1. PIERWSZY PROGRAM W JĘZYKU SFC

1.1. Kroki

W programie dostępne są dwa typy kroków: kroki uproszczone i kroki zgodne z normą IEC 61131-3 – kroki zgodne z normą będą w dalszej części opracowania nazywane krokami IEC. Korzystanie z kroków IEC jest uzależnione od ustawienia opcji „*Use IEC-Steps*" i dołączenia dodatkowej biblioteki: „Iecsfc.lib" – problem ten zostanie omówiony szczegółowo w punkcie 2.1.1.

1.2. Tranzycje

Warunki przejść w języku ST można w CoDeSys definiować bezpośrednio z poziomu edytora języka SFC obok symbolu tranzycji. Warunki w językach IL, LD, FBD muszą, a warunki w języku ST mogą, być zapisywane w postaci osobnych *funkcji* tworzonych w edytorze tranzycji.

Uwaga! Do zapisu koniunkcji nie można wykorzystywać symbolu „&" – dostępny jest tylko operator AND.

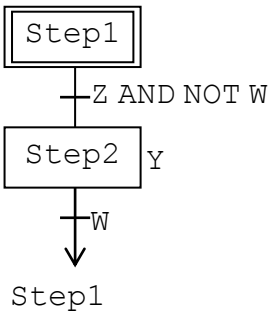
1.3. Skok do kroku

Edytor języka SFC programu ISaGRAF nie pozwala na łączenie odległych kroków za pomocą linii łączących, takie kroki można ze sobą połączyć jedynie wykorzystując operację *skoku do kroku*.

Sposób wprowadzania programu w języku SFC w CoDeSys zostanie wyjaśniony w oparciu o przykłady 1. i 3. z dodatku poświęconego językowi SFC.

2. Zadanie 1.

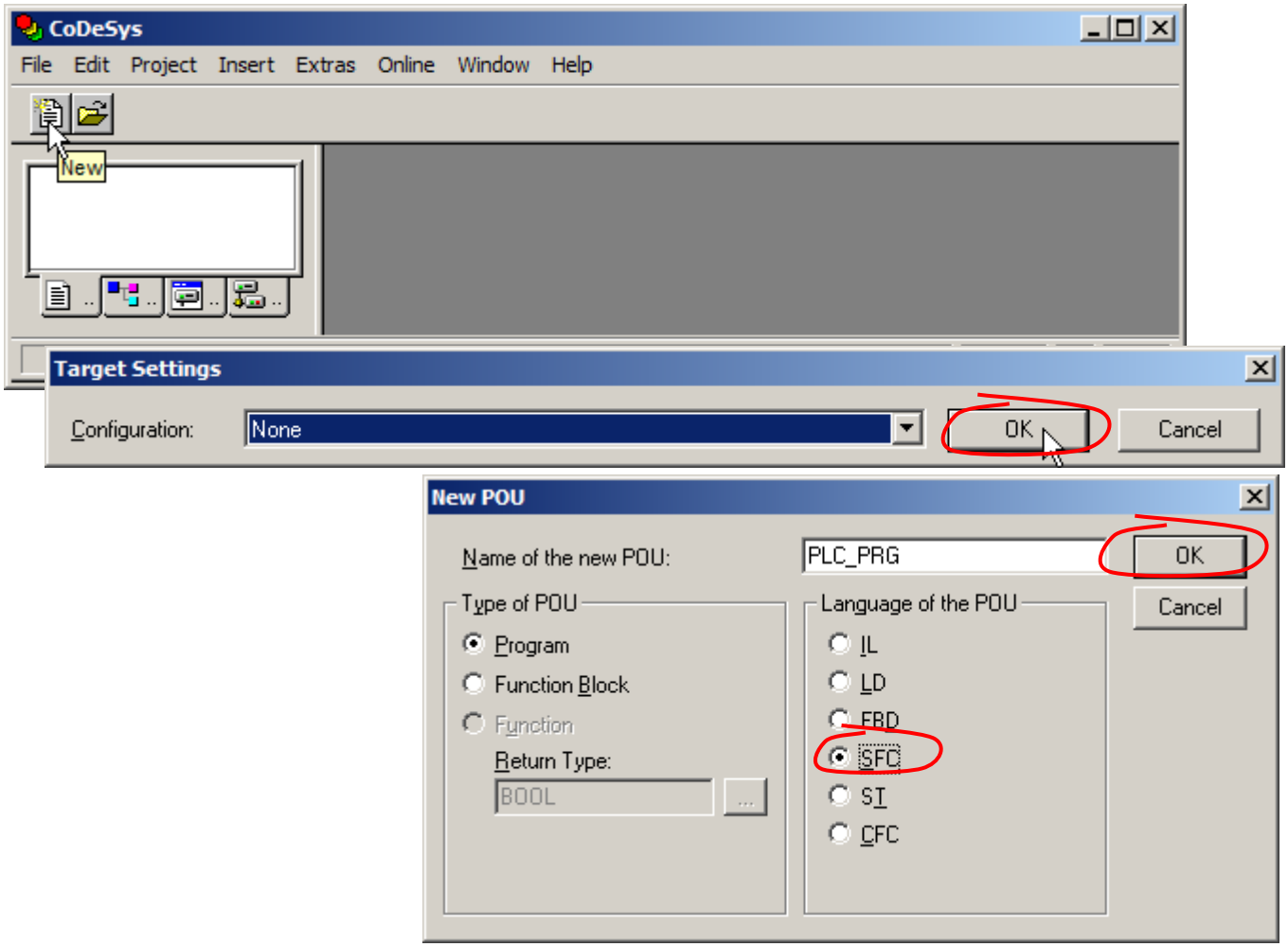
Program, którego schemat przedstawiony został na *rys. 1.* załącza i wyłącza urządzenie w zależności od stanu dwóch przycisków. Sterowanie pracą urządzenia sprowadza się do nadawania wartości zmiennej wyjściowej Y na podstawie wartości dwóch zmiennych wejściowych Z i W (wartości zmiennych wejściowych odpowiadają aktualnemu stanowi przycisków Załącz i Wyłącz). Graf tego programu został szczegółowo omówiony w przykładzie 1. w dodatku poświęconym językowi SFC.



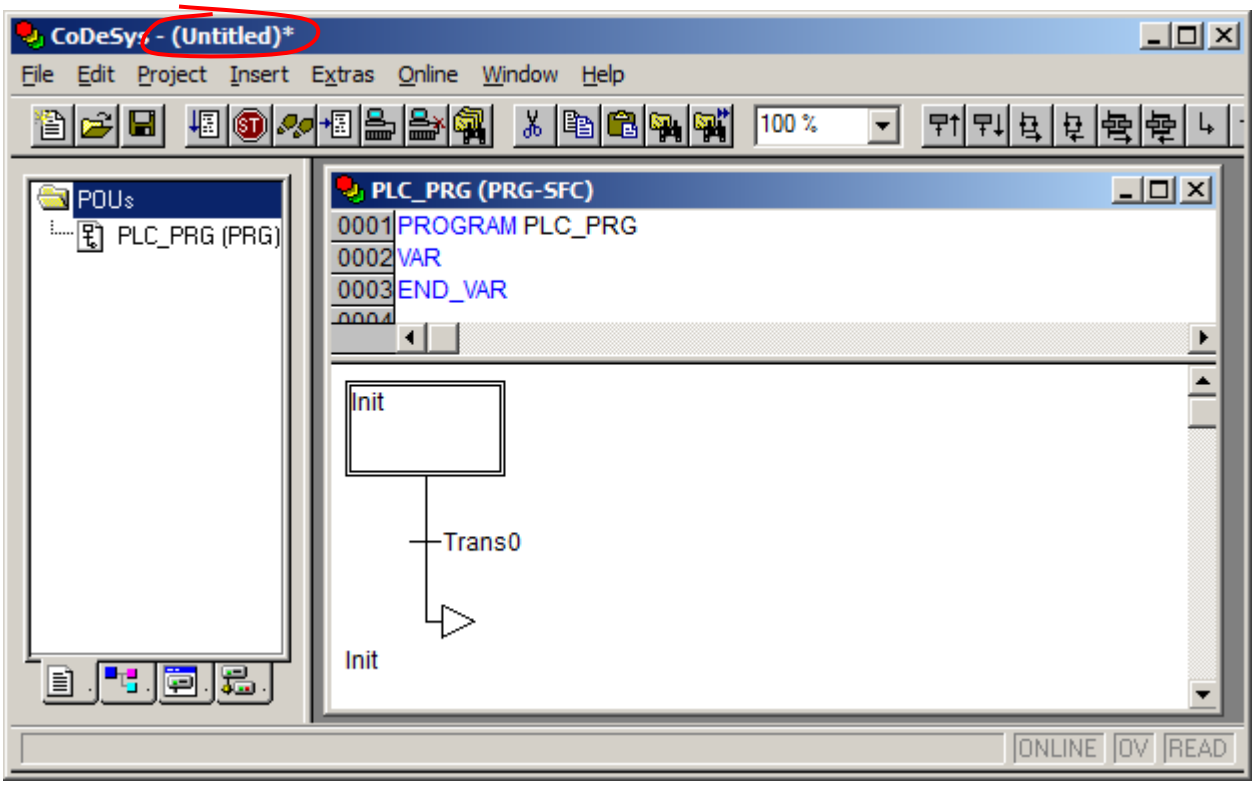
Rys. 1. Przykładowy program w języku SFC.

2.1. Tworzenie nowego projektu

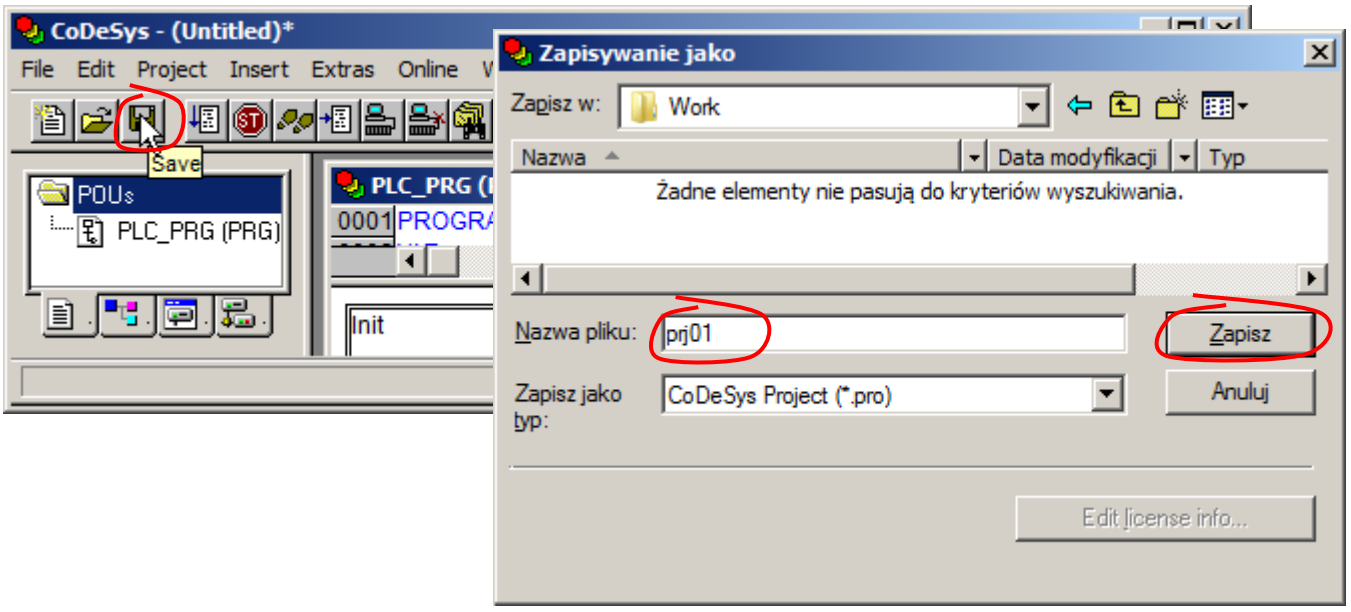
Przygotowanie projektu z programem głównym w języku SFC wymaga wykonania kroków przedstawionych na kolejnych rysunkach:



Po utworzeniu, nowy projekt otrzymuje nazwę **Untitled** i jest automatycznie otwierany w środowisku.



Na koniec, projekt należy zapisać na dysku nadając mu wybraną nazwę.



