



Instrukcja do ćwiczenia

Ćwiczenie nr	25.1.
Temat :	Programowanie cyfrowe Elementy podstawowe
Stanowisko laboratoryjne	Sterownik LOGO
Opracował :	A. Mielewczyk



Instrukcja nr 25.1.

1. Temat ćwiczenia:

Programowanie sterownika Siemens LOGO – bloki funkcyjne.

2. Cel ćwiczenia:

Celem ćwiczenia jest zapoznanie się z podstawowymi elementami w programowaniu FBD i drabinkowym sterownika LOGO i wykonanie symulacji załączania wyjść sterownika za pomocą programu LOGO Soft Comfort v8.

3. Zakres wymaganych wiadomości:

- sygnały dwustanowe i ich adresowanie,
- sygnały analogowe i ich adresowanie,
- elementy logiczne,
- programowanie w języku drabinkowym LAD oraz FBD – Function Block Diagram

4. Przebieg ćwiczenia:

Zapisać program załączania wyjść cyfrowych według wskazanych przykładów.

5. Pomoce i urządzenia:

Sterownik Siemens LOGO, program LOGO Soft Comfort v8.

6. Treść sprawozdania:

Część wstępna, program w wersji LAD i FBD.

WPROWADZENIE

Programowanie sterownika w Logo Soft Comfort v8 oraz tworzenie funkcji użytkownika

Prosty sterownik Siemens LOGO przedstawiono na rys. 1.



Rys. 1 Sterownik Siemens LOGO

Programowanie FBD - Function Block Diagram

Bloki to funkcje używane w celu określenia sposobu konwersji sygnału wejściowego na wyjściowy. Blokami są bramki logiczne (AND, NAND, OR, NOR, XOR itd.), przerzutniki, liczniki, rejestry oraz inne, bardziej rozbudowane funkcje. Pojedynczy schemat może zawierać do 400 bloków, ale pod warunkiem, że rozmiar otrzymanego w wyniku ich umieszczenia na schemacie programu wynikowego nie przekroczy wielkości pamięci wyrażonej w bajtach. Oprócz samych funkcji, w pamięci przechowywane są również robocze wartości programu (np. stan licznika godzin) oraz dane bloków pod warunkiem, że została uaktywniona opcja podtrzymania pamięci. W Logo! dostępne są następujące zasoby:



- pamięć programu: 3800 bajtów,
- maksymalna liczba bloków: 400,
- maksymalna liczba podtrzymywanych zmiennych: 250.

Aby wyświetlić informację o dostępnej pamięci, należy wykonać następujące czynności:

- Przełączyć Logo! w tryb programowania.
- Umieścić kursor w pozycji „Edit” i nacisnąć OK.
- Umieścić kursor w pozycji „Memory” i nacisnąć OK.

Na ekranie Logo! pojawi się komunikat „Free Memory” zawierający informację o zajętości pamięci programu, liczbie używanych bloków oraz podtrzymywanych zmiennych.

Umieszczenie bloku na schemacie powoduje automatyczne nadanie mu numeru porządkowego. Numeracja ułatwia orientację oraz wyszukiwanie błędów.

Każdy blok można dołączyć do wejścia bieżącego (wskazanego) posługując się jego numerem. Dzięki temu można w programie wykorzystywać wyniki działania bloków pośrednich. Maksymalnie 100 blokom można nadać nazwy składające się z co najwyżej 8 liter. Dzięki takiemu zabiegowi ułatwia się użytkownikowi orientację oraz modyfikację odpowiednich parametrów za pomocą klawiatury sterownika.

Ustanawianie połączenia Ethernet

Połączenie z nowym Logo! odbywa się za pomocą Ethernetu – służy do tego protokół TCP/IP v. 4. Komputer PC, na którym jest tworzony program przesyłany następnie do pamięci sterownika musi być w tej samej sieci, co Logo! Oznacza to, że maska podsieci sterownika i komputera PC muszą być takie same, natomiast adresy IP leżeć w puli dostępnej dla danej maski. Odpowiedni adres należy wpisać ręcznie korzystając z menu sterownika lub oprogramowania IDE – w naszym wypadku jest to Logo Soft Comfort (*Tools à Transfer à Configure Network Address*). Jednak przy pierwszym połączeniu sterownik i komputer PC muszą jakoś odnaleźć się w sieci, natomiast parametry protokołu TCP/IP prawie na pewno będą różne. Dlatego za pierwszym razem należy sprawdzić, jaki adres ma nasz komputer PC, a następnie ustawić ręcznie adres za pomocą menu sterownika Logo! Komputer i Logo! muszą pracować w tej samej sieci oraz adres sterownika musi być znany, wpisany i ustalony „ręcznie”.

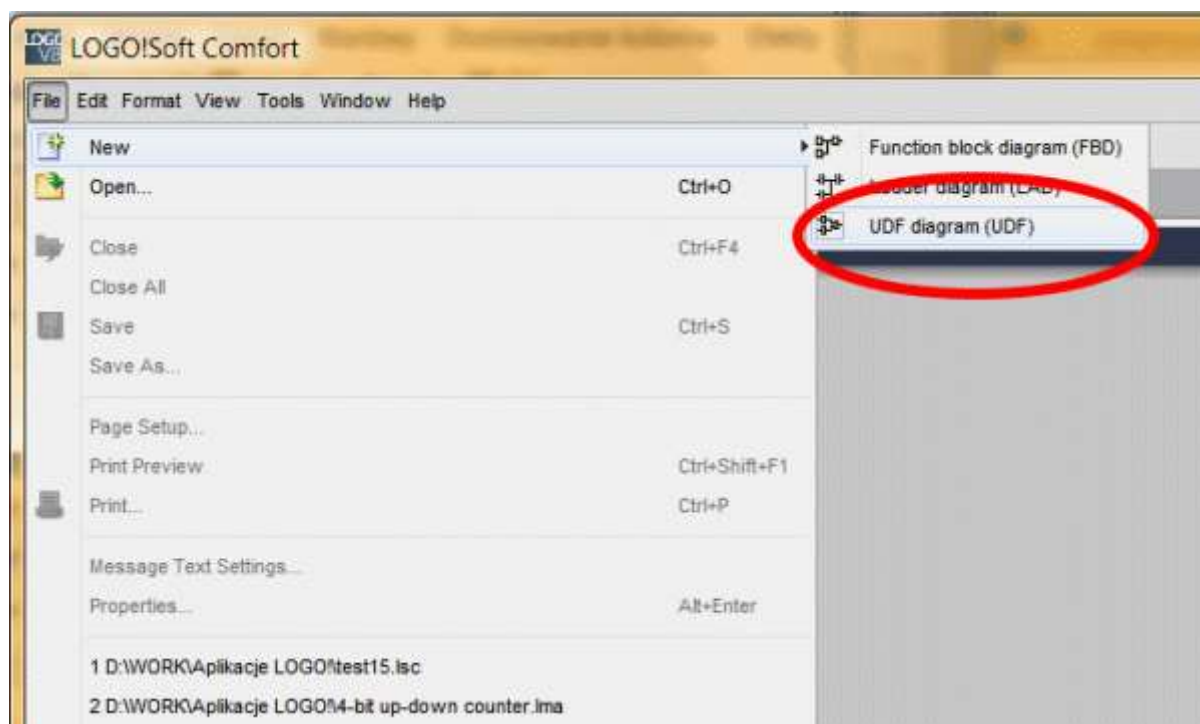
Tworzenie bloków użytkownika – UDF

Przydatną cechą środowiska Logo Soft Comfort jest możliwość tworzenia funkcji użytkownika, to znaczy – własnych bloków FBD, które można wielokrotnie umieszczać na schematach. Ich reprezentacja graficzna jest bardzo podobna do bloków utworzonych przez producenta oprogramowania, dzięki czemu nawet spory blok funkcjonalny nie będzie zajmował dużo miejsca na schemacie, który dzięki temu stanie się przejrzysty.

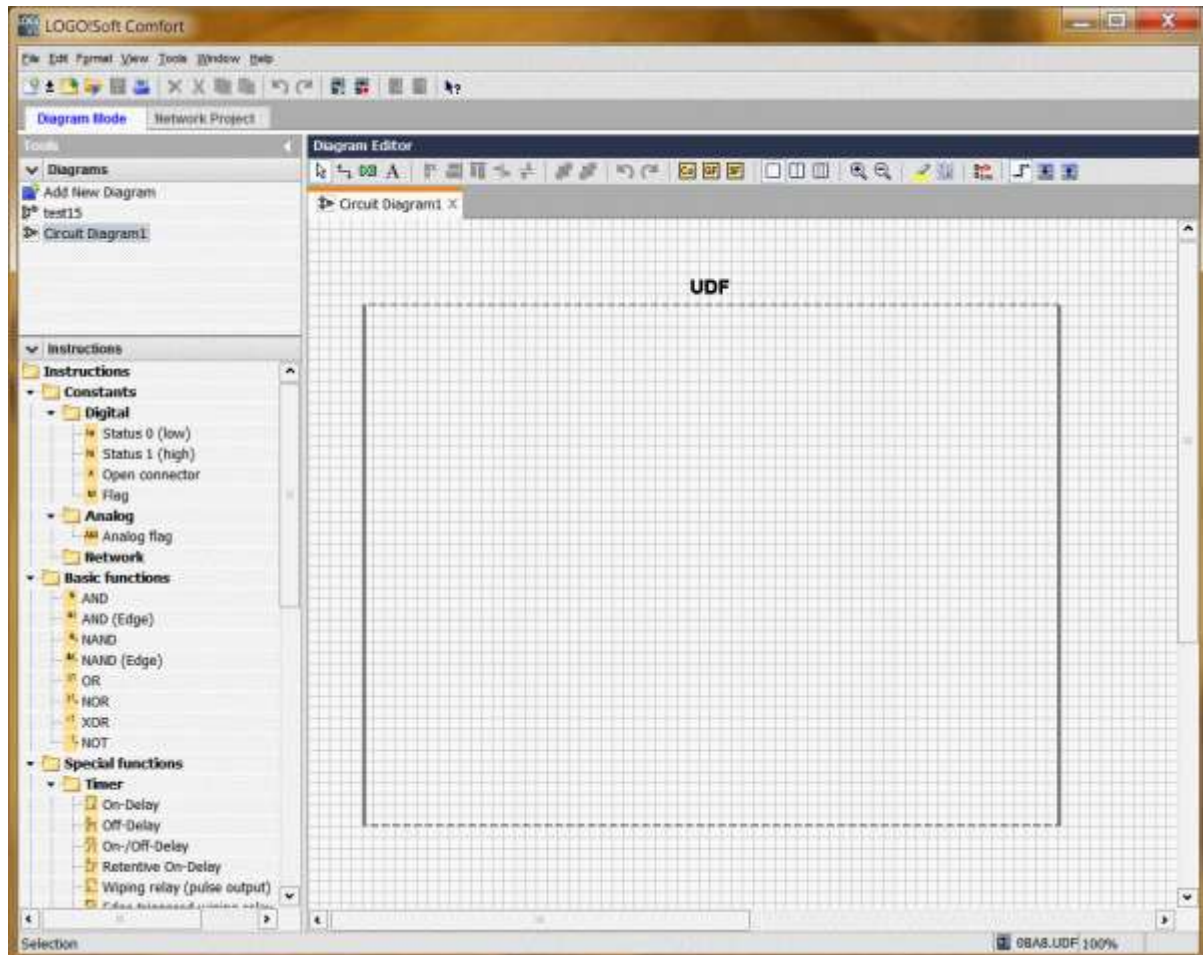
Środowisko Logo Soft Comfort umożliwia użytkownikowi tworzenie własnych bloków funkcjonalnych, które dla odróżnienia od bloków FBD, noszą nazwę UDF, co jest akronimem pochodzącym od słów *User Defined Function* (funkcje definiowane przez użytkownika). Po umieszczeniu na schemacie, blok UDF przypomina zwykły blok FBD. Istotnym ograniczeniem dla bloków UDF, które trzeba brać pod uwagę, są:

- Maksymalna liczba wejść wynosząca 8.
- Maksymalna liczba wyjść wynosząca 4.

