1. MOŻLIWOŚCI

ISaGRAF to zintegrowany pakiet oprogramowania firmy *CJ International*, który umożliwia programowanie sterowników *PEP9xxx*, *IUC9xxx* oraz *SMART*. Zasady programowania sterowników są zgodne wytycznymi normy IEC 61131-3. ISaGRAF pozwala na tworzenie aplikacji we wszystkich językach definiowanych w normie, tzn. w językach:

- *funkcjonalnych schematów blokowych FBD,*
- *schematów drabinkowych LD,*
- *tekstu strukturalnego ST,*
- *listy instrukcji IL,*
- *sekwencyjnego schematu funkcjonalnego SFC.*

Dodatkowo możliwe jest również programowanie własnych bloków funkcjonalnych w języku C.

Pakiet ISaGRAF zawiera narzędzie ułatwiające testowanie napisanego programu – tzw. debager (*ang. debugger*). Debager pozwala na śledzenie przebiegu wykonywania programu ułatwiając lokalizację błędów. Kontrolowany debagerem program może być wykonywany na rzeczywistym sterowniku lub na symulatorze sterownika, gdy sterownik nie jest dostępny.

2. INSTALACJA

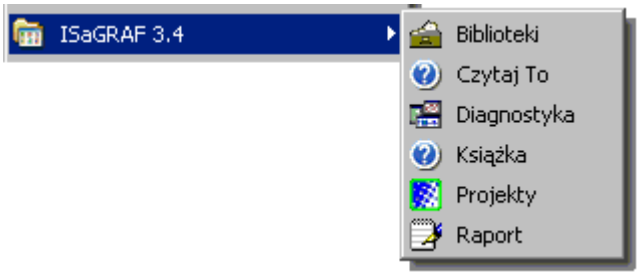
W sekcji **Linki** na stronie przedmiotu umieszczone zostały wybrane strony, z których można pobrać wersję instalacyjną programu (demo).

Instalacja programu przebiega w sposób standardowy. Po rozpakowaniu pliku zawierającego wersje instalacyjną należy uruchomić program INSTALL. Program ten kopiuje odpowiednie pliki we wskazane miejsce – domyślnym folderem ISaGRAF jest „\ISAWIN”. W trakcie instalacji użytkownik może wskazać elementy pakietu do zainstalowania, tzn.:

- właściwy program,
- przykładowe aplikacje,
- biblioteki standardowe,
- pliki pomocy.

3. ELEMENTY PAKIETU

Po zainstalowaniu wszystkich elementów pakietu w wersji 3.46 systemowe *Menu Start* zawiera widoczne na poniższym rysunku elementy.



Rys. 3.1. Elementy składowe pakietu ISaGRAF.

3.1. Czytaj To, Książka

Opcje uruchamiają odpowiednie pliki pomocy. Opcja *Czytaj To* wyświetla informacje o różnicach pomiędzy aktualną a poprzednimi wersjami ISaGRAF. Opcja *Książka* wyświetla informacje o wszystkich elementach pakietu.

3.2. Biblioteki

Biblioteki to zbiory zasobów, które mogą być wykorzystywane przez aplikacje tworzone w ISaGRAF. Po wybraniu opcji *Biblioteki* z *Menu Start* uruchamiany jest tzw. *Menadżer Bibliotek*. Program ten pozwala na definiowanie kart, zespołów oraz konfiguracji wejścia/wyjścia, umożliwia zarządzanie zbiorami funkcji i bloków funkcjonalnych.

Biblioteka kart We/Wy zawiera definicje dostępnych modułów wejściowych i wyjściowych. Podstawowymi parametrami każdej karty są:

- *kierunek* – karty wejściowa i wyjściowa,
- *typ* – karta z *kanałami* cyfrowymi i analogowymi (*kanałem* w programie nazywane jest pojedyncze wejście i pojedyncze wyjście),
- ilość kanałów.

Karta We/Wy jest zbiorem kanałów o takim samym kierunku i typie, karta zajmuje jedno gniazdo (*ang. slot*) sterownika.

Biblioteka zespołów We/Wy zawiera definicje zespołów kart, które mogą mieć różne typy i kierunki. *Zespół We/Wy* zajmuje, podobnie jak karta, pojedyncze gniazdo sterownika.

Biblioteki *kart* i *zespołów* We/Wy zawierają definicje rzeczywistych modułów wejściowych i wyjściowych, w które może być wyposażony oprogramowywany sterownik. *Menadżer Bibliotek* pozwala na definiowanie tzw. *konfiguracji We/Wy*. Określona *konfiguracja We/Wy* zawiera listę kart i zespołów We/Wy wraz z informacją o gniazdach, do których karty i zespoły te są podłączone. Dodatkowo, w konfiguracji We/Wy mogą być zdefiniowane domyślne nazwy dla ***zmiennych*** skojarzonych z *kanałami* We/Wy.

Biblioteki funkcji i bloków funkcjonalnych zawierają definicje napisanych przez użytkownika funkcji i bloków funkcjonalnych. Umieszczanie funkcji czy bloków funkcjonalnych w bibliotece pozwala na ich wielokrotne wykorzystanie w różnych projektach.

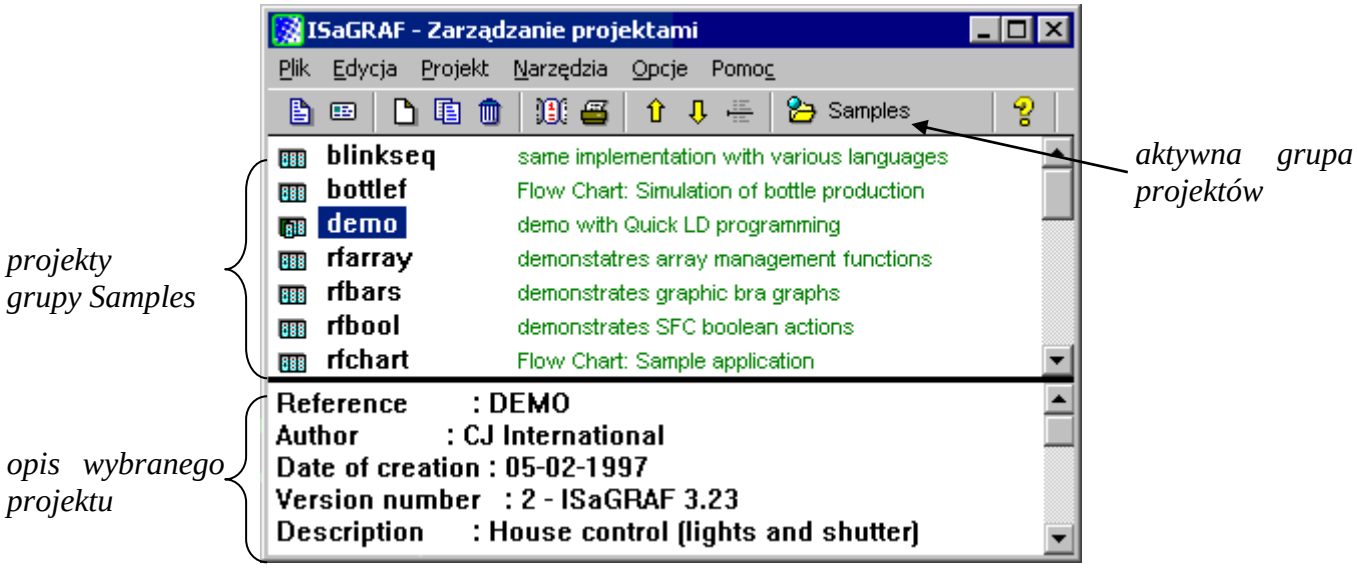
3.3. Projekty

Na *projekt* w ISaGRAF składają się konfiguracja sterownika, zmienne oraz programy służące rozwiązaniu określonego zadania sterowania. Wybór opcji *Projekty* z *Menu Start* powoduje uruchomienie narzędzia: *Zarządzanie projektami*. Program ten pozwala na tworzenie nowych projektów, edycję, archiwizację i usuwanie projektów już istniejących.

Pojedynczy projekt zapisywany jest na dysku w postaci zbioru plików umieszczonych we wspólnym folderze. W osobnych plikach projektu przechowywane są np. informacje o konfiguracji sterownika, wykorzystywanych zmiennych czy napisanych programach. Nazwa projektu jest jednocześnie nazwą folderu w którym zapisywane są pliki projektu.

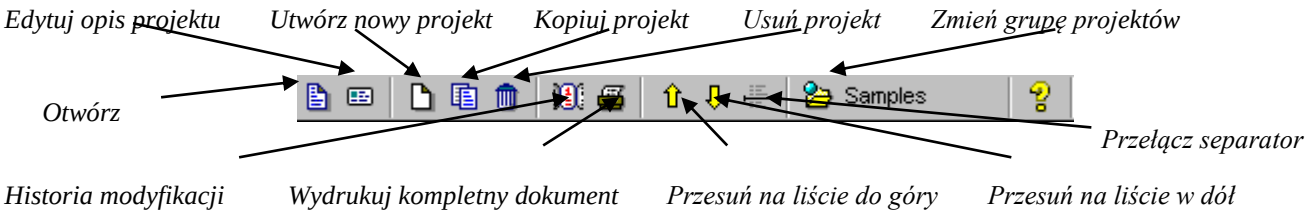
Projekty w ISaGRAF łączone są w tzw. *grupy projektów*. Fizycznie, na dysku, *grupa projektów* jest folderem w którym umieszczane są foldery projektów należących do tej grupy. Po instalacji programu domyślnie tworzone są dwie grupy projektów: *Default* i *Samples*. Grupa *Default* jest pustą grupą (właściwe zawiera czysty projekt *creation*) przygotowaną wstępnie na tworzone w programie projekty – grupie odpowiada znajdujący się w folderze głównym pakietu (domyślnie „\ISAWIN") podfolder „\Apl". Grupa *Samples* zawiera przykładowe projekty dostarczane z pakietem – projekty tej grupy zapisane są na dysku w podfolderze „\Smp". Narzędzie *Zarządzanie projektami* pozwala tworzenie nowych grup projektów i przełączanie pomiędzy już istniejącymi grupami.

Na *rys.3.2.* przedstawiony został wygląd narzędzia *Zarządzanie projektami* z aktywną grupą *Samples*.



Rys. 3.2. Okno Zarządzanie projektami.

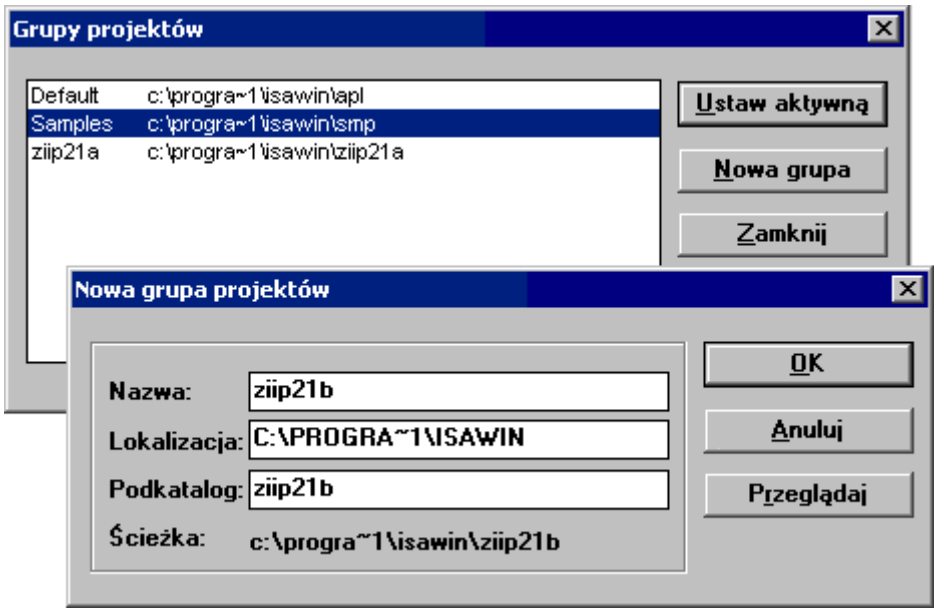
W górnej części okna programu znajdują się: *menu* z listą dostępnych funkcji oraz *pasek narzędzi* zawierający najczęściej wykonywane operacje.



Rys. 3.3. Pasek narzędzi okna Zarządzanie projektami.

3.3.1. Zmiana grupy aktywnej i tworzenie nowej grupy

Zmianę aktywnej grupy i tworzenie nowych grup projektów umożliwia przycisk paska narzędzi oraz opcja menu **Plik|Zmień grupę projektów**. Po wybraniu polecenia, wyświetlane jest okno *Grupy projektów*.



Rys. 3.4. Okno Grupy projektów i Nowej grupy projektów.

Przycisk **Ustaw aktywną** uaktywnia w oknie *Zarządzanie projektami* wybraną w liście grupę, okno *Grupy projektów* jest zamykane.

Stworzenie nowej grupy wymaga ustalenia nazwy i folderu dla grupy. Parametry te można określić po wybraniu polecenia **Nowa grupa**. Folder definiowany jest poprzez wskazanie folderu nadrzędnego (pole: *Lokalizacja*) oraz właściwego folderu (pole: *Podkatalog*).

