

KONFIGURACJA GRAFICZNA ROZPROSZONEGO SYSTEMU AUTOMATYKI

Z. Świder, L. Trybus, M. Śnieżek
Politechnika Rzeszowska
Katedra Automatyki i Informatyki

Streszczenie.

W artykule przedstawiono wyniki projektu, którego celem było opracowanie metody i pakietu oprogramowania dla konfiguracji graficznej rozproszonego systemu automatyki, składającego się z konfigurowalnych urządzeń obiektowych (regulatory i sterowniki wielofunkcyjne, stacje pomiarowe, sterowniki PLC) oraz z komputera nadrzędnego (stacja operatorska). Założono, że oprogramowanie użytkowe urządzeń i stacji jest oparte na blokach funkcyjnych realizujących cząstkowe algorytmy przetwarzania pomiarów, sterowania, komunikacji, wizualizacji, alarmowania i raportowania. Dotyczy to zdecydowanej większości systemów rozproszonych stosowanych obecnie. Konfiguracja graficzna sprowadza się do wygenerowania na ekranie komputera schematów blokowych strategii sterowania czy wizualizacji, na podstawie których automatycznie są tworzone liczbowe dane konfiguracyjne urządzeń i całego systemu. Schematy blokowe są bowiem najbardziej czytelną formą reprezentacji układów automatyki (taką metodę konfiguracji zastosowano między innymi w pakietach SIPROM-Siemens, LoopDraw-Turnbull czy MYCAD-Moore).

1. ELEMENTY SYSTEMÓW AUTOMATYKI

Typowy rozproszony system automatyki składa się z konfigurowalnych urządzeń obiektowych, takich jak regulatory i sterowniki wielofunkcyjne, dedykowane regulatory PID, stacje pomiarowe i sterowniki PLC, oraz z komputera nadrzędnego (stacji operatorskiej) z oprogramowaniem sterowania nadrzędnego i przetwarzania danych SCADA (*supervisory control and data acquisition*) [5].

Klasyfikacja regulatorów. Regulator lub sterownik cyfrowy jest urządzeniem, w którym mikrokomputer na podstawie programu zapisanego w pamięci steruje procesem technologicznym, instalacją czy maszyną. Zależnie od funkcji realizowanych przez program oraz rodzaju wejść/wyjść wyróżniamy: (1) regulatory aparaturowe PID, (2) przetworniki inteligentne z algorytmem PID, (3) sterowniki logiczne PLC, (4) aparaturowe sterowniki wielofunkcyjne, (5) modułowe regulatory kasetowe.

Regulatory PID to urządzenia aparaturowe obsługujące 1 lub 2 układy regulacji. W grupie tej konfiguracja sprowadza się do wyboru wariantu algorytmu PID oraz ustawienia opcji jego wykonania (poprzez przyporządkowanie odpowiednich wartości zmiennym konfiguracyjnym, pełniącym rolę flag sterujących przebiegiem programu). Podobnie w grupie przetworników inteligentnych konfiguracja jest stała. *Wielkości zadane i parametry są ustawiane za pomocą ręcznych programatorów (konfiguratorów)*, i z reguły podczas dalszej pracy nie są one zmieniane. Sterowniki logiczne PLC realizują dwustanowe sterowanie logiczne i sekwencyjne. *Programowane są za pomocą ręcznych programatorów lub komputerów przenośnych lap-top.* Program zwykle ma postać ciągu prostych instrukcji, schematu przekąznikowego lub sieci działań.

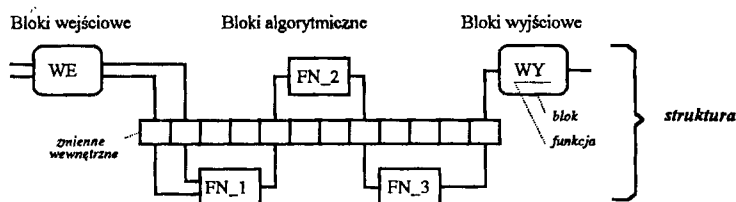
Aparaturowe sterowniki wielofunkcyjne mogą pracować jednocześnie jako regulatory PID i sterowniki logiczne. Ich oprogramowanie składa się z kilkudziesięciu bloków funkcyjnych (odpowiadających konwencjonalnym urządzeniom i elementom automatyki) łączonych stosownie do przeznacze-

nia układu. *Konfiguracja polega na zestawianiu układu sterowania z takich bloków. Przypomina to montaż układów tradycyjnych.* Modułowe regulatory kasetowe zazwyczaj obsługują kilkanaście układów regulacji PID oraz kilkadziesiąt obwodów sterowania logicznego. *Do konfiguracji używane są specjalizowane pakiety programowe (edytory graficzne) pozwalające na konfigurację zarówno stacji procesowej (regulatora) jak i stacji operatorskiej.* Program zwykle obejmuje konfigurację graficzną w różnych językach programowania (z bloków funkcyjnych, listy instrukcji, schematów drabinkowych), oraz graficzne dokumentowanie wszystkich struktur (łącznie z komunikacją).