Tworzenie bloku UDF rozpoczyna się od wybrania z menu programu *File -> New -> UDF Diagram (UDF)*, jak pokazano na rys. 2. W przestrzeni roboczej ekranu zostanie otwarte okno (rys. 3), w którym można umieścić niezbędne komponenty. Jeśli jest ono za małe, to można je powiększyć korzystając z menu podręcznego i pozycji *Properties*, podobnie jak dla arkusza schematu podając wielokrotność poziomą i pionową wielkości ramki.



Rys. 2 Tworzenie nowego bloku UDF

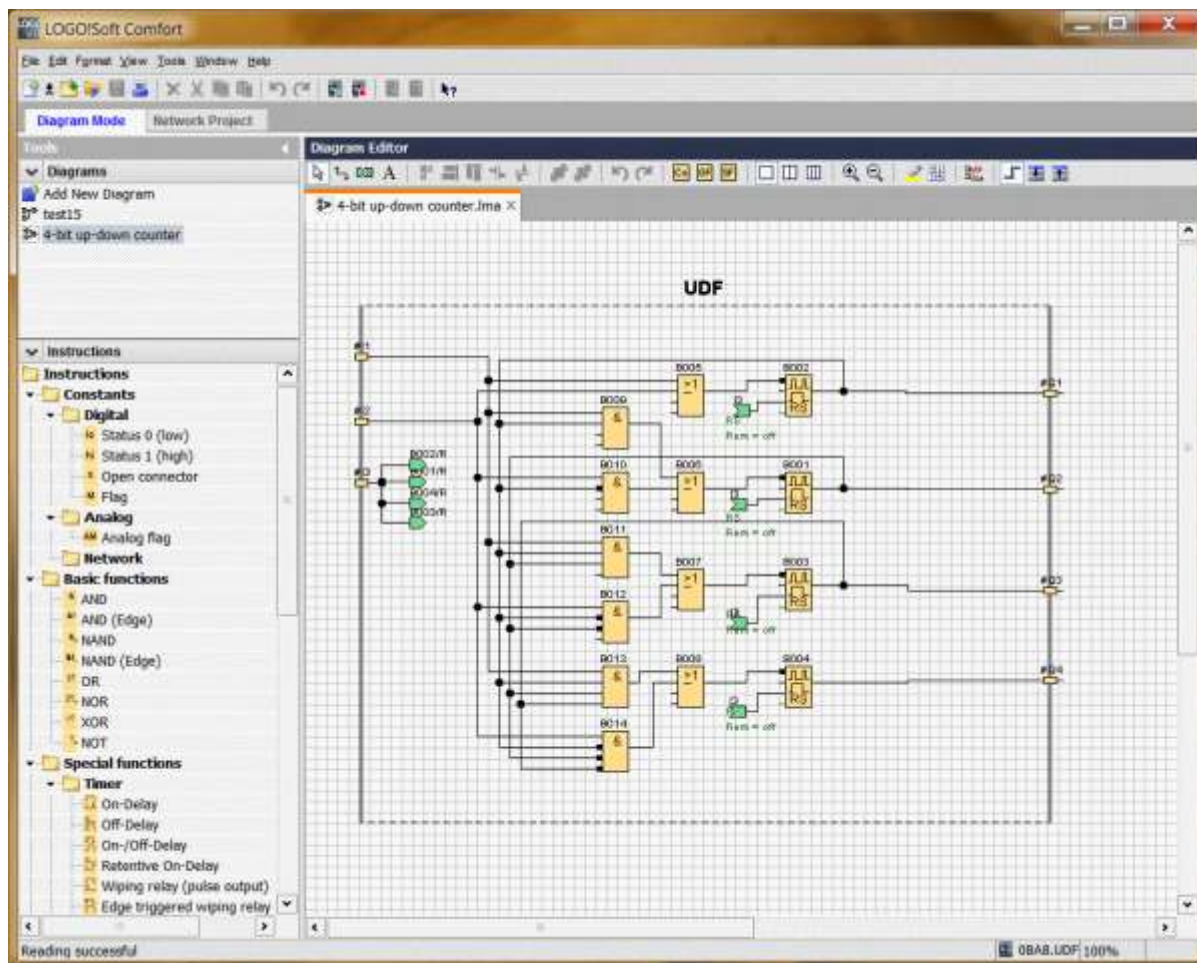


Rys. 3 Okno edycji bloku UDF (tworzeniu programu w obrębie ramki)

Dla przykładu, wykonano blok licznika dwukierunkowego z wejściami zliczającymi „w górę” (Up), „w dół” (Dn) oraz zerującym (Clr). Do jego wykonania wystarczy pojedyncza ramka. Bloki FBD, z których składa się licznik są następujące:

- Bramki OR.
- Bramki AND.
- Przerzutniki Pulse Relay (przerzutnik dwustanowy, w którym zmiana stanu następuje na zboczu narastającym impulsu doprowadzonego do wejścia Trg; ma wejścia ustawiające Set, zerujące Reset oraz pojedyncze wyjście Q).

Schemat licznika dwukierunkowego złożonego z bloków FBD pokazano na rys. 4. Schemat powstał na bazie schematu układu scalonego SN74193 firmy Texas Instruments.

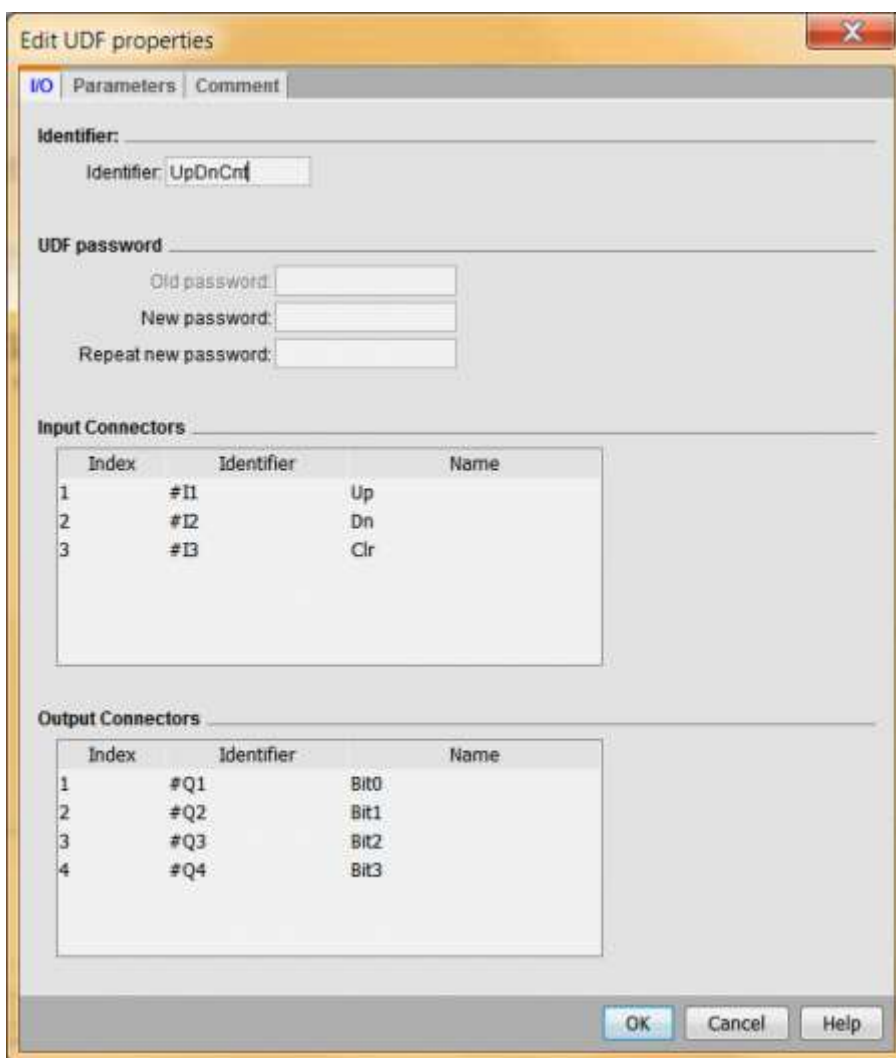


Rys. 4 Schemat licznika dwukierunkowego złożonego z bloków FBD

Licznik ma dwa wejścia impulsów zliczanych. Doprowadzenie przebiegu na wejście #1 powoduje zliczanie „w górę”, na wejście #2 „w dół”. Kropka na wejściu *Trg* przerzutnika oznacza, że przebieg wejściowy jest zanegowany i że przerzutnik będzie reagował na zbocze opadające. Kropkę ustawia się dwukrotnie klikając myszką na symbolu doprowadzenia. W związku z tym, że blok FBD ma tylko jedno wyjście – nazwijmy je umownie *Q*, to sygnał *!Q* (zanegowany *Q*) można uzyskać albo dołączając inwerter, albo stawiając kropki na wejściach bloków, do których jest on doprowadzony. Poziom wysoki na wejściu #3 powoduje wyzerowanie licznika, ponieważ to wejście jest dołączone bezpośrednio do wejść zerowania przerzutników. Aby zachować zgodność z pierwowzorem i aby takie liczniki można było łączyć kaskadowo należałoby jeszcze dodać co najmniej wyjście przepiętnienia (*CO* – *Carry Output*) i

pożyczki (*BO – Borrow Output*), ale nie pozwala na to ograniczenie liczby wyjść do 4. Jeśli zależałoby nam na łączeniu w kaskady większej liczby liczników, to ze względu na to ograniczenie, musiałyby one być 2-bitowy.