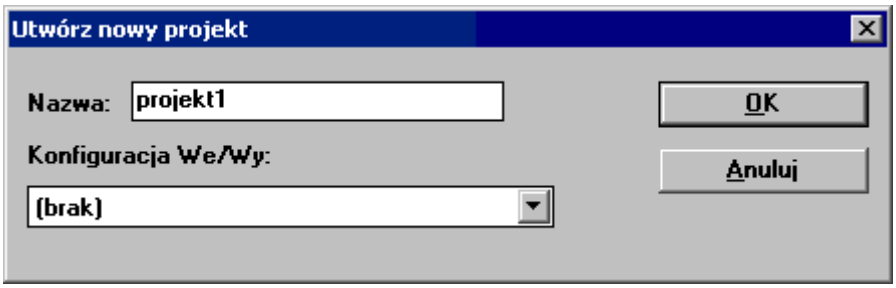
Uwaga 1. Wszystkie podawane w programie nazwy:

- *muszą składać się wyłącznie z liter, cyfr i znaku „_”,*
- *muszą zaczynać się od litery,*
- *nie mogą zawierać więcej niż 8 znaków.*

Domyślnym folderem głównym jest folder główny pakietu (na powyższym rysunku jest nim „C:\Program Files\ISAWIN” – w nazwach folderów obowiązuje ograniczenie długości do 8 znaków, dłuższe nazwy muszą być skracane przy użyciu symbolu „~”). Nazwa właściwego folderu ustalana jest domyślnie na podstawie nazwy grupy, nazwy te mogą się jednak różnić.

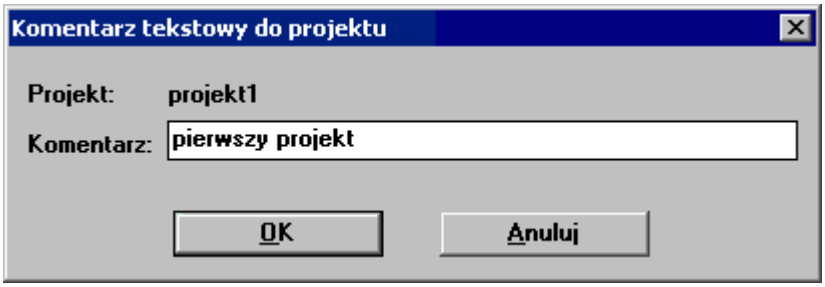
3.3.2. Tworzenie nowego projektu

Po wybraniu w oknie *Zarządzanie projektami* aktywnej grupy projektów, można przejść do zdefiniowania nowego projektu tej grupy. Przycisk paska narzędzi oraz opcja menu **Plik| Nowy**, otwierają okno za pomocą którego można ustalić nazwę projektu (ograniczenia stosowane przy nadawaniu nazw omówione zostały w *Uwadze 1.*). Nowy projekt może (ale nie musi) być utworzony ze wstępnie zdefiniowaną w *Menadżerze Bibliotek konfiguracją We/Wy* (patrz: punkt 3.2.).

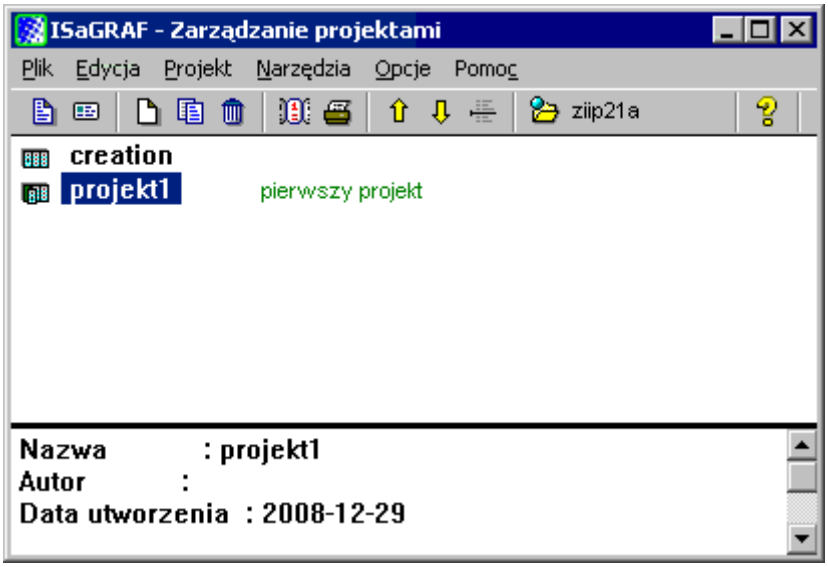


Rys. 3.5. Okno Utwórz nowy projekt.

Po zamknięciu okna, nowy projekt (tutaj: „*projekt1*") jest widoczny w liście projektów okna *Zarządzanie projektami*. Projekt można dodatkowo opisać krótkim komentarzem po wybraniu opcji menu **Edycja|Komentarz tekstowy**.



Rys. 3.6. Okno Komentarz tekstowy do projektu.



Rys. 3.7. Okno Zarządzanie projektami z utworzonym projektem „projekt1".

Właściwe przygotowanie projektu odbywa się za pomocą okna *Programy*, które jest wyświetlane po wybraniu przycisku , opcji menu **Plik|Otwórz** lub po dwukrotnym kliknięciu na wybranym projekcie w liście. Obsługa okna *Programy* omówiona została w punkcie 4.2.

3.4. Diagnostyka

Po wybraniu opcji *Diagnostyka* z *Menu Start* uruchamiane jest tzw. *narzędzie diagnostyczne*. Program uruchamia debagera dla wskazanego projektu tylko wtedy gdy projekt ten jest również załadowany i pracuje na sterowniku.

4. PROJEKT

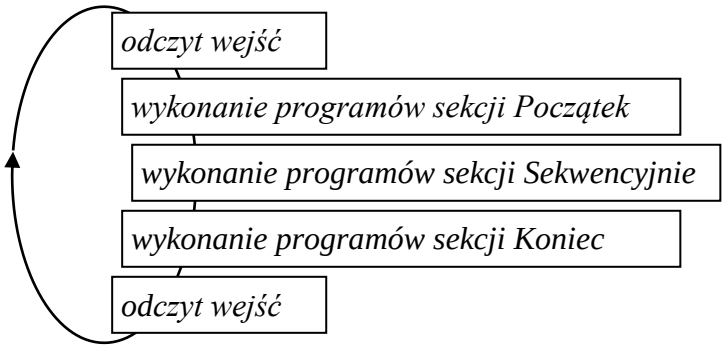
4.1. Elementy projektu

Konfiguracja We/Wy sterownika

Konfiguracja może być ustalana na etapie tworzenia nowego projektu (patrz punkt 3.3.2.) na podstawie elementów zdefiniowanych w *Menadżerze Bibliotek*. Wstępnie ustaloną konfigurację można modyfikować w trakcie pracy nad projektem w oknie projektu *Programy*,

Programy

Programy to podstawowe jednostki organizacyjne realizujące wybrane zadanie sterowania. W ISaGRAF programy pojedynczego projektu przypisywane są do jednej z trzech sekcji: *Początek*, *Sekwencyjnie* i *Koniec*. Programy wywoływane są w cyklu przedstawionym na poniższym rysunku.



Rys.4.1. Cykl programowy w ISaGRAF.

Programy sekcji *Początek i Koniec* mogą być napisane w językach FBD, LD, ST i IL, a programy sekcji *Sekwencyjnie* w języku SFC i w zbliżonym do niego, ale nie zdefiniowanym w normie, języku *grafów przepływowych* FC (*ang. Flow Chart*). Każdy program musi być zapisany w jednym języku, wyjątkiem są języki FBD i LD, które mogą być łączone.

Zmienne

Zmienne przechowują wartości danych wejściowych i wyjściowych oraz wartości danych pomocniczych. Program pozwala na odwołania do zmiennych poprzez podanie nazwy wcześniej zadeklarowanej

zmiennej (deklarowane odbywa się przy pomocy dostępnego w projekcie *Słownika*), a w przypadku danych We/Wy również w sposób *bezpośredni*. Odwołanie bezpośrednie rozpoczyna się od znaku „%", po którym określony jest kierunek (wejście, wyjście) i typ zmiennej. W przypadku gdy zmienna pobiera lub ustawia wartości wprost na *karcie We/Wy* w odwołaniu podawany jest numer *gniazda* do którego włączona jest karta oraz numer *kanału* z którym powiązana jest zmienna. Jeżeli zmienna odwołuje się do wartości odczytywanych lub zapisywanych przez *zespół We/Wy* dodatkowo podawany jest również indeks pojedynczej karty w zespole. W odwołaniach bezpośrednich stosowane są następujące symbole:

I, Q – zmienna skojarzona odpowiednio z kanałem wejściowym i wyjściowym,

X, D – zmienna typu bitowego i liczbowego.

Schemat odwołania bezpośredniego jest następujący:

%dtg.k – dla zmiennej skojarzonej z kartą We/Wy,

%dtg.i.k – dla zmiennej skojarzonej z zespołem We/Wy,

gdzie: *d, t* – kierunek i typ zmiennej, *g* – numer gniazda, *i* – indeks karty w zespole, *k* – numer kanału.

Przykłady:

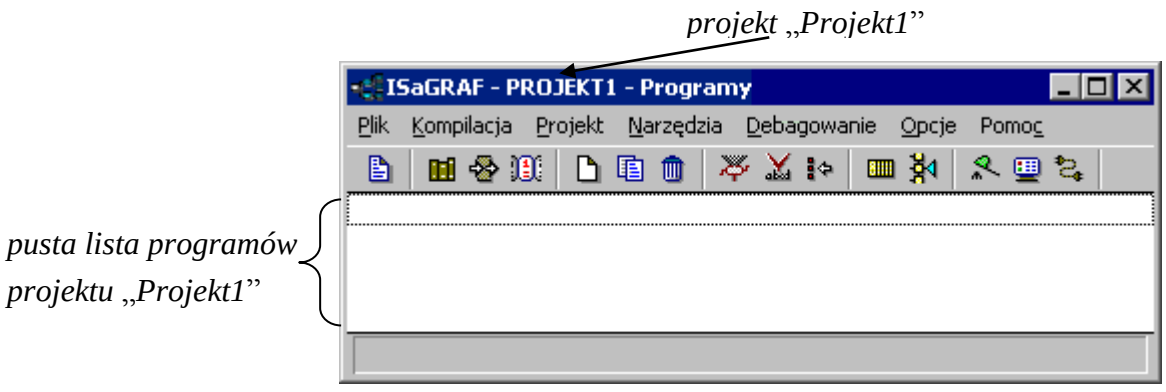
%IX0.1 – wejściowa zmienna bitowa skojarzona z pierwszym *kanałem karty We/Wy* włączonej do gniazda o numerze 0,

%QX1.2 – wyjściowa zmienna bitowa skojarzona z drugim *kanałem karty We/Wy* włączonej do gniazda o numerze 1,

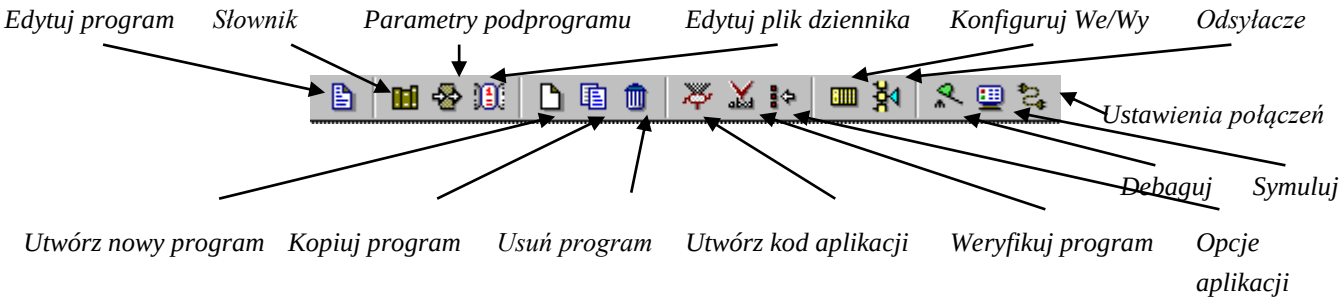
%IX2.1.3 – wejściowa zmienna bitowa skojarzona z trzecim *kanałem* pierwszej *karty zespołu We/Wy* włączonego do gniazda o numerze 2.

4.2. Okno Programy

Okno *Programy* daje dostęp do wszystkich elementów tworzonego projektu. Okno to wyświetlane jest z poziomu okna *Zarządzanie projektami*, po wybraniu przycisku , opcji menu **Plik|Otwórz** lub po dwukrotnym kliknięciu na wybranym projekcie w liście.



Rys. 4.2. Okno Programy.

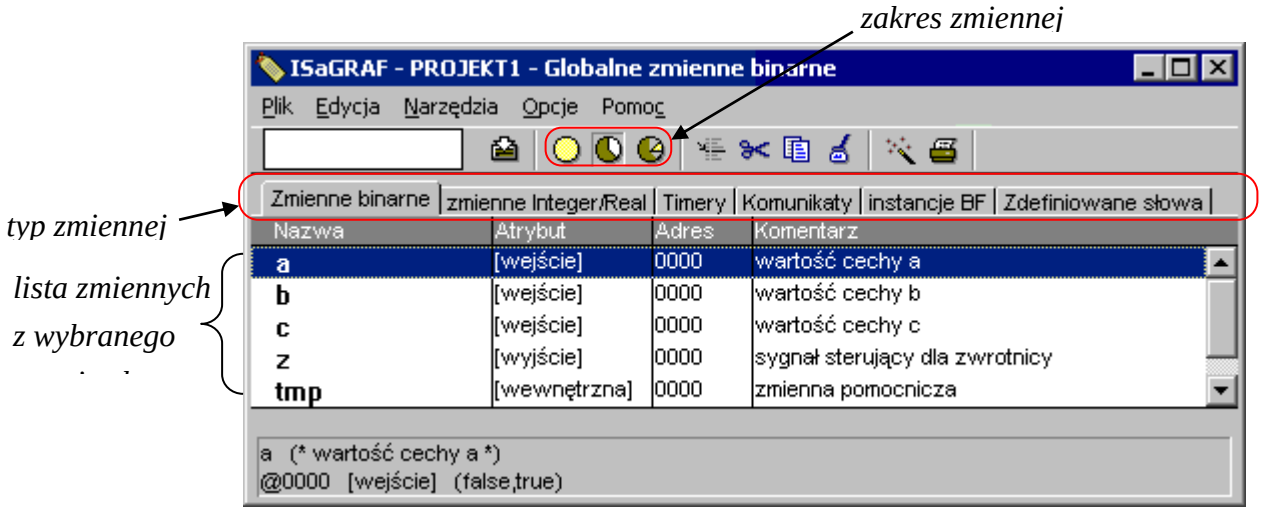


Rys. 4.3. Pasek narzędzi okna Programy.