Jak więc widać z powyższego przeglądu, w zależności od producenta, złożoności i przeznaczenia regulatora, czy też przyzwyczajień projektanta systemu, można spotkać różne metody i sposoby konfiguracji. Choć większość urządzeń posiada możliwość konfiguracji z ręcznych konfiguratorów lub z panelu czołowego urządzenia, to jednak ten sposób konfiguracji jest mocno uciążliwy przy dużej ilości bloków funkcyjnych. Wydaje się więc, że jedyną alternatywą jest konfiguracja graficzna zarówno poszczególnych elementów (sterowników i regulatorów) jak i całego systemu automatyki. Jej zaletą jest przejrzystość, jednolitość narzędzi i czytelna dokumentacja.

Konfiguracja z bloków funkcyjnych. W zdecydowanej większości systemów rozproszonych oprogramowanie użytkowe poszczególnych urządzeń i stacji operatorskich oparte jest na blokach funkcyjnych realizujących cząstkowe algorytmy przetwarzania pomiarów, sterowania, komunikacji, wizualizacji, alarmowania i raportowania.

Przyjmijmy, że w pamięci regulatora określona jest tablica jednowymiarowa na dane i wyniki obliczeń [5]. Jej elementy są nazywane *zmiennymi wewnętrznymi*. Przyjmuje się, że program jest podzielony na fragmenty (moduły odpowiadające poszczególnym elementom automatyki klasycznej, jak blok PID, przekładniki itp.), a wyniki obliczeń jednego fragmentu stanowią dane dla innych. Można więc mówić o wejściach i wyjściach poszczególnych bloków (elementów) schematu.



Rys.1. Idea konfiguracji regulatora z bloków funkcyjnych.

Ze względu na przeznaczenie bloki funkcyjne dzielimy na:

- **bloki wejściowe** - obsługują analogowe i binarne wejścia obiektowe, klawisze panelu operatorskiego oraz odbiorniki komunikacyjne; ich wyjścia są zmiennymi wewnętrznymi stanowiącymi źródła danych dla innych bloków; ich liczba i rodzaj zależy od sprzętu (*hardware*),
- **bloki algorytmiczne** - przetwarzają zmienne wewnętrzne zgodnie z przyporządkowanymi funkcjami (algorytmami), wyniki zapamiętują w zmiennych wewnętrznych - najczęściej spotykane są funkcje *arytmetyczne* (ADD, SUB, MUL, ...), *logiczne* (AND, OR, JKFF, ...), *czasowe* (FILTR, DEL, DIFR, SP, ...), *regulatory* (PID, POS, SERV), *programatory* (SEQ, SEQC) i inne,
- **bloki wyjściowe** - obsługują analogowe i binarne wyjścia obiektowe, elementy wizualne panelu (wskazniki, linijki), oraz nadajniki komunikacyjne, czerpiąc dane ze zmiennych wewnętrznych.

Jak widać z powyższego podziału, bloki są podstawowymi elementami ("cegółkami"), z których budujemy schemat (strukturę) układu regulacji. Proste struktury w małych regulatorach wymagają kilkanaście do kilkadziesiąt bloków, w dużych do kilkuset. Ręczna konfiguracja z panelu operatorskiego lub małego konfiguratora jest bardzo kłopotliwa i praktycznie pozwala jedynie na drobne modyfikacje wcześniej wprowadzonej struktury. Dla konfiguracji całości niezbędne jest wykorzystanie komputera klasy PC z odpowiednim oprogramowaniem (np. dla konfiguracji graficznej).

Fazy konfiguracji. Konfiguracja regulatora wielofunkcyjnego przypomina montaż tradycyjnego układu automatyki. Założmy, że schemat projektowanego układu został już sporządzony. Konfiguracja regulatora składa się z następujących faz:

- **definiowanie** - wybór bloków do układu i przyporządkowanie im funkcji (o ile nie są ustalone) - polega ono na ustawieniu właściwych par *blok* $\leftrightarrow$ *funkcja*, gdzie początkowo bloki algorytmiczne (proste i złożone) nie mają przyporządkowanej żadnej funkcji, natomiast bloki wejściowe i wyjściowe mają zdefiniowaną funkcję (ustaloną lub jedną z alternatywnych) - *tylko bloki zdefiniowane, którym przyporządkowano konkretne funkcje, uczestniczą w następnych fazach*,
- **łączenie** - łączenie wejść i wyjść zdefiniowanych wcześniej bloków zgodnie ze schematem - łączone są wejścia i wyjścia tego samego rodzaju (typu) i połączenie jest zawsze wykonywane od *wejścia do wyjścia*, tzn. nie dopuszcza się "zwierania" wyjść bloków,
- **pozycjonowanie** - określenie kolejności (sekwencji) obsługi bloków w ramach schematu - pozycjonowanie jest fazą opcjonalną (w większości regulatorów numer bloku określa zarazem kolejność obsługi, jest to jednak niewygodne przy wszelkich modyfikacjach struktury),
- **parametryzacja** - ustalenie wartości parametrów wewnętrznych dla poszczególnych bloków - obejmuje ona zarówno parametryzację *off-line* (dla parametrów, których wartość może być ustawiona jedynie podczas konfiguracji a następnie pozostaje stała w fazie obsługi procesu), jak i *on-line* (dla parametrów, których wartość jest wstępnie ustawiana przy konfiguracji, a następnie może być zmieniana na bieżąco podczas obsługi procesu).

