PIERWSZY PROGRAM W JĘZYKU FBD

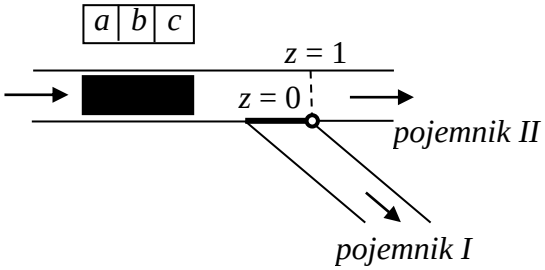
Materiał ten stanowi skróconą wersję opracowania omawiającego pracę w środowisku CoDeSys (plik CoDeSys na stronie przedmiotu). Poniżej przedstawiona została instrukcja przygotowania programu dla zadania sterowania zwrotnicą omówionego w punkcie 1.

1. Zadanie

Układ powinien sterować zwrotnicą z urządzenia sortującego kierującego produkowane detale do jednego z dwóch pojemników. Przed przesunięciem detalu do odpowiedniego pojemnika, badane są przy pomocy odpowiednich czujników trzy cechy (*a*, *b*, *c*) każdego z nich.

Każdy z czujników sygnalizuje zbadaną przez siebie cechę jedną z dwóch wartości (1 – wartość prawidłowa, 0 – wartość nieprawidłowa). Sygnały z czujników monitorujących cechy *a*, *b* i *c* odbierane są kolejno na pierwszym, drugim i trzecim wejściu modułu wejść cyfrowych, który jest włączony do gniazda sterownika o numerze 1.

Detale o co najmniej dwóch prawidłowych cechach, przy dodatkowym założeniu że cecha *a* jest prawidłowa, powinny być kierowane do *pojemnika I* ($z = 1$), pozostałe detale do *pojemnika II* ($z = 0$). Sygnałem sterującym położeniem zwrotnicy jest sygnał z pierwszego wyjścia modułu wyjść cyfrowych, który jest włączony do gniazda o numerze 2.



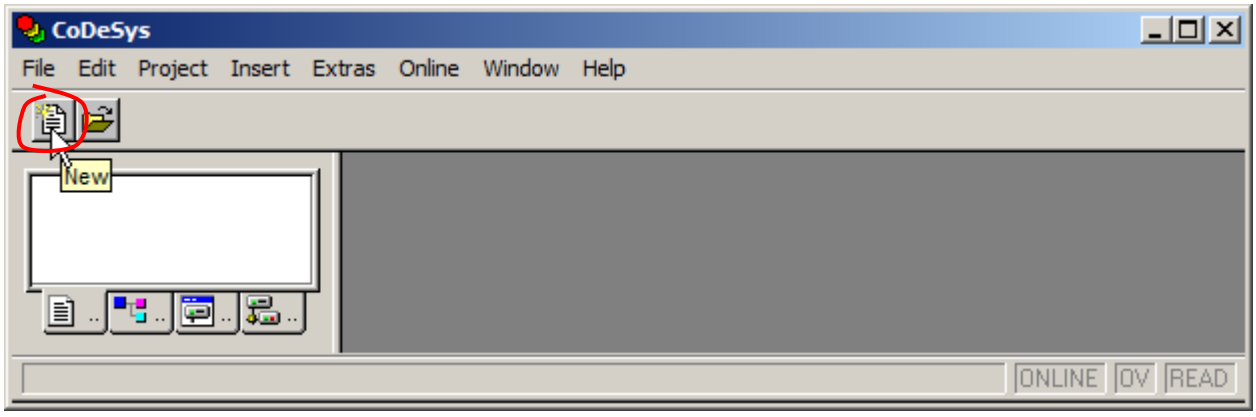
Wartość sygnału sterującego dla zwrotnicy można opisać na przykład funkcją logiczną w postaci kanonicznej:

$$z = abc + ab\bar{c} + a\bar{b}c\,.$$

2. Tworzenie nowego projektu

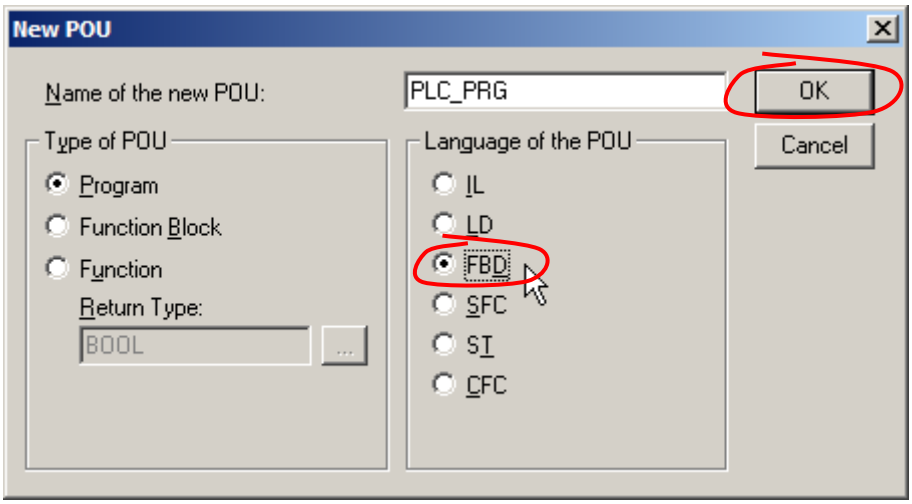
Przygotowanie projektu z programem głównym w języku FBD wymaga wykonania kroków przedstawionych na kolejnych rysunkach (szczegółowe omówienie zagadnienia można znaleźć w materiale CoDeSys zamieszczonym na stronie przedmiotu):

a)

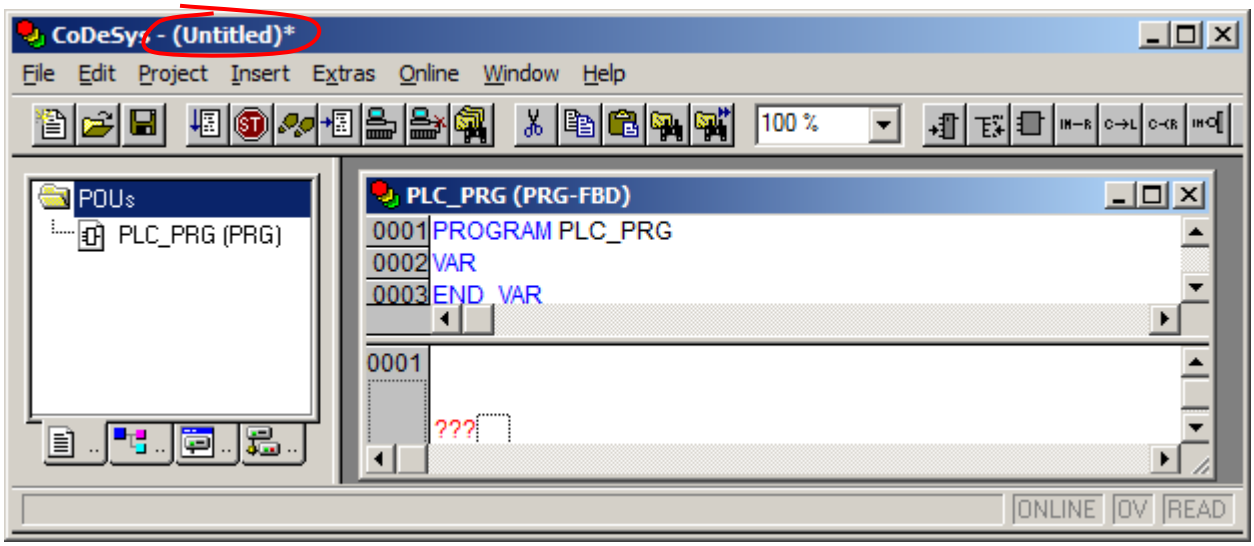


b)

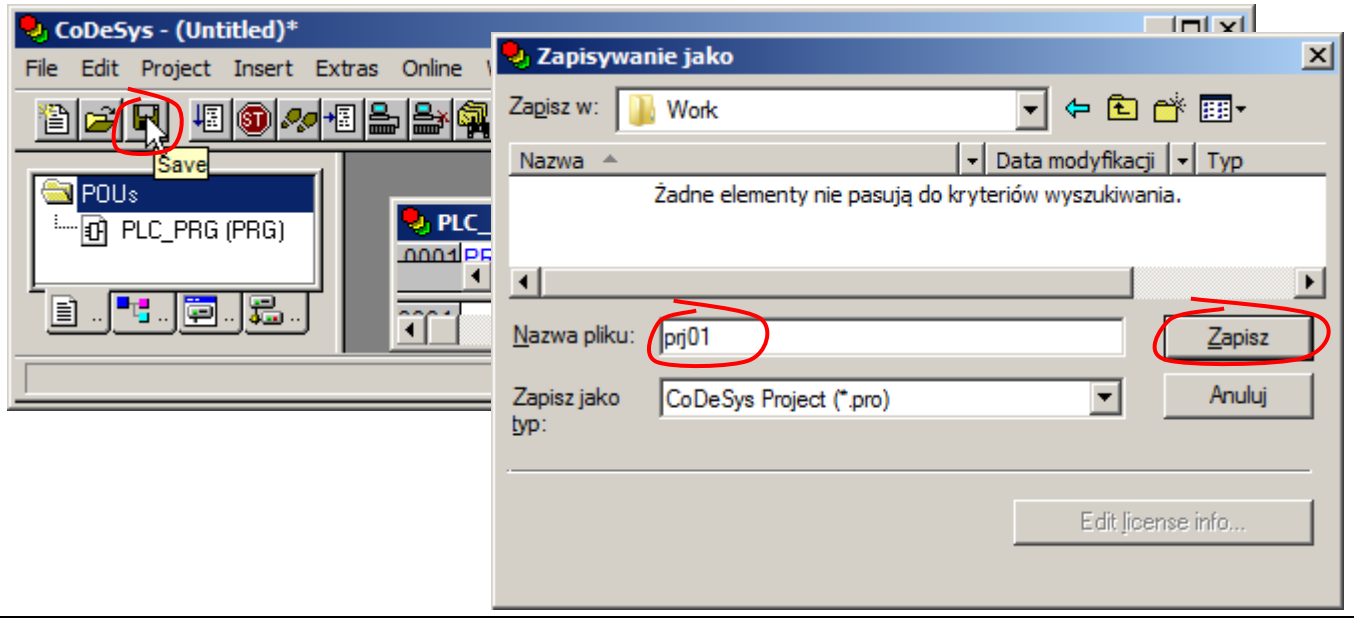
c)



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



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



3. Tworzenie programu

Program w języku FBD ma formę diagramu zawierającego powiązane ze sobą liniami przepływu sygnałów bloki funkcyjne oraz zmienne. Funkcja logiczna zapisana w języku FBD wykorzystuje bloki operacji logicznych:

Symbole graficzne a) koniunkcji b) alternatywy.

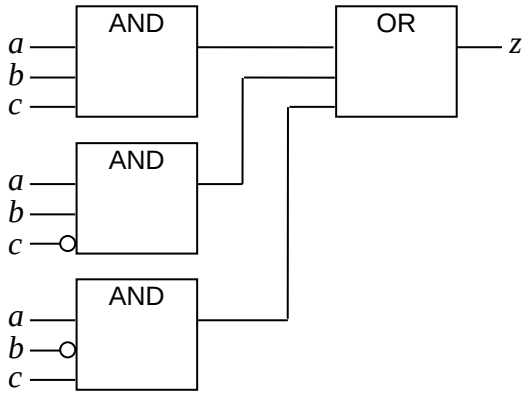
W punkcie 1. zaproponowana została funkcja logiczna opisująca wartość sygnału sterującego:

$$z = abc + ab\bar{c} + a\bar{b}c .$$

Określenie wartości przedstawionej powyżej funkcji wymaga kolejno:

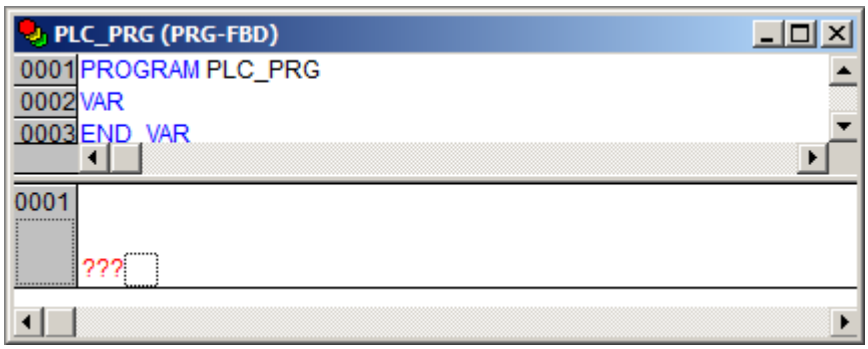
- obliczenia koniunkcji sygnałów wejściowych *a*, *b* i *c*,
- obliczenia koniunkcji sygnałów wejściowych *a*, *b* i zanegowanego *c*,
- obliczenia koniunkcji sygnałów wejściowych *a*, zanegowanego *b* i *c*,
- obliczenia alternatywy wyników powyższych wyrażeń.

Ostatecznie, diagram FBD odpowiadający tej funkcji logicznej został przedstawiony poniższym rysunku.



Program realizujący przykładowe zadanie można wprowadzić w CoDeSys na wiele różnych sposobów (szczegółowe omówienie zagadnienia można znaleźć w materiale CoDeSys – plik na stronie przedmiotu). W omówionym poniżej podejściu najpierw zdefiniowane zostaną zmienne, później krok po kroku zostanie narysowany diagram FBD.

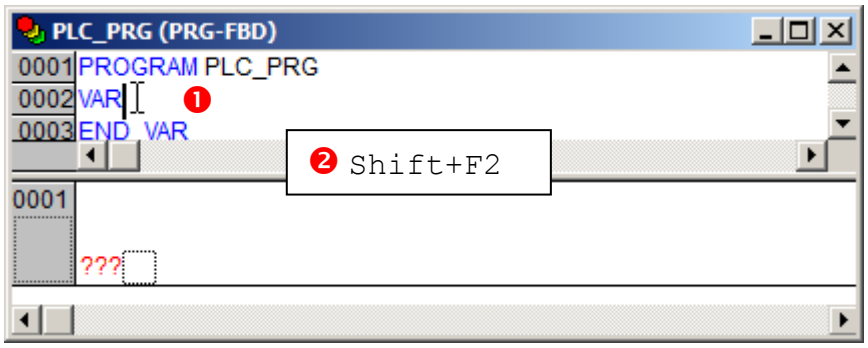
Edycja programu odbywa się z poziomu jego edytora:



Deklaracja zmiennych

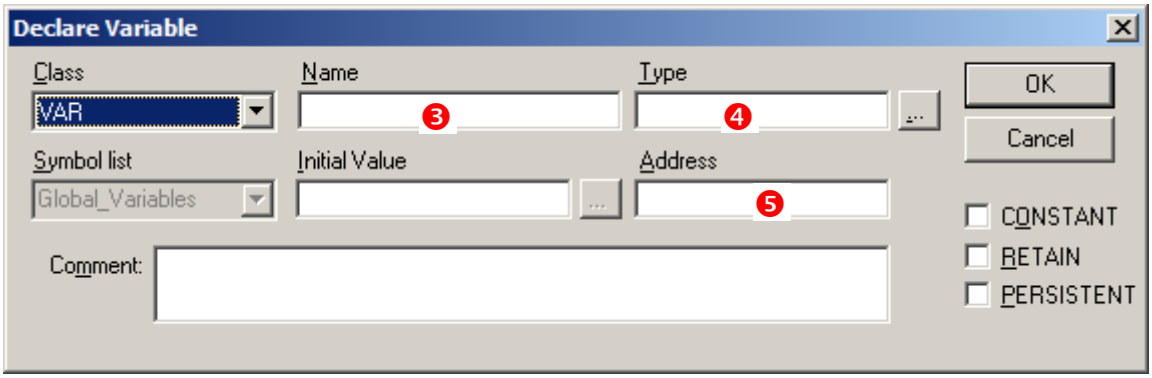
W celu zdefiniowania zmiennych:

1) ustawić kursor w górnej części edytora w linii zawierającej słowo VAR,
2) nacisnąć z klawiatury kombinację klawiszy: `Shift+F2`,