4.2.1. Słownik

ISaGRAF pozwala na odwołania do zmiennych skojarzonych z wejściami i wyjściami sterownika w sposób *bezpośredni* (patrz: punkt poprzedni), zmienne pomocnicze muszą być jednak specjalnie deklarowane. Odwołania bezpośrednie do zmiennych wejściowych i wyjściowych mogą przyczyniać się do powstawania błędów, dlatego też zmienne te lepiej jest zadeklarować przypisując im odpowiednie nazwy, które charakteryzują rolę jaką odgrywają one w projekcie. Deklaracje zmiennych można wykonać korzystając ze *Słownika*. Okno *Słownika* wyświetlane jest z poziomu okna *Programy*, po wybraniu przycisku lub opcji menu **Plik|Słownik**.

Zmienne wejściowe i wyjściowe zawsze mają charakter *globalny*, tzn. są dostępne we wszystkich programach projektu. Zmienne pomocnicze mogą być zarówno *globalne* jak i *lokalne*, tzn. mogą być dostępne tylko w wybranym programie. W przypadku deklarowania zmiennych *lokalnych*, przed wejściem do *Słownika* należy wybrać program, dla którego deklarowane będą zmienne.



Rys. 4.4. Okno Słownika.

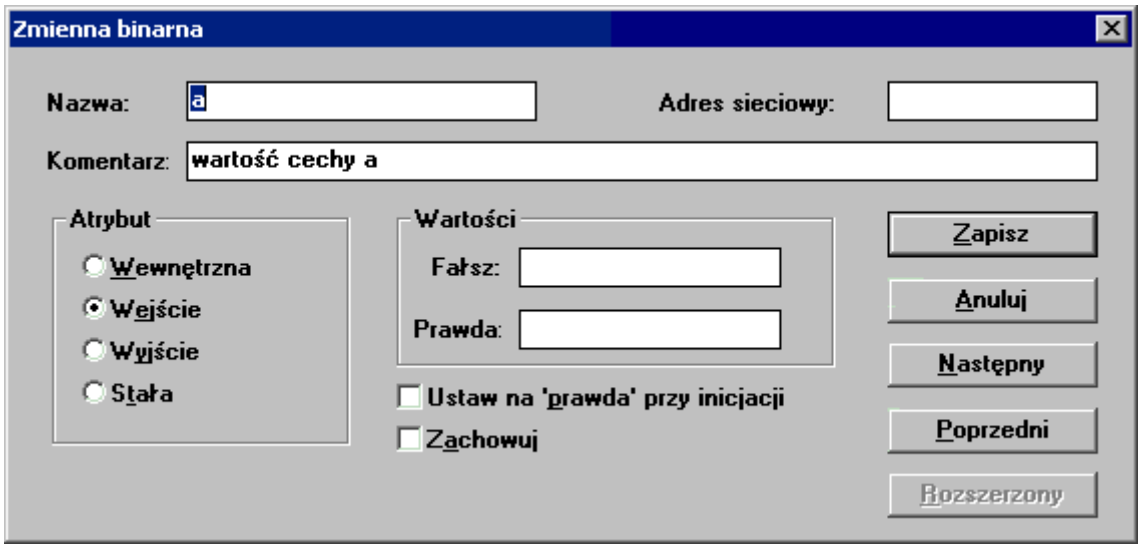
Okno *Słownika* zawiera listę zmiennych wybranego *typu* i *zakresu*. *Typ* zmiennej określa zbiór wartości, które może przyjmować zmienna, np. zmienne binarne (bitowe) przechowują wartości prawda i fałsz, zmienne Integer/Real przechowują liczby całkowite lub liczby rzeczywiste, itd. Zakres zmiennej określa jej charakter -. zmienna *globalna* lub *lokalna*.

Klikając na nagłówkach zakładek (zmienne binarne, zmienne Integer/Real, ...) można filtrować listę zmiennych Słownika. Wybór właściwej zakładki jest konieczny podczas deklarowania nowej zmiennej – typ zmiennej odpowiada typowi aktualnie wybranej zakładki.

Klikanie na przyciskach , pozwala na filtrowanie listy tak aby pokazywała ona tylko zmienne *globalne* (przycisk) lub tylko zmienne *lokalne* (przycisk). Podobnie jak w przypadku typu zmiennej, wybór właściwego zakresu jest konieczny podczas deklarowania nowej zmiennej – zakres zmiennej odpowiada aktualnemu zakresowi.

Typ i zakres wyświetlanych zmiennych można również określić wybierając z menu **Plik|Inne** opcje **Zmienne globalne|Zmienne binarne** (itp.) lub **Zmienne lokalne|Zmienne binarne** (itp.).

Nową zmienną o ustalonym wcześniej typie i zakresie można zadeklarować wybierając przycisk lub opcję menu **Edycja|Nowy**.



Rys. 4.5. Okno do tworzenia i edycji zmiennej binarnej.

Nazwa zmiennej oraz jej atrybut są podstawowymi cechami, które muszą być ustalone przy deklaracji.

Podczas nazywania zmiennych obowiązują podobne ograniczenia jak przy nazywaniu grup projektów (patrz: Uwaga na str. 4), jednak maksymalna długość nazwy zmiennej jest ograniczona do 16 (a nie do 8) znaków.

Atrybut definiuje rodzaj zmiennej, tzn. zmienna może być:

- zmienną *wewnętrzną* – czyli zmienną pomocniczą,
- zmienną *wejściową* skojarzoną z wybranym kanałem wejściowym,
- zmienną *wyjściową* skojarzoną z wybranym kanałem wyjściowym,
- stałą – czyli pomocniczą zmienną o ustalonej wartości początkowej, której nie można zmieniać w trakcie działania programu.

Kojarzenie zmiennych wejściowych i wyjściowych z odpowiednimi kanałami odbywa się przy pomocy narzędzia *Konfiguruj We/Wy* – narzędzie to zostało omówione w następnym podpunkcie. Brak połączenia

nawet pojedynczej zmiennej wejściowej czy wyjściowej powoduje błędy, które uniemożliwiają uruchomienie napisanych programów.

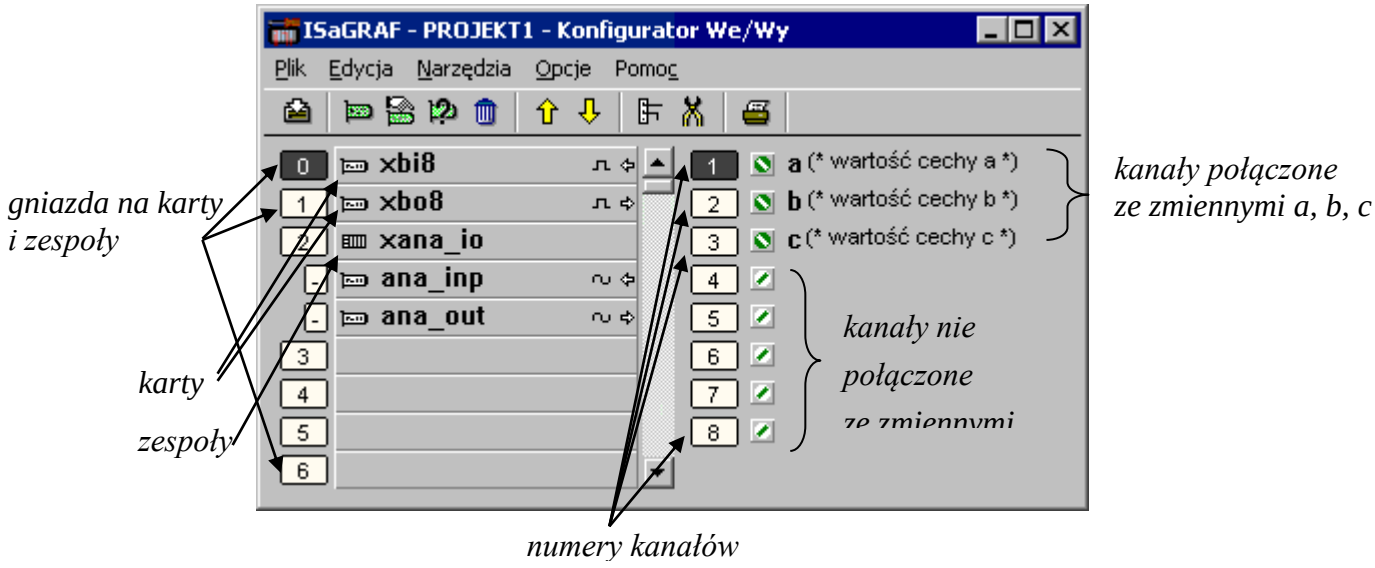
Deklarację wybranej zmiennej kończy operacja zapisu (przycisk *Zapisz*). Po zapisie, program automatycznie przechodzi do deklarowania kolejnej zmiennej. Deklarowanie zmiennych przerywa przycisk *Anuluj*, po wybraniu tego przycisku oknem bieżącym staje się ponownie okno *Słownika*. Zapis zmiennych realizowany przyciskiem *Zapisz* ma charakter tymczasowy – operacja ta musi być dodatkowo potwierdzona np. podczas wychodzenia z okna *Słownika*. Użytkownik może również sam zatwierdzić wprowadzone modyfikacje wybierając przycisk lub opcję menu **Plik|Zapisz**.

Na *rys. 4.5.* pokazana została deklaracja *wejściowej* zmiennej o nazwie „*a*”, zmiennej ten przypisany został dodatkowo komentarz „*wartość cechy a*”. Komentarz jest dłuższym opisem określającym rolę zmiennej w projekcie, może on być wyświetlany w wielu przypadkach obok nazwy zmiennej. Na *rys. 4.4.* widoczne są *binarne* zmienne *globalne* przykładowego projektu „*projekt1*”. W projekcie zadeklarowane zostały trzy zmienne wejściowe: *a*, *b* i *c*, jedna zmienna wyjściowa: *z* i jedna zmienna wewnętrzna: *tmp*.

4.2.2. Konfiguracja We/Wy

Konfiguracja We/Wy sterownika obejmuje listę *kart* i *zespołów We/Wy* wraz z informacją o gniazdach, do których *karty* i *zespoły* są podłączone. Dodatkowo w konfiguracji We/Wy mogą być definiowane powiązania pomiędzy zadeklarowanymi wcześniej *zmiennymi wejściowymi* i *wyjściowymi* a kanałami kart sterownika.

Okno *Konfiguratora We/Wy* wyświetlane jest z poziomu okna *Programy*, po wybraniu przycisku lub opcji menu **Projekt|Konfiguruj We/Wy**. Okno podzielone jest na dwie części – z lewej strony widoczna jest lista gniazd oraz przyłączonych do nich kart i zespołów, z prawej strony lista kanałów i połączonych z nimi zmiennych aktualnie wybranej karty.



Rys. 4.6. Okno Konfiguratora We/Wy.

W oknie *Konfiguratora* wykorzystywane są następujące symbole graficzne:

– *karta i zespół We/Wy* (symbole widoczne na prawo od numerów gniazd),

– *karta z kanałami binarnymi, liczbowymi, karta z kanałami wejściowymi i wyjściowymi* (symbole widoczne na prawo od nazwy *karty*),

– *kanał połączony i kanał niepołączony.*

Jeżeli na etapie tworzenia nowego projektu ustalona została wstępna konfiguracja sterownika w wybranych gniazdach widoczne są zdefiniowane w tej konfiguracji karty i zespoły oraz ewentualne powiązania zadeklarowanych w konfiguracji zmiennych z kanałami. Jeżeli konfiguracja nie została określona, okno *Konfiguratora* jest puste a odpowiednie ustawienia należy wykonać posługując się dostępnymi tutaj funkcjami.