W kolejnym kroku użytkownik może zdefiniować zmienne niezbędne do realizacji programu: 2 zmienne wejściowe: `Z` i `W` oraz 1 zmienną wyjściową `Y`. Sposób wprowadzania zmiennych zostanie pominięty (został on szczegółowo omówiony przy okazji omawiania edycji programów w językach FBD i LD).

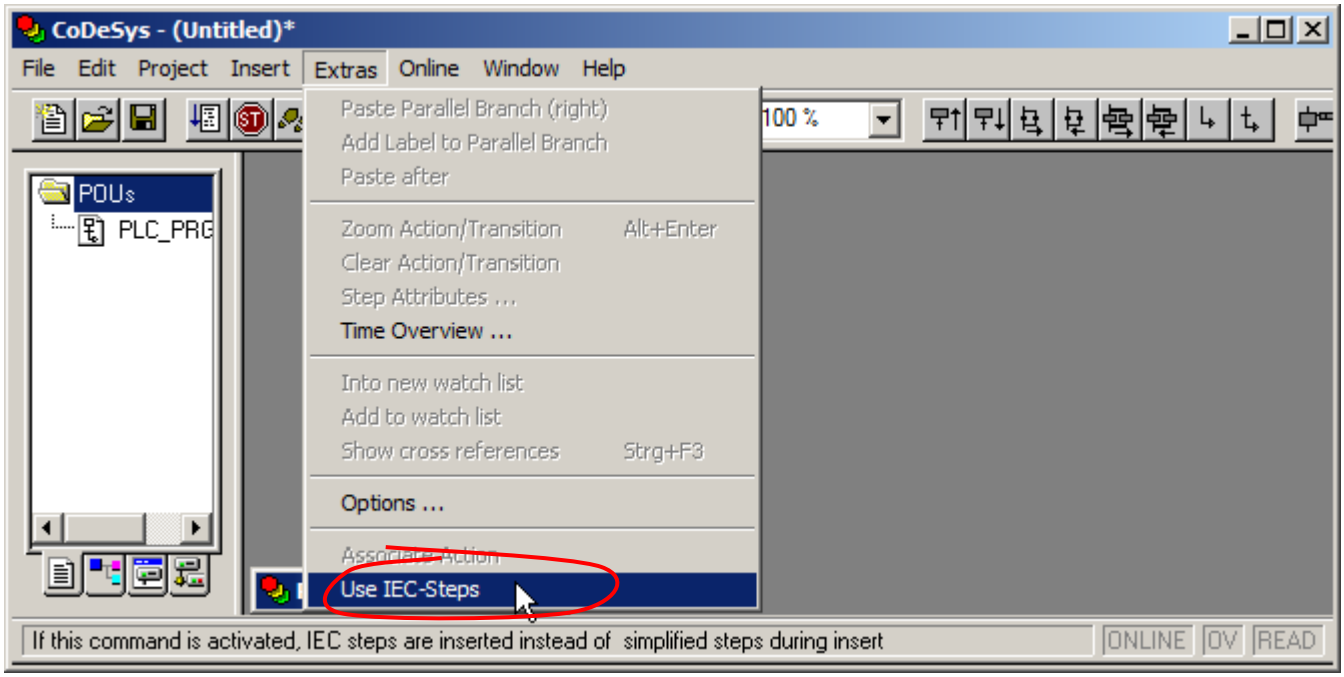
Program może napisany z użyciem kroków uproszczonych i/lub kroków IEC, obydwa typy kroków mogą występować w jednym programie, W materiale omówione zostaną dwa programy:

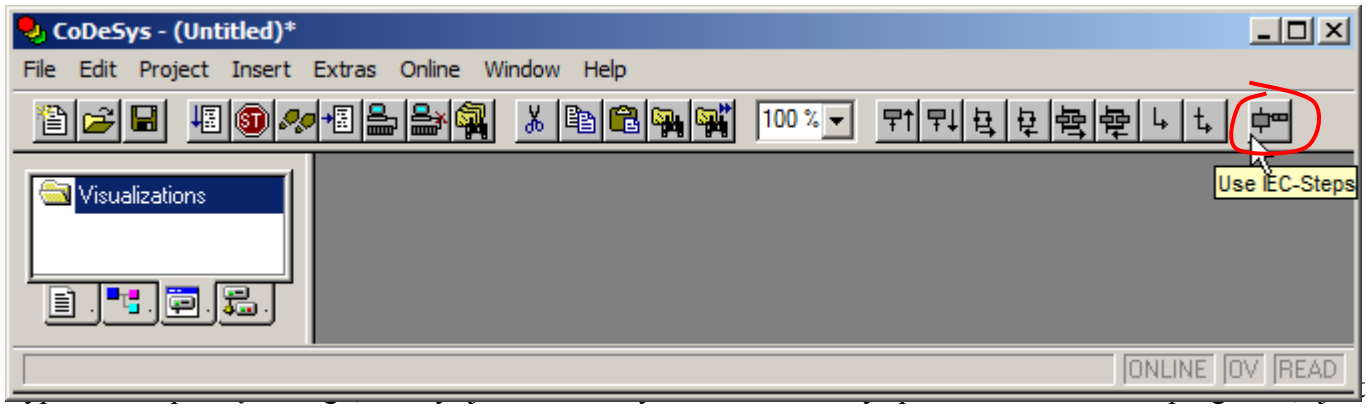
- do napisania pierwszego zostaną wykorzystane wyłącznie kroki IEC,
- drugi program zostanie napisany w oparciu o kroki uproszczone.

2.1.1. Program z krokami IEC

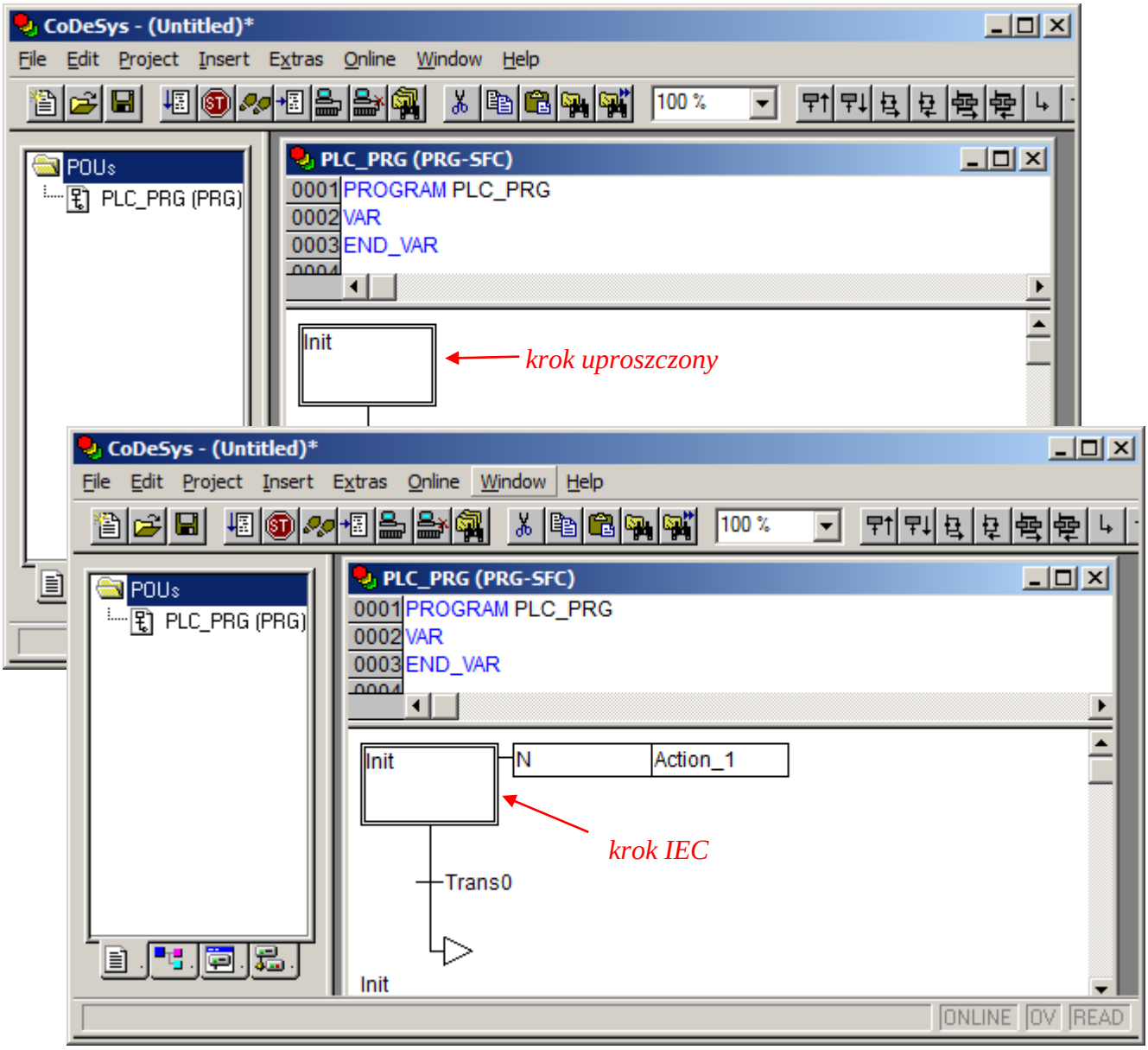
Korzystanie z kroków IEC

Korzystanie z kroków IEC jest uzależnione od ustawienia opcji „*Use IEC-Steps*” – opcja ta jest dostępna jedynie w przypadku edycji programu w języku SFC.





uzależniony od ustawienia opcji podczas edycji poprzedniego programu w SFC. Domyślnie opcja ta jest wyłączona więc podczas tworzenia pierwszego programu krok początkowy jest krokiem uproszczonym. Zmiana typu kroku już wstawionego nie jest możliwa więc jeżeli pierwszy krok ma być krokiem IEC należy utworzyć ponownie projekt z programem w języku SFC.



Porównując rysunki przedstawiające projekty z programami w SFC można zauważyć różnicę pomiędzy krokiem uproszczonym a krokiem IEC – po prawej stronie kroku IEC wyświetlane są informacje o zmiennych i akcjach skojarzonych z krokiem (domyślnie z krokiem kojarzona jest akcja lub zmienna

nieprzechowywana (N) o nazwie Action_1). Kojarzona z krokiem IEC zmienna może być kojarzona z krokiem przy pomocy kwalifikatorów:

N – nieprzechowywana (normalna), zmienna taka otrzymuje wartość *prawda* na czas aktywności kroku a wartość *fałsz* po utracie aktywności przez krok,

S – przechowywana, zmiennej przypisywana jest wartość *prawda* po uaktywnieniu kroku, zmienna utrzymuje swoją wartość nawet po utracie aktywności przez krok,

