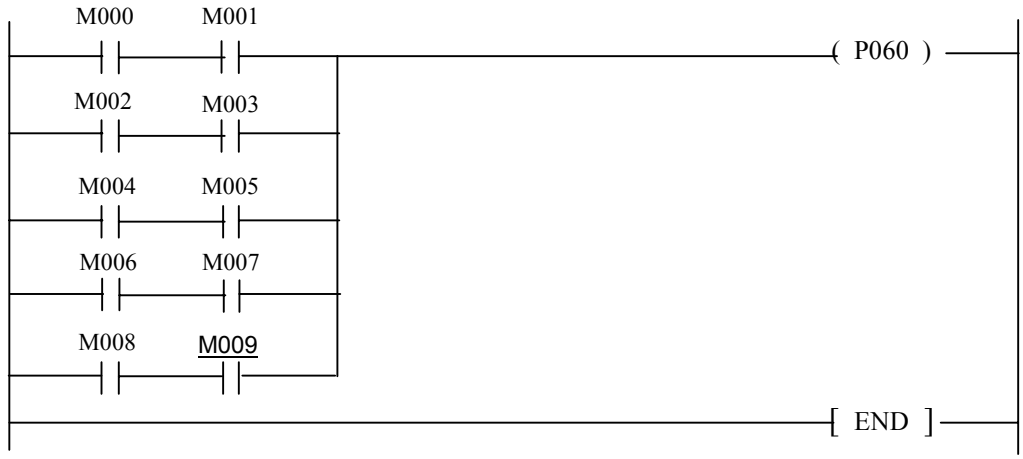
[Diagram czasowy]



[UWAGA] Kolejne użycie instrukcji OR LOAD

Są dwie metody łączenia kilku bloków równolegle. Patrz przykład poniżej.

[Program drabinkowy]



[Program mnemoniczny]

[A] Nie używaj instrukcji OR LOAD kolejno	
LOAD	M000
AND	M001
LOAD	M002
AND	M003
OR LOAD	
LOAD	M004
AND	M005
OR LOAD	
LOAD	M006
AND	M007
OR LOAD	
LOAD	M008
AND	M009
OR LOAD	
OUT	P060

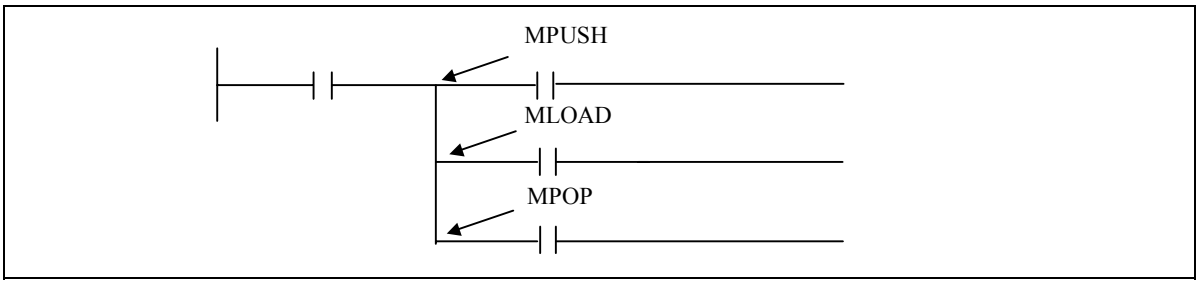
[B] Używaj instrukcji OR LOAD kolejno	
LOAD	M000
AND	M001
LOAD	M002
AND	M003
LOAD	M004
AND	M005
LOAD	M006
AND	M007
LOAD	M008
AND	M009
OR LOAD	
OR LOAD	
OR LOAD	
OR LOAD	
OUT	P060

Instrukcja OR LOAD może być użyta koljeno 7 razy (8 bloków). Gdy łączysz więcej niż 8 bloków, pisz program mnemonicznie jak pokazuje przykład [A]. Jeżeli używasz programu KGL-WIN i piszesz program drabinkowy, to KGL-WIN software dokonuje konwersji programu drabinkowego na mnemoniczny [A] automatycznie.

4.2.3 MPUSH, MLOAD, MPOP

MPUSH	FUN (005) MPUSH
MLOAD	FUN (006) MLOAD
MPOP	FUN (007) MPOP

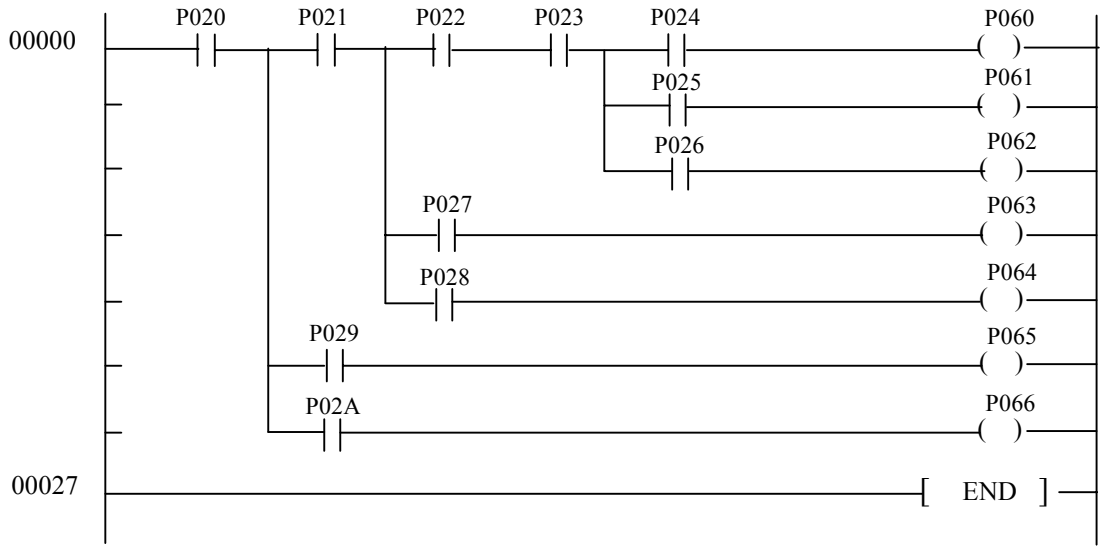
Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MPUSH MLOAD MPOP												1			



1) Funkcje
   a) MPUSH : Zapamiętuje wynik operacji (On/Off) natychmiast wykonuje instrukcję MPUSH.
   b) MLOAD : Czyta wynik operacji zapamiętany przez instrukcję MPUSH podejmuje operację od następnego kroku, z tym wynikiem operacji.
   c) MPOP :Czyta wynik operacji zapamiętany przez instrukcję MPUSH podejmuje operację od następnego kroku, z tym wynikiem operacji. Następnie zeruje wynik operacji zapamiętany przez instrukcję MPUSH.
   d) Instrukcja MPUSH może być użyta kolejno 8 razy. Liczba zostaje zredukowana o 1, gdy pomiędzy zostanie użyta instrukcja MLOAD.

2) Przykład programu

[Program drabinkowy]



[Program mnemoniczny]

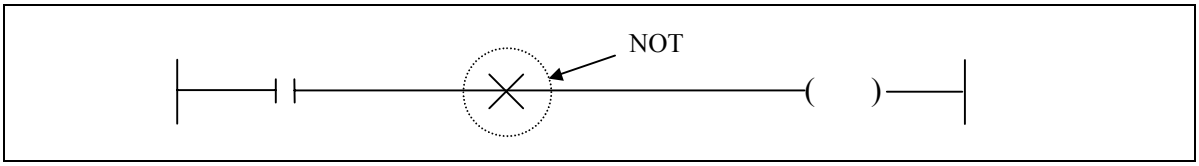
STEP	INSTRUKCJA	
0000	LOAD	P020
0001	MPUSH	
0002	AND	P021
0003	MPUSH	
0004	AND	P022
0005	AND	P023
0006	MPUSH	
0007	AND	P024
0008	OUT	P060
0009	MLOAD	
0010	AND	P025
0011	OUT	P061
0012	MPOP	
0013	AND	P026
0014	OUT	P061
0015	MLOAD	
0016	AND	P027
0017	OUT	P063
0018	MPOP	
0019	AND	P028
0020	OUT	P064
0021	MLOAD	
0022	AND	P029
0023	OUT	P065
0024	MPOP	
0025	AND	P02A
0026	OUT	P066
0027	END	
0028	NOP	
0029	NOP	
0030	NOP	

4.3 Instrukcje negacji

4.3.1 NOT

NOT	

Instrukcja	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
NOT												1			

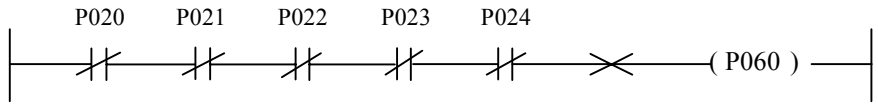


1) Funkcje
- Dokonuje inwersji wyniku operacji występującej przed instrukcją NOT.
-

Przed instrukcją NOT	Po instrukcji NOT
Styk NC	Styk NO
Styk NO	Styk NC
Połączenie szeregowe (AND)	Połączenie równoległe (OR)
Połączenie równoległe (OR)	Połączenie szeregowe (AND)

2) Przykład programu
Dwa programy poniżej wykonują tę samą operację.

Program A



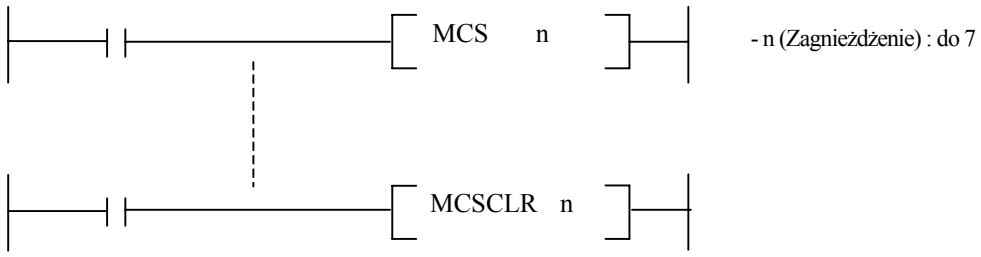
Program B

4.4 Instrukcje master control

4.4.1 MCS, MCSCLR

MCS	FUN (010) MCS
MCSCLR	FUN (011) MCSCLR

Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MCS MCSCLR											O	1			

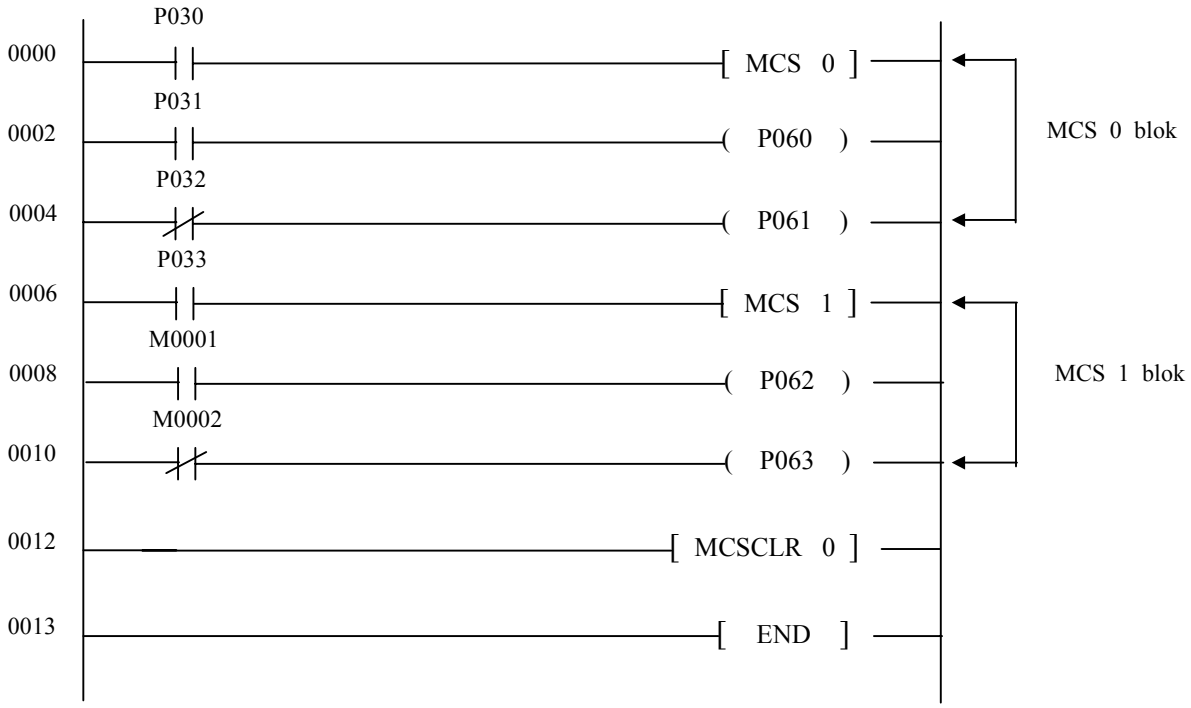


1)Funkcje

- Gdy warunek instrukcji MCS zostanie ustawiony (ON), to zostanie wykonana sekwencja programu znajdująca się pomiędzy instrukcjami MCS i MCSCLR o tym samym numerze n.
- Po każdej instrukcji MCS występuje numer (n) wskazujący priorytet master control. 0 jest najwyższym priorytetem a 7 najniższym. Instrukcja MCS powinna być używana zgodnie z porządkiem priorytetu.
- Instrukcja MCSCLR oznacza koniec master control. Gdy instrukcja MCSCLR n zostanie wykonana, wszystkie inne master control o niższym priorytecie niż ‘n’ zostają zakończone.

2) Przykład programu

Użyj 2 bloków master control (MCS 0 i MCS 1). Zerowanie następuje instrukcją MCSCLR 0. Blok MCS 1 jest zerowany automatycznie.



Uwagi

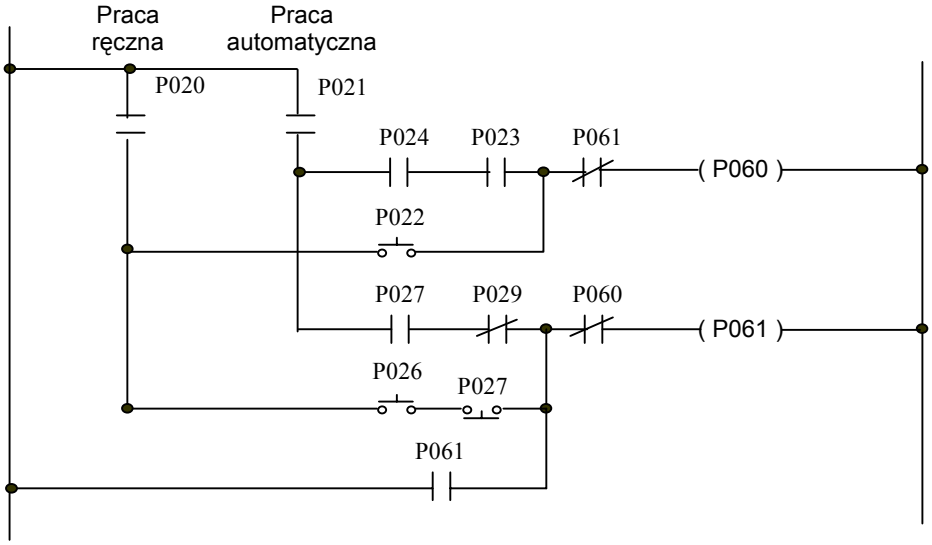
1. Scanning pomiędzy instrukcjami MCS i MCSCLR odbywa się nawet wtedy, gdy warunek na wejściu jest OFF. należy pamiętać, że czas sacn'u w tej sytuacji nie jest krótszy.
2. Gdy warunek na wejściu instrukcji MCS jest OFF, to wynik operacji pomiędzy MCS a MCSCLR jest jak pokazano poniżej.

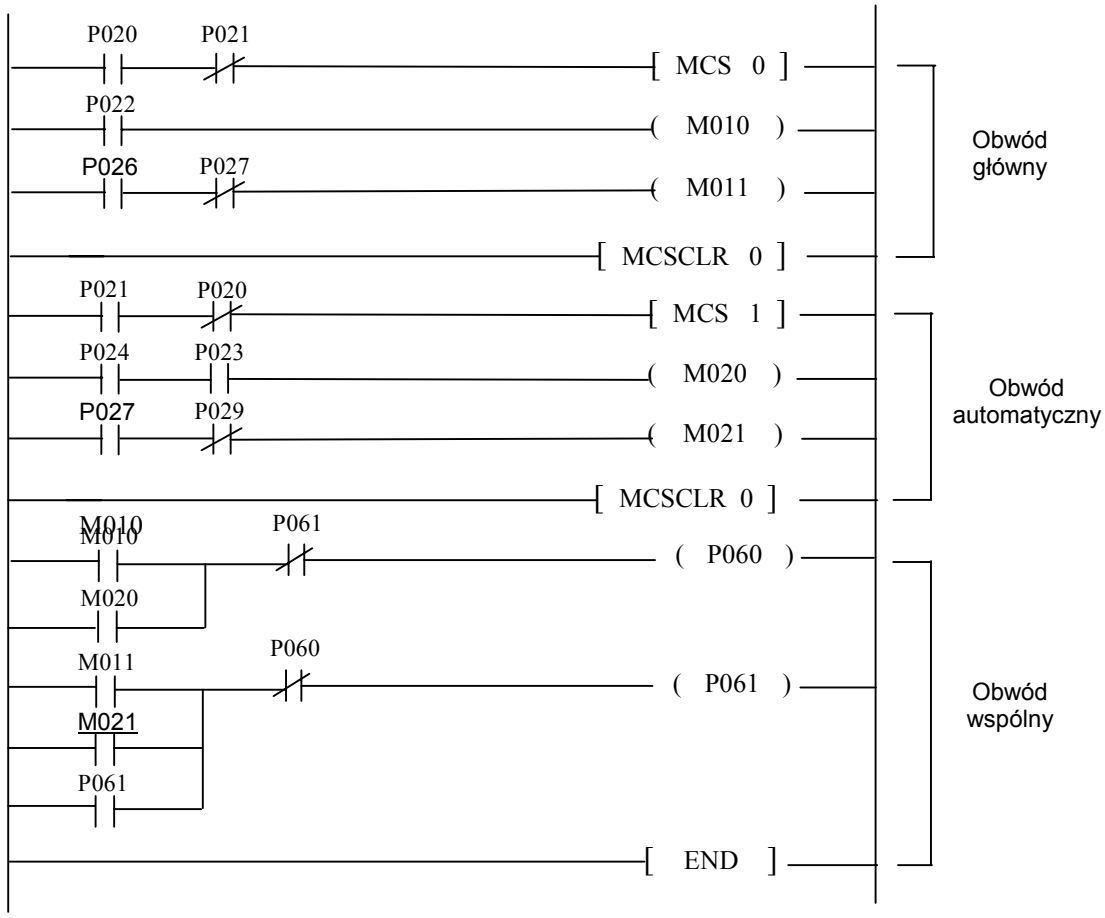
TIMER	Wyjście timera oraz jego wartość bieżąca przyjmuje wartość 0.
COUNTER	Wyjście countera oraz jego wartość bieżąca przyjmuje wartość 0.
OUT	Wszystkie OFF
SET, RST	Zachowuje bieżącą wartość

3. Jeżeli instrukcje które nie wymagają instrukcji styku tuż przed nimi (FOR, NEXT, EI, DI, etc.) a zawierają się w bloku MCS ~ MCSCLR, to CPU wykonuje te instrukcje niezależnie od statusu (On/Off) warunku na wejściu instrukcji MCS.

Obwód ze wspólną linią (Przykład instrukcji MCS i MCSCLR)

Obwód poniżej nie może być wpisany do PLC bezpośrednio. Do tego celu należy użyć instrukcji master control. (Instrukcje MCS i MCSCLR)



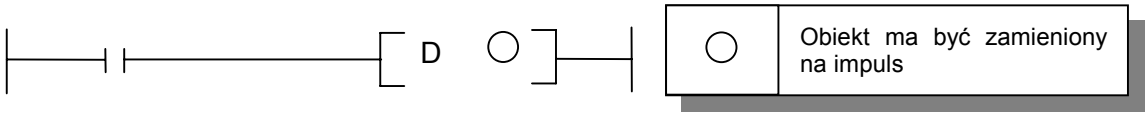


4.5 Instrukcje Output

4.5.1 D

D	FUN (017) D

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
D	Ⓓ	O	O	O									2			

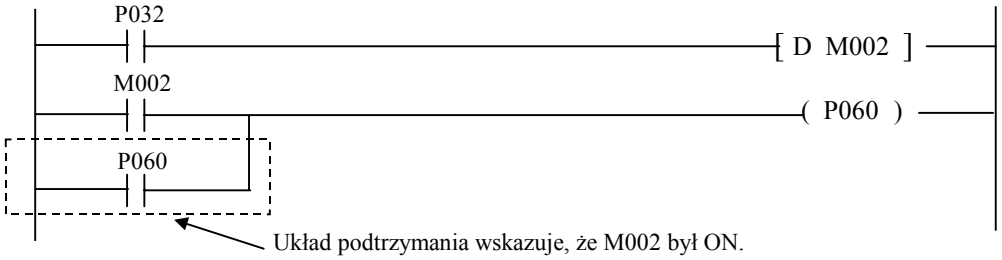


* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

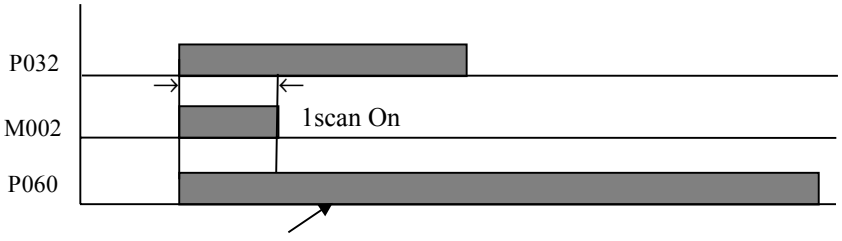
1) Funkcje
- Instrukcja D ustawia wskazany obiekt na czas jednego scan'u, gdy warunek
- na wejściu instrukcji D zostanie ustawiony ON.
- Bądź ostrożny w używaniu obszaru P jako ◯.

2) Przykład programu
Gdy P032 jest ustawione ON, to M002 jest ON na czas jednego scan'u.

[Program]



[Diagram czasowy]



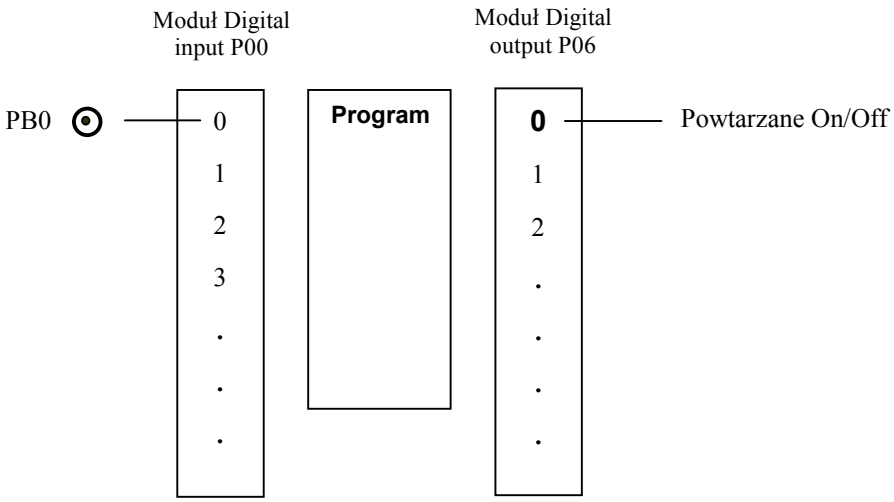
P060 zachowuje stan w wyniku działania obwodu podtrzymania.

Wysterowanie naprzemienne (Przykład instrukcji D)

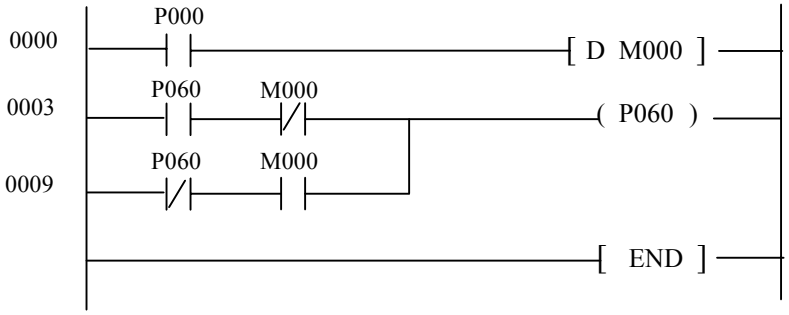
1. Praca

Gdy przycisk PB0 zostanie przyciśnięty, P060 ustawia się na ON. Gdy przycisk PB0 zostanie przyciśnięty ponownie, P060 ustawia się na OFF. P060 będzie powtarzał ON / OFF za każdym przyciśnięciem PB0.

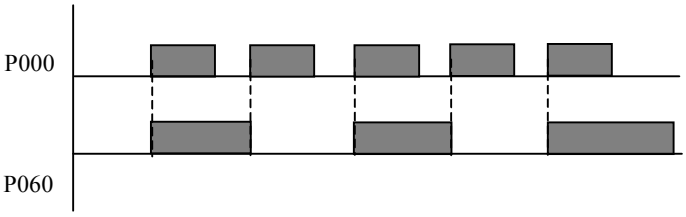
2. Struktura systemu



3. Program



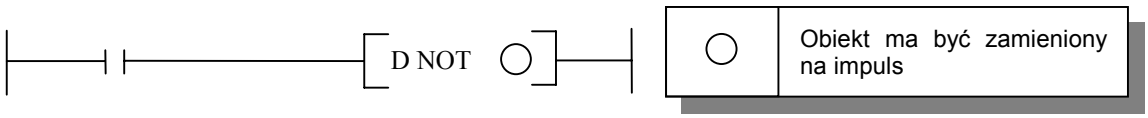
4. Diagram czasowy



4.5.2 D NOT

D NOT	FUN (018) D NOT

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
D NOT	Ⓓ	O	O	O									2			

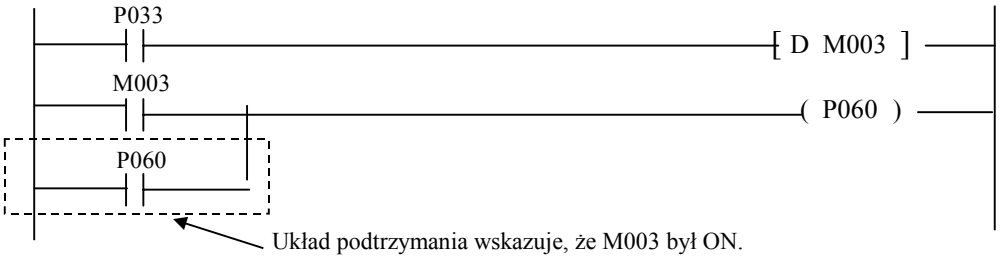


* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

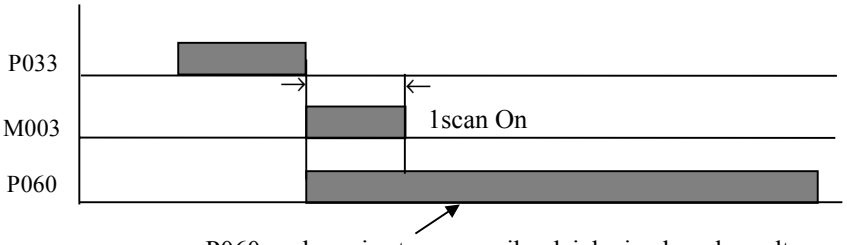
3) Funkcje
- Instrukcja D ustawia wskazany obiekt na czas jednego scan'u, gdy warunek
- na wejściu instrukcji D zostanie ustawiony ON.
- Bądź ostrożny w używaniu obszaru P jako . ◯

4) Przykład programu
Gdy P033 jest ustawione OFF, to M003 jest ON na czas jednego scan'u.

[Program]



[Diagram czasowy]

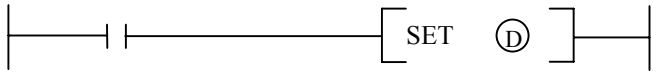


P060 zachowuje stan w wyniku działania obwodu podtrzymania.

4.5.3 SET

SET	

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SET	Ⓓ	O	O	O					O				1			



* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

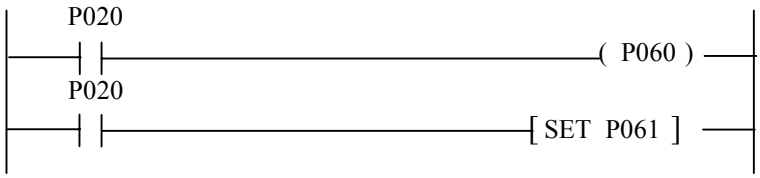
1) Funkcje

- Gdy warunek input instrukcji SET jest ustawiony na ON, to wskazany obiekt zostanie ustawiony na ON.
- Obiekt pozostanie ustawiony na ON, nawet gdy warunek input instrukcji SET został wyłączony. Obiekt może być wyłączony instrukcją RST.

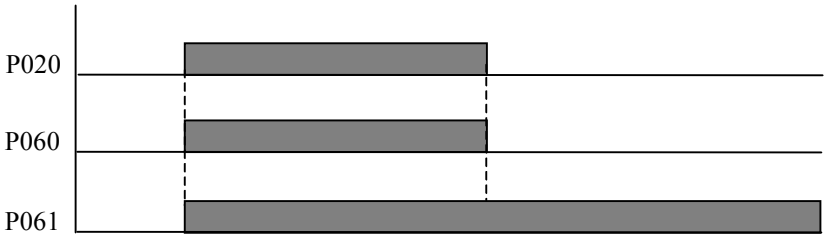
2) Przykład programu

Gdy warunek input P020 jest ON, P060 i P061 jest ustawiane na ON przez instrukcje OUT i SET.

[Program]



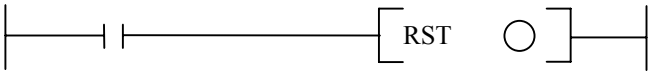
[Diagram czasowy]



4.5.4 RST

RST	

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RST	Ⓓ	O	O	O			O						1			



* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje

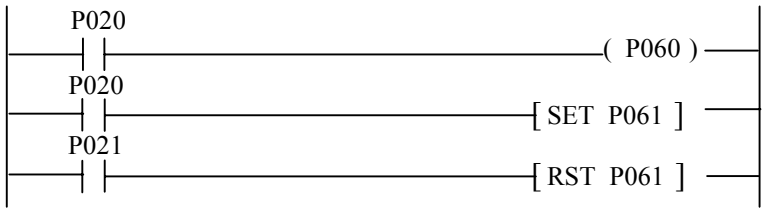
- Gdy warunek input instrukcji RST jest ustawiony na ON, to wskazany obiekt zostanie zmieniony jak poniżej.

Obiekt	Status
M, P, K, L	Wskazany bit ustawiony na OFF
T	Wyjście timera ustawia się na OFF a wartość bieżąca jest zerowana.

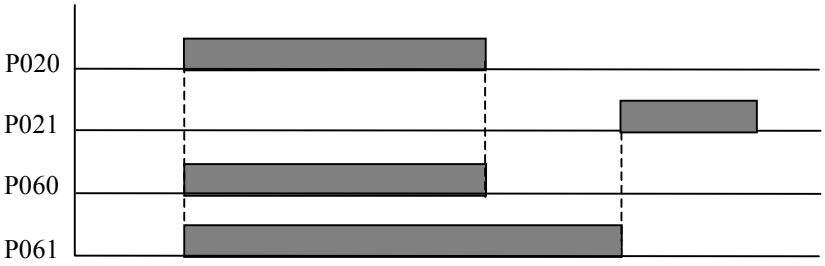
2) Przykład programu

Ustaw P061 z P020 i resetuj P061 z P021.

[Program]



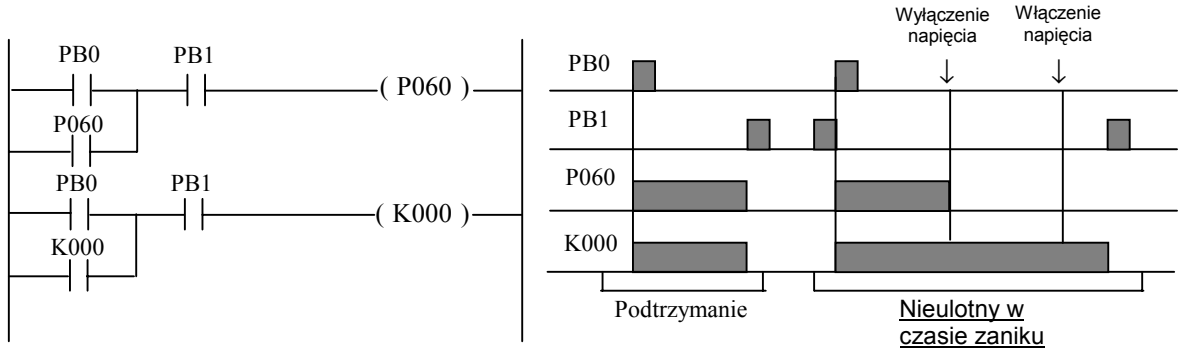
[Diagram czasowy]



Przeciwdziałanie skutkom zaniku napięcia zasilania (Różnice pomiędzy obszarem P i obszarem K)

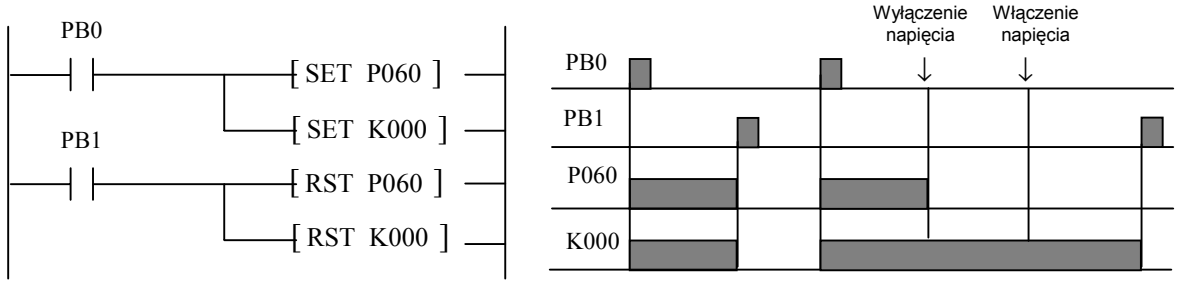
1. Różnice pomiędzy przekaźnikami I/O (P) i przekaźnikami keep (K) w stosunku do instrukcji OUT

Program poniżej pokazuje różnice pomiędzy obszarami P i obszarami K. P060 i K000 posiadają układ podtrzymania i praca tych dwóch styków jest identyczna. Jednakże, kiedy zostanie wyłączone zasilanie a następnie znowu włączone (zanik napięcia), wówczas zachowanie się P i K jest różne jak to pokazano poniżej.



2. Różnice pomiędzy przekaźnikami I/O (P) i przekaźnikami keep (K) w stosunku do instrukcji SET/RST

Instrukcja SET powoduje ustawienie wskazanego obiektu i utrzymywania jego statusu aż do wykonania instrukcji RST. Jednakże, obszary P i K zachowują się inaczej w momencie gdy nastąpi zanik napięcia.

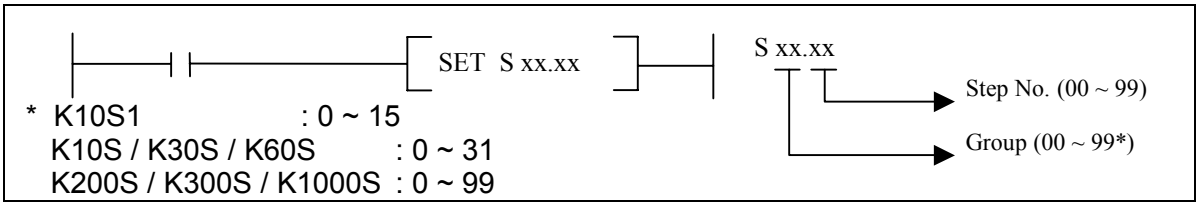


4.6 Instrukcje Step controller

4.6.1 SET Sxx.xx

SET S	

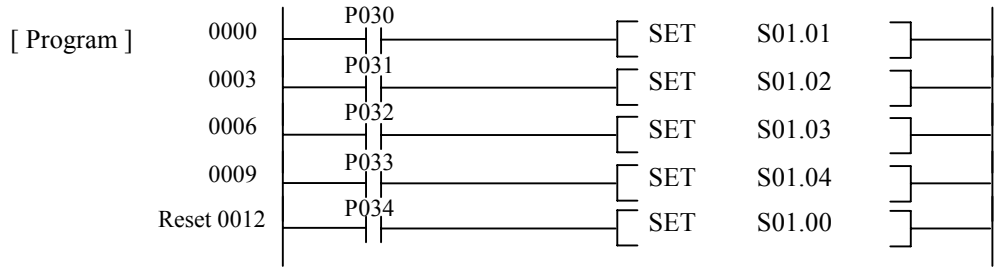
Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SET S	Ⓓ								O				2			



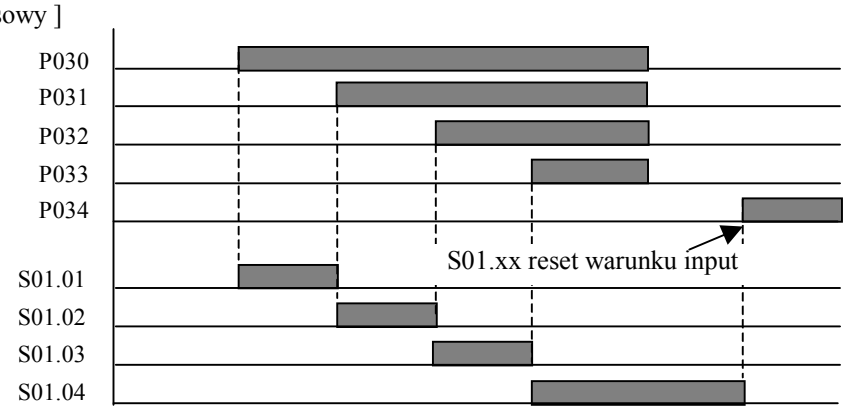
1) Funkcje
- The Sxx.xx contact will turn on when the previous step of same group and the input condition is on.
- Once a Sxx.xx is switched on, it keeps on state until the next step turns on or the step controller group is initialized. (The Sxx.00 is switched on)
- Even if multiple input condition turn on, only one step controller is switched on.
- The Sxx.00 is initialization step and the Sxx.xx group will be initialized by switching on the Sxx.00. When the CPU is turned to RUN mode, the Sxx.00 is set by default.
-

2) Przykład programu

Sekwencyjne sterowanie przy użyciu grupy S01.xx



[Diagram czasowy]

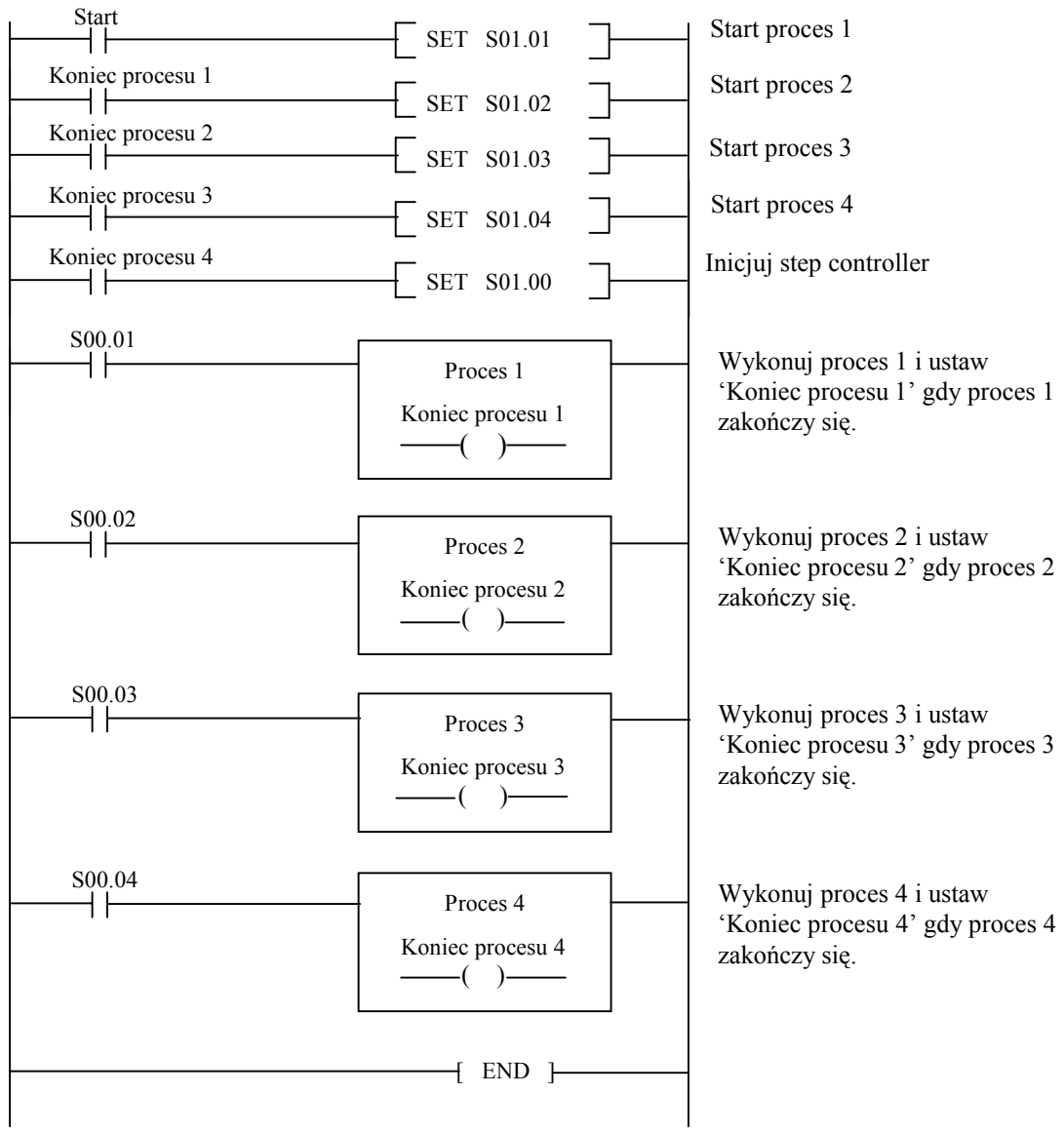


Sterowanie sekwencyjne (przykład instrukcji SET Sxx.xx)

1. Praca

Program ten pokazuje krótki przykład sterowania sekwencyjnego przy użyciu instrukcji SET Sxx.xx . W przykładzie tym są 4 procesy i każdy z nich jest realizowany sekwencyjnie. Proces 2 startuje gdy proces 1 skończył się. Proces 3 startuje gdy proces 2 skończył się. Gdy proces 4 skończył się, process1 startuje ponownie.