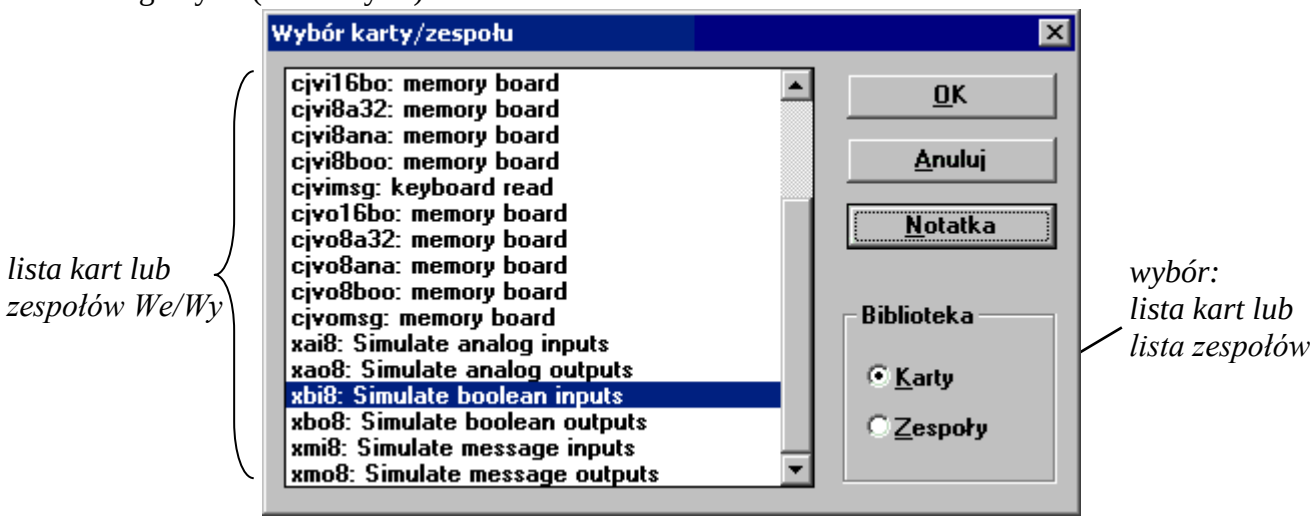
Sterownik, którego konfiguracja pokazana została na powyższym rysunku ma:

- w gnieździe nr 0 kartę z wejściowymi kanałami binarnymi, trzy pierwsze kanały tej karty połączone są ze zmiennymi *a*, *b*, i *c*,
- w gnieździe nr 1 kartę z wyjściowymi kanałami binarnymi,
- w gnieździe nr 2 zespół, w którego skład wchodzą dwie karty z kanałami analogowymi (jedną wejściową i jedną wyjściową).

Ustalanie karty lub zespołu

Przyłączając kartę lub zespół do gniazda sterownika należy:

- wybrać właściwe gniazdo – podświetlając myszą numer gniazda,
- ustalić typ karty lub zespołu – lista wszystkich możliwych typów jest dostępna pod przyciskiem lub w menu **Edycja|Ustaw kartę/zespół**. Wybór typu karty lub zespołu sprowadza się do wskazania nazwy elementu w liście okna *Wybór karty/zespołu*. Przełączanie pomiędzy dostępnymi kartami i zespołami odbywa się przy pomocy pól: *Karty* i *Zespoły*. Po instalacji wersji demonstracyjnej dostępne są np. wirtualne karty: xbi8, xbo8 – karta z 8 wejściami i karta z 8 wyjściami cyfrowymi (binarnymi), xai8, xao8 – karta z 8 wejściami i karta z 8 wyjściami analogowymi (liczbowymi).



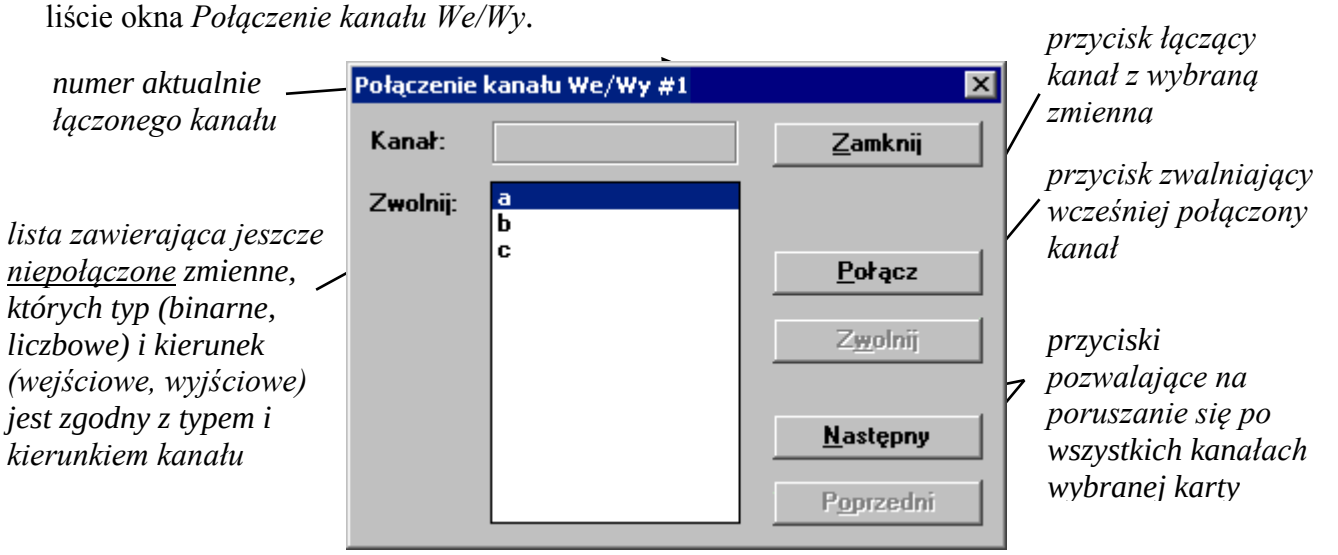
Rys. 4.7. Okno wyboru karty lub zespołu We/Wy.

Po ustaleniu kart i zespołów sterownika można jeszcze ustalić powiązania wcześniej zadeklarowanych zmiennych wejściowych i wyjściowych z odpowiednimi kanałami kart.

Ustalanie połączenia kanałów ze zmiennymi

Ustalając połączenie należy:

- wybrać właściwy kanał – podświetlając myszą numer kanału,
- wybrać odpowiednią zmienną – lista zmiennych jest dostępna pod przyciskiem lub w menu **Edycja|Ustaw kanał/parametr**. Wybór zmiennej sprowadza się do wskazania nazwy zmiennej w liście okna *Połączenie kanału We/Wy*.



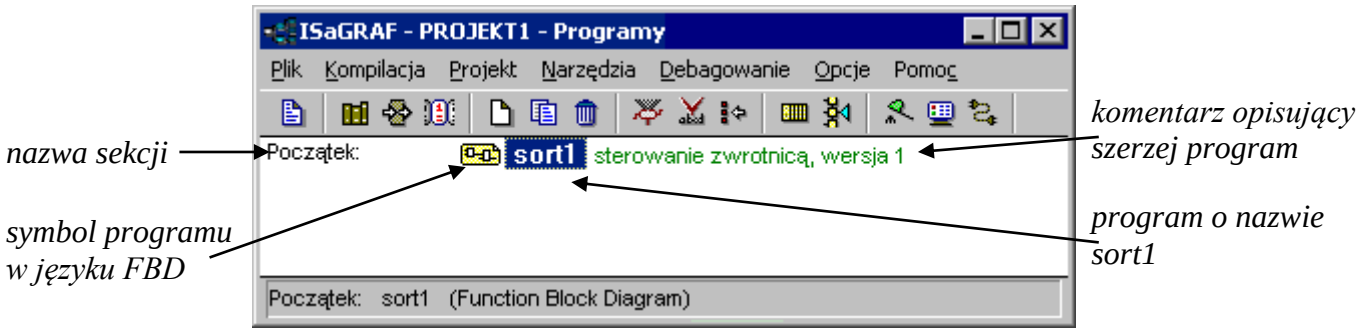
Rys. 4.8. Okno łączenia kanałów We/Wy.

4.2.3. Programy

Programy są podstawowymi jednostkami organizacyjnymi realizującymi wybrane zadanie sterowania. ISaGRAF pozwala na tworzenie aplikacji we wszystkich językach definiowanych w normie IEC 61131: FBD, LD, ST, IL oraz SFC.

Programy, które powinny być wykonane na początku każdego cyklu programowego sterownika powinny być przypisane do sekcji *Początek*, programy wykonywane na końcu cyklu do sekcji *Koniec*. Programy sekcji *Początek* i *Koniec* mogą być napisane w językach FBD, LD, ST i IL. Programy napisane w językach SFC i FC muszą być przypisane do sekcji *Sekwencyjnie*.

Operacje dodawania, edycji i usuwania programów dostępne są z poziomu okna *Programy*.



Rys. 4.9. Okno programy z programem „sort1" przypisanym do sekcji Początek.

W oknie *Programy* wykorzystywane są następujące symbole graficzne do oznaczenia języka, w którym został napisany program:

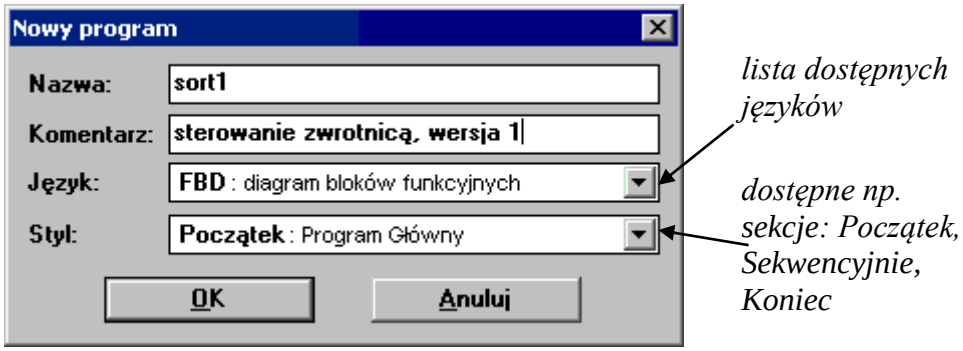
– program w języku FBD, – program w języku LD, – program w języku SFC,

– program w języku ST, – program w języku IL.

Dodanie nowego programu

Przycisk paska narzędzi oraz opcja menu **Plik|Nowy**, otwierają okno za pomocą którego można ustalić nazwę, język i sekcję nowego programu.

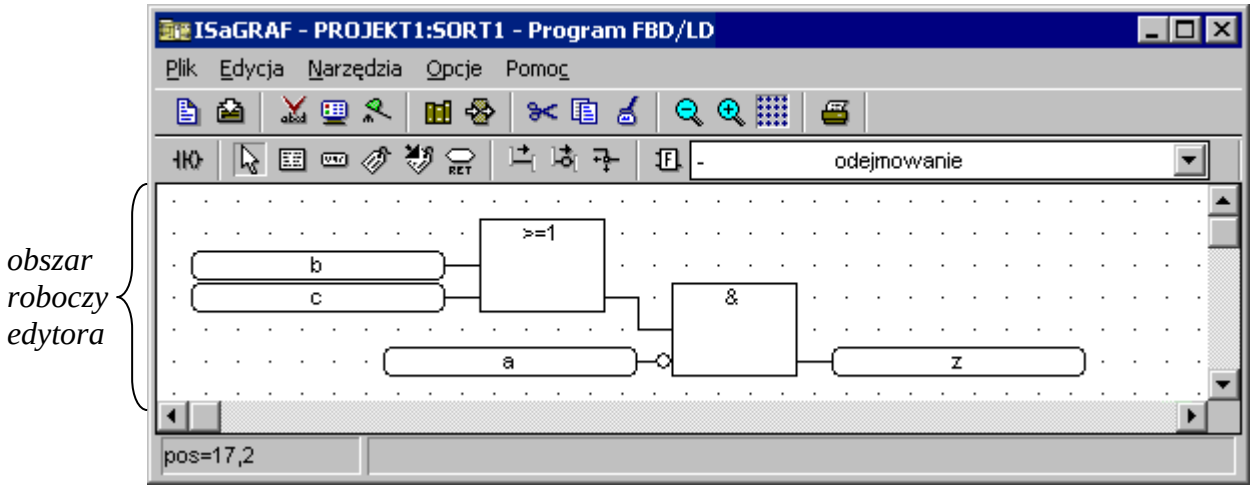
Podczas nazywania programów obowiązują takie same ograniczenia jak przy nazywaniu np. grup projektów (patrz: Uwaga na str. 4) – maksymalna długość nazwy jest ograniczona do 8 znaków. Tak krótka nazwa nie wyjaśnia funkcji realizowanych przez program, z tego powodu dobrze jest opisać program odpowiednim komentarzem.



Rys. 4.10. Tworzenie nowego programu.

Akceptacja okna *Nowy program* dodaje program do odpowiedniej sekcji w liście okna *Programy*. Sposób definiowania programu zależy od ustalonego wcześniej języka programu. Przycisk paska narzędzi oraz opcja menu **Plik|Otwórz**, otwierają właściwy dla języka programu edytor.

5. EDYCJA PROGRAMU W JĘZYKU FBD

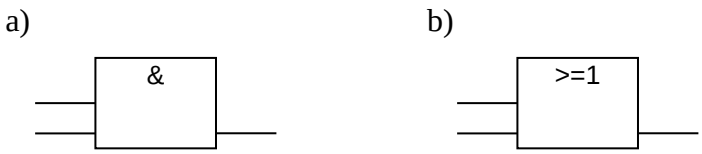


Rys. 5.1. Edytor programów w języku FBD/LD.

Edytor pozwala na edycję programów w języku FBD, których fragmenty mogą być zapisane w języku LD. ISaGRAF posiada również edytor dla programów pisanych w języku LD z fragmentami w języku FBD.