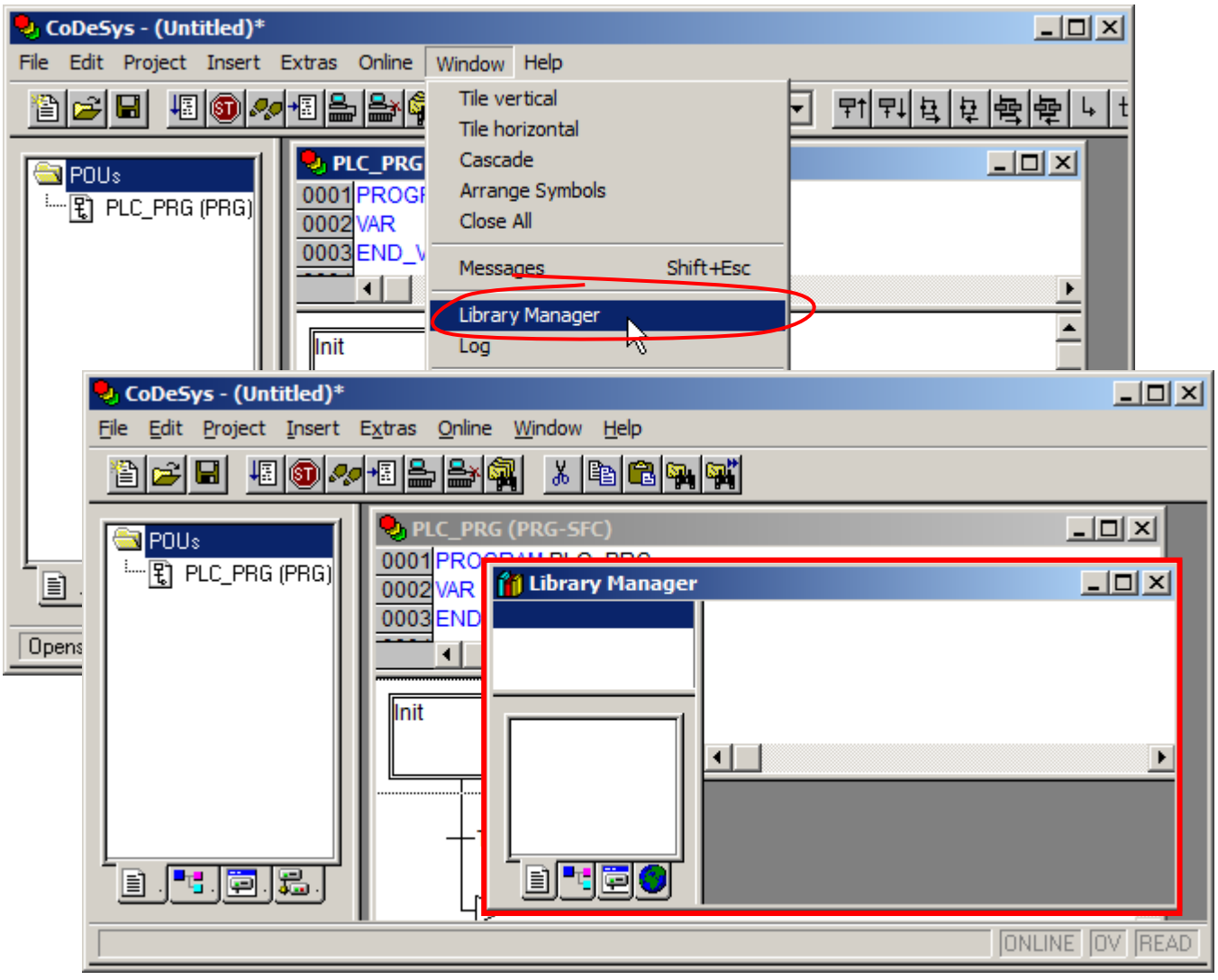
R – przechowywana, zmiennej przypisywana jest wartość *fałsz* po uaktywnieniu kroku, zmienna utrzymuje swoją wartość nawet po utracie aktywności przez krok.

Uwaga!

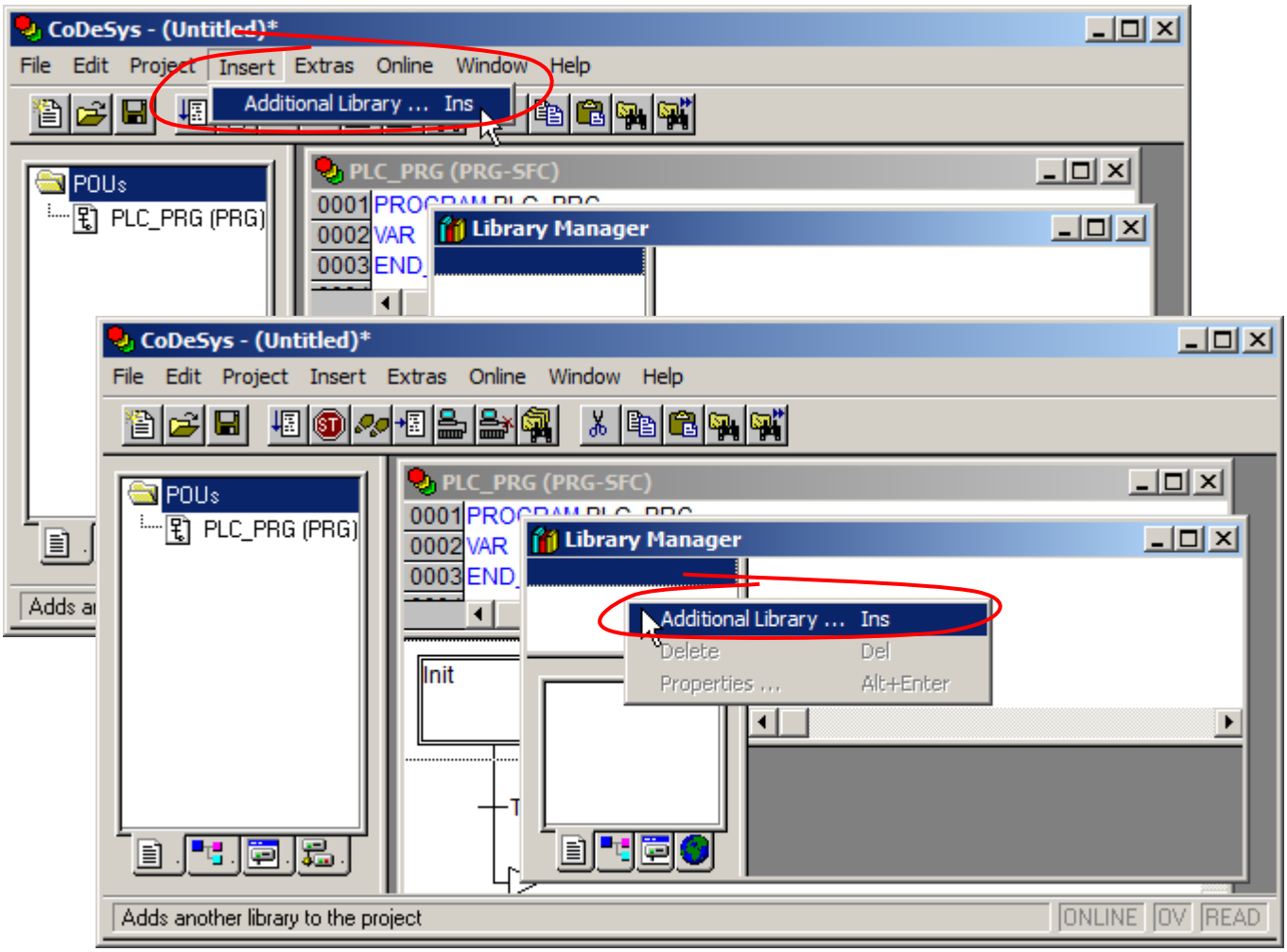
W programie nie jest możliwe skojarzenie z krokiem zmiennej nieprzechowywanej przy pomocy symbolu „/” pozwalającego zmiennej nadać wartość *fałsz* na czas aktywności kroku – nie stanowi to jednak problemu ponieważ, jeżeli program w wybranych krokach ma nadawać zmiennej wartość *prawda* a w pozostałych wartość *fałsz* – można zmienną skojarzyć wyłącznie z krokami, w których powinna ona mieć wartość *prawda* (w pozostałych krokach zgodnie z opisem kwalifikatora N – zmienna otrzymuje wartość *fałsz*).

Biblioteka Iecsfc.lib

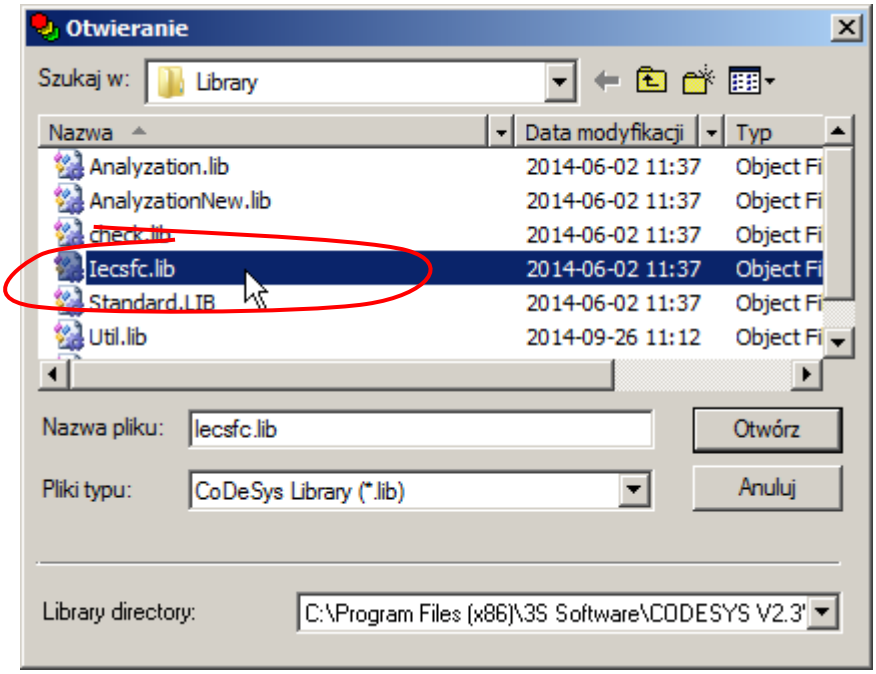
Korzystanie z kroków IEC wiąże się w CoDeSys z koniecznością dołączenia biblioteki „Iecsfc.lib”. Dodatkowe biblioteki mogą być dołączane do projektu z wykorzystaniem menadżera bibliotek:



Po uaktywnienie menadżera, zarówno w menu głównym programu jak i w menu podręcznym menadżera staje się dostępna opcja umożliwiająca dodanie nowej biblioteki.

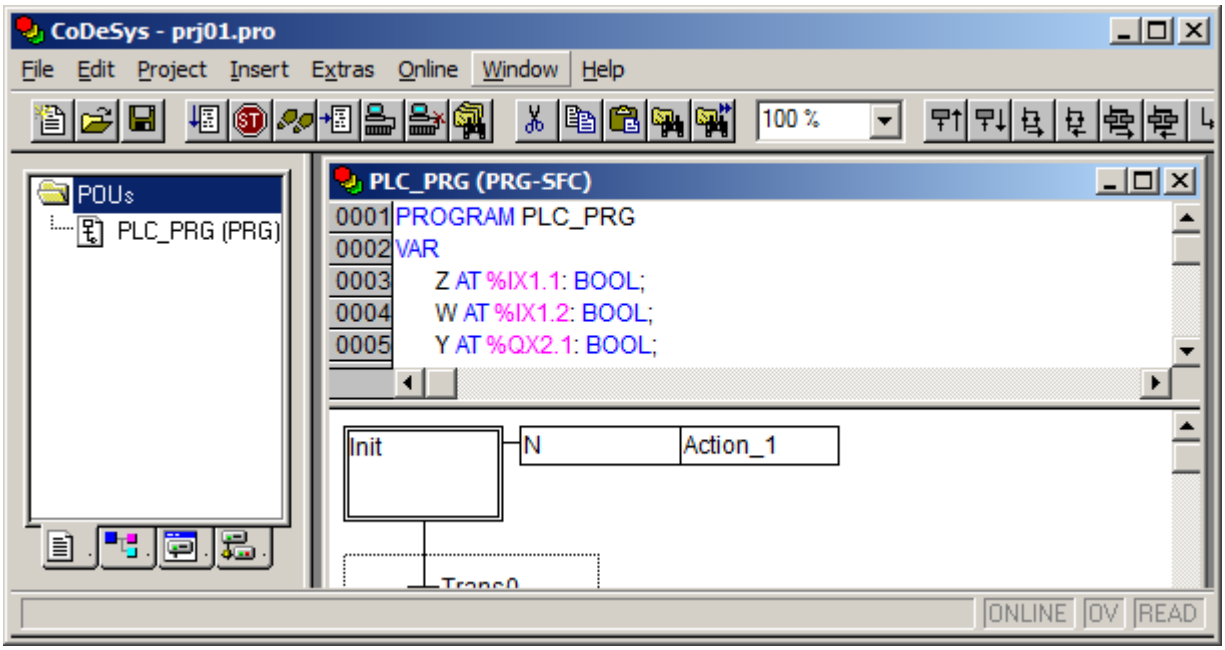


Po wskazaniu właściwej biblioteki okno menadżera może zostać zamknięte.



Edycja programu

Po zdefiniowaniu zmiennych należy uzupełnić program

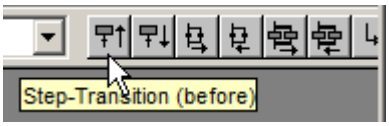


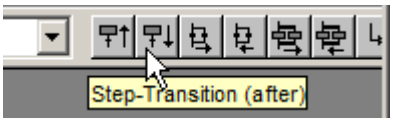
dodając:

- brakujący krok i tranzycję,
- warunki przejścia dla tranzycji,
- kojarząc z krokami zmienne.