2. KONFIGURACJA STRUKTURY STEROWANIA

Dla konfiguracji urządzeń rozproszonego systemu automatyki założono, że przygotowywana struktura (strategia) przy użyciu edytora graficznego nie jest bezpośrednio zamieniana na postać docelową (dane konfiguracyjne danego urządzenia), ale tłumaczona do pewnego języka pośredniego. Język ten jest jednocześnie językiem źródłowym dla opisu konfiguracji w trybie tekstowym. Dla uzyskania danych konfiguracyjnych należy przetłumaczyć przygotowaną strategię (zapisaną w postaci tekstu) używając odpowiedniego kompilatora. Kompilator ten, będąc ostatnim ogniwem pomiędzy człowiekiem a urządzeniem, sprawdza poprawność przygotowanej struktury (sygnalizując ewentualne błędy) i tworzy program (plik) wynikowy w postaci akceptowalnej przez docelowy element systemu automatyki (dane konfiguracyjne urządzenia).

Elementy języka CONF_DL. Opracowany język CONF_DL (*CONF*iguration *DL*escription *Language*) służy do opisu struktury wewnętrznej (strategii) urządzenia w formie tekstowej. Elementy tego języka umożliwiają jednoznaczne i pełne określenie tworzonej strategii, w tym: przypisanie blokom realizowanych funkcji, dokonanie połączeń pomiędzy blokami, określenie kolejności przetwarzania, ustalenie parametrów dla funkcji. Przedstawione w dalszej części opisu nazwy bloków, funkcji i parametrów są zgodne z nazwami stosowanymi przy konfiguracji sterownika PSW-8.

Konfigurację danego urządzenia opisuje program, który rozpoczyna dyrektywa `#program`, a kończy `#end_program`. Dyrektywa `#include "nazwa pliku"` powoduje włączenie zawartości wskazanego pliku (np. biblioteki makrobloków) do programu źródłowego. Blok nagłówka, umieszczony pomiędzy dyrektywami `#header` i `#end_header`, zawiera pewne dodatkowe informacje o przygotowywanej strukturze, m.in. nazwę konfigurowanego urządzenia czy numer struktury.

Dla większej elastyczności, dopuszcza się hierarchiczność budowy strategii, tzn. moduł *main* może zawierać dowolną liczbę modułów pomocniczych (makrobloków), wyznaczanych dyrektywami `#module` i `#end_module`. Pozwala to użytkownikowi definiować pewne *makrobloki* realizujące bardziej złożone funkcje. Taki moduł może odpowiadać jednemu schematowi przygotowywanemu w edytorze graficznym. Z nich następnie składa się docelowy schemat struktury sterowania.

Ze względu na blokową strukturę konfigurowanych urządzeń przyjęto założenie, że linia programu musi zawierać komplet informacji o bloku. Budowa linii jest więc następująca:

blok [pozycja] = funkcja ("wejście"=wyjście, "wejście"=wyjście, ...) {on-par} <off-par> ;

gdzie:

blok - jest nazwą bloku np. *di2*, *trdv*, itd.; jeśli strategia może zawierać większą liczbę bloków danego typu, np. *b__*, *c__*, wówczas po nazwie występuje numer tego bloku: bezwzględny (np. *b01*) lub względny (wewnątrz modułu, np. *b#01*),

pozycja - określa kolejność obsługi (pozycjonowania) bloku w tworzonej strukturze; dotyczy ona tylko bloków *b__* i *c__*, a dla pozostałych bloków pole to jest puste (nie występuje) - dopuszcza się pozycję bezwzględną (np. *p01*) i względną (np. *p#01*) wewnątrz modułu,

funkcja - podaje nazwę funkcji realizowanej przez blok (np. dla bloku wyjść binarnych *bo* możliwe są funkcje *bo*, *boc*, *bot*) lub nazwę pliku w strukturze hierarchicznej zawierającego opis makrobloku (poprzedzonego znakiem \$),

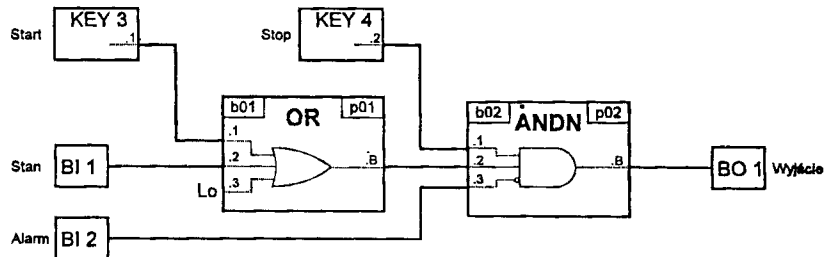
(...) - w nawiasach podaje się połączenia wejść danego bloku z wyjściami innych bloków wg zasady: "wejście"=wyjście, gdzie *wejście* jest nazwą wejścia bloku, a *wyjście* jest nazwą bloku i nazwą wyjścia, z którym dane wejście jest połączone, np. *.1"=b02.a*, *.c"=bi_1*,

{ ... } - nawiasy te wyznaczają pole opisu parametrów *on-line* rozdzielonych przecinkami wg formatu *nazwa_parametru=wartość* (np. *kp=1.0*, *Ti=30.0*),

< ... > - nawiasy te wyznaczają pole opisu parametrów *off-line* rozdzielonych przecinkami wg formatu *nazwa_parametru=wartość* (np. *d=-,-, No=3*),

; - znak średnika kończy linię opisującą dany blok.