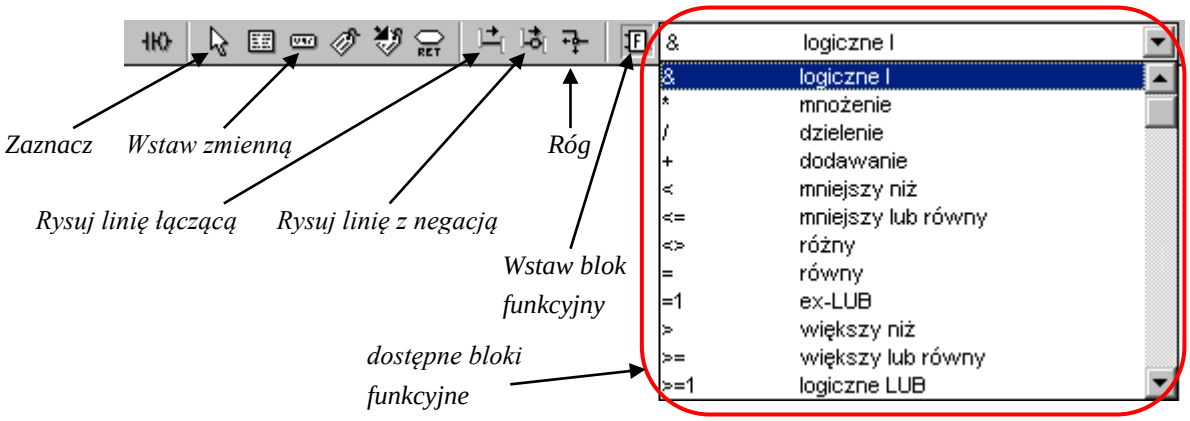
Język FBD pozwala na zapis programu w postaci bloków operacji logicznych powiązanych ze sobą liniami przepływu sygnałów, – jest wzorowany na schematach logicznych. Obok bloków operacji logicznych w języku można wykorzystywać również bloki operacji arytmetycznych oraz bloki funkcjonalne przerzutników, timerów, liczników. Wejścia i wyjścia bloków mogą być negowane poprzez wprowadzanie symbolu „○” na odpowiednich wejściach czy wyjściu.

Podstawowe operacje logiczne: koniunkcja i alternatywa reprezentowane są symbolami:



Rys. 5.2. Symbole graficzne a) koniunkcji b) alternatywy.

Wprowadzanie programu w edytorze sprowadza się do ustawiania w obszarze roboczym odpowiednich bloków, zmiennych a następnie rysowaniu połączeń pomiędzy tymi elementami. Na poniższym rysunku przedstawiony został pasek narzędzi edytora z zaznaczonymi elementami omawianymi w bieżącym punkcie.



Rys. 5.3. Pasek narzędziowy FBD.

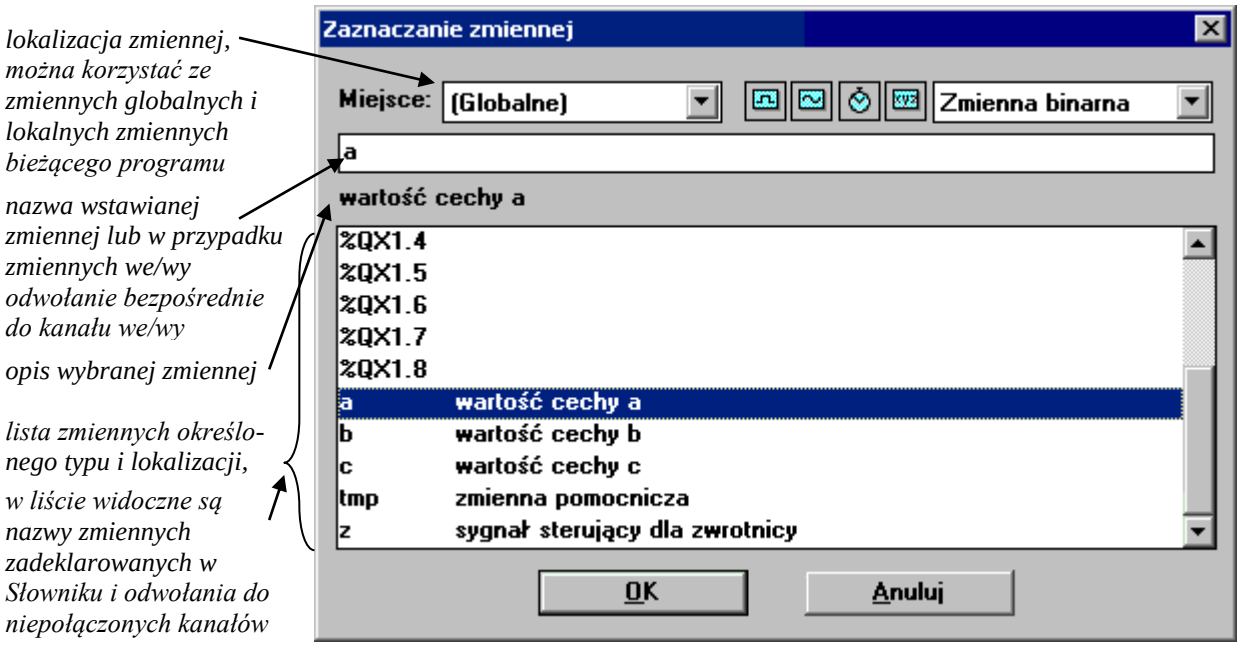
Przycisk – Wstaw blok funkcyjny

Przycisk wstawia, wybrany w liście znajdującej się po jego prawe stronie, blok. Blok reprezentowany jest na schemacie symbolem prostokąta z zaznaczonymi po lewej stronie wejściami (liczba wejść części bloków może być ustalana we właściwościach bloku) i wyjściami po prawej stronie. We wnętrzu prostokąta umieszczane jest oznaczenie bloku a w przypadku części bloków również nazwy ich wejść i wyjść.

Domyślnym blokiem wybranym w liście dostępnych bloków jest arytmetyczne odejmowanie. Operacji koniunkcji odpowiada w liście blok „& logiczne I", a alternatywie blok „>=1 logiczne LUB".

Przycisk – Wstaw zmienną

Przycisk wstawia wskazaną w oknie *Zaznaczanie zmiennej* zmienną. Zmienna reprezentowana jest na schemacie symbolem prostokąta z zaokrąglonymi rogami, nazwa zmiennej umieszczana jest wewnątrz tego prostokąta.



Rys. 5.4. Okno Zaznaczanie zmiennej.

Przyciski i – Rysuj linię łączącą i Rysuj linię z negacją

Przyciski wykorzystywane do łączenia elementów schematu. Połączenie należy rysować od wyjścia pierwszego obiektu do wejścia drugiego obiektu. W przypadku linii z negacją sygnał przesyłany linią jest dodatkowo negowany.

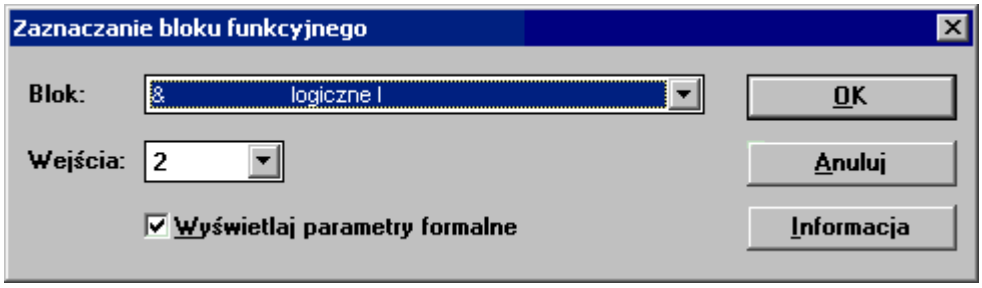
Przycisk – Róg

Przycisk wstawia węzeł reprezentowany na schemacie symbolem ●. Węzły pozwalają na dokładniejsze wskazanie dróg przepływu sygnał (zamiast automatycznie wyznaczanych przez program).

Przycisk – Zaznacz

Przycisk wykorzystywany jest do wybierania elementów umieszczonych w obszarze roboczym edytora. Wybór taki jest niezbędny np. w przypadku przesuwania, kasowania czy ustalania własności elementów schematu. Kliknięcie wewnątrz elementu czy na linii połączenia powoduje wybór wskazanego obiektu, zaznaczenie prostokątnego obszaru powoduje wybór wszystkich obiektów leżących wewnątrz tego obszaru.

Dwukrotne kliknięcie przy wybranym przycisku pozwala na zmianę własności wybranego elementu. Jeżeli elementem tym jest zmienna wyświetlane jest okno *Zaznaczanie zmiennej*, jeżeli elementem tym jest połączenie zmieniany jest jego typ – linia bez negacji zmieniana jest na linię z negacją i odwrotnie. Jeżeli wybranym elementem jest blok wyświetlane jest okno *Zaznaczanie bloku funkcyjnego*. Okno to pozwala na zmianę rodzaju bloku, a w przypadku części bloków również na zmianę liczby ich wejść.



Rys. 5.5. Okno Zaznaczanie bloku funkcyjnego.

Program, którego schemat przedstawiony został na *rys. 5.1.* realizuje funkcję logiczną daną wzorem:

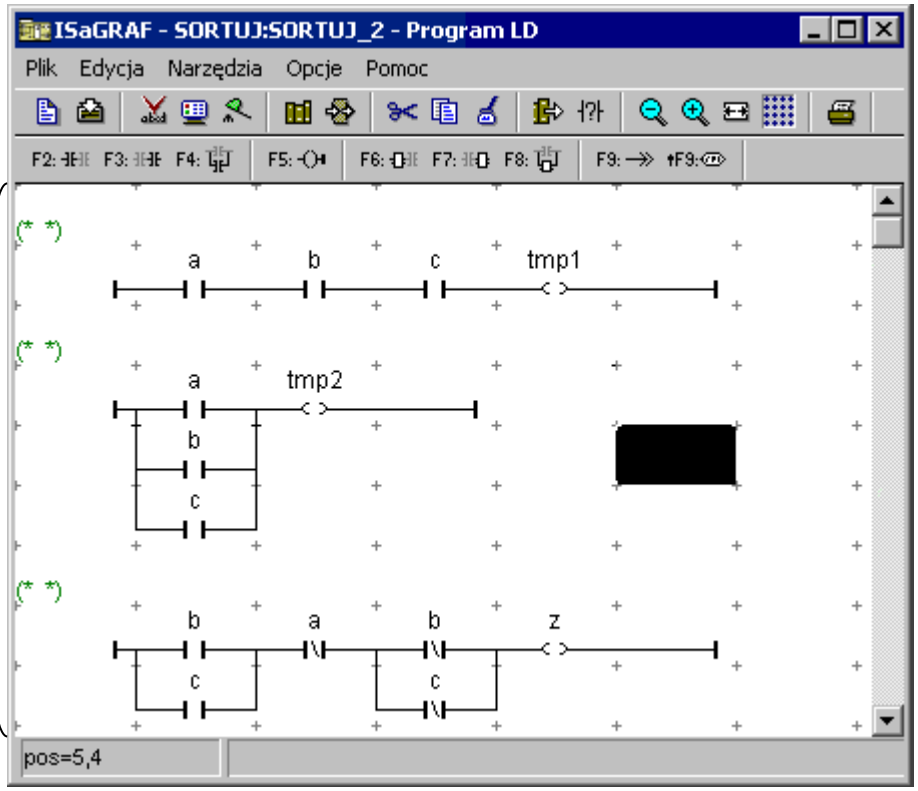
$$z = (b + c)\,\bar{a}\,.$$

Schemat układu realizującego tą funkcję w języku FBD zbudowany jest z:

- czterech zmiennych: *a*, *b*, *c*, *z*,
- jednego bloku „>=1 logiczne LUB”,
- jednego bloku „& logiczne I”,
- czterech zwykłych połączeń i jednego połączenia z negacją.

Po zdefiniowaniu programu okno edytora można już zamknąć zapisując wprowadzone modyfikacje. Kolejny etap pracy polega na przetestowaniu napisanej aplikacji (*patrz punkt Kompilacja i symulacja*).

6. EDYCJA PROGRAMU W JĘZYKU LD



Rys. 6.1. Edytor programów w języku LD.

Edytor pozwala na edycję programów w języku LD z fragmentami w języku FBD.

Język *schematów drabinkowych* LD jest odpowiednikiem schematów (obwodowych, drabinkowych) układów cyfrowych realizowanych w technologii stykowo – przekaźnikowej, dodatkowo język pozwala na wykorzystywanie funkcji realizujących operacje arytmetyczne, logiczne, porównania oraz bloków funkcjonalnych realizujących zadania przerzutników, timerów, liczników.

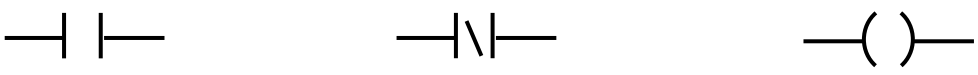
Schemat drabinkowy jest ograniczony z lewej i prawej strony „*szynami prądowymi*”:

Rys. 6.2. Szyny prądowe a) lewa, b) prawa.

Elementy położone pomiędzy lewą i prawą szyną tworzą tzw. „*szczebel*”. Pojedynczy szczebel odpowiada odrębnemu obwodowi prądowemu, w którym przepływ prądu zależy od stanu elementów tego obwodu. Upraszczając można stwierdzić, że każdy ze szczebli schematu definiuje jedną instrukcję programu.

Podstawowymi elementami używanymi podczas definiowania szczebla są: *zestyki zwierne* i *rozwierne* oraz *cewki*. *Zestyki* łączone są w programie ze zmiennymi wejściowymi, wyjściowymi lub wewnętrznymi, *cewki* mogą być łączone tylko ze zmiennymi wyjściowymi i wewnętrznymi. Symbol zmiennej skojarzonej z określonym elementem pokazywany jest nad symbolem elementu.

a) b) c)



Rys. 6.3. a) Zestyk zwierny, b) zestyk rozwierny, c) cewka.

Zestyk zwierny (normalnie otwarty) jest elementem, który pozwala na przepływ prądu (z jego lewej strony na prawą) jeżeli skojarzona z nim zmienna ma wartość *prawda*, jeżeli zmienna ma wartość *fałsz* element nie przewodzi prądu.