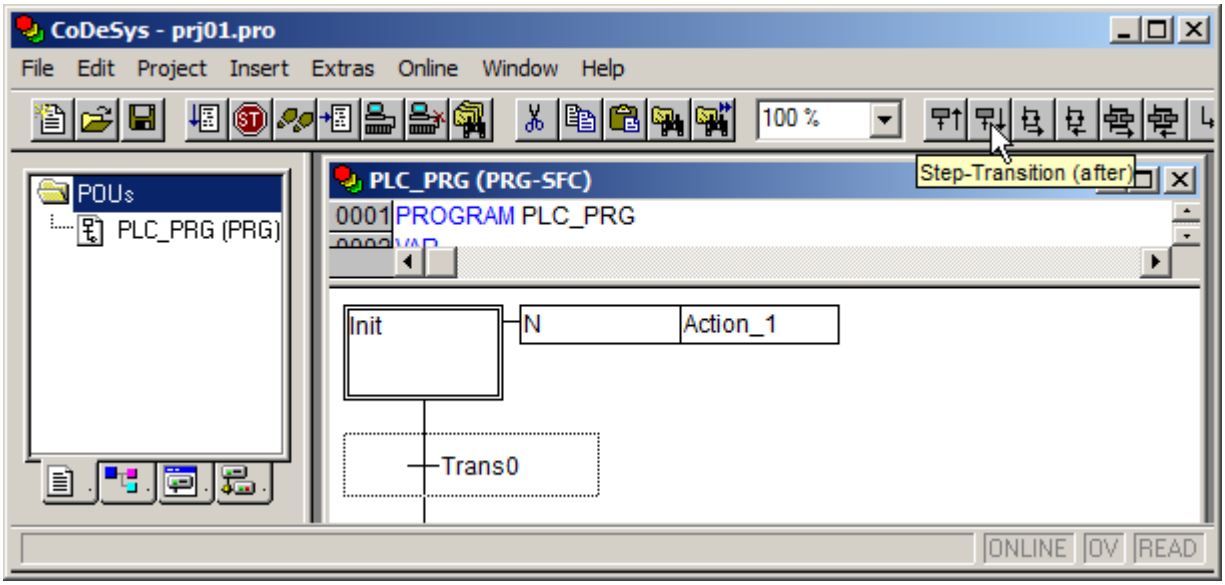
Dodanie pary „krok – tranzycja”

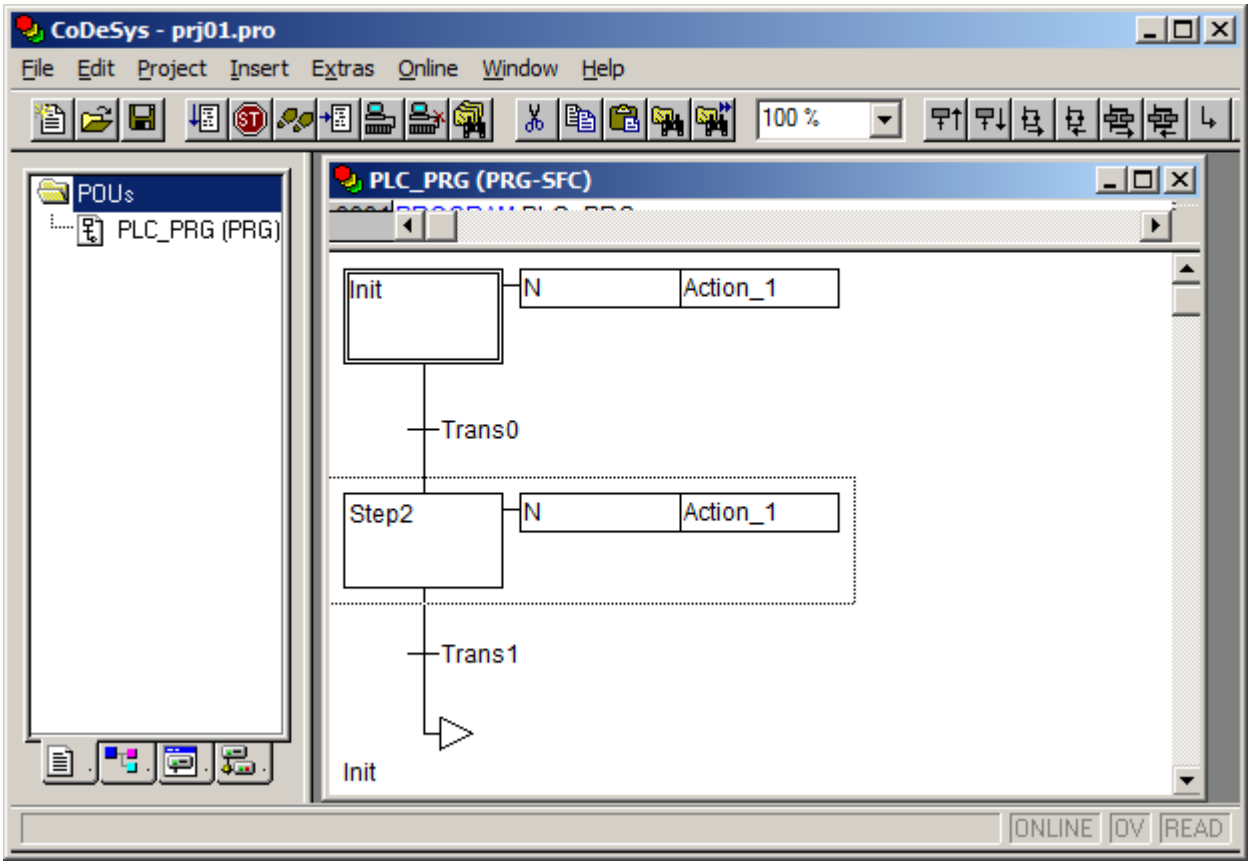
Para „*krok* – *tranzycja*” może zostać dodana przed lub po bieżącym kroku lub bieżącej tranzycji.





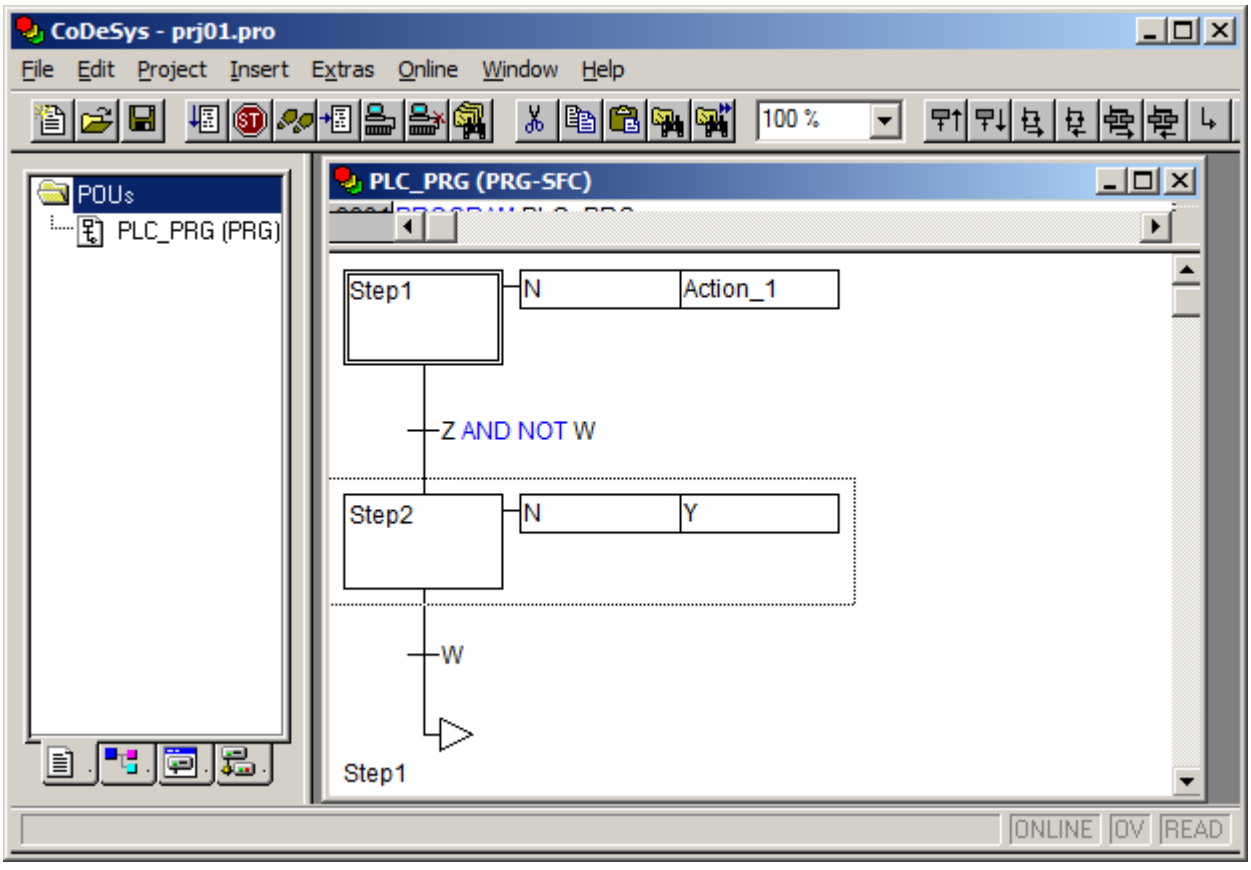
Na poniższym rysunku para „*krok* – *tranzycja*” zostanie wstawiona po tranzycji „Trans0”.



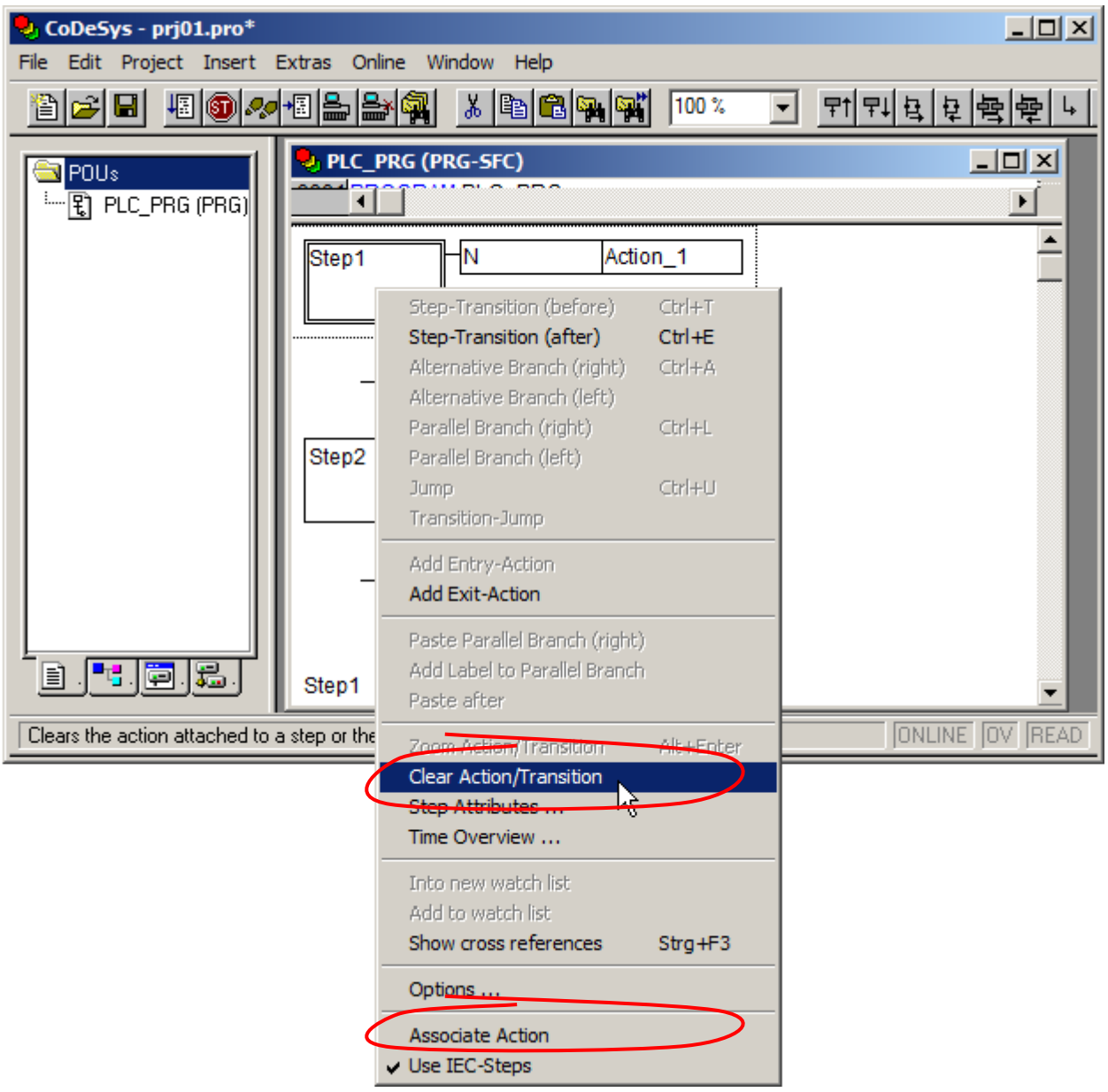


Edycja

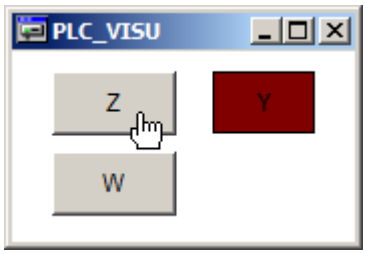
Nazwy kroków, warunki przejścia, nazwy zmiennych i ich kwalifikatory mogą być edytowane wprost z okna edytora programu. Stan programu po dokonaniu zmian został przedstawiony na poniższym rysunku.