Zatrzymanie kursora nad symbolem wejścia lub wyjścia bloku FBD powoduje wyświetlenie skojarzonej z nim nazwy. Wejścia i wyjścia są ponumerowane i nie mają żadnych nazw opisowych. Aby je nadać, trzeba zatrzymać kursor na obszarze schematu UDF, nacisnąć prawy przycisk na myszy i z menu podręcznego wybrać *Edit UDF Properties...* Zostanie wyświetlone okno pokazane na rys. 5, w którym nie tylko można nazwać wejścia i wyjścia, ale również blok UDF i zabezpieczyć go za pomocą hasła przed nieautoryzowaną edycją.



Edit UDF properties

I/O Parameters Comment

Identifier: UpDnCnt

UDF password

Old password:

New password:

Repeat new password:

Input Connectors

Index	Identifier	Name
1	#I1	Up
2	#I2	Dn
3	#I3	Clr

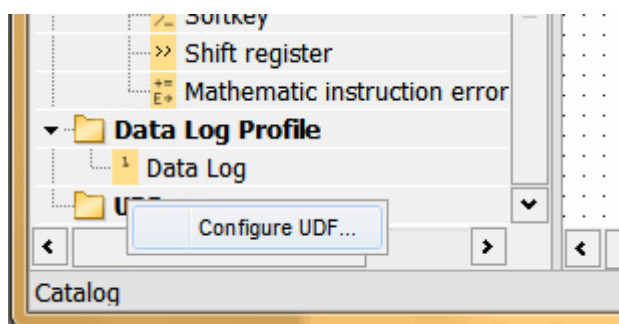
Output Connectors

Index	Identifier	Name
1	#Q1	Bit0
2	#Q2	Bit1
3	#Q3	Bit2
4	#Q4	Bit3

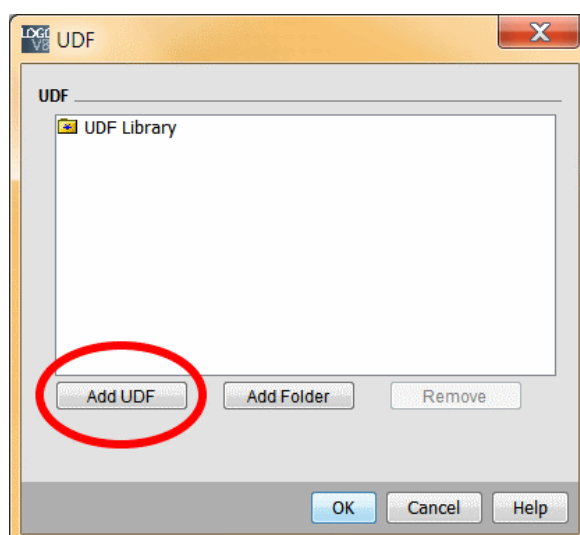
OK Cancel Help

Rys. 5 Okno własności bloku UDF, w którym można nazwać wejścia, wyjścia, blok UDF oraz zmienić wielkość ramki

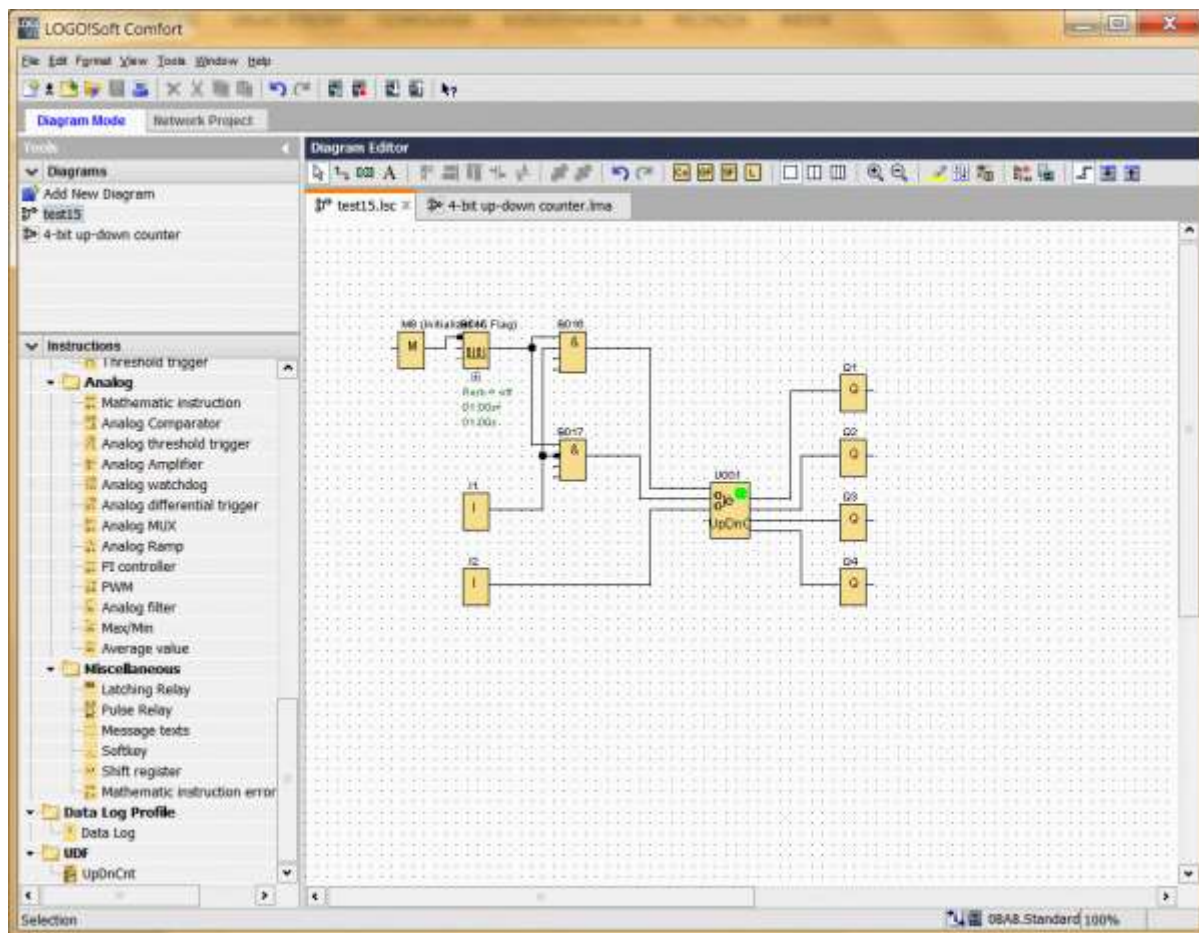
Po ustawieniu własności bloku UDF oraz zapamiętaniu pliku na dysku, należy dodać blok do biblioteki, aby został wyświetlony na drzewie projektu i aby było możliwe umieszczenie go na schemacie. W tym celu najeżdżamy kursorem myszy na pozycję UDF na drzewie projektu, naciskamy prawy przycisk i wybieramy *Configure UDF* (rys. 6). Z otwartego okna menu kontekstowego wybieramy *Add UDF* (rys. 7) i wskazujemy plik utworzonego przez nas bloku (ma on rozszerzenie *.lma*). Po kliknięciu na *OK* utworzona przez nas funkcja zostaje dodana do drzewa, skąd można ją przenosić na schemat jak zwykły blok FBD. Przykładowy schemat utworzony z użyciem omawianego w przykładzie licznika pokazano na rys. 8. Nazwa nadana blokowi UDF w oknie jego właściwości jest wyświetlana zarówno na drzewie, jak i na symbolu bloku na schemacie.



Rys. 6. Dodawanie bloku UDF do drzewa projektu – ustawianie dostępu



Rys. 7 Dodanie bloku UDF do drzewa projektu



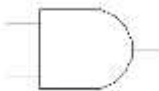
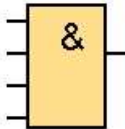
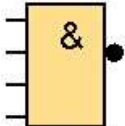
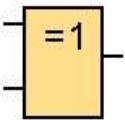
Rys. 8 Przykładowy schemat utworzony z użyciem bloku UDF

Z blokami UDF jest związana jeszcze jednak istotna funkcjonalność – wersjonowanie. Każde otwarcie bloku UDF do edycji, a następnie jego zapisanie na dysku powoduje otwarcie okna, w którym program proponuje nadanie numeru wersji. W ten sposób łatwo sprawdzić czy i kiedy blok był poprawiany oraz ile razy. Nie jest to jednak wersjonowanie pełne, takie jak w programach CAD lub EDA, ponieważ użytkownik nie ma możliwości przywołania i wyboru wcześniejszych wersji, a jedynie może dowiedzieć się, z którą definicją ma do czynienia i czy jest ona zgodna z oczekiwaną. Co ważne, jeśli blok UDF znajdujący się na schemacie FBD różni się od zapamiętanego w bibliotece, to na jego symbolu zostanie wyświetlony czerwony wykrzyknik. W przeciwnym wypadku jest wyświetlana zielona kropka o dosyć sporych wymiarach. W wypadku niezgodności można najechać myszką na symbol bloku, nacisnąć prawy przycisk i z menu podręcznego wybrać *Update UDF*. Jeśli chcemy podejrzeć jak jest wykonany blok, to w ten sam sposób z menu podręcznego należy wybrać *Expand UDF*.

Wspomnieć należy o istotnym ograniczeniu dotyczącym całego schematu. Mimo iż blok UDF wygląda jak predefiniowany przez producenta pakietu blok FBD, to składa się on z pewnej liczby bloków FBD, która pomniejsza liczbę dostępnych bloków w programie. To znaczy, jeśli do wykonania licznika z tego przykładu użyto 14 bloków FBD, to za każdym razem, maksymalna liczba dostępnych bloków FBD wynosząca 400 zostanie pomniejszona nie o 1, jak dla bloku UDF, ale o 14.