Zestyk rozwierny (normalnie zamknięty) pozwala na przepływ prądu jeżeli skojarzona z nim zmienna ma wartość *fałsz*.

Cewka jest elementem wykorzystywanym do nadawania wartości zmiennym wyjściowym lub wewnętrznym, jeżeli linia z lewej strony cewki jest „pod prądem" to skojarzona z cewką zmienna otrzymuje wartość *prawda*, w przeciwnym przypadku wartość *fałsz*.

Typowa definicja prostego szczebla, zaczynając od jego lewej strony, składa się z:

- symbolu lewej szyny prądowej,
- połączonych na różne sposoby symboli zestyków odpowiadających wybranemu wyrażeniu logicznemu,
- symbolu cewki, cewka nadaje wartość określonej zmiennej przypisując jej wartość wynikającą z wartości wyrażenia logicznego wynikającego z postaci szczebla,
- symbolu prawej szyny prądowej.

Przy budowie wyrażenia logicznego z symboli zestyków należy kierować się zasadami:

- połączenie szeregowe zestyków odpowiada koniunkcji zmiennych skojarzonych z tymi zestykami,
- połączenie równoległe zestyków odpowiada alternatywie skojarzonych z nimi zmiennych,
- zestykowi rozwiernemu odpowiada negacja połączonej z nim zmiennej.

Program, którego schemat przedstawiony został na *rys. 6.1.* zbudowany jest z trzech instrukcji (na schemacie widoczne są trzy szczeble). Instrukcje te przypisują wartości kolejno zmiennym: *tmp1*, *tmp2* i *z* (przed symbolem prawej szyny prądowej na każdym ze szczebli umieszczony jest symbol cewki, na pierwszym szczeblu cewka skojarzona jest ze zmienną *tmp1*, na drugim ze zmienną *tmp2*, ...).

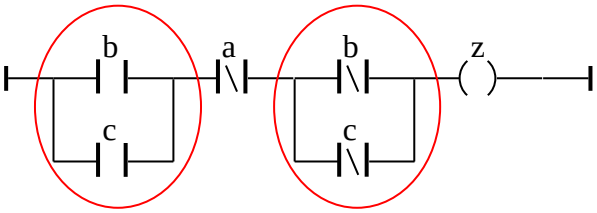
Zestyki szczebla pierwszego połączone są *szeregowo*, więc zmienna *tmp1* otrzymuje wartość wynikającą z wartości wyrażenia:

$$a \cdot b \cdot c\,.$$

Zestyki szczebla drugiego połączone są *równolegle*, więc zmienna *tmp2* otrzymuje wartość:

$$a + b + c\,.$$

Budowa trzeciego szczebla jest bardziej złożona. Na poniższym rysunku szczebel ten został przerysowany z właściwego schematu.



Rys. 6.4. Schemat trzeciego szczebla przykładowego programu.

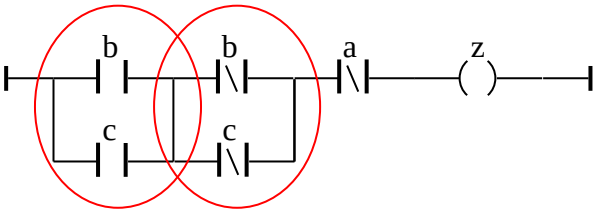
Czerwonymi elipsami zaznaczone zostały *równoległe* połączenia *zestyków zwiernych b* i *c* (pierwsza elipsa) i *równoległe* połączenia *zestyków rozwiernych b* i *c* (druga elipsa). Wyodrębnione fragmenty szczebla tworzą połączenie szeregowe z *zestykiem rozwiernym a*. Zmienna *z* otrzymuje więc wartość wynikającą z wartości wyrażenia:

$$(b+c)\cdot\bar{a}\cdot(\bar{b}+\bar{c}).$$

Wyrażenie to jest równoważne np. wyrażeniu:

$$(b+c)\cdot(\bar{b}+\bar{c})\cdot\bar{a}\,.$$

Na poniższym rysunku przedstawiony został szczebel nadający wartość zmiennej *z* na podstawie nowej postaci wyrażenia:

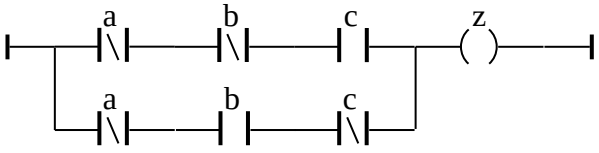


Rys. 6.5. Schemat szczebla równoważnego trzeciemu szczeblowi przykładowego programu.

Wykorzystując prawa algebry Boole'a powyższe wyrażenie można również przekształcić do postaci:

$$\bar{a}\,\bar{b}\,c+\bar{a}\,b\,\bar{c}\,.$$

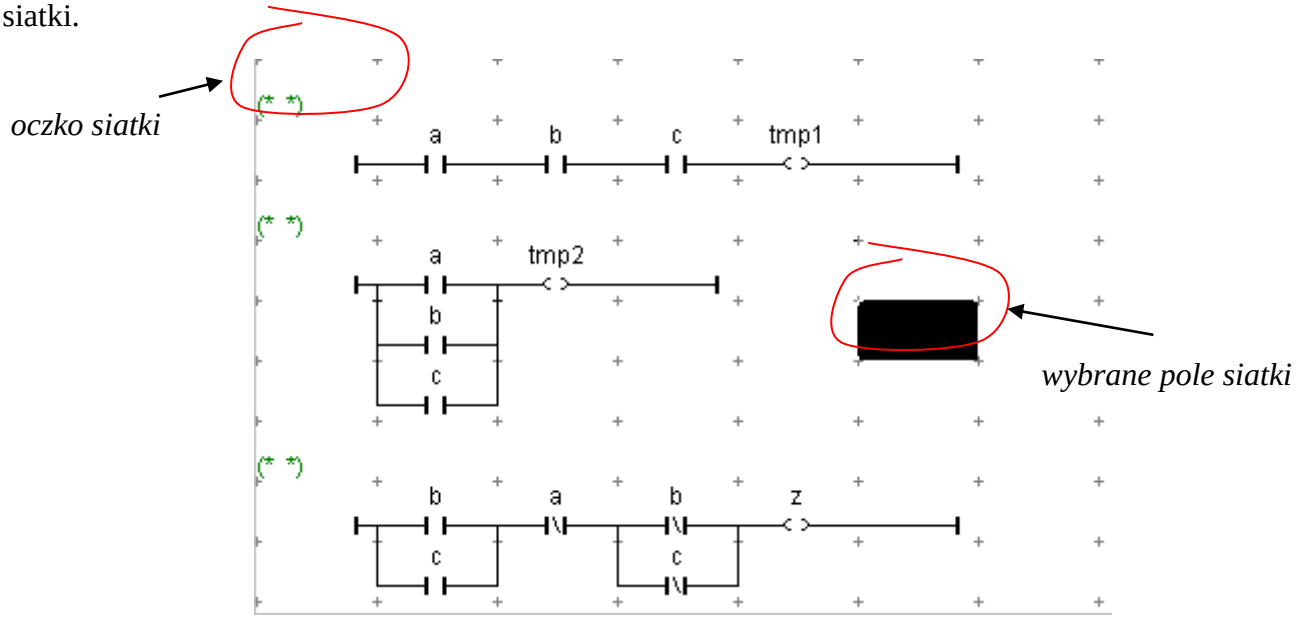
Definicja pierwszego składnika wyrażenia sprowadza się w języku LD do szeregowego połączenia trzech zestyków skojarzonych ze zmiennymi *a*, *b* i *c* – zestyki *a* i *b* muszą być *zestykami rozwiernymi*, a zestyk *c* musi być *zestykiem zwiernym*. Podobnie wygląda definicja drugiego składnika wyrażenia. Składnik ten odpowiada szeregowemu połączeniu zestyków: *rozwiernego a*, *zwiernego b* i *rozwiernego c*. Sumowanie składników sprowadza się do *równoległego* połączenia odpowiednich fragmentów szczebla.



Rys. 6.6. Schemat szczebla równoważnego trzeciemu szczeblowi przykładowego programu.

Wprowadzanie programu w edytorze sprowadza się do wstawiania w obszarze roboczym szczebli i umieszczaniu na nich symboli zestyków i cewek.

Każdy z symboli schematu drabinkowego zajmuje w obszarze roboczym pole o rozmiarze jednego oczka siatki.



Rys. 6.7. Obszar roboczy edytora.

Pola znajdujące się w pierwszej kolumnie siatki mogą zawierać:

- komentarz opisujący funkcje szczebla zdefiniowanego bezpośrednio pod komentarzem – pola te oznaczone są symbolami „(* *)” i wstawiane są automatycznie razem z każdym nowym szczeblem,
- symbol lewej szyny prądowej szczebla.

Wstawianie szczebla

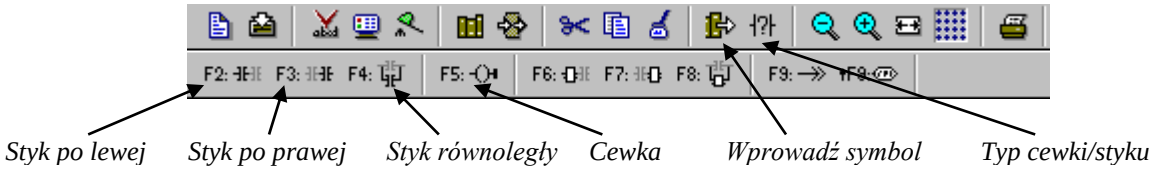
Jedna z metod wstawiania nowego szczebla polega na wyborze opcji menu **Edycja|Wstaw szczebel**. W wyniku tego polecenia, bezpośrednio nad szczeblem w obszarze którego znajduje się aktualnie wybrane pole siatki dodawany jest nowy szczebel. Szczebel ten zbudowany z jednego *zestyku zwiernego* i jednej *cewki*.

Usuwanie szczebla i usuwanie wybranych elementów szczebla

Zbędny szczebel programu można usunąć po wskazaniu lewej szyny prądowej tego szczebla i następnie wybraniu opcji **Edycja|Kasuj** lub po wciśnięciu klawisza DEL. Jeżeli przed wybraniem polecenia zostanie wskazany element lub kilka elementów schematu – polecenie kasowania usunie tylko te elementy. Operacją kasowania nie można zepsuć podstawowej struktury szczebla, czyli np. nie można usunąć prawej szyny prądowej, czy jedynej cewki szczebla.

Przygotowywanie struktury szczebla

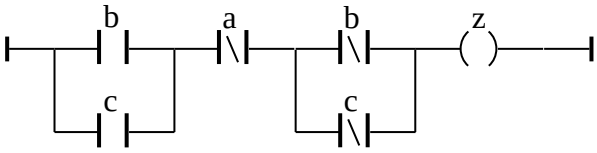
Podczas edycji szczebla wykorzystywane są polecenia menu **Edycja**, przyciski pasków narzędziowych edytora i związane z nimi klawisze szybkiego wyboru. Na poniższym rysunku przedstawione zostały dwa paski narzędzi edytora z zaznaczonymi elementami omawianymi w bieżącym punkcie.



Rys. 6.8. Paski narzędziowy edytora LD.

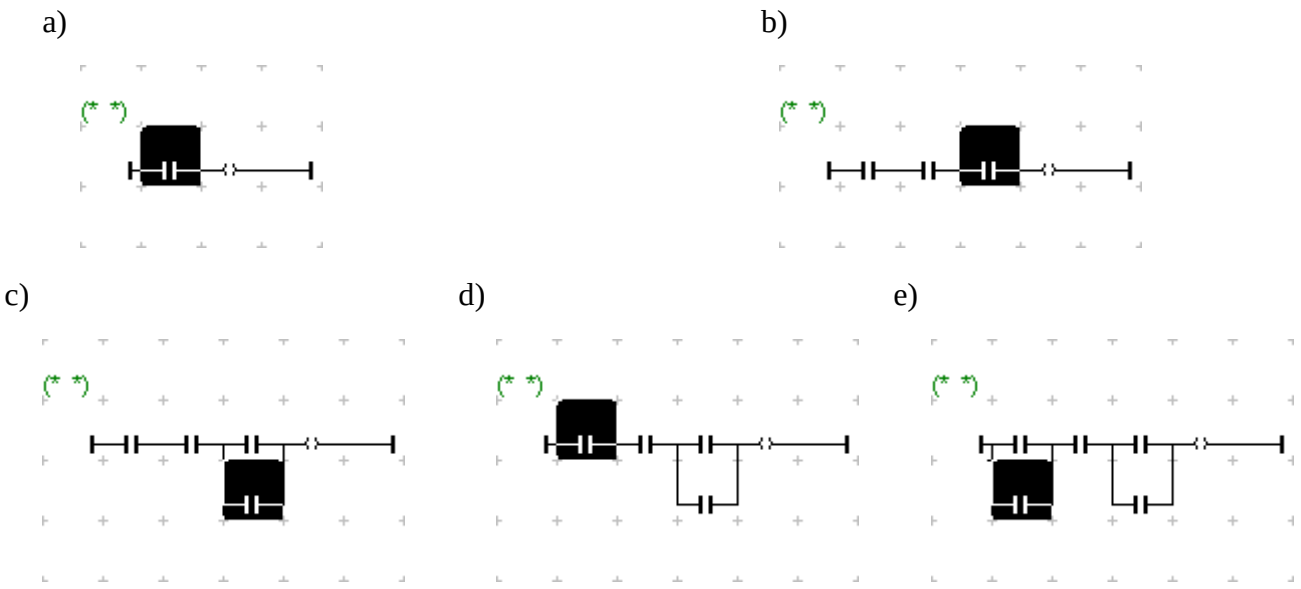
Przyciski F2:, F3:, F4: i odpowiadające im klawisze szybkiego wyboru F2, F3, F4 dodają na szczeblu nowy *zestyk zwierny*. Zestyk ten jest wstawiany w zależności od użytego polecenia: po lewej, po prawej lub równolegle do aktualnego pola lub pól siatki.