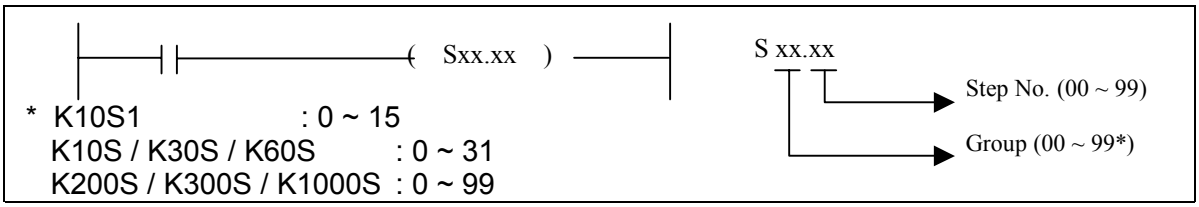
2. Program



4.6.2 OUT Sxx.xx

OUT S	

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OUT S	Ⓓ								O				2			

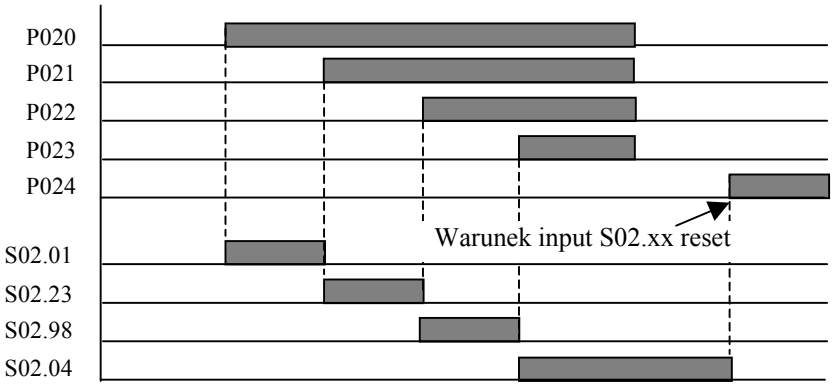


1) Funkcje
- Sterowanie z priorytetem Last-in (Włączony ostatni)
- Gdy warunek input jest na ON, wskazany step controller jest ustawiony na ON i utrzymuje swój status, aż inny step controller tej samej grupy ustawi się na ON.
- Tylko jeden step controller ustawi się na ON, nawet gdy wiele warunków input zostanie ustawionych. W tej sytuacji step controllerktóry włączył się ostatni, ma najwyższy priorytet.
- Sxx.00 jest krokiem inicjacyjnym i grupa Sxx.xx zostanie zainicjowana przez ustawienie Sxx.00 na ON. Gdy CPU wejdzie w mod RUN, to Sxx.00 zostanie ustawiony w wyniku nastawy pierwotnej.

2) Przykład programu

[Program]

[Diagram czasowy]

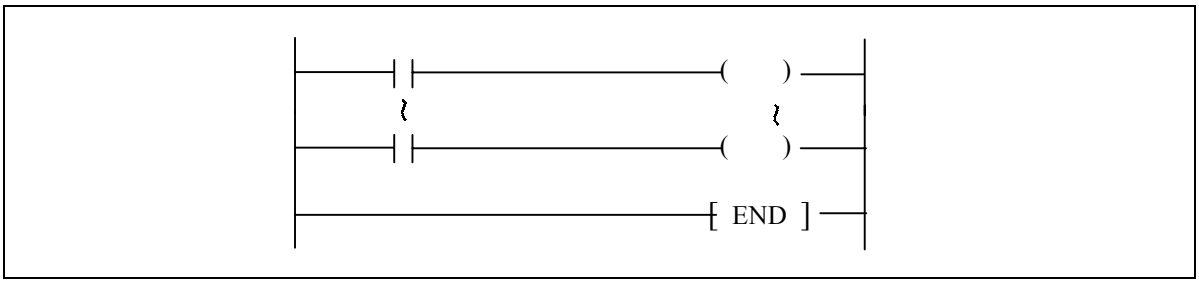


4.7 Instrukcja End

4.7.1 END

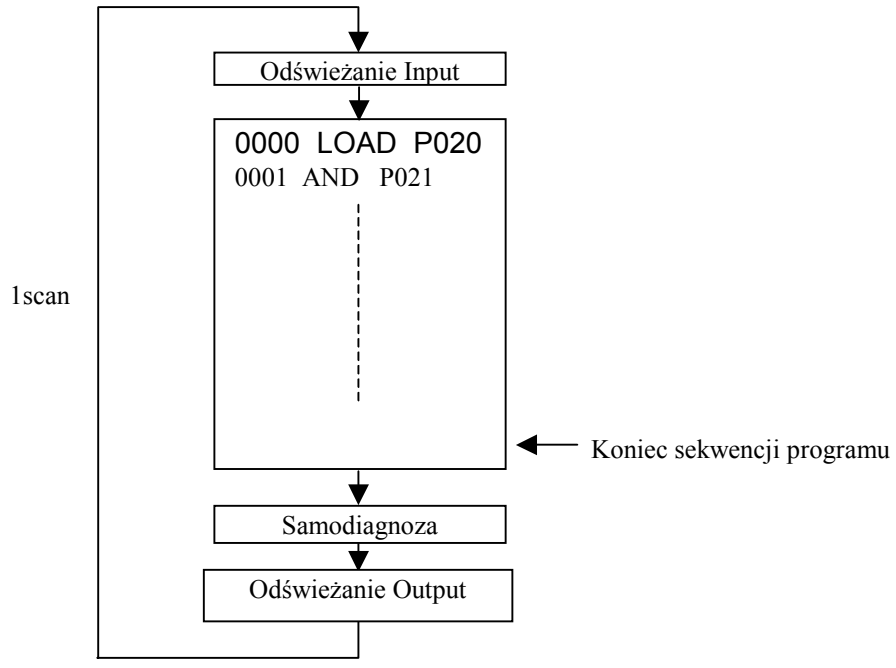
END	FUN (001) END

Instrukcja	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
END												1			



1)Funkcje

- Instrukcja END wskazuje koniec sekwencji programu. Gdy CPU napotyka instrukcję END, zatrzymuje wykonywanie sekwencji programu i wykonuje procedurę END.
- Wszystkie instrukcje występujące po instrukcji END oprócz podprogramów i procedur przerwań, są ignorowane i nie są wykonywane.
- Jeżeli brakuje instrukcji END, to pojawi się błąd programu.



4.7 Instrukcja No operation – nic nie rób

4.7.1 NOP

NOP	FUN (000) NOP

Instrukcja	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
NOP												1			

Brak symbolu drabinkowego (Dostępny tylko mod mnemoniczny)

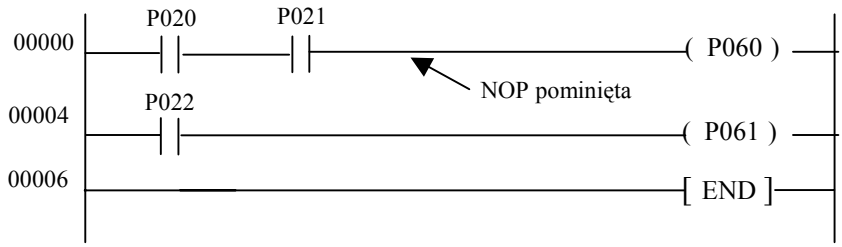
1) Funkcje
- Instrukcja no operation nie wpływu na wynik operacji poprzedzających.
- Instrukcja NOP jest używana w następujących przypadkach:
  a) Rezerwacja przestrzeni w sekwencji programu dla debuggingu.
  b) Umożliwienie usunięcia instrukcji bez zmiany num,eracji adresów.
  c) Umożliwienie tymczasowego usunięcia instrukcji.

2) Przykład programu

[Program mnemoniczny]

```
0000  LOAD   P020
0001  AND    P021
0002  NOP
0003  OUT    P060
0004  LOAD   P022
0005  OUT    P061
0006  END
```

[Program drabinkowy]

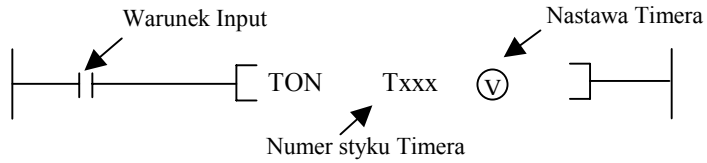


4.9 Instrukcje liczników czasu - Timer

4.9.1 TON

TON	Opóźnienie On

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TON	Txxx						O						3			
	Ⓥ									O		O				



* Nastawa czasu (t) = Jednostka czasu (0.1 lub 0.01 sek) x Nastawa Timera

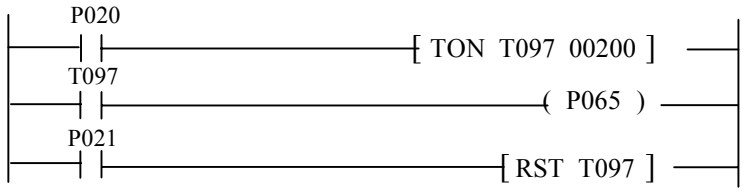
1) Funkcje

- Timer składa się ze styku timera, wartości bieżącej i nastawy.
- Wartość bieżąca timera zaczyna narastać, gdy warunek input ustawi się na ON. Wzrasta ona o 1 co każdą 0.1 lub 0.01 sek, aż osiągnie nastawę lub warunek input zostanie zgaszony.
- Styk timera zostanie ustawiony na ON, gdy wartość bieżąca osiągnie nastawę.
- Styk timera i wartość bieżąca zostanie wyzerowana, gdy warunek input zostanie zgaszony lub zostanie wykonana instrukcja RST.

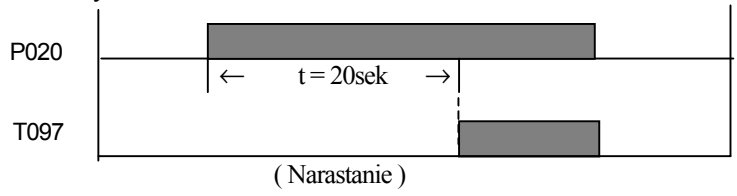
2) Program example

T097 (0.01 sek timer) zostanie ustawiony na ON w 20 sekund po załączeniu P020 na ON.

[Program]



[Diagram czasowy

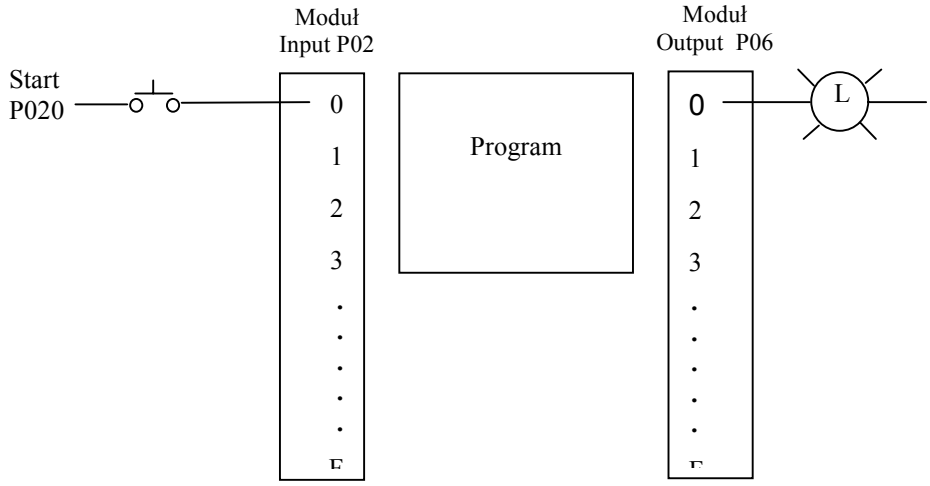


Migająca lampka (przykład instrukcji TON)

1. Praca

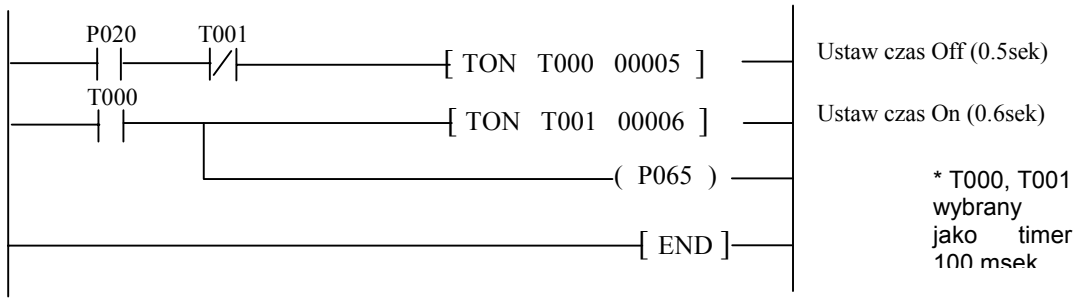
Zastosowanie dwóch timerów – lampka miga gdy P020 jest ON.

2. Struktura układu

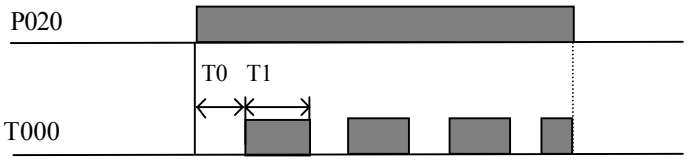


3. Program

[Program drabinkowy]



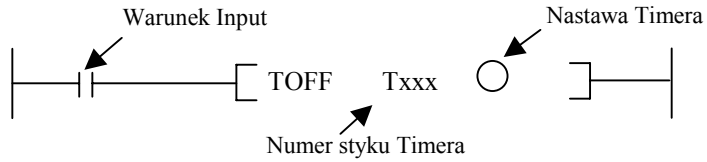
[Diagram czasowy]



4.9.2 TOFF

TOFF	Opóźnienie Off

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TOFF	Txxx						O						3			
	Ⓥ									O		O				

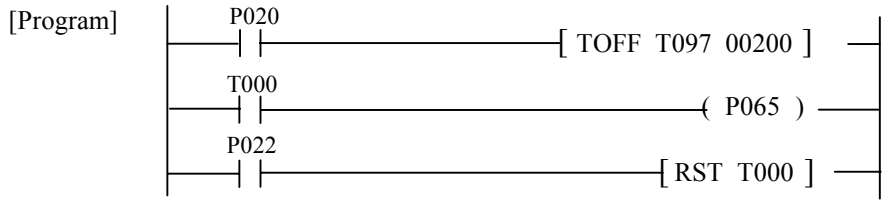


* Nastawa czasu (t) = Jednostka czasu (0.1 lub 0.01 sek) x Nastawa Timera

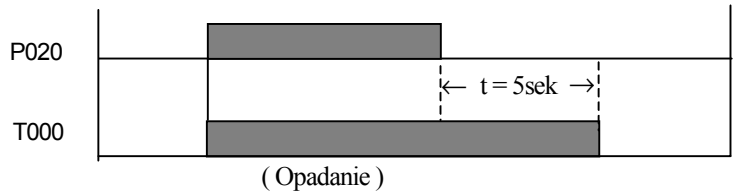
1) Funkcje
   - Timer składa się ze styku timera, wartości bieżąceji nastawy.
   - Gdy warunek input zostanie ustawiony, to wartość bieżąca przyjmie nastawę timera i styk timera ustawi się na ON.
   - Gdy warunek input zostanie zgaszony, to wartość bieżąca będzie maleć o 1 co każdą 0.1 lub 0.01 sek, aż osiągnie 0 lub warunek input zostanie zgaszony.
   - Styk timera zostanie zgaszony, gdy wartość bieżąca osiągnie 0.
   - Gdy warunek input zostanie zgaszony lub zostanie wykonana instrukcja RST, to styk timera i wartość bieżąca zostanie wyzerowana.

2) Przykład programu

T000 (0.1 sek timer) zostanie zgaszony w 5 sekund po wyłączeniu P020.



[Diagram czasowy]

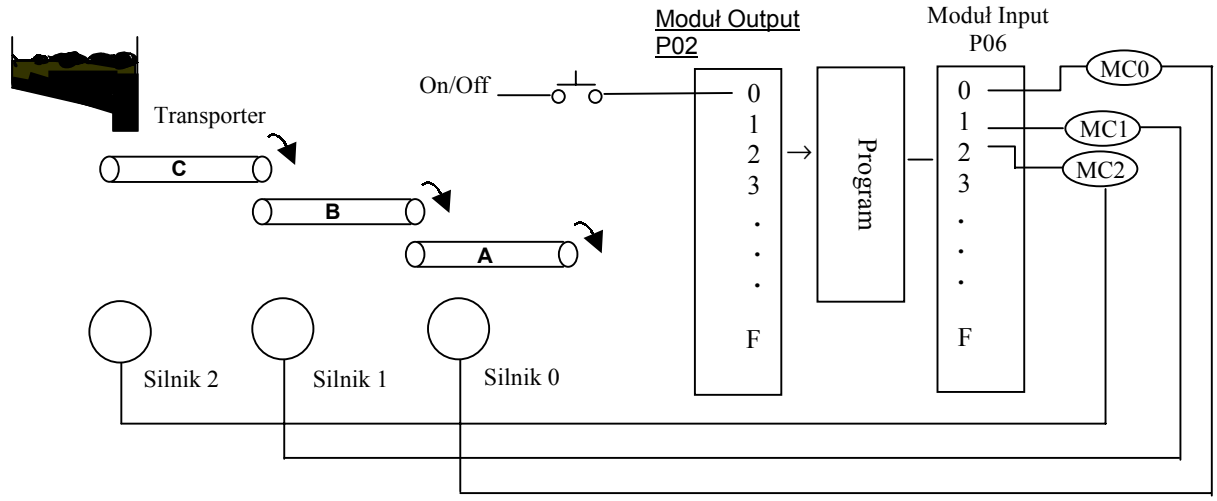


Sterowanie transporterem (przykład instrukcji TOFF)

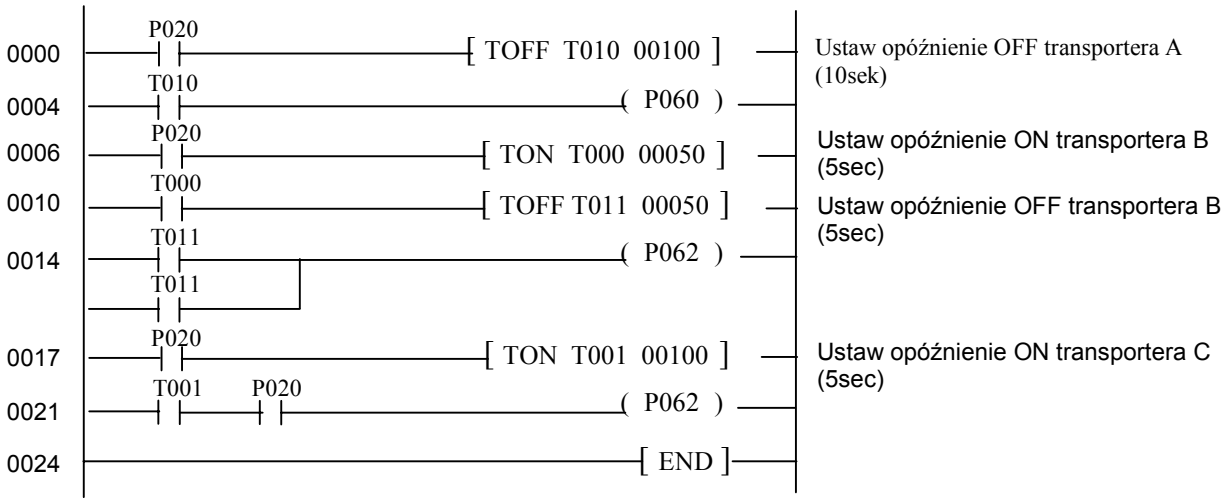
1. Praca

Trzy transportery (A, B, C) pracują sekwencyjnie używając timerów TOFF.
(Start : A – B – C, Stop : C – B – A)

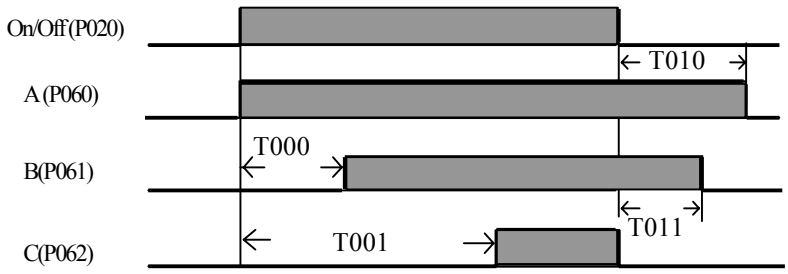
2. Struktura systemu



3. Program



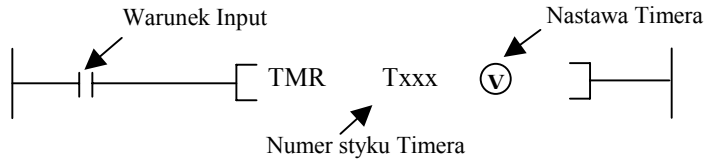
[Diagram czasowy]



4.9.3 TMR

TMR	Timer Sumujący

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TMR	Txxx						O						3			
	Ⓥ									O		O				

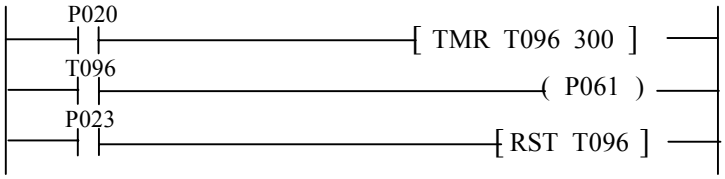


* Nastawa czasu (t) = Jednostka czasu (0.1 lub 0.01 sek) x Nastawa Timera

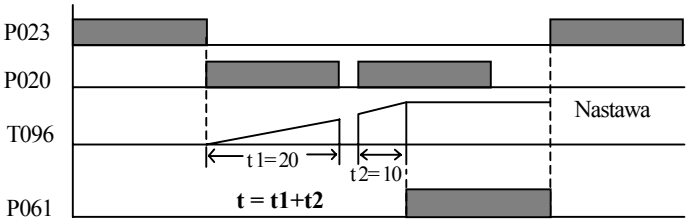
1) Funkcje
- Wartość bieżąca wzrośnie o 1, gdy warunek input jest ON.
- Gdy wartość bieżąca osiągnie nastawę timera, to styk timera ustawi się na ON.
- Nawet gdy warunek input jest OFF, to wartość bieżąca nie jest zerowana. Jeżeli timer używany jest w obszarze danych nieulotnych, to timer zachowa wartość bieżącą podczas wyłączenia CPU.
- Gdy warunek input zostanie zgaszony lub zostanie wykonana instrukcja RST, to styk timera i wartość bieżąca zostanie wyzerowana.

2) Przykład programu

[Program]



[Diagram czasowy]

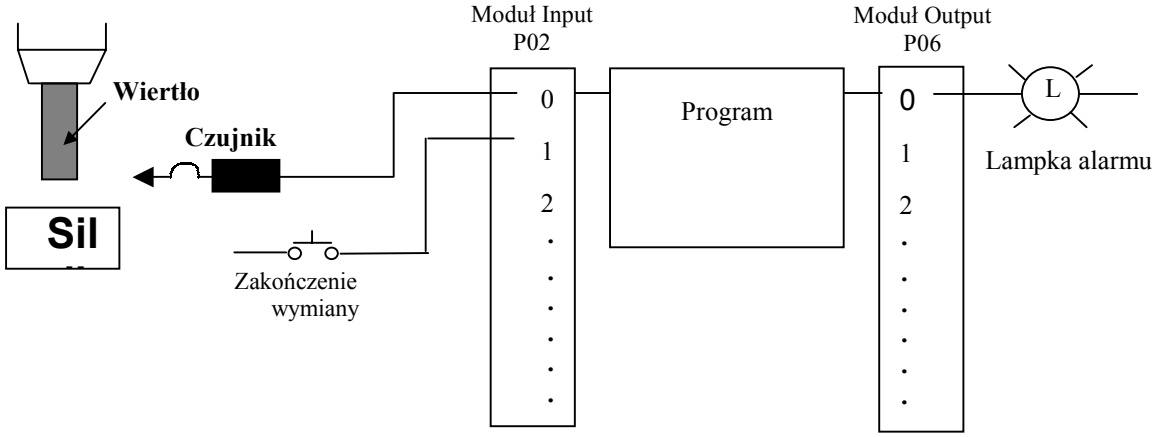


Alarm – wymiana wiertła (przykład instrukcji TMR)

1. Praca

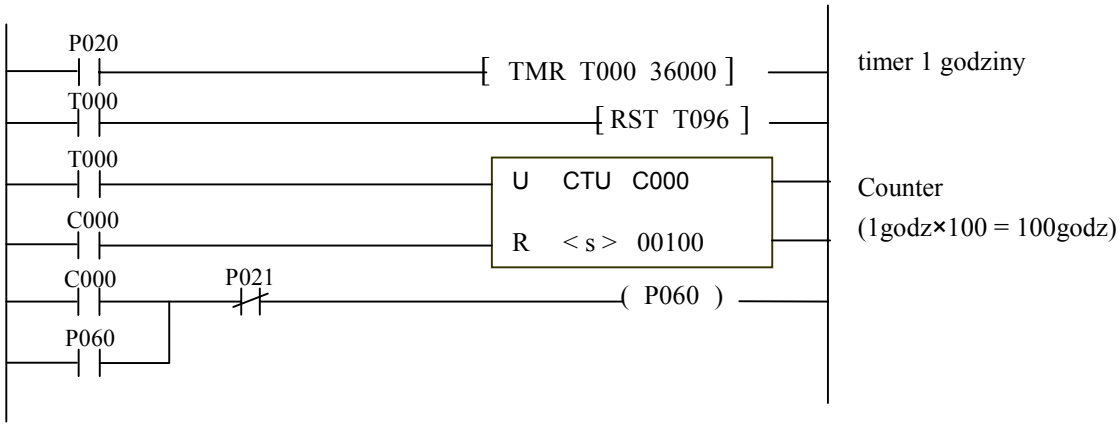
W centrach maszynowych całkowity czas używania wiertła jest kontrolowany przez PLC. Jeżeli całkowity czas używania przekroczy czas życia wiertła (100 hours), PLC wystawia sygnał alarmu pokazując, że należy wymienić wirtło.

2. Struktura systemu



I/O	Opis
P020	Detekcj opuszczenia wiertła
P021	Zakończenie wymiany
P060	Zapal lamkę alarm
T000	Timer czasu życia wiertła

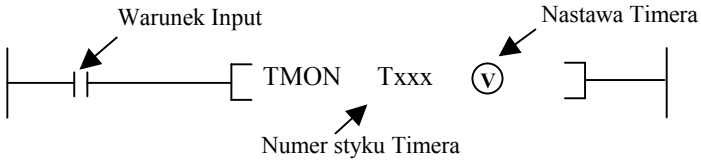
3. Program



4.9.4 TMON

TMON	Timer monostabilny

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TMON	Txxx						O						3			
	ⓥ									O		O				

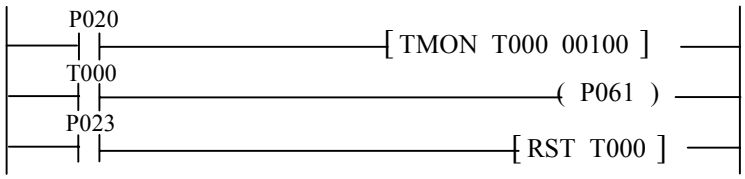


* Nastawa czasu (t) = Jednostka czasu (0.1 lub 0.01 sek) x Nastawa Timera

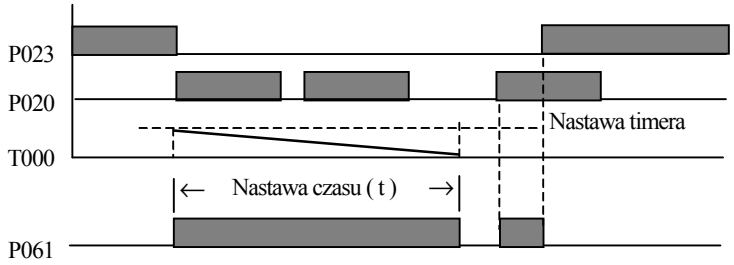
1) Funkcje
   - Gdy warunek input ustawi się na ON, to wartość bieżąca zostanie ustawiona na nastawę i zacznie maleć. Styk timera ustawi się na ON, gdy warunek input ustawi się na ON.
   - Gdy warunek input ustawi się na OFF, to wartość bieżąca zmniejszy się o 1 co każdą 0.1 lub 0.01 sek, aż osiągnie 0 i styk timera zostanie zgaszony gdy tylko wartość bieżąca osiągnie wartość 0.
   - Podczas pracy timera, zmiany on/off warunku input są ignorowane.
   - Gdy wykonywana jest instrukcja RST, styk timera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.

2) Przykład programu

[Program]



[Diagram czasowy]

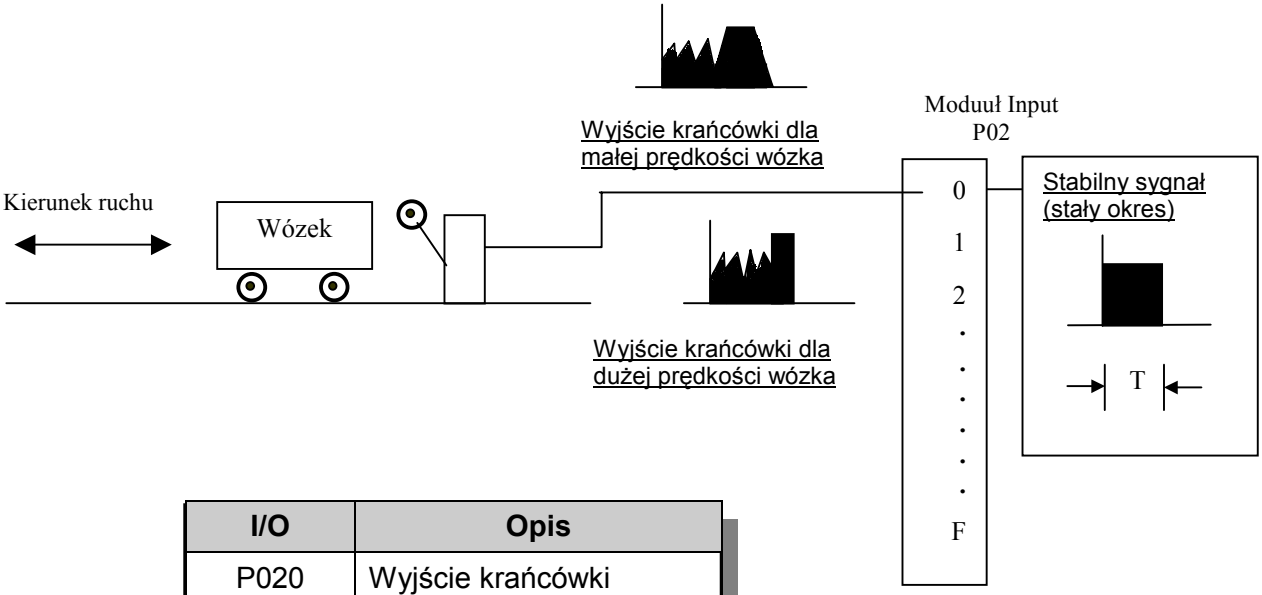


Układ zapobiegający drganiom (przykład instrukcji TMON)

1. Praca

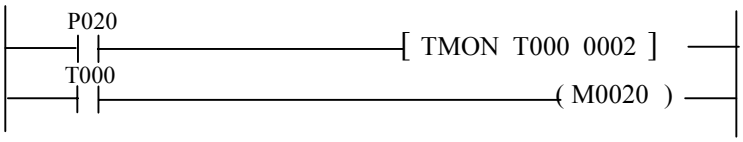
Sygnał wejściowy z krańcówki ma charakter drgań. Przy zastosowaniu instrukcji TMON, można uzyskać sygnał stabilny.

2. **Struktura systemu**



I/O	Opis
P020	Wyjście krańcówki
M0020	Stabiliny sygnał P020
T000	Timer zapobiegający drganiom

3. Program

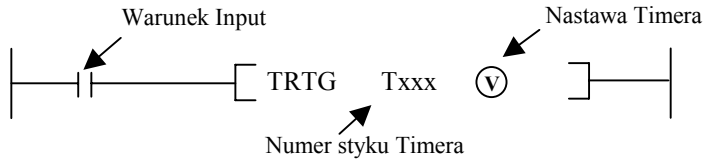


Jeżeli P020 drga przez moment po załączeniu P020, to M0020 utrzymuje go w stanie ON przez 0,2 sekundy.

4.9.5 TRTG

TRTG	Timer ponownie wyzwalany

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TRTG	Txxx						O						3			
	Ⓥ									O		O				

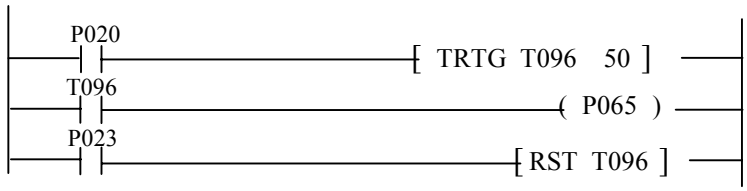


* Nastawa czasu (t) = Jednostka czasu (0.1 lub 0.01 sek) x Nastawa Timera

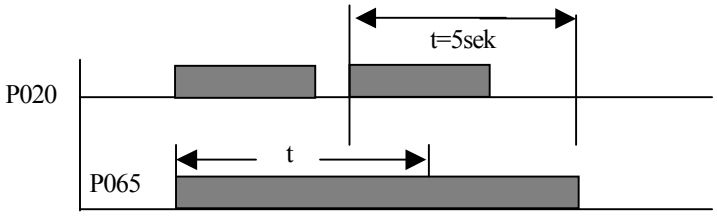
1) Funkcje
   - Gdy warunek input ustawi się na ON, to wartość bieżąca zostanie ustawiona jako nastawa timera i zacznie maleć. Styk timera ustawi się na ON, gdy warunek input jest ustawiony na ON.
   - Wartość bieżąca będzie zmniejszała się o 1 co każdą 0.1 lub 0.01sek aż do osiągdnięcia 0, a styk timera zostanie zgaszony gdy wartość bieżąca osiągnie 0.
   - Jeżeli warunek input ustawi się ponownie na ON podczas pracy timera, to wartość bieżąca zostanie zresetowana , czyli ustawiona jako nastawa timera i timer zacznie pracę od początku.
   - Gdy wykonywana jest instrukcja RST, styk timera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.

2) Przykład programu

[Program]



[Diagram czasowy]

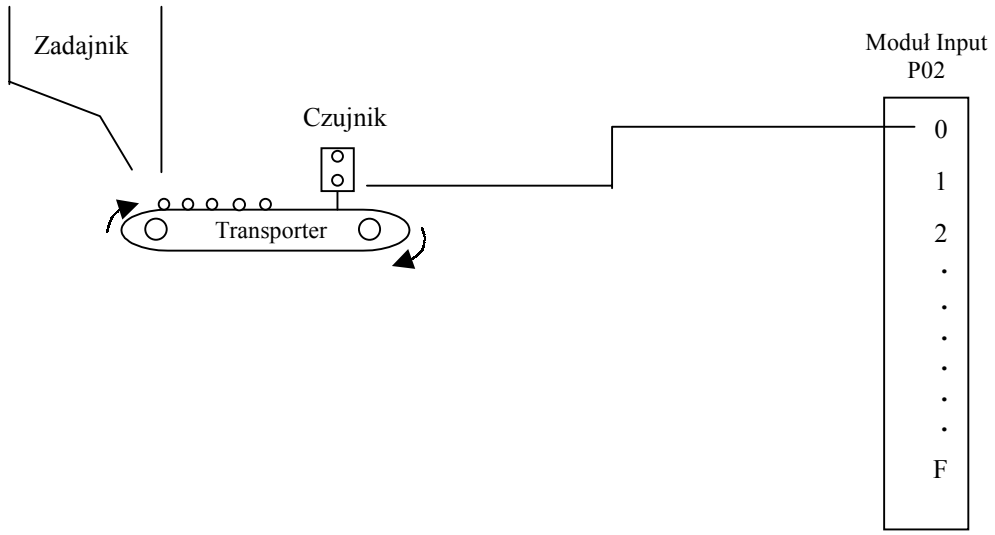


Układ wykrywający błędy transportera (przykład instrukcji TRTG)

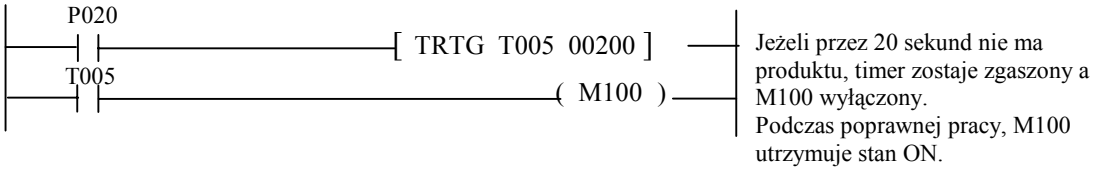
1. Praca

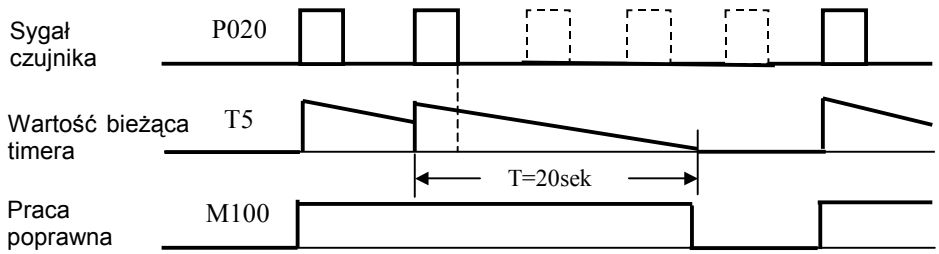
Wykrywaj błąd transportera sprawdzając czy produkt pojawił się w wyznaczonym czasie czy nie.

2. Struktura systemu



3. Program



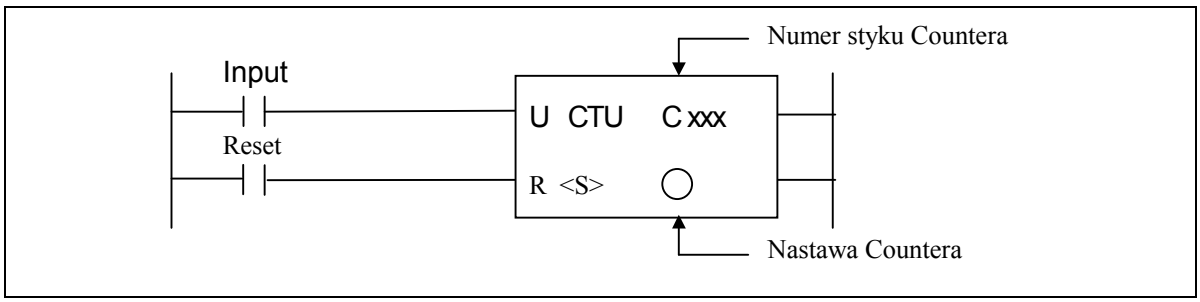


4.10 Instrukcje liczników - Counter

4.10.1 CTU

CTU	Up counter

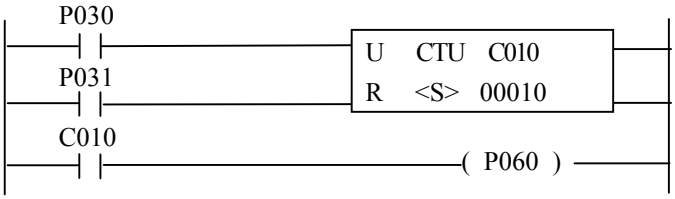
Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTU	Cxxx							O					3			
	ⓥ									O		O				



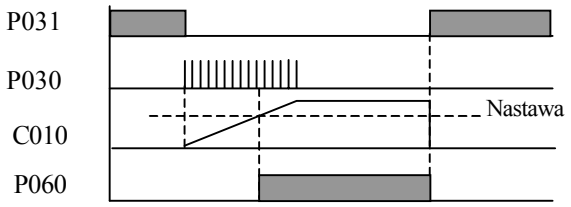
1)Funkcje
- Każde narastające zbocze pojawiające na wejściu input, zwiększa wartość bieżącą o 1.
- Początkowo wartość bieżąca ma wartość 0, a gdy wartość bieżąca zrówna się z nastawą, to styk countera ustawi się na ON.
- Po tym jak styk countera ustawi się na ON, wartość bieżąca wzrasta do wartości maksymalnej. (65535)
- Gdy wykonywana jest instrukcja RST, styk countera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.
-

2)Przykład programu
- Za każdą zmianą P030 z OFF na ON, wartość bieżąca C010 zwiększa się o 1.
- P031 jest warunkiem reset.

