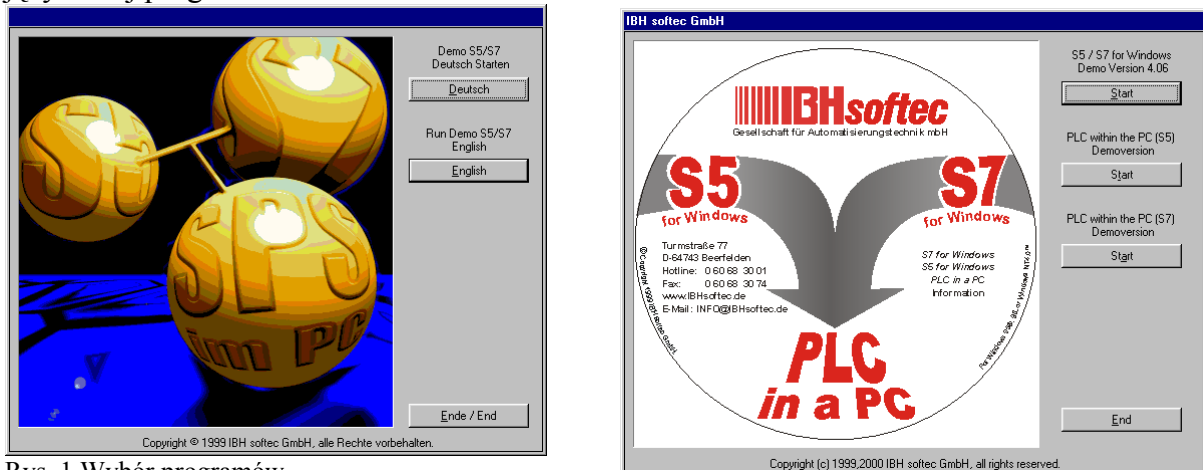


PROSTY PRZYKŁAD STEROWANIA BARIERKAMI NA PRZEJEŹDZIE KOLEJOWYM

Przykład ten został zrealizowany z wykorzystaniem sterownika z rodziny Simatic S5 z CPU 945. Program sterujący został napisany w programie S5/S7 Windows, jego praca symulowana jest z wykorzystaniem wirtualnego sterownika Real Time SPS, a wizualizacja procesu wykona została w programie WinCC firmy SIEMENS. Wszystkie z tych programów są wersjami demonstracyjnymi dostępnymi na dysku CD-ROM, który dołączony jest do książki "S5/S7 Windows Programowanie i symulacja sterowników PLC firmy SIEMENS".

Po włożeniu CD – ROM do napędu automatycznie uruchamia się okno wyboru wersji językowej programu.



Rys. 1 Wybór programów.

W naszym przypadku wybieramy angielską wersję językową co powoduje przejście do następnego okna w którym uruchamiana jest wersję demonstracyjna programu S5/S7Windows i symulator programu – PLC within the PC (S5) Demoversion. Na ekranie otworzy się okno deklaracji projektu, a w prawym dolnym rogu ekranu w pasku narzędzi pojawi się ikona sterownika programowego.

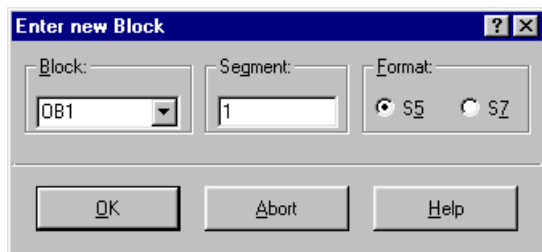
Tworzenie programu sterującego.

Poprawnie napisany program dla sterowników SIMENS'a powinien posiadać blok organizacyjny, blok programu, blok danych oraz bloki funkcyjne. Piszac program zawsze należy zwrócić uwagę na to dla jakiego sterownika jest on tworzony. Różne jednostki centralne wymagają umieszczenia instrukcji programu w różnych blokach. Informacje te dostępne są w podręcznej instrukcji dla wybranego CPU.

Założenia sterowania ruchem barierek

Jeśli nadjeżdżający pociąg uaktywni czujnik C3 (rysunek 10) zaczyna działać sygnalizacja świetlna i następuje zamykanie barierki. Po najeźdzeniu na czujnik C4 następuje otwarcie barierki. Analogiczna sytuacja występuje dla strony lewej - czujnik C1 zamyka barierkę, a C2 powoduje jej otwarcie.

Wybierając polecenie Block/New Block otwieramy okno dialogowe, w którym deklarujemy główny blok programu blok organizacyjny OB1 (rysunek 2).



Rys. 2 Deklaracja nowego bloku.

Polecenie to automatycznie uaktywni edytor, w którym definiujemy skok absolutny do bloku programu PB 10:

```
;Blok organizacyjny
    JU      PB 10
    BE
```

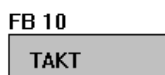
W kolejnym kroku wybierając ponownie polecenie **Block/New Block** deklarujemy blok programu PB10, który zawiera 10 segmentów z kolejnymi krokami sterowania. Segment pierwszy zawiera polecenie wywołania bloku danych DB 10:

```
;Blok programu
    C      DB 10
    ***
```

Kolejne segmenty dodajemy przez wybór polecenia **Modify/Add New Segment** (polecenie to automatycznie tworzy kolejny segment naszego programu).