Poniżej zamieszczony przykład pokazuje jak stosować przedstawione konstrukcje języka do opisu prostej strategii.



Rys.2. Układ Start-Stop (włączanie/wyłączanie silnika).

Układ Start-Stop. Niech dana będzie strategia, opisana w [5] (str. 153), a służąca do uruchamiania silnika napędowego pewnego urządzenia technologicznego. Do dyspozycji są klawisze *Start* i *Stop* w panelu operatorskim. Wejścia binarne *Stan* i *Alarm* sygnalizują stan sterowanego urządzenia, a wyjście binarne steruje urządzeniem. Odpowiedni program może mieć postać:

```
#program    Układ START-STOP

#header
    devc=psw-8s                // urządzenie -> PSW-8 wersja standard
    name=1                     // to pole określa numer strategii
#end_header

#main      przykład 1 - wersja 1
    b01 [1] = or (".1"=ke3.1, ".2"=bi_1, ".3"=lo);           // 1
    b02 [2] = andn (".1"=ke4.2, ".2"=b01.b, ".3"=bi_2);     // 2
    bo_1 = bo (b02.b);                                       // 3
#end_main

#end_program
```

Kompilator struktur CT_COMP. Zadaniem kompilatora CT_COMP (*Configuration Text COMpiler*) jest tłumaczenie strategii opisanych w języku CONF_DL do postaci zrozumiałej przez konfigurowane urządzenie (dla którego dana struktura jest tworzona). Strategia docelowa dla większości obecnie stosowanych urządzeń posiada format binarny, z zakodowanymi numerami funkcji dla poszczególnych bloków, wzajemnymi połączeniami, kolejnością obliczeń, wartościami parametrów i innymi danymi. Struktura tak opisana jest niezrozumiała i nieczytelna dla użytkownika. Dlatego w sterowniku PSW-8 do zapisu strategii docelowej stosuje się oprócz binarnych danych konfiguracyjnych równoległe opis tekstowy wzorowany na tabelach konfiguracyjnych stosowanych np. w regulatorach Siemens'a czy Foxboro. Opis ten zawiera wszystkie teksty, jakie pojawiają się na wskaźnikach panelu w kolejnych fazach konfiguracji sterownika. Dzięki temu użytkownik może łatwo dokonywać drobnych poprawek czy zmian w strukturze, a następnie przesłać ją do sterownika. Opracowany kompilator CT_CONF również tworzy strategię docelową (dane konfiguracyjne regulatora) zarówno w formacie binarnym jak i tekstowym.

Wywołanie kompilatora ma postać:

```
>ct_comp plik_źródłowy[.cnf] [plik_wynikowy] [/opcje_kompilacji] ,
```

Kompilator wczytuje program źródłowy strategii i zapisuje dane konfiguracyjne do wskazanego pliku. Gdy nie podano nazwy pliku wynikowego program rozpoznaje typ urządzenia, i zależnie od tego tworzy strategię docelową, zapisując ją w zbiorze o nazwie przyjętej dla danego urządzenia, np. dla PSW-8 są to: *psw_conf.mnn* dla formatu binarnego i *psw_list.mnn* dla tabel konfiguracyjnych, gdzie *mnn* jest numerem (nazwą) struktury.

Poniżej przedstawiono tablice konfiguracyjne wytworzone przez kompilator CT_COMP dla powyższego przykładu (por. Tabl.5.4 [5], str154).

NAME			CONN	
DEFN	001		b01.1	ke3.1
			b01.2	bi 1
	b01	or	b02.1	ke4.2
POSN	b02	andn	b02.2	b01.b
			b02.3	bi 2
	p 1	b01	bo 1	b02.b
	p 2	b02	RUN	

3. WYKORZYSTANIE NARZĘDZI CAD

Podczas konfigurowania regulatora lub sterownika wielofunkcyjnego w sposób tradycyjny, tzn. za pomocą panelu operatorskiego, pulpitu technicznego, ręcznego konfiguratora itp., jednorazowo są przedstawiane dane jednego lub co najwyżej kilku bloków układu. Z tego względu w ostatnich latach zaczęły się pojawiać specjalizowane pakiety CAD (*computer aided design*) przeznaczone do tworzenia na monitorze komputera PC lub stacji operatorskiej schematów blokowych układów sterowania, na podstawie których automatycznie generuje się dane konfiguracyjne. Niektóre firmy wykorzystują również ogólnie znane pakiety jak OrCAD czy AutoCAD dla konfiguracji graficznej swoich urządzeń automatyki.