Tabela 1. Podstawowe bramki logiczne dostępne w sterowniku LOGO

FUNKCJA LOGICZNA	ZAPIS MATEMATYCZNY FUNKCJI	SYMBOLE BRAMKI LOGICZNEJ		TABLICA PRAWDY
A, B – wejścia bramki				

				<table><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	1	0	0	1	1	0									
1	0	0																	
1	1	0																	
AND (iloczyn logiczny)	$Y = A \cdot B$			<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Y																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
NAND (negacja iloczynu)	$Y = \overline{A \cdot B}$			<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	1	1	0	1	1	1	0
A	B	Y																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
XOR (alternatywa wykluczająca)	$Y = A \oplus B$			<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	0
A	B	Y																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	

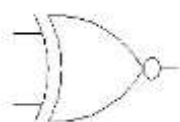
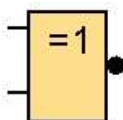
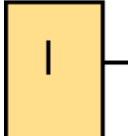
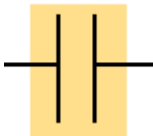
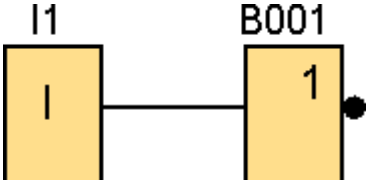
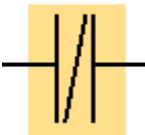
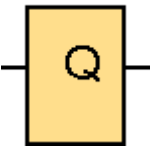
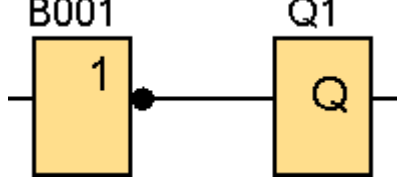
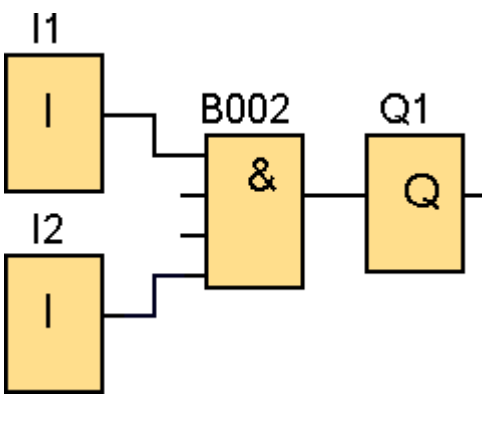
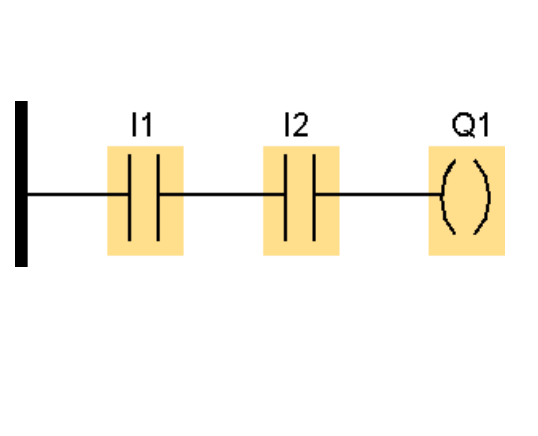
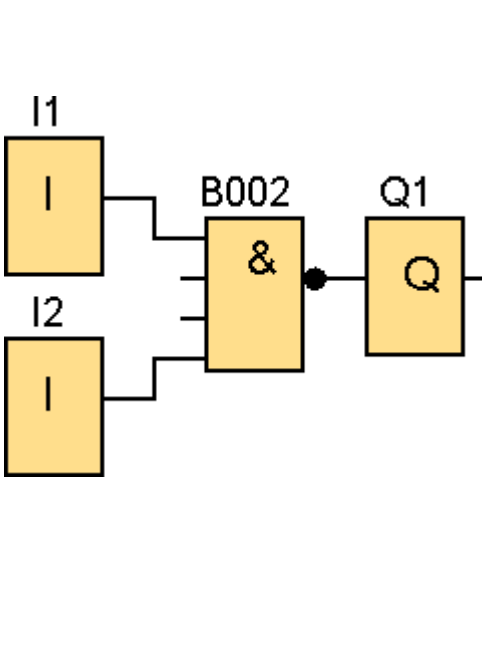
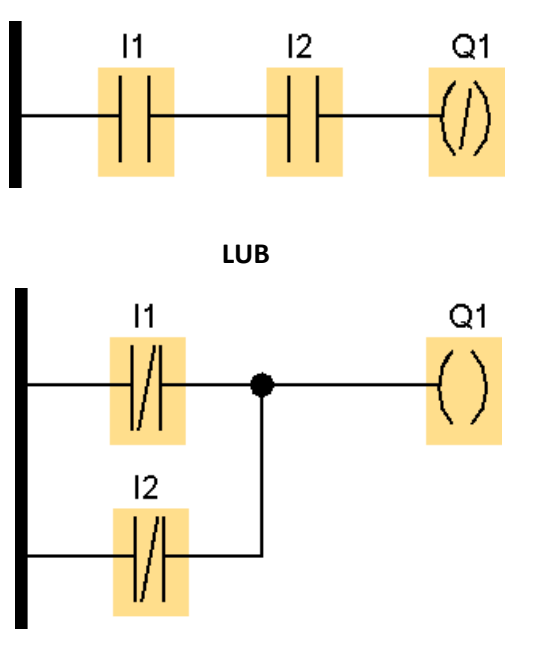
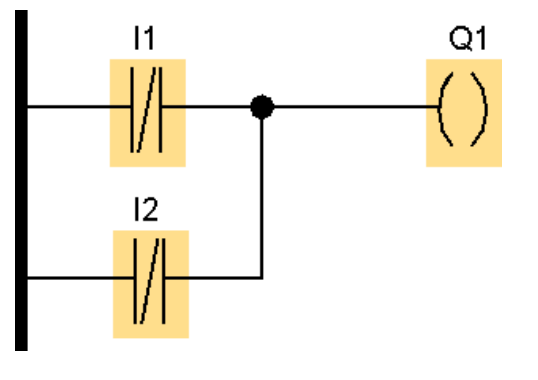
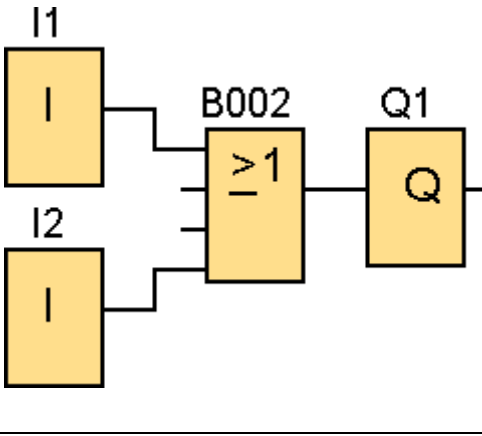
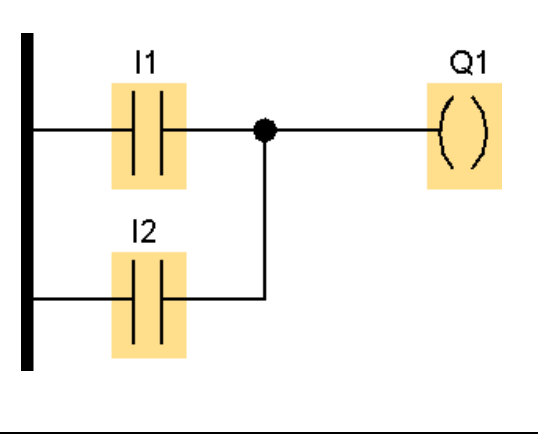
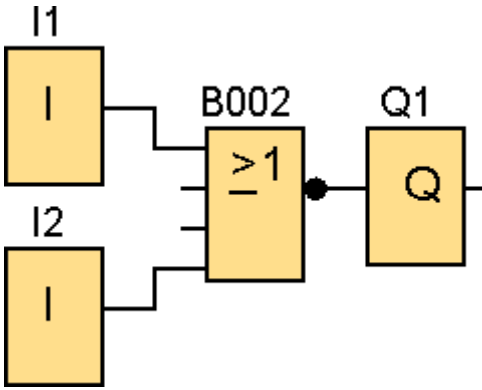
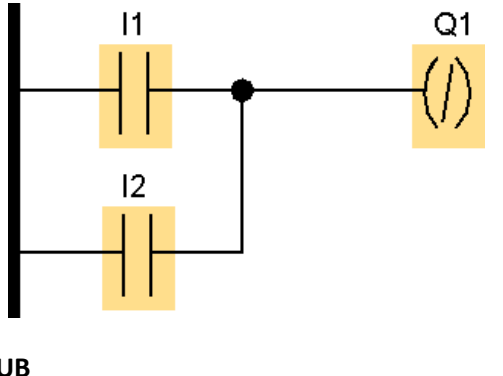
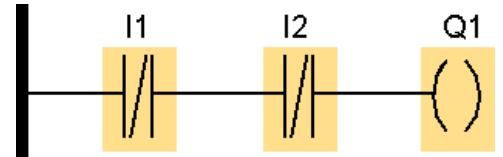
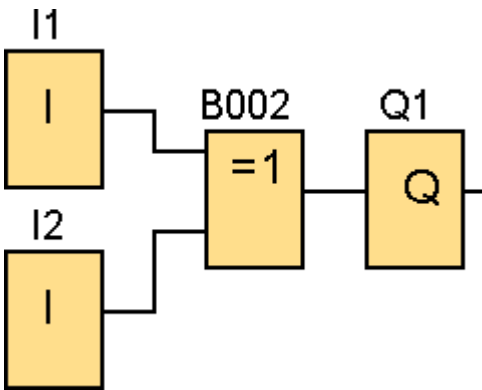
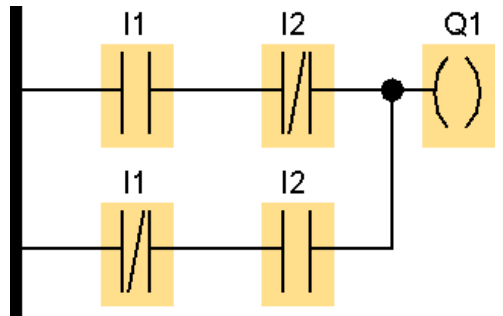
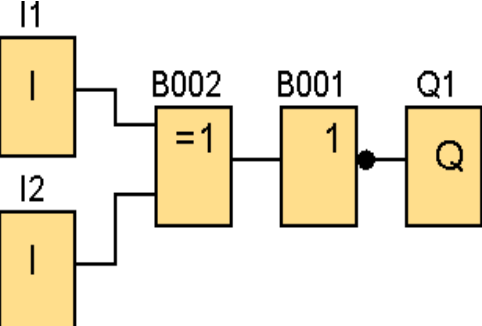
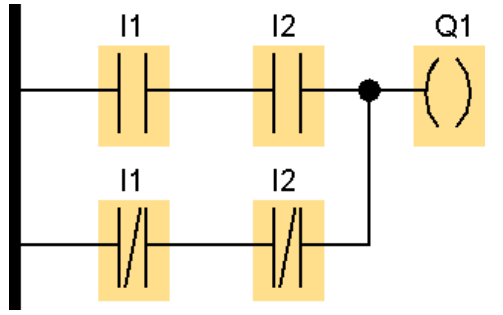
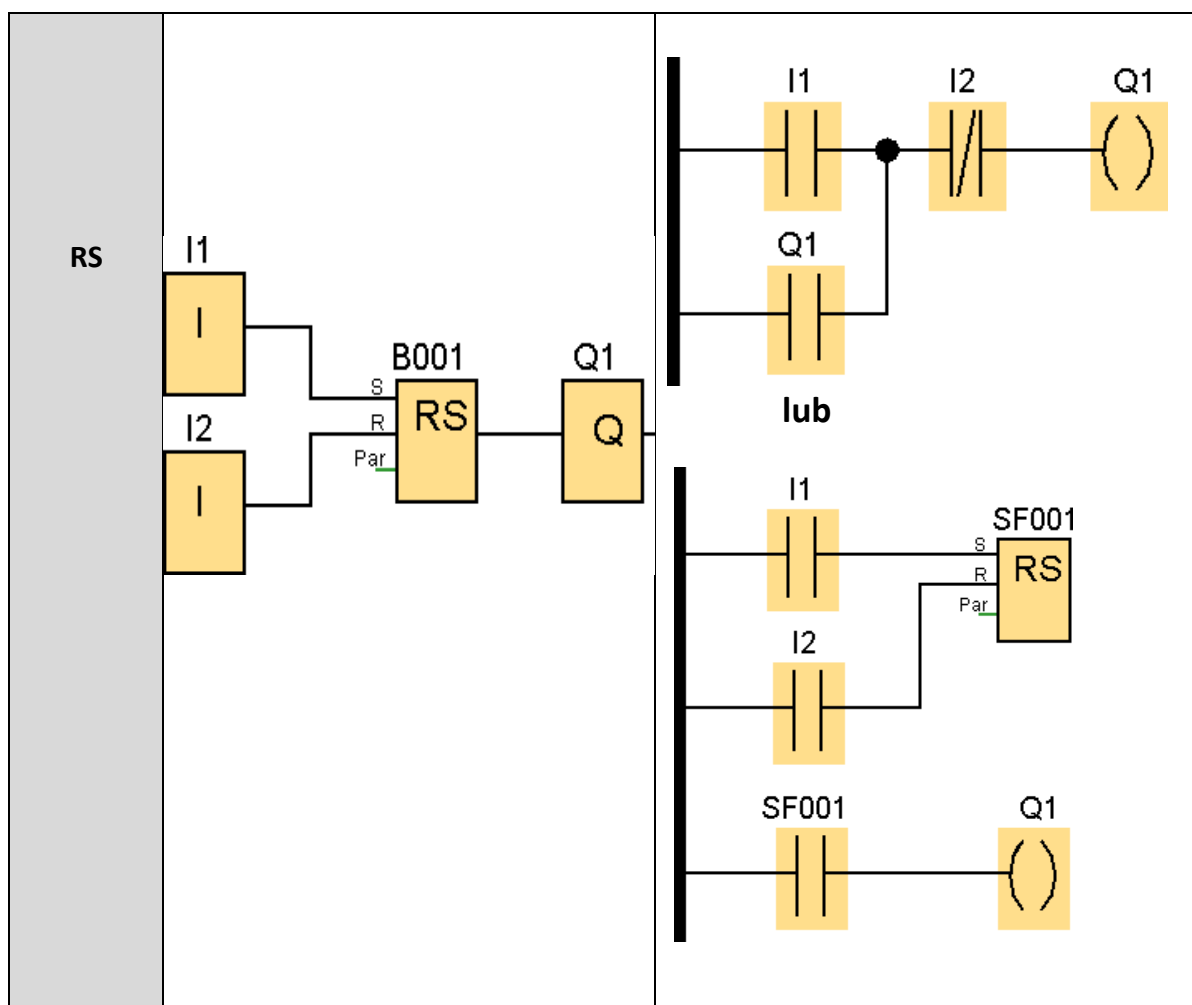
XNOR (negacja alternatywy)	$Y = \overline{A \oplus B}$			<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	1
A	B	Y																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

