ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2000
Seria: AUTOMATYKA z. 131 Nr kol. 1476

Lech GRODZKI
Politechnika Białostocka

GRAFICZNE ŚRODOWISKO PROJEKTOWE DO PROGRAMOWANIA NIESTANDARDOWYCH STEROWNIKÓW*

Streszczenie. Artykuł przedstawia oprogramowanie wspomagające programowanie niestandardowych sterowników binarnych. Graficzny edytor automatów sterujących opiera się na wcześniej opracowanej i prezentowanej już metodzie. Program pracuje w środowisku Windows i umożliwia: opisanie pracy sterownika odpowiednim grafem przejść, zdefiniowanie słowa wejściowego i wyjściowego automatu oraz przygotowanie jego opisu w składni tekstowego języka opisu lub od razu docelowego języka programowania mikrosterownika.

GRAPHICAL PROGRAMMING ENVIRONMENT FOR NON-STANDARD CONTROLLERS

Summary. The article presents software tools for programming of the non-standard binary controllers. Graphical editor of the controlling automata is based on earlier worked-out and presented method. Programme works in Windows environment and allows: to describe the controller action by appropriate transition graph, to define automata input and output words, to receive the automata description in syntax of special description language or directly in syntax of chosen microprocessor programming language.

1. Wstęp

Pomimo dostępności na rynku sprzętu automatyki dużej gamy gotowych sterowników o różnorodnych możliwościach, nadal znajdują zastosowanie niestandardowe rozwiązania. Realizowane są one jako konstrukcje jednostkowe, na specjalne zamówienie lub mikrosterowniki wbudowane do wnętrza innych urządzeń produkowanych seryjnie. Programowanie takich nietypowych sterowników polega na wykorzystaniu dostępnych dla stosowanego w systemie mikroprocesora języków programowania (makroasemblery, kroskompilatory). Ponieważ z reguły nie są to języki o składni zorientowanej na realizację algorytmów sterowania binarnego, przydatne byłoby narzędzie umożliwiające przejście

* Praca finansowana z projektu badawczego W/WE/1/99 finansowanego przez KBN.

z „potocznego" opisu algorytmu do składni języka docelowego. Jednym z możliwych rozwiązań jest opracowana wcześniej i już prezentowana metoda JODA [1, 3, 4] oraz translator tekstowego opisu automatu na język docelowy [2, 3].

Wspomniana wyżej metoda zakłada możliwość opisu pracy sterownika jako automatu typu Moore'a. Jednym ze sposobów opisu tego typu automatów jest graf przejść, którego węzły odpowiadają stanom, a skierowane krawędzie - możliwym przejściom między tymi stanami. Ponadto węzły takiego grafu są opisane symbolami stanów, a krawędzie - wyrażeniami logicznymi warunkującymi przejścia pomiędzy stanami. Argumentami wyrażeń logicznych są, oczywiście, zmienne wejściowe automatu. Taka graficzna forma prezentacji automatu jest dosyć przejrzystym narzędziem opisu i z powodzeniem nadaje się do zaimplementowania w graficznym środowisku programowym. Niniejszy artykuł przedstawia jedną z możliwych jego realizacji.