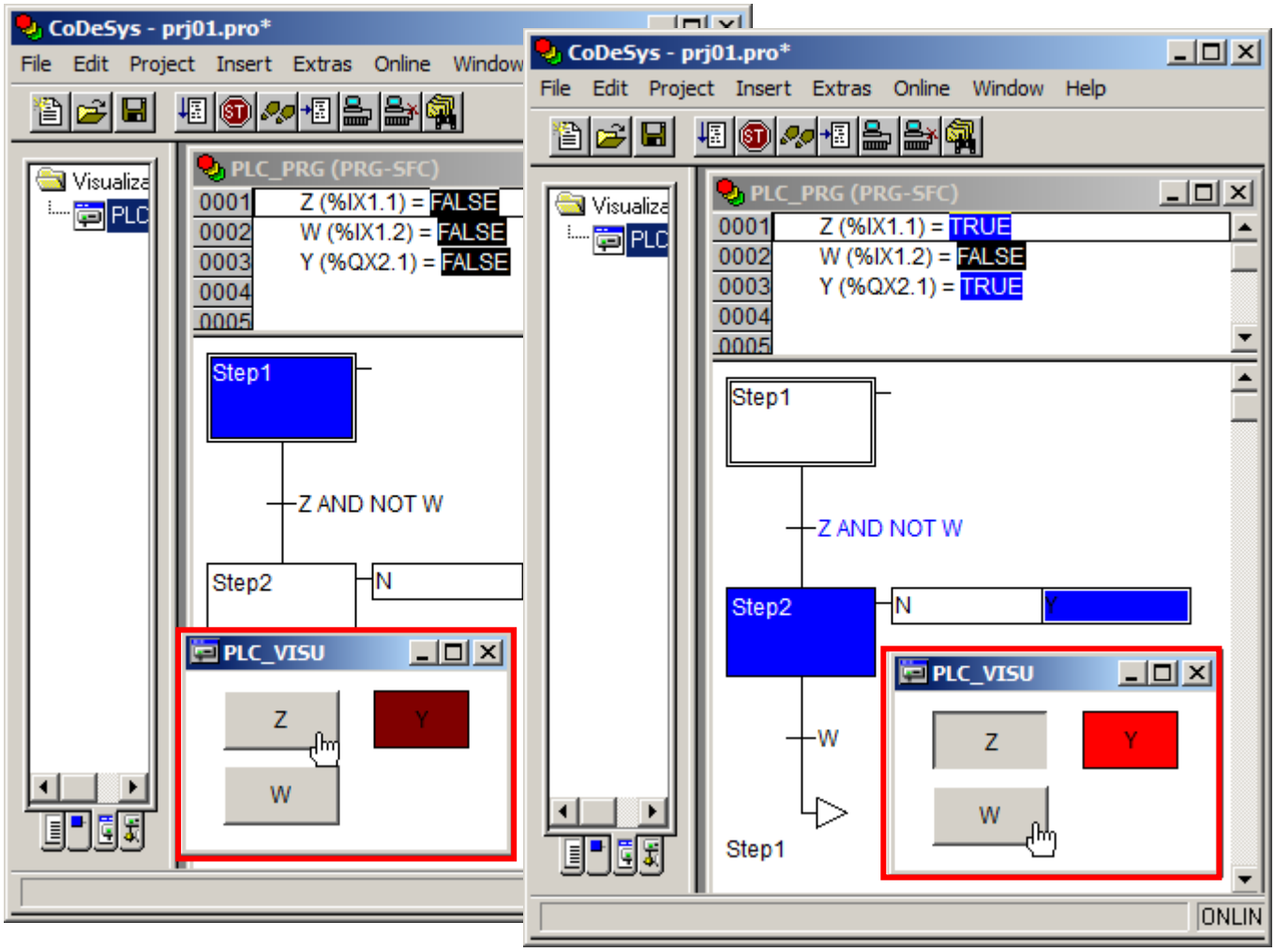
Z pierwszym krokiem nie musi zostać skojarzona żadna zmienna (skojarzenie zmiennej `Y` z kwalifikatorem N w kroku nr 2 nada zmiennej `Y` w kroku 1. wartość *fałsz*). Kroki IEC mogą być w CodeSys kojarzone z maksymalnie dziewięcioma akcjami lub zmiennymi. Usunięci skojarzenia umożliwia opcja: „*Clear Action/Transition”*, dodanie skojarzenia opcja: „*Associate Action*”.



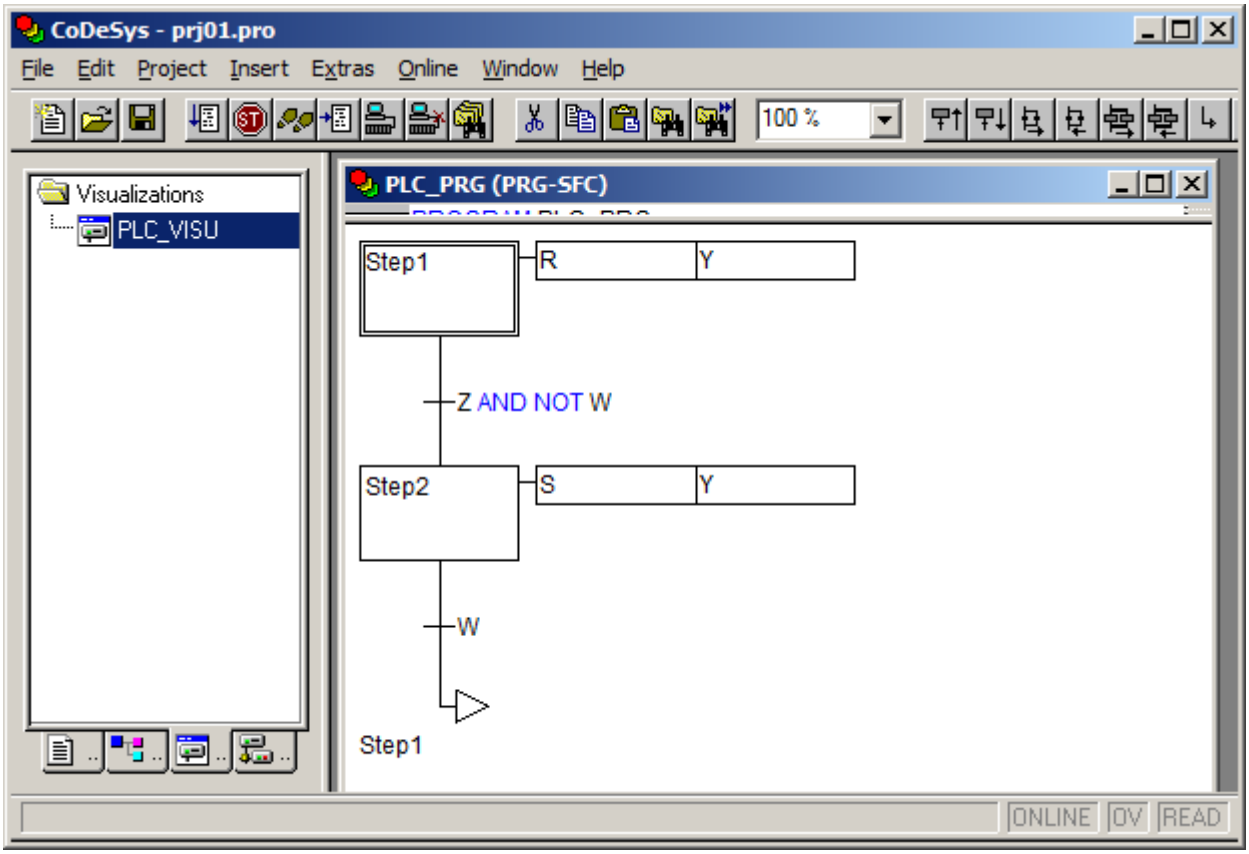
Po usunięciu skojarzenia z kroku pierwszego program jest już gotowy. W celu ułatwienia jego testowania została jeszcze przygotowana wizualizacja:



Po uruchomieniu, aktywny krok jest w oknie edytora programu wyróżniany na niebiesko.

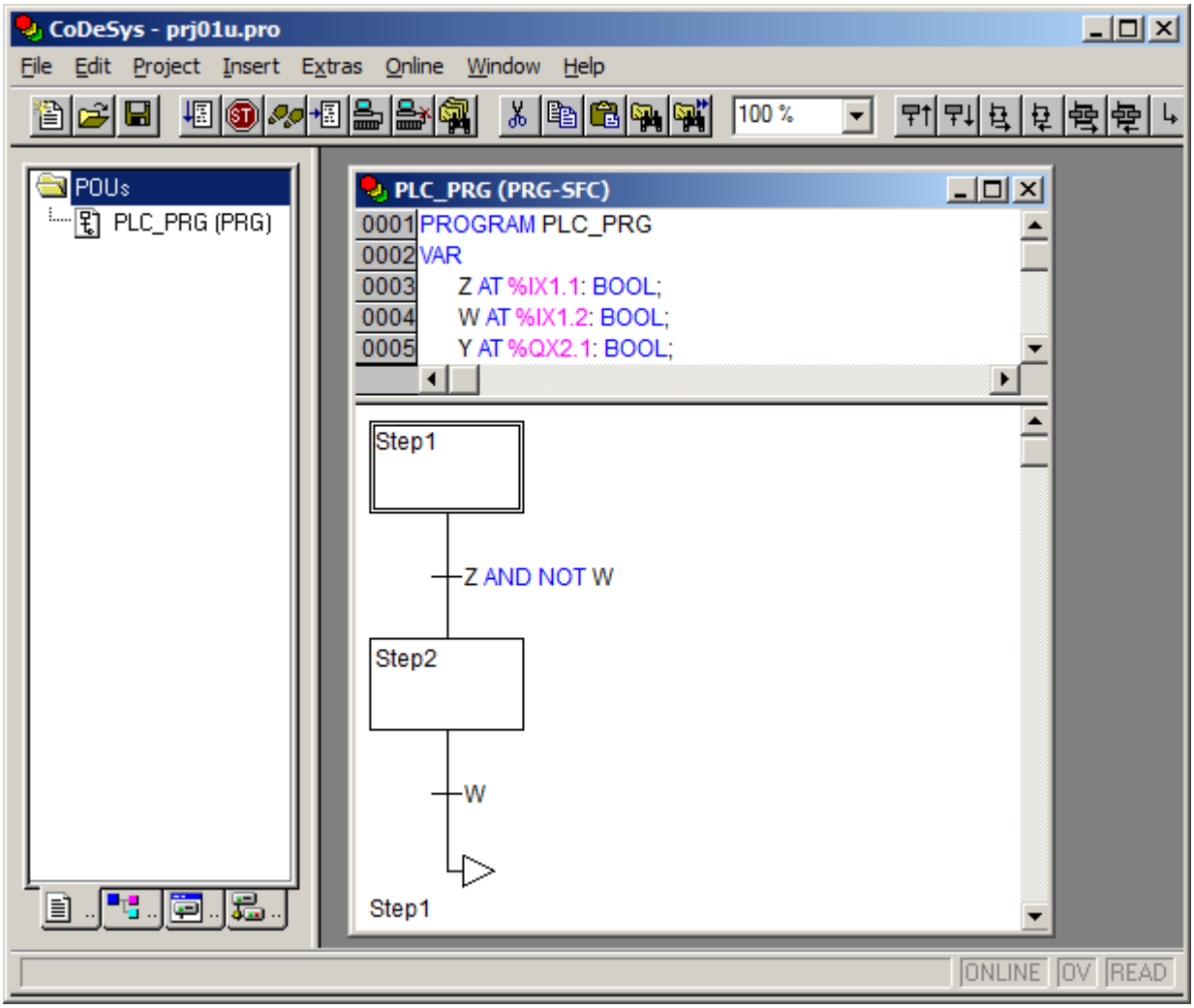


Program można byłoby napisać również w alternatywny sposób z wykorzystaniem kwalifikatorów: R i S.