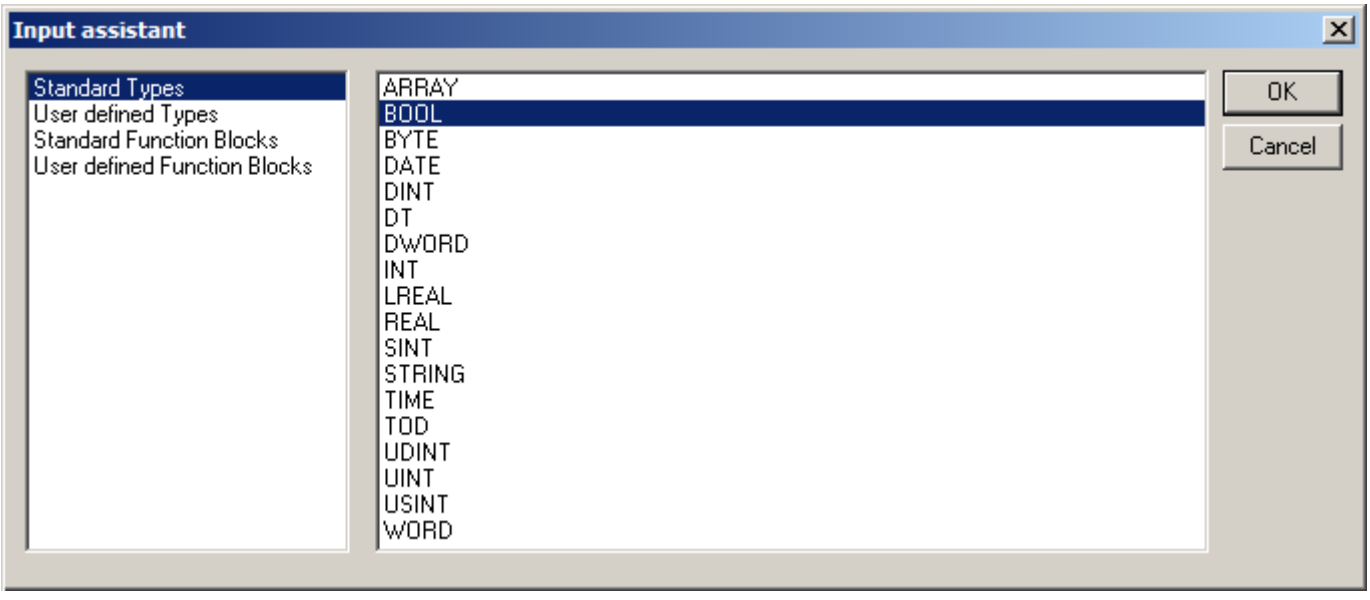


wprowadzić parametry deklarowanych zmiennych.

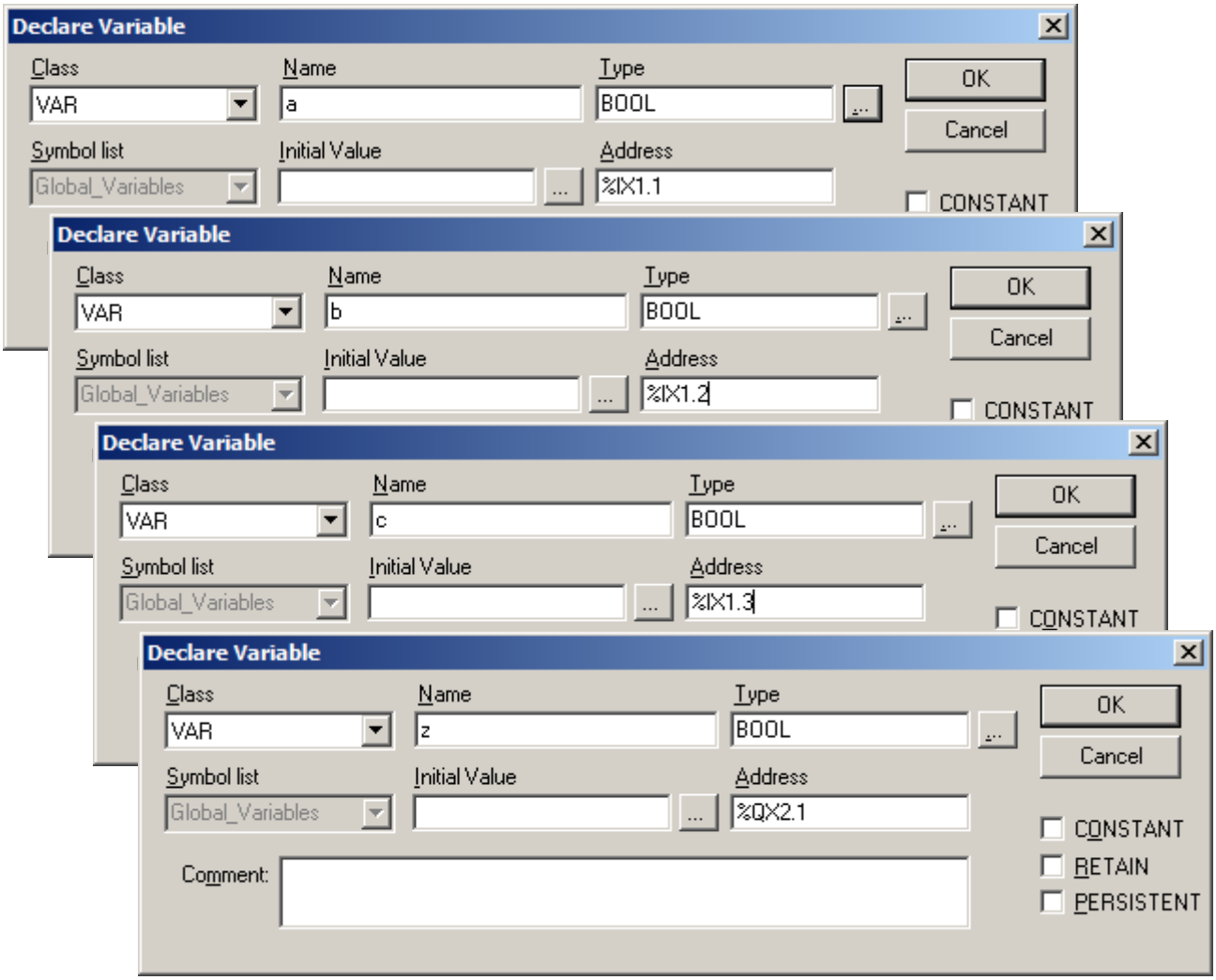
3) nazwę,
4) typ,
5) adres w kartach *We/Wy* sterownika.



Wartości parametrów można wprowadzić wypełniając odpowiednie pola tekstowe okna deklaracji zmiennej, w przypadku wypełniania typu można skorzystać z pomocniczego okna z listą dostępnych typów (dostępnego pod przyciskiem ...).



Zgodnie z treścią zadania w programie należy zdefiniować 4 zmienne o rozmiarze 1 bitu – zmienne takie muszą być zadeklarowane jako zmienne boolowskie, tzn. zmienne typu BOOL. Na poniższych rysunkach pokazane zostały deklaracje zmiennych: *a*, *b*, *c* i *z*, ich adresy zostały ustawione na podstawie opisu w punkcie 1. (zmienne *a*, *b* i *c* odpowiednio: `%IX1.1`, `%IX1.2`, `%IX1.3`, zmienna z: `%QX2.1`).



Po wykonaniu powyższych kroków, deklaracje zmiennych widoczne są w oknie edytora.

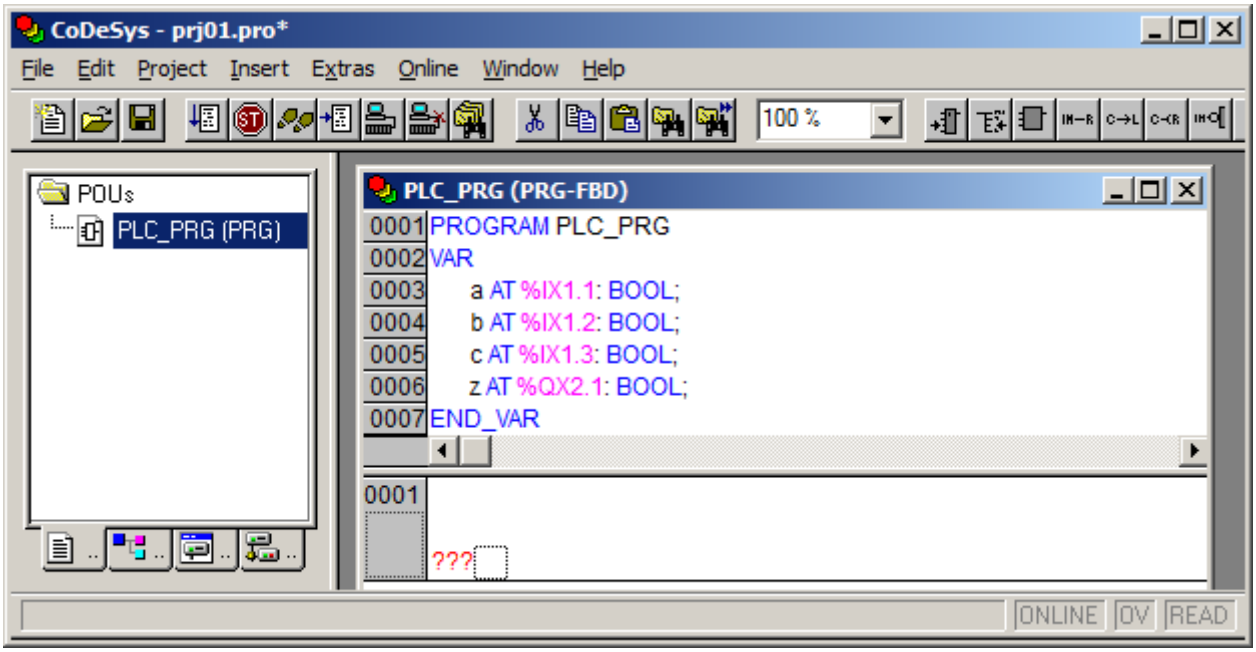


Diagram FBD

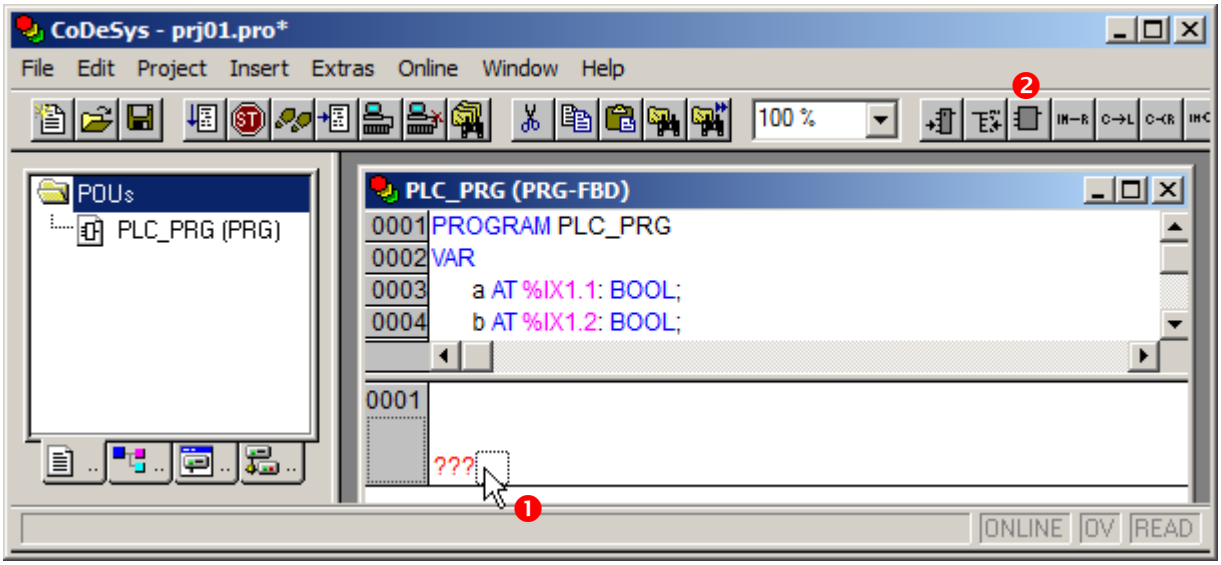
Narysowanie diagramu FBD przykładowego programu sprowadza się do odpowiedniego wstawienia trzech bloków AND i jednego bloku OR. W przedstawionym poniżej sposobie postępowania:

- najpierw wstawiony zostanie pierwszy blok AND,
- następnie za blokiem AND zostanie wstawiony blok OR,
- przed pustymi wejściami bloku OR zostaną wstawione pozostałe bloki AND.

Ze względu na to, że w przykładowym programie wszystkie bloki powinny mieć trzy wejścia a w programie bloki AND i OR wstawiane są domyślnie z dwoma wejściami, po każdorazowym wstawieniu bloku zostanie on uzupełniony o brakujące wejście.

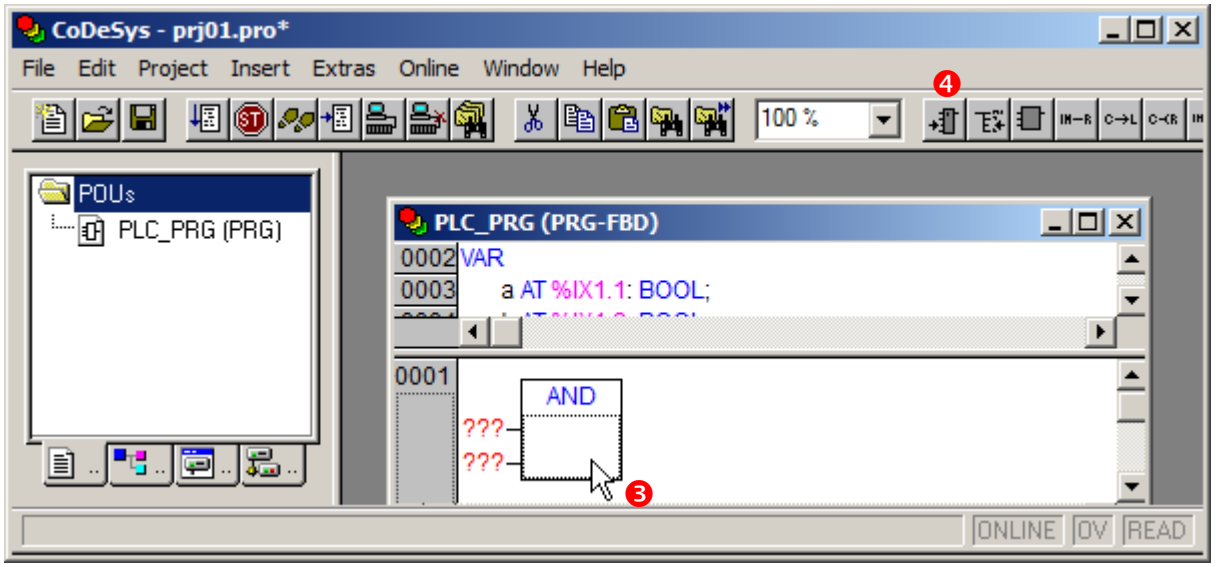
Wstawienie pierwszego bloku AND wymaga:

1) kliknięcia w edytorze programu na pustym polu,
2) wstawienia bloku.