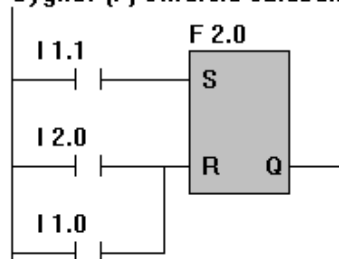
W segmencie drugim ujęta została deklaracja wywołania generatora zapisanego w postaci bloku funkcyjnego:

Wywołanie generatora



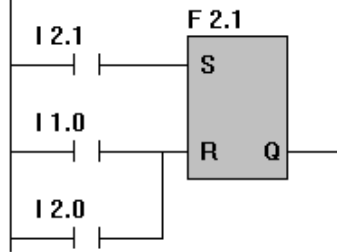
Segment trzeci realizuje warunek otwarcia barierki dla ruchu pociągu z prawej strony (rysunek 10):

Sygnal (P) otwarcia szlabanu



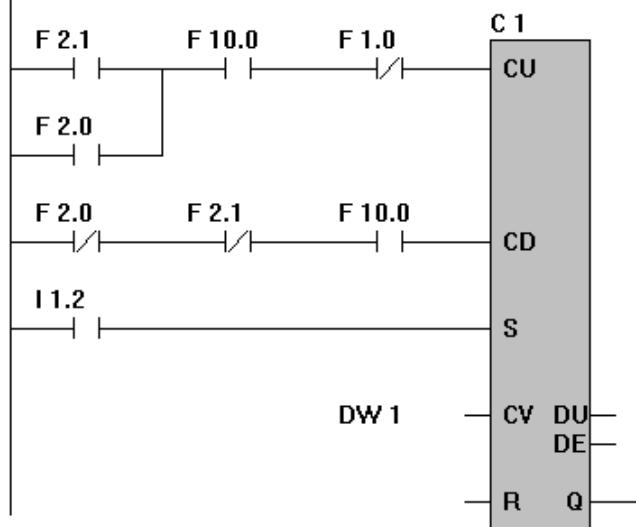
Segment czwarty analogicznie jak poprzedni realizuje sygnał otwarcia barierki dla ruchu pociągu z lewej strony:

Sygnal (L) otwarcia szlabanu



Segmenty piąty i szósty realizują ruch otwarcia barierki jeżeli warunki zawarte w segmentach trzecim i czwartym są spełnione:

Segment realizujący ruch szlabanu 1



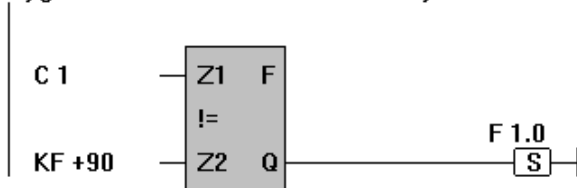
;Ruch szlabanu 2

```

L      KF +90
L      C 1
-F
T      FW 20
***
    
```

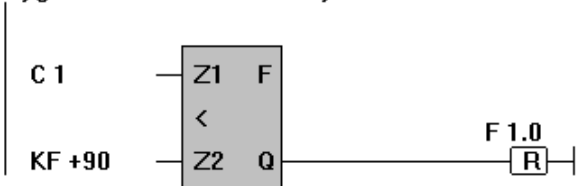
Po pełnym otwarciu barierki ustawiana jest flaga F1.0:

Sygnal - szlaban całkowicie otwarty



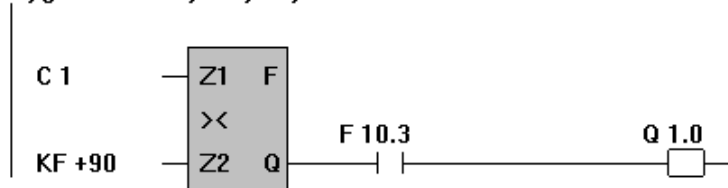
Segment ósmy zawiera informację o położeniu barierki w fazie otwierania:

Sygnal - szlaban nieotwarty

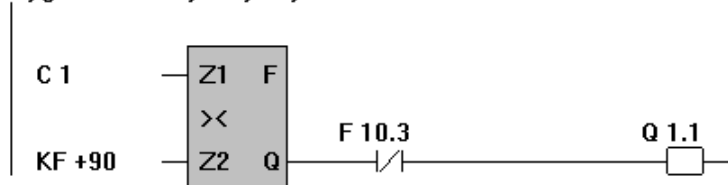


Działanie sygnalizacji świetlnej realizowane jest w segmencie dziewiątym i dziesiątym:

Sygnal świetlny aktywny dla '1'



Sygnal świetlny aktywny dla '0'



Po stworzeniu bloku programu musimy zadeklarować blok funkcyjny do którego odwołaliśmy się w bloku programu w segmencie drugim. Jest to blok realizujący funkcję pulsowania świateł sygnalizacji.