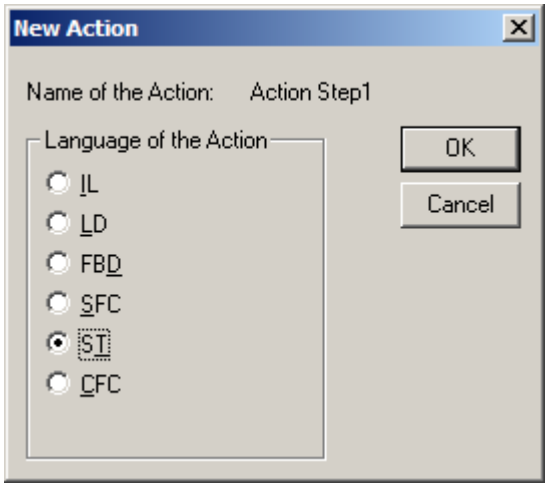


2.1.2. Program z uproszonymi krokami

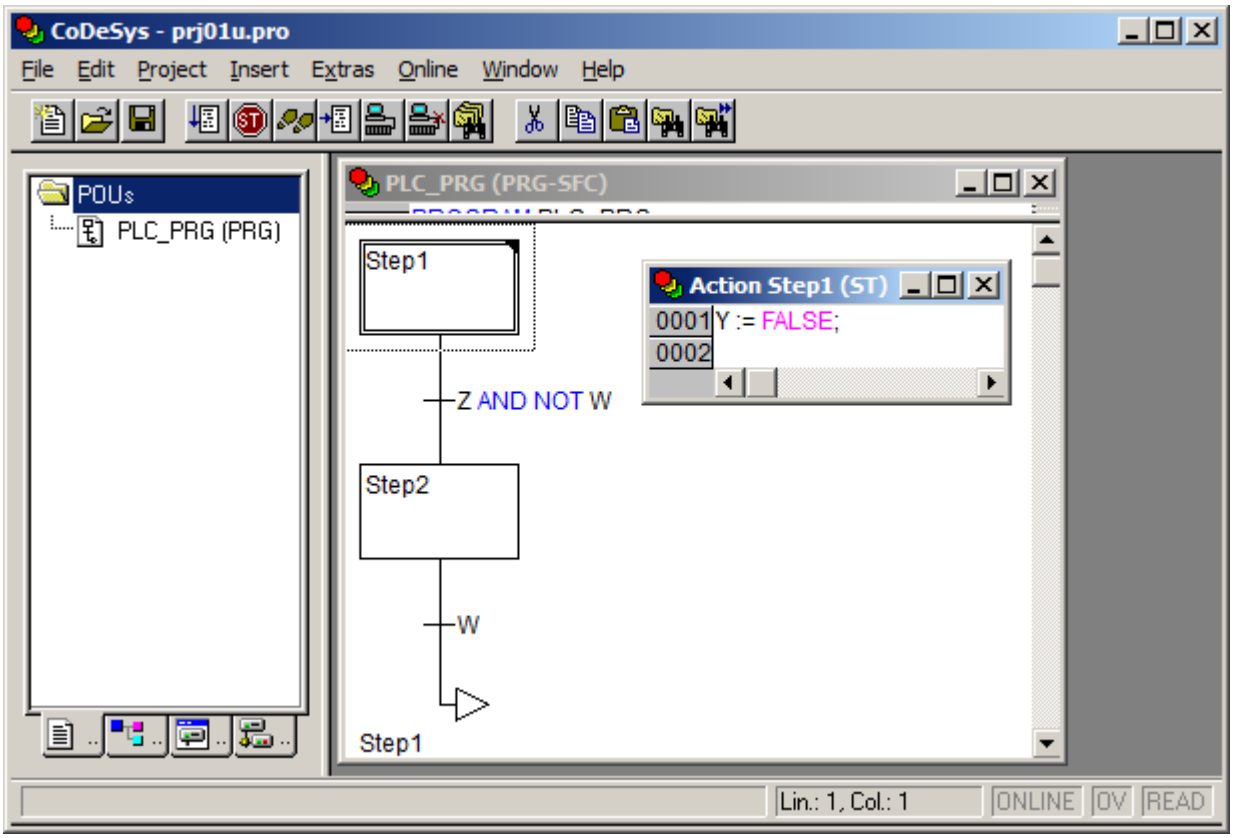
Korzystanie z kroków uproszczonych nie wymaga w CodeSys żadnych dodatkowych zabiegów. Konieczne jest natomiast napisanie akcji dla każdego z kroków. Na poniższym rysunku przedstawiony został program napisany z użyciem kroków uproszonych.



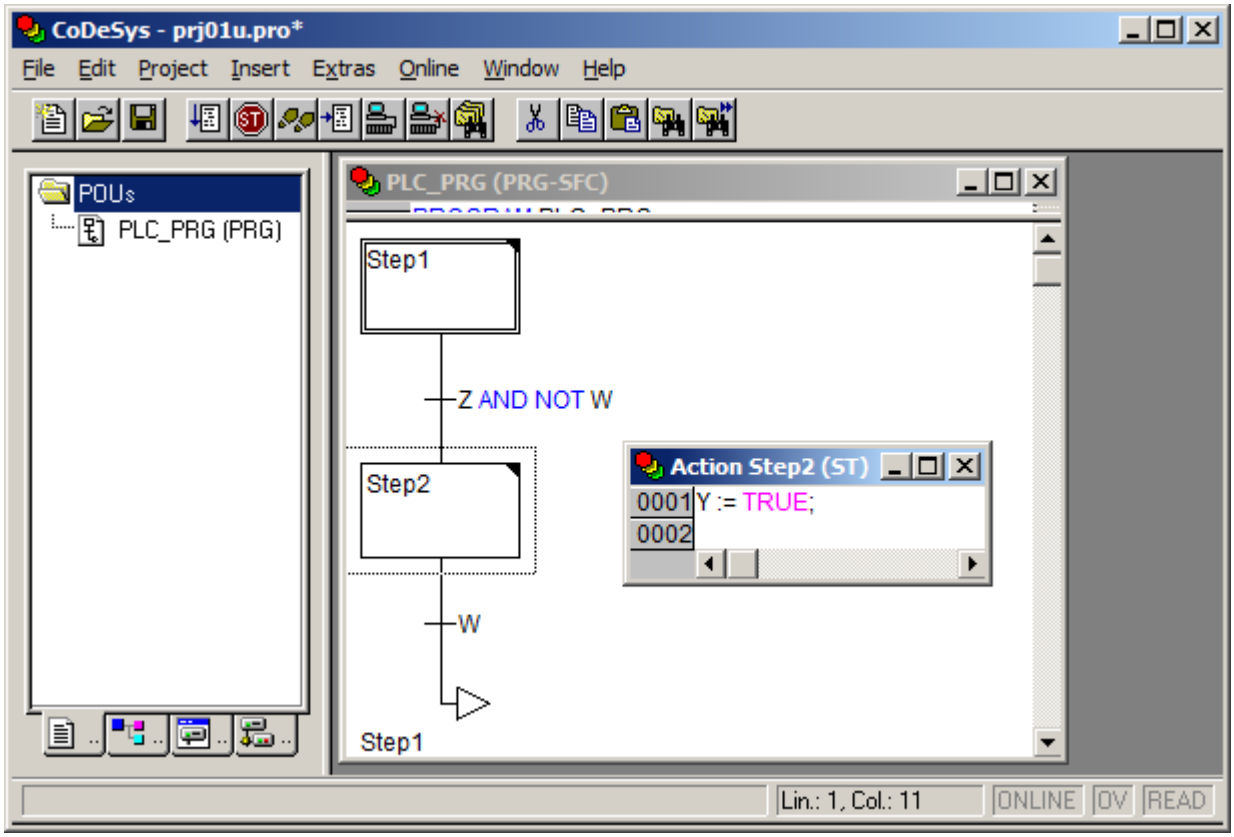
Po dwukrotnym kliknięciu na kroku otwierane jest okno pozwalające na ustalenie języka, w którym zostanie napisana akcja – w omawianym przykładzie akcja zostanie napisana w podobnym do Pascala języku ST.



Po wybraniu języka otwierany jest odpowiedni edytor pozwalający na zapisanie akcji – na poniższym rysunku akcję dla kroku pierwszego zapisano w postaci polecenia: `Y := FALSE;`

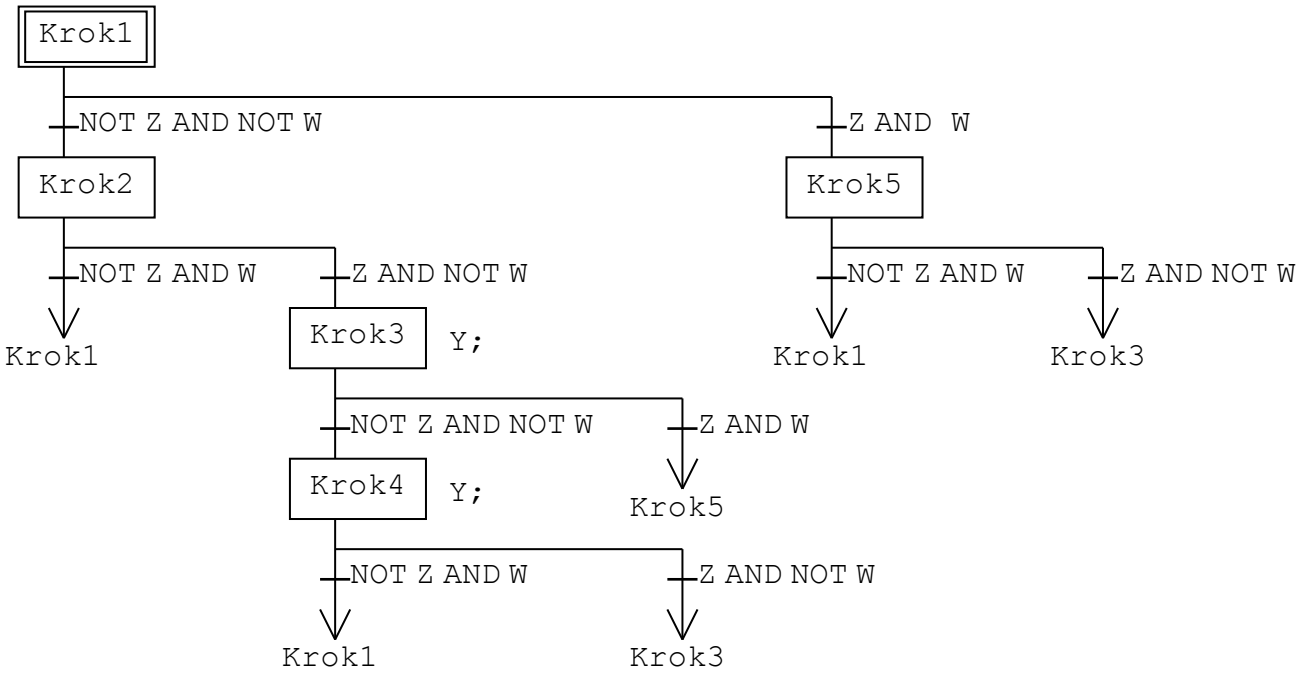


Kroki z przypisanym akcjami zaznaczane są w oknie edytora w postaci prostokąta z zamalowanym górnym prawym narożnikiem. Po ustaleniu akcji dla kroku drugiego w postaci polecenia: `Y := TRUE;` program jest już gotowy.