2. Możliwości graficznego edytora automatów sterujących

2.1. Rysowanie grafu automatu

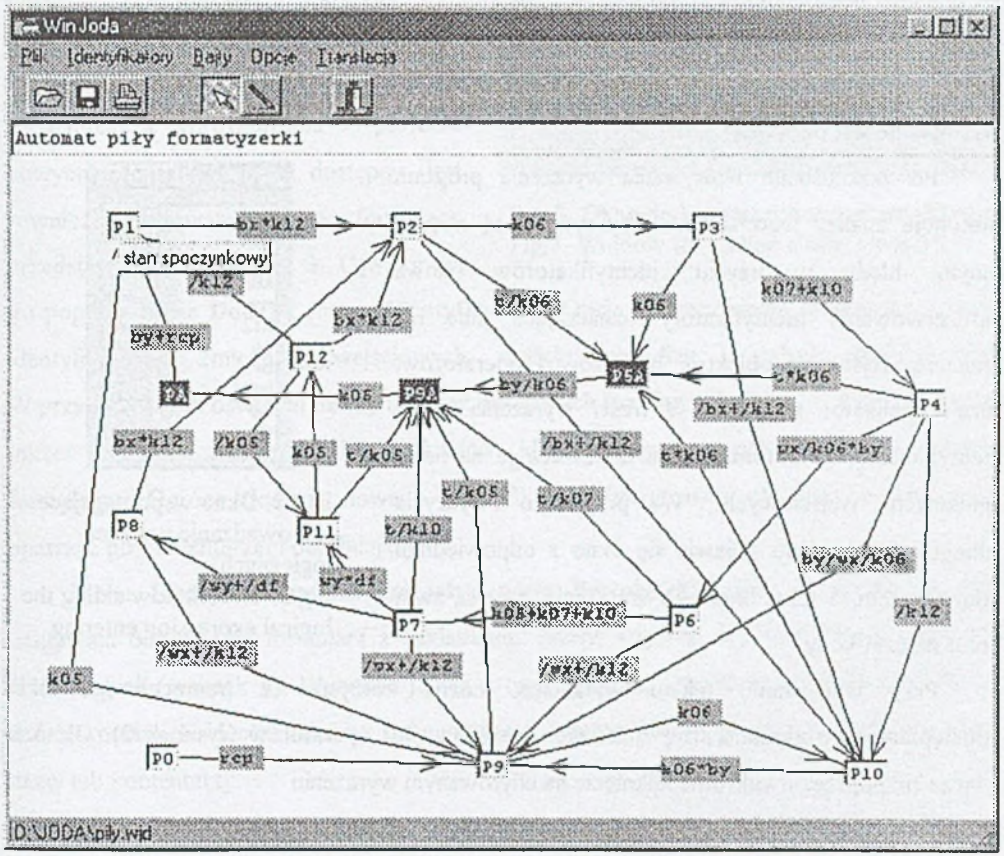
Rysunek 1 przedstawia główny ekran programu WinJoda, zawierający arkusz do rysowania grafu automatu sterującego. Rysowanie grafu automatu odbywa się w dwóch trybach (wybieranych przyciskami **Strzałka** i **Ołówek**):

- trybie wstawiania - umożliwiającym dodawanie do rysunku nowych stanów (lewy klawisz myszy z shiftem) oraz przesuwanie elementów składowych grafu, takich jak pola stanów i wyrażeń logicznych;
- trybie rysowania/edycji - umożliwiającym łączenie pól stanów skierowanymi krawędziami, opatrzonymi polami na funkcje przejść, a także wpisywanie tych funkcji i edycję nazw stanów.

Przy wstawianiu nowych stanów do projektu, program automatycznie nadaje im identyfikatory w rodzaju: Sx. Nazwy te mogą być później zmienione w trybie edycji na bardziej opisowe.

Ze względu na zastosowanie w konstrukcji grafu automatu gotowych obiektów języka Delphi, zarówno pola stanów, jak i pola funkcji przejść mają postać prostokątów. Mimo to pola te są łatwo rozróżnialne, ponieważ standardowo pola funkcji nie mają obwódki i są jasnoszare, a pola stanów mają prostą obwódkę i białe tło. Ponadto pola funkcji są rozmieszczone na krawędziach grafu, a pola stanów - w jego węzłach, z reguły wskazywanych przez groty

krawędzi grafu. Także tekstowe zawartości obu rodzajów pól sugerują ich znaczenie. Projektant ma także możliwość modyfikacji koloru tła każdego pola indywidualnie. Aby tego dokonać, należy użyć prawego klawisza myszy wskazując jednocześnie kursorem wybrane pole - pojawi się wtedy menu podręczne, którego jedną z opcji jest **Kolor**. Opcja ta uruchamia standardowy dialog wyboru koloru tła. Kolor czcionki wewnątrz pola jest dobierany automatycznie na zasadzie kontrastowości. Operowanie kolorami pól może znaleźć zastosowanie przy konieczności wyróżnienia wybranych fragmentów grafu, odpowiedzialnych na przykład za obsługę stanów alarmowych.