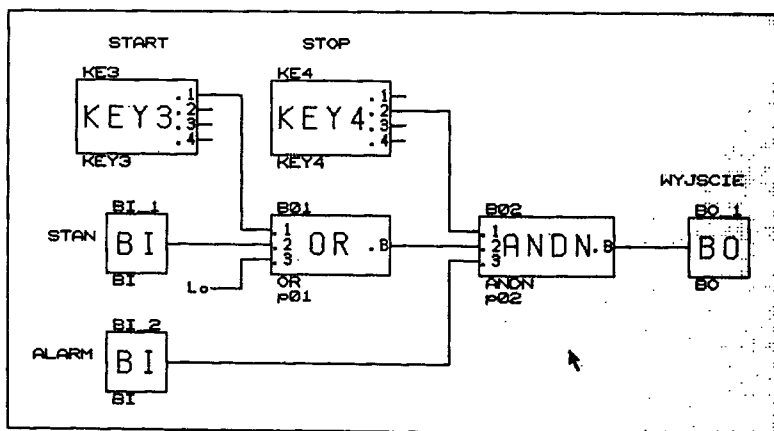
Charakterystyka pakietu OrCAD. Pakiet OrCAD - SDT [3] jest zestawem programów służących do projektowania schematów ideowych układów elektronicznych. Umożliwia on zestawianie schematu z elementów zawartych w bibliotece, oglądanie go na ekranie, drukowanie na drukarce, zapamiętanie schematu w formie graficznej jak i przetworzenie go do listy połączeń (*wire-list*). Bibliotekę elementów można łatwo modyfikować przy pomocy wbudowanego edytora czy też dodawać własne elementy (bloki funkcyjne).

Małe schematy mieszczące się na jednym arkuszu roboczym mogą być wykonane oddzielnie bez powiązań z innymi arkuszami. Dla większych schematów, mieszczących się na wielu arkuszach i

wymagających wielu różnych połączeń między sobą, pakiet OrCAD dostarcza ułatwienia w postaci hierarchicznej organizacji arkuszy roboczych. Każdy arkusz roboczy stanowi makroblok lub zawiera część schematu. Połączenia między arkuszami zorganizowane są w postaci dwustronnej siatki połączeń. Podobnie jak w strukturze płaskiej jednoarkuszowej, moduły portów (*ports*) służą do oznaczania sygnałów opuszczających arkusz roboczy. W strukturze hierarchicznej sieć połączeń powoduje połączenia modułów portów oznaczonych takimi samymi symbolami na różnych arkuszach.

Po wykonaniu i uaktualnieniu projektu możemy poddać go działaniu programów narzędziowych, które dokonają stosownych poprawek i sprawdzeń oraz utworzą dodatkowe pliki z wynikami swoich działań. Najpierw należy projekt poddać działaniu programu ANNOTATE, który sprawdzi kompletność oznaczeń i wstawi w miejsca brakujących własne standardowe oznaczenia. W celu uzyskania wykazu połączeń hierarchicznych należy użyć programu NETLIST. Utworzona lista zawiera nazwy wczytanych bibliotek, definicje etykiet lokalnych i modułów portów, opis elementów (bloków) użytych na schemacie i wykaz wszystkich połączeń.

Etapy tworzenia konfiguracji. Konfiguracja graficzna regulatora wchodzącego w skład rozproszonego systemu automatyki przypomina tworzenie schematu ideowego układu elektronicznego (z układów scalonych, rezystorów itp.). Dzięki temu stosunkowo niewielkim kosztem można zaadaptować pakiet OrCAD do tego celu [4]. Opracowano więc odpowiednie biblioteki bloków funkcyjnych dla poszczególnych elementów systemu automatyki. Opis każdego bloku funkcyjnego zawiera: nazwę bloku (*reference*), funkcję przyporządkowaną do bloku (*part value*), ilość i nazwę wejść/wyjść (*pins*) oraz wygląd graficzny (*body*) symbolu na ekranie. Z biblioteką ściśle współpracuje program tłumaczący plik z formatu *net-list* do języka opisu konfiguracji (tłumacz formatów).



Rys.3. Konfiguracja układu Start/Stop z wykorzystaniem pakietu OrCAD.

Tworzenie schematu. Poniżej podano przykład tworzenia konfiguracji dla sterownika PSW-8 (konfiguracja graficzna innych regulatorów przebiega podobnie, wymaga tylko zainstalowania innych bibliotek i programów pomocniczych). W szczególności:

- z głównego menu *Schematic Design Tools* wybieramy *Draft→Execute* wchodząc do edycji arkusza (o nazwie *test_153.sch*), następnie wybieramy z menu opcję *GET* (pobierz z biblioteki) i zleceniem *place* umieszczamy wybrane bloki funkcyjne na arkuszu (schemacie), łączymy je ze sobą w zadaną strukturę zleceniem *place→wire* (oraz *place→junction*), wpisujemy nazwy bloków zleceniem *Edit→edit→reference→name* i zapisujemy utworzony schemat,
- uruchamiamy programy pomocnicze: *User→PSW_P??* (wstawiający w pola *Field_1* elementów tekst "p??" dla bloków pozycjonowanych) oraz *User→PSW_PAR* (uzupełniający pola *Field_2...6* o wartości domyślne parametrów *on-line* i *off-line*),

- ponownie wracamy do programu *Draft*→*Execute* i wybierając z menu *Edit*→*edit*→*field_1* wpisujemy kolejne numery pozycji (dla bloków pozycjonowanych) oraz w polach *field_2...6* ew. zmieniamy wartości parametrów, następnie zapisujemy schemat (*Quit*→*update_file*→*abandon_edits*) i powracamy do menu głównego *Schematic Design Tools*,
- z menu głównego uruchamiamy program *Create_Netlist*→*Execute*, który tworzy plik opisujący projekt oraz listę połączeń w formacie wewnętrznym *INET* systemu OrCAD,
- jeśli tworzenie listy połączeń nie wykazało błędów na schemacie, to możemy teraz uruchomić translator *Or_CTL* (zleceniem *User*→*Or_PSW*→*Execute*) tworzący z listy połączeń program w języku *CONF_DL* (opisany szczegółowo w punkcie 2).