Wybierając polecenie **Block/New Block...** deklarujemy blok funkcyjny FB10. W oknie edycji bloku wpisujemy następujący kod programu:

```
;Generator sygnału impulsowego  
NAME:      TAKT  
  
      C      DB 10  
  
      A      T 10  
      CU     C 10  
      L      C 10  
      T      FY 10  
  
      A      F 10.4  
      R      C 10  
  
      AN     T 10  
      L      DW 0  
      SE     T 10  
  
      BE
```

Ostatnim blokiem który pozostał do napisania jest blok danych DB10. Podobnie jak poprzednio deklarujemy nowy blok i wpisujemy podane instrukcje:

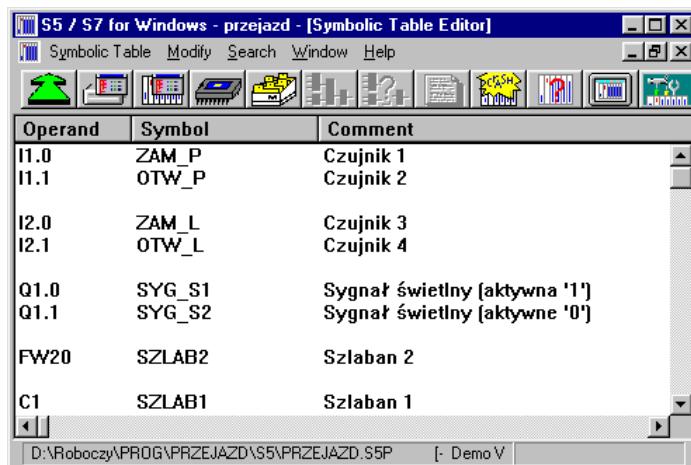
```
;Blok danych  
0:      KT 005.0  
1:      KC 000
```

Wersja demonstracyjna programu S5/S7 Windows nie posiada możliwości zapisu naszego programu sterownia, ale inny sposób zachowania projektu został w całości opisany we wspomnianej książce.

Po stworzeniu programu w tabeli symboli deklarujemy zmienne symboliczne, które w znaczny sposób ułatwiają interpretację programu. Polecenie **Window/Symbolic Table** powoduje otwarcie edytora.

Jeżeli z poziomu okna edytora symboli wybierzemy polecenie **Symbolic Table/Complete program** automatycznie wyszuka wszystkie użyte przez nas w programie zmienne i pokaże je

w edytorze, jeżeli chcemy przyporządkować im nazwy symboliczne w kolumnie Symbol wpisujemy wybrany przez nas opis. Przykład tabeli symboli dla naszego programu pokazany został na rysunku 3.

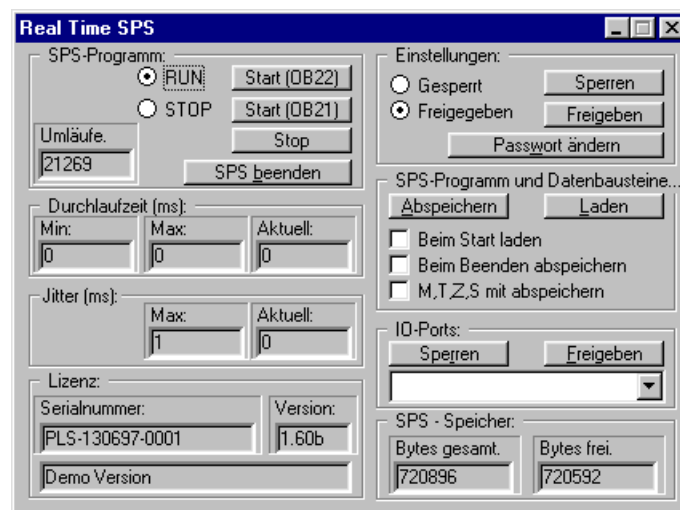


Operand	Symbol	Comment
I1.0	ZAM_P	Czujnik 1
I1.1	OTW_P	Czujnik 2
I2.0	ZAM_L	Czujnik 3
I2.1	OTW_L	Czujnik 4
Q1.0	SYG_S1	Sygnal świetlny (aktywna '1')
Q1.1	SYG_S2	Sygnal świetlny (aktywne '0')
FW20	SZLAB2	Szlaban 2
C1	SZLAB1	Szlaban 1


Rys. 3 Tabela symboli.


Symulacja działania programu.

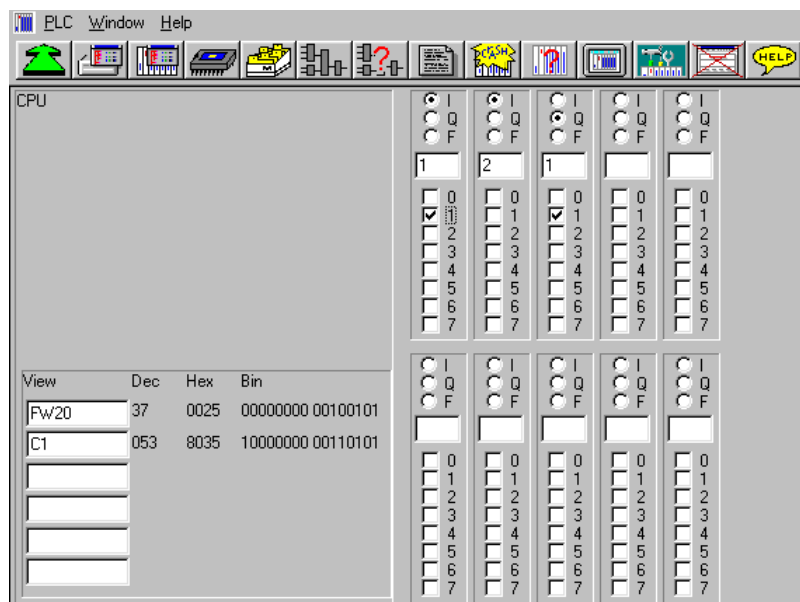
Napisany program można przetestować za pomocą specjalnego programu Real Time SPS, który jest bardzo wygodnym narzędziem pozwalającym na sprawdzenie działania programu. Program ten symuluje pracę rzeczywistego sterownika Simatic S5 z procesorem CPU 945. Program ten uruchomiliśmy już przy starcie programu narzędziowego - jego stan widoczny jest w prawym dolnym rogu paska narzędzi, ikona sterownika podświetlona na kolor żółty. Różne kolory podświetlenia tej ikony symbolizują poszczególne stany pracy symulatora: kolor żółty – brak programu w symulatorze, kolor zielony – normalna praca, kolor czerwony – stan stop.