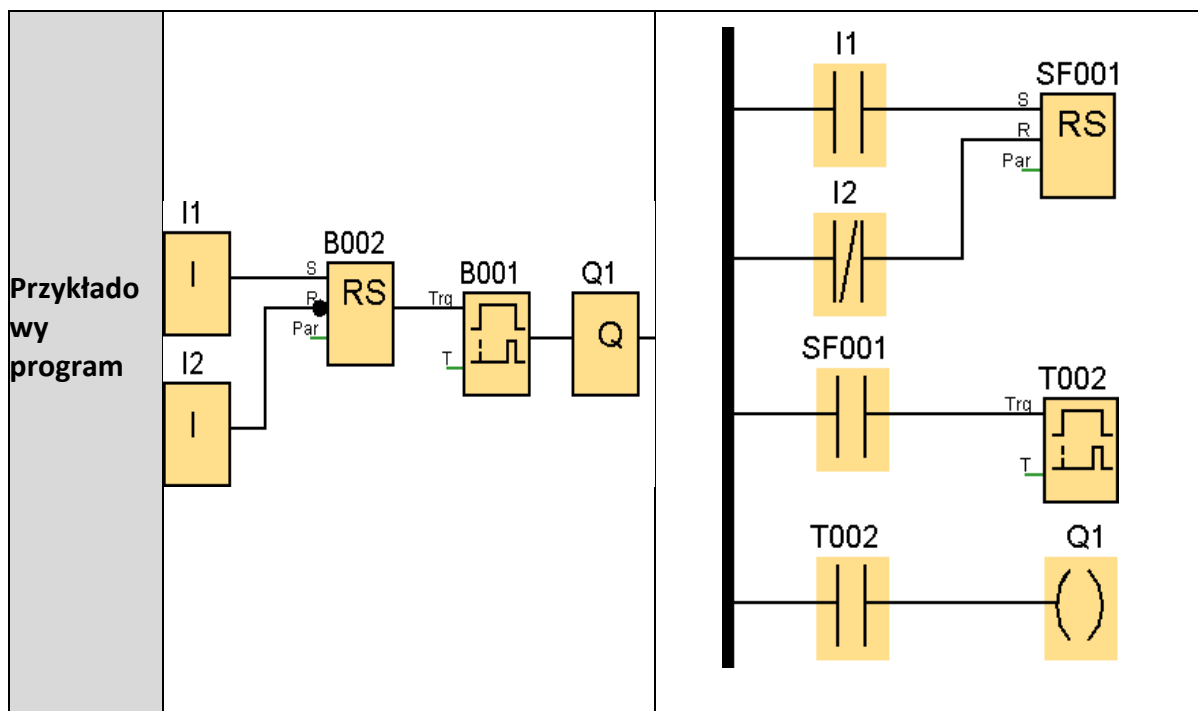
Tabela 2. Porównanie elementów FBD (function blok diagram – schemat bloków funkcjonalnych) z LD (ladder diagram – schemat w języku drabinkowym)

	FBD	LD
Wejście		
Wejście zanegowane		
Wyjście		
Wyjście zanegowane		

AND		
NAND		 LUB 
OR		

NOR		 LUB 
Ex-OR (XOR)		
Ex-NOR		





7. Pytania kontrolne

1. Bloki i programowanie sterowników LOGO w języku blokowym FBD.
2. Bloki i programowanie sterowników LOGO w języku drabinkowym LAD.

Literatura

1. Podręcznik programowania Siemens LOGO8 PL



Instrukcja do ćwiczenia

Ćwiczenie nr	25.2.
Temat :	Programowanie cyfrowe Timery
Stanowisko laboratoryjne	Sterownik LOGO
Opracował :	A. Mielewczyk



Instrukcja nr 25.2.

1. Temat ćwiczenia:

Programowania sterownika LOGO - Timery.

2. Cel ćwiczenia:

Celem ćwiczenia jest zapoznanie się z podstawowymi elementami sterowania czasem w programowaniu sterownika LOGO i wykonanie symulacji załączania wyjść sterownika za pomocą programu LOGO Soft Comfort v8.

3. Zakres wymaganych wiadomości:

- sygnały wejścia i wyjścia,
- elementy czasowe (timery),
- programowanie czasowe w języku drabinkowym LAD oraz FBD

4. Przebieg ćwiczenia:

- Zapoznać się z elementami systemu sterowania,
- Napisać program według polecenia.
- Przegrać program do sterownika.
- Przetestować program na stanowisku.

5. Stanowisko laboratoryjne:

Stanowiskiem badawczym jest sterownik SIEMENS LOGO wyposażony wejścia przyciskowe monostabilne oraz wyjścia typu LED. Pozwala to przetestować napisany program wgrany do sterownika

6. Sprawozdanie z ćwiczenia:

Część wstępna, opis elementów użytych do programowania oraz diagram programu w FBD i LAD.

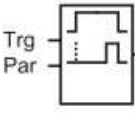
WPROWADZENIE

Timery w LOGO:

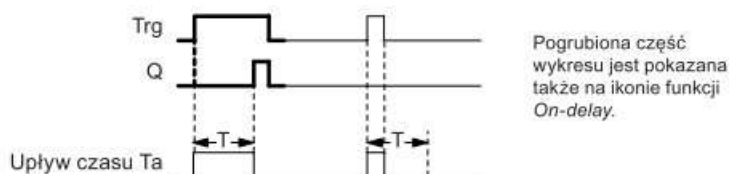
1. On-Delay (opóźnienie włączenia)

Skrócony opis

Wyjście jest włączane dopiero po upływie zadanego czasu, wprowadzanego jako parametr wejściowy.

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Sygnał 1 na wejściu Trg (<i>Trigger</i>) rozpoczyna odliczanie czasu.
	Parametr	T określa czas, po którym zostanie włączone wyjście (zmiana z 0 na 1 sygnału wyjściowego). Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku.
	Wyjście Q	Stan Q jest zmieniany z 0 na 1 po upływie czasu T, jeżeli wejście Trg jest nadal w stanie 1.

Wykres czasowy



Opis działania

Zmiana stanu na wejściu Trg z 0 na 1 powoduje rozpoczęcie odliczania czasu T_a (T_a jest czasem roboczym LOGO!).

Jeśli na wejściu Trg stan 1 trwa co najmniej tak długo, ile wynosi zadany czas T, po upływie czasu T na wyjściu pojawia się stan 1 (wyjście zostaje włączone z zadaniem opóźnieniem względem wejścia).

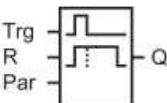
Odliczanie zadanego czasu T ulega przerwaniu, jeśli przed jego upływem wejście Trg powróci do stanu 0.

Wyjście jest zerowane, gdy wejście Trg powróci do stanu 0.

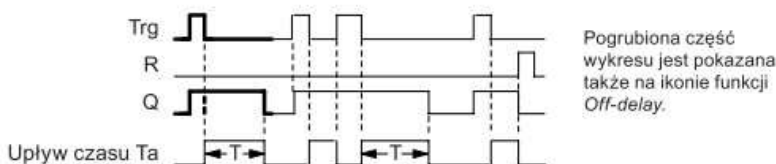
2. Off-Delay (opóźnienie wyłączenia)

Skrócony opis

Wyjście bloku przyjmuje stan 0 po upływie zadanego czasu.

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Start odliczania czasu następuje przy ujemnym zboczu (przejście z 1 na 0) sygnału na wejściu Trg (<i>Trigger</i>).
	Wejście R	Sygnał 1 na wejściu R zeruje czas opóźnienia oraz wyjście.
	Parametr	Wyjście zostaje wyłączone (przejście z 1 na 0) po upływie czasu opóźnienia T. Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku
	Wyjście Q	Wyjście Q jest włączane sygnałem na wejściu Trg. Stan ten jest utrzymywany przez czas opóźnienia T.