Rys. 1. Główne okno programu z grafem automatu
Fig.1. Programme main window with automata graph

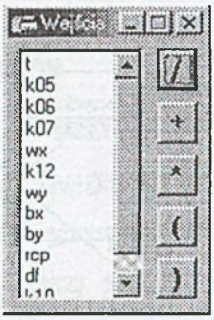
Menu podręczne w odniesieniu do pól stanów i funkcji przejść ma jeszcze dwie opcje: **Usuń** i **Komentarz**. Pierwsza z nich usuwa z grafu wskazane pole wraz ze wszystkimi krawędziami z nim połączonymi. Opcja **Komentarz** pozwala na wpisanie dodatkowego tekstu

(do 250 znaków) opisującego przeznaczenie danego stanu lub funkcji. Komentarz ten będzie się później pojawiał po zatrzymaniu kursora myszy na danym polu.

2.2. Definiowanie funkcji przejść

Przy włączonym trybie rysowania/edycji (aktywny przycisk **Ołówek**), po wskazaniu myszą stanu wyjściowego i docelowego, na ekranie pojawia się odpowiednio skierowana krawędź przejścia, z umieszczonym na niej polem na wyrażenie logiczne. Po przeniesieniu kursora do wnętrza tego pola, można wpisać tekst wyrażenia. Składnia wyrażenia pozwala na stosowanie identyfikatorów zmiennych wejściowych, operatorów logicznych (+ - suma, * - iloczyn, / - negacja) oraz wielopoziomowych nawiasów. Można także stosować zapis uproszczony, polegający na pomijaniu operatora iloczynu wszędzie tam, gdzie jego wystąpienie jest oczywiste.

Po zakończeniu wpisywania wyrażenia program dokonuje analizy jego składni. Wykrywane są między innymi błędy w użyciu: identyfikatorów (słowa zarezerwowane, identyfikatory oznaczające inne niż zmienne wejściowe obiekty), nawiasów i operatorów. Jeżeli analizator rozpozna w treści wyrażenia nowe identyfikatory, to automatycznie umieszcza je na liście zmiennych wejściowych. W przypadku wykrycia jakiegokolwiek błędu pojawia się okno z odpowiednim komunikatem, a tekst błędnego wyrażenia zmienia swój kolor na czerwony.



Rys.2. Okno wspomagające wprowadzanie wyrażeń logicznych
Fig.2. The window aiding the logical expression entering

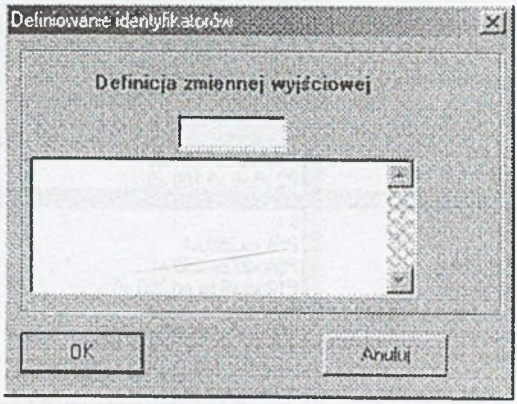
Przy wpisywaniu tekstu wyrażenia można korzystać z pomocniczego okna udostępniającego aktualną listę zmiennych wejściowych i operatorów (rysunek 2). Okno to włącza się poprzez dwukrotne kliknięcie na edytowanym wyrażeniu.