[Program]



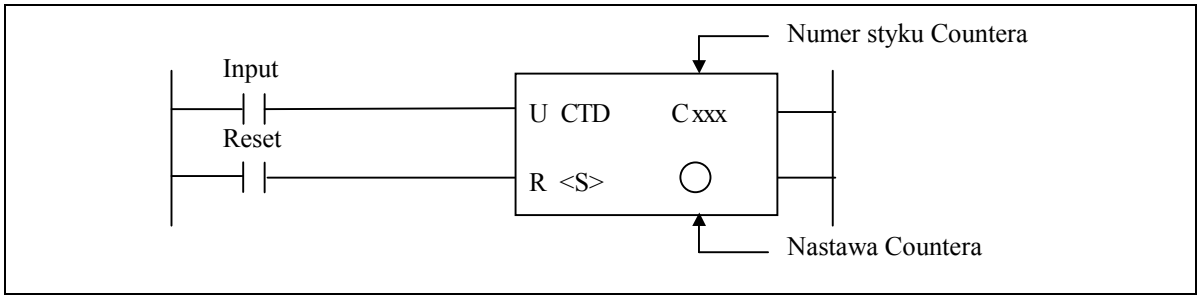
[Diagram czasowy]



4.10.2 CTD

CTD	Down counter

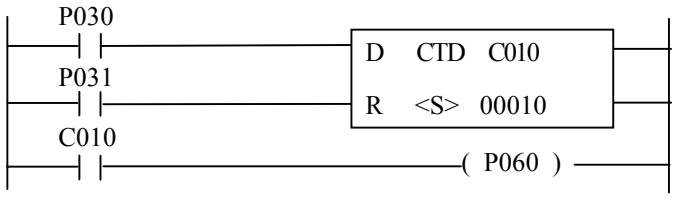
Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTD	Cxxx							O					3			
	Ⓥ									O		O				



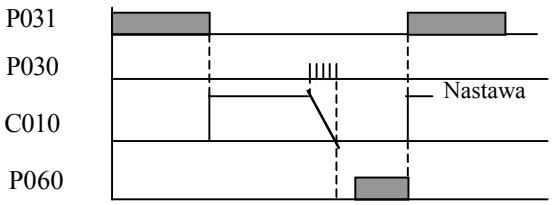
1) Funkcje
- Każde narastające zbocze pojawiające na wejściu input, zmniejsza wartość bieżącą o 1.
- Początkowo wartość bieżąca ma wartość nastawy, a gdy wartość bieżąca osiągnie 0, to styk countera ustawi się na ON.
- Gdy wykonywana jest instrukcja RST, styk countera zostaje zgaszony a do wartości bieżącej zostaje wpisana wartość nastawy.

2) Przykład programu
- Za każdą zmianą P030 z OFF na ON, wartość bieżąca C010 zmniejsza się o 1.
- P031 jest warunkiem reset.

[Program]



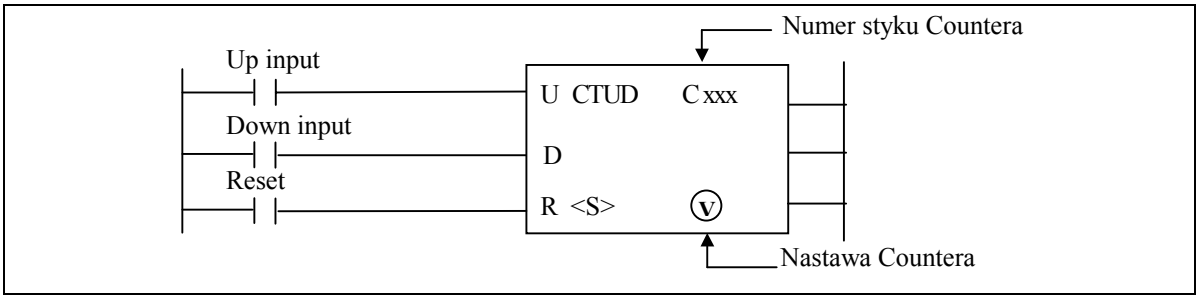
[Diagram czasowy]



4.10.3 CTUD

CTUD	Up-down counter

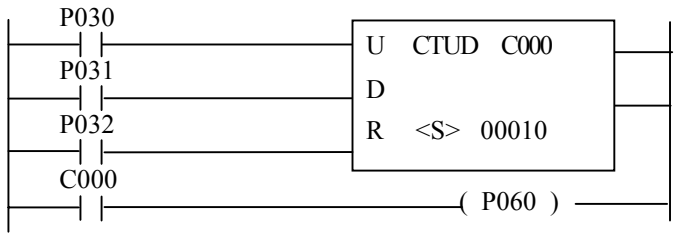
Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTUD	Cxxx							O					3			
	Ⓥ									O		O				



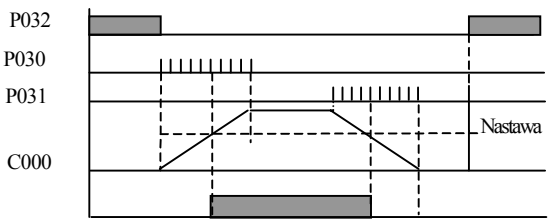
1) Funkcje
- Każde narastające zbocze pojawiające na wejściu Up input, zwiększa wartość bieżącą o 1.
- Każde narastające zbocze pojawiające na wejściu Down input, zmniejsza wartość bieżącą o 1.
- Początkowo wartość bieżąca ma wartość 0.
- Styk countera ustawia się na ON, gdy wartość bieżąca zrówna się lub będziewiększa od nastawy.
- Gdy wykonywana jest instrukcja RST, styk countera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.

2) Przykład programu
- P030 jest up input, a P031 jest down input.
- P032 jest sygnałem reset.
-

[Program]



[Diagram czasowy]

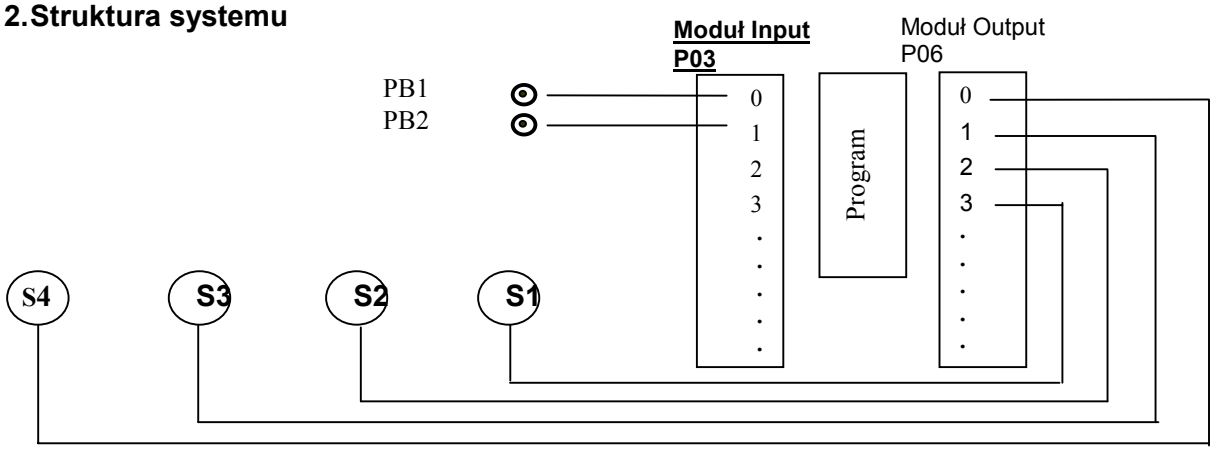


Układ sterowania pracą silników (przykład instrukcji CTUD)

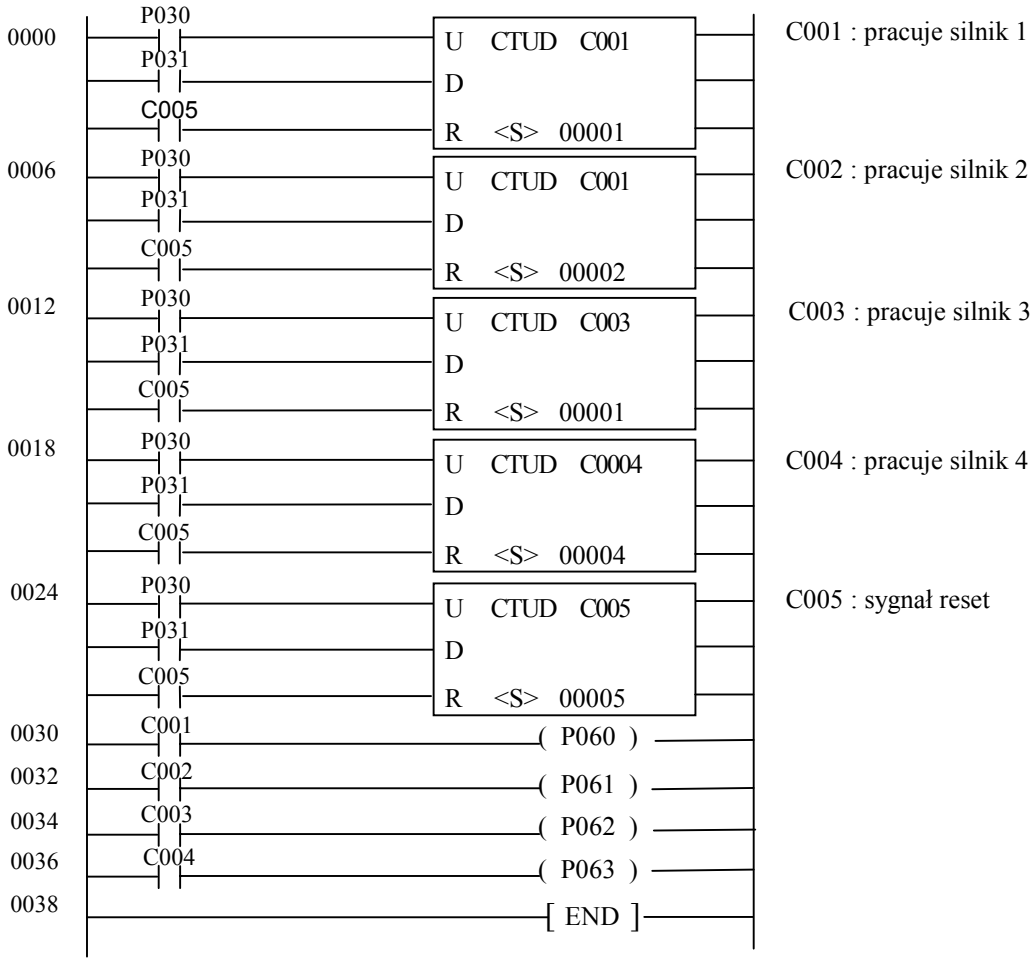
1. Praca

Praca 4 silników sterowana jest przez PLC. Za każdym przyciśnięciem przycisku PB1, ilość pracujących silników zwiększa się o 1. Przyciśniecie PB2 zmniejsza ilość pracujących silników. Gdy pracują 4 silniki i zostanie użyty przycisk PB1, to wszystkie silniki zatrzymają się.

2. Struktura systemu



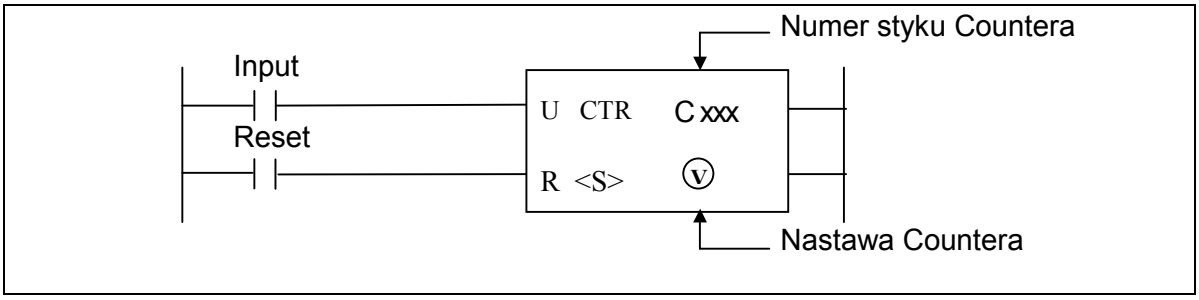
3. Program



4.10.4 CTR

CTR	Ring counter

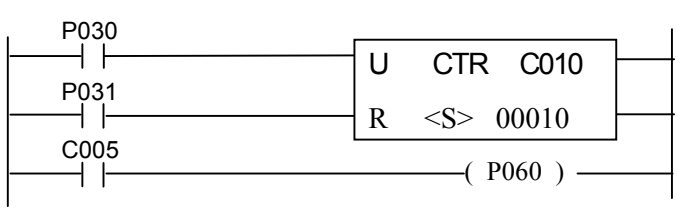
Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	# D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CTR	Cxxx							O					3			
	◯									O		O				



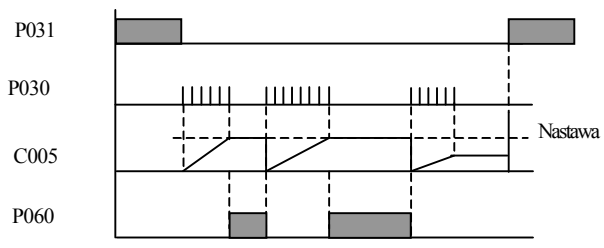
1) Funkcje
- Każde narastające zbocze pojawiające na wejściu input, zwiększa wartość bieżącą o 1.
- Gdy wartość bieżąca zrówna się z nastawą, to styk countera ustawi się na ON. Następnie gdy pojawi się narastające zbocze na wejściu input, to styk countera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.
- Gdy wykonywana jest instrukcja RST, styk countera zostaje zgaszony a wartość bieżąca zostaje wyzerowana.

2) Przykład programu
- P030 jest input i kiedy wartość bieżąca jest taka sama jak nastawa, to styk countera ustawi się na ON.
- Gdy P030 jest włączony po raz jedenasty, to styk countera (P060) zostaje zgaszony a wartość bieżąca ptrzyjmuje wartość 0.

[Program]



[Diagram czasowy]



5 Instrukcje Application

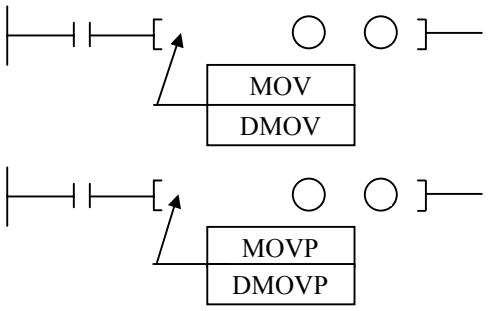
5.1 Instrukcje Data transfer

5.1.1 MOV, MOVP, DMOV, DMOVP

MOV (Move)	FUN(80) MOV FUN(82) DMOV FUN(81)MOVP FUN(83)DMOCP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	**P**	**K**	**L**	**F**	**T**	**C**	**S**	**D**	**#D**	Integer		**Error (F110)**	**Zero (F111)**	**Carry (F112)**
MOV(P)	○	O	O	O	O*	O	O	O		O	O	O	5/7	O		
DMOV(P)	○	O	O	O	O*		O	O		O	O					



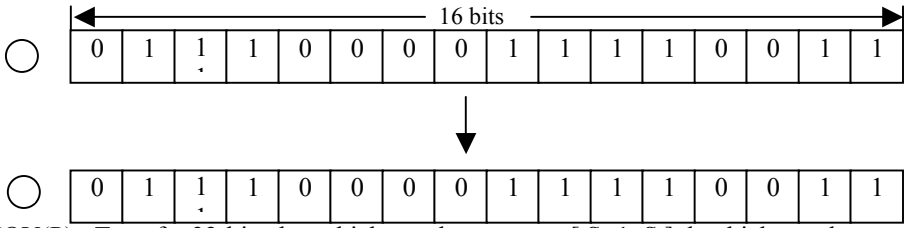
Nastawa operandu

○	Obiekt źródłowy gdzie zapamiętywane są dane przeznaczone do transferu
○	Obiekt przeznaczenia transferu

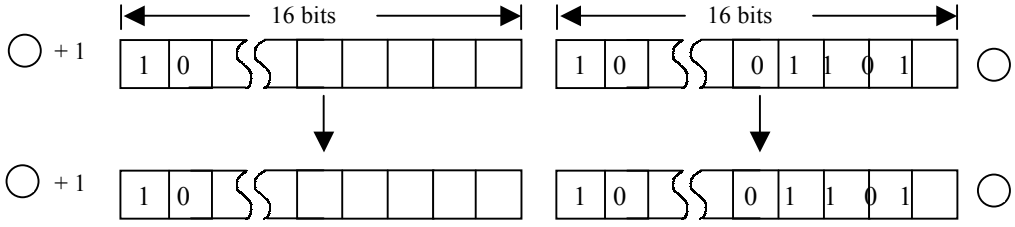
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje

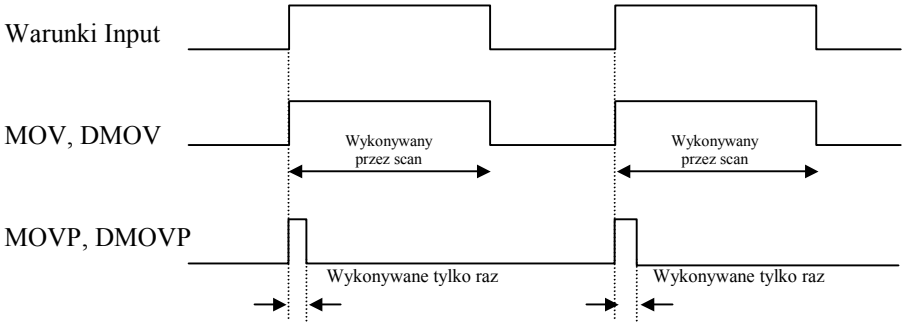
- MOV(P) : Transfer 16-bits data obiektu wskazanego w [S] do obiektu wskazanego w [D].



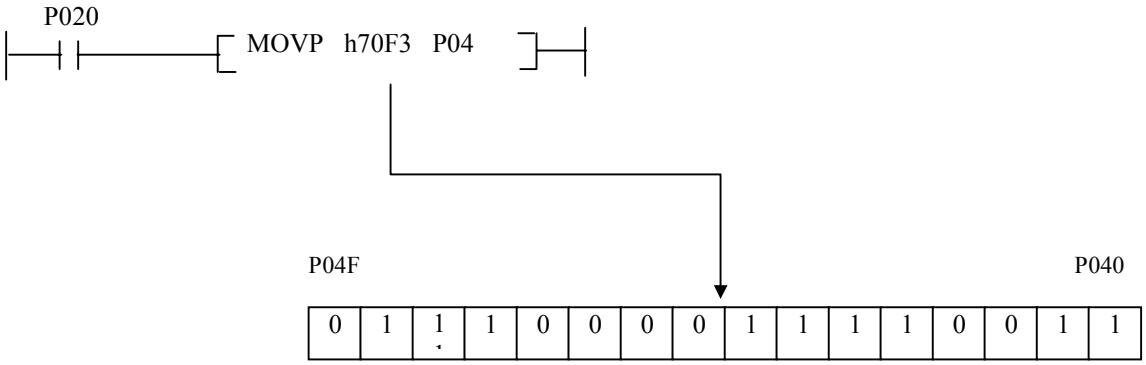
- DMOV(P) : Transfer 32-bits data obiektu wskazanego w [S+1, S] do obiektu wskazanego w [D+1, D].



- Warunki wykonywania



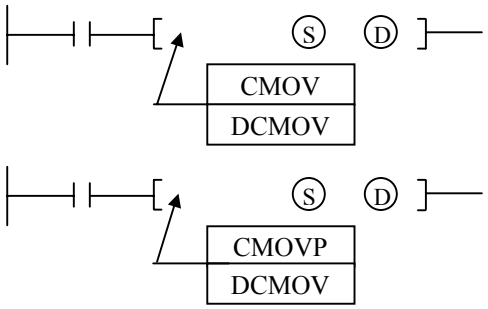
2) Przykład programu
Zawsze kiedy na P020 wykryte jest narastające zbocze, to ‘h00F3’ jest przeniesione do P04 word.



5.1.2 CMOV, CMOVP, DCMOV, DCMOVP

CMOV Complement move)	FUN(84) CMOV DCMOV FUN(85) CMOVP DCMOCP	FUN(86) FUN(87)	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CMOV(P)	Ⓢ	O	O	O	O*	O	O	O		O	O	O	5/7	O		
DCMOV(P)	Ⓓ	O	O	O	O*		O	O		O	O					



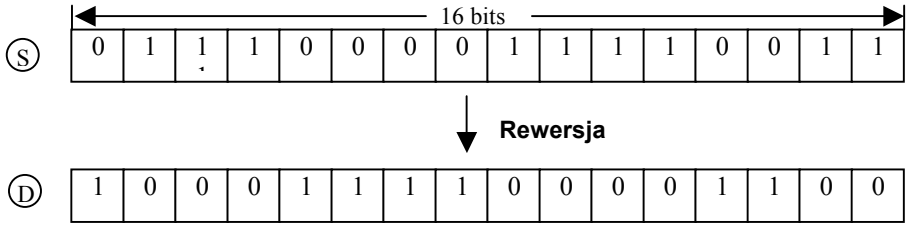
Nastawa operandu

Ⓢ	Obiekt źródłowy gdzie zapamiętywane są dane przeznaczone do transferu
Ⓓ	Obiekt przeznaczenia transferu

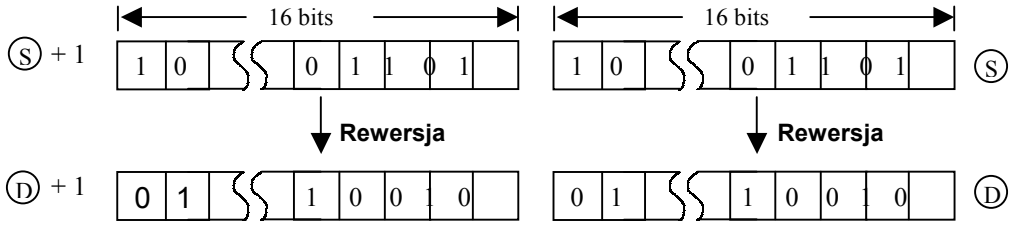
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje

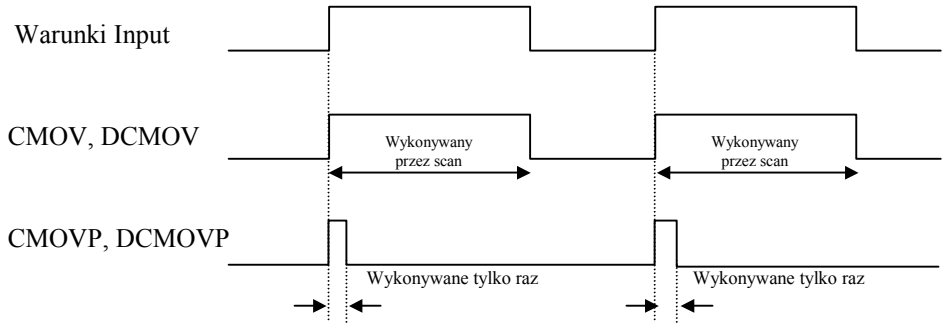
- CMOV(P) : Dokonuje bitowo rewersji 16-bits data z [S] i transferuje wynik do [D]



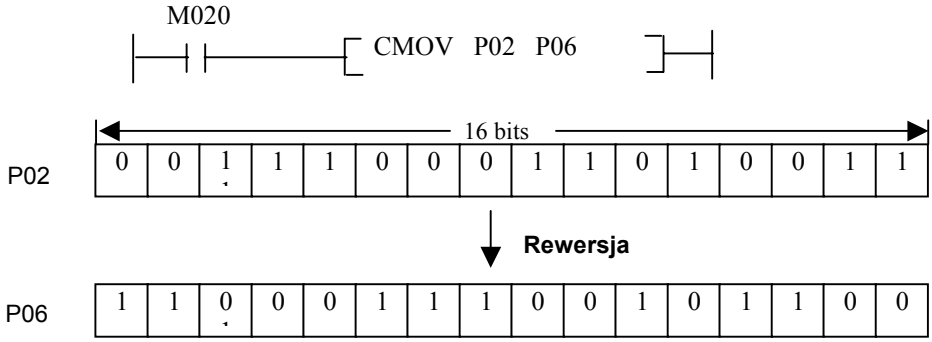
- DCMOV(P) : Dokonuje bitowo rewersji 32-bits data z [S+1, S] i transferuje wynik do [D+1,D].



- Warunki wykonywania



2) Przykład programu
- Gdy M020 jest ON, następuje rewersja danych P02 word i przeniesienie wyniku do P06 word.

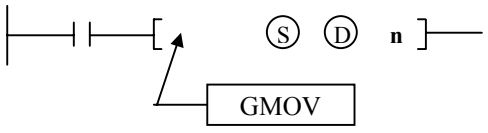


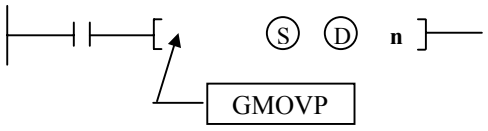
5.1.3 GMOV, GMOVP

GMOV (Group move)	FUN(90) GMOV FUN(91) GMOVP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
GMOV GMOVP	Ⓢ	O	O	O	O*	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				



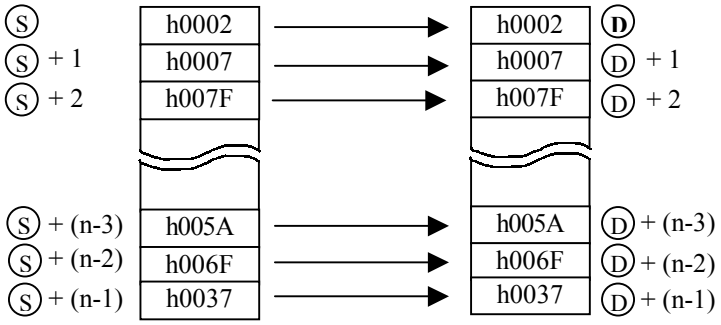


Nastawa operandu

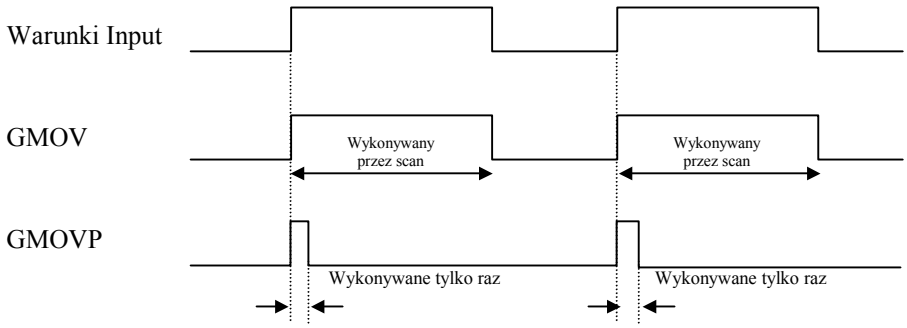
Ⓢ	Adres startu obszaru danych źródłowych
Ⓓ	Adres startu obszaru przeznaczenia przenoszonych danych
n	Liczba przenoszonych słów

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

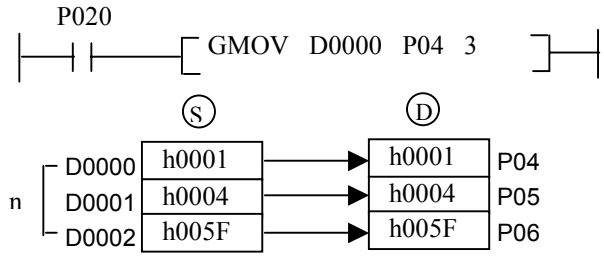
1) Funkcje
- Przenosi blokami zawartość ‘n’ words, poczynając od obiektu wyspecyfikowane-go w [S], do ‘n’ words poczynając od obiektu wyspecyfikowanego w [D].



- Warunki wykonywania



2) Przykładowy program
- Gdy P020 jest ON, przenieś dane z obszaru D000, D001, i D002 do obszaru P04, P05, i P06 .

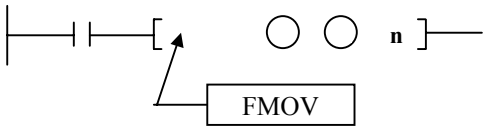


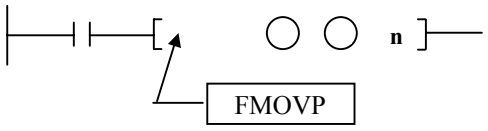
5.1.4 FMOV, FMOVP

FMOV (File move)	FUN(92) FMOV FUN(93) FMOVP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
FMOV FMOVP	◯	O	O	O	O*	O	O	O		O	O		7	O		
	◯	O	O	O	O*		O	O		O	O					
	n									O		O				

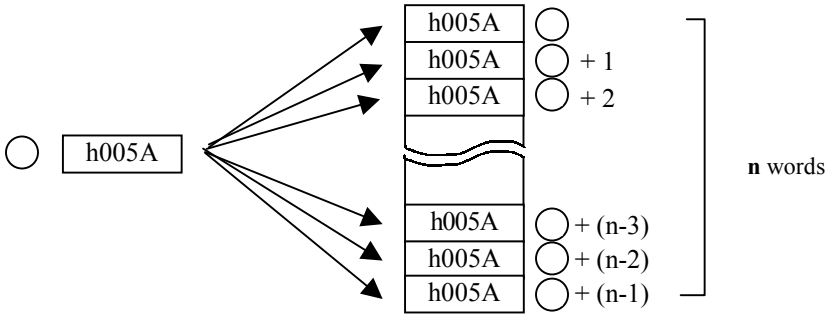




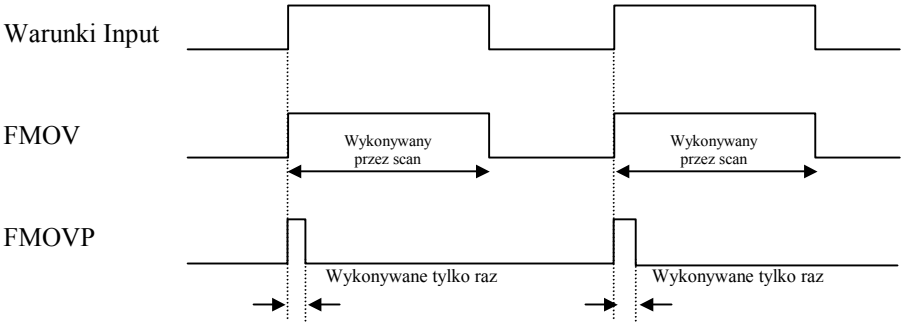
Nastawa operandu	
◯	Obiekt w którym zapamiętywany jest obszar danych źródłowych
◯	Adres startu obszaru przeznaczenia przenoszonych danych
n	Liczba przenoszonych słów

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

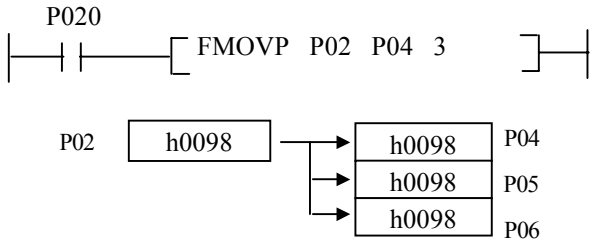
1) Functions
- Przenosi blokami zawartość obiektu wskazanego w [S] do ‘n’ punktów, które rozpoczynają się w obiekcie wskazanym w [D].



- Warunki wykonywania



2) Przykład programu
- Gdy na P030 pojawi się narastające zbocze, nastąpi transfer zawartości
- P02 word do bloku P04, P05 i P06.

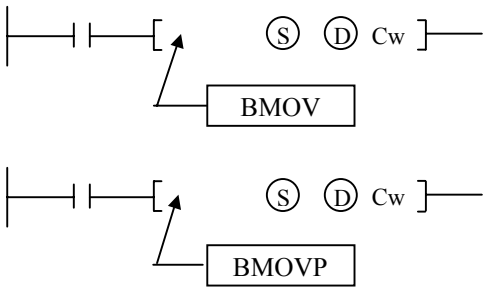


5.1.5 BMOV, BMOVP

BMOV (Bit move)	FUN(100) BMOV FUN(101) BMOVP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BMOV BMOVP	Ⓢ	O	O	O	O*	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	Cw											O				



Nastawa operandu

Ⓢ	Obiekt w którym zapamiętywane są dane źródłowe
Ⓓ	Obiekt który zapamięta przenoszone dane
Cw	Informacja o bicie startu i liczbie przenoszonych bitów

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

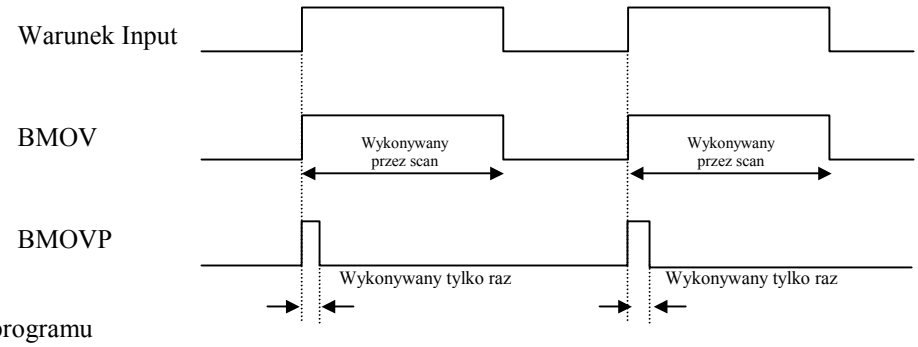
1) Funkcje
- Format ‘Cw’

h	s	d	z	z

a) s : Bit startu w [S].
b) d : Bit startu w [D]
c) zz : Liczba przenoszonych bitów. (Hexadecimal)

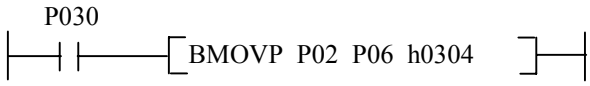
- Przenosi zawartość ‘zz’ bitów z ‘s’ bitów obiektu wskazanego w [S] do ‘zz’ bitów z ‘d’ bitów obiektu wskazanego w [D].
- Wartość maksymalna ‘zz’ jest h10 (= 16). Jeżeli wartość ‘zz’ jest 0 lub ponad h10, instrukcjazostanie zignorowana. (Error flag F110 jest ustawiany gdy ‘zz’ wynosi ponad h10.)

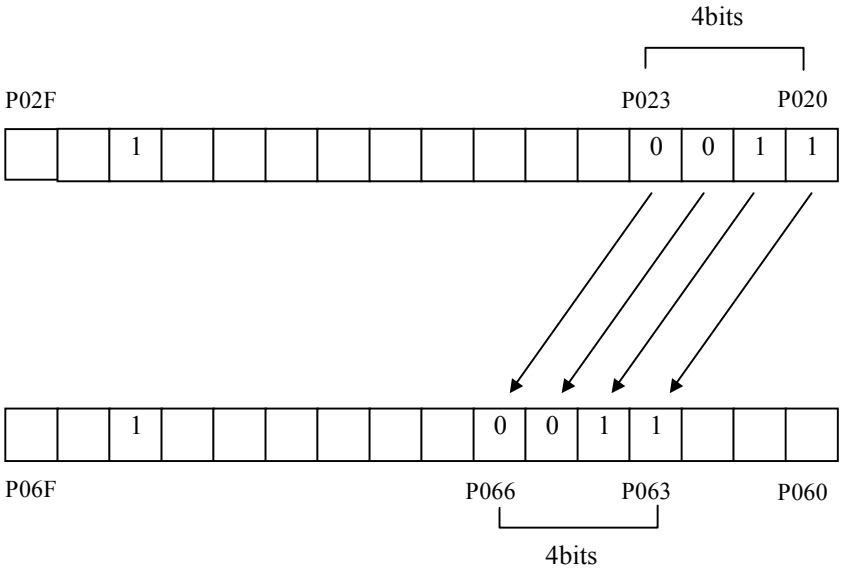
- Warunki wykonywania



2) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P030, 4 bity z P020 zostaną przeniesione do P063.





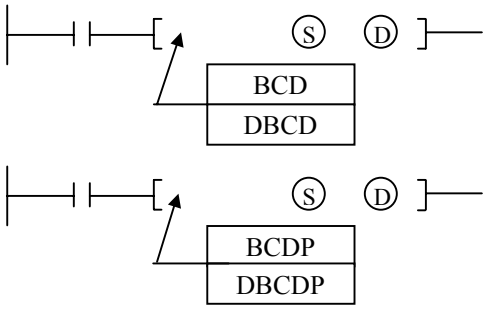
5.2 Instrukcje Conversion

5.2.1 BCD, BCDP, DBCD, DBCDP

BCD (Binary coded decimal)	FUN(60) BCD FUN(62) DBCD FUN(61) BCDP FUN(63) DBCDP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BCD(P)	Ⓢ	O	O	O	O*	O	O	O		O	O		5	O		
DBCD(P)	Ⓓ	O	O	O	O*		O	O		O	O					



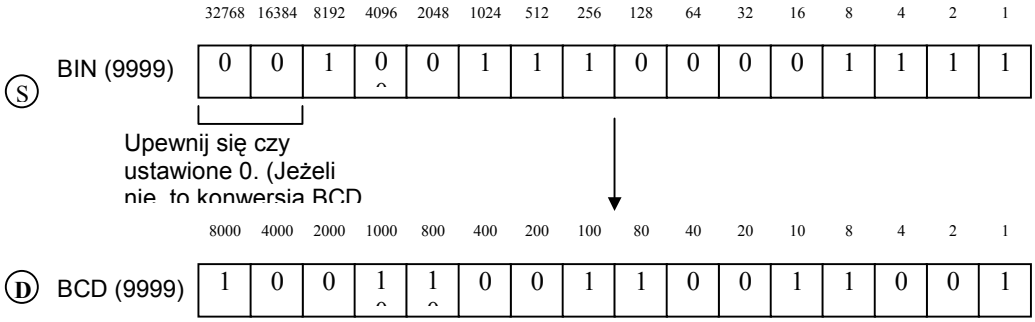
Nastawa operandu

Ⓢ	Obiekt źródłowy gdzie zapamiętywane są dane przeznaczone do konwersji na format BCD
Ⓓ	Obiekt przeznaczenia wyniku konwersji

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

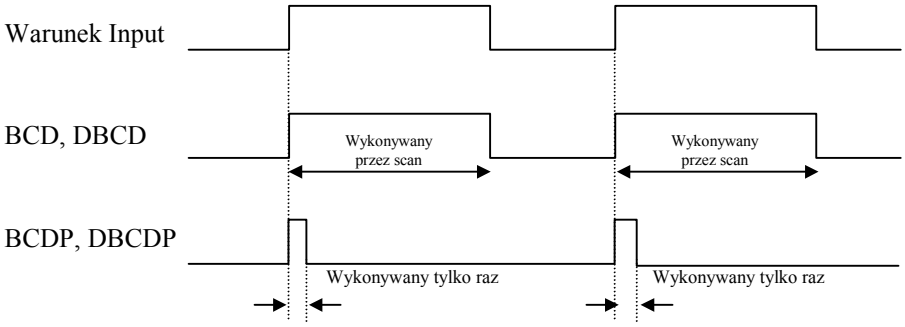
1) Functions

- BCD : Konwersja binary data (0 do 9999) obiektu wskazanego w [S] na BCD format i transfer wyniku do obiektu wskazanego w [D].



- DBCD : Konwersja binary data (0 do 99999999) obiektu wskazanego w [S] na BCD format i transfer wyniku do obiektu wskazanego w [D].

- Warunki wykonywania



- Błąd operacji

Błąd operacji oraz error flag (F110) ustawi się na ON, w następujących przypadkach.

a) Gdy używana jest instrukcja BCD(P)

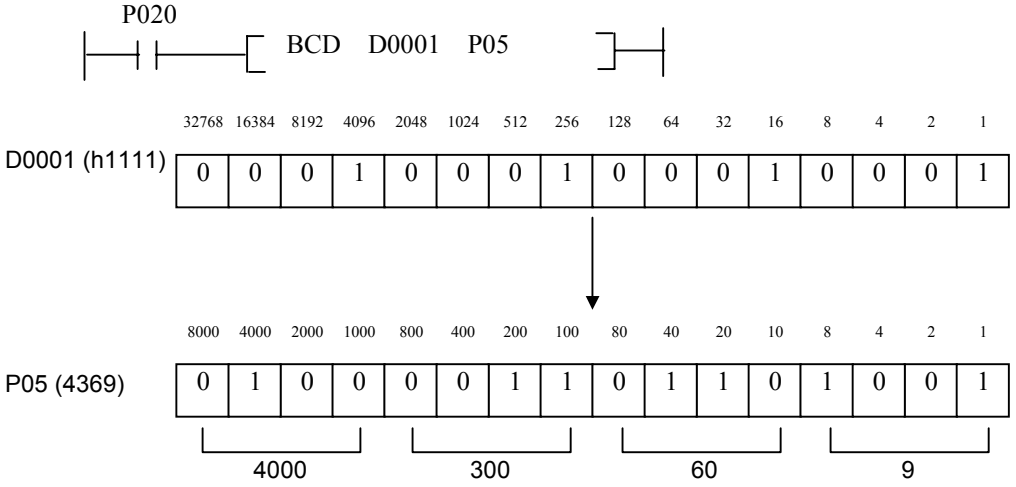
Dane źródłowe [S] znajdują się poza zakresem 0 do 9999

b) Gdy używana jest instrukcja BCD(P)

Dane źródłowe [S] znajdują się poza zakresem 0 do 99999999

2) Przykład programu

- Gdy P020 jest ON, binary data D001 poddawane są konwersji i wynik jest przenoszony do word P05.