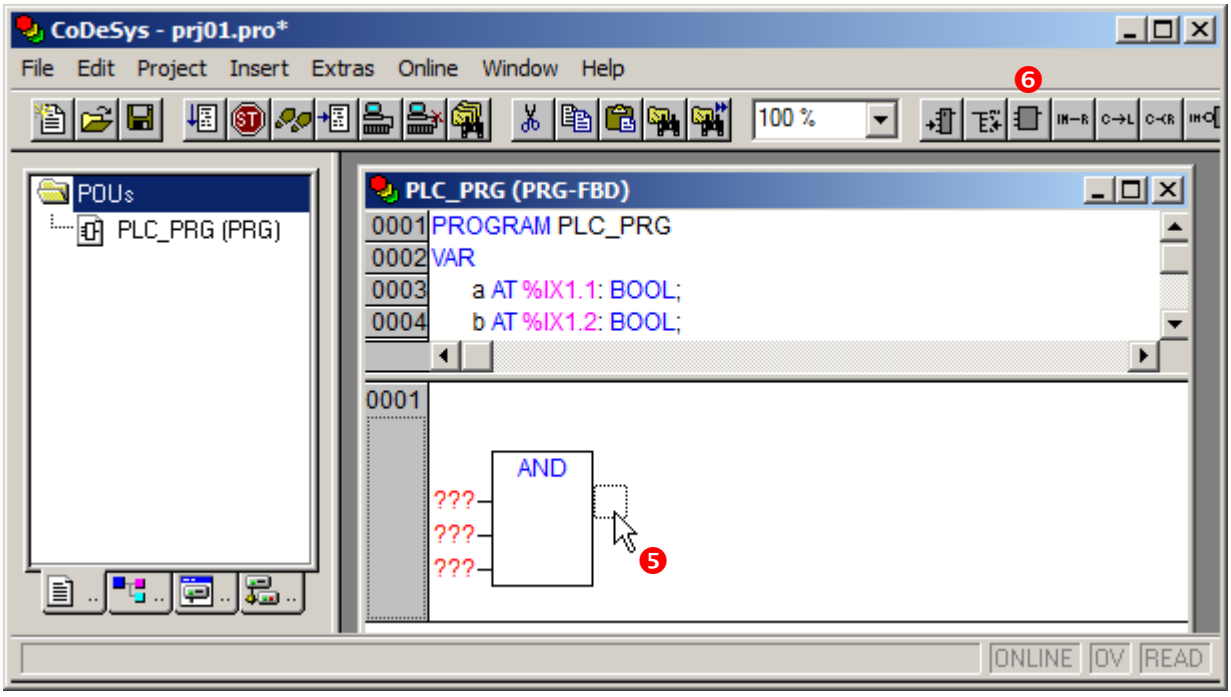
Po wstawieniu bloku, program automatycznie zaznacza jego nazwę pozwalając na jej zmianę (w tym przypadku nie jest to potrzebne), konieczne jest natomiast dodanie trzeciego wejścia, w tym celu należy:

3) kliknąć na bloku (na poniższym rysunku pokazany jest stan po kliknięciu),
4) dodać wejście.

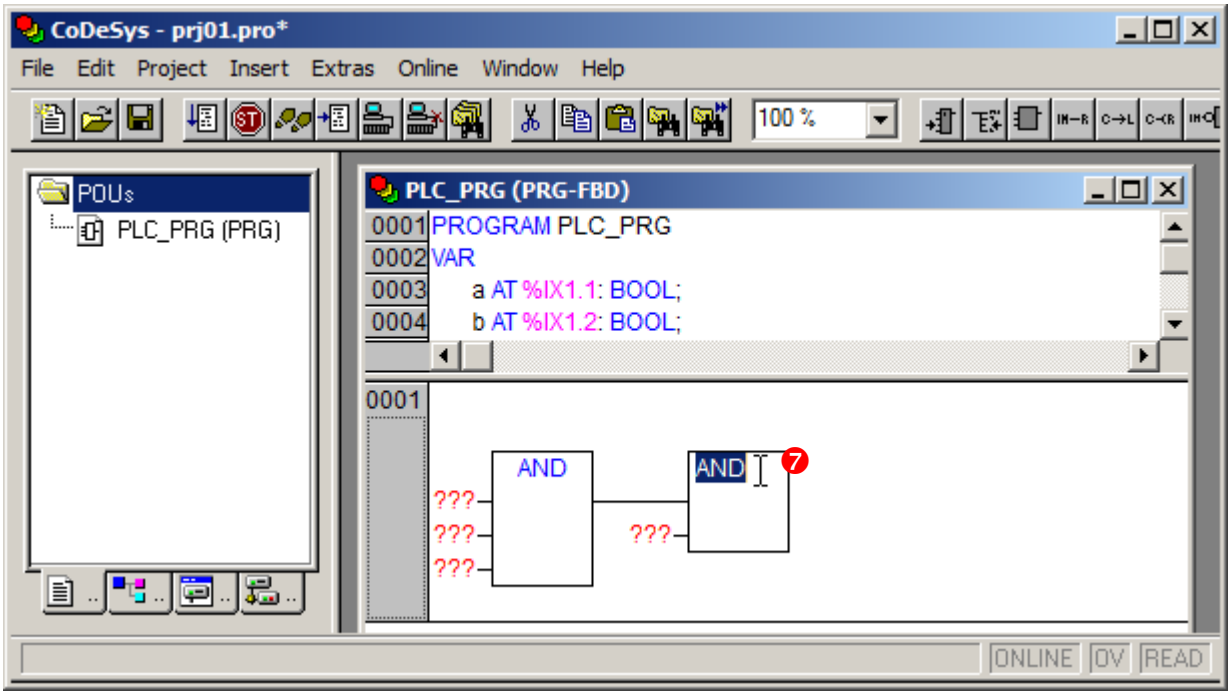


Po wstawieniu wejścia, program automatycznie zaznacza skojarzoną z nim zmienną pozwalając na jej zmianę. W omawianym sposobie postępowania kojarzenie zmiennych z wejściami zostanie wykonane na ostatnim etapie wprowadzania programu. W kolejnym kroku diagram zostanie uzupełniony o blok OR, który na swoim pierwszym wejściu odbiera sygnał z bloku AND, powinien więc zostać wstawiony za tym blokiem. W celu dodania bloku należy:

5) kliknąć na wyjściu bloku AND (na rysunku pokazany jest stan po kliknięciu na wyjściu bloku),
6) wstawić blok.

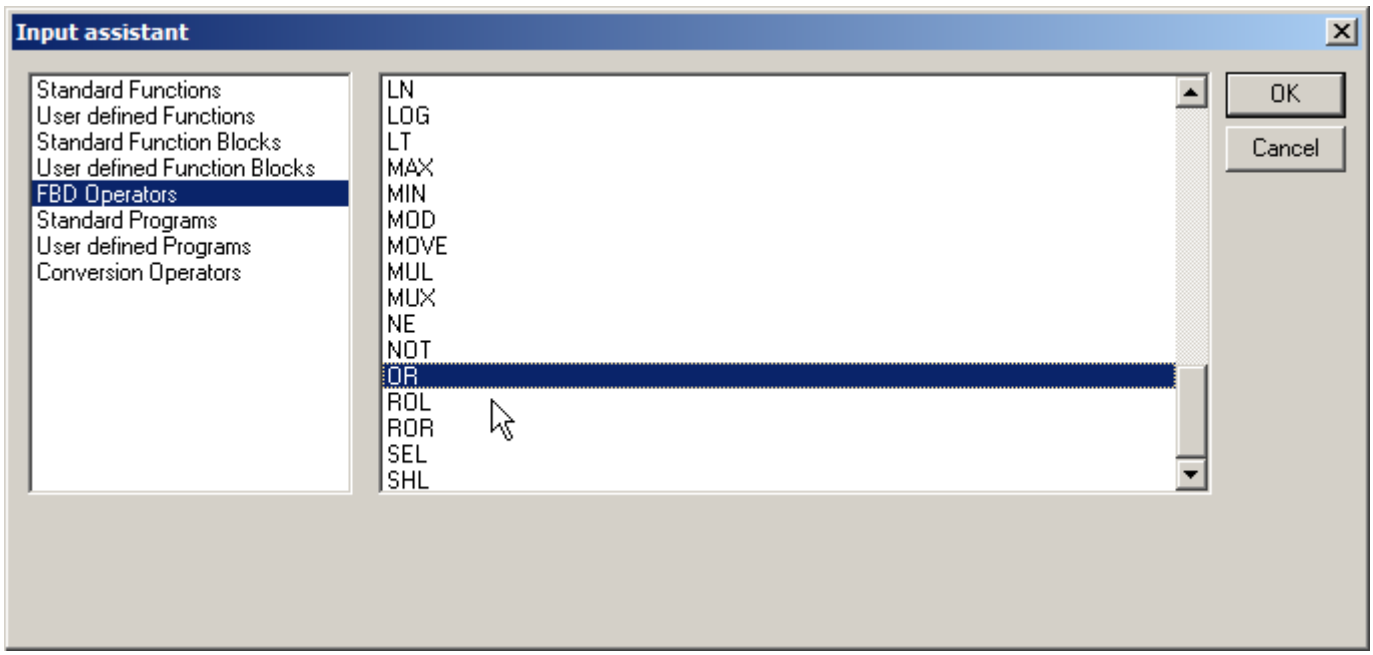


Po wstawieniu bloku

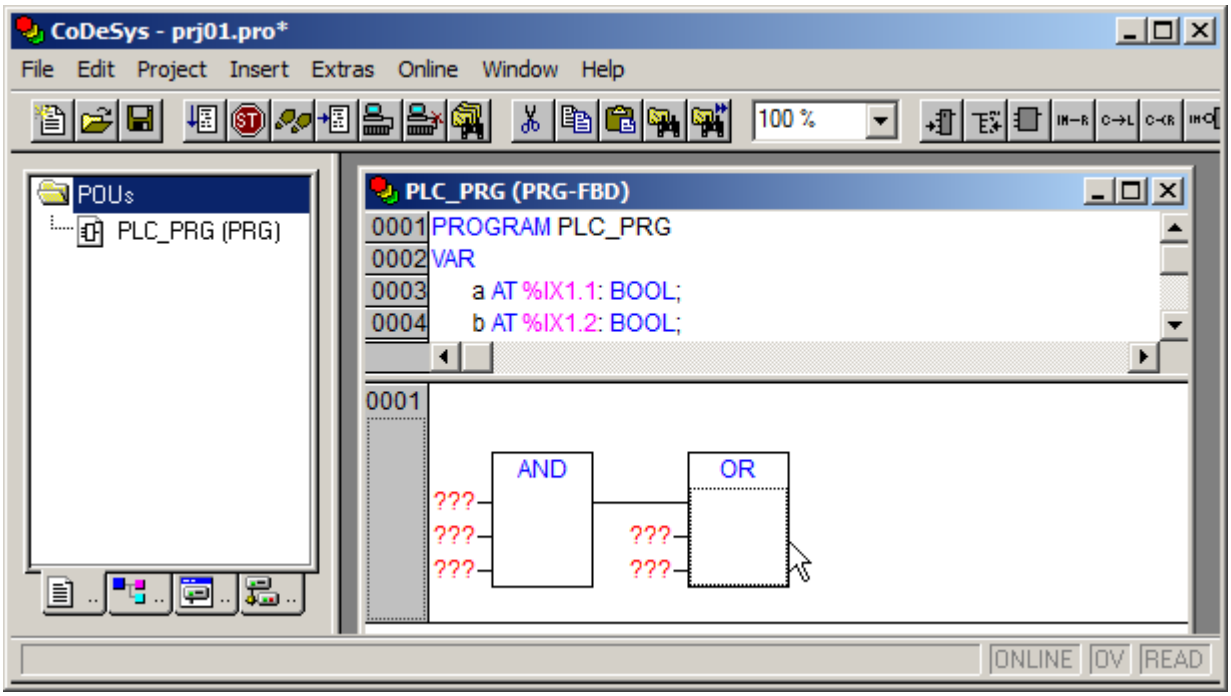


należy:

7) zmienić jego typ na OR, zastępując nazwę AND nazwą OR lub wykorzystując okno wyboru dostępne pod przyciskiem `F2`:



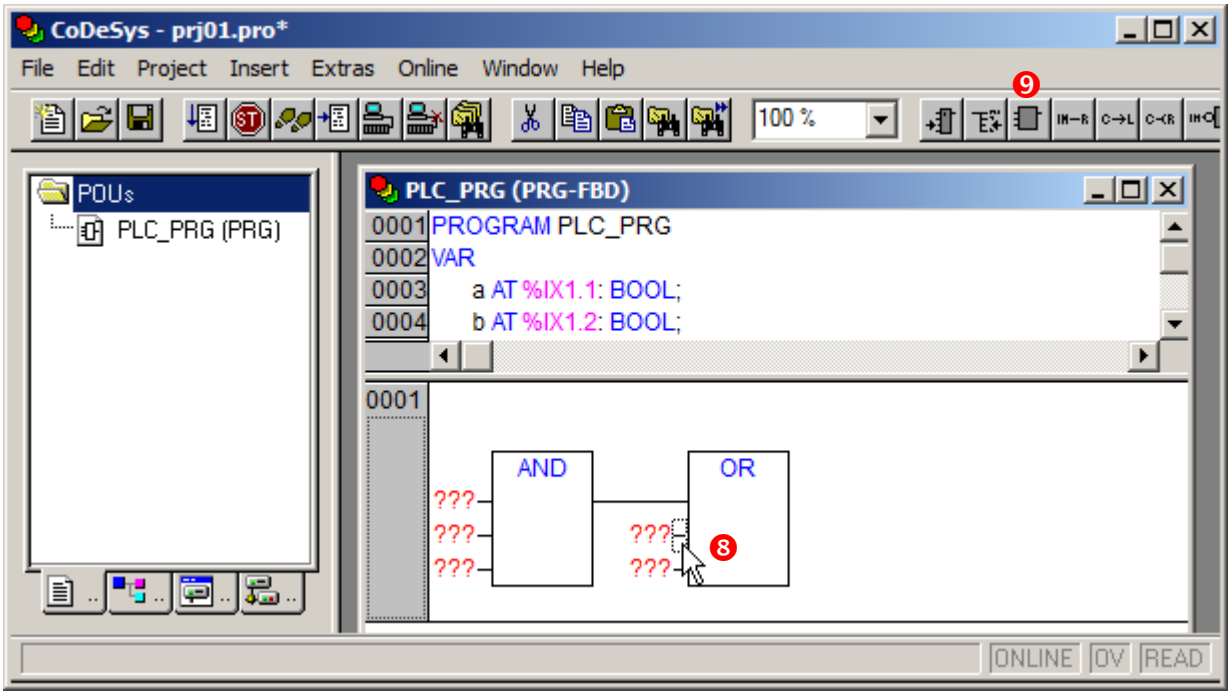
Po zmianie typu bloku należy jeszcze, posługując się techniką opisaną w punktach 3) i 4), dodać brakujące trzecie wejście. Na poniższym rysunku pokazany został stan programu modyfikacji bloku OR:



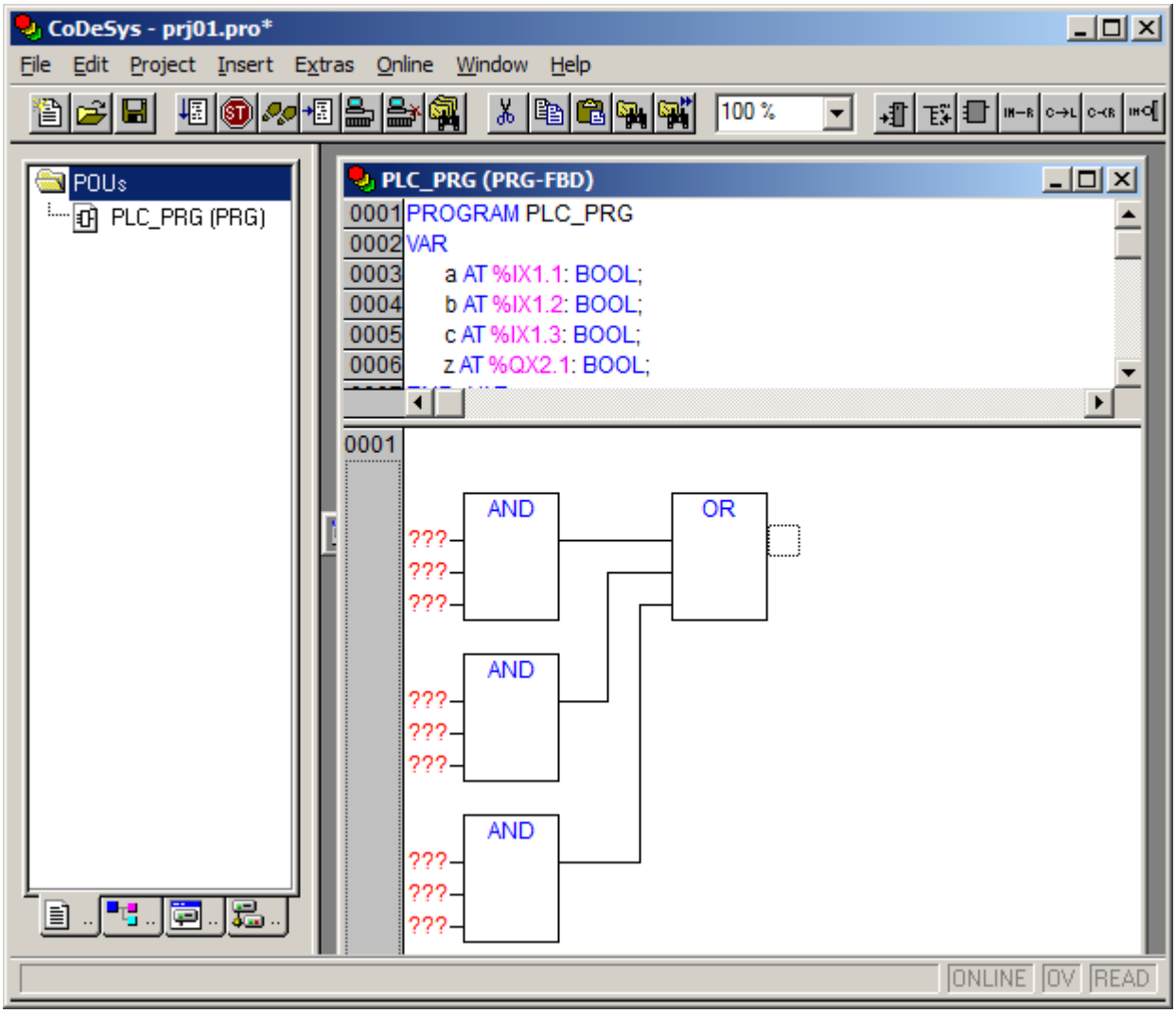
Na koniec należy wstawić dwa brakujące bloki AND. Bloki te, zgodnie ze schematem pokazanym w punkcie 1., generują sygnały wyjściowe, które podawane są następnie na wejście bloku OR.W celu dodania brakujących bloków należy:

8) kliknąć na wybranym wejściu bloku OR (na rysunku pokazany jest stan po kliknięciu na drugim wejściu bloku),

9) wstawić blok.