Translator Or_CTL. Jak wspomniano, translator *Or_CTL* (*OrCAD to Configuration Text Language*) jest programem usługowym tłumaczącym listę połączeń utworzoną zleceniem *Create_Netlist* systemu OrCAD do programu źródłowego w języku *CONF_DL*. Translator został napisany jako trójprzebiegowy. Na początku wczytywany jest (podany w linii zlecenia) plik projektu (np. *test_153.inx*) utworzony przez OrCAD i zawierający nazwy wszystkich plików wchodzących w skład całej struktury (strategii). Następnie wczytywane są nagłówki (*headers*) wszystkich plików projektu i budowane jest drzewo hierarchii (arkusz główny, schematy makrobloków). W dalszej części kolejno przetwarzane są poszczególne listy połączeń (*net-list*) według algorytmu:

- *pass 1*: wczytuj kolejne linie, rozpoznaj typ instrukcji, wypełnij odpowiednie tablice robocze,
- *pass 2*: wyszukaj i przetwarzaj połączenia między blokami, sprawdzaj zgodności typów, eliminuj etykiety lokalne,
- *pass 3*: buduj strukturę pliku wynikowego (z programem w języku *CONF_DL*) na podstawie danych zapamiętanych w tablicach roboczych.

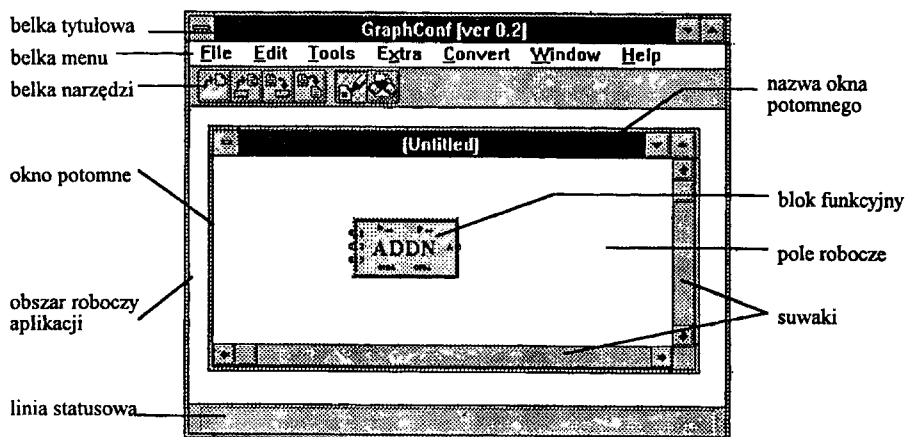
Na koniec program z opisem struktury zapisywany jest do pliku wyjściowego i program kończy pracę. *Nie jest sprawdzana poprawność semantyczna pliku wyjściowego.* To zadanie spoczywa na kompilatorze języka *CONF_DL* (opisanym wcześniej).

4. EDYTOR GRAFICZNY GraphConf

W wyniku przeprowadzonej analizy dostępnych na rynku zagranicznym (gdyż brak jest krajowych) programów dla konfiguracji graficznej systemów automatyki przy opracowywaniu nowego pakietu przyjęto następujące założenia:

- jądrem pakietu będzie uniwersalny edytor graficzny, pozwalający na "narysowanie" schematu układu automatyki na ekranie komputera, umożliwiający jego edycję, zmianę parametrów (nasław), ustawienie przełączników konfiguracyjnych itp.,
- bazą danych dla edytora będą biblioteki elementów, zawierające rysunki bloków funkcyjnych, informacje o parametrach, wyglądzie okien dialogowych oraz dane dla pliku wynikowego,
- wynikiem działania edytora będzie pewien plik pośredni (z tekstowym opisem konfiguracji), który następnie będzie kompilowany do właściwego pliku binarnego (zawierającego dane konfiguracyjne dla konkretnego regulatora, np. *PSW-8*) - umożliwi to bardziej elastyczną strukturę procesu konfiguracji regulatorów,
- kontrola poprawności schematu, zakresy wartości parametrów będą sprawdzane dopiero przez kompilator języka opisu konfiguracji i w miarę możliwości korygowane automatycznie,
- jądro pakietu (edytor graficzny) zostanie napisane w języku C++ (Borland C++ v.4.0 [1]) w systemie MS Windows i zewnętrznie przypominać będzie znane pakiety, jak CorelDraw, Symulink czy Word, co znacznie przyspieszy naukę obsługi programu,
- dodatkowo, opracowany zostanie translator plików wynikowych jednego ze znanych pakietów CAD (ostatecznie wybrano OrCAD jako najczęściej spotykany - punkt 3) do programu w języku *CONF_DL* oraz biblioteki "elementów" (bloków funkcyjnych) wybranych regulatorów.