Rys. 4 Okno dialogowe symulatora programu


Aby przesłać programu postępujemy tak jak w przypadku sterownika rzeczywistego - w oknie edycji schematu projektu podświetlamy wszystkie stworzone bloki programu i wysyłamy je do symulatora programu dzięki ikonie  lub poleceniu Block/Transfer to PLC.

Przez wybór ikony  uaktywniamy okno podglądu pracy sterownika. Symulator programu traktowany jest tak samo jak sterownik rzeczywisty. Użytkownik deklaruje tak jak na rysunku 5 wykorzystywane w programie wejścia, wyjścia, flagi oraz wybrane liczniki, elementy czasowe, słowa itd. Zmieniając stan wejść obserwujemy reakcję wyjść sterownika.



Rys. 5 Okno podglądu stanu.

Na rysunku 5 w oknie podglądu stanu zmiennych widać wartości na wejściu I1.1, wyjściu I2.1 oraz wartość licznika C1 i słowa flagi FW20.

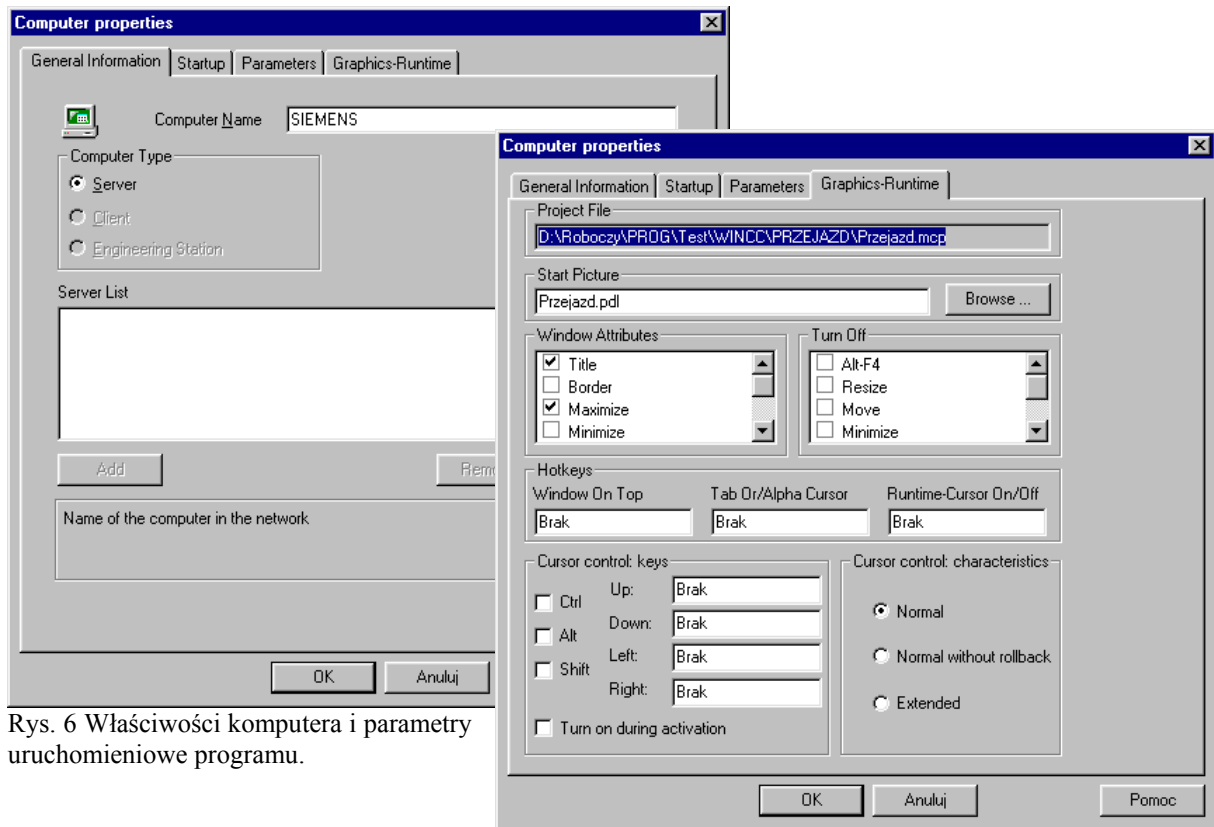
Jeżeli program zawierałby jakieś błędy to po uaktywnieniu specjalnego okna dialogowego za pomocą ikony  lub po wybraniu polecenia Window/PLC Interrupt Stack przedstawiona zostanie przyczyna zatrzymania pracy sterownika.

Wizualizacja projektu.