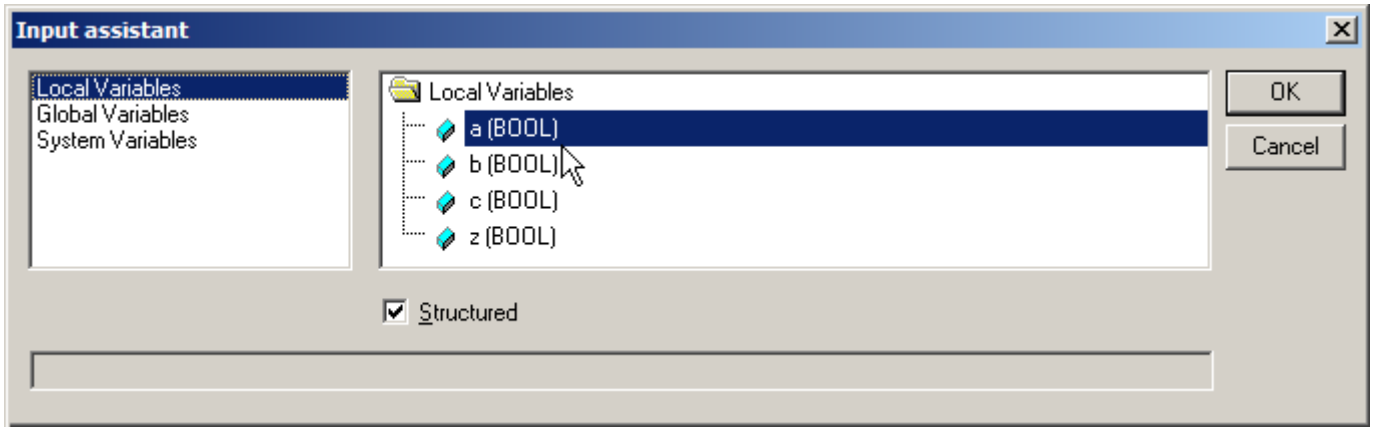
Posługując się techniką opisaną w punktach 3) i 4), należy jeszcze dodać brakujące trzecie wejście obydwu nowym blokom AND. Stan programu po uzupełnieniu diagramu został przedstawiony na poniższym rysunku:



Na koniec należy skojarzyć zadeklarowane wcześniej zmienne z wejściami poszczególnych bloków. W tym celu należy zastąpić ??? nazwami odpowiednich zmiennych. Po zaznaczeniu zastępowanej nazwy

nazwę właściwej zmiennej można:

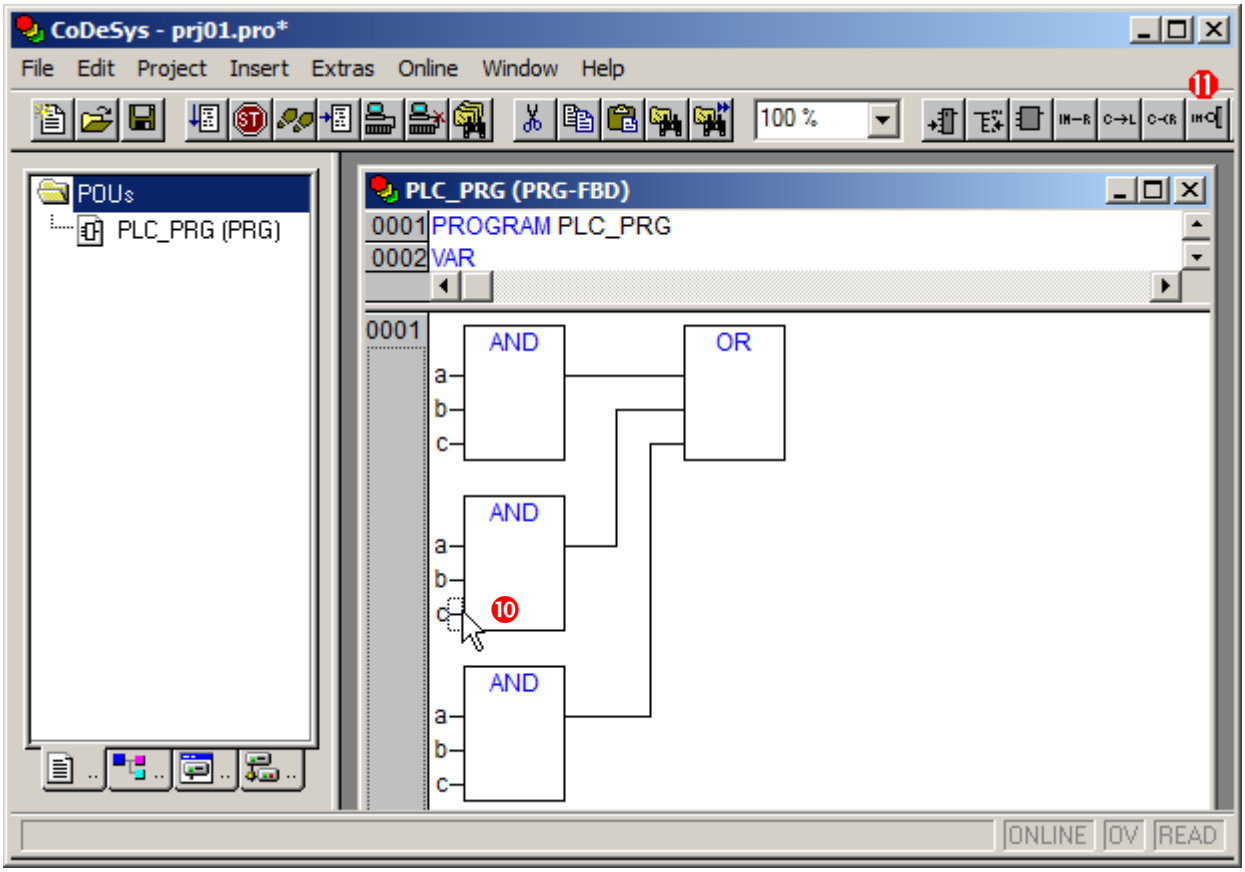
- wpisać z klawiatury,
- wybrać z okna wyboru dostępnego pod przyciskiem `F2`.



Po ustaleniu wartości wszystkich wejść diagramu, należy jeszcze zanegować trzecie wejście drugiego i drugie trzeciego bloku AND. W celu dodania brakujących negacji należy:

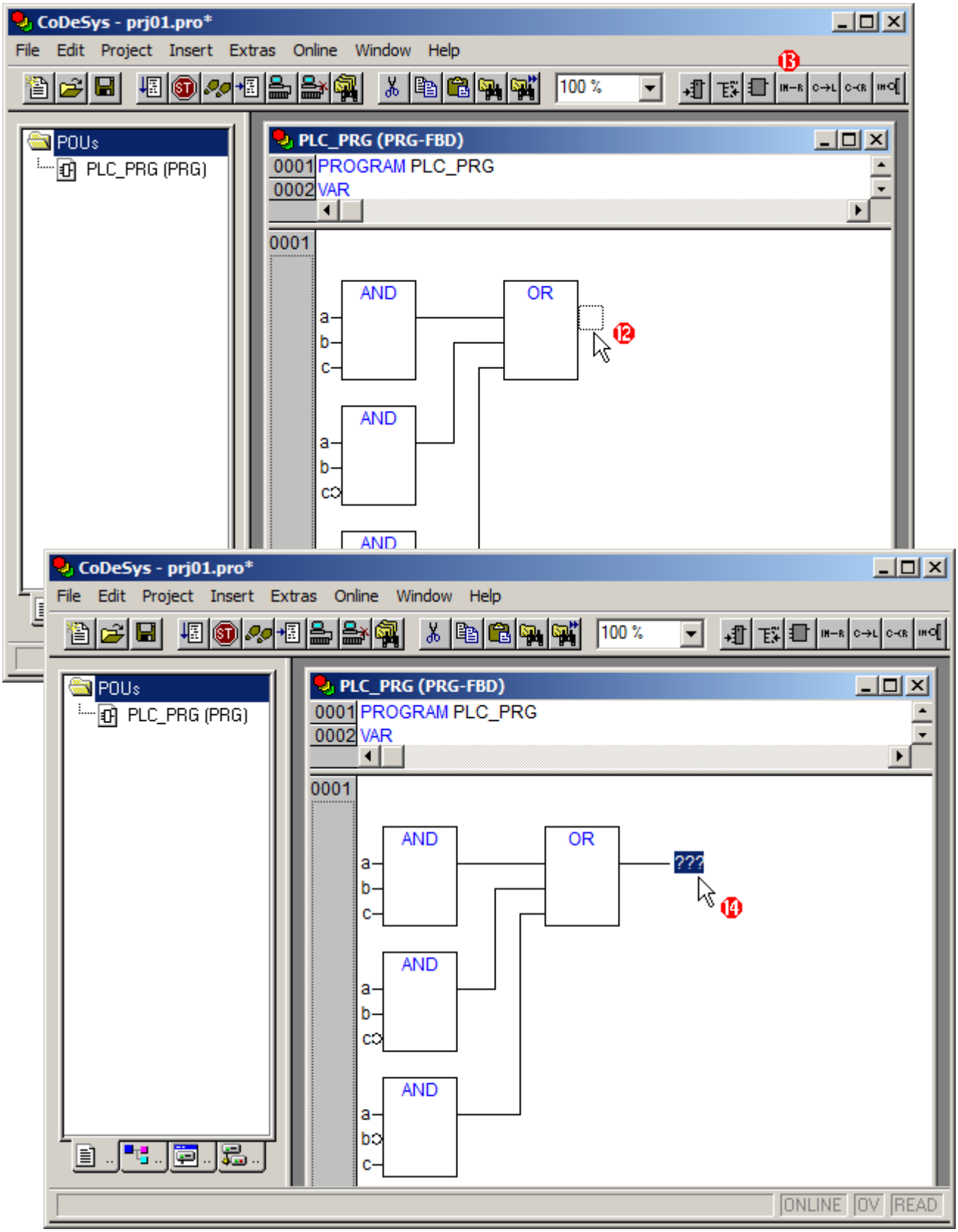
10) kliknąć na wybranym wejściu bloku (na rysunku pokazany jest stan po kliknięciu na trzecim wejściu drugiego bloku AND),

11) zanegować wejście.

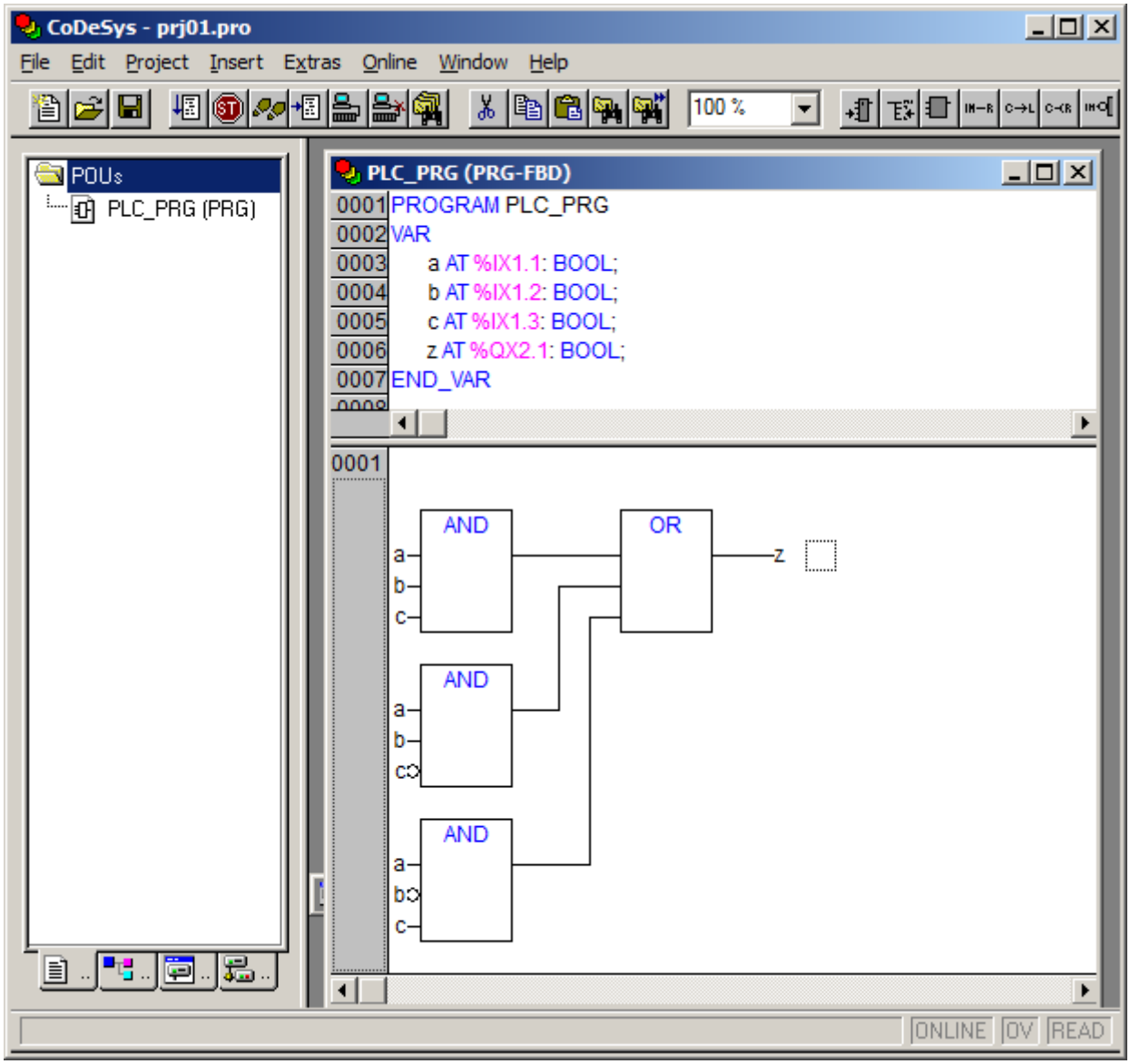


Po zanegowaniu odpowiednich wejść pozostaje jeszcze skojarzenie wyjścia bloku OR ze zmienną z, w tym celu należy:

12) kliknąć na wyjściu bloku OR (na rysunku pokazany jest stan po kliknięciu),
13) wstawić połączenie,
14) skojarzyć połączenie ze zmienną.