Działanie przycisków zostanie wyjaśnione na przykładzie szczebla przedstawionego na poniższym rysunku.



Rys. 6.9. Przykładowy szczebel.

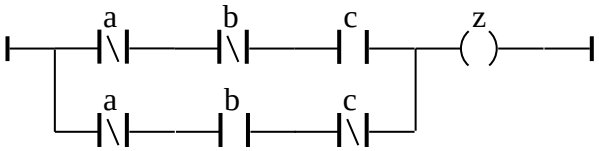
Po wstawieniu nowego szczebla (opcja **Edycja|Wstaw szczebel**) podświetlony jest jego zestyk zwierny (*rys. 6.10.a*). Można więc wybrać przycisk F3:, który doda nowy zestyk na prawo od aktualnie zaznaczonego. Nowo dodany zestyk zostanie automatycznie wybrany, więc kolejny wybór tego samego przycisku spowoduje wstawienie trzeciego zestyku zwiernego (*rys. 6.10.b*)



Rys. 6.10. Kolejne fazy budowy szczebla z rys. 6.9.

Po utworzeniu głównego poziomu szczebla można przejść do dodawania zestyków połączonych równolegle. Ostatnio dodany zestyk jest wybrany, więc użycie przycisku F4: spowoduje dodanie zestyku równoległego do tego zestyku (*rys. 6.10.c*). Teraz brakuje jeszcze tylko zestyku równoległego do pierwszego zestyku szczebla, po jego podświetleniu (*rys. 6.10.d*) wybór przycisku F4: zakończy przygotowywanie struktury przykładowego szczebla.

W niektórych przypadkach, podczas edycji szczebli niezbędny jest wybór kilku pól. Załóżmy, że należy zbudować szczebel przedstawiony na *rys. 6.11*.



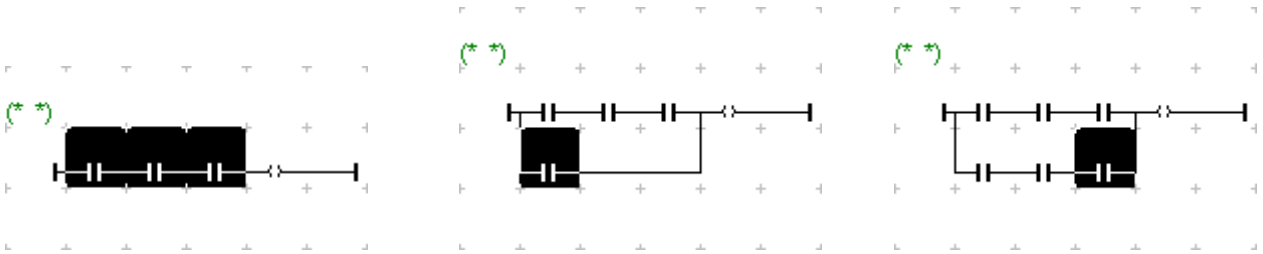
Rys. 6.11. Przykładowy szczebel.

Główny poziom szczebla zbudowany jest z trzech szeregowo połączonych zestyków. Identycznie wyglądał poprzednio edytowany szczebel przed dodaniem dwóch zestyków połączonych równolegle. Etap definiowania tego poziomu zostanie więc pominięty.

Brakujące zestyki muszą tworzyć gałąź równoległą do zestyków znajdujących się na poziomie głównym. Przed dodaniem pierwszego zestyku gałęzi równoległej należy zaznaczyć pola gałęzi podstawowej które utworzą z nowym zestykiem połączenie równoległe (*rys. 6.12.a*). Po wybraniu przycisku F4: pierwszy zestyk drugiej gałęzi będzie znajdował się na właściwym miejscu.

Teraz należy już tylko uzupełnić schemat dwoma zestykami połączonymi szeregowo z nowo dodanym zestykiem. Dodany zestyk został automatycznie wybrany (*rys. 6.12.b*) więc wystarczy użyć dwukrotnie przycisku F3: aby dokończyć budowę struktury tego szczebla (*rys. 6.12.c*).

a) b) c)



Rys. 6.12. Kolejne fazy budowy szczebla z rys. 6.11.

Przycisk F5: może być wykorzystywany tylko do wstawiania cewki równoległej do już dodanej. Konstrukcja taka może być wykorzystywana np. do jednoczesnego przypisywania tej samej wartości kilku zmiennym.

Elementy, tzn. zestyki i cewki, szczebli przedstawionych na *rys. 6.10.e* i *6.12.c* nie zostały jeszcze skojarzone z odpowiednimi zmiennymi. Dodatkowo, typy elementów nie są zgodne z założonymi (*rys.6.9.* i *rys. 6.11.*) – w przypadku zestyków typ określa czy zestyk jest *zwierny* czy *rozwierny*.

Zmiana typu elementów

Po wybraniu elementu, którego typ będzie modyfikowany należy wybrać przycisk , opcję menu **Edycja|Zmień typ styku/cewki** lub nacisnąć klawisz SPACJI. W ISaGRAF można używać zestyków:

- zwiernych,
- rozwiernych,
- reagujących na narastające zbocze sygnału,
- reagujących na opadające zbocze sygnału.

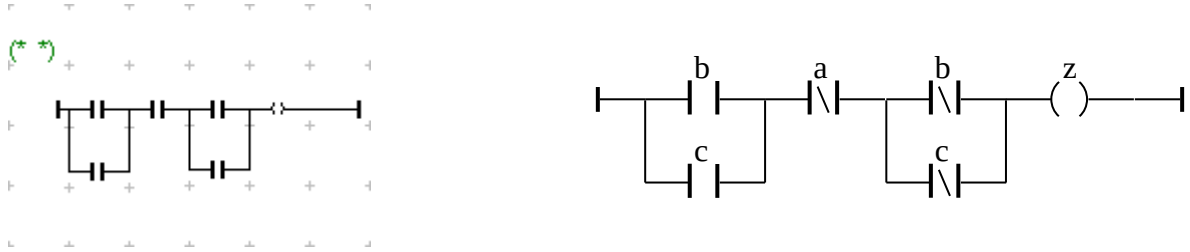
Cewki mogą być:

- zwykłe,
- zanegowane,
- ustawiające,
- kasujące,
- mogą reagować na narastające zbocze sygnału,
- mogą reagować na opadające zbocze sygnału.

Wybór polecenia zmiany typu, powoduje zmianę typu na kolejny – zgodnie z listami typów przedstawionymi powyżej.

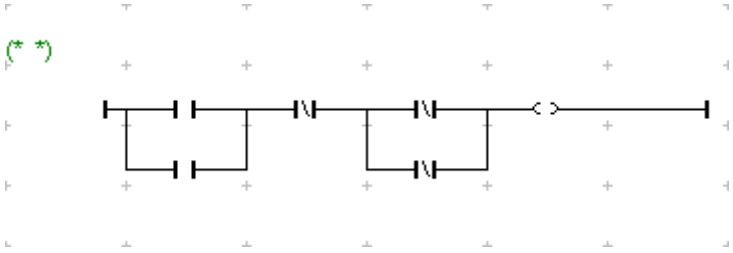
Dopasowanie struktury szczebla przedstawionego na *rys. 6.10.e* do postaci z *rys. 6.9*. wymaga zmiany typu trzech zestyków. Aktualna struktura szczebla i wzorzec tego szczebla zostały przypomniane na poniższym rysunku.

a) b)



Rys. 6.13. Przykładowy szczebel a) aktualna struktura b) wzorzec.

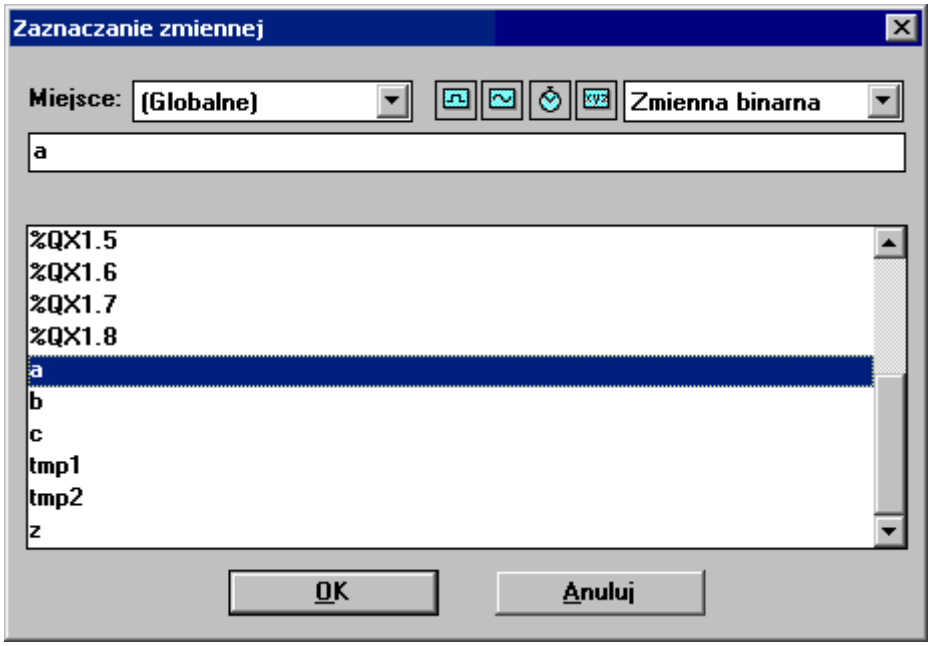
Typy dwóch zestyków z drugiej kolumny arkusza są prawidłowe. Modyfikacji wymagają pozostałe zestyki. *Zestyk rozwierny* jest następnym typem zestyku po *zestyku zwiernym*. W przypadku każdego z modyfikowanych elementów wystarczy więc jednokrotnie wybrać polecenie zmiany typu, czyli np. raz nacisnąć klawisz SPACJI – spowoduje to właściwą zmianę typu.



Rys. 6.14. Poprawiony szczebel z rys. 6.13.

Kojarzenie elementów ze zmiennymi

Zestyki i *cewki* w programie muszą być połączone z odpowiednimi zmiennymi. Jeżeli na schemacie wybrany jest *zestyk* lub *cewka* to przycisk paska narzędzi, opcja menu **Edycja|Edytuj symbol/tekst**, klawisz ENTER czy dwukrotne kliknięcie myszą otwierają okno wyboru zmiennej.



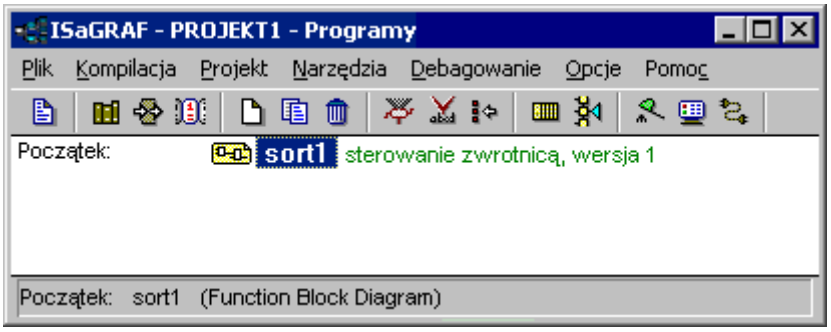
Rys. 6.15. Okno Zaznaczanie zmiennej.

Po wskazaniu w liście okna odpowiedniej zmiennej, proces łączenia jest zakończony a nazwa zmiennej pokazywana jest w edytorze nad symbolem modyfikowanego elementu.

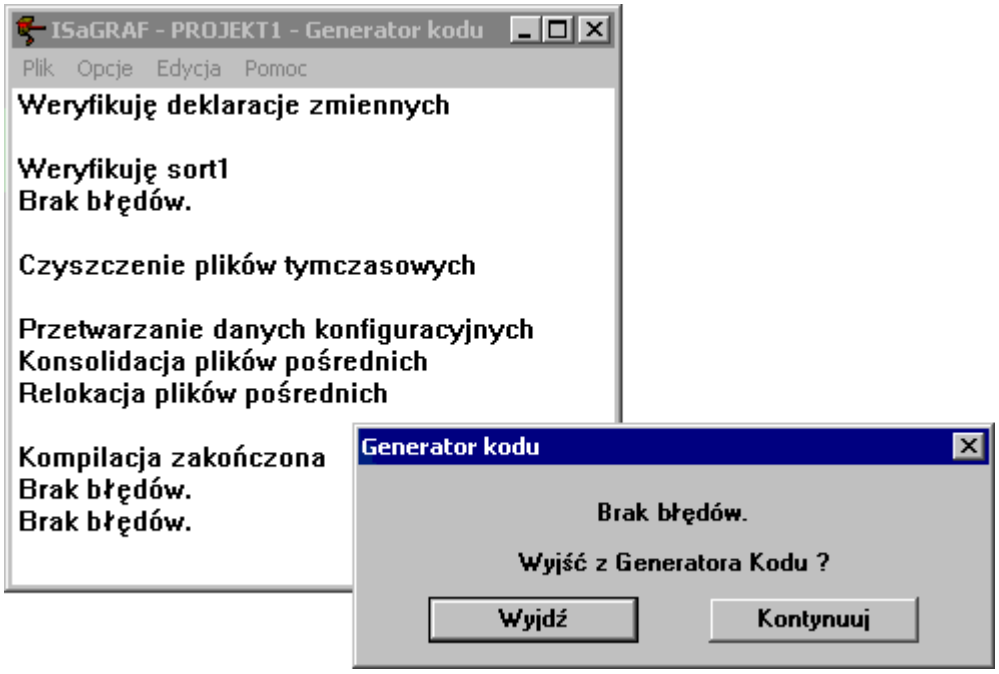
7. KOMPILACJA I SYMULACJA PROJEKTU

Działanie aplikacji można sprawdzić uruchomiając projekt na symulatorze sterownika. Przed przesłaniem aplikacji do sterownika należy wykonać kompilację. Kompilacja jest procesem tłumaczenia kodu napisanego w języku programowania na ciąg rozkazów zrozumiałych dla procesora, w tym przypadku procesora sterownika. Procesory różnią się listą rozkazów, stąd wykonując kompilację należy ustalić typ procesora sterownika, dla którego przygotowywany będzie kod. Po instalacji pakietu ISaGRAF, domyślne ustawienia opcji kompilatora wskazują symulator sterownika jako docelowy sterownik, na którym wykonywana będzie skompilowana aplikacja.