Gotowy program sterowania możemy zwizualizować w WinCC firmy SIEMENS. Wersja demonstracyjna tego programu posiada wszystkie możliwości wersji pełnej, a ograniczona jest jedynie do czasu pracy. Po dwóch godzinach program zamyka projekt ale wcześniej zachowuje stworzoną już aplikację. Ponowne uruchomienie programu pozwala na dalszą jego pracę.

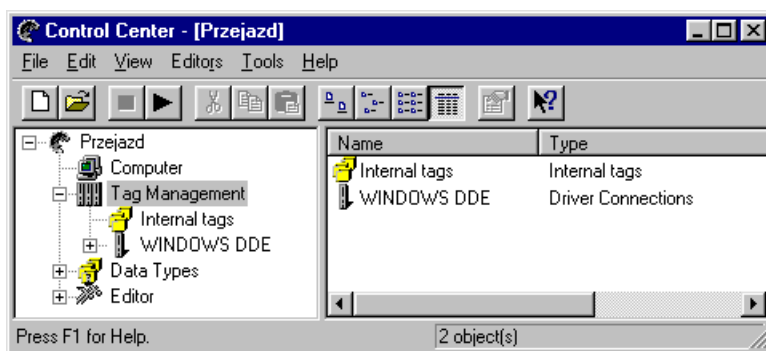
Tworzenie aplikacji w WinCC.

Po instalacji programu (dołączonego do książki) uruchamiamy go poleceniem Simatic/WinCC/Windows Control Center. Samoczynnie stworzony zostanie wirtualny serwer, który zawiera nazwę komputera oraz jego właściwości. Podglądając właściwości tego serwera zauważamy, że określona została nazwa naszego komputera. W następnym oknie deklarujemy właściwości startowe dla stworzonego projektu: określamy grafikę startową oraz wygląd ekranu startowego. Grafikę startową określamy dopiero w momencie kiedy stworzymy ją w edytorze graficznym.



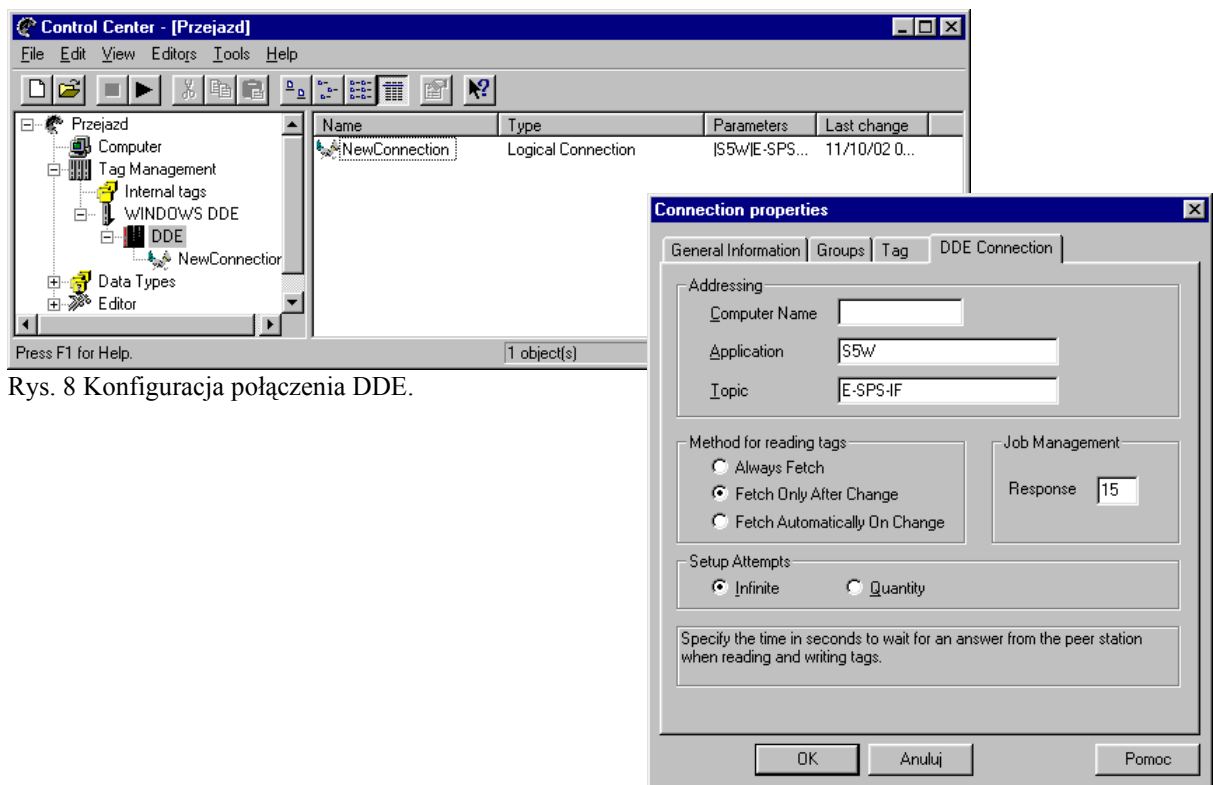
Rys. 6 Właściwości komputera i parametry uruchomieniowe programu.

Następnym krokiem który musimy wykonać jest konfiguracja drajwera komunikacyjnego. Podświetlamy polecenie Tag Management i klikając na nim prawym klawiszem myszy wybieramy polecenie Add New Driver.... Ze względu na to, że nie posiadamy w tej chwili rzeczywistego sterownika nasze dane będziemy wymieniać tylko przez łącze DDE, dlatego wybieramy drajwer Windows DDE.chn.