Gotowy program w języku FBD został przedstawiony na poniższym rysunku.



4. Symulacja działania programu

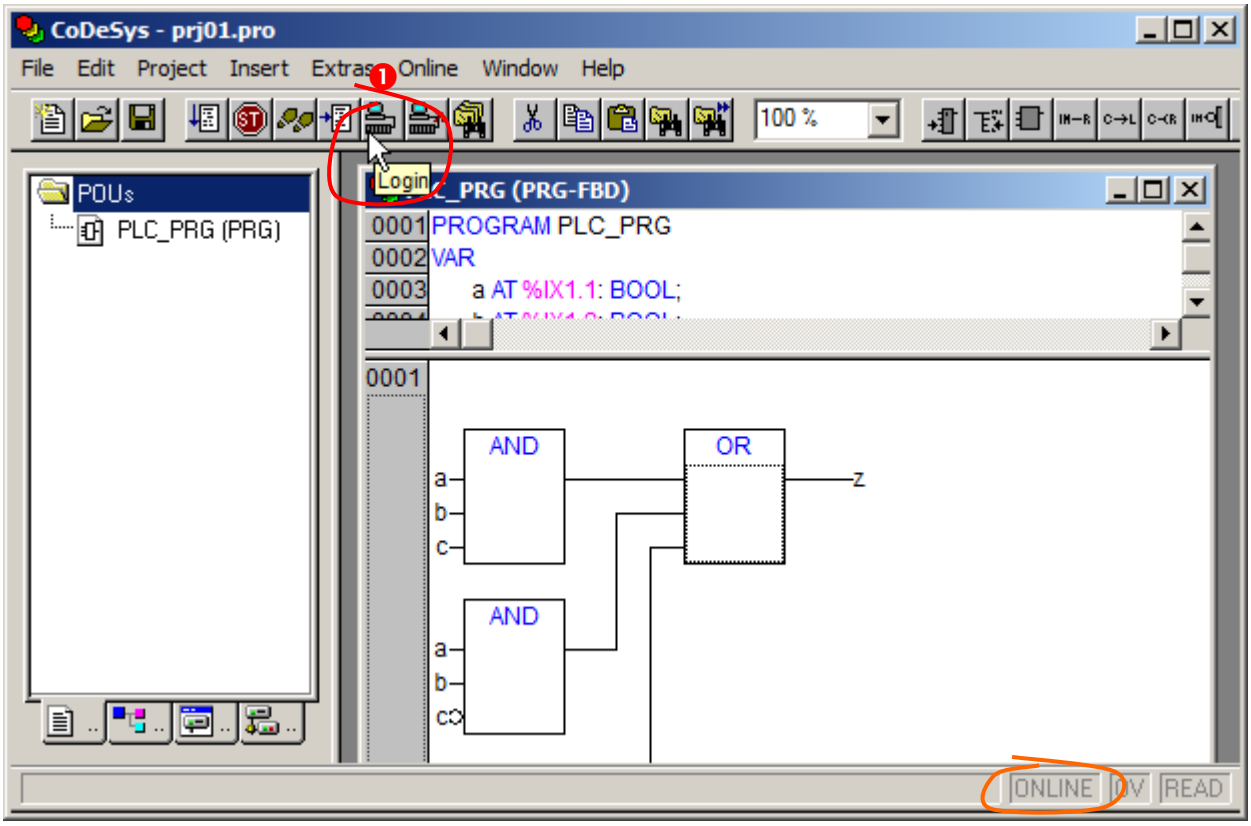
Działanie aplikacji można przetestować uruchamiając projekt na symulatorze sterownika (szczegółowe omówienie zagadnienia można znaleźć w materiale CoDeSys zamieszczonym na stronie przedmiotu).

W tym celu należy:

0) skompilować projekt,
1) nawiązać połączenie ze sterownikiem,
2) uruchomić program na sterowniku.

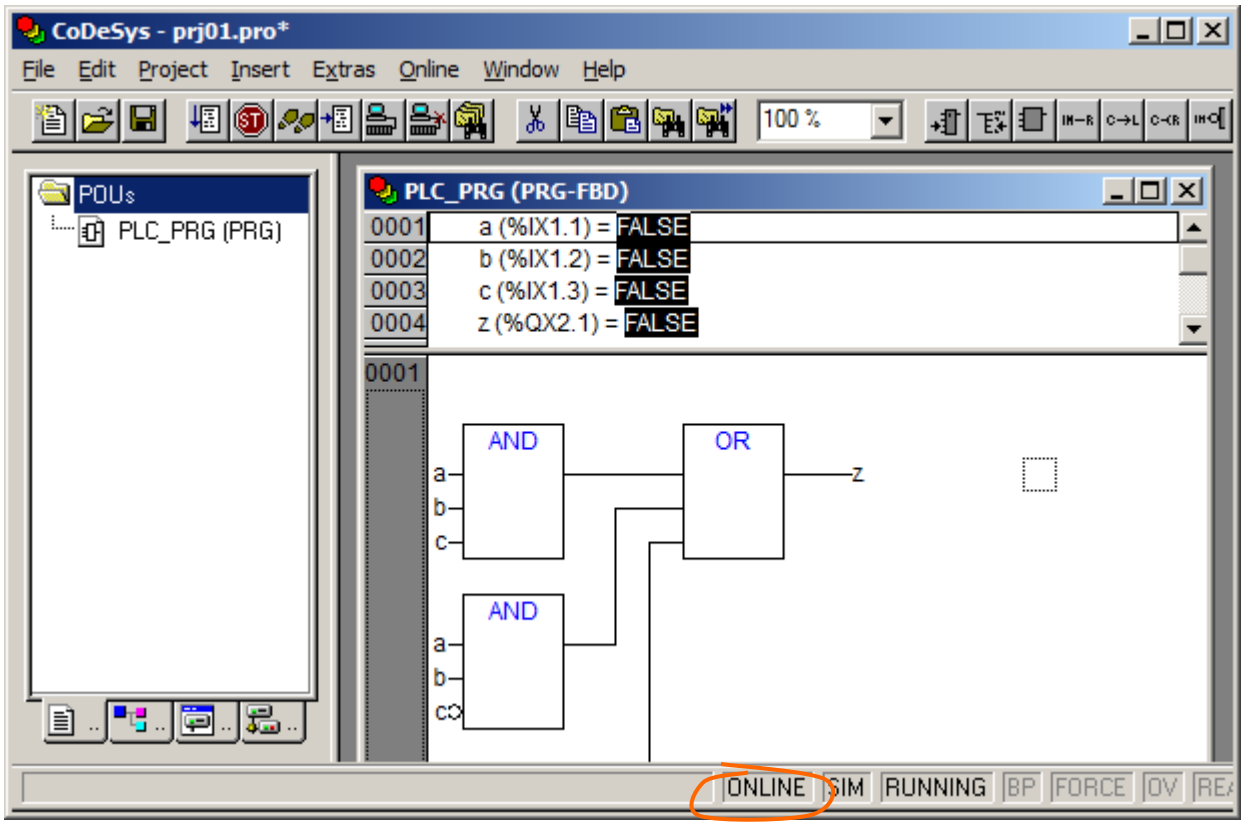
Ze względu na to, że środowisko rejestruje zmiany projektu i wykonuje jego automatyczną kompilację w przypadku próby nawiązywania połączenia ze sterownikiem ze zmodyfikowaną wersją projektu, krok 0. można pominąć wykonując od razu krok 1.

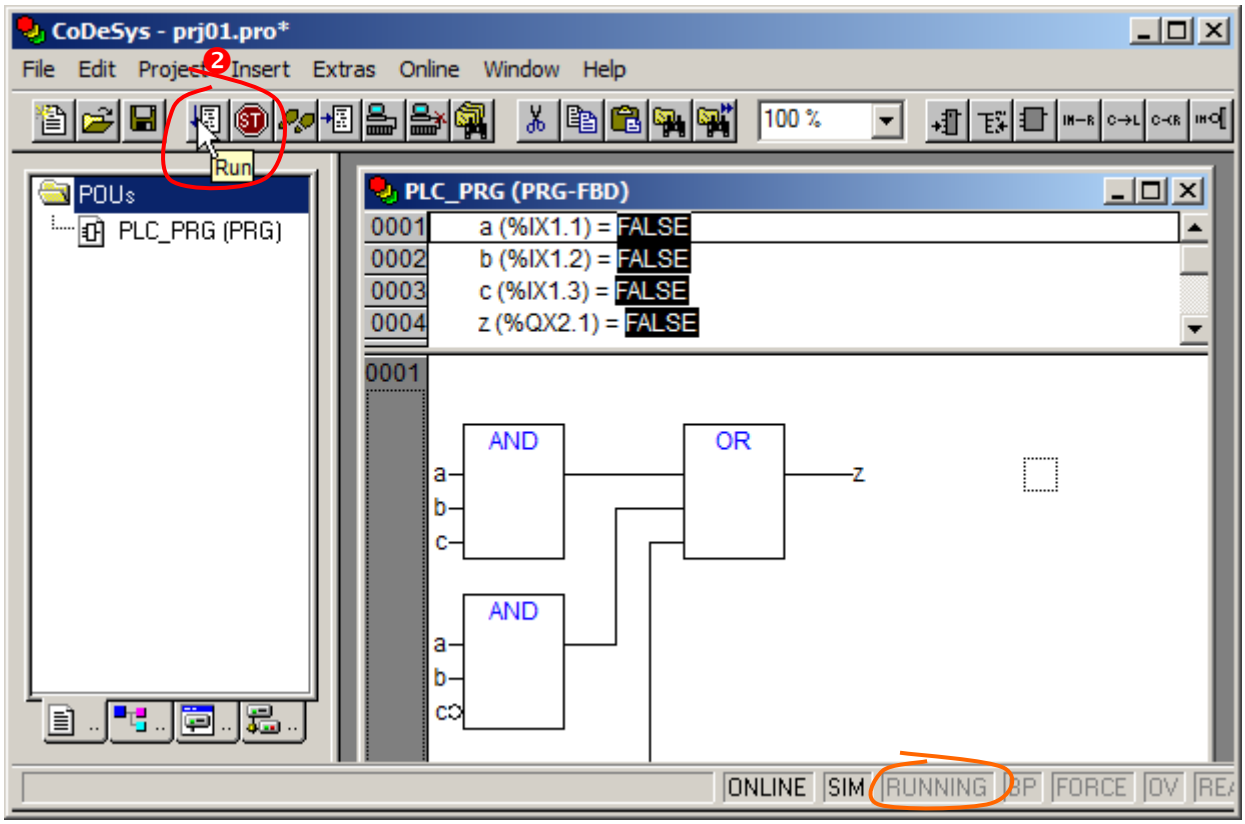
Kroki 1) i 2) zostały przedstawione na poniższych rysunkach.



Po wykonaniu kroku 1., w przypadku braku błędów kompilacji, CoDeSys przechodzi w tryb pracy ONLINE.

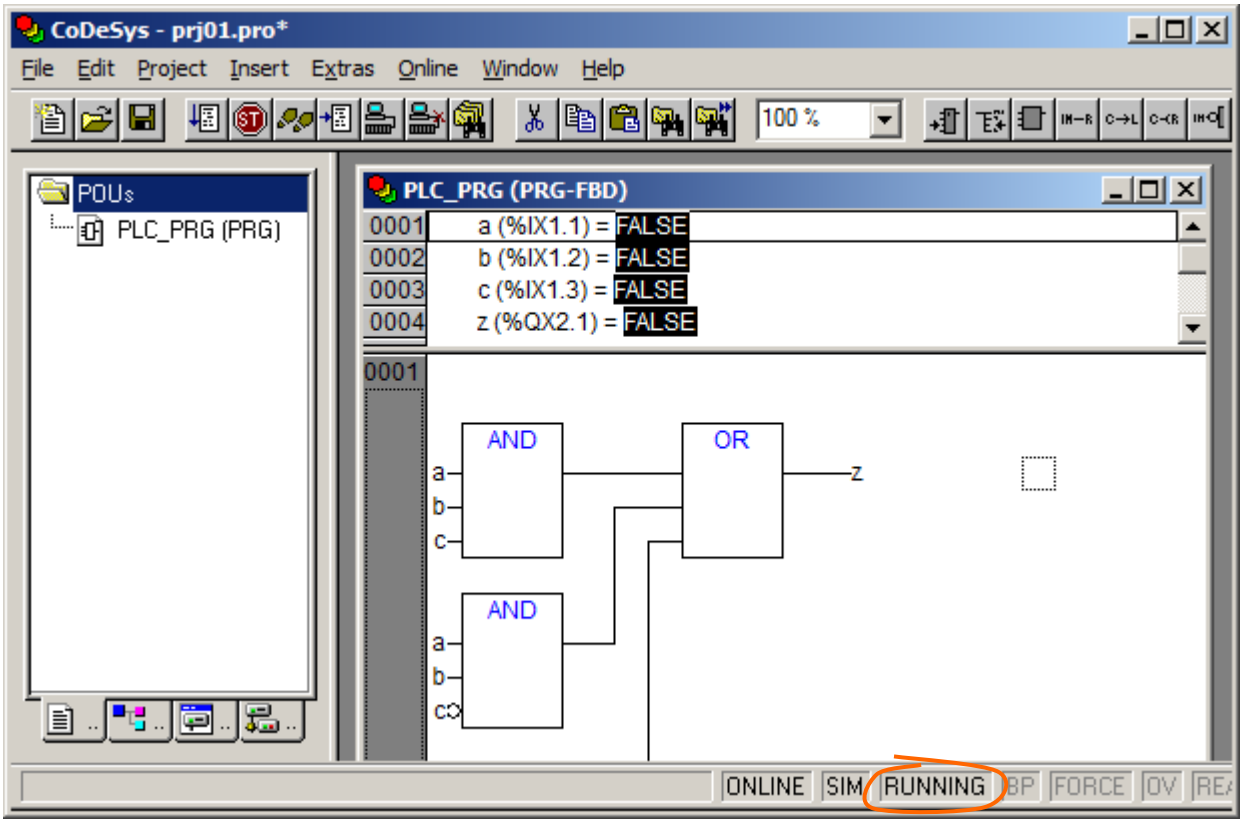
Linia statusu wyświetla aktualny stan pracy programu: na rysunku powyżej nie nawiązano jeszcze połączenia ze sterownikiem (napis ONLINE jest szary), na rysunku poniżej połączenie zostało nawiązane (napis ONLINE jest czarny).





Po wykonaniu kroku 2. CoDeSys uruchamia program na sterowniku

Po nawiązaniu połączenia, tzn. w trybie ONLINE, linia statusu wyświetla również tryb pracy sterownika: na rysunku powyżej sterownik pracuje w trybie STOP (napis RUNNING jest szary), na rysunku poniżej sterownik pracuje w trybie RUN (napis RUNNING jest czarny).



Testowanie programu sprowadza się do sprawdzania wartości zmiennych wyjściowych dla różnych kombinacji zmiennych wejściowych.