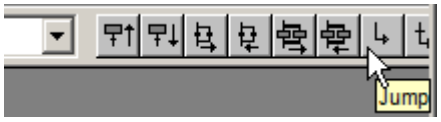
3. Zadanie 2.

Program załączający i wyłączający urządzenie został zrealizowany w poprzednim punkcie w oparciu o zredukowany graf przejść. Gdyby redukcja nie została wykonana program musiałby być narysowany w postaci przedstawionej na *rys. 2*.



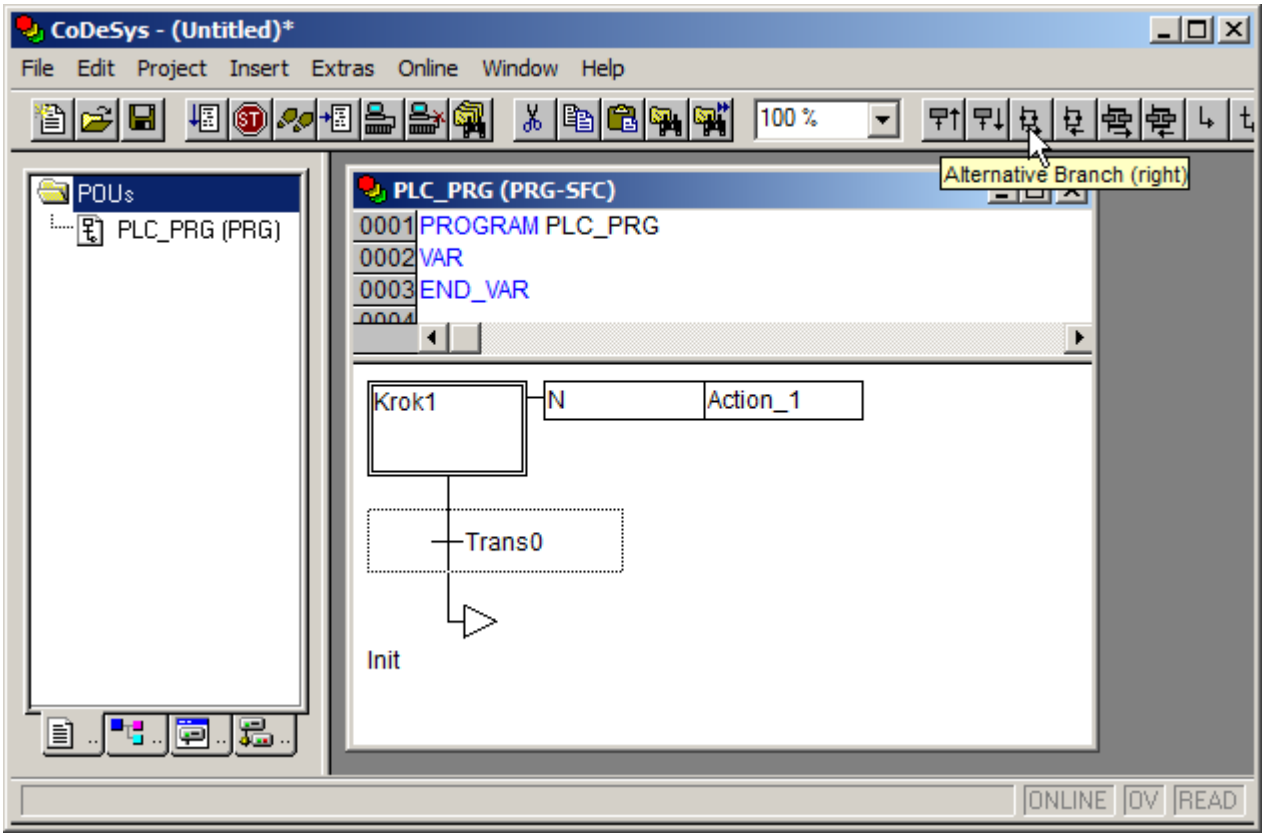
Rys. 2. Przykładowy program w języku SFC.

Program w tej wersji wymaga użycia *symboli rozbieżności* (gałąź alternatywna może być dodana z prawej lub lewej strony bieżącej tranzycji) oraz *symbolu skoku do kroku* (który może być dodany po bieżącej tranzycji).

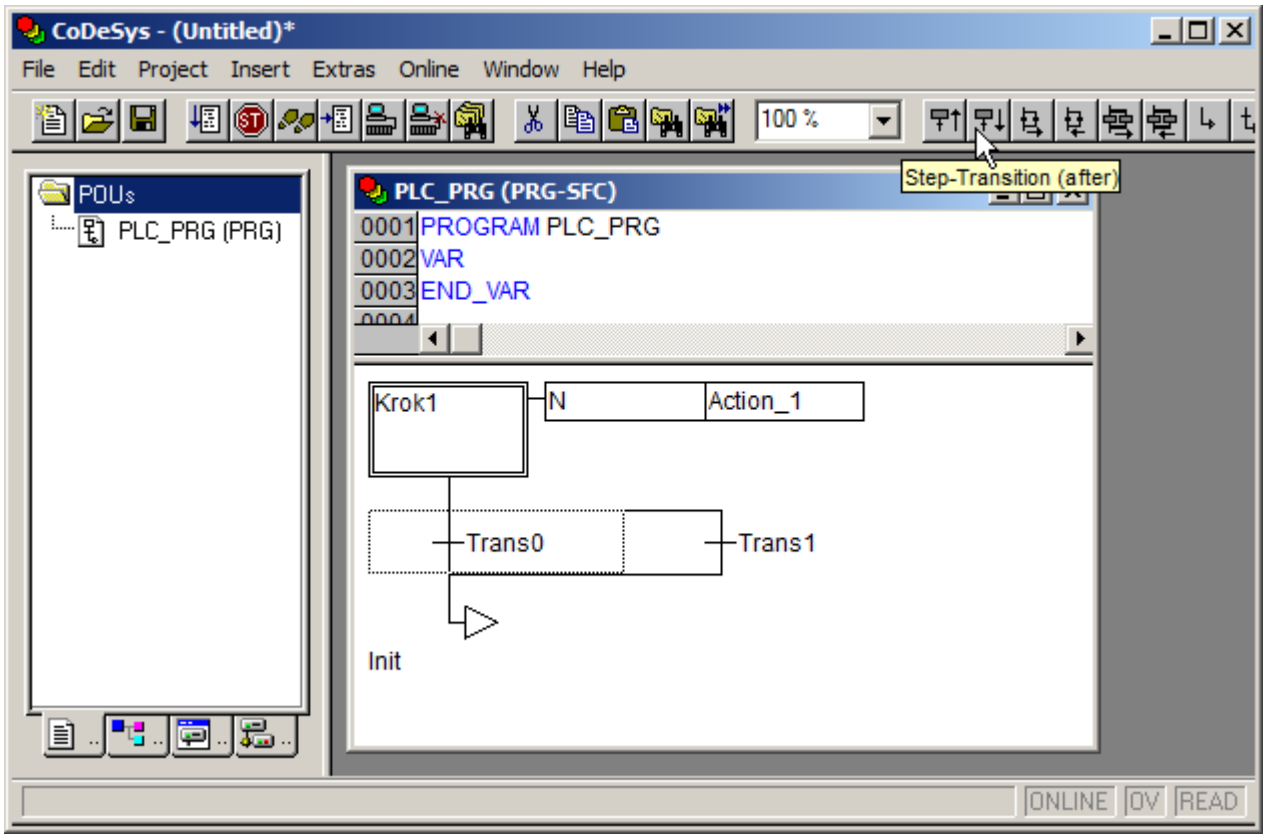


Dodanie gałęzi alternatywnej powoduje automatyczne uzupełnienie programu o symbole rozbieżności i zbieżności. Łączenie gałęzi alternatywnych jest przerywane po wstawieniu *symbolu skoku do kroku*.

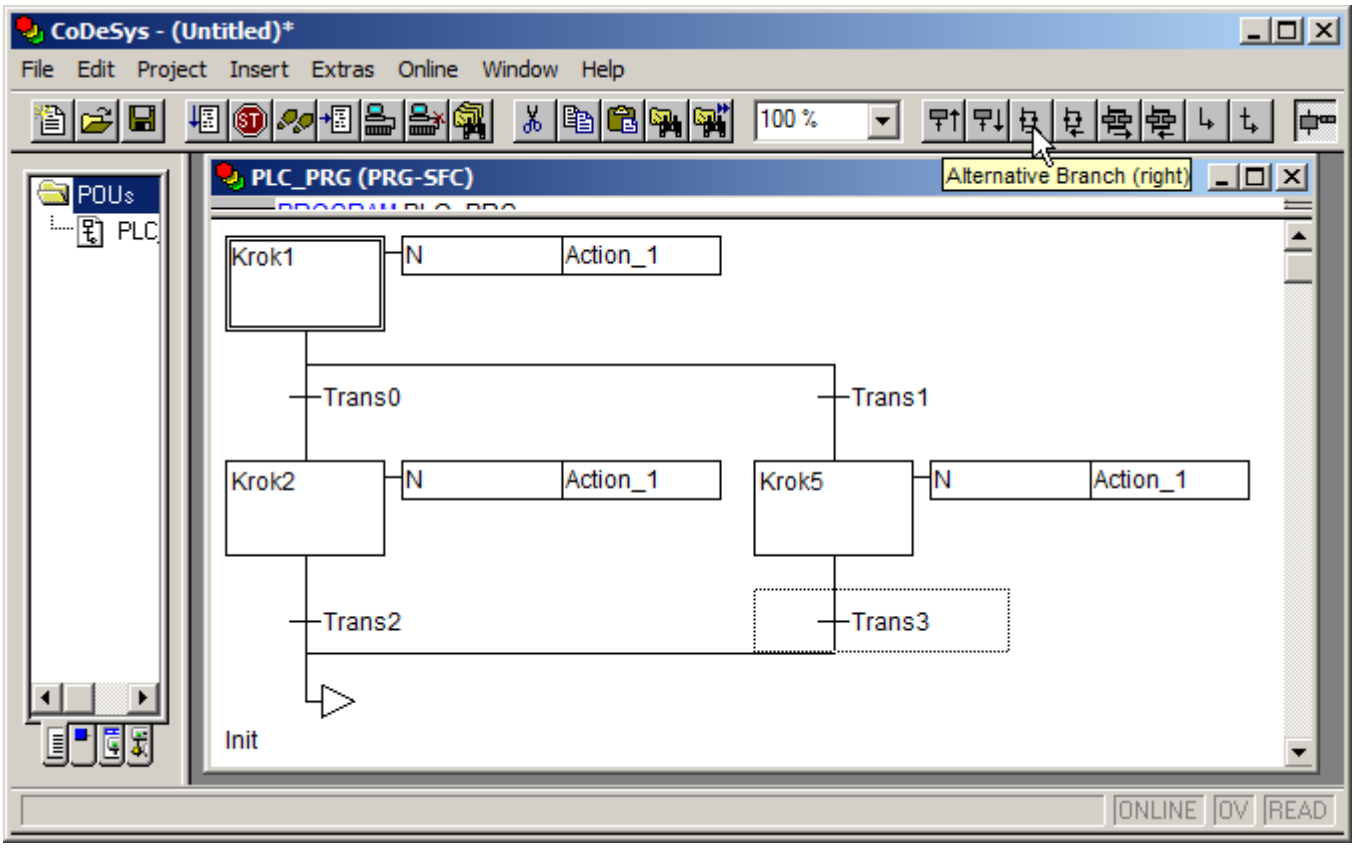
W rozważanym programie, w każdym przypadku, po kroku występują dwie alternatywne gałęzie. Na początek zostanie dodana gałąź po kroku pierwszym.