2.3. Posługiwanie się identyfikatorami

Program umożliwia operowanie symbolicznymi nazwami stanów oraz zmiennych, tak wejściowych, jak i wyjściowych. Nazwy te mogą mieć długość do 10 znaków i muszą zaczynać się literą. Ponieważ program bazuje na składni tekstowego języka opisu automatów sterujących JODA [2, 3], istnieje lista zarezerwowanych identyfikatorów, które nie mogą być użyte jako nazwy stanów lub zmiennych. Identyfikatorami tymi są w bieżącej wersji programu

słowa: *automat, az80, a8051, bity, c, c16, c32, dekoduj, flagi, funkcje, jezyk, koniec, list, off, on, opt, pascal, stany, wyjscia*. Użycie ich przez projektanta, niezależnie od wielkości liter, jako nazw stanów lub zmiennych jest sygnalizowane jako błąd.

Podczas wprowadzania i edycji nazw nowych stanów automatycznie uzupełniany jest słownik identyfikatorów. Również wpisywanie wyrażeń logicznych w pola funkcji przejść powoduje dodawanie do tego słownika pojawiających się po raz pierwszy w projekcie nazw zmiennych wejściowych. Oprócz tych automatycznych mechanizmów wprowadzania do projektu nowych identyfikatorów, dostępny jest również dedykowany do tego formularz, przedstawiony na rysunku 3. Otwiera się go poprzez opcje **Dodaj ...** menu **Identyfikatory**. Opcje te umożliwiają dodawanie nowych identyfikatorów zmiennych wejściowych, wyjściowych, flag i stałych wielobitowych. W przypadku tych ostatnich należy również określić ich rozmiar w bajtach (1, 2 lub 4). Można także korzystać z odpowiednich skrótów klawiaturowych przyśpieszających otwarcie formularza. Oprócz dodania nowego identyfikatora (w górnym okienku), można także opatrzyć go komentarzem objaśniającym.

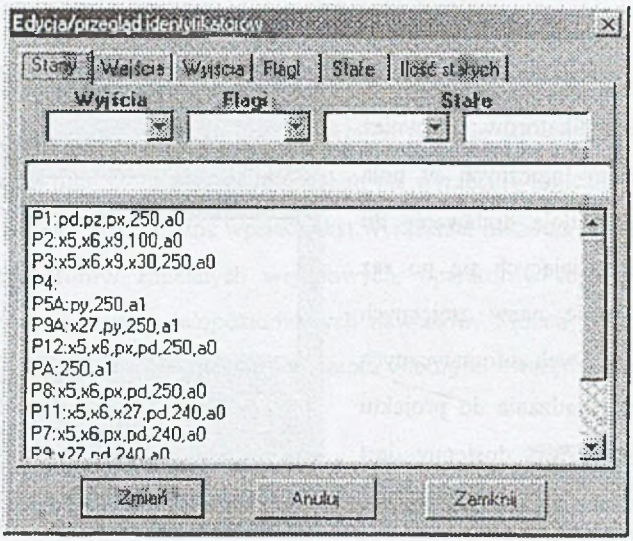


Rys.3. Okno dodawania nowego identyfikatora
Fig.3. Window for adding a new identifier

Menu **Identyfikatory** zawiera także opcję **Przegląd/Edycja**. Wybranie tej opcji uaktywnia odpowiedni formularz z zakładkami: *Stany*, *Wejścia*, *Wyjścia*, *Flagi*, *Stałe* oraz *Ilość stałych* (rysunek 4). Wybranie jednej z tych zakładek umożliwia sprawdzenie listy już zadeklarowanych identyfikatorów danego typu i wprowadzenie do niej potrzebnych zmian nazw lub komentarzy.