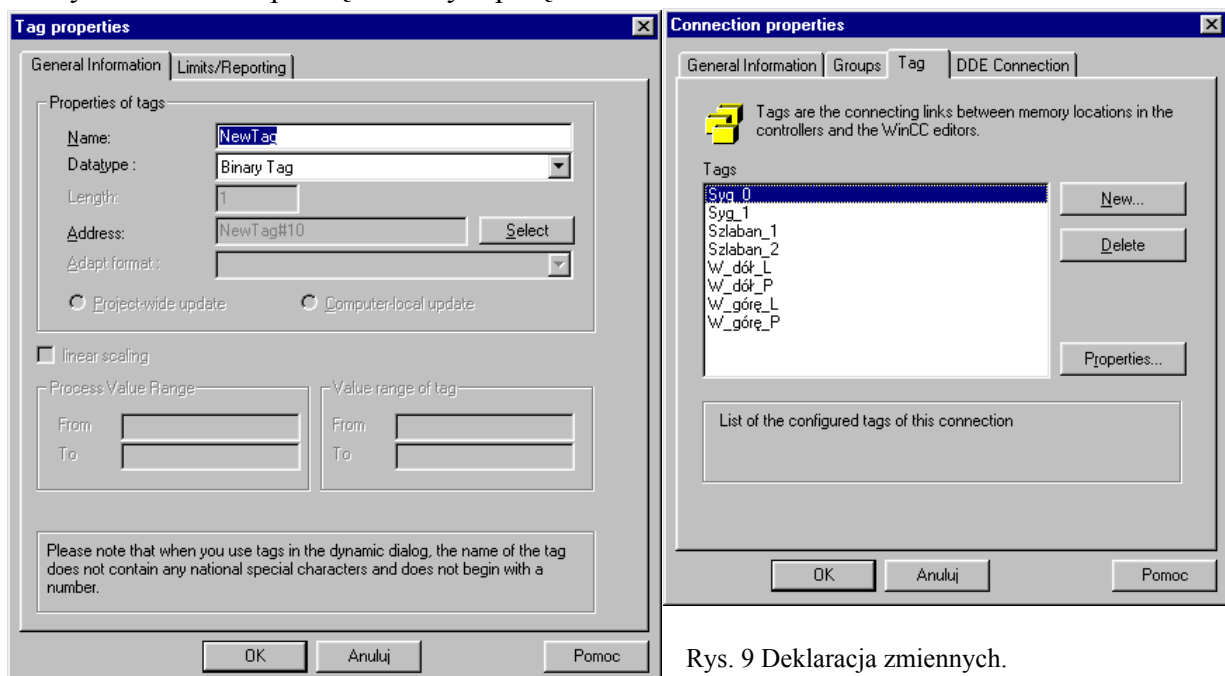
Rys. 7 Deklaracja właściwości komunikacyjnych.

Definicja właściwości DDE jest niezbędna aby wskazać program, z którym mają być wymieniane dane. Rozwijając drzewo powiązań w lewym oknie programu podświetlamy DDE i klikając prawym klawiszem myszy wybieramy polecenie New Driver Connecton.... W otwartym oknie deklarujemy unikalną nazwę połączenia, aplikację z którą mają być wymieniane dane oraz sposób połączenia tak jak zostało pokazane to na rysunku 8.



Rys. 8 Konfiguracja połączenia DDE.

Następną bardzo ważną czynnością jest konfiguracja zmiennych w programie. Dla naszego projektu będziemy wykorzystywali w deklaracji zmienne symboliczne zadeklarowane w programie narzędziowym. W oknie Connection properties w zakładce Tag deklarujemy wszystkie zmienne powiązane z tym połączeniem.

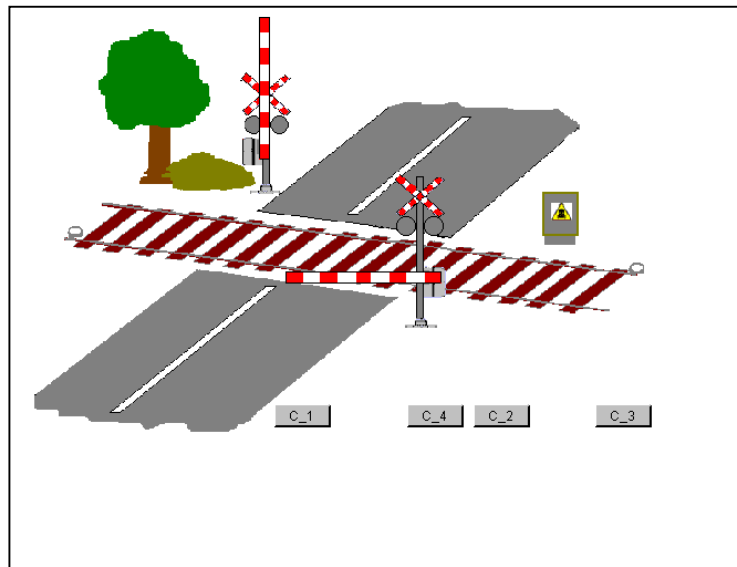


Rys. 9 Deklaracja zmiennych.