Proces kompilacji uruchamiany jest z poziomu okna *Programy* po naciśnięciu  lub po wybraniu opcji **Kompilacja|Utwórz kod aplikacji**. Polecenie to kompiluje wszystkie programy aktualnego projektu generując pełen kod aplikacji.



Rys. 7.1. Okno programy z programem „sort1”.

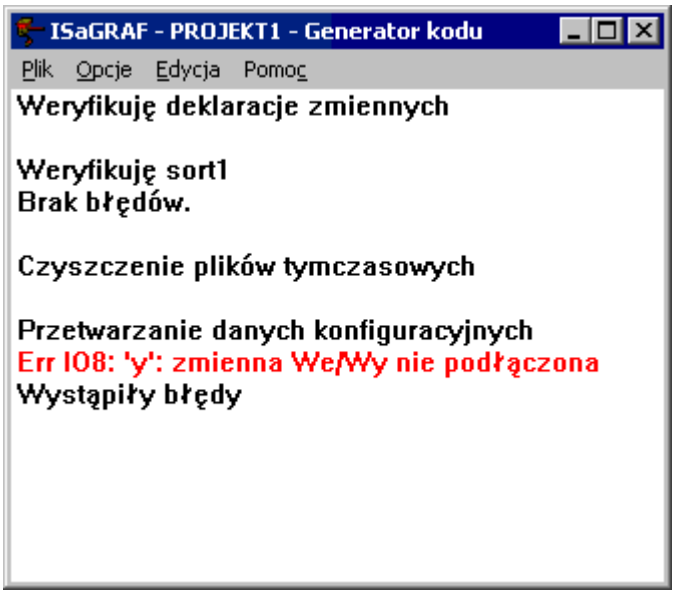


Rys. 7.2. Okno Generatora kodu uruchamiane w trakcie kompilacji.

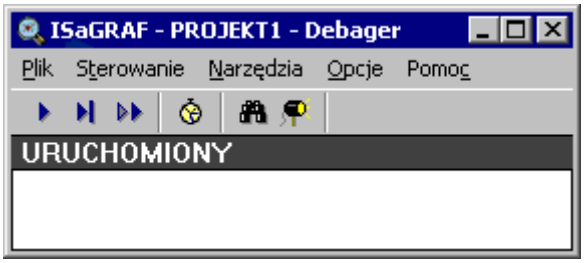
Przebieg kompilacji pokazywany jest w oknie *Generatora kodu*. Okno to w przypadku pomyślnie przeprowadzonej operacji jest zamykane na życzenie użytkownika. Jeżeli podczas kompilacji zostaną wykryte błędy (*rys. 7.3*), *Generator kodu* przerywa kompilację wyświetlając informacje o znalezionych błędach. Po poprawieniu błędów należy ponownie przeprowadzić proces kompilacji.

Po wykonaniu kompilacji można uruchomić aplikację na symulatorze sterownika. W tym celu z poziomu okna *Programy* należy wybrać przycisk lub opcję **Debagowanie|Symuluj**. Uruchomienie symulacji wprowadza okno *Programy* w tzw. tryb debagowania. Otwieranie okien *Słownika*, *Konfiguracji We/Wy* czy edytorów języków powoduje, że okna te również otwierane są w trybie debagowania. Tryb ten pozwala na monitorowanie wartości widocznych w oknach zmiennych, niemożliwa jest natomiast edycja.

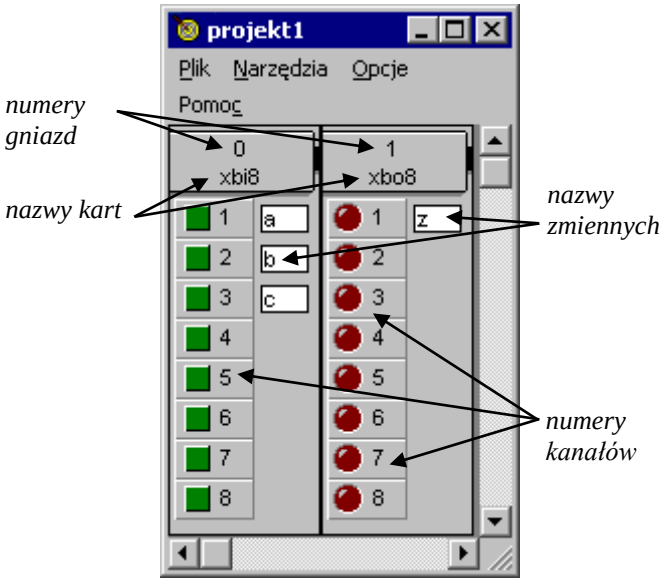
Po uruchomieniu symulacji otwierane są: okno *Debagera* i okno symulatora sterownika z widocznymi kartami We/Wy. Zamknięcie dowolnego z tych dwóch okien przerywa symulację.



Rys. 7.3. Generator kodu z błędem w postaci niepołączonej z odpowiednim kanałem zmiennej We/Wy.



Rys. 7.4. Okno Debagera



Rys. 7.5. Okno symulatora.

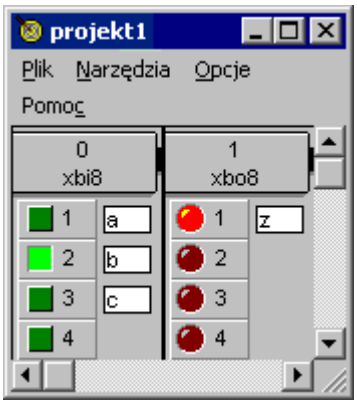
Wygląd okna symulatora zależy od ustawień jego opcji. Symulator może być wyświetlany w wersji czarno białej lub kolorowej zależnie od ustawień w menu **Opcje|Kolory**. Widoczność nazw kart We/Wy i nazw zmiennych połączonych z kanałami tych kart zależy od ustawień w menu **Opcje|Nazwy zmiennych**. Na *rys. 7.5.* symulator wyświetlany jest w wersji kolorowej z widocznymi nazwami zmiennych.

Binarne kanały wejściowe przedstawiane są w postaci zielonego, kwadratowego przycisku. Nadanie wartości *prawda* określonej zmiennej wejściowej wiąże się z wciśnięciem odpowiedniego przycisku. Wciśnięcie przycisku powoduje jego podświetlenie. Kliknięcie lewym klawiszem myszy włącza odpowiedni przycisk na stałe, tzn do następnego kliknięcia, kliknięcie prawym klawiszem włącza przycisk tylko na czas wciśnięcia klawisza myszy. *Binarne kanały wyjściowe* przedstawiane są w postaci czerwonych kół. Nadanie wartości *prawda* określonej zmiennej wyjściowej powoduje podświetlenie odpowiedniego koła.

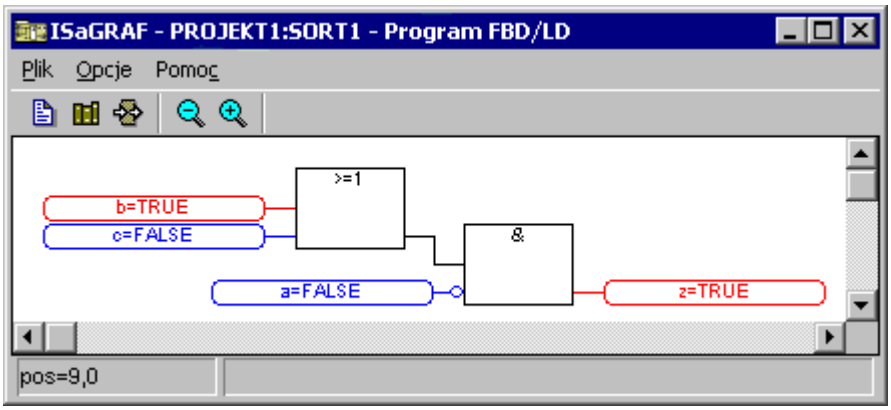
Analogowe kanały wejściowe i wyjściowe mają postać pół numerycznych. Wartości pól reprezentujących kanały wejściowe można edytować, pola związane z kanałami wyjściowym służą jedynie do wyświetlania wartości.

W trakcie pracy z symulatorem można również otworzyć okno programu – jego wygląd zależy oczywiście od języka w którym program został napisany. Na *rys. 7.6 i 7.7* pokazane zostały okno symulatora i okno programu w języku FBD wprowadzonego w punkcie 5.

Na *rys. 7.6.* pokazane zostało okno symulatora z wciśniętym drugim przyciskiem – zmienne wejściowe *a* i *c* mają więc wartość *fałsz* a zmienna *b* wartość *prawda*. Podświetlenie kanału wyjściowego skojarzonego ze zmienną *z* oznacza, że zmienna ta otrzymała w wyniku działania programu wartość *prawda*. Otwarcie okna programu (*rys. 7.7*) w czasie symulacji włącza tzw. tryb debagowania. Zmienne, którym nadana została wartość *prawda* podświetlane są na czerwono, na niebiesko podświetlane są zmienne o wartościach fałszywych.



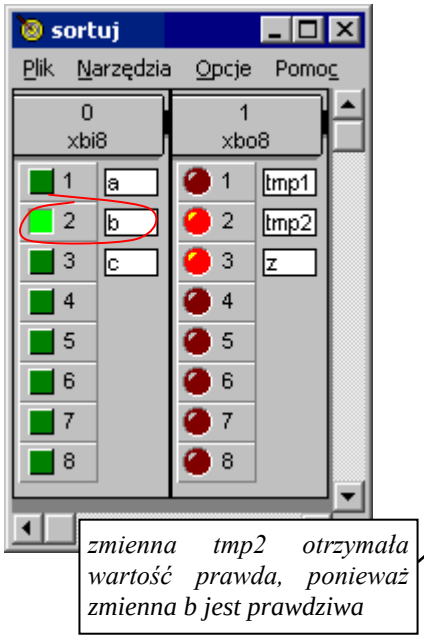
Rys. 7.6. Okno symulatora



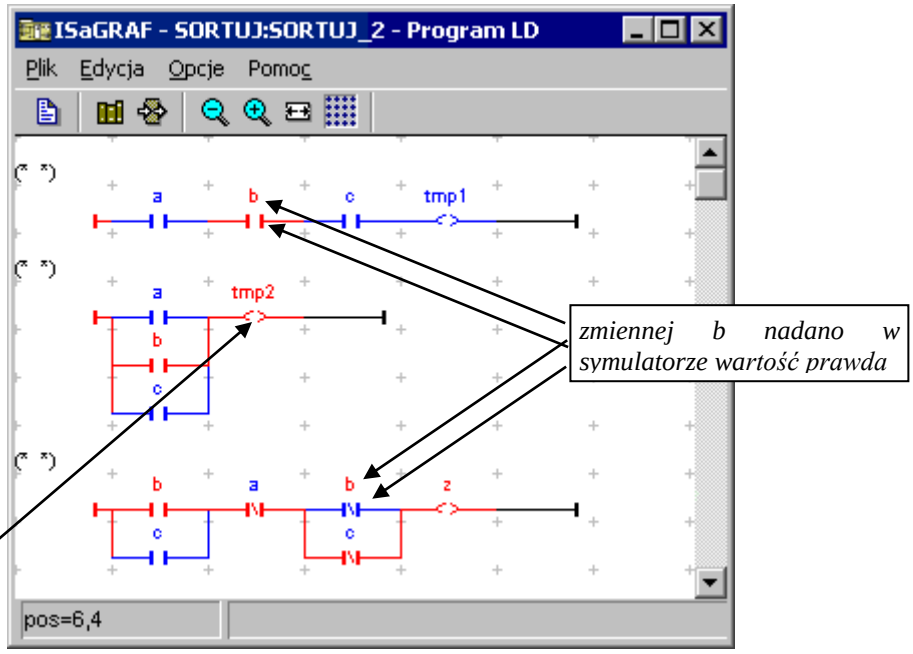
Rys. 7.7. Okno przykładowego programu

Na rys. 7.8 i 7.9 pokazane zostały okno symulatora i okno programu w języku LD wprowadzonego w punkcie 6.

Podobnie jak w przypadku programu w języku FBD, nazwy zmiennych, którym nadana została wartość *prawda* wyświetlane są na czerwono, na niebiesko wyświetlane są nazwy zmiennych o wartościach *fałszywych*. Taka sama zasada jest stosowana przy podświetlaniu symboli graficznych elementów skojarzonych ze zmiennymi.



Rys. 7.8. Okno symulatora



Rys. 7.9. Okno edytora LD z przykładowym programem.