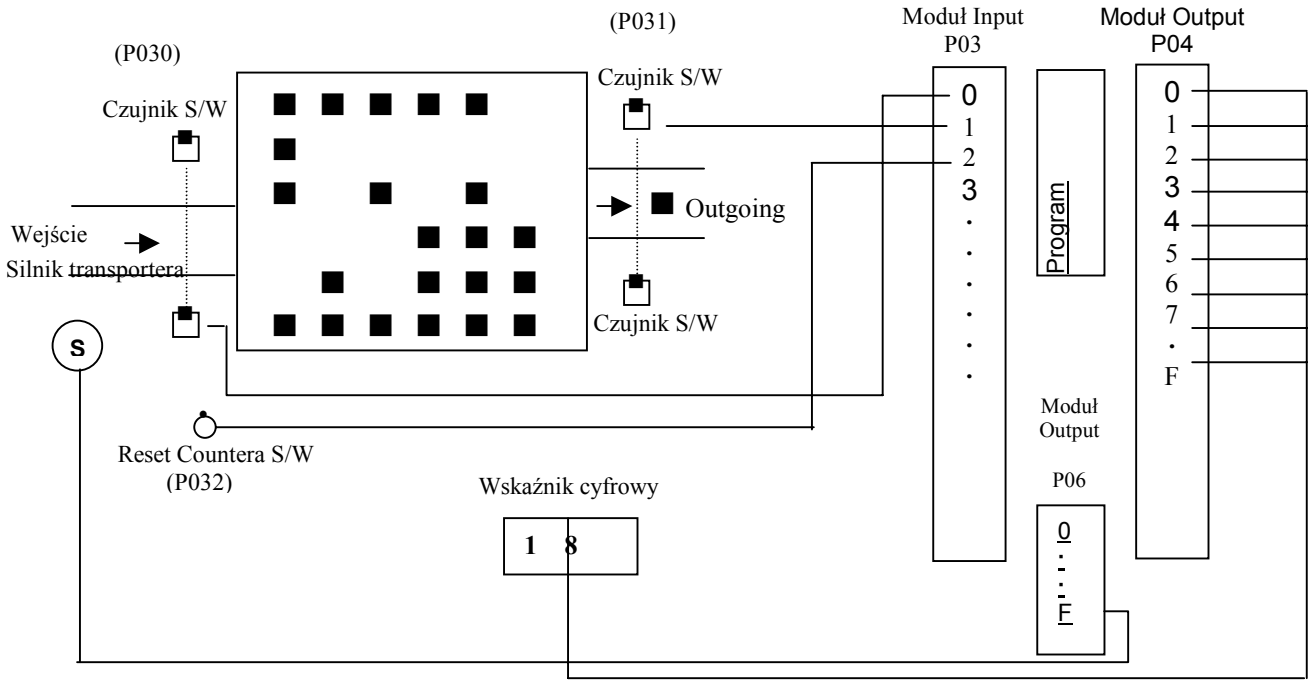


Wyświetlanie wartości bieżącej licznika (przykład instrukcji BCD i BMOV)

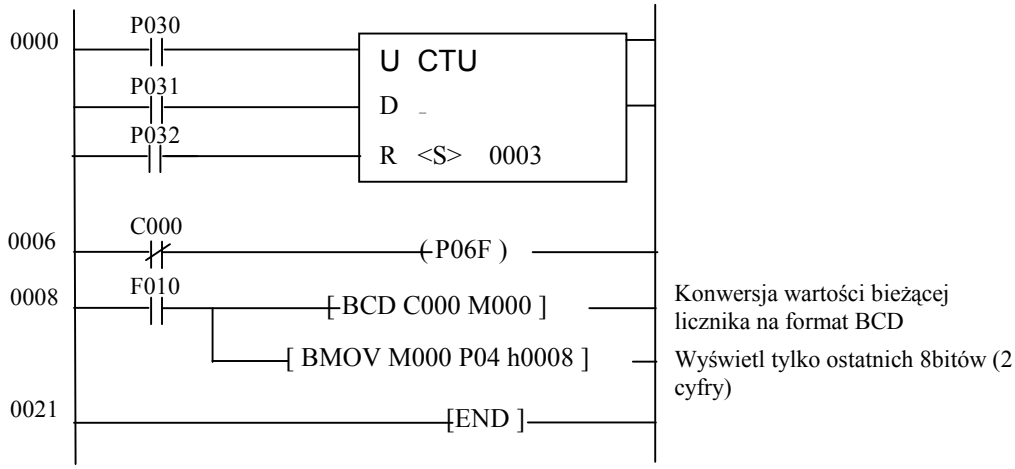
1. Praca

Do magazynu wchodzą i wychodzą produkty które są liczone czujnikami świetlnymi. Aktualny stan jest pokazywany na wyświetlaczu cyfrowym. Gdy stan magazynu osiągnie wartość 30, transporter zostaje zatrzymany.

2. Sruktura systemu



3. Program

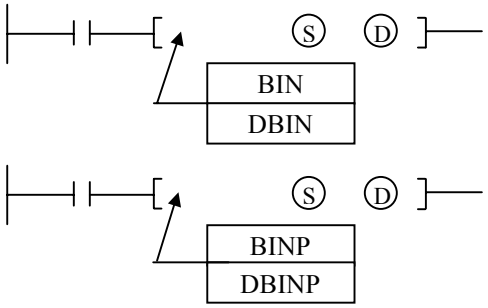


5.2.2 BIN, BINP, DBIN, DBINP

BIN (Binary)	FUN(64) BIN FUN(66) DBIN FUN(65) BINP FUN(67) DBINP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BIN(P)	Ⓢ	O	O	O	O*	O	O	O		O	O		5	O		
DBIN(P)	Ⓓ	O	O	O	O*		O	O		O	O					

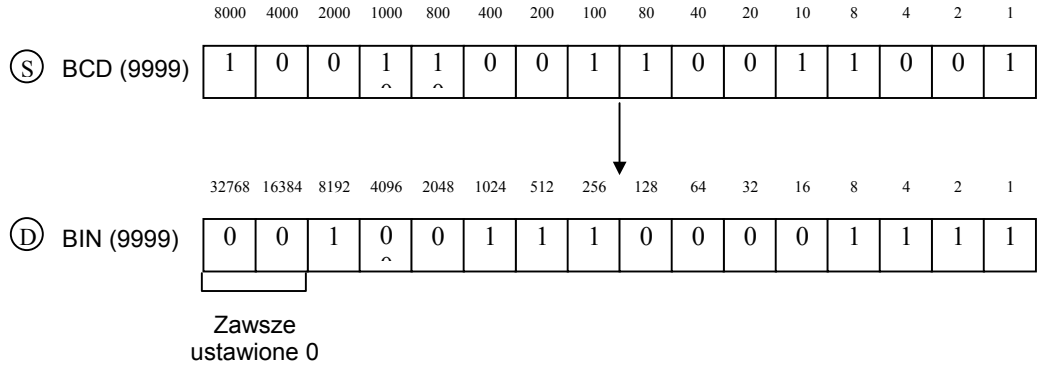


Nastawa operandu

Ⓢ	Obiekt źródłowy gdzie zapamiętywane są dane przeznaczone do konwersji na format BCD
Ⓓ	Obiekt przeznaczenia wyniku konwersji

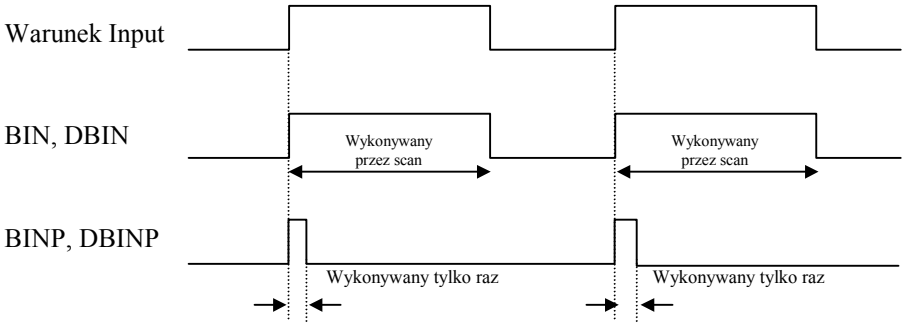
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - BIN : Dokonuje konwersji BCD data (0 do 9999) obiektu wskazanego w [S] do formatu binarnego i przeniesienie wyniku do obiektu wskazanego w [D].



   - DBIN : Dokonuje konwersji BCD data (0 do 99999999) obiektu wskazanego w [S] do formatu binarnego i przeniesienie wyniku do obiektu wskazanego w [D].

- Warunki wykonywania



- Błąd operacji

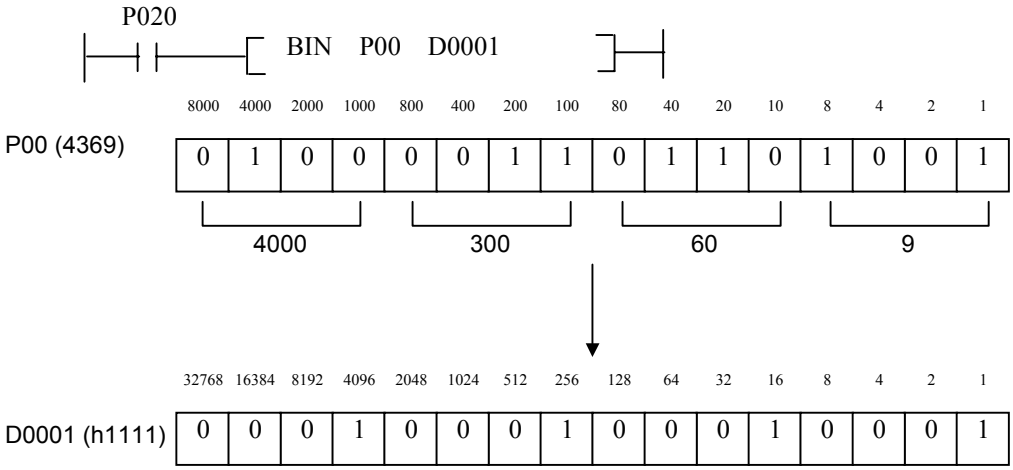
Błąd operacji oraz error flag (F110) ustawi się na ON, w następujących przypadkach.

a) Każda cyfra (4 bity) obiektu [S] znajduje się poza zakresem 0 do 9

(Przykład : [S] = h78A5)

2) Przykład programu

- Gdy P020 jest ON, następuje konwersja BCD data word P00 i przeniesienie wyniku do D0001.



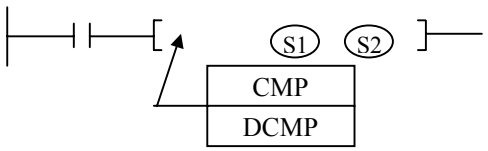
5.3 Comparison instructions

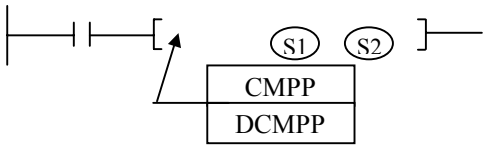
5.3.1 CMP, CMPP, DCMP, DCMPP

CMP (Compare)	FUN(50) CMP FUN(52) DCMP FUN(51) CMPP FUN(53) DCMPP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CMP(P)	S1	O	O	O	O	O	O	O		O	O	O	5 / 9	O		
DCMP(P)	S2	O	O	O	O	O	O	O		O	O	O				





Nastawa operandu

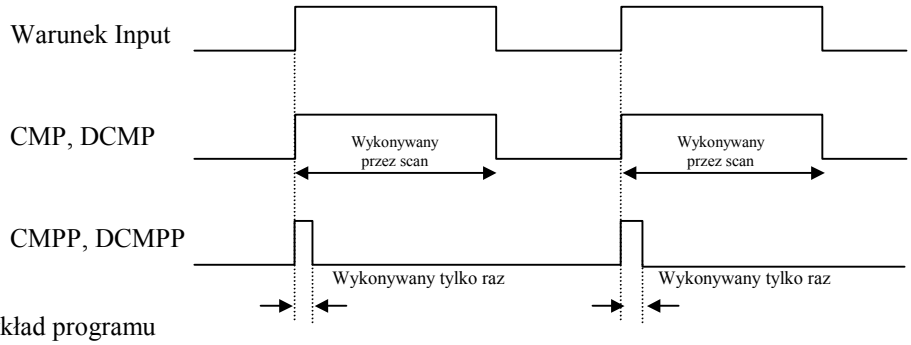
S1 S2	Dane do porównania

1) Funkcje
- Porównuje zawartość dwóch obiektów wskazanych w [S1] i [S2].
- Po dokonaniu porównania, ustawia odpowiednią flagę z zakresu pomiędzy F120 ~ F125.

Flaga	F120	F121	F122	F123	F124	F125
	<	≤	=	>	≥	≠
S1 > S2	0	0	0	1	1	1
S1 < S2	1	1	0	0	0	1
S1 = S2	0	1	1	0	1	0

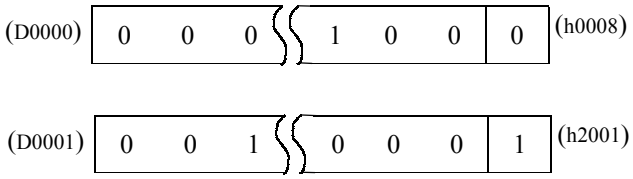
- Flagi powyżej wskazują na wyniki instrukcji CMP ostatnio wukonywanych.
- Flaga błędu (F110) jest ustawiana gdy [S1] lub [S2] wyspecyfikowane jako format #D jest poza zakresem obiektu. Instrukcja w której miał miejsce błąd nie jest wykonywana.

- Warunki wykonywania

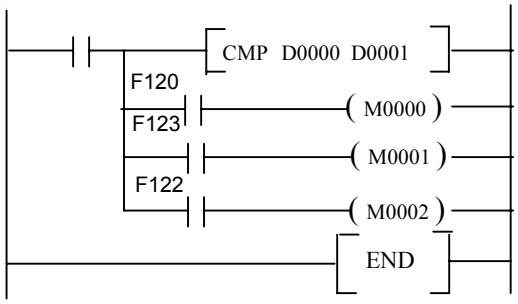


2) Przykład programu

- Gdy P020 jest ON, porównaj zawartość D000 i D001 następnie ustaw flagi w zależności od wyników.



[Program]



[Ustawianie flag]

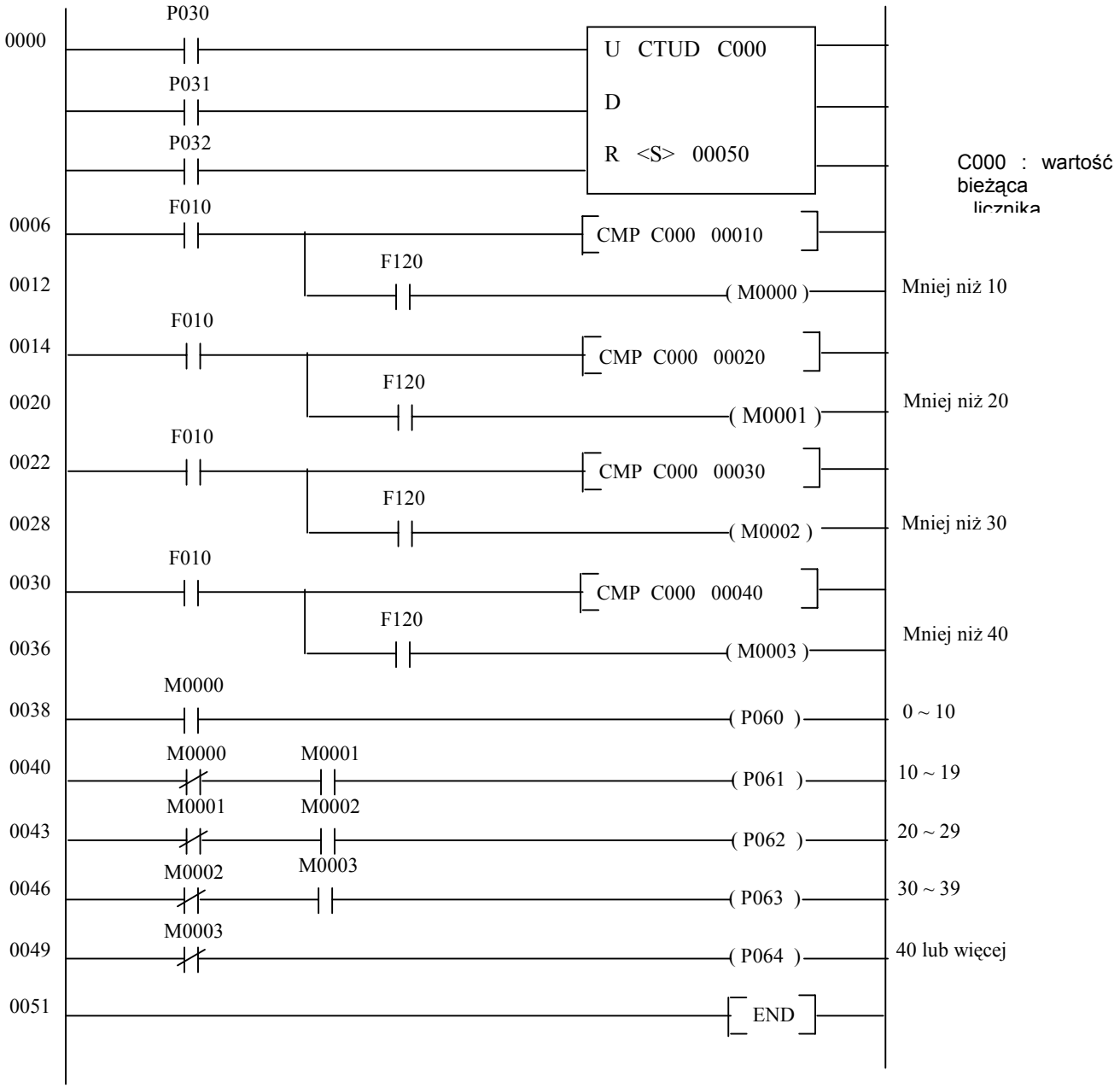
Flaga	F120	F121	F122	F123	F124	F125
	<	≤	=	>	≥	≠
Wynik	1	1	0	0	0	1

Układ porównania (Przykład instrukcji CMP)

1. Praca

Jest up-down counter C000. P030 jest up input a P031 jestis down input. Jeżeli wartość bieżąca timera jest 0 ~ 9, P060 ustawi się na ON. Jeżeli wartość bieżąca timera jest 10 ~ 19, P061 ustawi się na ON. P062 będzie ON, gdy 20 ~ 29, P063 będzie ON, gdy 30 ~ 39 i P064 będzie ON, gdy wartość bieżąca będzie 40 lub większa.

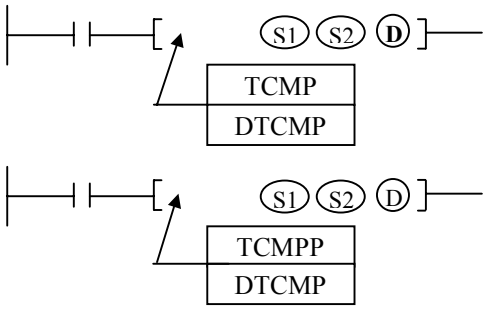
2. Program



5.3.2 TCMP, TCMPP, DTCMP, DTCMPP

TCMP (Table compare)	FUN(54) TCMP FUN(56) DTCMP FUN(55) TCMPP FUN(57) DTCMPP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
TCMP(P) DTCMP(P)	S1	O	O	O	O	O	O	O		O	O	O	7 / 9	O	O	
	S2	O	O	O	O	O	O	O		O	O					
	D	O	O	O	O*		O	O		O	O					

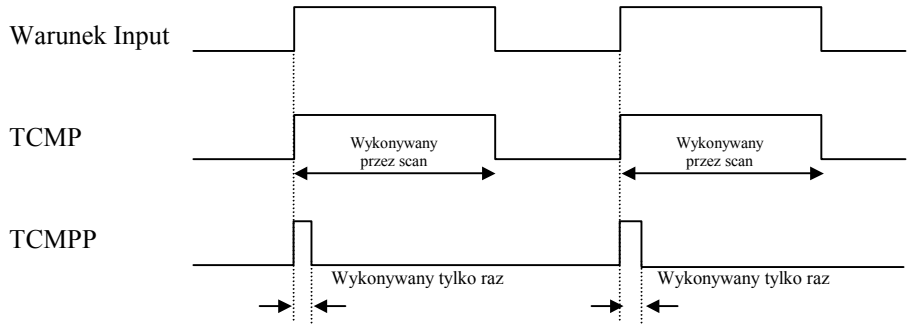


Nastawa operandu

S1	Dane do porównania
S2	Początek adresu bloku który ma być porównany z S1
D	Obiekt w którym ma być zapamiętany wynik porównania

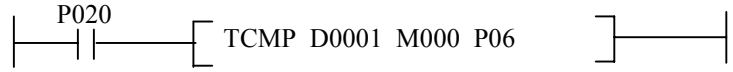
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

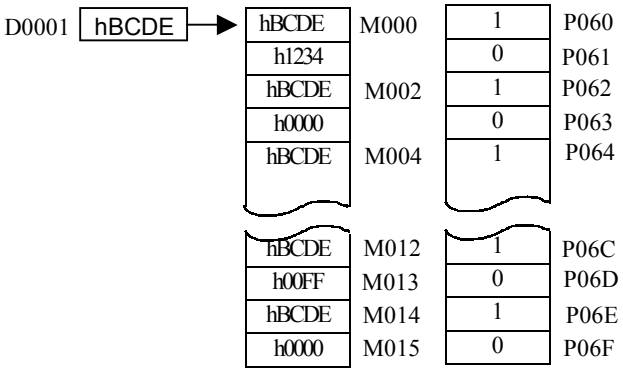
1) Funkcje
- Porównuje zawartość obiektu wskazanego w [S1] z każdą zawartością 16 words z obiektu wskazanego w [S2].
- Wynik porównania (Jeżeli dwa słowa są takie same, wystawiane jest 1. Jeżeli nie, wystawiane jest 0) stanowi 16 bits i są one zapamiętywane w obiekcie wskazanym w [D].
- Jeżeli wszystkie wyniki porównań są 0, to wtedy zapalana jest flaga zero flag (F111). ([D] = 0)
- Warunki wykonywania



2) Przykład programu

- Gdy P020 jest ON, porównaj zawartość D0001 z 16 słowami poczynając od M00 (M00 ~ M15) i wystaw wynik porównania w słowie P06 (P060 ~ P06F).



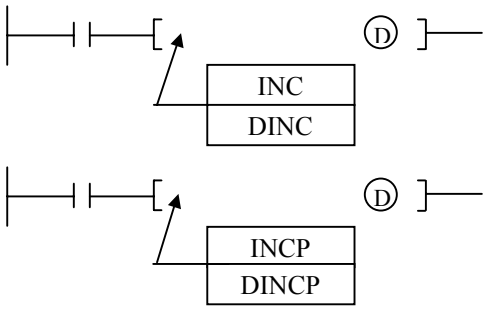


5.4 Operacje Increment/decrement

5.4.1 INC, INCP, DINC, DINCP

INC (Increment)	FUN(20) INC FUN(22) DINC FUN(21) INCP FUN(23) DINCP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
INC(P) DINC(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O	O	O

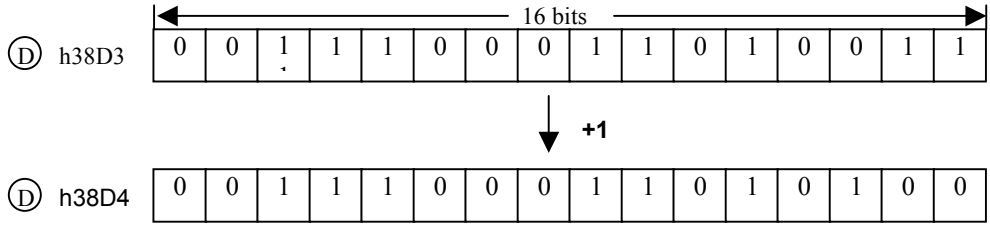


Nastawa operandu

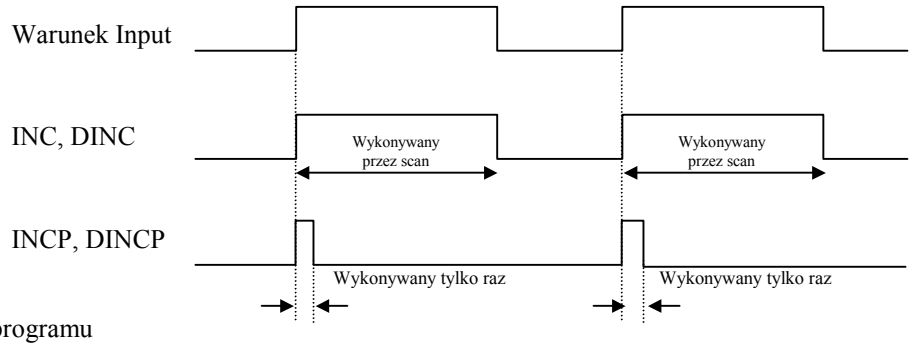
Ⓓ	Obiekt który zwiększy swoją wartość przez instrukcję INC.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- INC(P) : Wykonuje dodanie 1 do obiektu (16-bits data) wskazanego w [D].
- DINC(P) : Wykonuje dodanie 1 do obiektu (32-bits data) wskazanego w [D +1, D].
- Jeżeli INC(P) lub DINC(P) są wykonywane gdy zawartość obiektu jest hFFFF lub hFFFFFFFF, to zawartość obiektu po wykonaniu jest 0. Jednocześnie flaga zero (F111) i flaga carry (F112) są ustawione.
- Jeżeli obiekt wskazany przez #D jest poza zakresem, to pojawi sie błąd operacji oraz zostanie ustawiony error flag (F110).

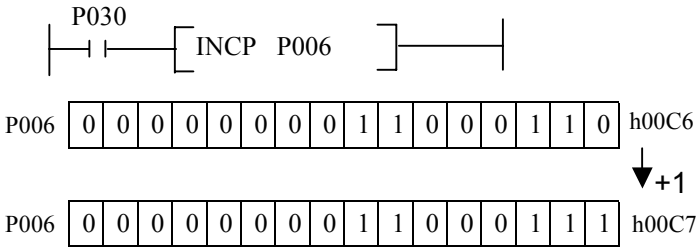


- Warunki wykonywania



2) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P030, to zawartość słówa P06 zostanie powiększona o 1.

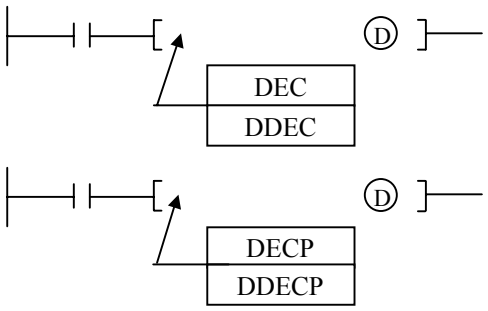


5.4.2 DEC, DECP, DDEC, DDECP

DEC (Decrement)	FUN(24) DEC FUN(26) DDEC FUN(25) DECP FUN(27) DDECP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DEC(P) DDEC(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O	O	O

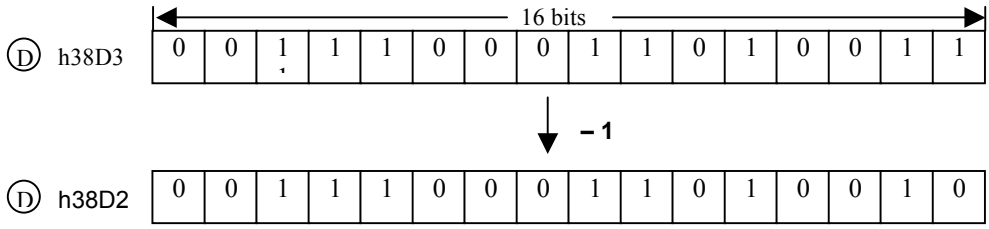


Nastawa operandu

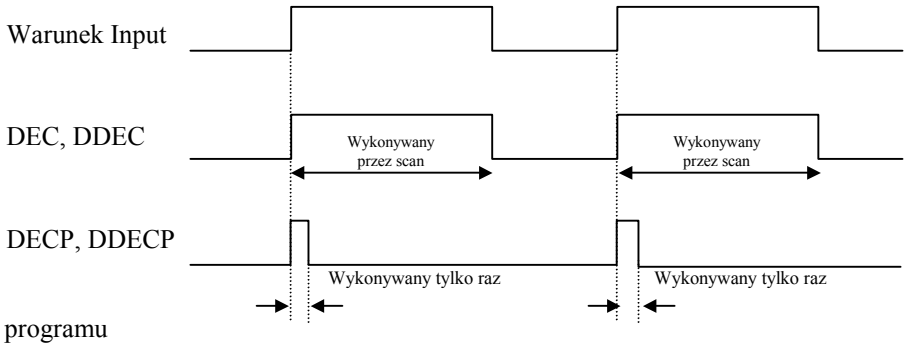
Ⓓ	Obiekt który zmniejszy swoją wartość przez instrukcję DEC.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

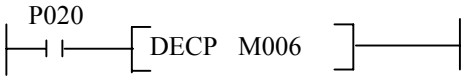
1) Funkcje
   - DEC(P) : Wykonuje odjęcie 1 od obiektu (16-bits data) wskazanego w [D].
   - DDEC(P) : Wykonuje odjęcie 1 od obiektu (32-bits data) wskazanego w [D +1, D].
   - Jeżeli DEC(P) lub DDEC(P) są wykonywane gdy zawartość obiektu jest 0, to zawartość obiektu po wykonaniu jest hFFFF lub hFFFFFFFF. Jednocześnie flaga carry (F112) zostaje ustawiona.
   - Jeżeli obiekt wskazany przez #D jest poza zakresem, to pojawi sie błąd operacji oraz zostanie ustawiony error flag (F110).

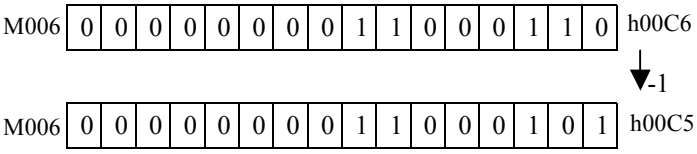


- Warunki wykonywania



2) Przykład programu
   - Zawsze kiedy zbocze narastające pojawi się na P020, to zawartość słowa M06 zmniejszy się o 1.



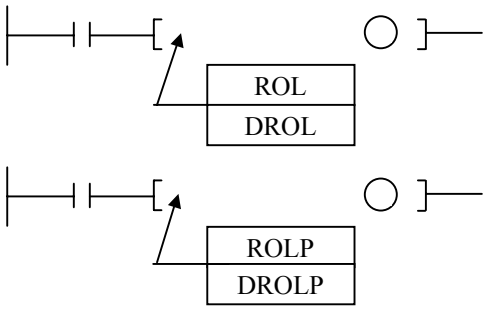


5.5 Instrukcje Rotacji

5.5.1 ROL, ROLP, DROL, DROLP

ROL (Rotuje w lewo)	FUN(30) ROL FUN(32) DROL FUN(31) ROLP FUN(33) DROLP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ROL(P) DROL(P)	◯	O	O	O	O*		O	O		O	O		3	O		O

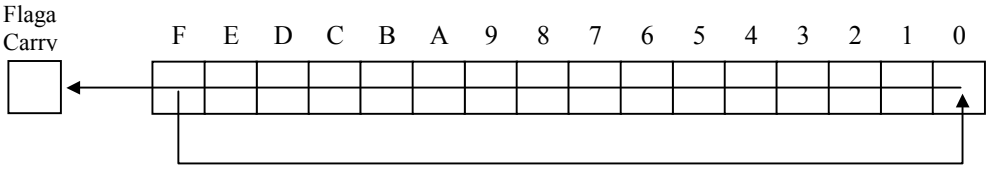


Nastawa operandu

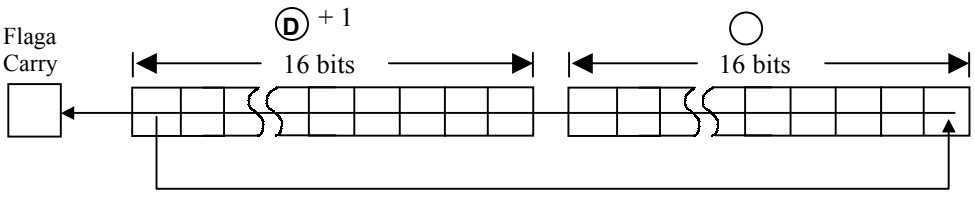
◯	Obiekt który ma być rotowany w lewo przez instrukcję ROL.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

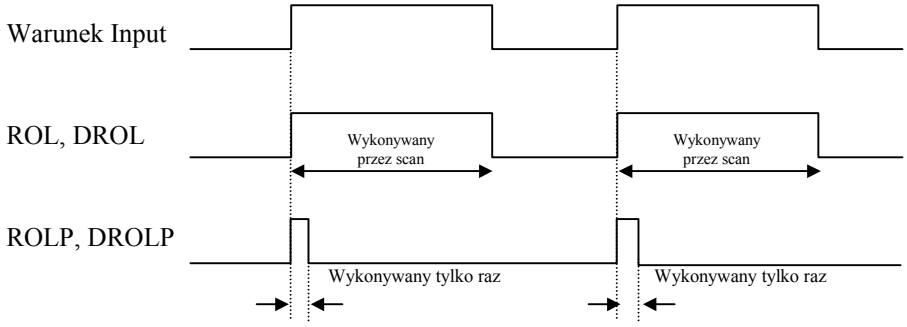
1) Funkcje
- ROL(P) : Dokonuje rotacji w lewo 16 bitów obiektu wskazanego w [D].
- MSB zostanie przeniesiony do LSB i do flagi carry (F112).



- DROL(P) : Dokonuje rotacji w lewo 32-bitów obiektu wskazanego w [D+1, D].
- MSB [D+1] zostanie przeniesiony do LSB [D] i do flagi carry (F112).

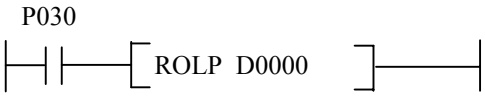


- Warunki wykonywania

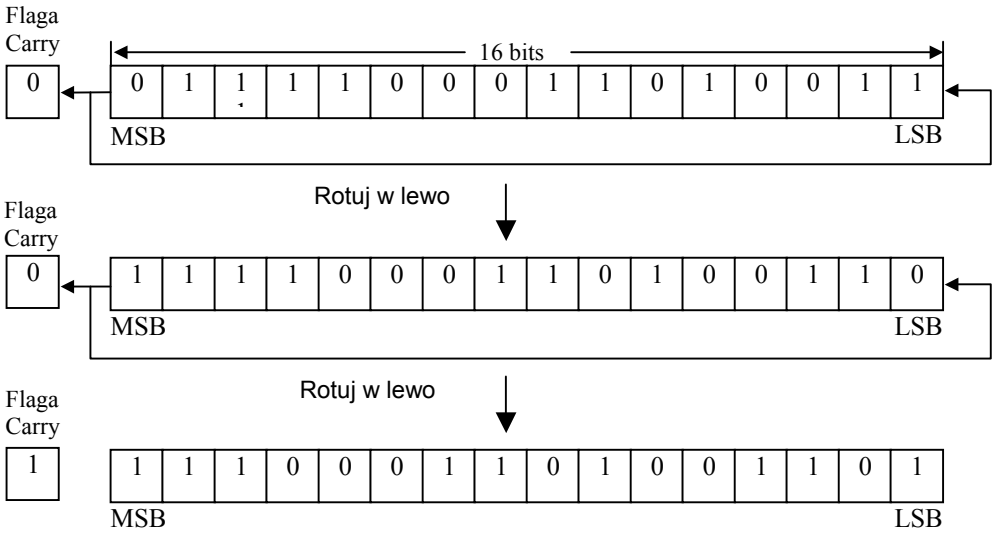


2) Program example

- Zawsze kiedy zbocze narastające pojawi się na P030, 16-bitów słowa D0000 zostanie przesuniętych w lewo.



D0000 = h78D3

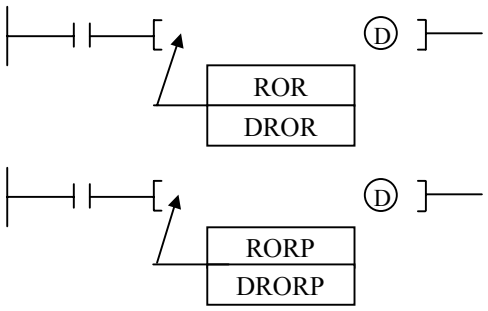


5.5.2 ROR, RORP, DROR, DRORP

ROR (Rotuje w prawo)	FUN(34) ROR FUN(36) DROR FUN(35) RORP FUN(37) DRORP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ROR(P) DROR(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O

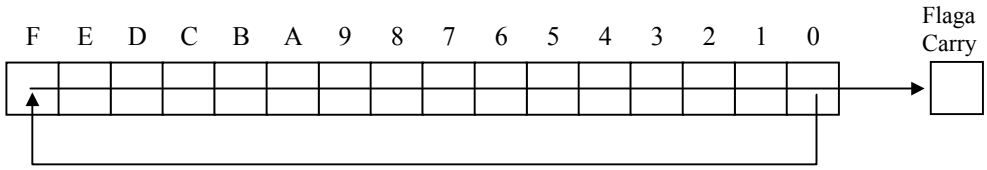


Nastawa operandu

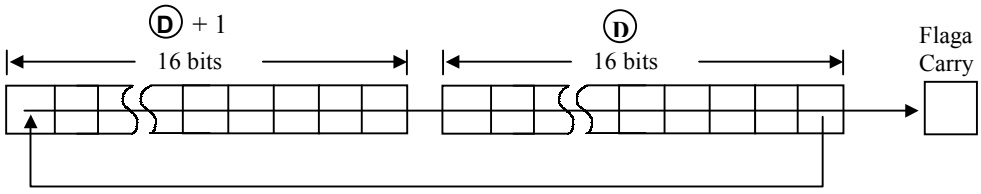
Ⓓ	Obiekt który ma być rotowany w prawo przez instrukcję ROR.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

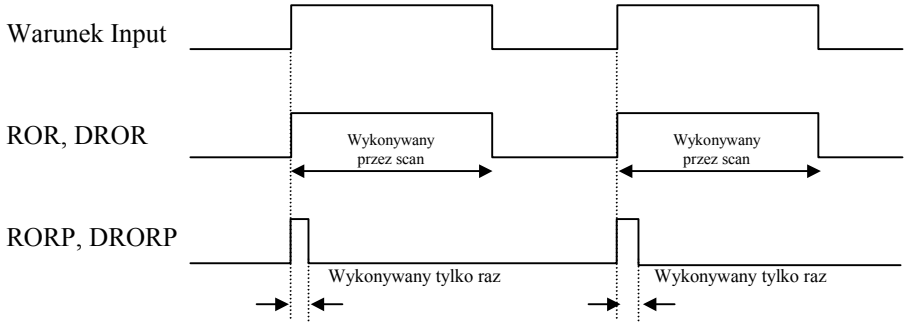
1) Funkcje
   - ROR(P) : Dokonuje rotacji w prawo 16 bitów obiektu wskazanego w [D].
   - LSB zostanie przeniesiony do MSB i do flagi carry (F112).



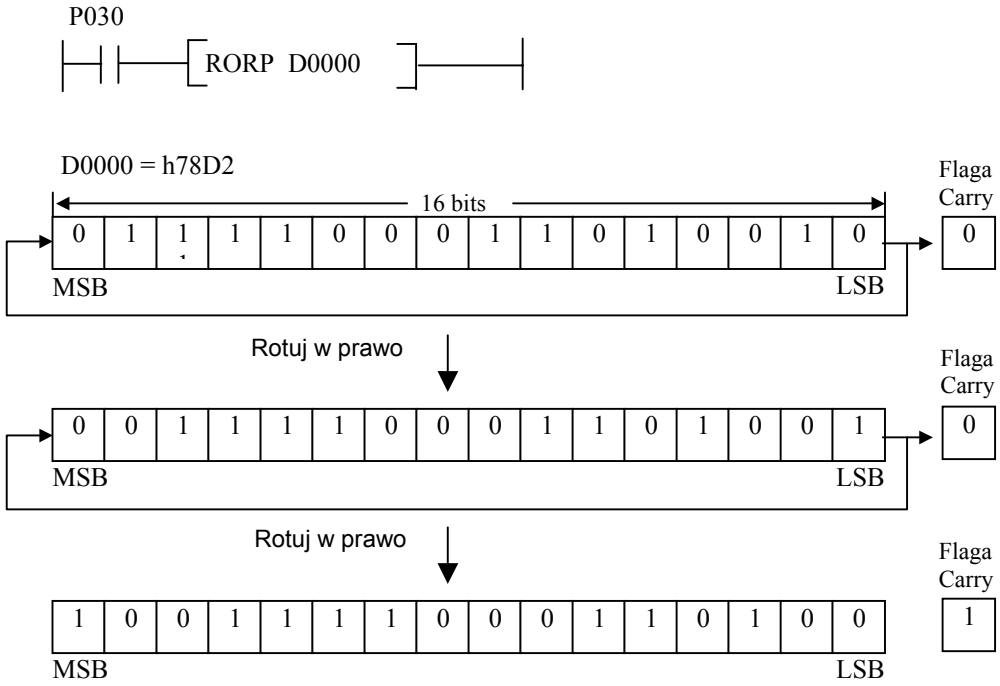
   - DROR(P) : Dokonuje rotacji w prawo 32 bitów obiektu wskazanego w [D+1, D].
   - LSB [D] zostanie przeniesiony do MSB [D+1] i do flagi carry (F112).



- Warunki wykonywania



2) Przykład programu
   - Zawsze kiedy zbocze narastające pojawi się na P030, 16-bitów słowa D0000 zostanie przesuniętych w prawo.

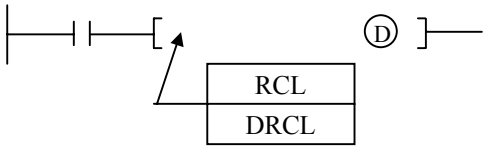


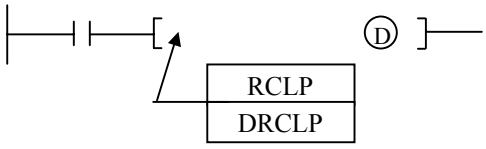
5.5.3 RCL, RCLP, DRCL, DRCLP

ROL (Rotuj w lewo łącznie z flagą)	FUN(40) RCL FUN(42) DRCL FUN(41) RCLP FUN(43) DRCLP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RCL(P) DRCL(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O



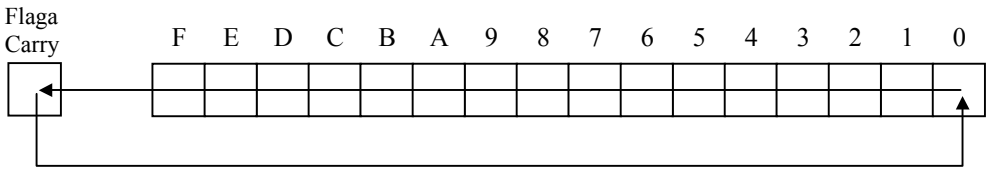


Nastawa operandu

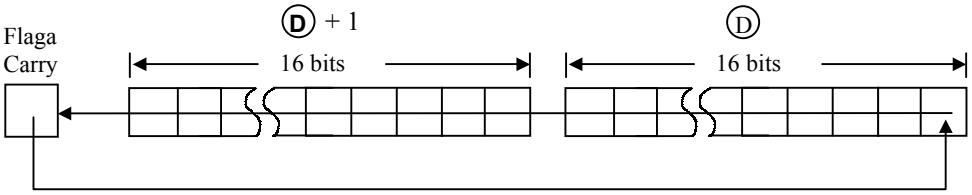
Ⓓ	Obiekt który ma być rotowany w lewo przez instrukcję RCL.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

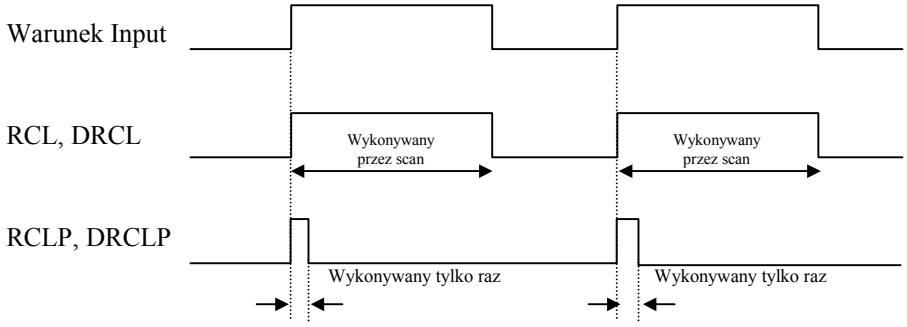
1) Funkcje
   - RCL(P) : Dokonuje rotacji w lewo 16 bitów obiektu wskazanego w [D] oraz flagi carry (F112) .
   - MSB zostanie przeniesiony do flagi carry (F112), a flaga carry do LSB.