Wybranie zakładki *Stany* pozwala nie tylko przejrzeć listę zdefiniowanych stanów, ale także określić wartości wyjść dla każdego z tych stanów. Wykorzystuje się tu fakt, że wartości wyjść automatu Moore'a zależą wyłącznie od bieżącego stanu automatu. Wskazaniu listy aktywnych wyjść służą, pojawiające się tylko na tej zakładce, listy wyboru zmiennych wyjściowych, flag i stałych. Jeżeli jakieś wyjście ma mieć wartość logicznej jedynki w danym stanie, to należy wybrać z listy identyfikator tego wyjścia i dodać go do wiersza opisu stanu

w okienku poniżej. Ponadto, jeżeli w słowie wyjściowym używane są, zgodnie z metodą JODA, także stałe wielobitowe, to dla każdego ze stanów należy podać wartości tych stałych.



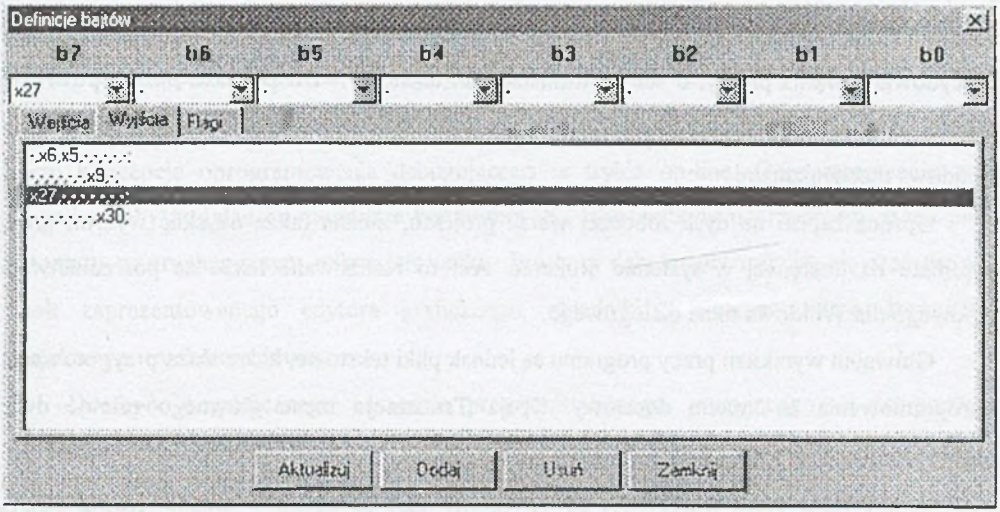
Rys. 4. Okno edycji i przeglądu identyfikatorów z wybraną zakładką *Stany*
Fig. 4. Viewing and modification window with active *Stany* tab

2.4. Projektowanie struktury słów wejściowych i wyjściowych

Słowo wejściowe automatu może składać się z maksimum 32 bajtów. Jest to ograniczenie przeniesione z wcześniej opracowanego translatora tekstowego języka opisu automatów [2, 3]. Bajty składające się na to słowo nie muszą być w pełni wykorzystane. Użycie lub nie wybranego bitu w bajcie słowa wejściowego może zależeć na przykład od struktury i wykorzystania układów wejść dwustanowych sterownika. Analogiczne reguły odnoszą się do struktur słów wyjść i flag automatu.

Projektowanie słów wejściowych i wyjściowych automatu odbywa się na odrębnym formularzu, uaktywnianym opcją **Bajty** z menu głównego. Wygląd tego formularza przedstawia rysunek 5. Projektant ma do dyspozycji trzy zakładki: *Wejścia*, *Wyjścia* i *Flagi*, które służą wyborowi interesującego go zestawu bajtów. Wybór jednej z zakładek powoduje wyświetlenie w polu poniżej aktualnego zestawu bajtów składających się na słowo danego typu. Powyżej zaś dostępnych jest osiem list wyboru identyfikatorów, odpowiadających ośmiu pozycjom bitów w bajcie: od najstarszego po lewej stronie do najmłodszego - po prawej. Jeżeli któraś z pozycji w bajcie ma być niewykorzystana, to wybiera się dla niej znak '-'. Zadeklarowanie nowego bajtu polega na dokonaniu odpowiednich wyborów w listach wyboru

i wciśnięciu klawisza **Dodaj**. Nowy bajt jest umieszczany na końcu widocznej poniżej listy bajtów albo przed bajtem wcześniej wyróżnionym na tej liście. Wyróżnienie takie uzyskuje się pojedynczym kliknięciem myszy na wybranym wierszu listy.