Wykres czasowy



Opis działania

Stan wyjścia Q zmienia się na 1 równocześnie ze zmianą wejścia Trg do stanu 1. Odliczanie czasu roboczego T_a w LOGO! jest wznawiane przy zmianie stanu wejścia Trg ze stanu 1 na 0, przy czym wyjście pozostaje włączone. Po osiągnięciu przez T_a zadanej wartości opóźnienia ($T_a = T$) wyjście Q powraca do stanu 0 (opóźnienie wyłączenia).

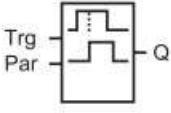
Impuls (zmiana z 0 na 1 i ponownie na 0) na wejściu Trg powoduje wznowienie odliczania czasu T_a .

Ustawienie wejścia R (Reset) powoduje wyzerowanie czasu T_a oraz stanu wyjścia przed upływem czasu opóźnienia.

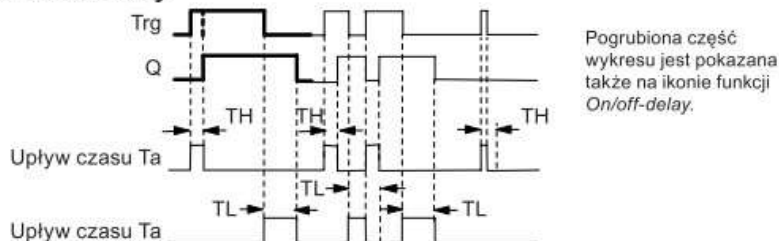
3. On-/Off-Delay (opóźnienie włączenia/wyłączenia)

Skrócony opis

Funkcja opóźnienia włączenia/wyłączenia powoduje włączenie wyjścia po czasie opóźnienia włączenia, a następnie jego wyłączenie po upływie czasu wyłączenia.

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Dodatnie zbocze (przejście z 0 na 1) na wejściu Trg (<i>Trigger</i>) rozpoczyna odliczanie czasu opóźnienia włączenia T_H . Ujemne zbocze (przejście z 1 na 0) na wejściu Trg (<i>Trigger</i>) rozpoczyna odliczanie czasu opóźnienia wyłączenia T_L .
	Parametr	T_H wyznacza czas, po którym wyjście zostaje włączone (sygnał wyjściowy zmienia się z 0 na 1). T_L wyznacza czas, po którym wyjście zostaje wyłączone (sygnał wyjściowy zmienia się z 1 na 0). Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku.
	Wyjście Q	Wyjście Q jest włączane po upływie czasu T_H jeżeli na wejściu Trg jest utrzymywany stan 1. Wyjście Q jest wyłączane po upływie czasu T_L jeżeli na wejściu Trg nie wystąpił ponownie stan 1.

Wykres czasowy



Opis działania

Odliczanie czasu T_H jest inicjowane przez zmianę z 0 na 1 sygnału na wejściu Trg.

Jeżeli wejście Trg pozostaje w stanie 1 co najmniej przez czas T_H , to wyjście przyjmuje stan 1 po upływie czasu T_H (wyjście powtarza stan wejścia z opóźnieniem o czas włączenia).

Czas opóźnienia jest zerowany gdy sygnał na wejściu Trg powraca do stanu 0 przed upływem czasu T_H .

Przejście z 1 na 0 na wejściu Trg inicjuje odliczanie czasu T_L .

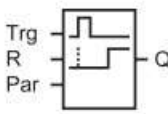
Jeżeli wejście Trg pozostaje w stanie 0 co najmniej przez czas T_L , to wyjście przyjmuje stan 0 po upływie czasu T_L (wyjście powtarza stan wejścia z opóźnieniem o czas wyłączenia).

Czas opóźnienia jest zerowany gdy sygnał na wejściu Trg zmienia się na 1 przed upływem czasu T_H .

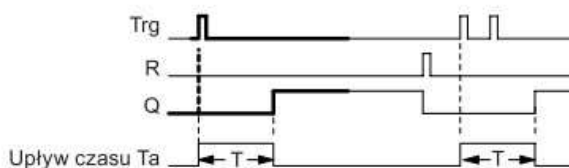
4. Retentive On-Delay (opóźnione załączenie z podtrzymaniem)

Skrócony opis

Impuls na wejściu rozpoczyna odliczanie zadanego czasu. Włączenie wyjścia następuje po upływie tego czasu.

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Zmiana sygnału wejściu Trg (<i>Trigger</i>) z 0 na 1 inicjuje odliczanie opóźnienia włączenia.
	Wejście R	Sygnał na wejściu R zeruje czas opóźnienia oraz wyjście.
	Parametr	T reprezentuje czas opóźnienia włączenia wyjścia (zmiana stanu wyjścia z 0 na 1). Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku
	Wyjście Q	Wyjście Q jest włączane po upływie czasu opóźnienia T.

Wykres czasowy



Pogrubiona część wykresu jest pokazana także na ikonie funkcji *Retentive on -delay*.

Opis działania

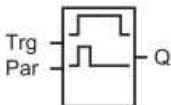
Zmiana stanu na wejściu Trg z 0 na 1 powoduje rozpoczęcie odliczania czasu T_a . Stan wyjścia Q zmienia się w chwili gdy $T_a = T$. Dalsze zmiany sygnału na wejściu Trg nie wpływają na czas T_a .

Stan wyjścia oraz czas T_a sa zerowane dopiero po pojawieniu się stanu 1 na wejściu R.

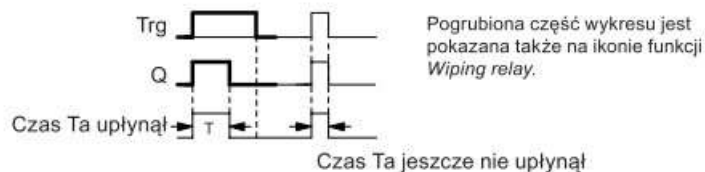
5. Wiping relay (pulse output) (przełącznik czasowy z wyjściem impulsowym)

Skrócony opis

Sygnał wejściowy powoduje wygenerowanie na wyjściu sygnału o określonym czasie trwania.

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Sygnał 1 na wejściu Trg (<i>Trigger</i>) rozpoczyna odliczanie czasu.
	Parametr	Wyjście zostaje wyłączone po upływie czasu T (sygnał wyjściowy zmienia się z 1 na 0). Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku.
	Wyjście Q	Wyjście przyjmuje stan 1, gdy na wejściu Trg pojawi się sygnał 1 i pozostaje wzbudzone przez zadany czas T_a , chyba że wcześniej zaniknie sygnał na wejściu.

Wykres czasowy



Opis działania

Zmiana stanu na wejściu Trg z 0 na 1 powoduje ustawienie wyjścia w stan 1 i rozpoczęcie odliczania czasu T_a podczas którego wyjście pozostaje włączone.

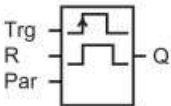
Stan wyjścia zmienia się na 0 (wyjścia impulsowe) w chwili, gdy czas T_a osiągnie zadaną wartość parametru T ($T_a = T$).

Jeżeli przed upływem zadanego czasu opóźnienia nastąpi zmiana sygnału na wejściu Trg z 1 na 0, to stan wyjścia zmienia się z 1 na 0.

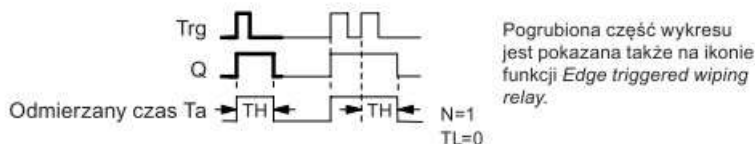
6. Edge triggered wiping relay (przełącznik czasowy wyzwalany zboczem)

Skrócony opis

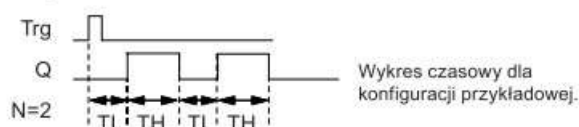
Sygnał na wejściu powoduje wygenerowanie na wyjściu, po określonym czasie opóźnienia, zadanej liczby impulsów o określonym stosunku czasu trwania impulsu do przerwy między impulsami (z możliwością wznowienia).

Symbol w LOGO!	Konektor	Opis
	Wejście Trg	Sygnał 1 na wejściu Trg (<i>Trigger</i>) rozpoczyna odliczanie czasu w przełączniku czasowym wyzwalanym zboczem.
	Wejście R	Sygnał na wejściu R zeruje licznik czasu (T_a) oraz wyjście.
	parametr	Konfigurowany jest odstęp między impulsami T_L oraz szerokość impulsu T_H . N określa liczbę cykli impuls/pauza T_L/T_H : Zakres wartości parametru N: 1...9. Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku.
	Wyjście Q	Wyjście Q jest włączane po upływie czasu opóźnienia T_L , a następnie zerowane po upływie czasu T_H .

Wykres czasowy A



Wykres czasowy B



Opis działania

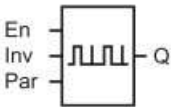
Zmiana stanu na wejściu Trg z 0 na 1 powoduje rozpoczęcie odliczania czasu T_L (*Time Low*). Po upływie czasu T_L stan wyjścia Q zmienia się na 1 na czas T_H (*Time High*).

Jeśli przed upływem czasu ($T_L + T_H$) na wejściu Trg pojawi się ponowny impuls wyzwalający (zmiana stanu z 0 na 1), licznik czasu roboczego T_a zostaje wyzerowany, a cykl impuls/przerwa zaczyna się od początku.

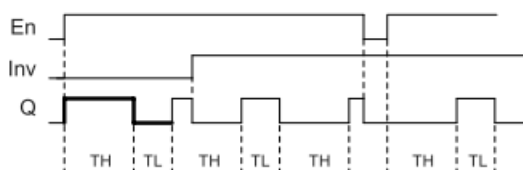
7. Asynchronous Pulse Generator (asynchroniczny generator impulsów)

Skrócony opis

Postać sygnału wyjściowego jest konfigurowana za pośrednictwem czasu trwania impulsu i przerwy między impulsami.

Symbol w LOGO!	Konektor	Opis
	Wejście En	Wejście EN służy do ustawiania i zerowania wyjścia generatora impulsów asynchronicznych.
	Wejście INV	Wejście INV służy do wprowadzania inwersji sygnału wyjściowego generatora impulsów asynchronicznych.
	parametr	Użytkownik może konfigurować wartości szerokości impulsu T_H oraz odstępu między impulsami T_L . Podtrzymanie: / = brak podtrzymania, R = podtrzymanie stanu bloku.
	Wyjście Q	Wyjście Q jest włączane i wyłączane cyklicznie zgodnie z zadanymi wartościami czasu T_H i T_L .