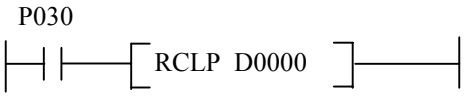
   - DRCL(P) : Dokonuje rotacji w lewo 32 bitów obiektu wskazanego w [D+1, D] oraz flagi carry (F112) .
   - MSB [D+1] zostanie przeniesiony do flagi carry (F112), a flaga carry do LSB [D].



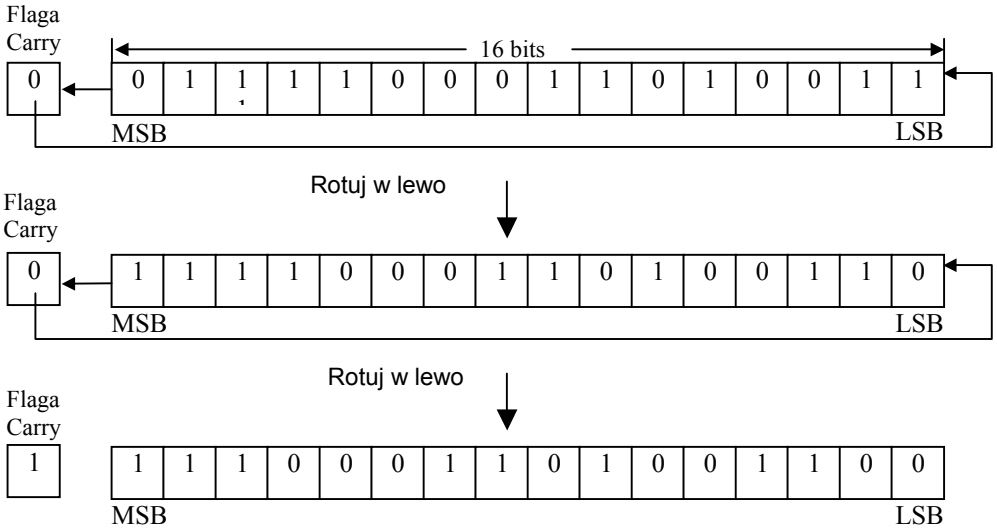
- Warunki wykonywania



2) Przykład programu
   - Zawsze kiedy zbocze narastające pojawi się na P030, 16-bitów słowa D0000 i flaga carry zostanie przesuniętych w lewo.



D0000 = h78D3

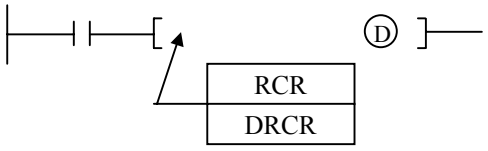


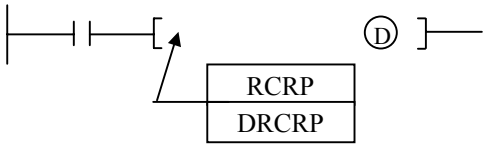
5.5.4 RCR, RCRP, DRCR, DRCRP

RCR (Rotuj w prawo łącznie z flagą)	FUN(44) RCR FUN(46) DRCR FUN(45) RCRP FUN(47) DRCRP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RCR(P) DRCR(P)	Ⓓ	O	O	O	O*		O	O		O	O		3	O		O



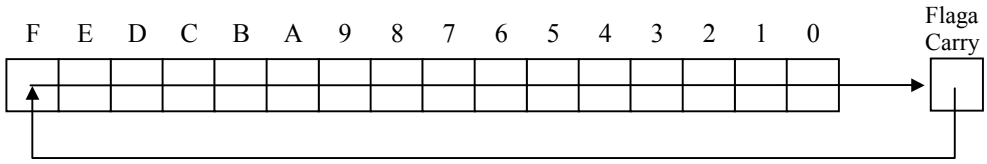


Nastawa operandu

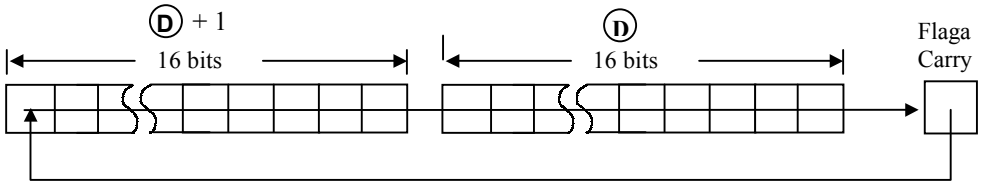
Ⓓ	Obiekt który ma być rotowany w prawo przez instrukcję RCR.

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

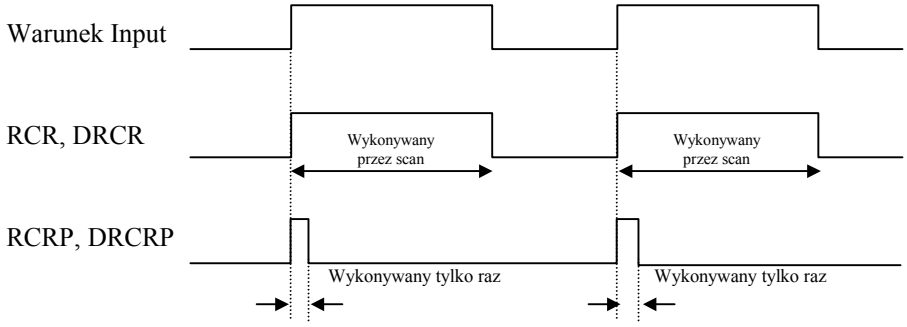
1) Funkcje
   - RCR(P) : Dokonuje rotacji w prawo 16 bitów obiektu wskazanego w [D] oraz flagi carry (F112) .
   - LSB zostanie przeniesiony do flagi carry (F112), a flaga carry do MSB.



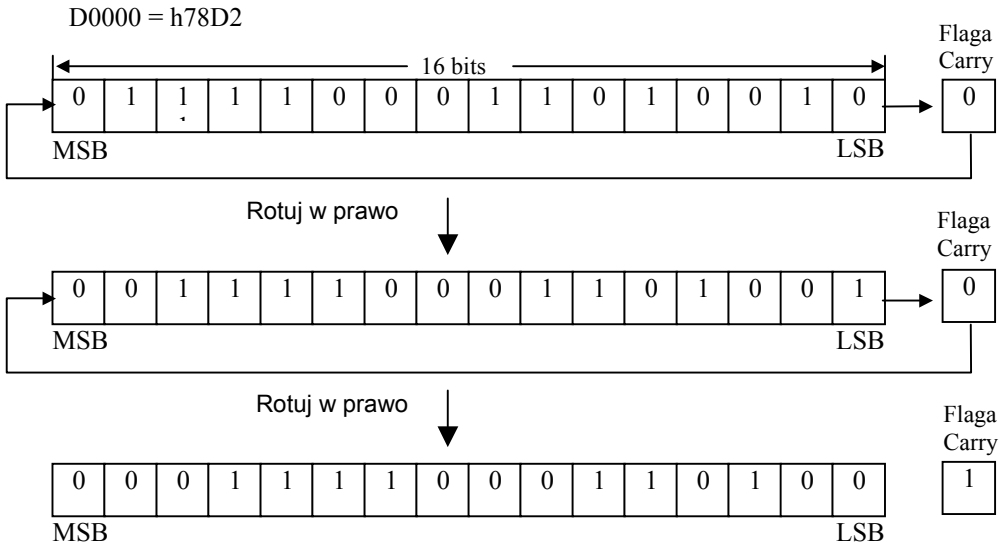
   - DRCR(P) : Dokonuje rotacji w prawo 32 bitów obiektu wskazanego w [D+1, D] oraz flagi carry (F112) .
   - LSB [D] zostanie przeniesiony do flagi carry (F112), a flaga carry do MSB [D+1].



- Warunki wykonywania



2) Przykład programu
   - Zawsze kiedy zbocze narastające pojawi się na P030, 16-bitów słowa D0000 i flaga carry zostanie przesuniętych w prawo.



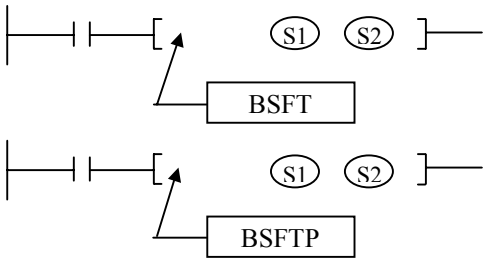
5.6 Instrukcje Shift

5.6.1 BSFT, BSFTP

BSFT (Bit shift)	FUN(74) BSFT FUN(75) BSFTP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BSFT(P)	S1	O	O	O	O*								5	O		
	S2	O	O	O	O*											

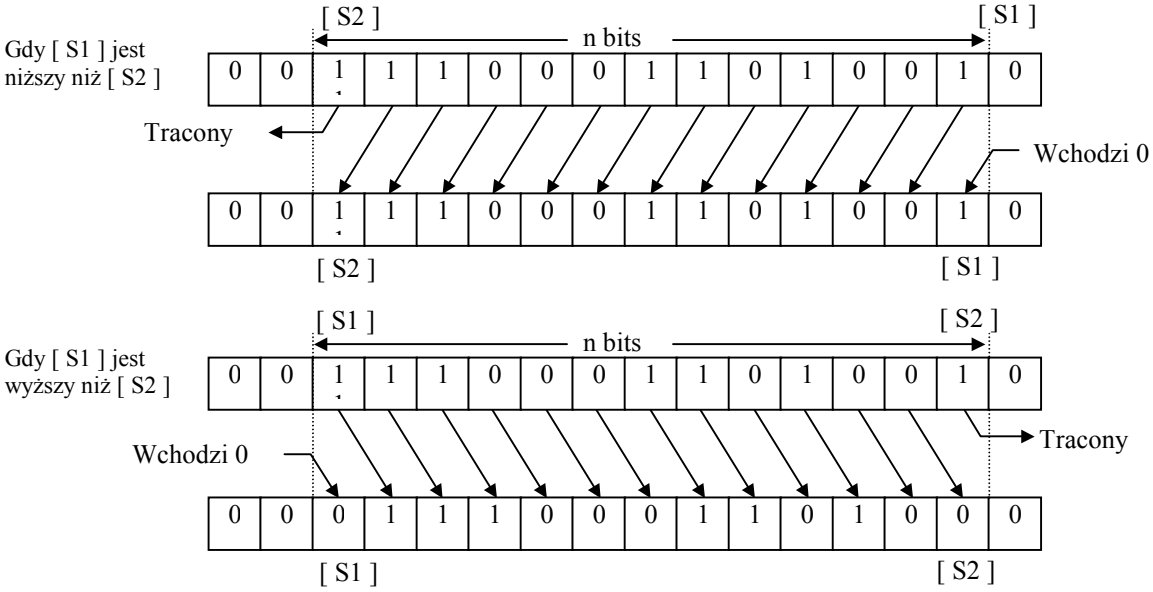


Nastawa operandu

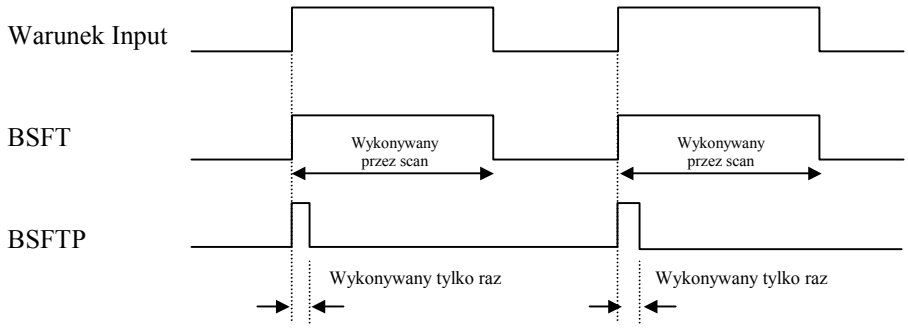
S1	Początkowy bit bloku do przesunięcia
S2	Końcowy bit bloku do przesunięcia

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- Przesuń blok wskazany jako [S1] ~ [S2] o 1 bit .
- Kierunek przesunięcia jest od [S1] do [S2]. Jednakże, jeżeli [S1] jest niższy niż [S2], to blok zostanie przesunięty w lewo. W przeciwnym przypadku blok jest przesuwany w prawo.

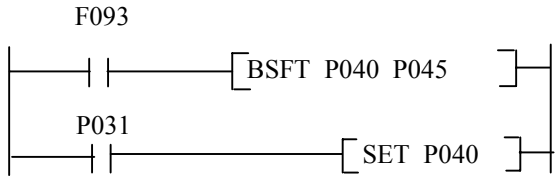


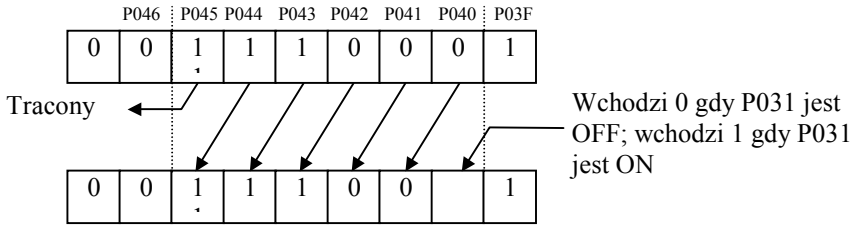
- Warunki wykonywania



3) Przykład programu

- Co sekundę, przesuwany jest w lewo blok z P040 do P045 o 1 bit. Jednosekundowy zegar flagi (F093) jest używany jako warunek input. P040 jest ustawione na 1 gdy P031 jest ON.



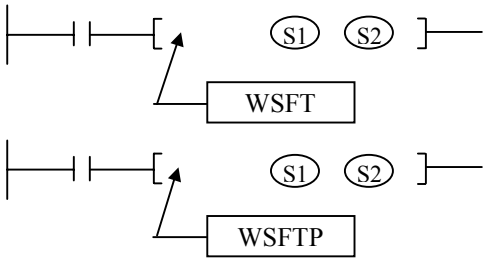


5.6.2 WSFT, WSFTP

WSFT (Word shift)	FUN(70) WSFT FUN(71) WSFTP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WSFT(P)	S1	O	O	O	O*		O	O		O	O		5	O		
	S2	O	O	O	O*		O	O		O	O					

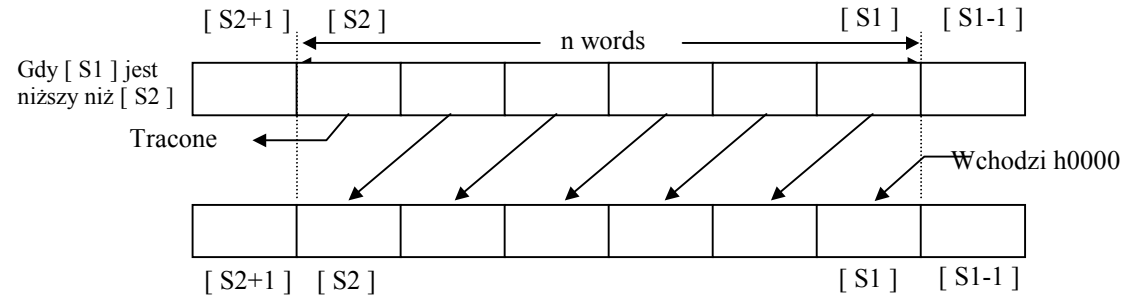


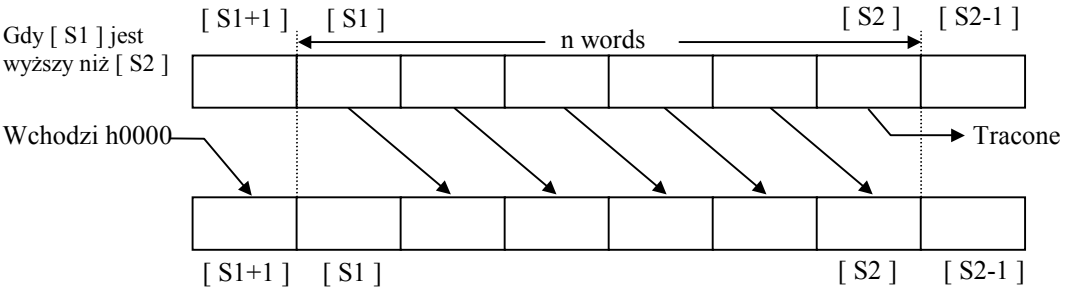
Nastawa operandu

S1	Początkowe słowo bloku do przesunięcia
S2	Końcowe słowo bloku do przesunięcia

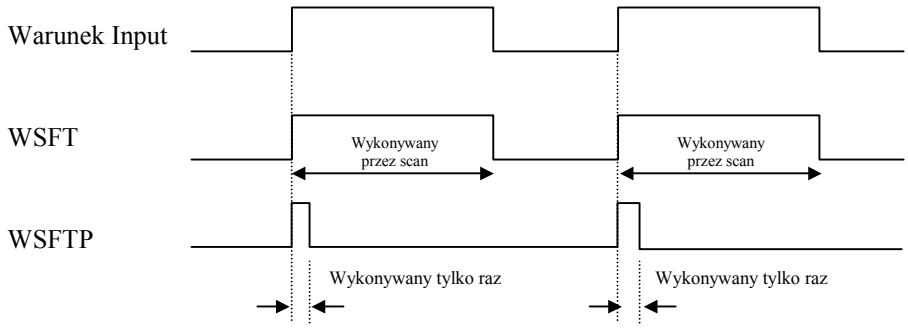
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - Przesuń blok wskazany jako [S1] ~ [S2] o 1 słowo.
   - Kierunek przesunięcia jest od [S1] do [S2]. Jednakże, jeżeli [S1] jest niższy niż [S2], to blok zostanie przesunięty w lewo. W przeciwnym przypadku blok jest przesuwany w prawo.



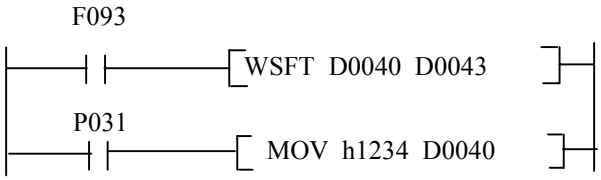


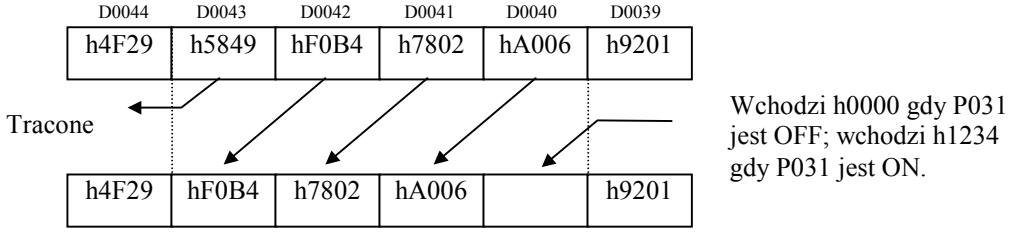
- Warunki wykonywania



4) Przykład programu

- Co sekundę, przesuwany jest w lewo blok z D0040 do D0043 o 1 słowo. Jednosekundowy zegar flagi (F093) jest używany jako warunek input. D0040 jest ustawione na h1234 gdy P031 jest ON.



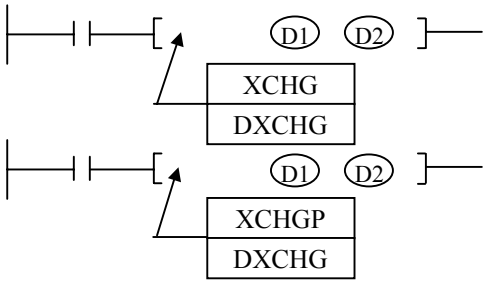


5.7 Instrukcje Exchange

5.7.1 XCHG, XCHGP, DXCHG, DXCHGP

XCHG (Word exchange)	FUN(102) XCHG FUN(104) DXCHG FUN(103) XCHGP FUN(105) DXCHGP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
XCHG(P)	(D1)	O	O	O	O*		O	O		O	O		5	O		
DXCHG(P)	(D2)	O	O	O	O*		O	O		O	O					



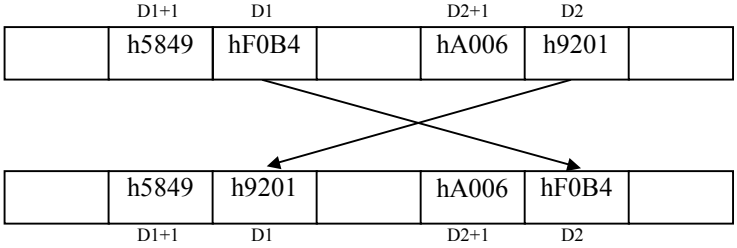
Nastawa operandu

(D1) (D2)	Dwa słowa których zawartość ma być zamieniona.

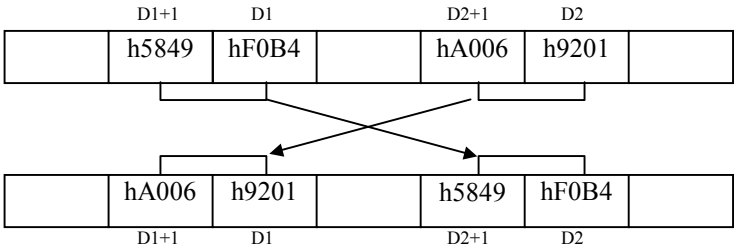
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje

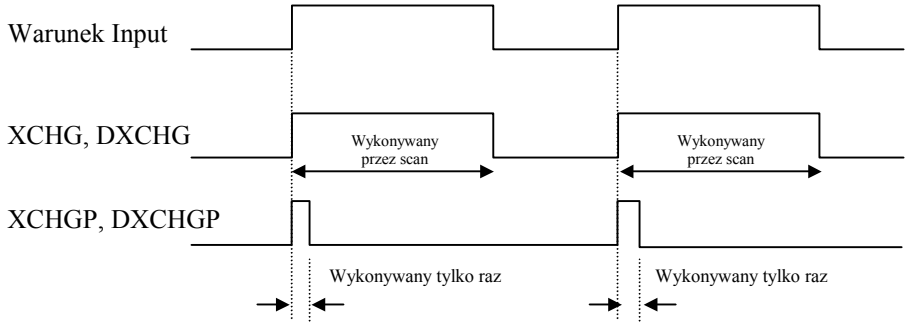
- XCHG(P) : Zamienia 16 bitową zawartość obiektów wskazanych w [D1] i [D2].



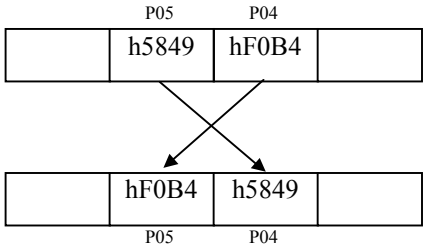
- DXCHG(P) : Zamienia 32 bitową zawartość dwóch obiektów wskazanych jako [D1+1, D1] i [D2+1, D2].



- Warunki wykonywania



5) Przykład programu
- Gdy P020 jest ON, wymień zawartość słów P04 i P05.



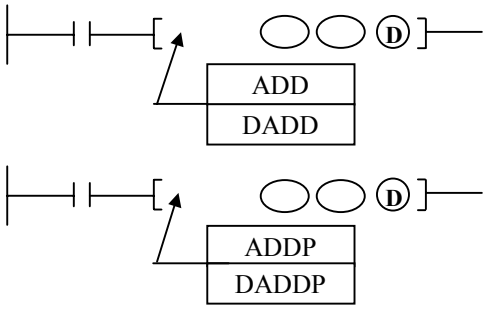
5.8 Instrukcje BIN arithmetic

5.8.1 ADD, ADDP, DADD, DADDP

ADD (Binary addition)	FUN(110) ADD FUN(112) DADD FUN(111) ADDP FUN(113) DADDP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ADD(P) DADD(P)	⬭	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	⬭	O	O	O	O	O	O	O		O	O	O				
	○	O	O	O	O*		O	O		O	O					



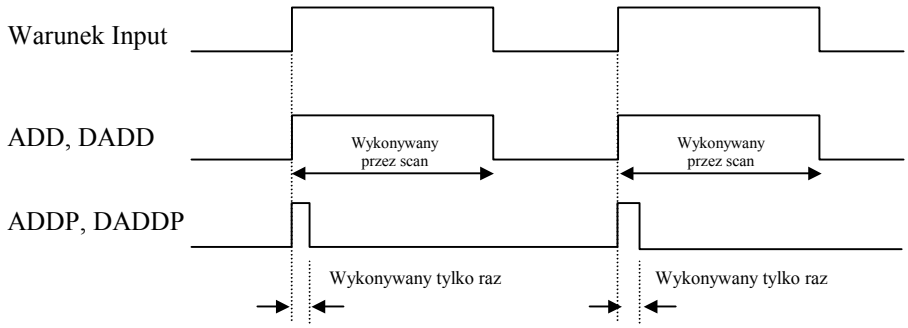
Nastawa operandu

⬭	Obiekt zapamiętujący pierwszy składnik
⬭	Obiekt zapamiętujący drugi składnik
○	Obiekt zapamiętujący wynik

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

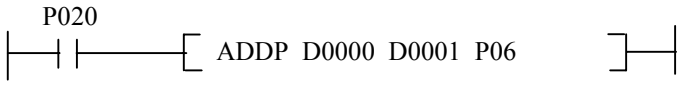
1) Funkcje
- ADD(P) : Wykonuje dodawanie binarne 16-bitowych danych wskazanych w [S1] i [S2]. Wynik dodawania jest zapamiętywany w obiekcie wskazanym w [D].
- DADD(P) : Wykonuje dodawanie binarne 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik dodawania jest zapamiętywany w obiekcie wskazanym w [D1+1, D1].
- Gdy wynik dodawania przekroczy hFFFF(ADD / ADDP) lub hFFFFFFFF(DADD / DADDP), to ustawi się flaga carry (F112).
- Gdy wynik dodawania jest 0, to ustawi się flaga zero.
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

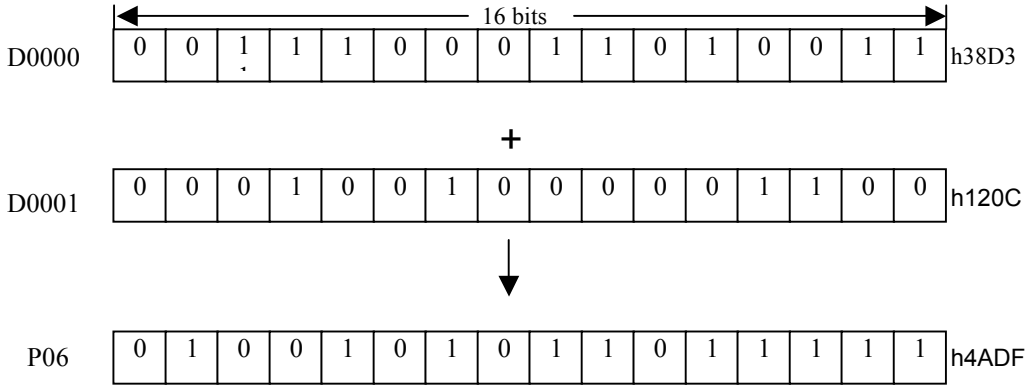
- Warunki wykonywania



6) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P020, dodaj zawartość D0000 i D0001 i zapamiętaj wynik dodawania w słowie P06.



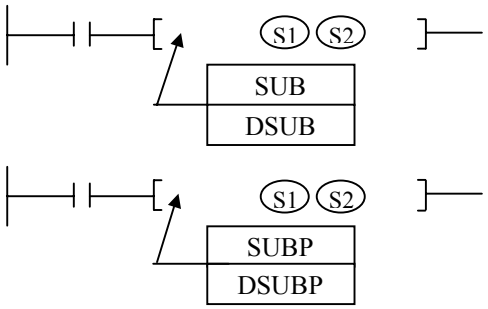


5.8.2 SUB, SUBP, DSUB, DSUBP

SUB (Binary subtraction)	FUN(114) SUB FUN(116) DSUB FUN(115) SUBP FUN(117) DSUBP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SUB(P) DSUB(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					



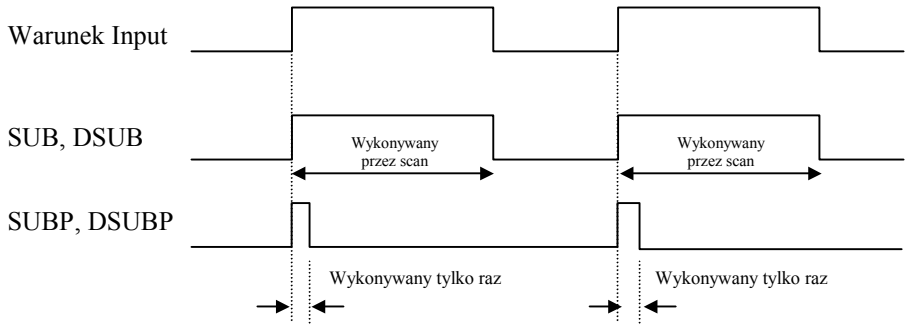
Nastawa operandu

S1	Obiekt zapamiętujący odjemną
S2	Obiekt zapamiętujący odjemnik
D	Obiekt zapamiętujący wynik

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

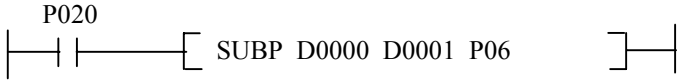
1) Funkcje
- SUB(P) : Wykonuje odejmowanie binarne 16-bitowych danych wskazanych w [S1] i [S2]. Wynik odejmowania jest zapamiętywany w obiekcie wskazanym w [D].
- DSUB(P) : Wykonuje odejmowanie binarne 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik odejmowania jest zapamiętywany w obiekcie wskazanym w [D1+1, D1].
- Gdy odjemna jest mniejsza od odjemnika, to LSB będzie niedopełniony i ustawi się flaga carry (F112).
- Gdy wynik odejmowania jest 0, to ustawi się flaga zero.
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

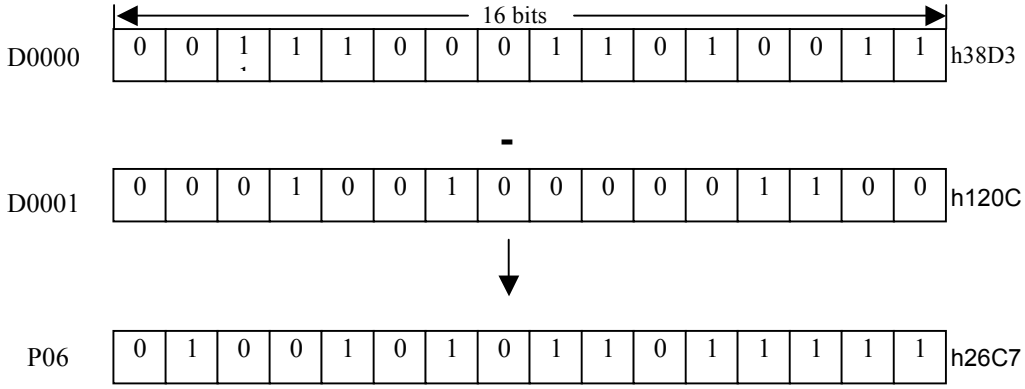
- Warunki wykonywania



7) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P020, odejmij zawartość D0001 od zawartości D0000 i zapamiętaj wynik odejmowania w słowie P06.



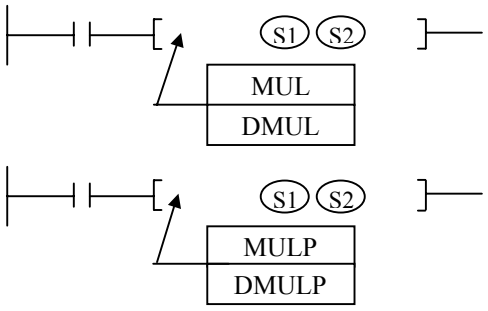


5.8.3 MUL, MULP, DMUL, DMULP

MUL (Binary multiply)	FUN(120) MUL FUN(122) DMUL FUN(121) MULP FUN(123) DMULP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MUL(P) DMUL(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					

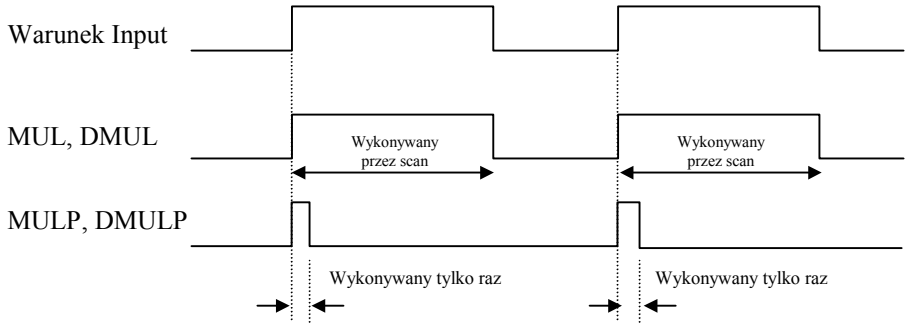


Nastawa operandu

S1	Obiekt zapamiętujący mnożną
S2	Obiekt zapamiętujący mnożnik
D	Obiekt zapamiętujący wynik

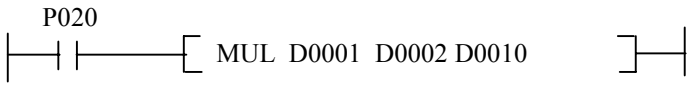
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

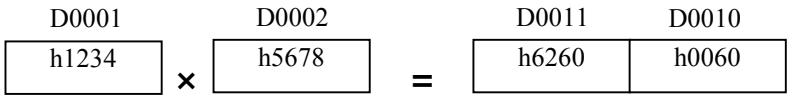
1) Funkcje
- MUL(P) : Wykonuje mnożenie binarne danych wskazanych jako [S1] i [S2]. Wynik mnożenia jest zapamiętywany w obiekcie wskazanym jako [D+1, D].
- DMUL(P) : Wykonuje mnożenie binarne danych wskazanych jako [S1+1, S1] i [S2+1, S2]. Wynik mnożenia jest zapamiętywany w obiekcie wskazanym jako [D+3, D+2, D+1, D]
- Gdy wynik mnożenia jest 0, to ustawi się flaga zero.
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).
- Warunki wykonywania



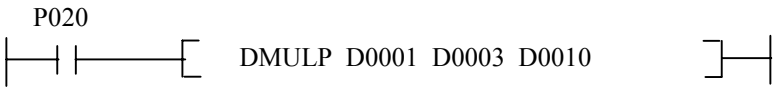
2) Przykład programu

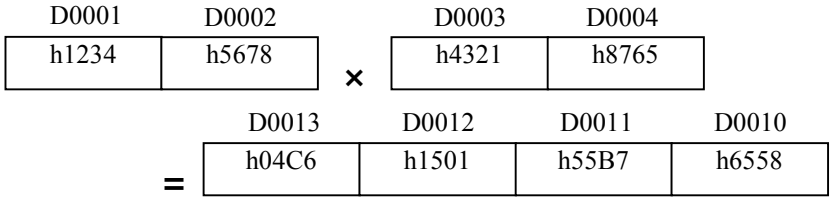
- Program zapamiętuje wynik mnożenia D0001 i D0002 w D0010, D0011 gdy P020 jest ON.





- Program zapamiętuje wynik mnożenia D0001, D0002 i D0003,D0004 w D0010 ~ D0013 gdy P020 jest ON.



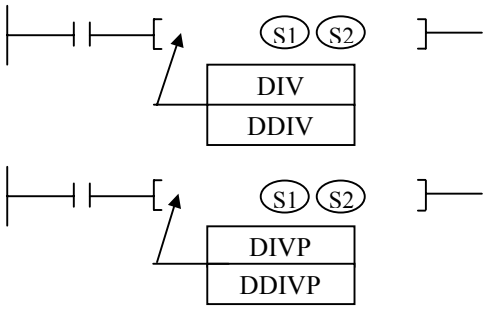


5.8.4 DIV, DIVP, DDIV, DDIVP

DIV (Binary divide)	FUN(124) DIV FUN(126) DDIV FUN(125) DIVP FUN(127) DDIVP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIV(P) DDIV(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					

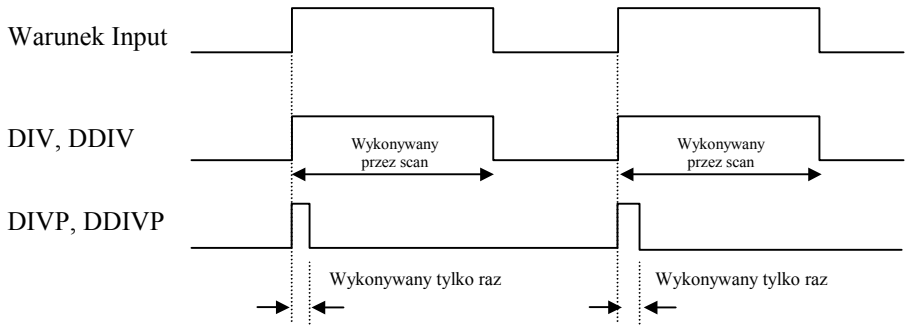


Nastawa operandu

S1	Obiekt zapamiętujący dzielną
S2	Obiekt zapamiętujący dzielnik
D	Obiekt zapamiętujący wynik

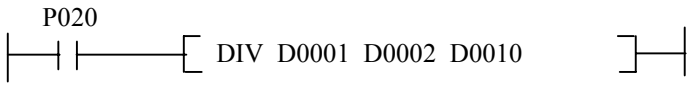
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

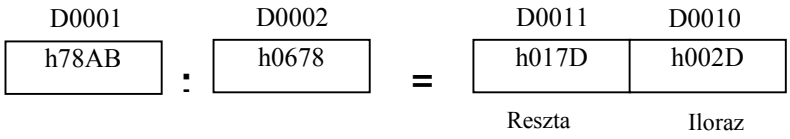
1) Funkcje
   - DIV(P) : Wykonuje dzielenie binarne danych wskazanych jako [S1] i [S2]. Wynik dzielenia jest zapamiętywany w obiekcie wskazanym jako [D+1, D]. Iloraz zapamiętywany jest w [D], a reszta w [D+1].
   - DDIV(P) : Wykonuje dzielenie binarne danych wskazanych jako [S1+1, S1] i [S2+1, S2]. Wynik dzielenia jest zapamiętywany w obiekcie wskazanym jako [D+3, D+2, D+1, D]. Iloraz zapamiętywany jest w [D+1,D], a reszta w [D+3, D+2].
   - Gdy iloraz 0, to ustawi się flaga zero.
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu lub zawartość dzielnika jest 0, to powstanie błąd operacji i ustawi się error flag (F110).
   - Warunki wykonywania



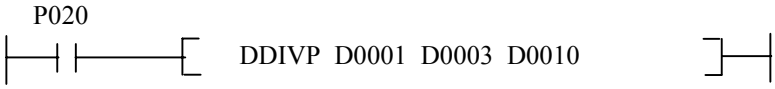
2) Przykład programu

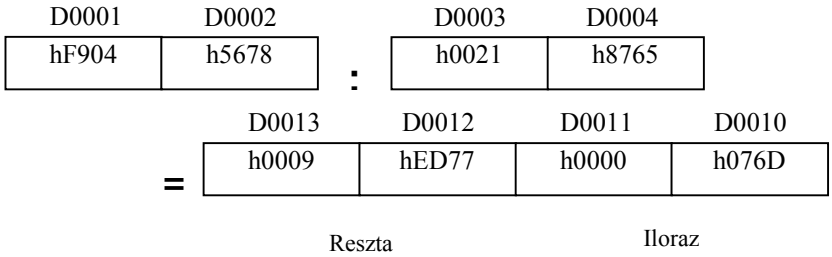
- Program zapamiętuje wynik dzielenia D0001 i D0002 w D0010, D0011 gdy P020 jest ON.





- Program zapamiętuje wynik dzielenia D0001, D0002 i D0003,D0004 w D0010 ~ D0013 gdy P020 jest ON.