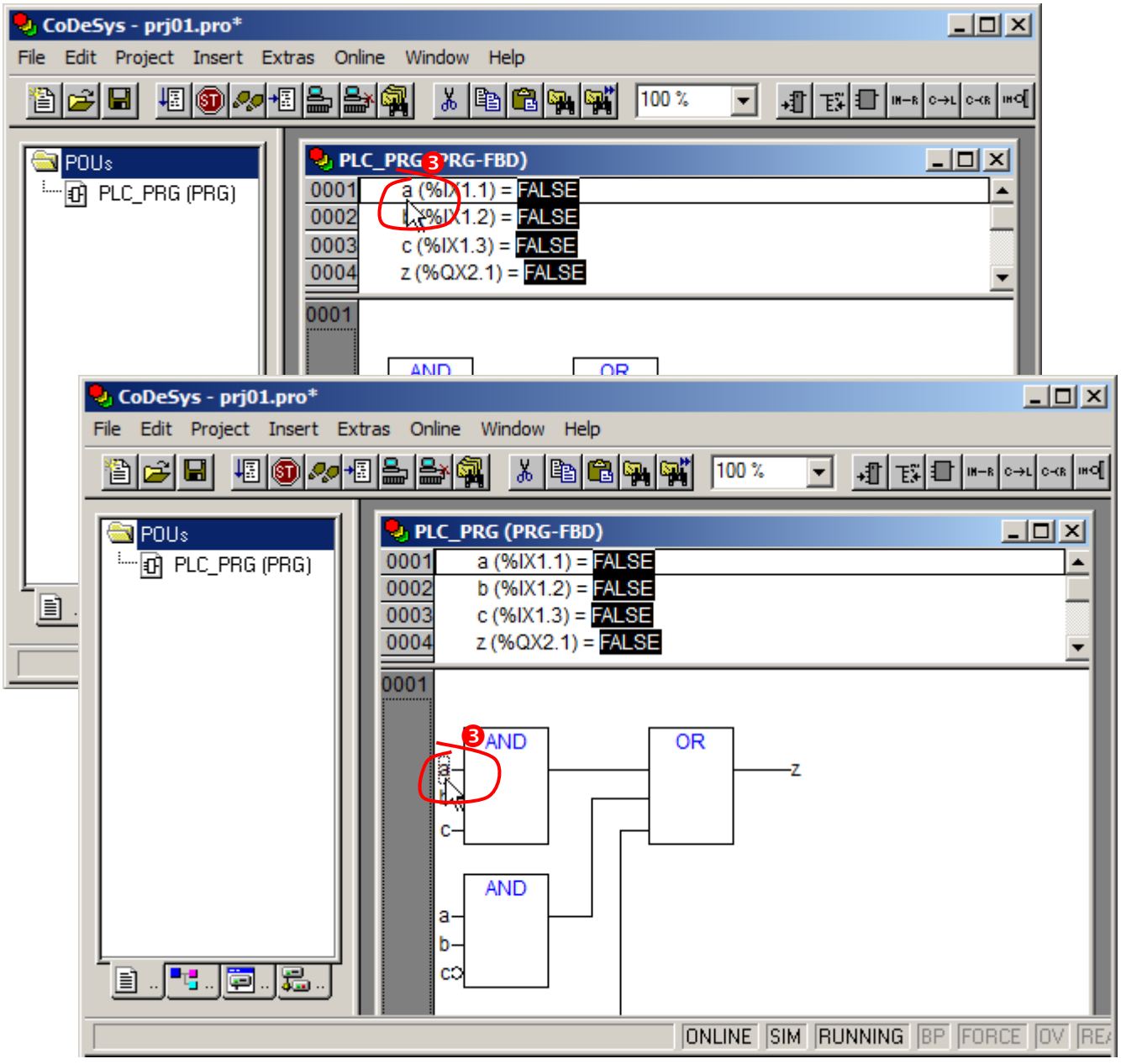
Na powyższym rysunku widać, że na starcie wszystkie zmienne wejściowe otrzymały wartość FAŁSZ co odpowiada nieprawidłowym wartościom odpowiednich cech badanego detalu. W takim przypadku zmienna *z* otrzymuje również wartość FAŁSZ co prowadzi do umieszczenia nieprawidłowego detalu w *pojemniku II*.

Reakcję programu na inną kombinację sygnałów wejściowych można sprawdzić:

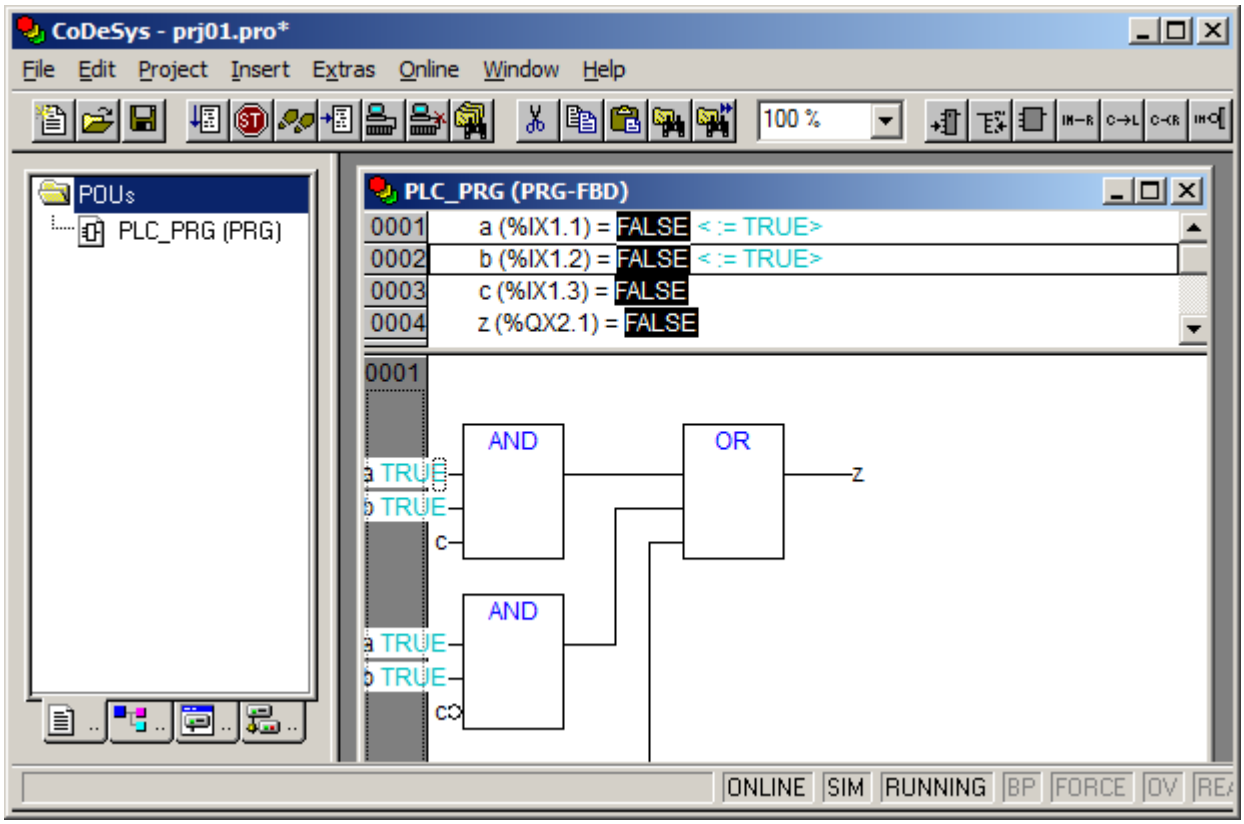
3) ustawiając wartości odpowiednich zmiennych wejściowych (wykonując podwójne kliknięcie na nazwie zmiennej)
4) przekazując nowe wartości zmiennych do sterownika (opcja Online->Write Values, Ctrl+F7).

Na poniższych rysunkach pokazany został stan programu

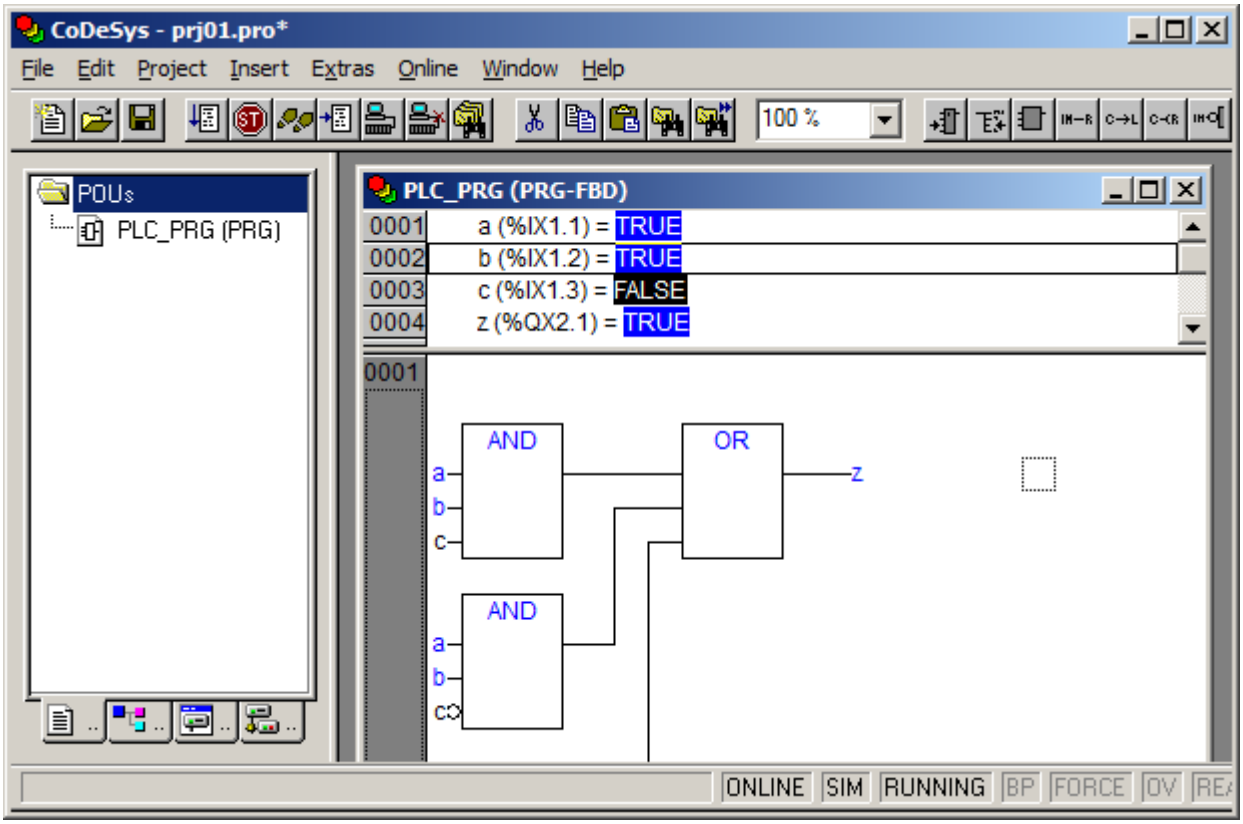
- przed zamianą wartości zmiennych:



- po ustawieniu cech *a* i *b* na prawidłowe (zmienne *a* i *b* ustawione na TRUE) :

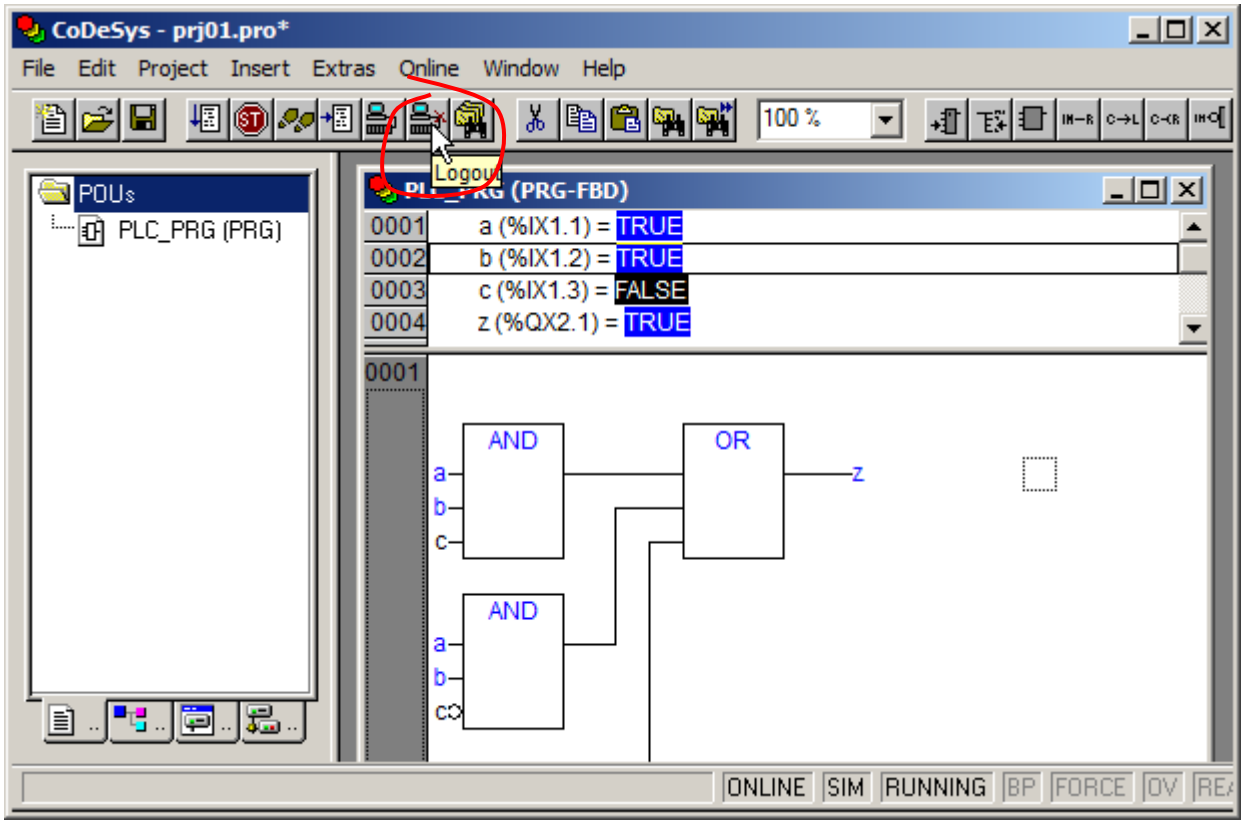


- po przekazaniu zmiennych do sterownika:



Detal o prawidłowych cechach *a* i *b* jest więc kierowany, zgodnie z opisem w punkcie 1., do *pojemnika I.*

Symulację można przerwać zrywając połączenie ze sterownikiem tak jak to zostało pokazane na poniższym rysunku.

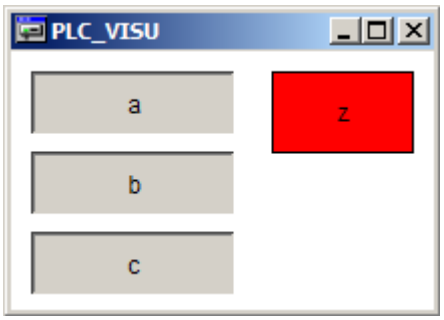


5. Wizualizacja

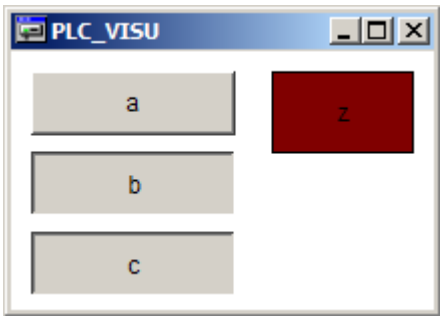
Testowanie działania aplikacji ułatwiają wizualizacje (szczegółowe omówienie zagadnienia można znaleźć w materiale CoDeSys zamieszczonym na stronie przedmiotu). Poniżej zostanie omówiona najprostsza z możliwych:

- pozwalająca przyciskami przełączającymi *a*, *b* i *c* zmieniać wartości sygnałów z czujników monitorujących odpowiednie cechy detalu,
- wyświetlająca stan sygnału sterującego zwrotnicą przy pomocy kolorowego prostokąta rozjaśnianego z chwilą otrzymania przez sygnał wartości „1” (TRUE) i przygaszanego gdy sygnał otrzyma wartość „0” (FALSE).

a)