Etapy tworzenia konfiguracji. Bezpośrednio po uruchomieniu edytora na ekranie zgłasza się okno podstawowe aplikacji o wyglądzie jak na rysunku poniżej:



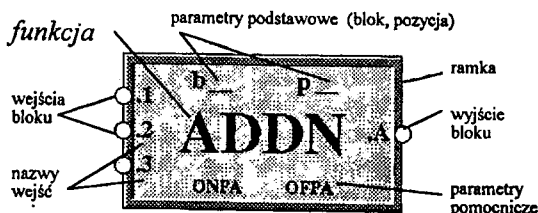
Rys.4. Okno podstawowe pakietu GraphConf.

W belce menu znajdują się następujące polecenia (wersja v.0.2 programu):

- **File:** podstawowe operacje na plikach: *New* - otwarcie nowego arkusza, *Open* - wczytanie arkusza z pliku, *Save* - zapisanie arkusza do pliku, *Save As* - zapisanie z nadaniem nowej nazwy, *Close* - zamknięcie arkusza bez zapisu, *Exit* - koniec pracy programu,
- **Edit:** edycja aktualnego arkusza: *Undo* - anulowanie ostatniej operacji, *Clear* - usunięcie (wymazanie) wszystkich obiektów (bloków, połączeń) z bieżącego arkusza,
- **Tools:** podstawowe narzędzia: *Edit Mode* - tryb edycji, *Zoom Mode* - zmiana skali rysunku (powiększanie/zmniejszanie), *Font* - wybór rodzaju i wielkości fontów,
- **Extra:** dodatkowe narzędzia: *Vertical Scrollbar*, *Horizontal Scrollbar* - umieszczenie suwaków na brzegach arkusza roboczego (okna potomnego),
- **Convert:** uruchomienie programu tworzącego opis konfiguracji w języku CONF_DL,
- **Window:** wygląd i rozmieszczenie okienek: *Cascade*, *Tile*, *Arrange Icons* - sposób rozmieszczenia okienek potomnych w oknie głównym, *AddView* - podgląd i edycja plików *.sch, *Block Selection* - wybór bloków funkcyjnych z biblioteki elementów,
- **Help:** informacje pomocnicze o programie.

Otwarcie okna *Block Select* udostępnia listę wyboru bloków funkcyjnych regulatora z aktualnej biblioteki. Wybrany blok może być umieszczony w polu roboczym przy aktywnym trybie *Edit Mode*.

Wszystkie bloki w bibliotece mają ujednolicony wygląd. Obrzeże bloku otoczone jest szarą ramką, z lewej strony znajdują się wejścia (białe kółeczka) i ich nazwy, wzdłuż prawego boku wyjścia, a wewnątrz znajduje się nazwa funkcji (przyporządkowanej do bloku) i opisy pomocnicze.



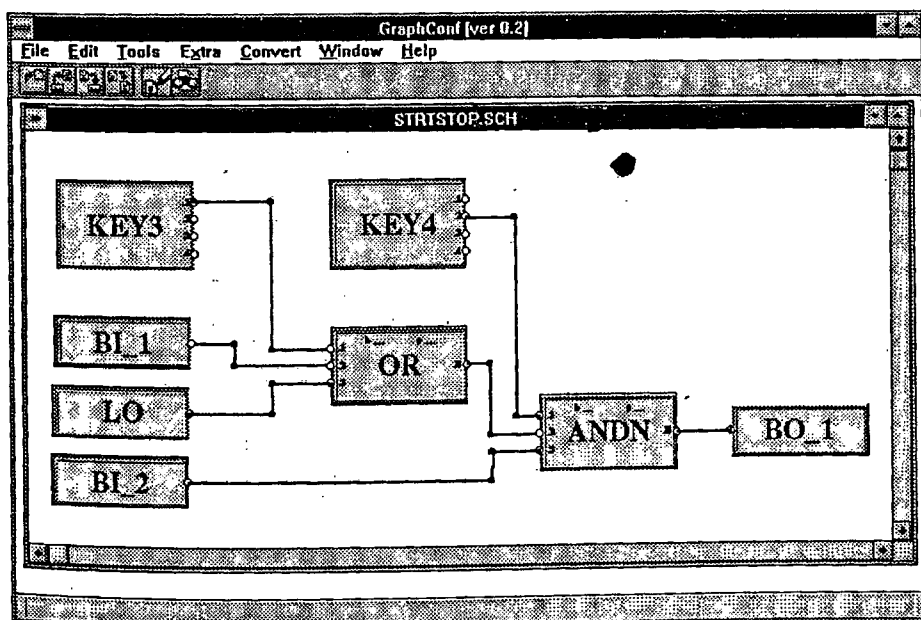
Rys.5. Wygląd bloku funkcyjnego w programie GraphConf.

Podstawowe operacje myszą podczas edycji schematu to:

- kliknięcie lewym przyciskiem myszy w obszarze roboczym → umieszczenie aktualnie wybranego z biblioteki (podświetlonego) bloku funkcyjnego na arkuszu roboczym (schemacie),
- wskazanie obiektu i kliknięcie lewym przyciskiem → zaznaczenie obiektu (bloku, połączenia),
- klawisz *DEL* → usunięcie ze schematu zaznaczonego uprzednio obiektu (bloku, połączenia),
- wskazanie obiektu i ciągnięcie → przesunięcie obiektu na nowe miejsce arkusza roboczego,
- kliknięcie prawym przyciskiem w obszarze połączenia → dodanie nowego punktu załamania,
- kliknięcie prawym przyciskiem w obszarze pinu → wyświetlenie informacji o typie połączenia.