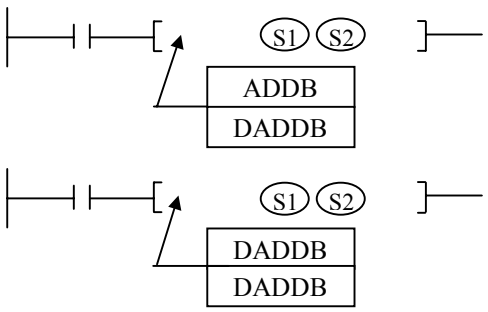


5.9 Instrukcje BCD arithmetic

5.9.1 ADDB, ADDBP, DADDB, DADDBP

ADDB (BCD addition)	FUN(130) ADDB FUN(132) DADDB FUN(131) ADDBP FUN(133) DADDBP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ADDB(P) DADDB(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					



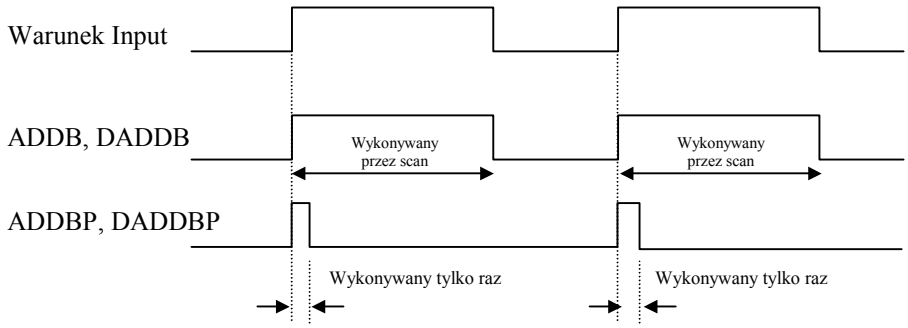
Nastawa operandu

S1	Obiekt zapamiętujący pierwszy składnik
S2	Obiekt zapamiętujący drugi składnik
D	Obiekt zapamiętujący wynik

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

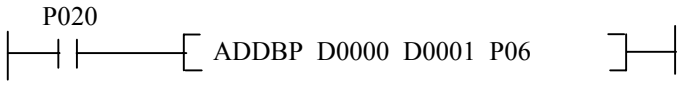
1) Funkcje
- ADDB(P) : Wykonuje dodawanie BCD 16-bitowych danych wskazanych w [S1] i [S2]. Wynik dodawania jest zapamiętywany w obiekcie wskazanym w [D].
- DADD(P) : Wykonuje dodawanie BCD 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik dodawania jest zapamiętywany w obiekcie wskazanym w [D1+1, D1].
- Gdy wynik dodawania przekroczy h9999(ADD / ADDP) lub h99999999(DADD / DADDP), to ustawi się flaga carry (F112).
- Gdy wynik dodawania jest 0, to ustawi się flaga zero.
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu lub zawartość [S1] i [S2] nie jest w kodzie BCD (poza 0 ~ 9), to powstanie błąd operacji i ustawi się error flag (F110).

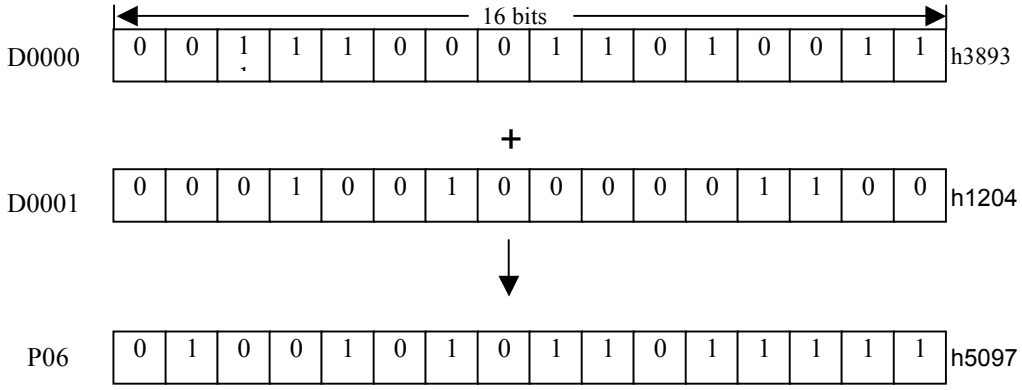
- Warunki wykonywania



8) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P020, dodaj zawartość BCD data D0000 i D0001 i zapamiętaj wynik dodawania w słowie P06.

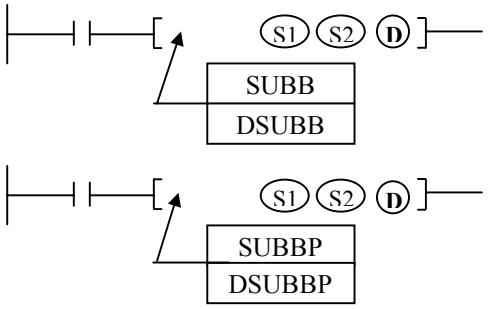




5.9.2 SUBB, SUBBP, DSUBB, DSUBBP

SUBB (BCD subtraction)	FUN(134) SUBB FUN(136) DSUBB FUN(135) SUBBP FUN(137) DSUBBP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SUBB(P) DSUBB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	O
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					



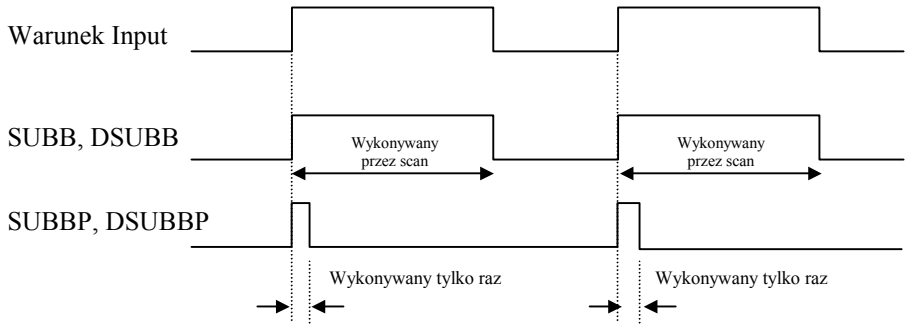
Nastawa operandu

(S1)	Obiekt zapamiętujący odjemną
(S2)	Obiekt zapamiętujący odjemnik
(D)	Obiekt zapamiętujący wynik

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

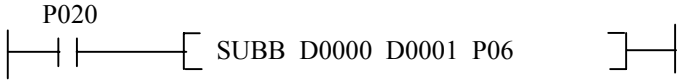
1) Funkcje
   - SUBB(P) : Wykonuje odejmowanie BCD 16-bitowych danych wskazanych w [S1] i [S2]. Wynik odejmowania jest zapamiętywany w obiekcie wskazanym w [D].
   - DSUBB(P) : Wykonuje odejmowanie BCD 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik odejmowania jest zapamiętywany w obiekcie wskazanym w [D1+1, D1].
   - Gdy odjemna jest mniejsza od odjemnika, to LSB będzie niedopełniony i ustawi się flaga carry (F112).
   - Gdy wynik odejmowania jest 0, to ustawi się flaga zero.
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu lub zawartość [S1] i [S2] nie jest w kodzie BCD (poza 0 ~ 9), to powstanie błąd operacji i ustawi się error flag (F110).

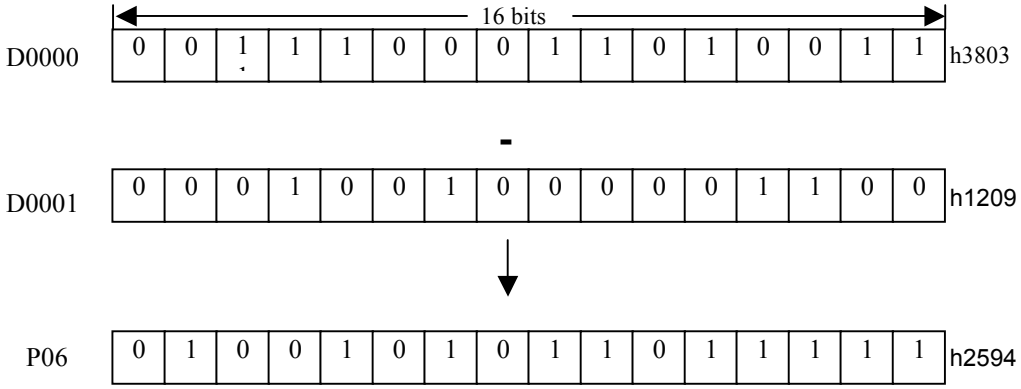
- Warunki wykonywania



9) Przykład programu

- Zawsze kiedy zbocze narastające pojawi się na P020, odejmij zawartość D0001 od zawartości D0000 i zapamiętaj wynik odejmowania w słowie P06.

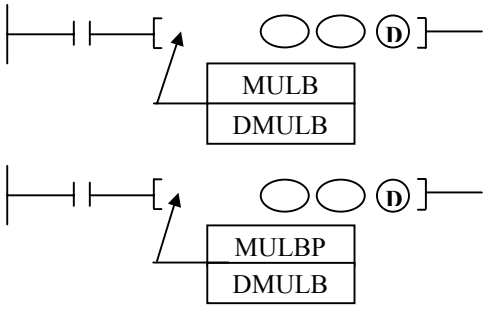




5.9.3 MULB, MULBP, DMULB, DMULBP

MULB (BCD multiply)	FUN(140) MULB FUN(142) DMULB FUN(141) MULBP FUN(143) DMULBP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
MULB(P) DMULB(P)	⬭	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	⬭	O	O	O	O	O	O	O		O	O	O				
	◯	O	O	O	O*		O	O		O	O					

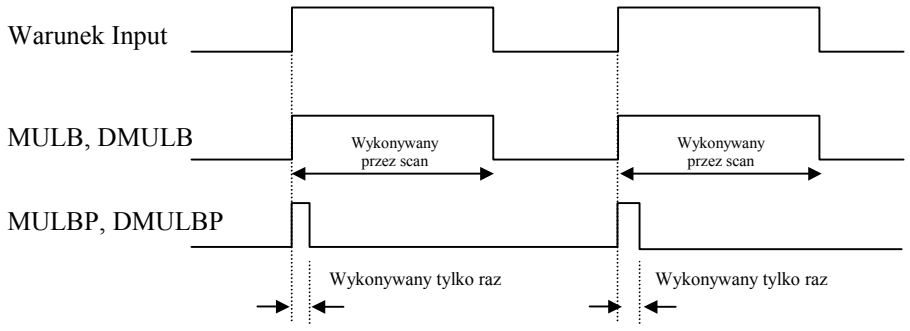


Nastawa operandu

⬭	Obiekt zapamiętujący mnożną
⬭	Obiekt zapamiętujący mnożnik
Ⓓ	Obiekt zapamiętujący wynik

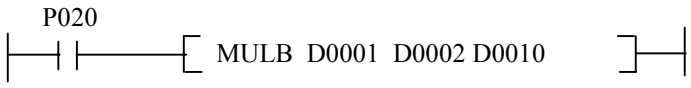
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

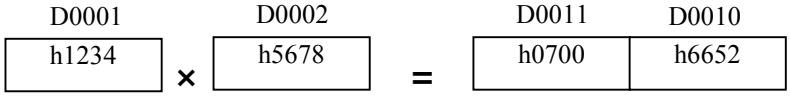
1) Funkcje
   - MULB(P) : Wykonuje mnożenie BCD danych wskazanych jako [S1] i [S2]. Wynik mnożenia jest zapamiętywany w obiekcie wskazanym jako [D+1, D].
   - DMULB(P) : Wykonuje mnożenie BCD danych wskazanych jako [S1+1, S1] i [S2+1, S2]. Wynik mnożenia jest zapamiętywany w obiekcie wskazanym jako [D+3, D+2, D+1, D]
   - Gdy wynik mnożenia jest 0, to ustawi się flaga zero.
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).
   - Warunki wykonywania



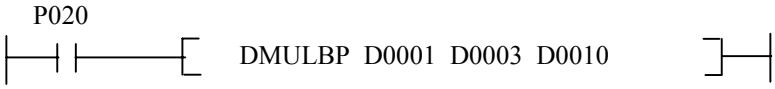
3) Przykład programu

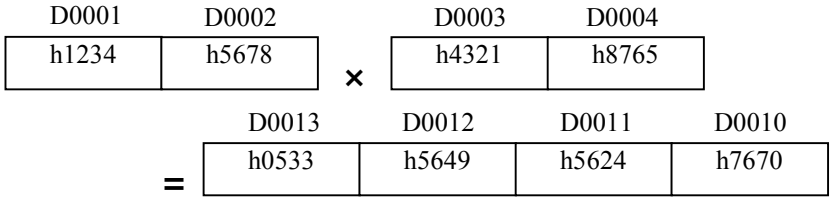
- Program zapamiętuje wynik mnożenia D0001 i D0002 w D0010, D0011 gdy P020 jest ON.





- Program zapamiętuje wynik mnożenia D0001, D0002 i D0003,D0004 w D0010 ~ D0013 gdy P020 jest ON.



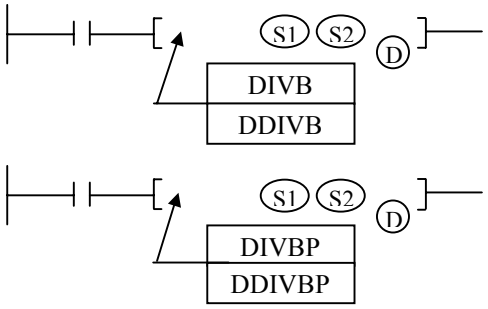


5.9.4 DIVB, DIVBP, DDIVB, DDIVBP

DIVB (BCD divide)	FUN(144) DIVB FUN(146) DDIVB FUN(145) DIVBP FUN(147) DDIVBP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIVB(P) DDIVB(P)	(S1)	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	(S2)	O	O	O	O	O	O	O		O	O	O				
	(D)	O	O	O	O*		O	O		O	O					



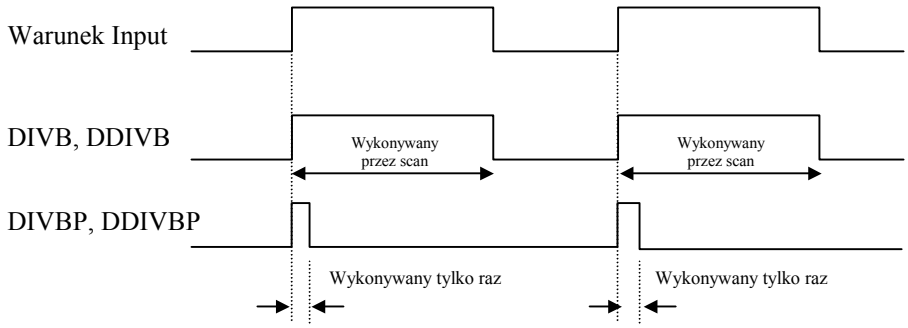
Nastawa operandu

(S1)	Obiekt zapamiętujący dzielną
(S2)	Obiekt zapamiętujący dzielnik
(D)	Obiekt zapamiętujący wynik

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

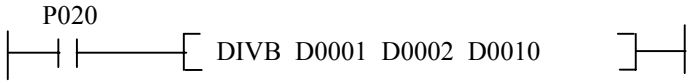
1) Funkcje
   - DIVB(P) : Wykonuje dzielenie BCD danych wskazanych jako [S1] i [S2]. Wynik dzielenia jest zapamiętywany w obiekcie wskazanym jako [D+1, D]. Iloraz zapamiętywany jest w [D], a reszta w [D+1].
   - DDIVB(P) : Wykonuje dzielenie BCD danych wskazanych jako [S1+1, S1] i [S2+1, S2]. Wynik dzielenia jest zapamiętywany w obiekcie wskazanym jako [D+3, D+2, D+1, D]. Iloraz zapamiętywany jest w [D+1,D], a reszta w [D+3, D+2].
   - Gdy iloraz 0, to ustawi się flaga zero.
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu lub zawartość dzielnika jest 0 lub zawartość [S1] i [S2] nie jest w kodzie BCD (poza 0 ~ 9), to powstanie błąd operacji i ustawi się error flag (F110).

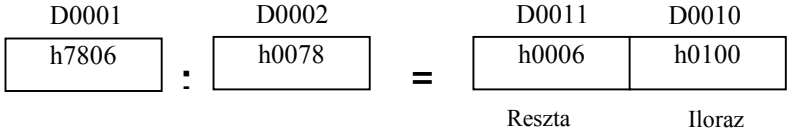
- Warunki wykonywania



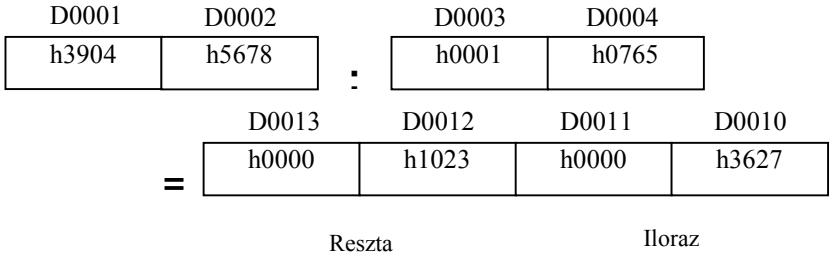
2) Przykład programu

- Program zapamiętuje wynik dzielenia D0001 i D0002 w D0010, D0011 gdy P020 jest ON.





- Program zapamiętuje wynik dzielenia D0001, D0002 i D0003,D0004 w D0010 ~ D0013 gdy P020 jest ON.

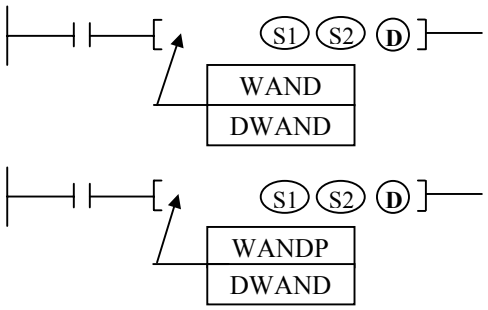


5.10 Instrukcje Logical arithmetic

5.10.1 WAND, WANDP, DWAND, DWANDP

WAND (Word AND)	FUN(150) WAND DWAND FUN(151) WANDP DWANDP	FUN(152) FUN(153)	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WAND(P) DWAND(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					



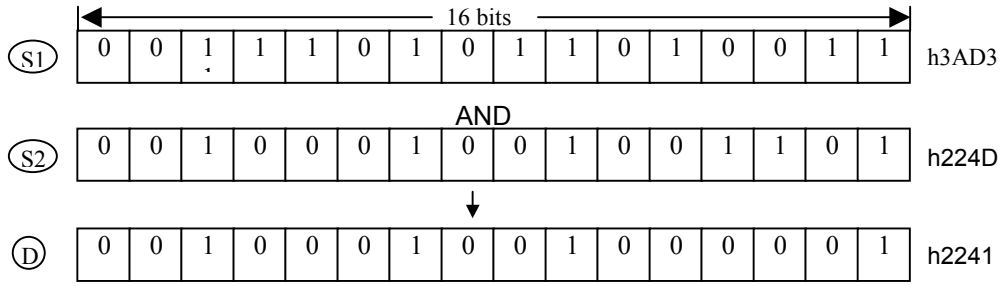
Nastawa operandu

S1 S2	Dane na których będą wykonywane funkcje logiczne
D	Obiekt w którym jest zapamiętywany wynik operacji logicznej

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

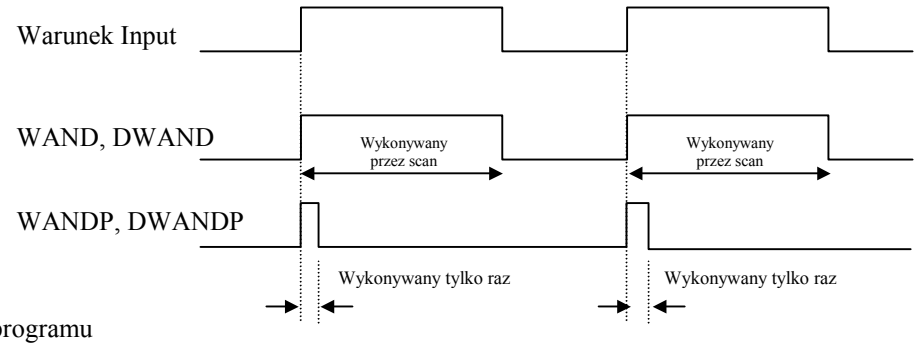
1) Funkcje

- WAND(P) : Wykonuje operację logiczną na 16-bitowych danych wskazanych w [S1] i [S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D]



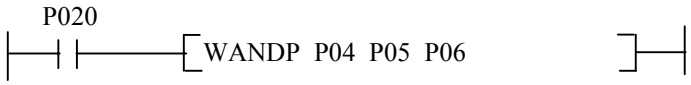
- DWAND(P) : Wykonuje operację logiczną na 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D+1, D].
- Gdy wynik mnożenia jest 0, to ustawi się flaga zero (F111).
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

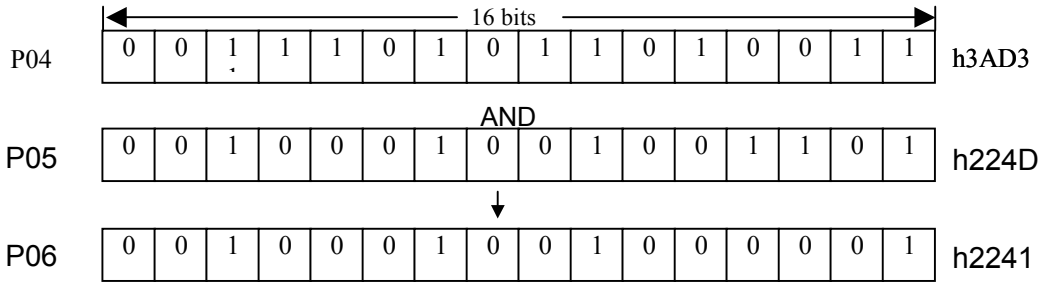
- Warunki wykonywania



2) Przykład programu

- Program wykonuje iloczyn logiczny na słowach P04 i P05 oraz zapamiętuje wynik w słowie P06, gdy P020 jest ON.



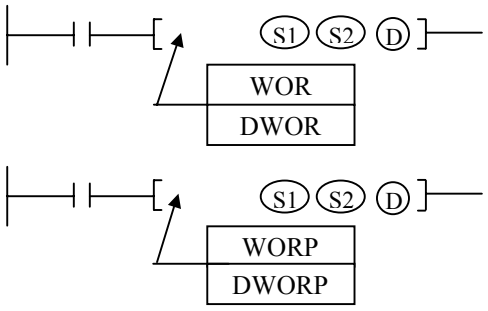


5.10.2 WOR, WORP, DWOR, DWORP

WOR (Word OR)	FUN(154) WOR FUN(156) DWOR FUN(155) WORP FUN(157) DWORP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WOR(P) DWOR(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					

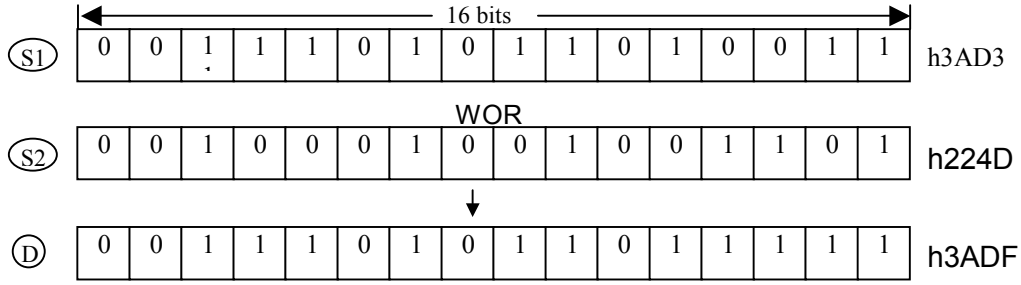


Nastawa operandu

S1 S2	Dane na których będą wykonywane funkcje logiczne
D	Obiekt w którym jest zapamiętywany wynik operacji logicznej

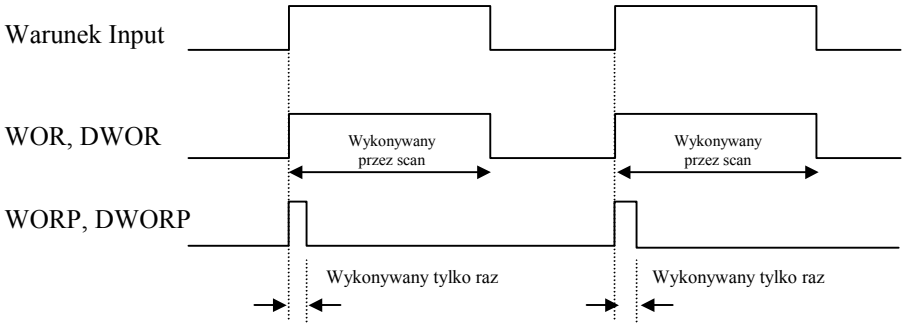
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - WOR(P) : Wykonuje logiczną sumę na 16-bitowych danych wskazanych w [S1] i [S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D]



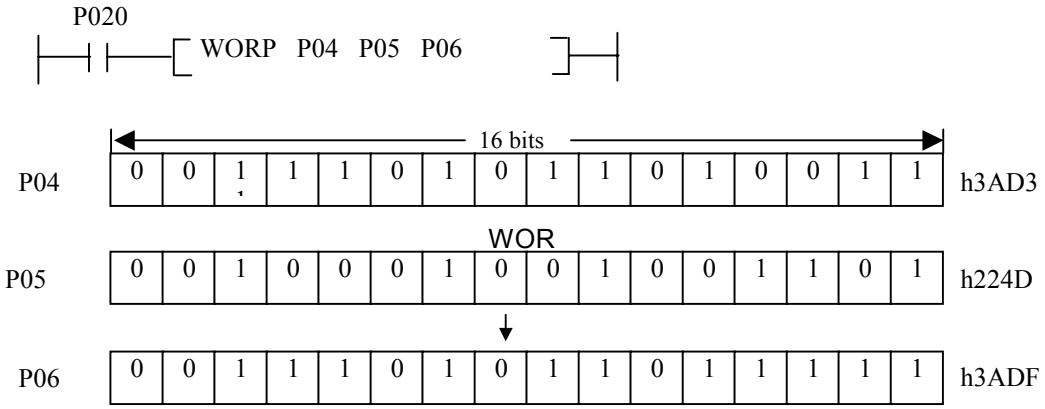
   - DWOR(P) : Wykonuje sumę logiczną na 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D+1, D].
   - Gdy wynik mnożenia jest 0, to ustawi się flaga zero (F111).
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

- Warunki wykonywania



2) Przykład programu

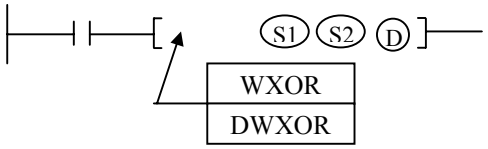
- Program wykonuje sumę logiczną na słowach P04 i P05 oraz zapamiętuje wynik w słowie P06, gdy P020 jest ON.



5.10.3 WXOR, WXORP, DWXOR, DWXORP

WXOR (Word exclusive OR	FUN(160) WXOR FUN(162) DWXOR FUN(161) WXORP FUN(163) DWXORP	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WXOR(P) DWXOR(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					

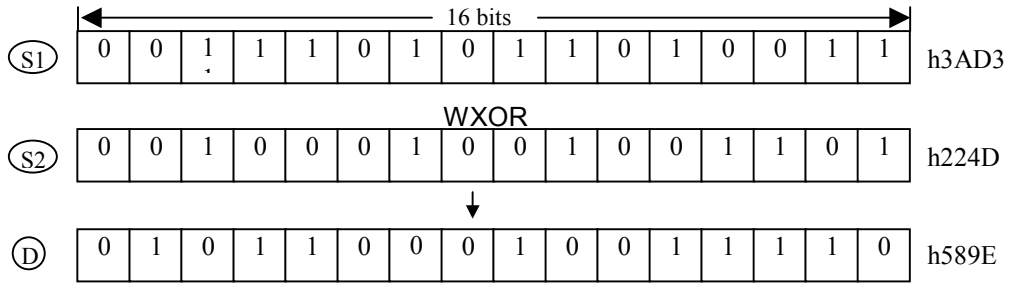


Nastawa operandu

S1 S2	Dane na których będą wykonywane funkcje logiczne
D	Obiekt w którym jest zapamiętywany wynik operacji logjcznej

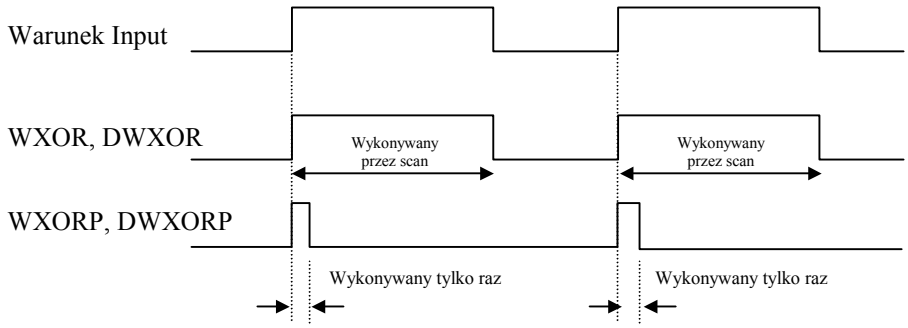
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - WXOR(P) : Wykonuje exclusive OR na 16-bitowych danych wskazanych w [S1] i [S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D]



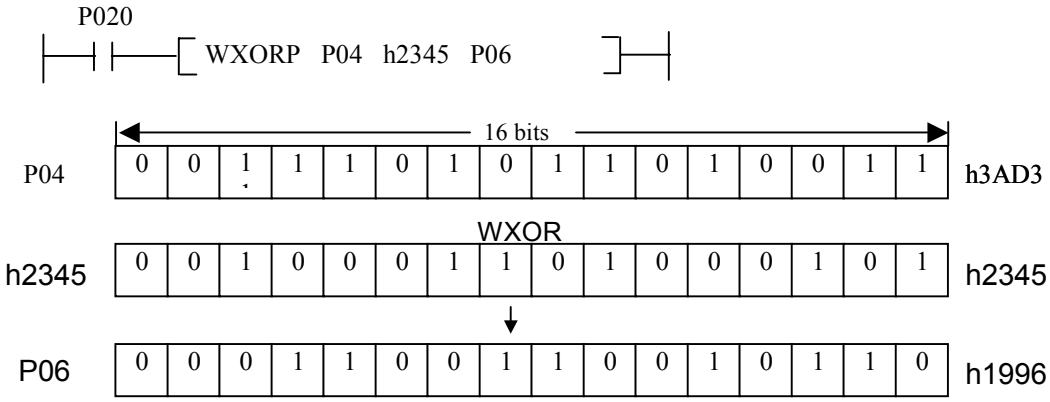
   - DWXOR(P) : Wykonuje exclusive OR na 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D+1, D].
   - Gdy wynik mnożenia jest 0, to ustawi się flaga zero (F111).
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

- Warunki wykonywania



2) Przykład programu

- Program wykonuje exclusive OR na słowach P04 i h2345 oraz zapamiętuje wynik w słowie P06, gdy P020 jest ON.

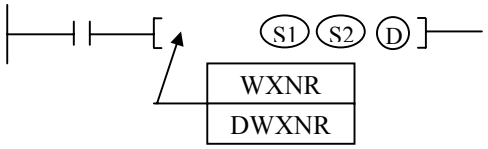


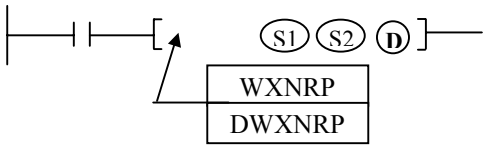
5.10.4 WXNR, WXNRP, DWXNR, DWXNRP

WXOR (Word exclusive NOR)	FUN(164) WXNR FUN(166) DWXNR FUN(165) WXNRP FUN(167) DWXNRP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
WXNR(P) DWXNR(P)	S1	O	O	O	O	O	O	O		O	O	O	7/9/11	O	O	
	S2	O	O	O	O	O	O	O		O	O	O				
	D	O	O	O	O*		O	O		O	O					



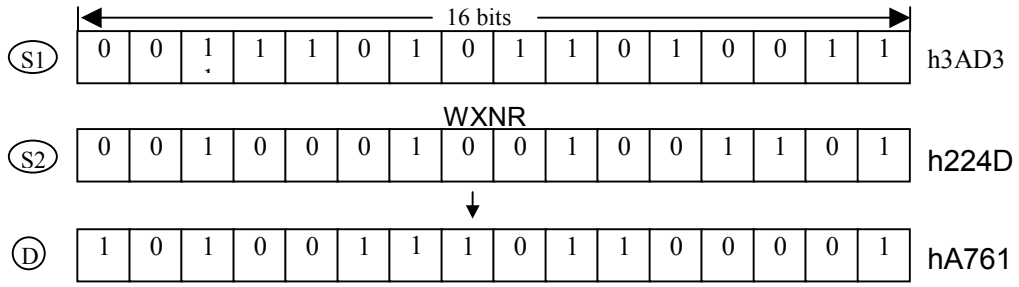


Nastawa operandu

S1 S2	Dane na których będą wykonywane funkcje logiczne
D	Obiekt w którym jest zapamiętywany wynik operacji logjcznej

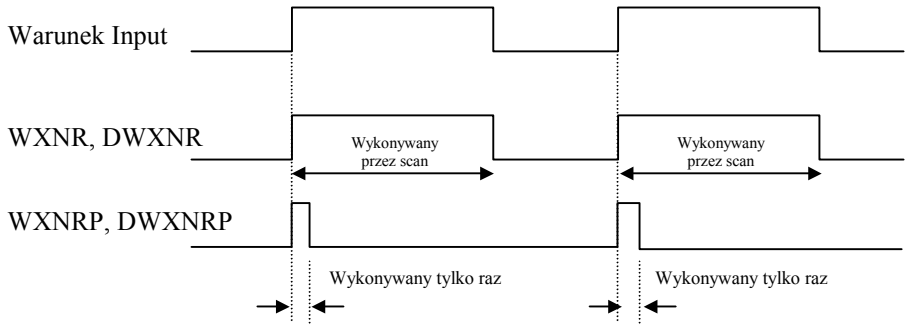
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - WXNR(P) : Wykonuje exclusive NOR na 16-bitowych danych wskazanych w [S1] i [S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D]



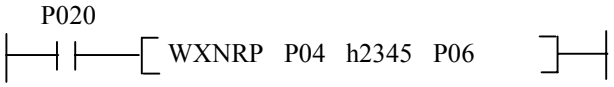
   - DWXNR(P) : Wykonuje exclusive NOR na 32-bitowych danych wskazanych w [S1+1, S1] i [S2+1, S2]. Wynik operacji jest zapamiętywany w obiekcie wskazanym w [D+1, D].
   - Gdy wynik exclusive NOR jest 0, to ustawi się flaga zero (F111).
   - Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).

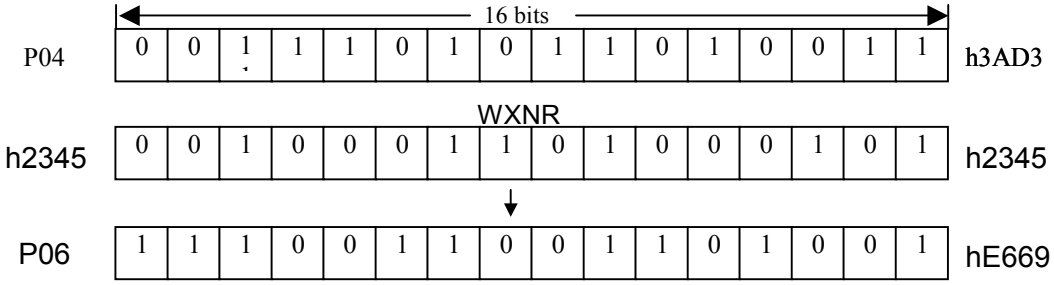
- Warunki wykonywania



2) Przykład programu

- Program wykonuje exclusive NOR na słowach P04 i h2345 oraz zapamiętuje wynik w słowie P06, gdy P020 jest ON.





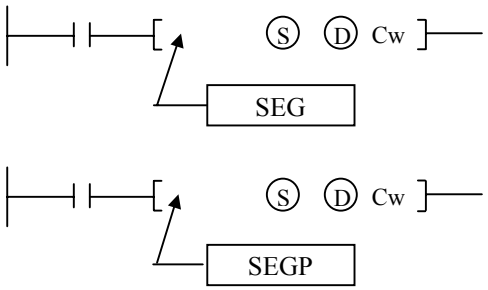
5.11 Instrukcje Data processing

5.11.1 SEG, SEGP

SEG (7 segment)	FUN(174) SEG FUN(175) SEGP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SEG SEGP	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	Cw											O				



Nastawa operandu

Ⓢ	Obiekt w którym zapamiętywane są dane źródłowe
Ⓓ	Obiekt w którym zapamiętywane są dane 7 segmentowe
Cw	Informacje o bicie startu i ilości bitów

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

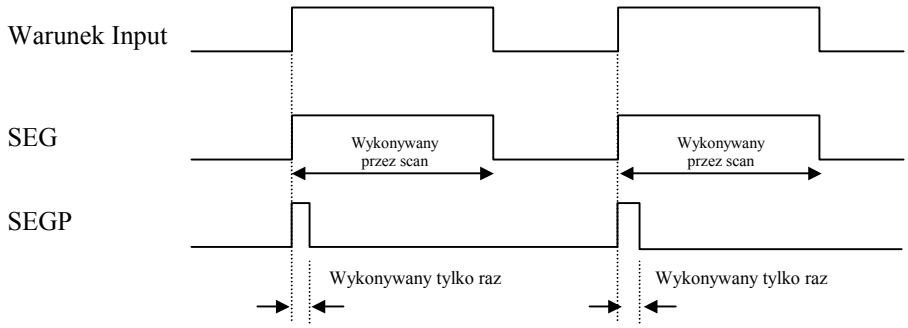
1) Funkcje
- Format ‘Cw’

h	s	d	x	z

a) s : Bit startu w [S].
b) d : Bit startu w [D]
c) x : Nieistotne
d) z : Liczba dekodowanych elementów. (zakres : 0 ~ F)

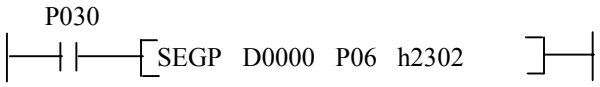
- Dekoduje dane z×4 bitowe bloki poczynając od bitu s obiektu wskazanego w [S] do danych display’a 7 segmentowego i zapamiętuje wynik w z×8 bitowych blokach rozpoczynających się od bitu s obiektu [D].

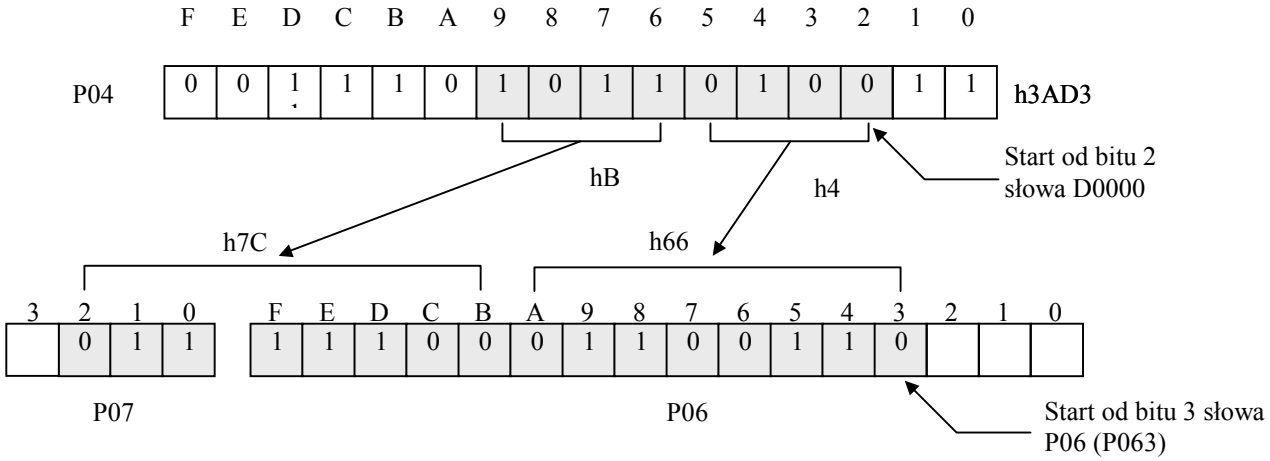
- Warunki wykonywania



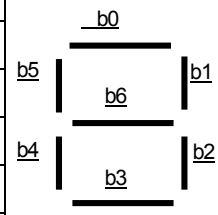
2) Przykład programu

- Gdy P030 jest ON, program dekoduje 8-bitów poczynając od bitu 2 słowa D0000 do formatu 7 segmentowego display'a i zapamiętuje 16-bitowy wynik w słowie P06 poczynając od bitu3.