Rys.5. Formularz do projektowania słów wejściowych automatu
Fig.5. Designing form for input and output words

Raz zadeklarowane bajty można edytować. Robi się to przez dwukrotne kliknięcie lewym przyciskiem myszy na wybranym bajcie albo poprzez jego zaznaczenie i wybranie przycisku **Zmień**. Operacje te powodują skopiowanie wybranego bajtu do okienek list wyboru i zmianę funkcji klawisza ze **Zmień** na **Aktualizuj**. Po dokonaniu poprawek wciskamy klawisz **Aktualizuj**, który ponownie przyjmuje funkcję **Zmień** albo dwukrotnie klikamy na wcześniej wyróżnionym bajcie. Jeżeli podczas aktualizacji wybierzemy myszką inny bajt słowa, procedura edycji bajtu ulega anulowaniu - pojawia się ponownie klawisz **Zmień**.

Oprócz zmiany zawartości raz wprowadzonych bajtów, projektant może także zmienić ich kolejność w słowie. Wykonuje się to w typowy dla środowisk graficznych sposób. Wystarczy zaznaczyć bajt, który chcemy przemieścić, a następnie przytrzymując lewy klawisz myszy przesunąć go na wybraną pozycję.

Widoczny u dołu formularza klawisz **Usuń** pozwala skasować zaznaczony bajt, a klawisz **Zamknij** kończy pracę formularza.

2.5. Wyniki pracy programu

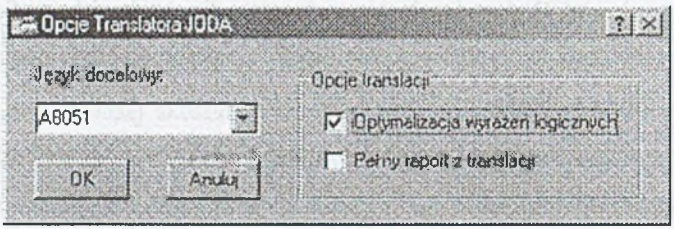
Przygotowany w całości lub tylko częściowo projekt może być zapisany w pliku dyskowym. Realizuje się to poprzez wybranie opcji **Zapisz** z menu **Plik** lub wciśnięcie na górnym pasku odpowiedniego klawisza. Program otwiera standardowy dialog umożliwiający wskazanie miejsca i nazwy generowanego pliku z projektem. W wersji rozwojowej programu zdecydowano się na prosty, o stałym formacie plik tekstowy. Do projektu można powrócić w kolejnej sesji pracy z programem, otwierając ten plik również typowymi dla środowiska Windows mechanizmami.

Oprócz zapisu na dysk roboczej wersji projektu, można także uzyskać wydruk grafu automatu na dostępnej w systemie drukarce. Jest to realizowane także za pośrednictwem typowego dla Windows okna dialogowego.

Głównym wynikiem pracy programu są jednak pliki tekstowe, które służą przygotowaniu oprogramowania na system docelowy. Opcja **Translacja** menu głównego oferuje dwie możliwości:

- wygenerowanie pliku tekstowego zawierającego opis automatu w składni języka JODA [2, 3] - plik ten może być następnie przetwarzany przez wcześniej opracowany translator tego języka (podopcja **Plik źródłowy**);
- wygenerowanie od razu pliku tekstowego zawierającego opis zakodowanego automatu w składni docelowego języka programowania mikrosterownika (podopcja **Plik wynikowy**).