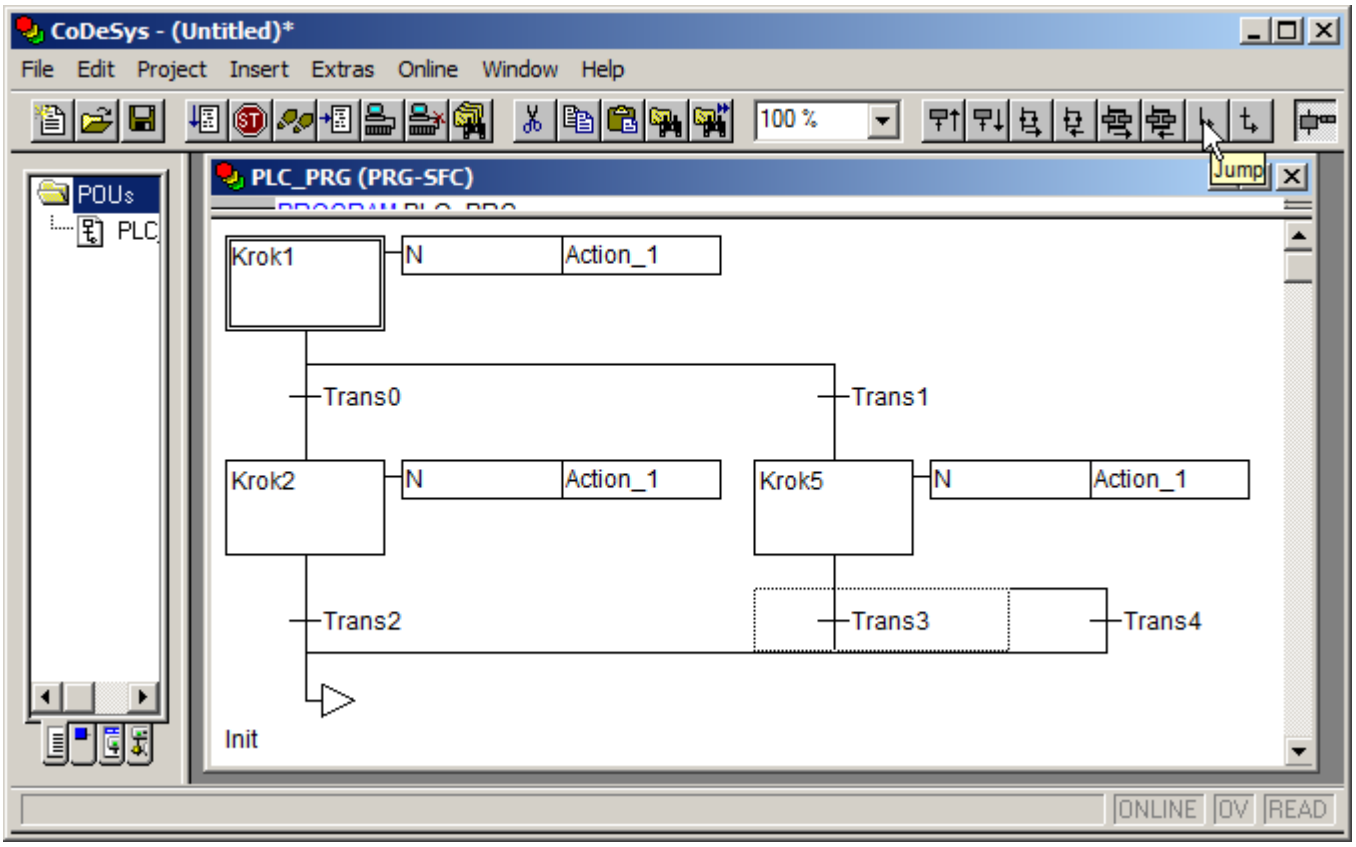
Po dodaniu gałęzi alternatywnej można uzupełnić program o `Krok2` i `Krok5`, które są aktywowane po spełnieniu warunków tradycji w gałęziach alternatywnych.

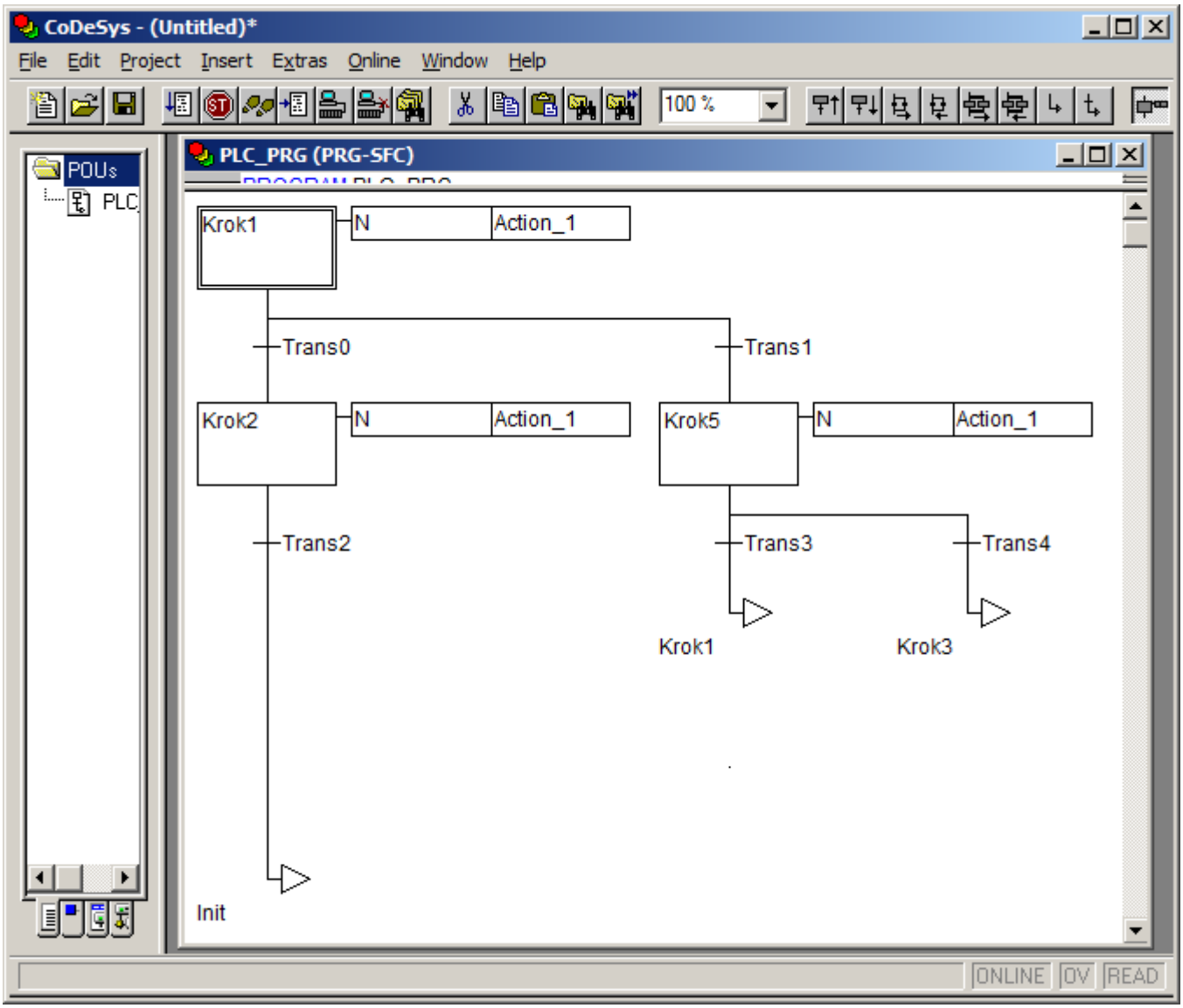


Po dodaniu kroków drugiego i piątego można uzupełnić graf programu o gałęzie alternatywne po kroku piątym.



Po spełnieniu warunków po tranzycjach w kroku piątym następują skoki do kroków pierwszego i trzeciego.





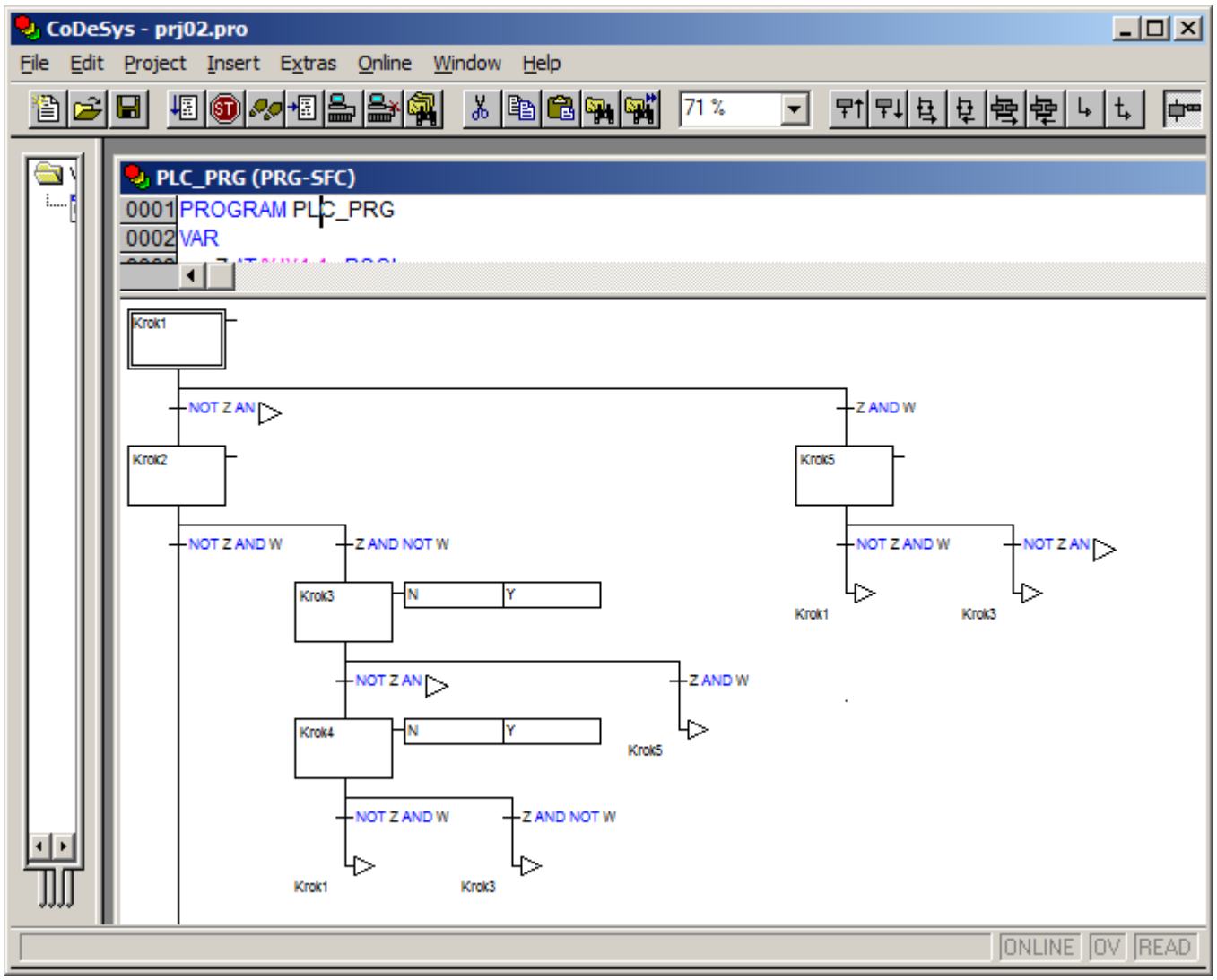
Po dodaniu rozgałęzienia za krokiem drugim, dodaniu kroku trzeciego i rozgałęzienia za krokiem trzecim, następnie kroku czwartego i rozgałęzienia za czwartym krokiem wystarczy uzupełnić już tylko graf programu o trzy symbole skoku do kroku.

Po narysowaniu grafu należy jeszcze uzupełnić warunki przejścia tranzycji. Ze względu na to, że graf został narysowany z wykorzystaniem kroków IEC na koniec należy skojarzyć zmienne z krokami. W przedstawionym poniżej rozwiązaniu zmienna `Y` została skojarzona z krokami związana z krokami trzecim i czwartym z wykorzystaniem kwalifikatora N.

Gotowy program został przedstawiony na kolejnym rysunku.

Uwaga!

W przypadku błędnego wstawienia kroku lub tranzycji, elementy te mogą zostać usunięte jedynie po uprzednim zaznaczeniu pary: *krok –tranzycja* (nie jest możliwe usunięcie wyłącznie jednego elementu).


CoDeSys - prj02.pro
File Edit Project Insert Extras Online Window Help
71 %
PLC_PRG (PRG-SFC)
0001 PROGRAM PLC_PRG
0002 VAR
Krok1
NOT Z AN
Krok2
NOT Z AND W
Z AND NOT W
Krok3
N
Y
NOT Z AN
Krok4
N
Y
Z AND W
Krok5
NOT Z AND W
Z AND NOT W
Krok1
Krok3
Z AND W
Krok5
NOT Z AND W
NOT Z AN
Krok1
Krok3
ONLINE
OV
READ