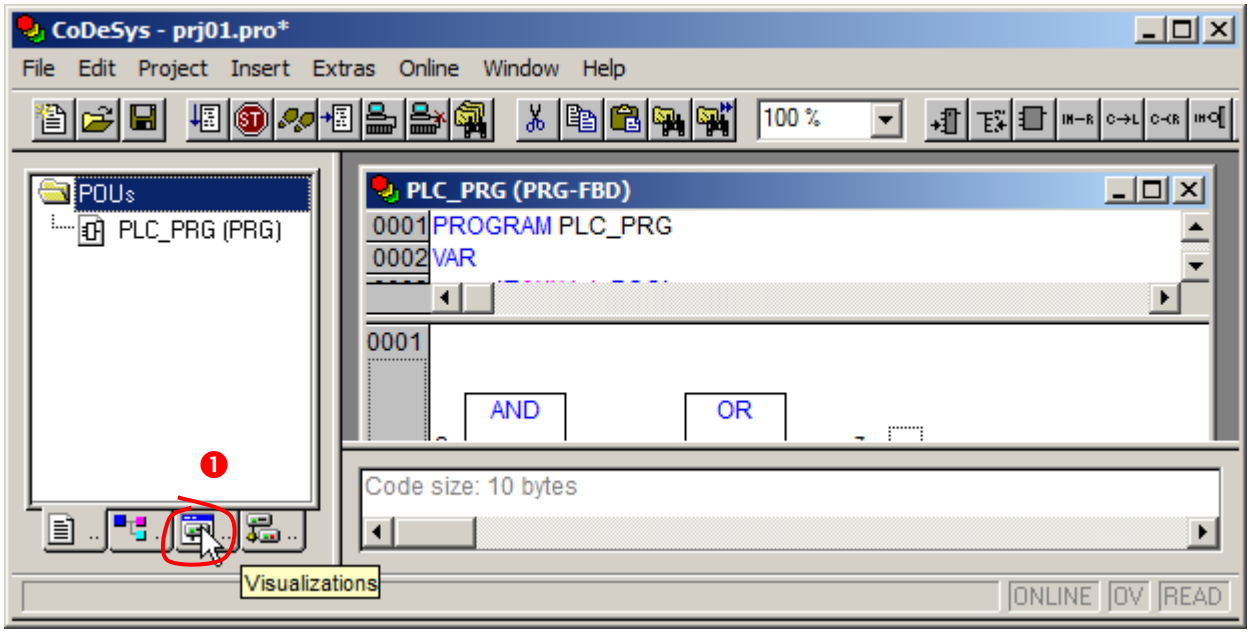
b)

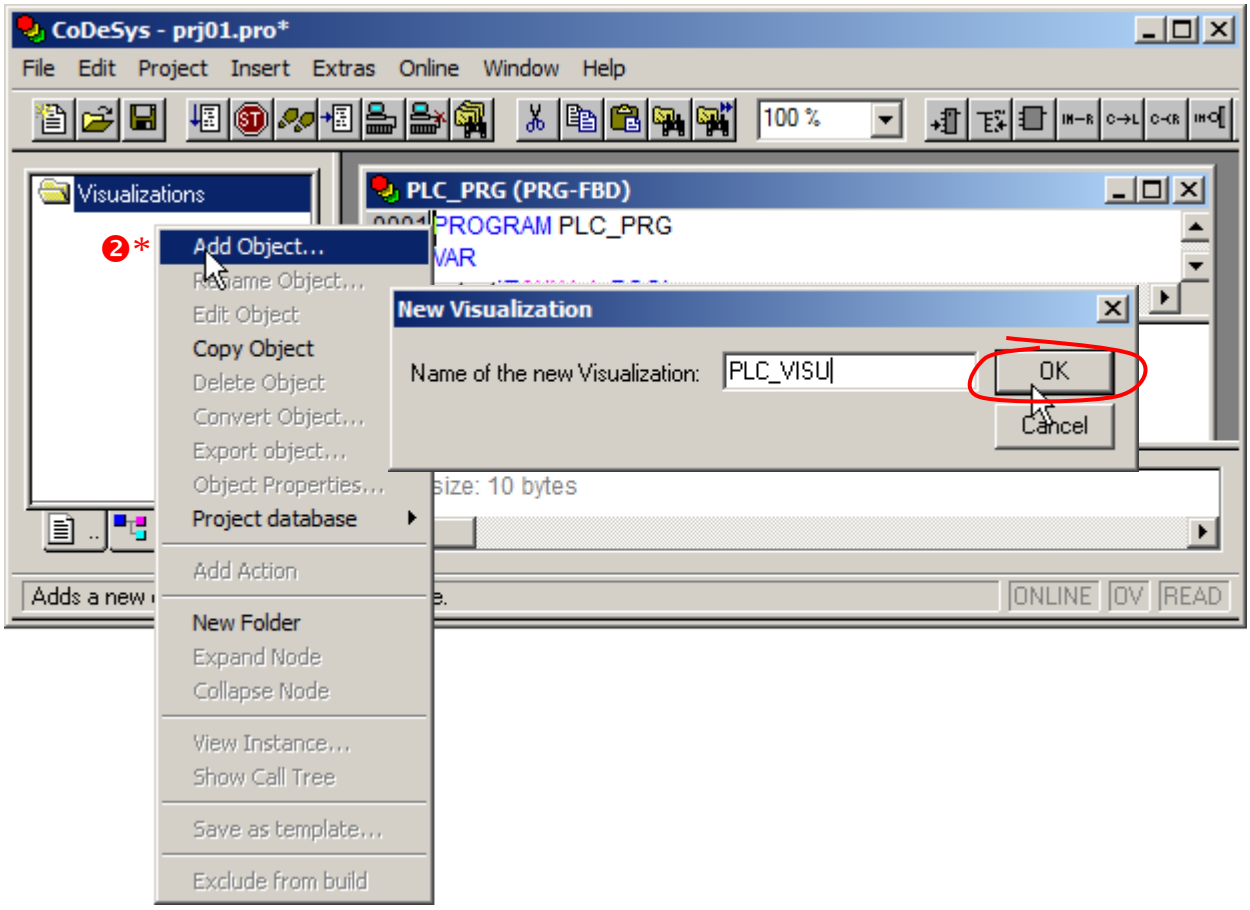


Wizualizacja a) wszystkie cechy detalu prawidłowe – zwrotnica kieruje detal do pojemnika I b) prawidłowe tylko cechy b i c – zwrotnica kieruje detal do pojemnika II.

Do stworzenia przykładowej wizualizacji niezbędne jest rozbudowanie projektu. W tym celu w oknie zasobów należy:

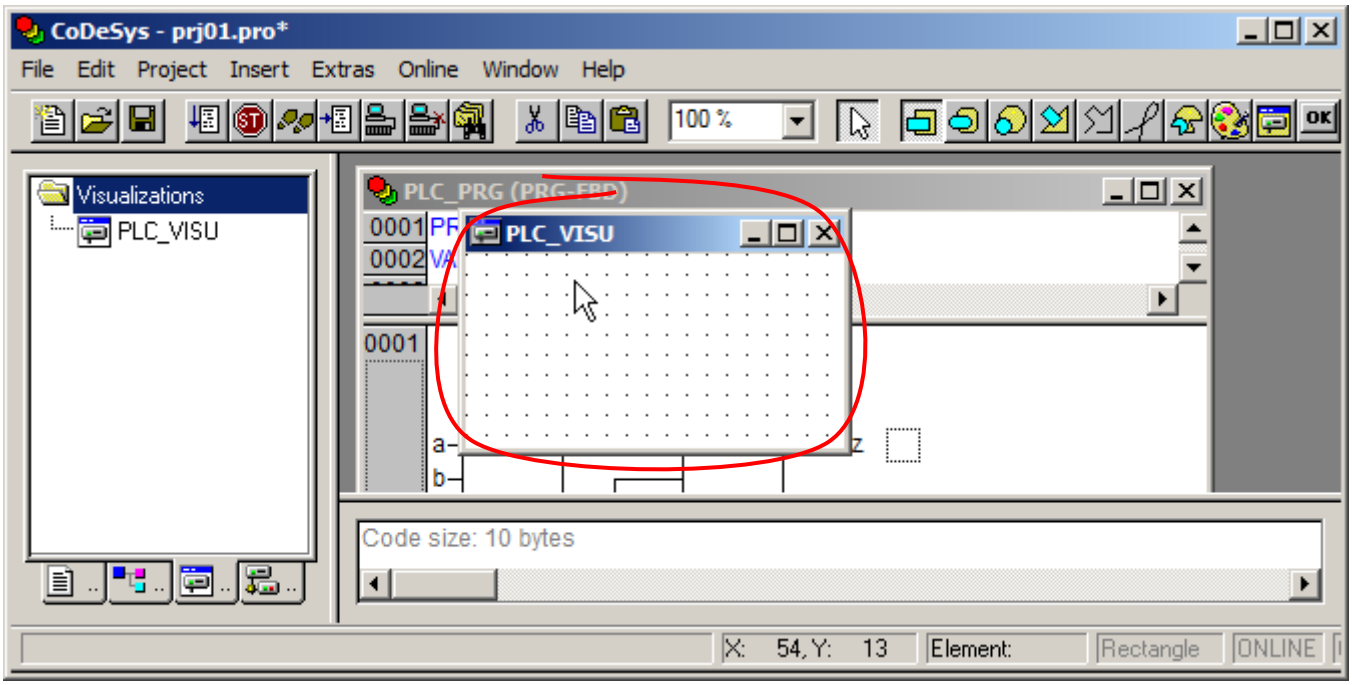
1) przełączyć się na zakładkę z wizualizacjami,
2) dodać nową wizualizację nadając jej nazwę PLC_VISU (domyślna nazwa dla wizualizacji startowej).





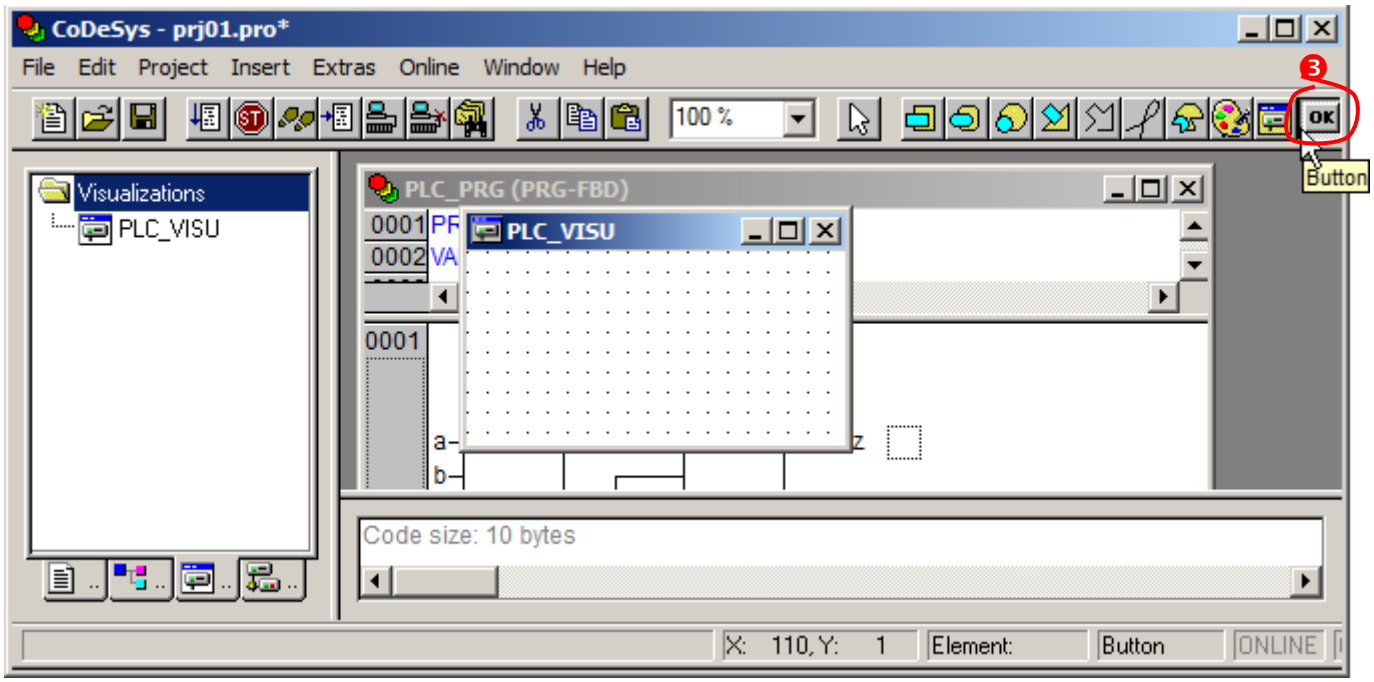
❷* - kliknięcie wykonać prawym przyciskiem myszy.

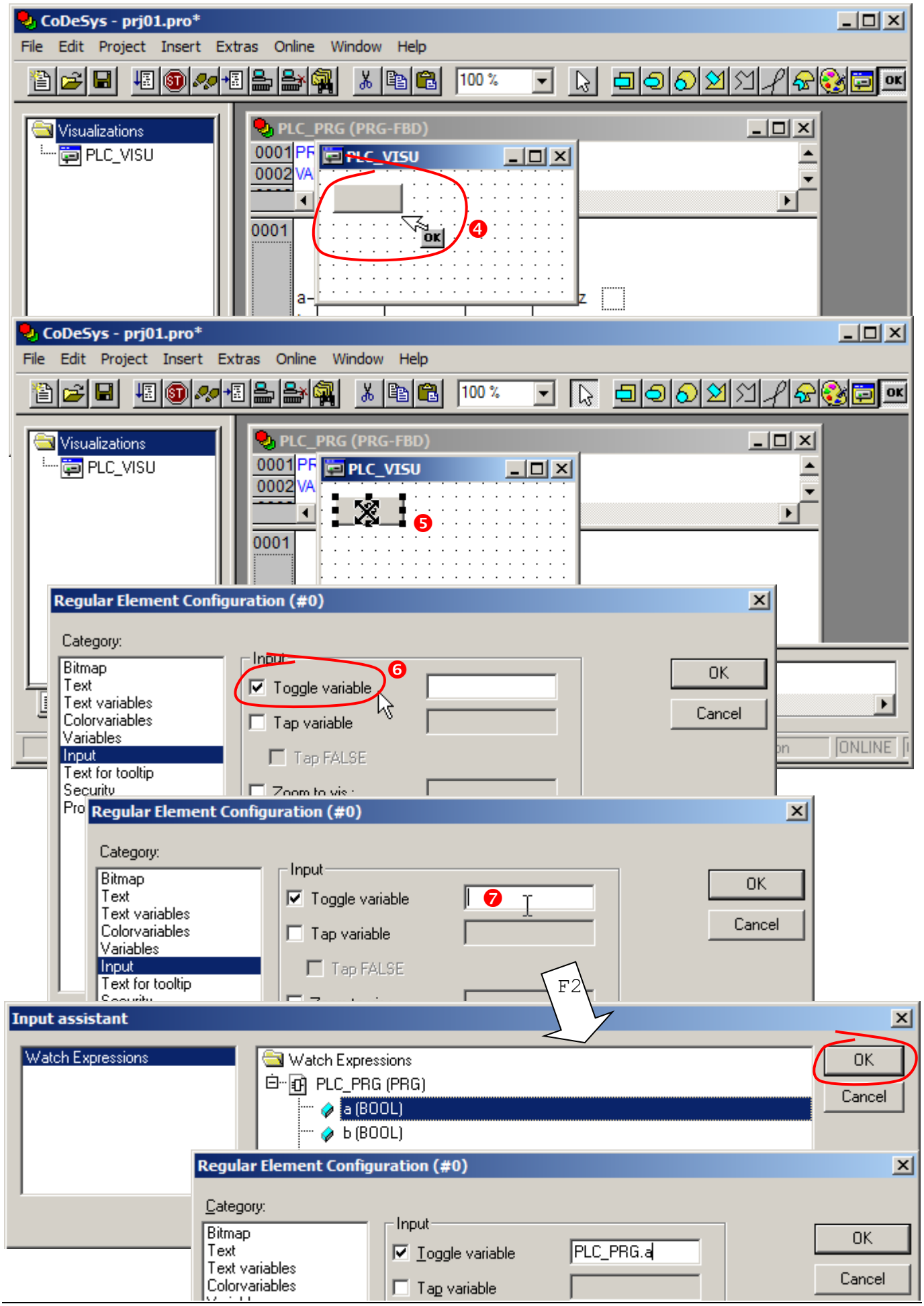
Po wykonaniu kroków 1) i 2) otwierane jest puste okienko projektu wizualizacji.