3) Dane wyświetlacza 7 segmentowego

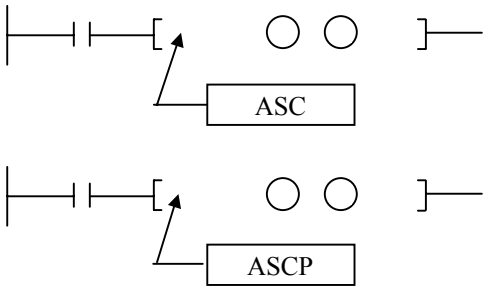
Ⓢ		Konfiguracja 7 segmentowa	Ⓓ								Display
Hex	Binary		b7	b6	b5	b4	b3	b2	b1	b0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	B
C	1100		0	0	1	1	1	0	0	1	C
D	1101		0	1	0	1	1	1	1	0	D
E	1110		0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

5.11.2 ASC, ASCP

ASC (ASCII code)	FUN(190) ASC FUN(191) ASCP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ASC ASCP	○	O	O	O	O	O	O	O		O	O		7	O		
	○	O	O	O	O*		O	O		O	O					
												O				



Operand setting

○	Obiekt w którym zapamiętywane są dane źródłowe
○	Obiekt w którym zapamiętywane są dane ASCII
	Obiekt w którym zapamiętywane są dane źródłowe

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

2) Funkcje

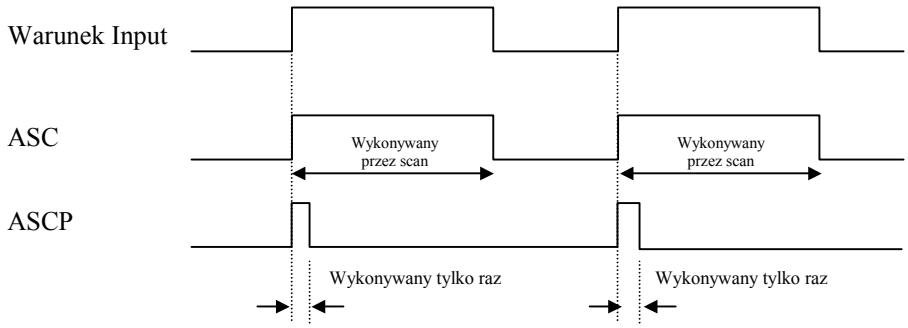
- Format 'Cw'

h	s	d	x	z

e) s : Bit startu w [S].
f) d : Bit startu w [D]
g) x : Nieistotne
h) z : Liczba dekodowanych elementów. (zakres : 0 ~ F)

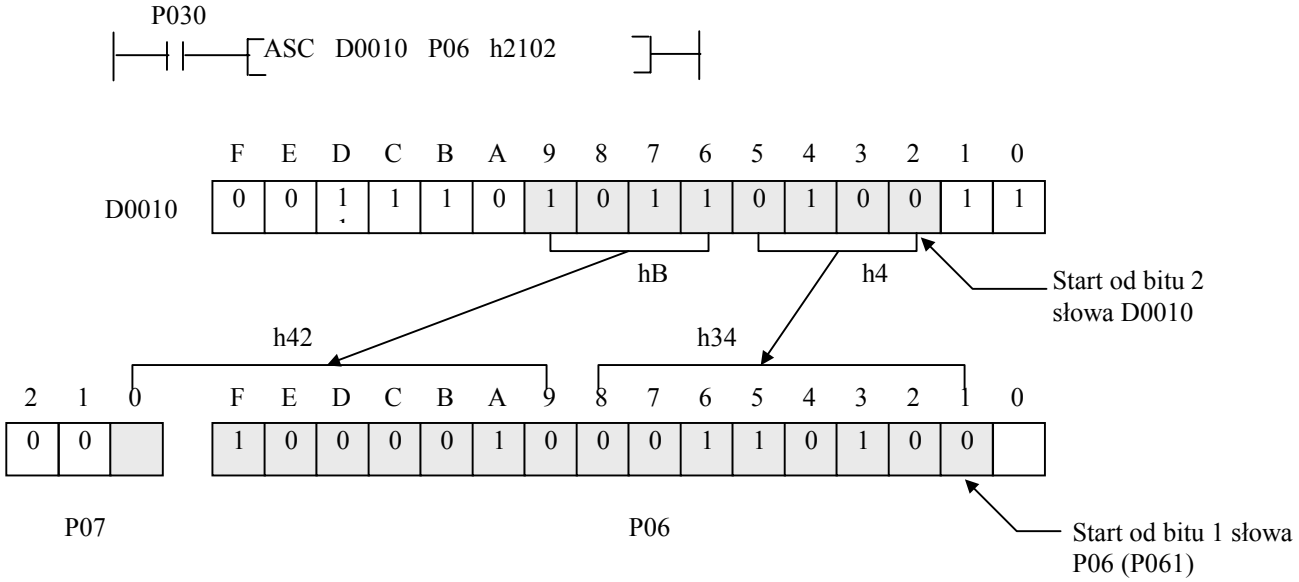
- Dokonuje konwersji danych z×4 bitowe bloki poczynając od bitu s obiektu wskazanego w [S] do danych w kodzie ASCII i zapamiętuje wynik w z×8 bitowych blokach rozpoczynających się od bitu s obiektu [D].

- Warunki wykonywania



2) Przykład programu

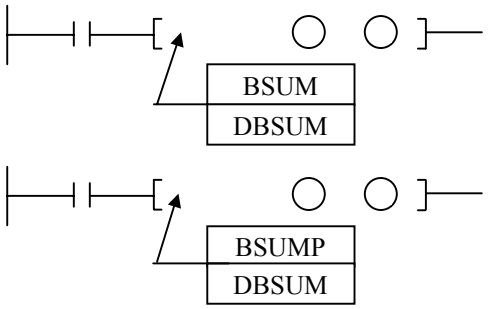
- Gdy P030 jest ON, program dokonuje konwersji 8-bitów poczynając od bitu 2 słowa D0010 na dane kodu ASCII i zapamiętuje 16-bitowy wynik w słowie P06 poczynając od bitu1.



5.11.3 BSUM, BSUMP, DBSUM, DBSUMP

BSUM (Bit summary)	FUN(170) BSUM DBSUM FUN(171) BSUMP DBSUMP	FUN(172) FUN(173)	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Steps	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
BSUM(P)	◯	O	O	O	O	O	O	O		O	O	O	5	O	O	
DBSUM(P)	◯	O	O	O	O*		O	O		O	O					

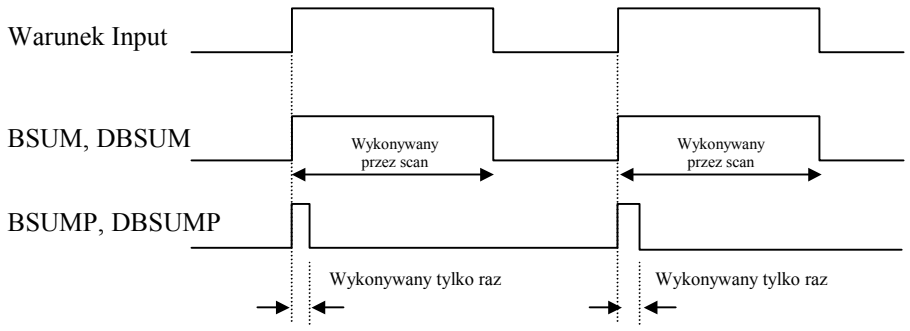


Nastawa operandu

◯	Obiekt w którym liczone są '1'
◯	Obiekt w którym zapamiętywany jest wynik

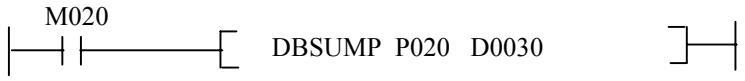
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

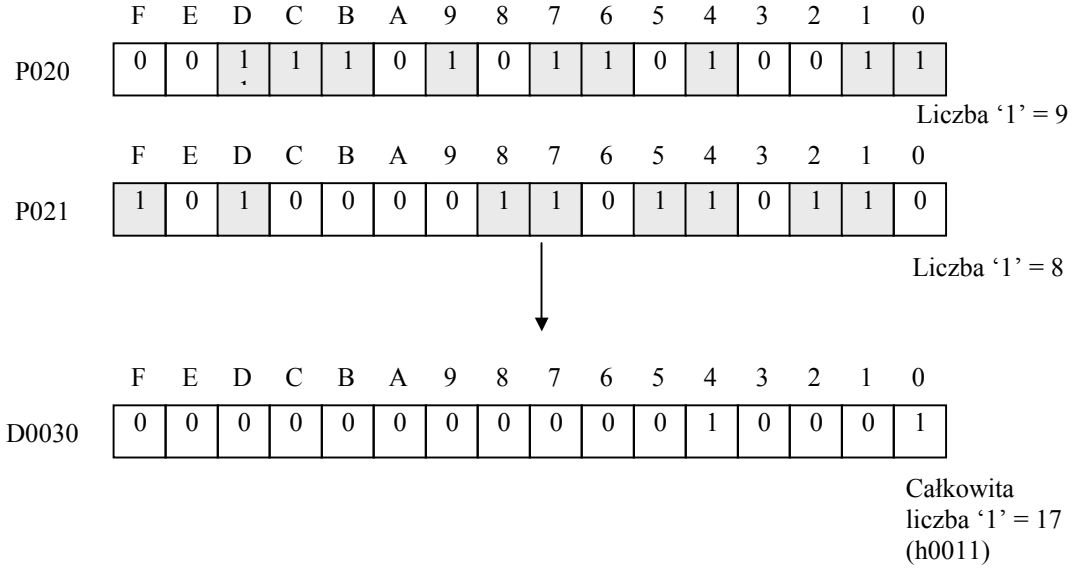
1) Funkcje
- BSUM(P) : Liczy ilość '1' w obiekcie wskazanym jako [S], następnie zapamiętuje wynik w obiekcie wskazanym jako [D] w formacie hexadecy-malnym.
- DBSUM(P) : Liczy ilość '1' w obiekcie wskazanym jako [S+1, S], następnie zapamiętuje wynik w obiekcie wskazanym jako [D] w formacie hexadecy-malnym.
- Gdy wynik liczenia jest 0, to ustawi się flaga zero (F111).
- Jeżeli adres pośredni wskazany przez #D jest poza zakresem obiektu, to powstanie błąd operacji i ustawi się error flag (F110).
- Warunki wykonywania



2) Przykład programu

- Gdy M020 jest ON, program liczy ilość ‘1’ w P020 i P021, następnie zapamiętuje wynik liczenia w D0030.



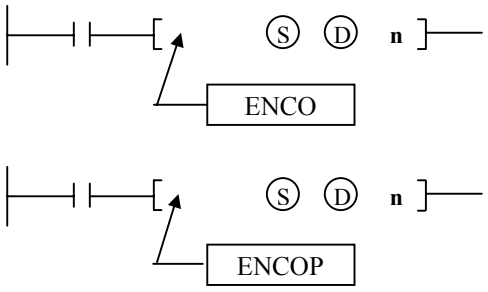


5.11.4 ENCO, ENCOP

ENCO (Encode)	FUN(176) ENCO FUN(177) ENCOP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
ENCO ENCOP	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

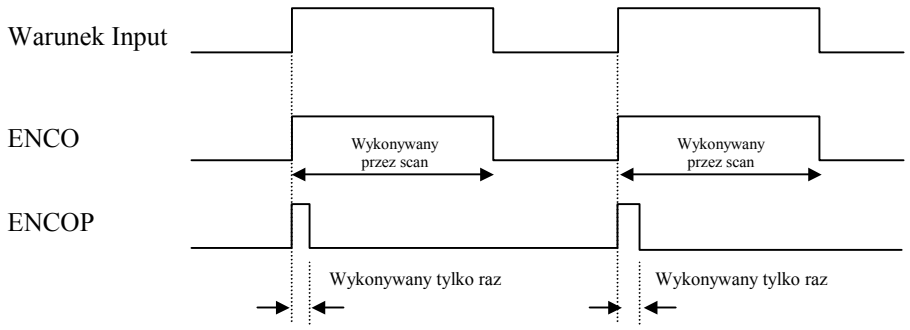


Nastawa operandu

Ⓢ	Adres początku obszaru danych źródłowych
Ⓓ	Adres początku obszaru przeznaczenia zapamięta wynik enkodowania
n	Efektywna długość bitu 2^n (1 ~ 8)

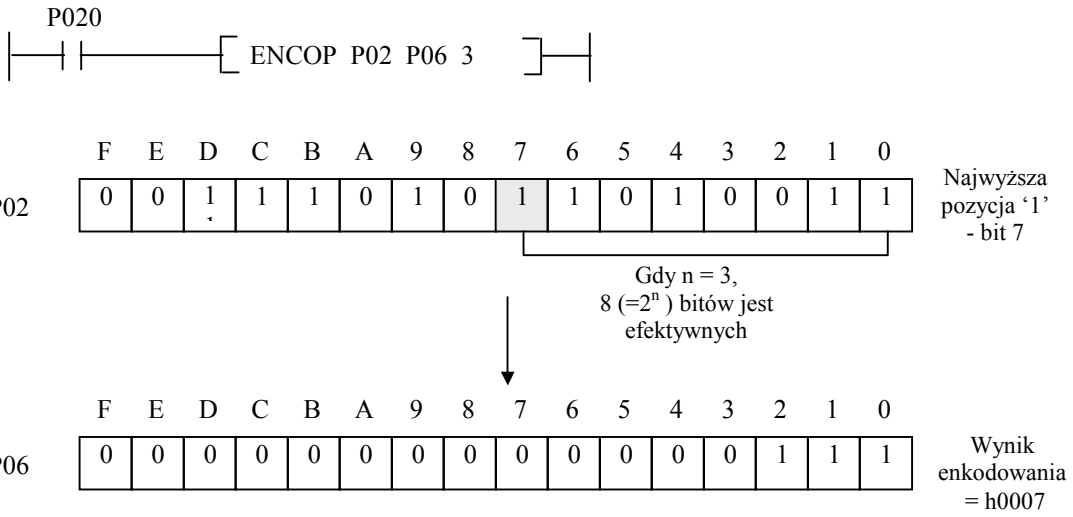
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- Enkoduje dane 2^n bitów, począwszy od bitu 0 obiektu wskazanego jako [S], a wynik zapamiętuje w obiekcie wskazanym jako [D].
- Dla 'n', 1 ~ 8 może być wskazane. Jeżeli wartość n jest poza zakresem, to brak działania i zawartość [D] nie zmienia się.
- Gdy wiele bitów jest 1, to działanie obejmuje najbardziej znaczący bit. Gdy wynik jest 0, to ustawi się flaga zero (F111).
- Gdy wartość n jest większa niż 4, to obszar danych źródłowych rozszerzy się jak [S+1], [S+2], … Gdy n=8, to długość danych źródłowych wynosi 256 bitów. ([S+15, S+14, …, S+1, S])
- Warunki wykonywania



2) Przykład programu

- Gdy P020 jest ON, program enkoduje 8-bitów (bit 0 ~ bit 7) słowa P02 i zapamiętuje wynik w słowie P06.

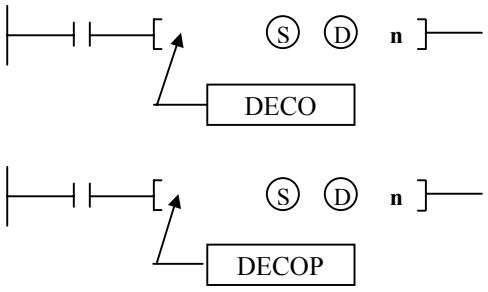


5.11.5 DECO, DECOP

DECO (Decode)	FUN(178) DECO FUN(179) DECOP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DECO DECOP	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

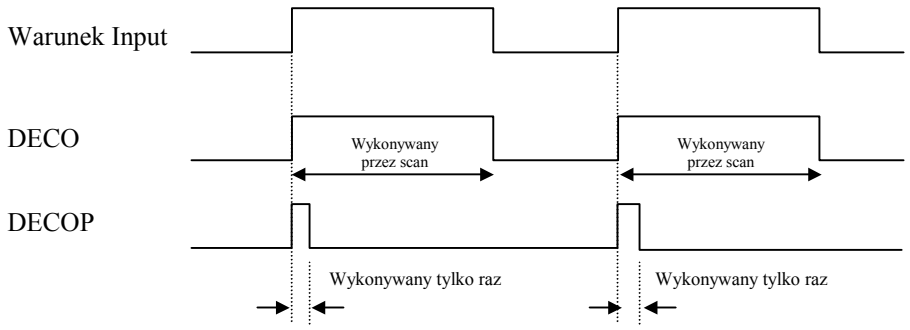


Nastawa operandu

Ⓢ	Adres początku obszaru danych źródłowych
Ⓓ	Adres początku obszaru przeznaczenia zapamięta wynik dekodowania
n	Efektywna długość bitu 2^n (1 ~ 8)

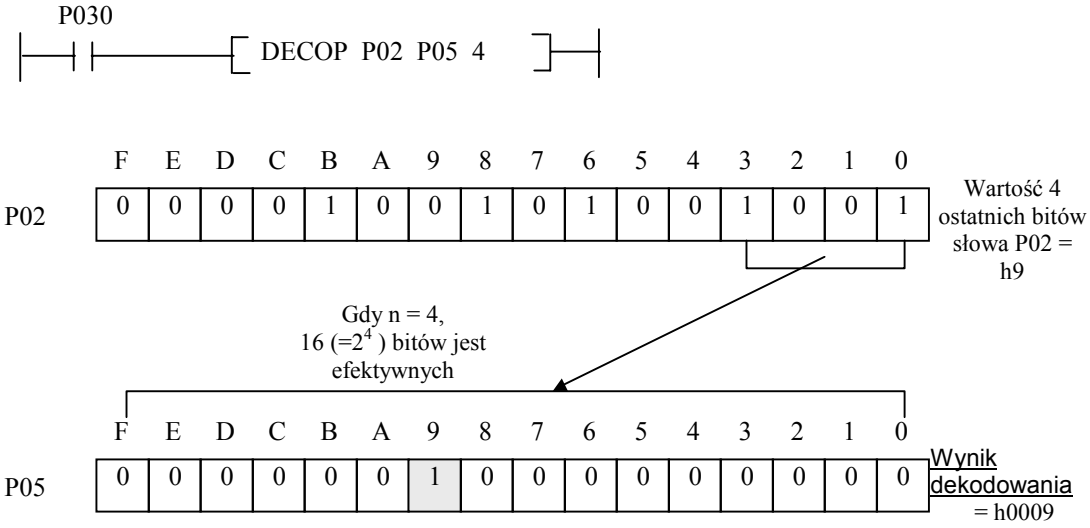
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- Dekoduje dane 2^n bitów, począwszy od bitu 0 obiektu wskazanego jako [S], a wynik zapamiętuje w obiekcie wskazanym jako [D].
- Dla 'n', 1 ~ 8 może być wskazane. Jeżeli wartość n jest 0, to brak działania i zawartość [D] nie zmienia się. Jeżeli wartość n jest powyżej 8, to brak działania i zostaje ustawiona flaga error (F110).
- Gdy wiele bitów jest 1, to działanie obejmuje najbardziej znaczący bit. Gdy wynik jest 0, to ustawi się flaga zero (F111).
- Gdy wartość n jest większa niż 4, to obszar danych źródłowych rozszerzy się jak [D+1], [D+2], … Gdy n=8, to długość danych źródłowych wynosi 256 bitów. ([D+15, D+14, …, D+1, D])
- Warunki wykonywania

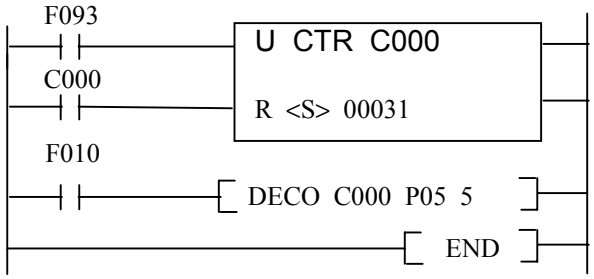


2) Przykład programu

- Gdy P030 jest ON, program dekoduje 4 młodsze bity słowa P02 i zapamiętuje wynik dekodowania w słowie P05.



- Program dekoduje wartość bieżącą licznika C000 i zapamiętuje wynik dekodowania w słowach P05 i P06. Wartość bieżąca licznika zwiększa się co 1 sekundę i gdy osiągnie wartość 31, licznik C000 zeruje się.

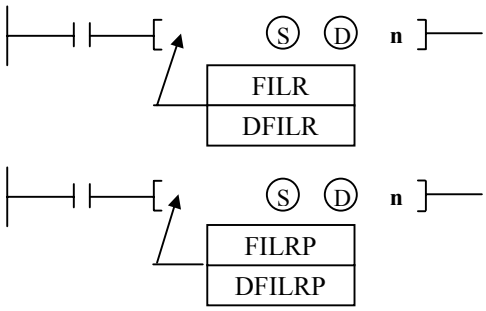


5.11.6 FILR, FILRP, DFILR, DFILRP

FILR (File table read)	FUN(180) FILR FUN(182) DFILR FUN(181) FILRP FUN(183) DFILRP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
FILR(P) DFILR(P)	Ⓢ	O	O	O	O	O	O	O		O	O	O	7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

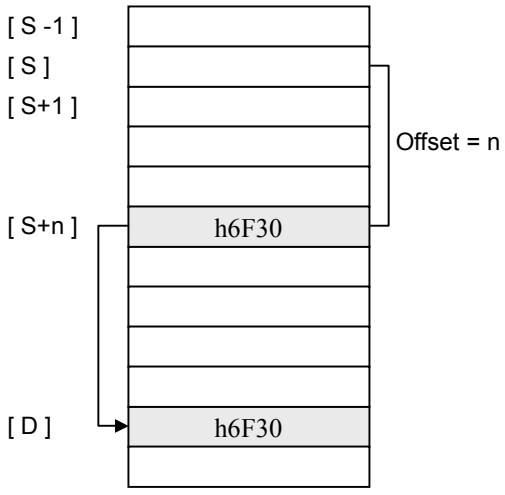


Nastawa operandu

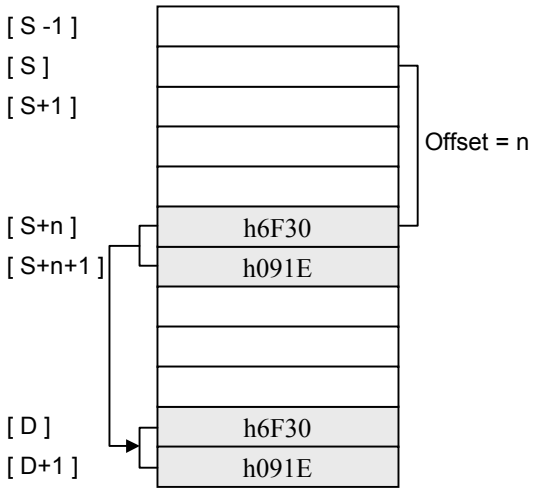
Ⓢ	Pierwotny adres słowa danych źródłowych
Ⓓ	Słowo przeznaczenia do którego zawartość [S+n] jest przenoszona
n	Offset

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- FILR(P) : Przenosi zawartość słowa [S+n] do obiektu wskazanego jako [D]
- DFILR(P) : Przenosi zawartość słowa [S+n+1, S+n] do obiektu wskazanego jako [D+1, D].
- Gdy [S+n] jest poza zakresem korespondującego obszaru obiektu, to brak działania i ustawia się flaga error.

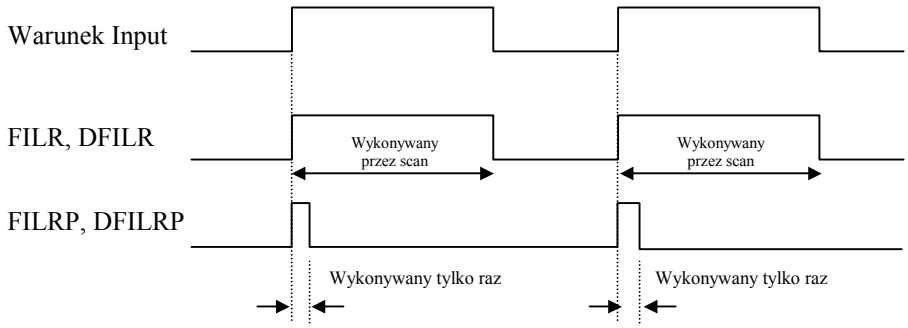


FILR(P)



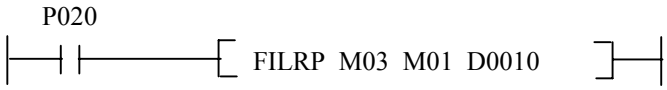
DFILR(P)

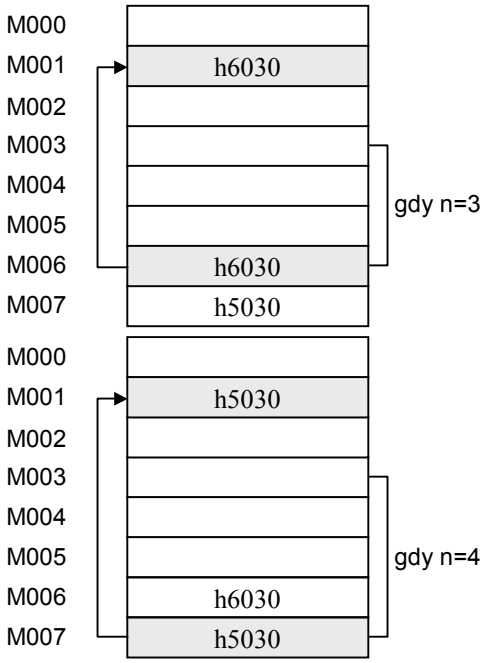
- Warunki wykonywania



2) Przykład programu

- Program przenosi zawartość słowa M03+n do słowa M01, gdy P020 jest ON. Offset n zapamiętywany jest w słowie D0010.

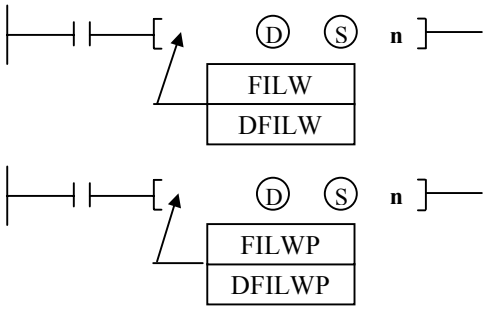




5.11.7 FILW, FILWP, DFILW, DFILWP

FILW (File table write)	FUN(184) FILW DFILW FUN(185) FILWP DFILWP	FUN(186) FUN(187)	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
FILW(P) DFILW(P)	Ⓓ	O	O	O	O	O*	O	O		O	O	O	7	O		
	Ⓢ	O	O	O	O	O	O	O		O	O					
	n									O		O				

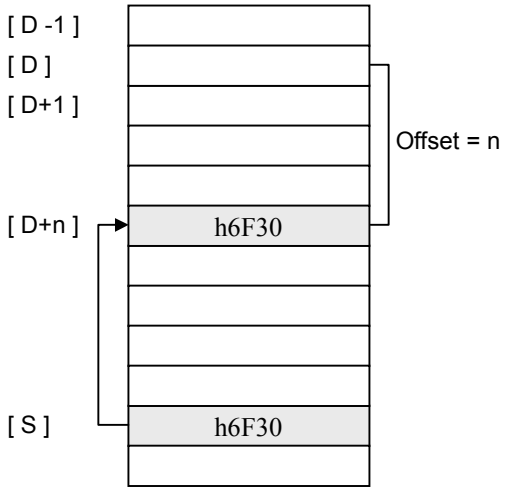


Nastawa operandu

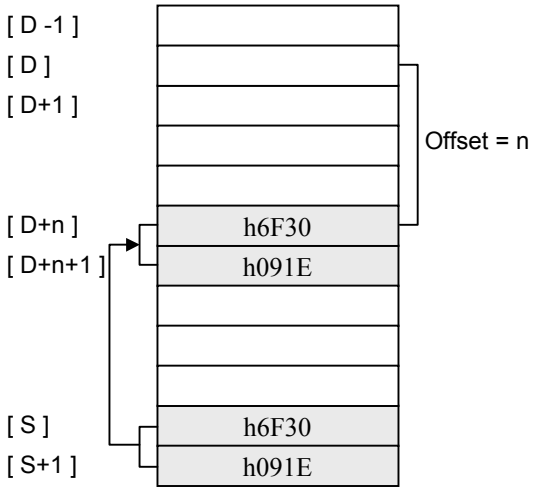
Ⓓ	Pierwotny adres słowa danych źródłowych
Ⓢ	Dane źródłowe lub obiekt w którym dane źródłowe są zapamiętywane
n	Offset

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- FILW(P) : Przenosi zawartość słowa [S] do obiektu wskazanego jako [D+n]
- DFILW(P) : Przenosi zawartość słowa [S+1, S] do obiektu wskazanego jako [D+n+1, D+n].
- Gdy [D+n] jest poza zakresem korespondującego obszaru obiektu, to brak działania i ustawia się flaga error.

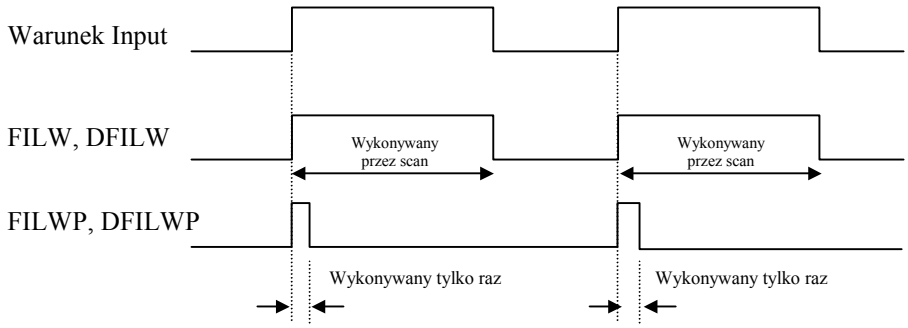


FILW(P)



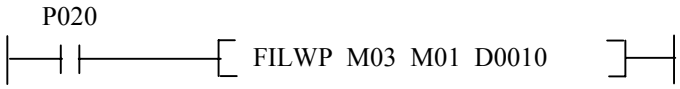
DFILW(P)

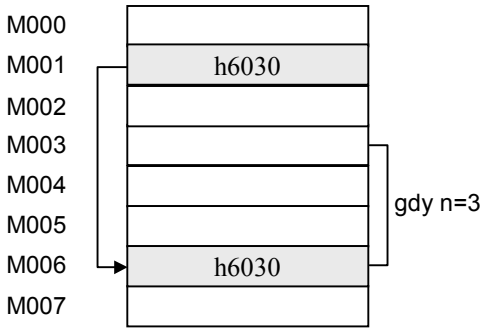
- Warunki wykonywania

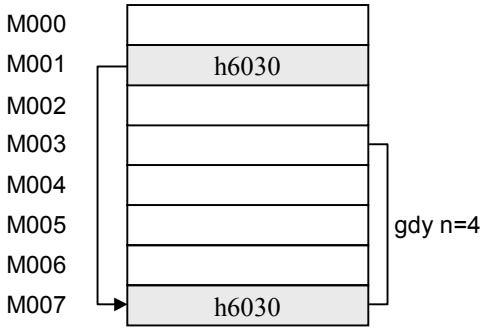


2) Przykład programu

- Program przenosi zawartość słowa M01 do słowa M03+n, gdy P020 jest ON. Offset n zapamiętywany jest w słowie D0010.





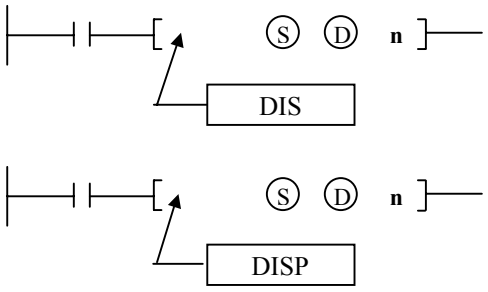


5.11.8 DIS, DISP

DIS (Data dissociation)	FUN(194) DIS FUN(195) DISP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DIS(P)	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				



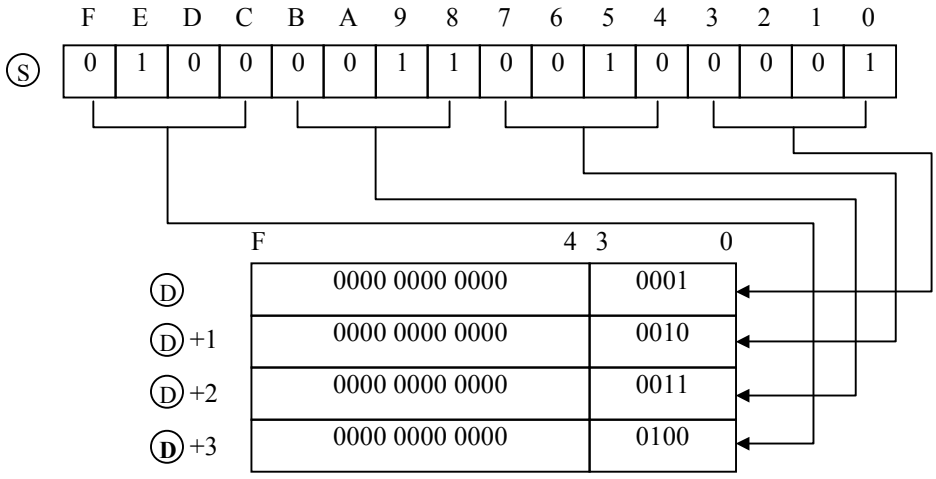
Nastawa operandu

Ⓢ	Obiekt źródłowy
Ⓓ	Początek adresu obiektu przeznaczenia
n	Liczba elementów do oddzielenia (1 ~ 4)

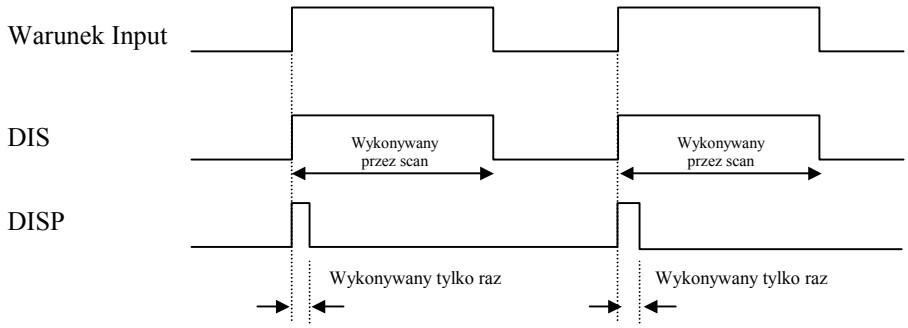
* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
- Przenosi n elementów począwszy od bitu 0 obiektu wskazanego jako [S] do 4 niższych bitów bloku wskazanego jako [D+n-1] ~ [D].
- 12 bits (bit 4 ~ bit F) bloku wskazanego jako [D+n-1] ~ [D] jest zerowanych.
- Gdy n=0, brak działania.
- Gdy n > 4, to brak działania flaga error flag zostaje ustawiona.

Gdy n = 4,
16 (=4×4) bitów jest efektywnych

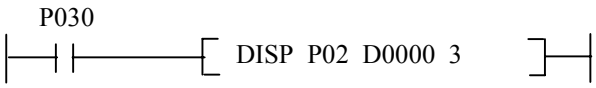


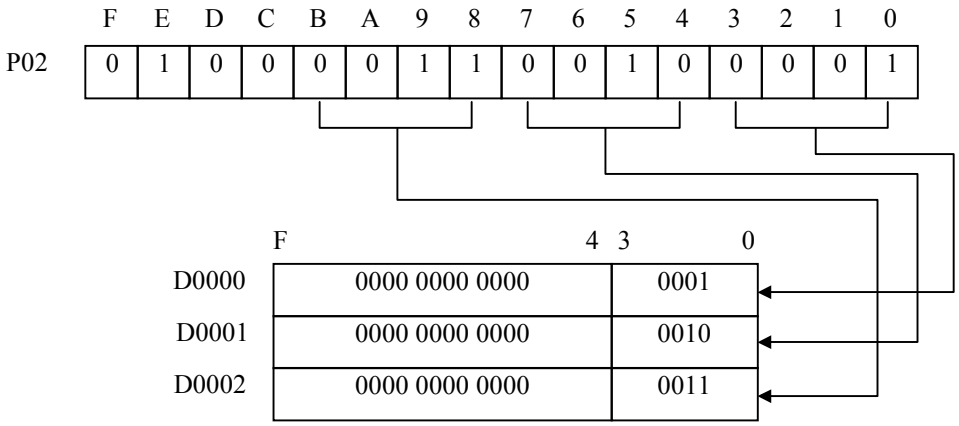
- Warunki wykonywania



2) Przykłady programów

- Program oddziela zawartość 3 niższych elementów słowa P02 do 4 niższych bitów słów D0000 ~ D0003, gdy P030 jest ON.



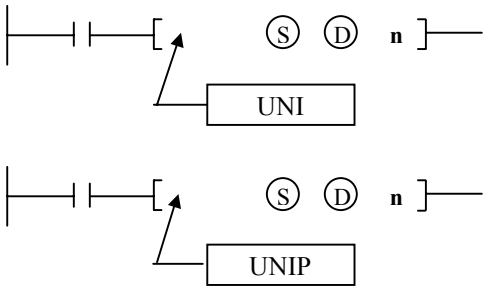


5.11.9 UNI, UNIP

UNI (Data association)	FUN(192) UNI FUN(193) UNIP

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
UNI(P)	Ⓢ	O	O	O	O	O	O	O		O	O		7	O		
	Ⓓ	O	O	O	O*		O	O		O	O					
	n									O		O				

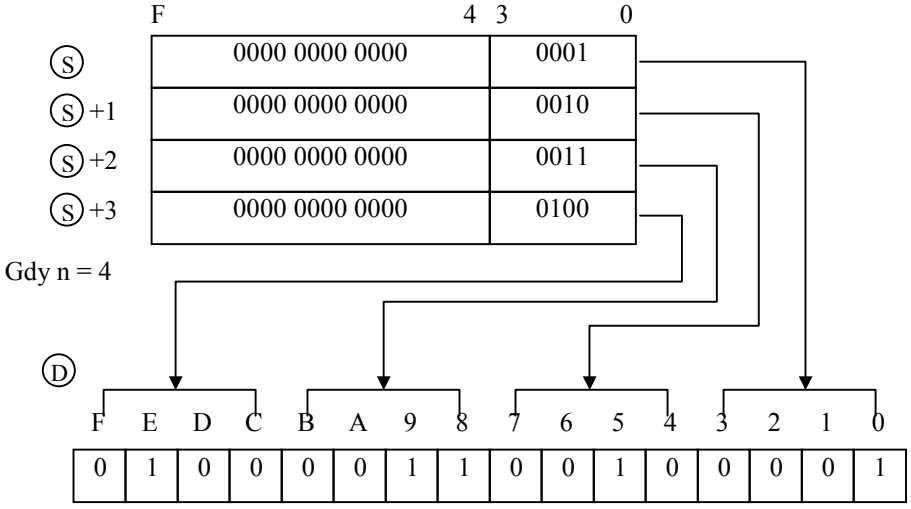


Nastawa operandu

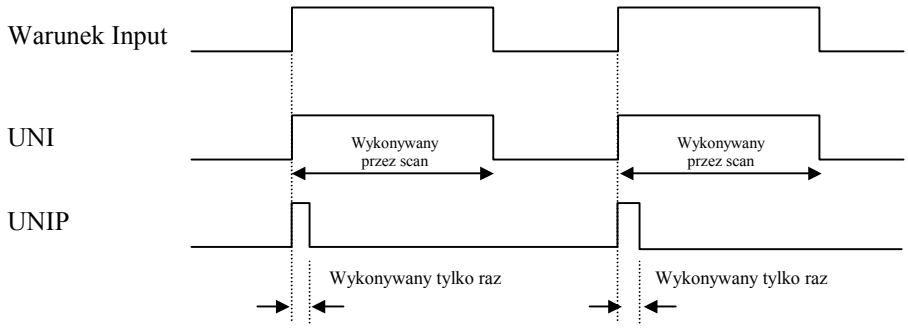
Ⓢ	Początek adresu obiektu źródłowego
Ⓓ	Obiekt przeznaczenia
n	Liczba elementów do połączenia (1 ~ 4)

* Dostępne tylko wtedy, gdy nie jest używany moduł computer link lub data link

1) Funkcje
   - Przenosi najniższe 4 bity bloku wskazanego jako [S+n-1] ~ [S] do n młodszych elementów obiektu wskazanego jako [D].
   - Wyższe bity (bit 2^n ~ bit F) obiektu wskazanego jako [D] są zerowane.
   - Gdy n=0, brak działania.
   - Gdy n > 4, to brak działania i flaga error zostaje ustawiona.

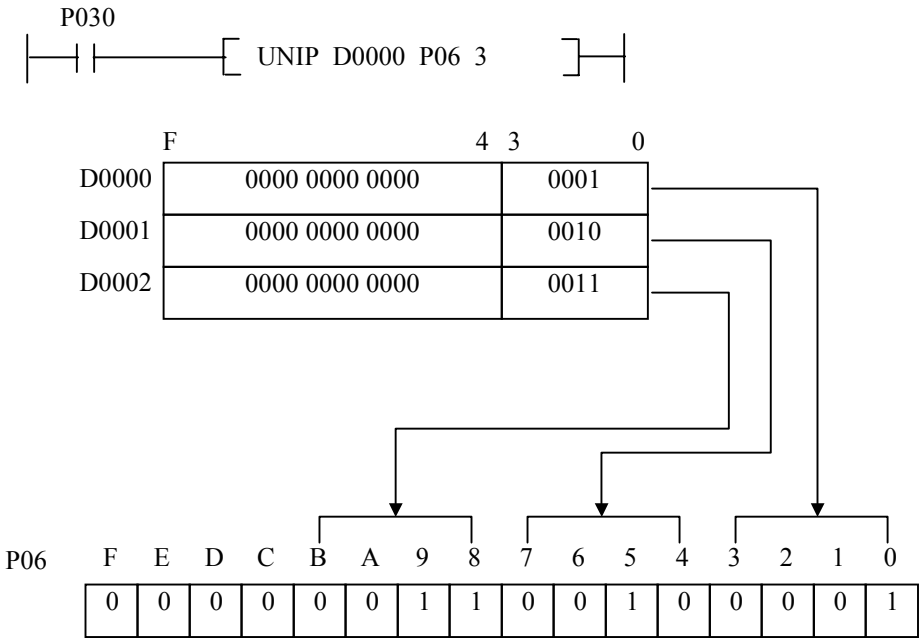


- Warunki wykonywania



2) Przykłady programów

- Program łączy zawartość 4 niższych bitów słów D0000 ~ D0003 do 3 niższych elementów słowa P02, gdy P030 jest ON.



5.12 System instructions

5.12.1 DUTY

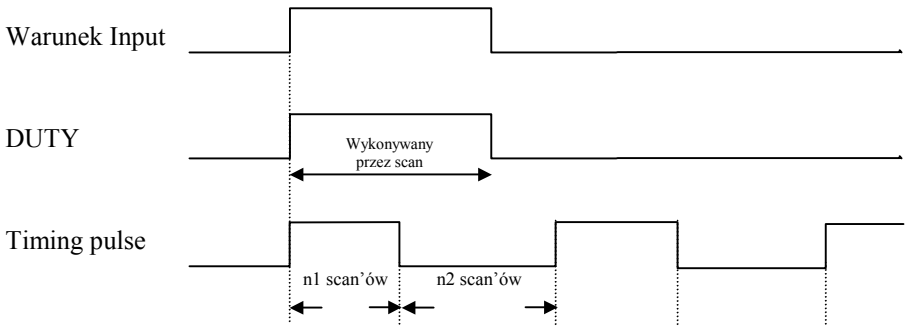
DUTY (User defined pulse)	FUN(205) DUTY	Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
DUTY	Ⓓ					O							7			
	n1											O				
	n2											O				

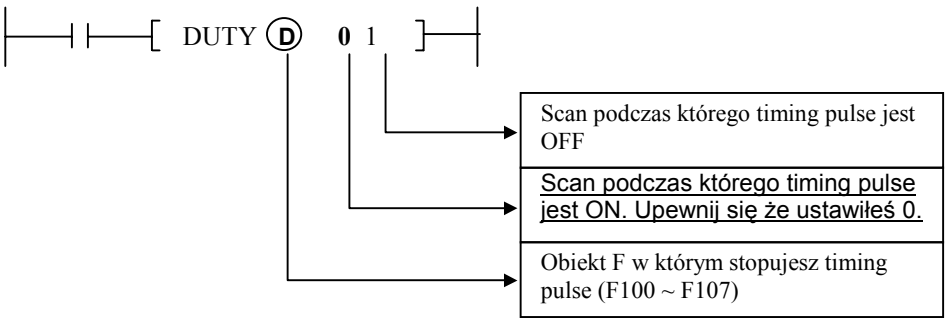
Nastawa operandu

Ⓓ	Styk obiektu F na który wystawiany impuls
n1	**Liczba scan'ów podczas których impuls jest ON**
n2	Liczba scan'ów podczas których impuls jest OFF

1) Funkcje
- Generuje impulsy czasowe (timing clock) wskazane jako [D], zdefiniowany przez użytkownika, gdzie timing pulse jest ON podczas scan'ów wskazanych jako 'n1' i OFF podczas scan'ów wskazanych jako 'n2'.
- Status początkowy - timing pulse OFF.
- Gdy 'n1' =0, timing pulse jest zawsze OFF.
- Gdy 'n1' >0 i 'n2' =0, to timing pulse jest zawsze ON.