W obu przypadkach konieczne jest jednak wcześniejsze określenie opcji pracy translatora. Służy do tego przedstawione na rysunku 6 okno dialogowe. Pozwala ono wskazać język docelowy programowania mikrosterownika (az80, a8051, c16, c32, Pascal, C) oraz ustawić opcje translacji (optymalizacja wyrażeń logicznych, rodzaj raportu z translacji). Informacje podane za pośrednictwem tego okna są uwzględnianie podczas pracy wbudowanego translatora albo przekazywane odpowiednimi dyrektywami w treści pliku źródłowego.



Rys.6. Okno opcji translacji
Fig.6. Translation options windows

3. Podsumowanie

Zaprezentowany w artykule program jest wersją rozwojową większego pakietu oprogramowania narzędziowego metody JODA. Sam edytor graficzny powinien być jeszcze wzbogacony o: możliwość pracy z większym niż ekran arkuszem roboczym i powiązane z tym mechanizmy skalowania obrazu, a także udoskonalone procedury wydruku rysunku grafu. Oprócz tego wypełnienia treścią wymaga jeszcze opcja **Pomoc** menu głównego.

W trakcie prac badawczych, których owocem jest przedstawiony program, opracowano także koncepcję oprogramowania debugującego w trybie on-line. Oprogramowanie takie umożliwiałoby podgląd na monitorze komputera PC, jako systemu nadrzędnego, stanu pracy automatu w uruchamianym mikrosterowniku. Program debugujący mógłby się stać drugim, obok zaprezentowanego edytora graficznego, składnikiem zintegrowanego środowiska projektowo-uruchomieniowego.

Dostępność graficznego narzędzia programistycznego, pracującego w powszechnie używanym środowisku Windows, ułatwi przygotowanie oprogramowania wbudowanych mikrosterowników, także programowanych w innych niż asembler językach programowania.

LITERATURA

1. Grodzki L.: Metoda oprogramowania sterowników binarno-ciągłych procesów przemysłowych na poziomie asemblera, Materiały VIII KKADPP Kozubnik, Zeszyty Naukowe Pol.Śl. s.Automatyka, z.109, Gliwice 1992, s.67-76.
2. Grodzki L.: Zastosowanie translatora JODA do programowania algorytmów sterowania binarnego, Materiały X KKADPP Zakopane, Zeszyty Naukowe Pol.Śl. s.Automatyka, z.118, Gliwice 1996, s.65-74.
3. Grodzki L.: Realizacja sterowania binarnego w mikroprocesorowych sterownikach procesów binarno-ciągłych, rozprawa doktorska, Politechnika Białostocka, Białystok 1996.
4. Grodzki L.: Zastosowanie automatów w sterowaniu i nadzorze urządzeń technologicznych, Materiały VI KKN-D Automatyzacja i eksploatacja systemów sterowania Gdynia, 1997, s.180-187.
5. Grodzki L.: Automata in Programming of Microprocessor-Based Binary Process Controllers Proceedings of the 4th MMAR, Międzyzdroje 1997, vol. 3, pp.1001-1006.

Recenzent: Dr inż. R.Jakuszewski

Abstract

Sometimes, we have to construct non-standard binary controller, like embedded or individually designed system. For programming of such systems we have only symbolic or sometimes high-level languages. Because these languages are not binary control oriented,

programming is not easy. That is why, methods and software tools for programming of non-standard controllers are looked for.

Software described in the paper is based on the earlier presented method called JODA. This method assumes, that the binary controller can be treated as Moore's automata. Such an automata can be described by appropriate transition graph with states and transition functions, and a output functions table. The paper presents the graphical environment for designing the controlling automata. Design process consist of several steps: drawing the transition graph, defining transition functions, declaring the structure of automata input and output words, declaring output variable values for each state, optionally adding comments to all objects of the automata description and finally generating the automata description in chosen destination language. On each step, programme aids the designer by verifying the data in lexical analysis of the state and variable identifiers and syntax analysis of the logic expressions. The paper describes briefly each of the mentioned designing steps with appropriate illustrations.

Presented software is the first evaluation version and will be improved in next editions. The larger designing system, which should contain presented graphical editor and debugging tools is planned in the future.