Wykres czasowy



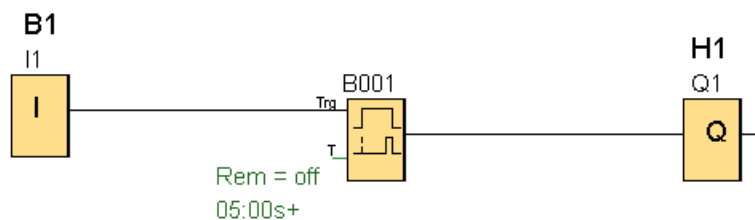
Opis działania

Użytkownik może konfigurować wartości czasu trwania impulsu T_H (*Time High*) oraz przerwy między impulsami T_L (*Time Low*).

Wejście Inv służy do inwertowania wartości sygnału wyjściowego, pod warunkiem, że blok jest aktywowany przez sygnał 1 podany na wejście EN.

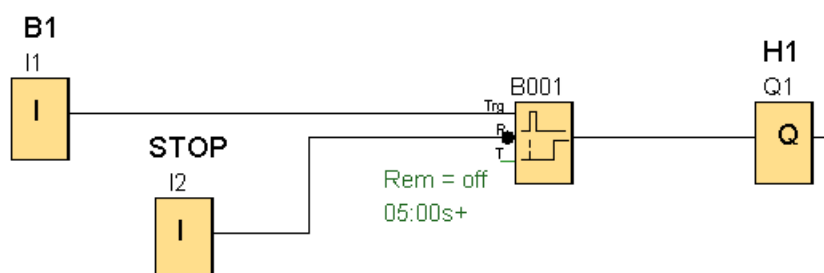
1. Opóźnienie załączenia 1

Pojawienie się obiektu przed czujnikiem B1 powoduje załączenie po upływie 5 sekund alarmu (lampki sygnalizacyjnej H1). Alarm nie załączy się jeżeli obiekt zniknie przed upływem 5 sekund. Załączony alarm wyłączy się samoczynnie po zniknięciu obiektu.



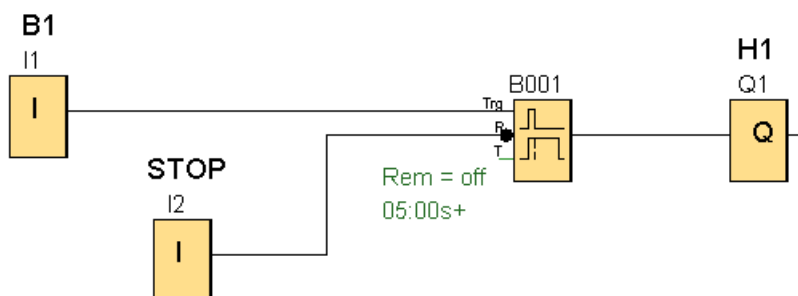
2. Opóźnienie załączenia 2

Pojawienie się obiektu przed czujnikiem B1 powoduje załączenie po upływie 5 sekund alarmu (lampki sygnalizacyjnej H1). Alarm załączy się nawet jeżeli obiekt zniknie przed upływem 5 sekund. Alarm można wyłączyć tylko poprzez krótkotrwałe naciśnięcie przycisku monostabilnego (NC) STOP.



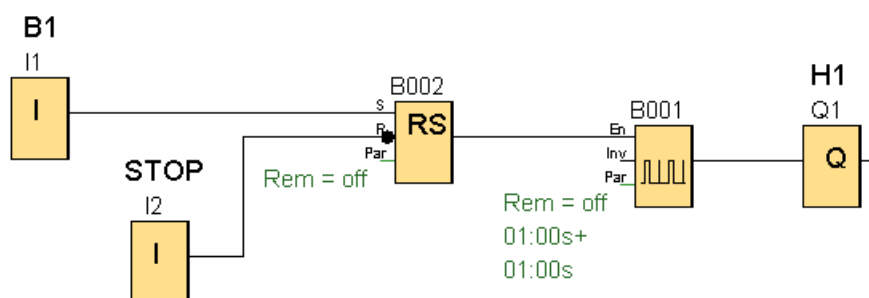
3. Opóźnienie wyłączenia

Pojawienie się obiektu przed czujnikiem B1 powoduje natychmiastowe załączenie alarmu (lampki sygnalizacyjnej H1). Alarm wyłączy się samoczynnie dopiero po 5 sekundach od zniknięcia obiektu. Alarm można też wyłączyć wcześniej poprzez krótkotrwałe naciśnięcie przycisku monostabilnego (NC) STOP.



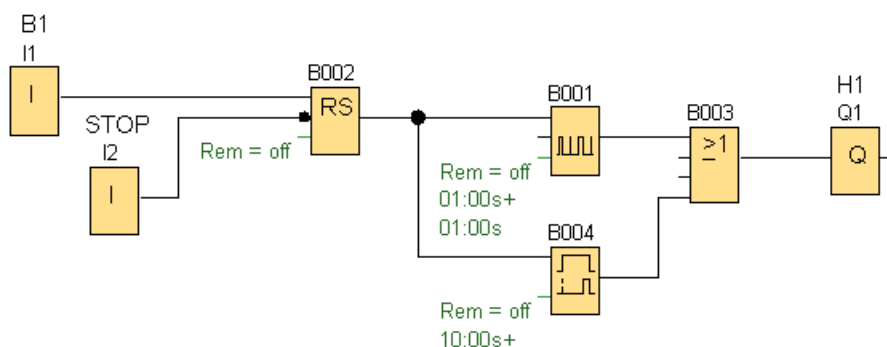
4. Miganie lampki sygnalizacyjnej

Pojawienie się obiektu przed czujnikiem B1 powoduje natychmiastowe załączenie alarmu - miga lampka sygnalizacyjna H1. Alarm pozostaje załączony nawet po zniknięciu obiektu. Alarm można wyłączyć tylko poprzez krótkotrwałe naciśnięcie przycisku monostabilnego (NC) STOP.



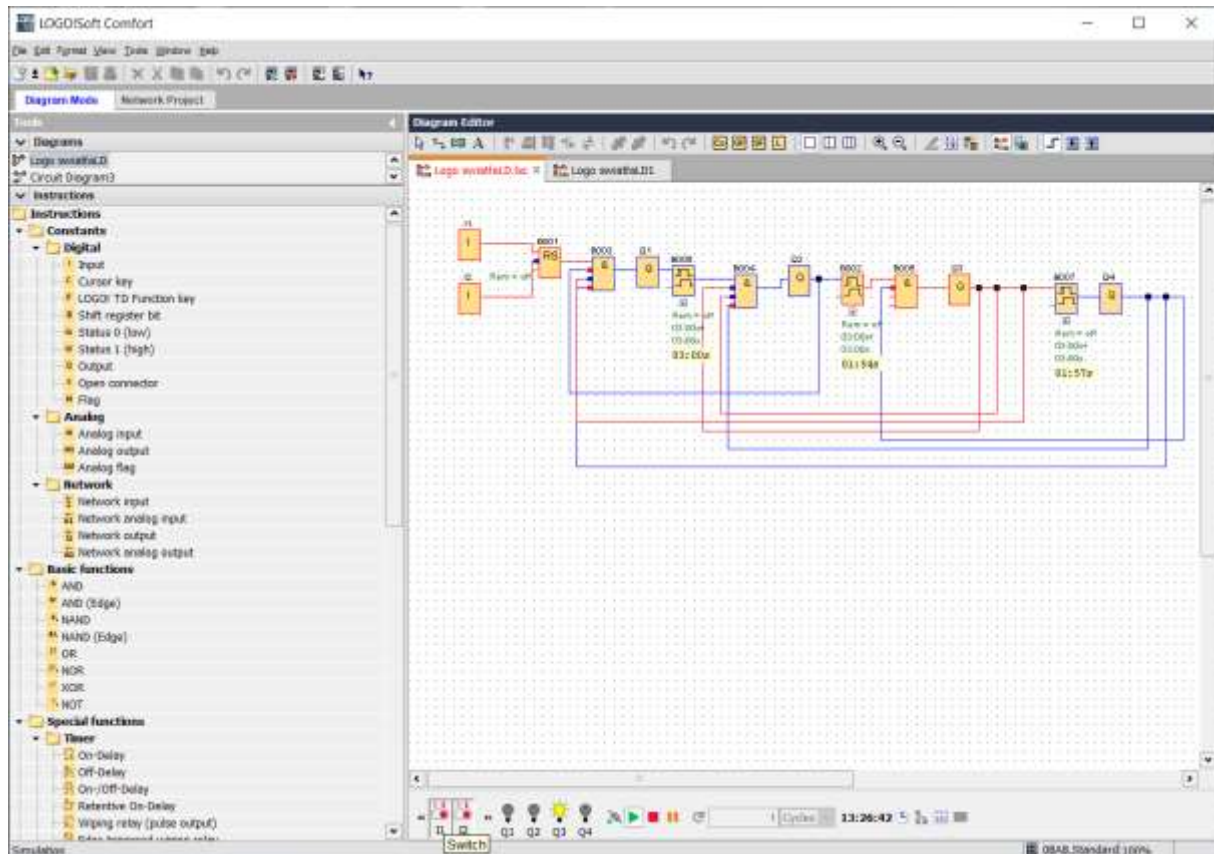
5. Miganie i świecenie ciągle lampki sygnalizacyjnej

Pojawienie się obiektu przed czujnikiem B1 powoduje natychmiastowe załączenie alarmu - miga lampka sygnalizacyjna H1. Po 10 sekundach od załączenia alarmu lampka świeci światłem ciągłym. Alarm pozostaje załączony nawet po zniknięciu obiektu. Alarm można wyłączyć tylko poprzez krótkotrwałe naciśnięcie przycisku monostabilnego (NC) STOP.