Układ Start/Stop. Dla przykładu, poniżej podany zostanie sposób postępowania przy konfiguracji układu Start/Stop z poprzedniego punktu. Obejmuje on następujące czynności:

- uruchamiamy program GraphConf (z *Menadżera Programów*), na ekranie zgłasza się główne okno aplikacji (początkowo aktywne są tylko dwie ikony: *New* oraz *Open*), otwieramy nowy arkusz roboczy wybierając *New* → *Schematic Files* (okno *Untitled*) oraz otwieramy bibliotekę bloków funkcyjnych sterownika PSW-8 zleceniem *Window* → *Block Selection*,
- wybieramy potrzebny blok z biblioteki (kliknięcie myszą podświetla nazwę), ustawiamy kursor w danym miejscu okna roboczego i kliknięcie lewym przyciskiem umieszcza blok na schemacie - następnie wybieramy i umieszczamy kolejny blok itd.,
- ustawiamy kursor na wybranym wyjsciu bloku i ciągnąc (trzymając wciśnięty lewy przycisk myszy) kursor do wyjścia bloku ustalamy połączenie, kliknięcie prawym przyciskiem myszy na połączeniu dodaje kolejny punkt załamania, zaznaczenie i przeciągnięcie punktu umożliwia dowolne kształtowanie połączenia,
- podwójne kliknięcie na bloku otwiera okno dialogowe, dzięki któremu możemy nadać blokowi numer, ustalić jego pozycję w łańcuchu obliczeń, czy zmienić wartość parametru,
- po zakończeniu edycji całego schematu zapisujemy go wybierając zlecenie *File* → *Save As* i tworzymy plik z opisem konfiguracji w języku CONF_DL (zlecenie *Convert*).



Rys.6. Konfiguracja układu Start/Stop z wykorzystaniem pakietu GraphConf.

PODSUMOWANIE

Projekt obejmował opracowanie metody i pakietu oprogramowania dla konfiguracji graficznej rozproszonego systemu automatyki składającego się z konfigurowalnych urządzeń obiektowych oraz komputera nadrzędnego. Opracowano trzy podstawowe narzędzia:

- translator **OrCTL** dla tłumaczenia plików w formacie pakietu OrCAD (*.inf) na pliki tekstowe z opisem konfiguracji w języku CONF_DL, oraz odpowiednie biblioteki bloków funkcyjnych,
- uniwersalny edytor graficzny **GraphConf** dla graficznej konfiguracji elementów rozproszonego systemu automatyki (zbliżony do formatu FBD - *function block diagram*),
- kompilator specjalizowanego języka tekstowego CONF_DL opisu struktur sterowania, będący ogniwem pośrednim między schematem blokowym (w formie graficznej) a binarnymi danymi konfiguracyjnymi (obrazem pamięci wewnętrznej) regulatora.

Pierwsze z nich pozwala wykorzystać szeroko znany pakiet OrCAD (wersja 4.20) do tworzenia danych konfiguracyjnych dla sterowników wielofunkcyjnych. Zbiór w formacie *net-list* przetwarzany jest przez translator **OrCTL** do pliku tekstowego z opisem konfiguracji dla wybranego sterownika.

Drugim narzędziem jest pakiet do konfiguracji graficznej zarówno elementów składowych (regulatorów, sterowników) jak i całego systemu automatyki. Jego jądrem jest uniwersalny edytor graficzny **GraphConf** umożliwiający edycję schematu układu składającego się z bloków funkcyjnych importowanych z biblioteki (dostarczanej wraz z pakietem lub tworzonej przez użytkownika). Wynikiem działania programu są dwa pliki: *.grf będący opisem schematu tworzonego na ekranie komputera oraz *.cnf zawierający opis konfiguracji w specjalnie stworzonym języku CONF_DL.

Programy te łączy kompilator **CT_COMP**, którego zadaniem jest przetworzenie pliku z opisem struktury na postać zrozumiałą dla konkretnego urządzenia systemu automatyki (np. regulatora). Wykrywa on i sygnalizuje wszelkie błędy w konfiguracji (np. błędne połączenia, wartość parametru poza zakresem, itp.), stara się je korygować lub podpowiadać użytkownikowi możliwe warianty poprawienia błędu. Stanowi on łącznik pomiędzy uniwersalnym edytorem graficznym a specjalizowanym urządzeniem automatyki (regulatorem).

Należy jeszcze raz podkreślić, że głównym założeniem przy opracowywaniu pakietu była jego uniwersalność i jednolity sposób konfiguracji zarówno poszczególnych urządzeń rozproszonego systemu automatyki jak i stacji operatorskiej. Autorzy sądzą, że zostało ono spełnione.

LITERATURA

- [1] Barkakati N., *Borland C++ 4 - sztuka programowania*, Oficyna READ ME, W-wa, 1995
- [2] *ObjectWindows for C++ - reference guide*., Borland International, 1993
- [3] Safinowski G., *OrCAD SDT/PCB*., Wydawnictwo PLJ, W-wa, 1991
- [4] Świder Z., Trybus L., Śniezek M., *Graphical configuration of a multifunction controller using OrCAD*., I Kolokwium CAE Bielefeld-Rzeszów, 1994
- [5] Trybus L., *Regulatory wielofunkcyjne*., WNT, W-wa, 1992