Każdej zmiennej nadajemy nazwę, deklarujemy typ danych oraz adres. Należy pamiętać, że adres deklarujemy z myślnikiem z przodu, bo tylko wtedy tak zadeklarowana zmienna będzie aktywna. Jeżeli mamy zadeklarowany rodzaj połączenia oraz zmienne komunikacyjne pozostaje nam stworzenie grafiki. Wybierając polecenie Editor/Graphics Designer/New

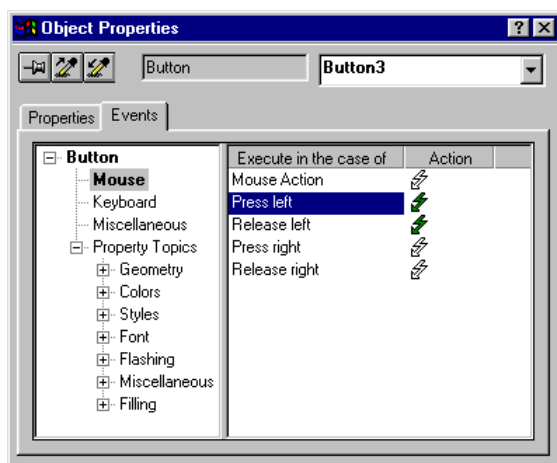
picture zostanie stworzony plik z rozszerzeniem *.pdl. Dwukrotne kliknięcie na jego nazwę powoduje otwarcie edytora grafiki, w którym można stworzyć własny obraz synoptyczny procesu.

Każdy z obiektów dynamicznych należy powiązać z akcją jaka powinna nastąpić po kliknięciu myszą, naciśnięciu klawisza. Podkład rysunku może stanowić obraz mapy bitowej, ale obiekty dynamiczne muszą być zdefiniowane w edytorze graficznym WinCC. Wybierając polecenie Smart Objects/Graphic Object w palecie obiektów wyznaczamy rozmiar dla obrazu. Automatycznie otwarte zostaje również okienko, w którym podajemy ścieżkę dostępu dla naszego rysunku. Rysujemy pozostałe elementy dynamiczne barierek, lampy sygnalizatora oraz przyciski które są odpowiednikiem zamontowanych czujników powodujące zadziałanie układu.

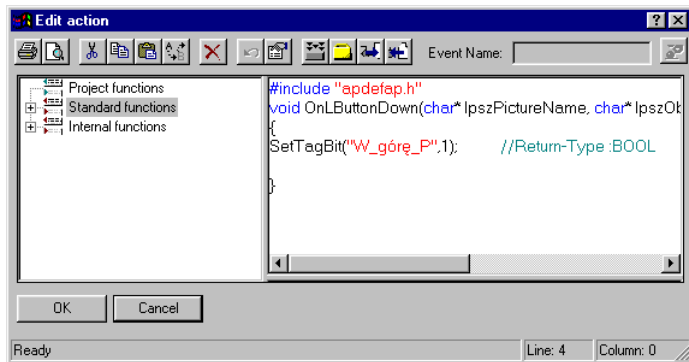


Rys. 10 Rysunek przejazdu kolejowego – mapa bitowa z obiektami dynamicznymi.

Przyciski pobieramy z biblioteki Windows Objects/Button umieszczamy na rysunku i w otwartym oknie dialogowym zmieniamy ich nazwy. Przez podwójne kliknięcie na wybranym elemencie otwieramy okno deklaracji właściwości. Przyciski chcemy aktywować za pomocą kliknięcia myszą dlatego przechodzimy do zakładki Events gdzie deklarujemy właściwości akcji.

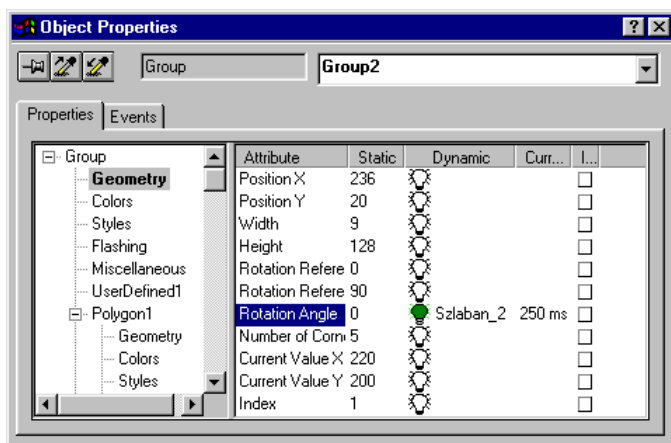


Rys. 11 Deklaracja akcji przycisku.



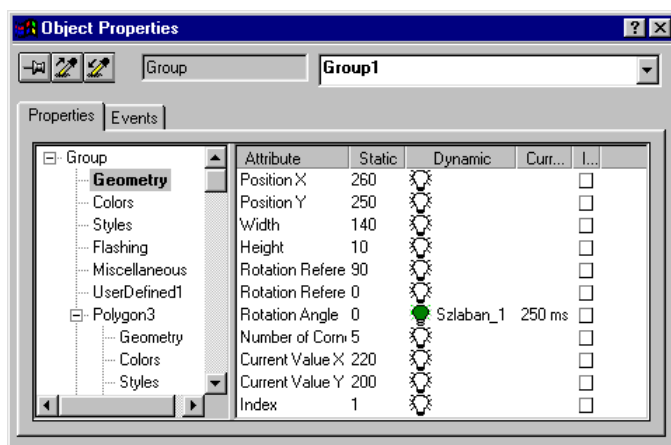
Rys. 12 Deklaracja właściwości dynamicznych akcji przycisku.