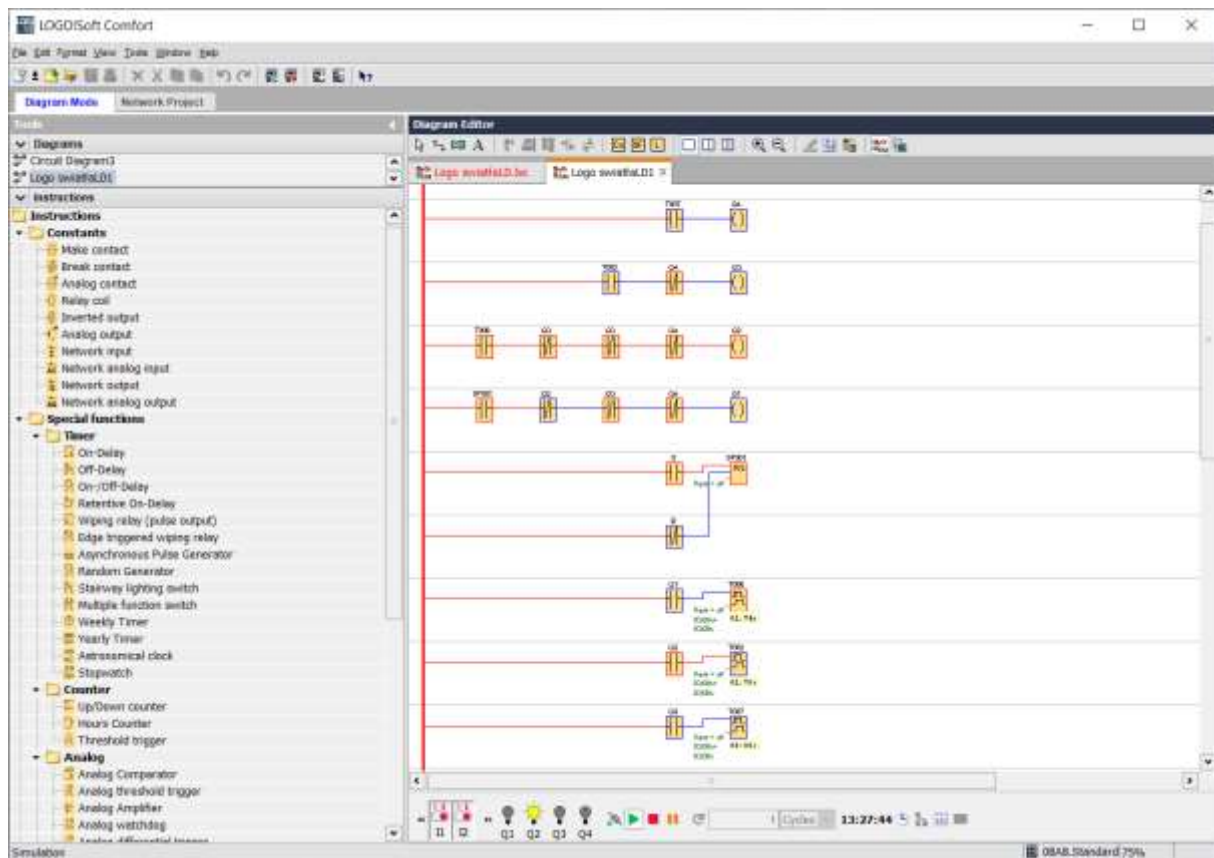


Zadanie 1

Przedstaw program na sekwencyjne załączanie lampek.



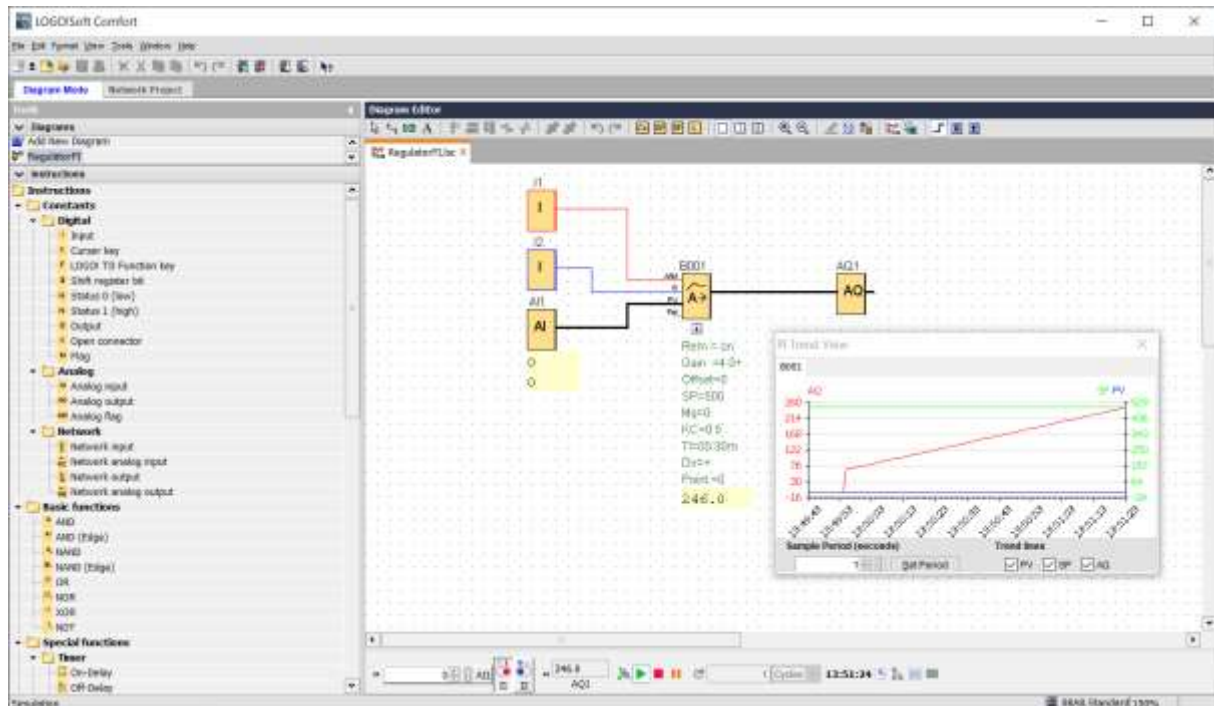
Rys. 3 Program na sekwencyjne załączanie lampek – bloki FBD



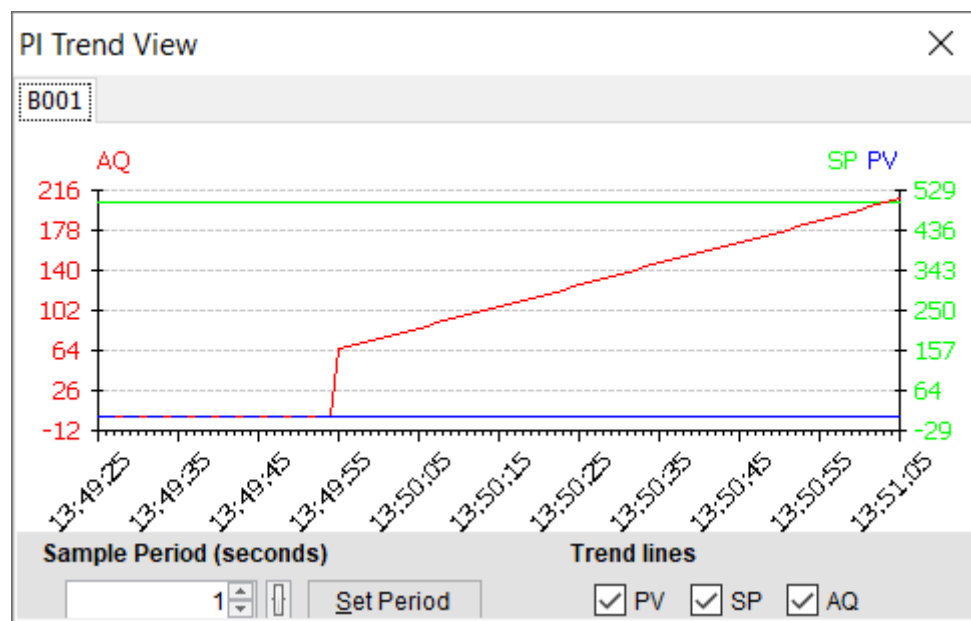
Rys. 4 Program na sekwencyjne załączanie lampek – program LAD

Zadanie 2

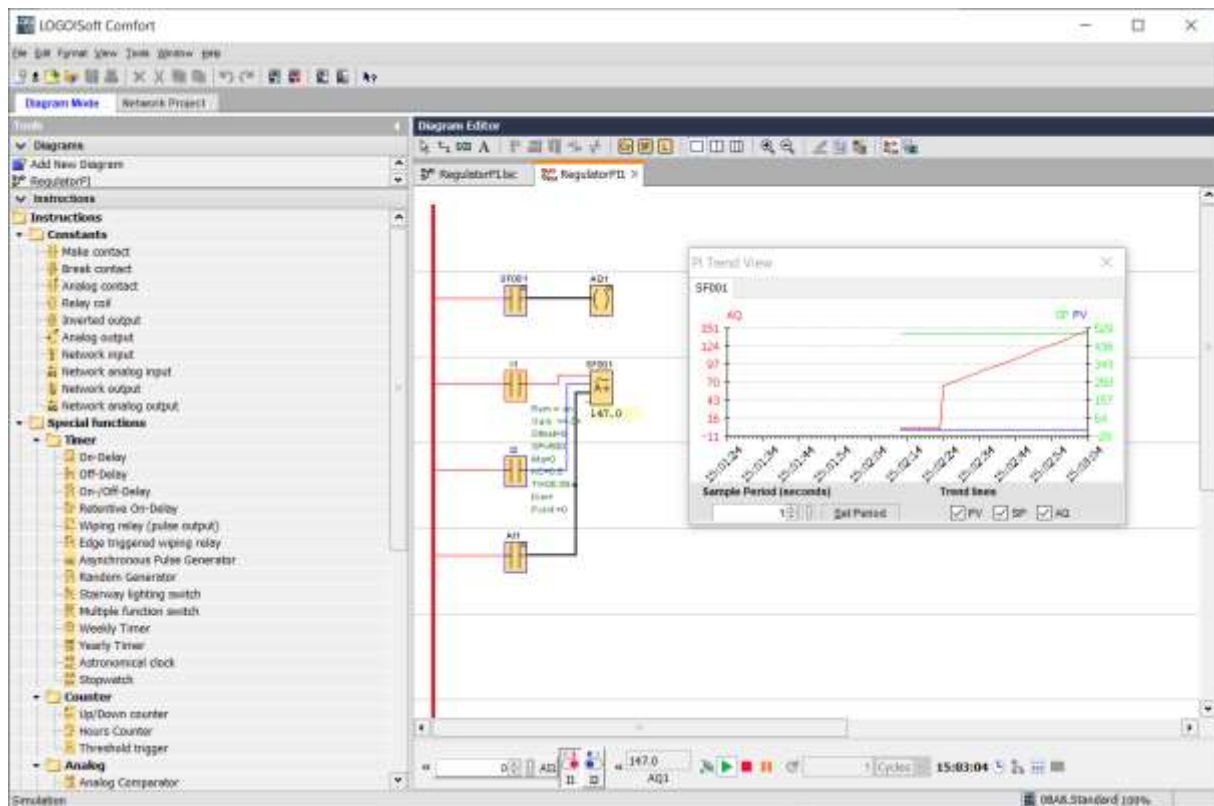
Przedstaw program z regulacją PI.



Rys. 5. Regulator PI na sterowniku LOGO – bloki FBD



Rys. 6 Linia trendu wyjścia regulatora PI sterownika LOGO



Rys. 7 Regulator PI na sterowniku LOGO – program LAD

1. Pytania kontrolne

- Elementy programowe w systemie FBD
- Elementy programowe w systemie LAD
- Timery obsługiwane przez sterownik LOGO
- Programowanie przełączeń wyjść w LOGO
- Programowanie regulatora PI

Literatura

1. Podręcznik programowania Siemens LOGO8 PL