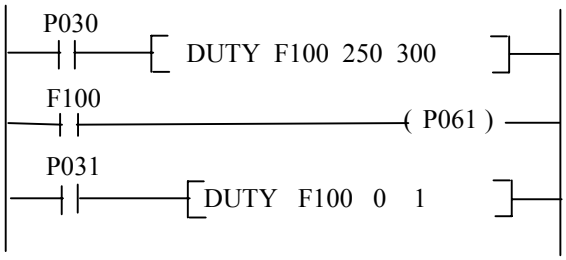


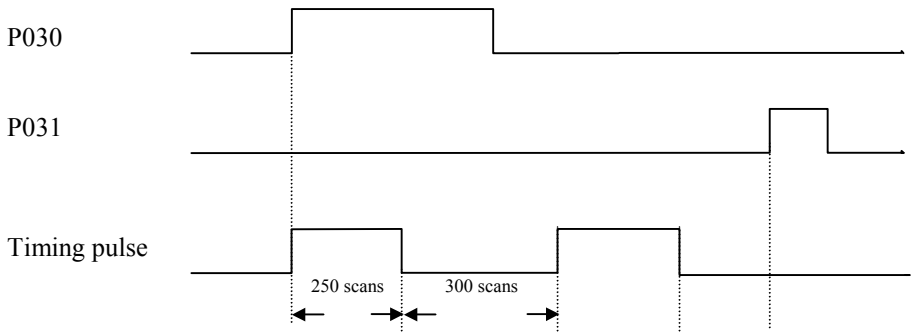
- Jeżeli warunek input instrukcji DUTY zostanie ustawiony na OFF, to timing pulse nie przejdzie w stan OFF. Aby spowodować zatrzymanie timing pulse, należy wykonać nową instrukcję DUTY jak pokazano poniżej.



2) Przykład programu

- Program generuje timing pulse o 250 scan'ach ON, i 300 scan'ach OFF i wystawia go na styk F100, gdy P030 ustawi się na ON. Gdy P031 ustawi się na ON, to timing pulse zostanie zatrzymany.

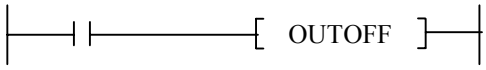




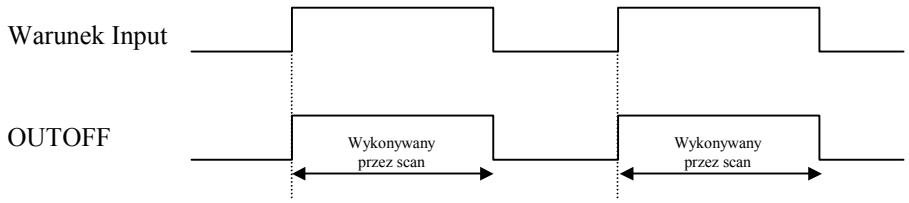
5.12.2 OUTOFF

OUTOFF (Wszystkie output off)	FUN(208) OUTOFF	Stosowane w CPU	Wszystkie CPU

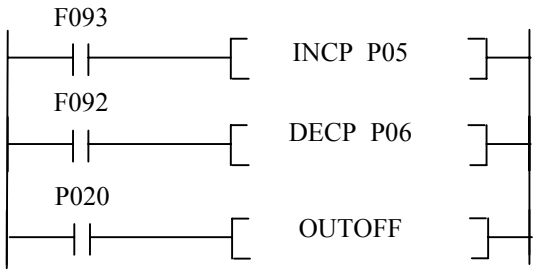
Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
OUTOFF												1			



1) Funkcje
- Zatrzymuje wystawianie wyników operacji obszaru P na zewnętrzne urządzenia i ustawia flagę OUTOFF (F113), gdy warunek input jest ON. Jednakże, obiekty P są aktualizowane zgodnie z wynikami operacji.
- Gdy warunek input zostanie wyłączony, CPU restartuje wystawianie wyników operacji obszaru P na zewnętrzne urządzenia.
- Użyteczny test pracy systemu PLC.
- Warunki wykonywania



2) Przykład programu
- Program stopuje wystawianie output do zewnętrznych urządzeń, gdy P020 jest ON.



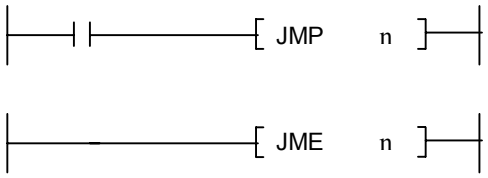
5.13 Instrukcje skoków - Branch

5.13.1 JMP, JME

JMP (Jump)	FUN(012) JMP FUN(013) JME

Stosowane w CPU	Wszystkie CPU

Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
JMP JME	n											O	1			

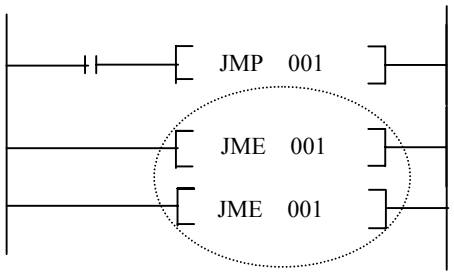


Nastawa operandu

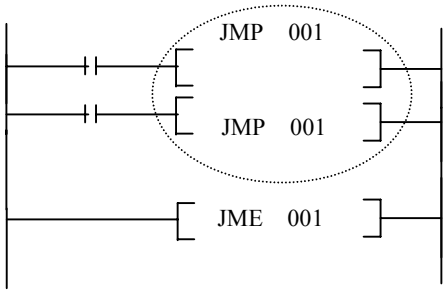
n	K10S1/K10S/K30S/K60S : 0 ~ 63 K200S/K300S/K1000S : 0 ~ 127

1) Funkcje

- Gdy instrukcja ‘JMP n’ jest wykonywana przez ustawienie warunku input, to CPU skacze do instrukcji JME, która ma takie same ‘n’, a instrukcje pomiędzy ‘JMP n’ i ‘JME n’ nie są wykonywane.
- Instrukcja ‘JMP n’ powinna pasować tylko do jednej instrukcji ‘JME n’. Powielanie ‘JME n’ jest niedozwolone. Natomiast, powielanie instrukcji ‘JMP n’ jest możliwe.



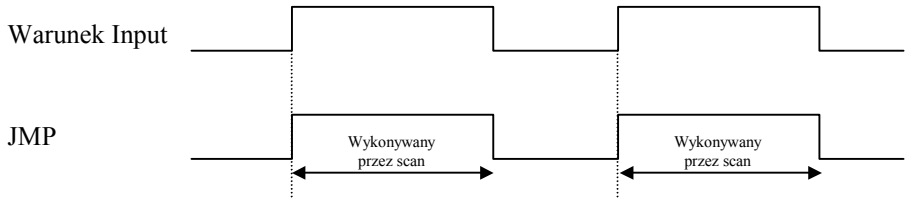
Błąd



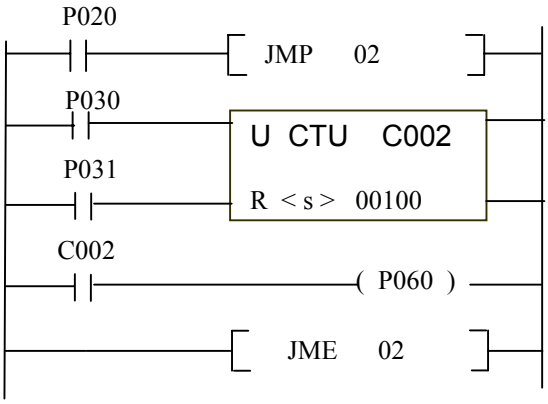
Nie ma błędu

- Instrukcja ‘JMP n’ bez odpowiadającej jej instrukcji ‘JME n’ spowoduje powstanie program error. Jeżeli JME lub JMP znajduje się wewnątrz pętli (podprogram, FOR ~ NEXT lub procedura interrupt), to powstanie błąd operacji, gdy instrukcja JMP jest efektywna. (Szczegóły patrz rozdz. 2.7.1)

- Warunki wykonywania



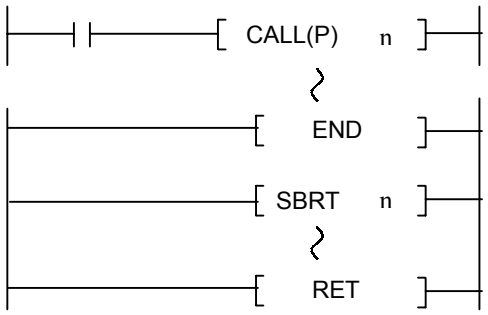
2) Przykład programu
- Program opuszcza operację ring counter pomiędzy ‘JMP 2’ i ‘JME 2’, gdy P020 jest ON.



5.13.2 CALL, CALLP, SBRT, RET

CALL / SBRT (Subroutine)	FUN(014) CALL FUN(015) CALLP FUN(016) SBRT FUN(004) RET	Stosowane w CPU	Wszystkie CPU

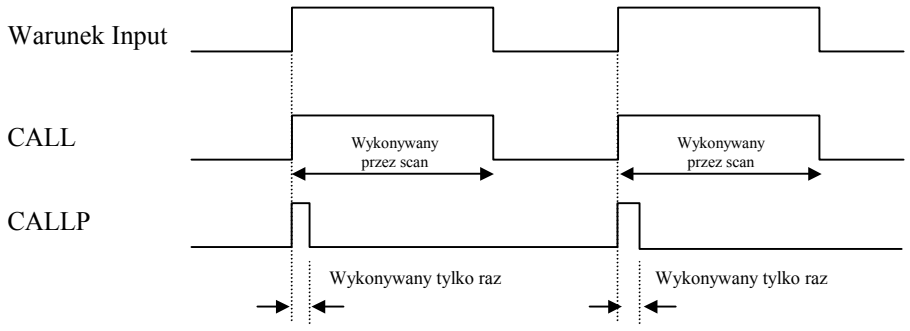
Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
CALL(P) SBRT	n											O	1			
RET																



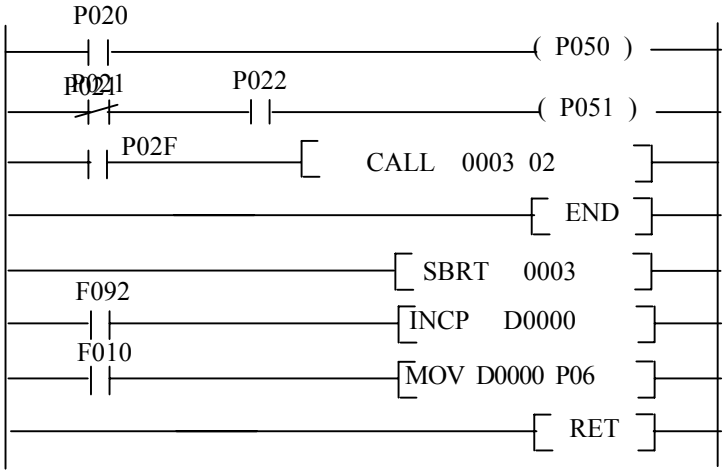
Nastawa operandu

n	K10S1 : 0 ~ 15 K10S/K30S/K60S : 0 ~ 63 K200S/K300S/K1000S : 0 ~ 127

1) Funkcje
- Gdy warunek input jest ON, stopuje sekwencję programu i wykonuje odpowiadający podprogram wskazany przez wskaźnik ‘n’. Po zakończeniu podprogramu następuje podjęcie wykonywania sekwencji programu w miejscu następującego po instrukcji ‘CALL n’ kroku(step).
- Wielokrotne zagłębianie instrukcji CALL(P) jest dozwolone do poziomu 64.
- Zakres wskaźnika ‘n’ różni się w zależności od typu CPU. (Patrz rysunek powyżej)
- Jeżeli instrukcja ‘CALL(P) n’ nie posiada odpowiadającej jej instrukcji ‘SBRT’ o tym samym wskaźniku ‘n’, to wystąpi błąd instrukcji.
- Warunki wykonywania



2) Przykład programu



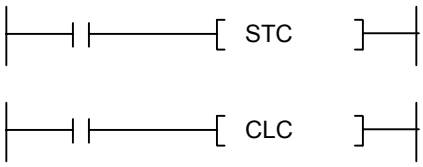
5.14 Instrukcje Flag

5.14.1 STC, CLC

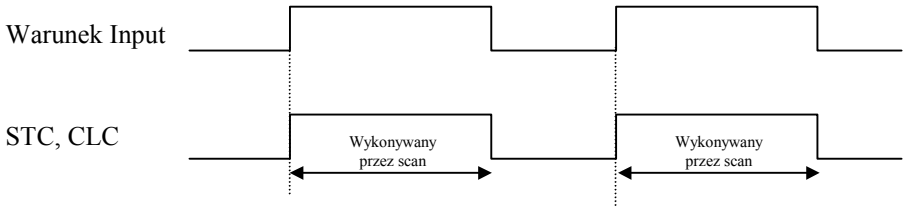
STC, CLC (Set / Reset the carry flag)	FUN(002) STC FUN(003) CLC

Stosowane w CPU	Wszystkie CPU

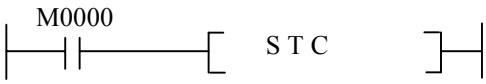
Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
STC CLC												1			O



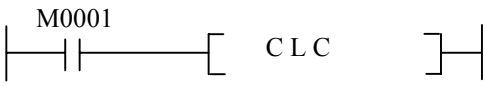
1) Funkcje
- STC : Ustawia falgę carry (F112) na ON, gdy warunek input zostanie ustawiony.
- CLC : Resetuje falgę carry (F112) na OFF, gdy warunek input zostanie ustawiony.
- Warunki wykonywania



2) Przykład programu
- Program ustawia flagę carry (F112), gdy M0000 jest ON.



- Program resetuje flagę carry (F112), gdy M0001 jest ON.



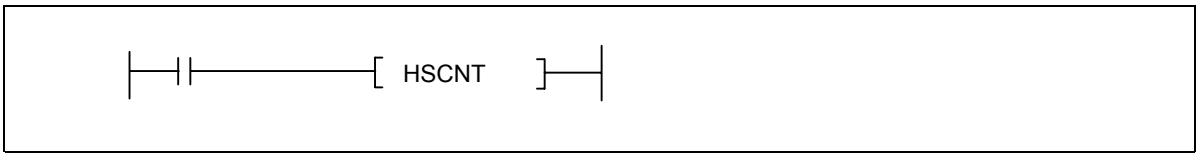
5.15 Instrukcje High speed counter

5.15.1 HSCNT

HSCNT (Enable high speed counter)	FUN(210) HSCNT

Stosowane w CPU	K10S1 / K10S K30S / K60S

Instrukcje	Dostępne Obiekty											Step	Flaga		
	M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
HSCNT												1			

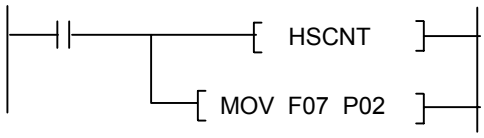


1) Funkcje
- Umożliwia szybkie liczenie, gdy warunek input jest ON.
- Po otrzymaniu zezwolenia liczenia, licznik pracuje zgodnie z nastawionymi parametrami.
- Gdy warunek input jest OFF, liczenie jest resetowane.
- Instrukcja HSCNT nie może być użyta jednocześnie z instrukcją HSC w tej samej sekwencji programu.
- Specyfikacja szybkiego licznika

Pozycja	Zawartość	
Liczba wejść	1 punkt	
Faza	1 faza	
Szybkość liczenia	8kpps(kHz)	
Zakres liczenia	0 ~ hFFFF (16 bits)	
Nastawy	Poziom	Max. 20 level can be set up. (0 ~ 19)
	Dane	Nastawa danych (16 bits)
	SET	Bity do ustawienia ON (8 bits)
	RESET	Bity do ustawienia OFF (8 bits)
Wartość bieżąca	F140 ~ F14F (16 bits)	
Nastawa	F150 ~ F15F (16 bits)	
Obiekt Output	F070 ~ F077 (8 bits)	

2) Przykład programu

- Program wystawia szybki licznik do słowa P002.



< Parametry ustawione przez KGL-WIN>

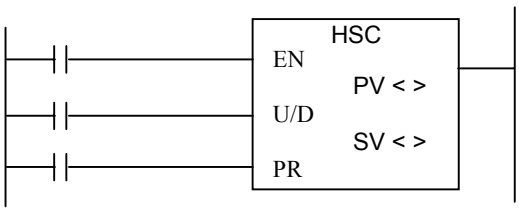
a) Gdy warunek input zostaje ustawiony na ON, to wartość bieżąca jest zapamiętywana w F14, a nastawa kroku 0 jest zapamiętywana w F15.
b) Gdy wartość bieżąca osiągnie nastawę #0, F070 ~ F077 jest ustawiane / resetowane zgodnie z nastawionym parametrem, a F15 jest odnawiane jako nastawa kroku 1(step 1).
c) Gdy wartość bieżąca osiągnie nastawę ostatniego kroku (step 5 w tym przykładzie), F15 jest odnawiany jako nastawa kroku 0, a wartość bieżąca (F14) jest zerowana.
d) Jeżeli warunek input zostanie zresetowany na OFF, wartość bieżąca i output HSC (F070 ~ F077) jest zerowana.

5.15.2 HSC

HSC (High speed counter)	FUN(215) HSC

Stosowane w CPU	K10S1 / K10S K30S / K60S

Instrukcje		Dostępne Obiekty											Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
HSC	PV	O	O	O	O	O	O	O		O	O	O	7/9/11			
	SV	O	O	O	O	O	O	O		O	O	O				



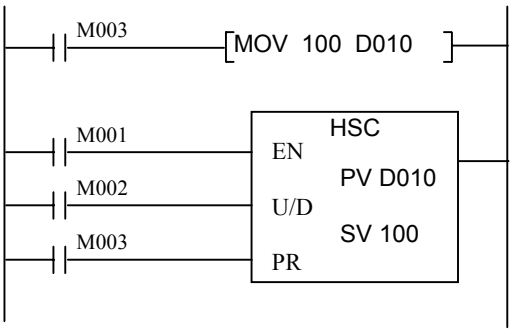
Nastawa operandu

PV	Wartość zadana
SV	Wartość nastawy

1) Funkcje
- Instrukcja HSC nie może być używana razem z instrukcją HSCNT w tym samym programie. Tylko jeden z nich może być użyty w sekwencji programu.
- 32-bitty, up / down high speed counter. (HSCNT : 16-bitów, up-counter)
- Jeżeli wartość bieżąca jest równa lub większa niż SV, to HSC wystawia bit (F070) .
- Wartość bieżąca nie może być zmieniona przez użytkownika.
- Wartość bieżąca jest zapamiętywana w F14 (niższe słowo) i F15 (wyższe słowo).
- Gdy instrukcja HSC jest używana, to nastawy high speed counter są ignorowane.
- Opis operandów
- EN : Styk zezwolenia high speed counter
  a) U/D : pracuje jako up counter gdy U/D jest 0, i jako down counter gdy U/D jest 1.
  b) PR : Jeżeli input PR ustawi się na ON, to wartość bieżąca jest zmieniana na wartość zadaną (PV).
  c) PV : Wartość zadana. Jeżeli PV jest wskazana jako obiekt [A], to PV jest zawartością [A+1, A].
  d) SV : Nastawa. Jeżeli SV jest wskazana jako obiekt [A], to SV jest zawartością [A+1, A].

2) Przykład programu

- M1 : HSC reset, M2 : U/D input (0 = up, 1 = down), M3 : Zmień wartość bieżącą na PV
- Jeżeli wartość bieżąca jest taka sama lub większa niż SV, to bit F070 ustawi się na ON.



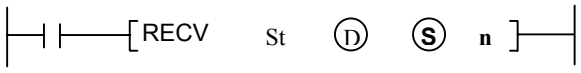
5.16 Instrukcje RS-485 communication

5.16.1 RECV (K10S1)

RECV (Receive data)	FUN(158) RECV

Stosowane w CPU	K10S1 / K10S K30S / K60S

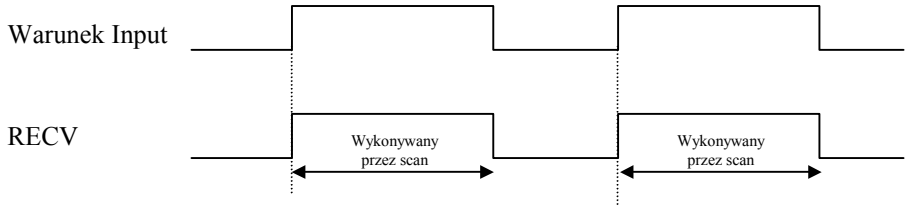
Instrukcje		Dostępne Obiekty										Step	Flaga		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
RECV	St	O	O	O	O	O	O	O		O	O	O	9	O		
	Ⓓ	O	O	O	O	O	O	O		O	O					
	Ⓢ	O	O	O	O		O	O		O						
	n	O	O	O	O	O	O	O		O	O	O				



Nastawa operandu

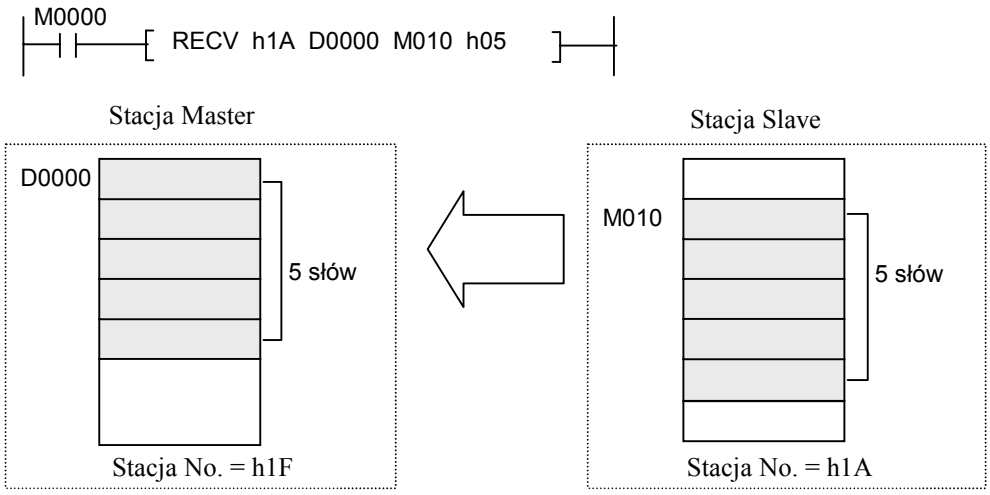
St	Numer stacji slave do odczytu
Ⓓ	Adres początku obiektu stacji master pod którym zostaną zapamiętane otrzymane dane
Ⓢ	Adres początku obiektu stacji slave pod którym zapamiętane są dane do wysłania
n	Liczba słów do odczytu (n : h00 ~ h1F)

1) Funkcje
- Czyta ‘n’ słów z obiektu wskazanego jako [S] stacji slave (Numer stacji = ‘st’) i zapamiętuje przeczytane dane w bloku który zaczyna się obiektem wskazanym jako [D] stacji master.
- Instrukcja RECV może być użyta tylko ze stacją master (numer stacji = h1F).
- Warunki wykonywania



2) Przykład programu

- Program czyta 5 słów z M010 stacji slave (numer stacji = h1A) i zapamiętuje dane do D0000 ~ D0004 stacji master, gdy M0000 ustawi sięna ON.



5.16.2 SEND (K10S1)

SEND (Send data)	FUN(159) SEND

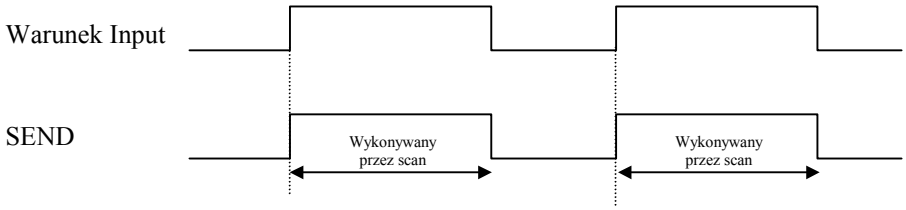
Stosowane w CPU	K10S1 / K10S K30S / K60S

Instructions		Available Device										Step	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SEND	St	O	O	O	O	O	O	O		O	O	O	9	O		
	Ⓢ	O	O	O	O	O	O	O		O	O					
	Ⓓ	O	O	O	O		O	O		O						
	n	O	O	O	O	O	O	O		O	O	O				

Operand setting

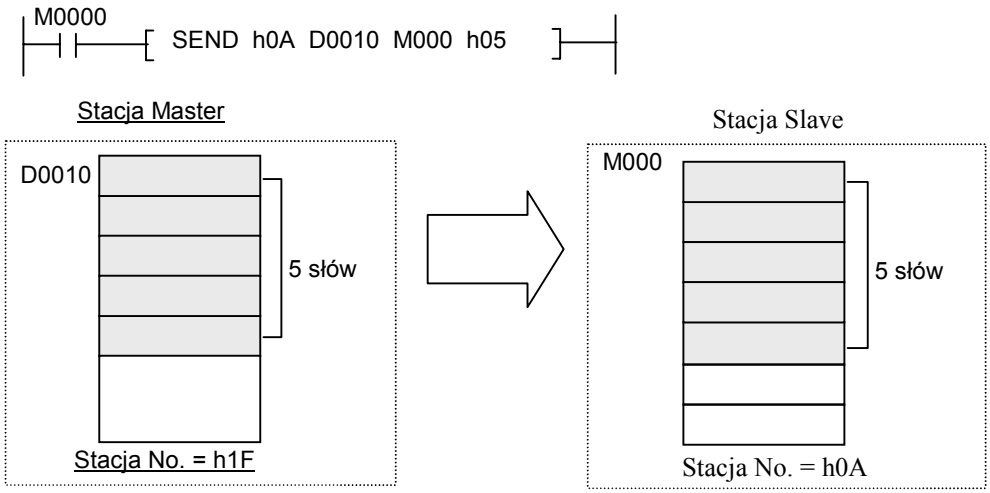
St	Station number of slave station to which data to be written
Ⓢ	The start address of device of master station at which the source data is stored
Ⓓ	Start address of device of slave station at which the sent data is stored
n	Numbers of word to be read (n : h00 ~ h1F)

1) Funkcje
   - Wysyła ‘n’ słów z obiektu wskazanego jako [S] stacji master i zapamiętuje przeczytane dane w bloku który zaczyna się obiektem wskazanym jako [D] stacji slave (Numer stacji = ‘st’).
   - Instrukcja SEND może być użyta tylko ze stacją master (numer stacji = h1F).
   - Warunki wykonywania



2) Przykład programu

- Program wysyła 5 słów z D0010 stacji master i zapamiętuje dane w M0000 ~ M0004 stacji slave (numer stacji = h0A), gdy M0000 ustawi się na ON.



5.16.3 MODBUS (K80S)

MODBUS (Send data)		Stosowane w CPU	K80S

Instructions		Available Device										Step	Flag		
		M	P	K	L	F	T	C	S	D	#D	Integer		Error (F110)	Zero (F111)	Carry (F112)
SEND	S1	O	O	O	O	O	O	O		O	O	O	7	O		
	S2	O	O	O	O	O	O	O		O	O					
	S3	O	O	O	O		O	O		O						

Operand setting

S1	Adres tablicy zaw. nr stacji i,kod funkcji, adres i liczbę komórek
S2	Adres komórki zawierającej wyniki operacji (przesłane dane)
S3	Adres komórki zaw. status operacji

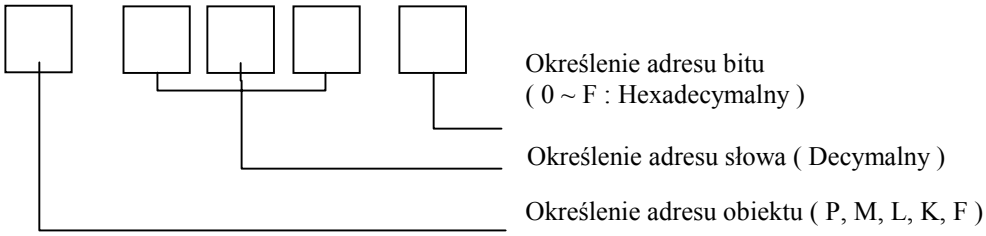
Dodatek

A.1 Konfiguracja pamięci

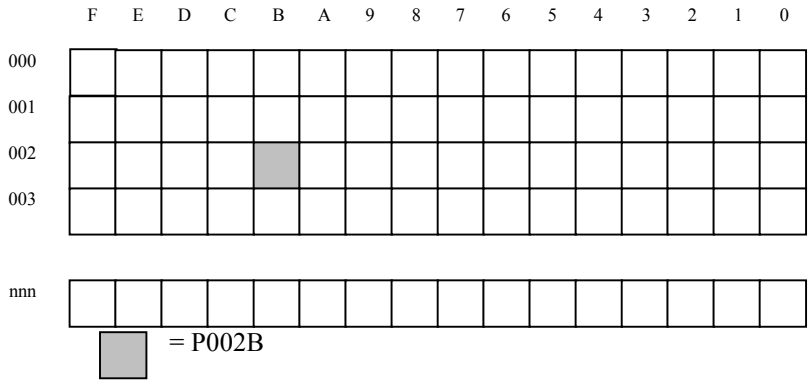
A.1.1Obiekty pamięci bitowej

Bitowy obiekt pamięci jest obszarem pamięci który może być odczytywany / zapisywany z rozdzielczością jednego bitu. Obszary P, M, L, K, F są obiektami pamięci bitowej. Jednakże, obiekty pamięci bitowej mogą być używane jako obszary obiektów słowowych.

< Obraz obiektów pamięci bitowej >



< Struktura pamięci obiektu pamięci bitowej >

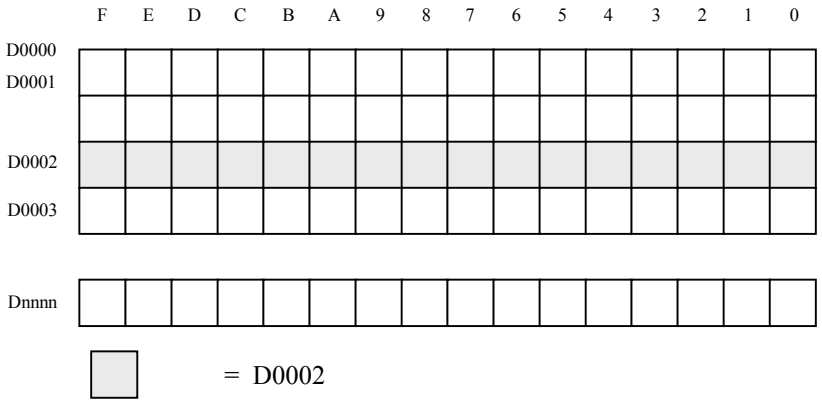


A.1.2 Obiekt pamięci Bit / Word (timer & counter)

Obszary pamięci timera i countera składają się z 3 części – bit output, słowo wartości bieżącej i słowo nastawy. Gdy obiekt T lub C jest używany jako operand instrukcji bitowej, to instrukcja ma wpływ na stan bitu output timera lub countera. Jeżeli obiekt T lub C jest używany jako operand w instrukcji słowowej, to instrukcja ma wpływ na wartość bieżącą słowa. Nastawa nie może być zmieniona przez użytkownika.

A.1.3 Obiekt pamięci word(słowowej)

Obiekt D jest kontrolowany przez słowo. Aby kontrolować obiekt D bitowo, należy użyć instrukcji specjalnych takich jak BLD, BAND, BOR, etc. Należy jednak pamiętać, że obiekt D nie może być użyty jako operand takich instrukcji bitowych jak LOAD, OUT, etc.

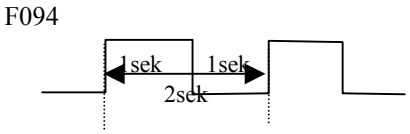
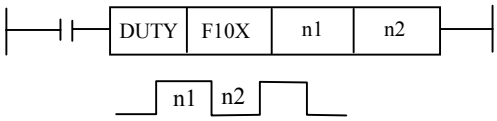


A.2 Przekaźniki specjalne

A.2.1 K10S1 / K10S / K30S / K60S

1) Obiekt F

Przekaźnik	Nazwa	Opis	
F000	Flaga Run	Ustaw gdy PLC jest w modzie RUN	
F001	Flaga PGM	Ustaw gdy PLC jest w modzie PGM	
F002	Flaga Pause	Ustaw gdy PLC jest w modzie Pause	
F007	Mod EPROM	Ustaw gdy PLC jest w modzie EPROM run	
F010	Zawsze ON	Używany jako przekaźnik sztuczny lub inicjator przy pisaniu programów	
F011	Zawsze OFF		
F012	1 scan ON	ON podczas pierwszego scan'u po przejściu modu PGM→RUN	
F013	1 scan OFF	OFF podczas pierwszego scan'u po przejściu modu PGM→RUN	
F014	Zmienia stan za każdym scan'em	Powtarza set/reset gdy PLC jest w modzie RUN	
F020 ~ F02F	Informacja o Communication Error	● Przepełniony - tylko do instrukcji SEND, RECV ● Starszy byte: Numer stacji w której nastąpił error Młodszy byte: error kodu ● Kod error – przekroczenie czasu:h20 ● Brak error:h000	
F030	Poważny błąd	Ustawiany gdy jest wewnętrzny błąd ROM, błąd 24V	
F031	Lekki błąd	Ustawiany gdy jest WDT error, program error, error kombinacji I/O, brak END/RET error	
F03A	Flaga RTC data error	Ustawiany gdy został wykryty error RTC data	
F040 ~ F045	Kombinacja I/O error	Ustawiany gdy wkładany / wyjmowany jest moduł I/O podczas pracy lub niepoprawne połączenie	
F050 ~ F05F	Kod error	● h0000:Brak error ● h0023:Kod error ● h0014:I/O error ● h0024:Brak END error ● h0021:Parametr error ● h0025:Brak RET error	
F060 ~ F06F	Numer step gdy powstanie error	● Zapamiętywany jest step No.(numer kroku) w którym wystąpił program error ● W przypadku błędu instrukcji branch, zapamiętywany jest numer kroku przeznaczenia	
F070 ~ F077	Rejestr HSC	Obszar szybkiego licznika	
F080 ~ F08F	Model PLC	● K10S:h0031 ● K60S:h0036 ● K30S:h0033 ● K100S:h0035	Starszy byte: No.stacji PLC Młodszy byte: model PLC

Przekaźnik	Nazwa	Opis
F090	Zegar 20msek	Przekaźniki te powtarzają On/Off z ustalonym okresem i są generowane tylko w modzie RUN . 
F091	Zegar 100ms	
F092	Zegar 200ms	
F093	Zegar 1sek	
F094	Zegar 2sek	
F095	Zegar 10sek	
F096	Zegar 20sek	
F097	Zegar 1minuta	
F100 ~ F107	Zegar zdefiniowany przez użytkownika F100:clock0 ~ F107:clock7 (zegar7)	Przekaźniki powtarzają ON/OFF bazując na czasie scan. (Stan początkowy = OFF) 
F110	Flaga błędu arytmetycznego	Ustawiana gdy wystąpi błąd arytmetyczny podczas operacji
F111	Flaga Zero	Ustawiana gdy wynik jest zero
F112	Flaga Carry	Ustaw gdy ma miejsce Carry lub Borrow jako wynik operacji
F11A	Flaga ON wysyłania	Przekaźniki te pokazują status komunikacji Gdy używane są instrukcje DIN, DOUT
F11C	Flaga ON odbioru	
F11E	Flaga Odbiór zakończony	
F11F	Flaga błąd komunikacji	● DIN, DOUT:Ustawiona gdy zajdzie błąd ‘czas minął’ ● SEND, RECV:Ustawiona gdy zajdzie błąd ‘czas minął’ lub został wykryty komunikat NAK.
F120	<	Przekaźniki te ustawiane są zgodnie z wynikiem instrukcji Compare(porównania) (CMP, CMPP, DCMP, DCMPP)
F121	≤	
F122	=	
F123	>	
F124	≥	
F125	≠	
F130 ~ F135	Status I/O	Każdy przerzutnik pokazuje czy odpowiadający mu moduł I/O jest włożony czy nie.
F140 ~ F14F	Wartość bieżąca HSC	HSCNT : Wartość bieżąca szybkiego licznika (HSC) jest zapamiętana. HSC : Młodsze słowo wartości bieżącej szybkiego licznika (HSC) jest zapamiętane.
F150 ~ F15F	HSC wartość zadana	HSCNT : Wartość bieżąca szybkiego licznika (HSC) jest zapamiętana HSC : Starsze słowo wartości zadanej szybkiego licznika (HSC) jest zapamiętane .

2) Inne przekaźniki specjalne

Obszar	Opis		Uwagi
M310	RTC Możliwa zmiana ustawienia przez użytkownika		Gdy M310 jest ON, dane RTC mogą być zmieniane jako dane D249~D252
L12~ L15	Dane RTC		
D240	Jednostka	Wejście danych A/D Ch.0	Tylko typy

D241	analogowa #1	Wejście danych A/D Ch.1	K30S-A / K60S-A
D242		Wyjście danych A/D	
D243	Jednostka analogowa #2	Wejście danych A/D Ch.0	
D244		Wejście danych A/D Ch.1	
D245		Wyjście danych A/D	
D247	Obszar modu ustawiania High Speed Counter		
D248	Wartość ‘czas minął’ komunikacji RS485		O/S V1.5 lub później
D249~ D252	RTC Możliwa zmiana ustawienia przez użytkownika		Seria MK-S Ver1.3 lub używany później gdy punkt M310 jest takiego samego formatu jak L12~L15
D253	Czas scan’u bieżącego		
D254	Czas scan’u Minimum		
D255	Czas scan’u Maximum		

A.3 Lista instrukcji

Numer Funkcji	0	1	2	3	4	5	6	7	8	9
00x	NOP	END	STC	CLC	RET	MPUSH	MLOAD	MPOP	STOP	CLE
01x	MCS	MCSCLR	JMP	JME	CALL	CALLP	SBRT	D	DNOT	
02x	INC	INCP	DINC	DINCP	DEC	DECP	DDEC	DDECP	LD=*	LDD=*
03x	ROL	ROLP	DROL	DROLP	ROR	RORP	DRDR	DRDRP	LD>*	LDD>*
04x	RCL	CMPP	DRCL	DRCLP	RCR	RCRP	DRCR	DRCRP	LD<*	LDD<*
05x	CMP	BCDP	DCMP	DCMPPP	TCMP	TCMPP	DTCMP	DTCMPP	LD>=*	LDD>=*
06x	BCD	WSFTP	DBCD	DBCDP	BIN	BINP	DBIN	DBINP	LD<=*	LDD<=*
07x	WSFT	MOVP	MULS*	MULSP*	BSFT	BSFTP	DMULS*	DMULSP*	LD<>*	LDD<>*
08x	MOV	GMOVP	DMOV	DMOVP	CMOV	CMOVP	DCMOV	DCMOV	DIVS*	DIVSP*
09x	GMOV	BMOVP	FOMV	FOMVP	AND=*	ANDD=*	AND>*	ANDD>*	AND<*	ANDD<*
10X	BMOV	ADDP	XCHG	XCHGP	DXCHG	DXCHGP	AND>=*	ANDD>=*	ANDD>=*	ANDD>=*
11X	ADD	ADDP	DADD	DADDP	SUB	SUBP	DSUB	DSUBP	AND<>*	ANDD<>*
12X	MUL	MULP	DMUL	DMULP	DIV	DIVP	DDIV	DDIVP	DDIVS	DDIVSP
13X	ADDB	ADDBP	DADDB	DADDBP	SUBB	SUBBP	DSUBB	DSUBBP		
14X	MULB	MULBP	DMULB	DMULBP	DIVB	DIVP	DDIVB	DDIVBP		
15X	WAND	WANDP	DWAND	DWAND	WOR	WORP	DWIR	DWORP	RECV ■	SEND ■
16X	WXOR	WXORP	DWXOR	DWXOR	WXNR	WXNRP	DWXNR	DWXNR	RCV*	SND*
17X	BSUM	BSUMP	DBUSM	DBUSMP	SEG	SEGP	ENCO	ENCOP	DECO	DECOP
18X	BSUM	FILRP	DFILR	DFILRP	FILW	FILWP	DFILW	DFILWP	OR=*	ORD=*
19X	ASC	ASCP	UNI	DSI	DIS	DISP	OR>*	ORD>*	OR<*	ORD<*
20X	IORF*	IORFP*	WDT*	WDTP*	FALS*	DUTY	FOR*	NEXT*	OUTOFF	.
21X	HSCNT■	DIN	DINP	DOUT	DOUTP	HSC ■	OR>=*	ORD>=*	OR<=*	ORD<=*
22X	BREAK*	EI*	DI*	BSET*	BRST*	IRET*	TDINT*	INT*	OR<>*	ORD<>*
23X	GET*	GETP*	RGET*	RPUT*	PUT*	PUTP*	BOUT*	SR*	EI*	DI*
24X	NEG*	NEGP*	DNEG*	DNEGP*	READ*	WRITE*	CONN*	STATUS*	BLD*	BLDN*
25X	BAND*	BANDN*	BOR*	BORN*						

* : Dostępny tylko w serii K1000S, K300S, K200S

■ : Dostępny tylko w serii K10S,K10S1, K30S, K60S