W oknie edycji akcji wpisujemy zgodnie z rysunkiem 12 powiązanie ze zdefiniowanymi zmiennymi w edytorze. Zgodnie z tym opisem programujemy kolejne przyciski akcji. Następnie rysujemy barierki przejazdu w takiej pozycji jak na rysunku 10 i edytujemy właściwości akcji. Zapisujemy w **Rotation Reference Y** pozycję 90, a w **Rotation Angle** klikając na deklarację dynamiczną wybieramy powiązanie ze zmienną Szlaban_2.




Rys. 13 Właściwości dynamiczne dla szlabanu 2.

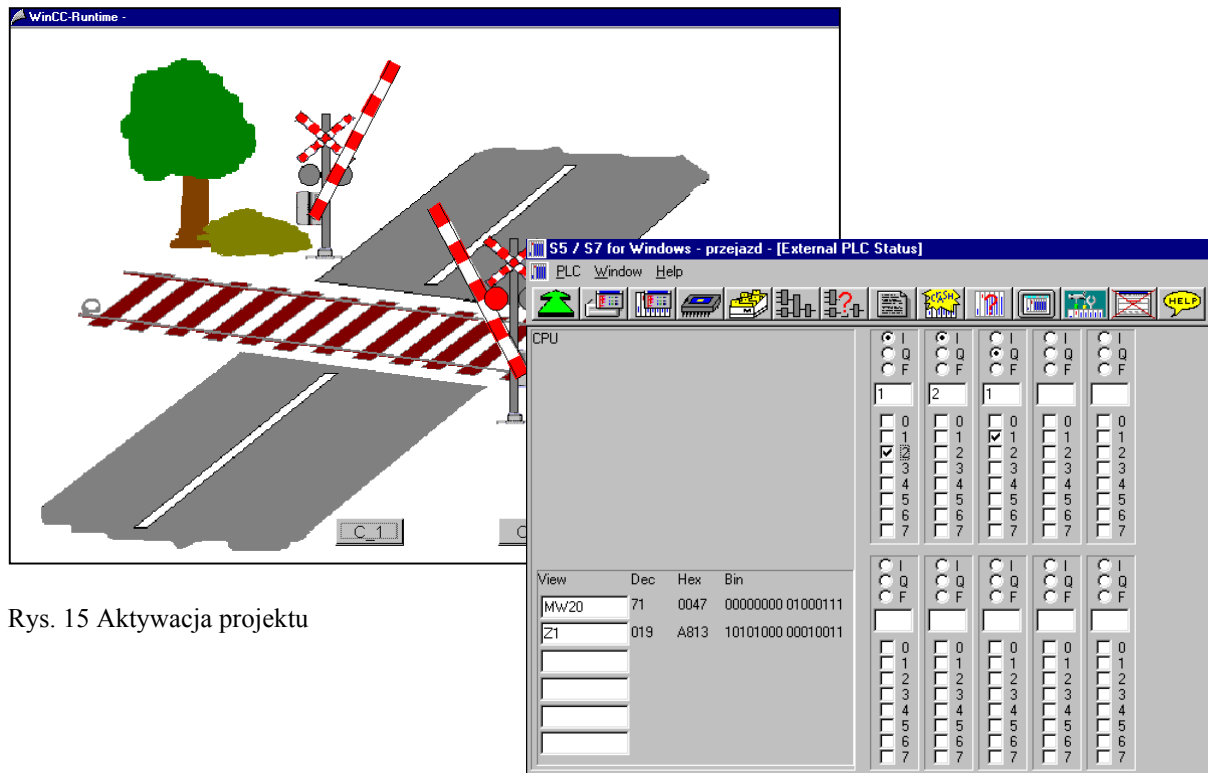
Dla drugiego szlabanu deklarujemy właściwości zgodnie z rysunkiem 14.



Rys. 14 Właściwości dynamiczne dla szlabanu 1.

Mając napisany program sterujący oraz zrobiony obraz synoptyczny w programie wizualizacyjnym możemy w całości zobaczyć działanie naszego projektu. W programie S5/S7 Windows podświetlamy wszystkie boki i wysyłamy je do sterownika, uruchamiamy okno podglądu stanu. Następnie uruchamiamy WinCC i jeżeli we właściwościach zdefiniowaliśmy obraz startowy to będzie się on nam uruchamiał po naciśnięciu klawisza aktywacji  lub po wyborze polecenia File/Activate.

Przez uaktywnianie wejść w oknie podglądu stanu S5/S7 Windows możemy również zaobserwować pracę elementów zdefiniowanych w programie wizualizacji.



Rys. 15 Aktywacja projektu

Tworząc połączenie pomiędzy WinCC i S5/S7 Windows konieczne należy pamiętać o sekwencji załączania i wyłączania programów. Podczas uruchamiania zawsze najpierw uruchamiamy S5/S7 Windows, następnie Real Time SPS i przekazujemy do niego program, dopiero na końcu WinCC. Odwrotnie postępujemy wyłączając programy: najpierw deaktywujemy projekt, zamykamy projekt w WinCC, następnie wyłączamy okno podglądu stanu, później symulator Real Time SPS, a dopiero na końcu zamykamy S5/S7 Windows.

Joanna Moczko-Król

j_mk@poczta.onet.pl

Artur Król

akrol0@poczta.onet.pl