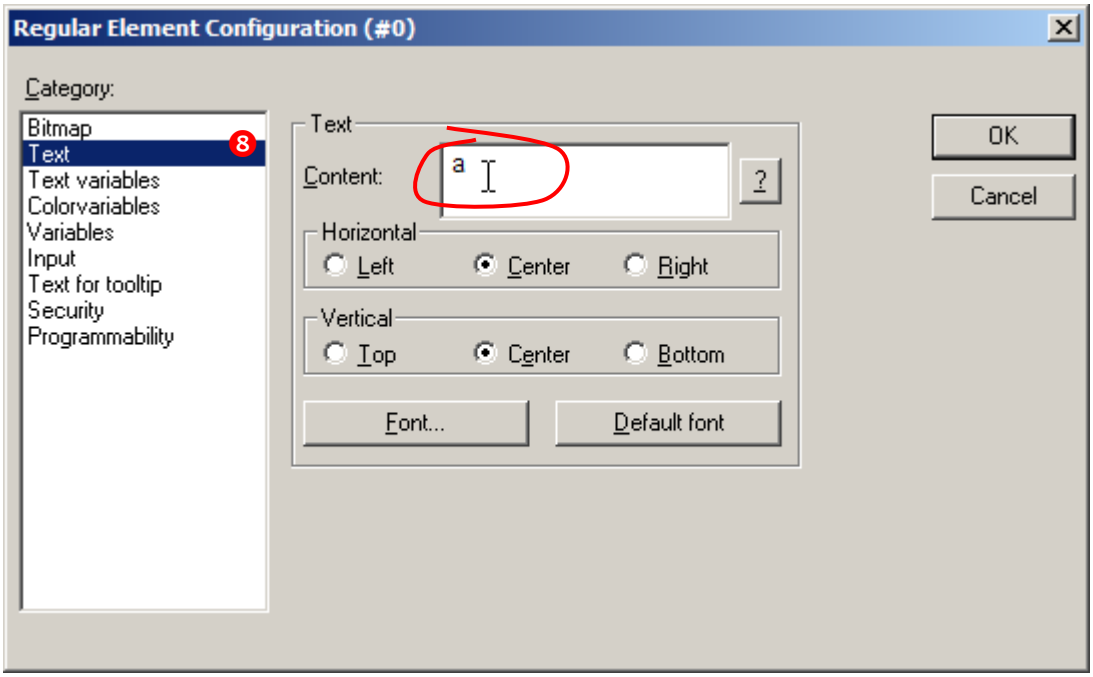


W przykładowej wizualizacji wartości sygnałów wejściowych zmieniane są przyciskami przełączającymi, w celu dodania przycisku należy:

3) wskazać na pasku narzędziowym przycisk OK,
4) zaznaczyć w projekcie obszar przeznaczony na dodawany obiekt,
5) dwukrotnie kliknąć na przycisku,
6) w oknie konfiguracyjnym obiektu ustalić jego wejście na zmienną przełączającą – opcja Toggle **variable**,
7) w polu obok opcji wprowadzić nazwę zmiennej, najbezpieczniej wybierając ją z okna wywołanego przy pomocy przycisku `F2`,
8) bez wychodzenia z okna konfiguracyjnego ustawić odpowiednią etykietę przycisku.




CoDeSys - prj01.pro*
File Edit Project Insert Extras Online Window Help
Visualizations
PLC_VISU
PLC_PRG (PRG-FBD)
PLC_VISU
4
5
Regular Element Configuration (#0)
Category:
Bitmap
Text
Text variables
Colorvariables
Variables
Input
Text for tooltip
Security
Input
Toggle variable
Tap variable
Tap FALSE
6
7
OK
Cancel
F2
Input assistant
Watch Expressions
PLC_PRG (PRG)
a (BOOL)
b (BOOL)
PLC_PRG.a



Po trzykrotnym powtórzeniu kroków 3) – 8) w wizualizacji brakuje jedynie elementu wyświetlającego stan sygnału sterującego zwrotnicą. przełączającymi – element ten dodawany jest w podobnie jak przyciski reprezentujące sygnały wejściowe, tzn. należy:

9) wskazać na pasku narzędziowym przycisk z symbolem prostokąta,

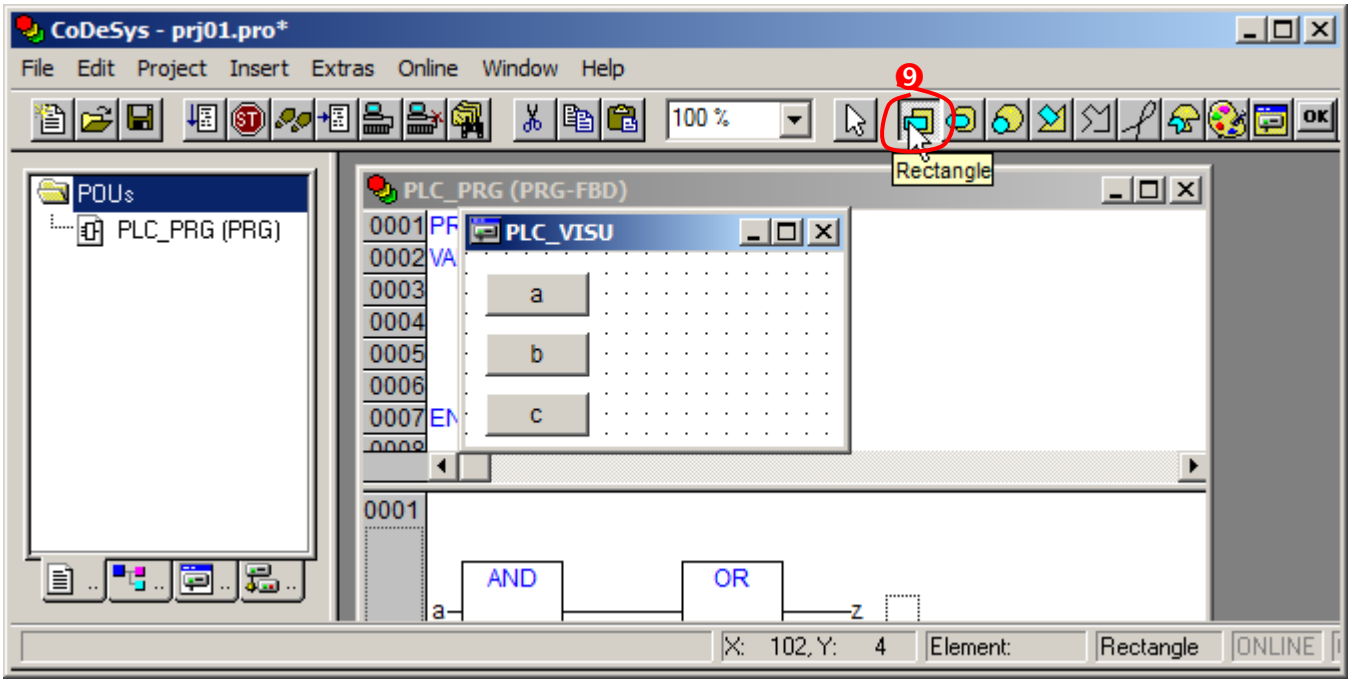
10) zaznaczyć w projekcie obszar przeznaczony na dodawany obiekt,

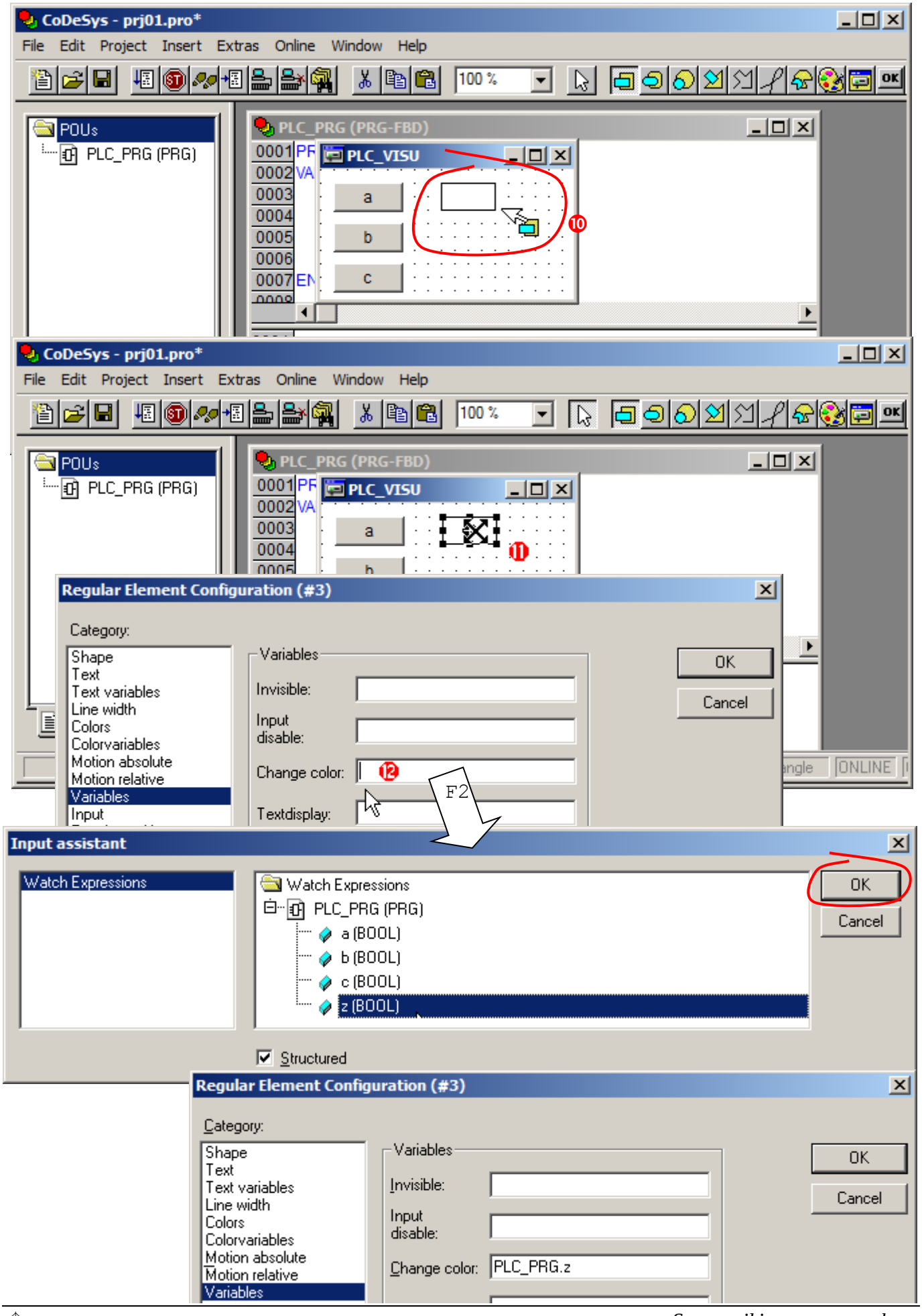
11) dwukrotnie kliknąć na elemencie,

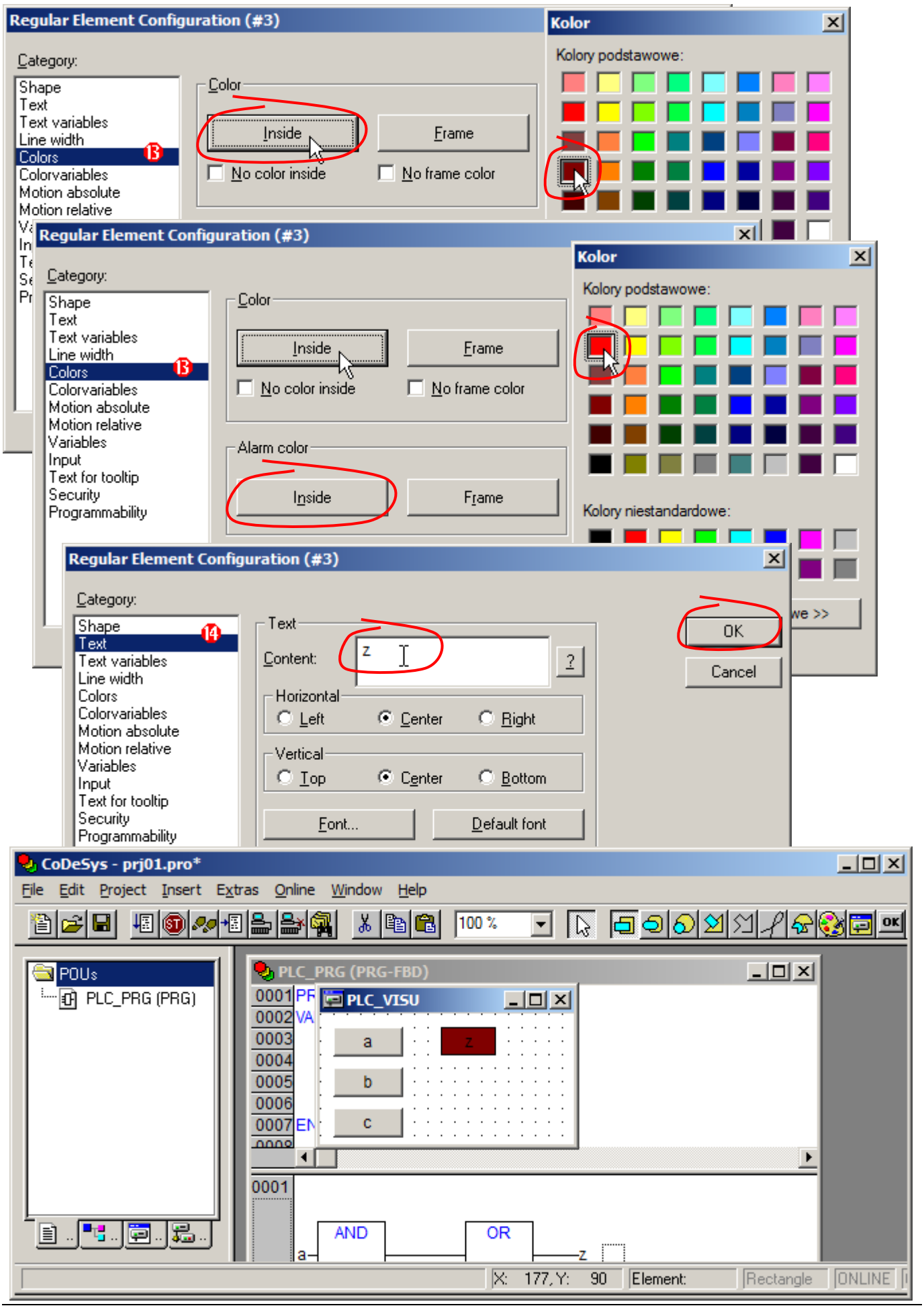
12) w oknie konfiguracyjnym obiektu ustalić zmienną odpowiedzialną za zmianę koloru prostokąta – opcja **Change color** (wprowadzanie zmiennej z okna wywołanego przy pomocy `F2`),

13) bez wychodzenia z okna konfiguracyjnego ustawić kolory przygaszony i rozjaśniony (przygaszony jako kolor zwykły tzn. **Color** i rozjaśniony jako **Alarm color**),

14) oraz etykietę elementu.




CoDeSys - prj01.pro*
File Edit Project Insert Extras Online Window Help
100 %
POUs
PLC_PRG (PRG)
PLC_PRG (PRG-FBD)
PLC_VISU
a
b
c
10
11
Regular Element Configuration (#3)
Category:
Shape
Text
Text variables
Line width
Colors
Colorvariables
Motion absolute
Motion relative
Variables
Input
Variables
Invisible:
Input disable:
Change color:
Textdisplay:
12
OK
Cancel
F2
Input assistant
Watch Expressions
PLC_PRG (PRG)
a (BOOL)
b (BOOL)
c (BOOL)
z (BOOL)
Structured
Regular Element Configuration (#3)
Category:
Shape
Text
Text variables
Line width
Colors
Colorvariables
Motion absolute
Motion relative
Variables
Variables
Invisible:
Input disable:
Change color:
PLC_PRG.z
OK
Cancel


Regular Element Configuration (#3)
Category:
Shape
Text
Text variables
Line width
Colors
Colorvariables
Motion absolute
Motion relative
Variables
Input
Text for tooltip
Security
Programmability
Color
Inside
Frame
No color inside
No frame color
Alarm color
Kolor
Kolory podstawowe:
Kolory niestandardowe:
Content:
Horizontal
Left
Center
Right
Vertical
Top
Bottom
Font...
Default font
OK
Cancel
CoDeSys - prj01.pro*
File Edit Project Insert Extras Online Window Help
POUs
PLC_PRG (PRG)
PLC_PRG (PRG-FBD)
PLC_VISU
a
b
c
z
AND
OR
X: 177, Y: 90
Element:
Rectangle
ONLINE

Tak przygotowana wizualizacja jest już gotowa do użycia. Teraz po połączeniu ze sterownikiem i wystartowaniu programu można jej używać do przetestowania programu.