STUDIA INFORMATICA 2001
Volume 22 Number 3 (45)

Halina KAMIONKA-MIKUŁA
Politechnika Śląska, Instytut Informatyki

KOMPUTEROWE WSPOMAGANIE NAUCZANIA W ZAKRESIE UKŁADÓW SYNCHRONICZNYCH

Streszczenie. Sieci komputerowe coraz częściej są stosowane w nauczaniu, w tym nauczaniu na odległość. W tym przypadku ważnym elementem są programy wspomagające nauczanie. W pracy omówiono rozwiązania stosowane w programach dydaktycznych w celu umożliwienia wszechstronnych ćwiczeń w ramach tematyki nauczania i zaprezentowano rozwiązania zastosowane w programach wspomagających nauczanie w zakresie układów synchronicznych.

COMPUTER AIDED TEACHING OF SYNCHRONOUS SWITCHING CIRCUITS

Summary. Computer networks are used more and more often for teaching, including process of remote teaching. In this case, structure of programs for computer aided teaching is very important. The paper presents problems connected with design of didactic programs used for teaching of synchronous switching circuits. To make possible to cover in such programs a wide area of exercises there is a need to implement algorithms: for circuits synthesis, simulations and for knowledge checking.

1. Wprowadzenie

W przypadku zastosowania sieci komputerowych w nauczaniu, w tym nauczaniu na odległość, ważnym elementem są programy edukacyjne.

Warunkiem powodzenia procesu edukacji na odległość jest uwzględnienie w oprogramowaniu dydaktycznym możliwości interaktywnego uczenia się, symulacji procesów i przebiegu projektowania oraz atrakcyjnej interaktywnej formy motywującej do samokształcenia.

Zależnie od tematyki nauczania programy dydaktyczne w różny sposób umożliwiają zdobywanie wiedzy, np. do nauki języków obcych potrzebne są programy pozwalające na pamięciowe przyswojenie wiedzy.

W przypadku nauczania zagadnień syntezy układów cyfrowych mało są przydatne programy, które pozwalają na pamięciowe przyswojenie wiedzy. Istnieje natomiast potrzeba prezentowania wiedzy teoretycznej, demonstrowania przykładów syntezy, symulacji układów i sprawdzania nabytej wiedzy.

Chcąc uwzględnić zagadnienia programowanego nauczania, programy dydaktyczne powinny analizować wyniki sprawdzania wiedzy i zależnie od tych wyników proponować dalsze etapy nauki.

W zależności od zastosowanych w programie rozwiązań, demonstracja przykładów a także sprawdzanie wiedzy możliwe jest:

- tylko dla danych „zaszytych w programie" (narzuconych przez twórcę programu), albo
- dla danych dowolnie wybranych przez użytkownika, dla których tylko pewne parametry są ograniczone, np. liczba zmiennych opisujących układ.

W celu umożliwienia wszechstronnych ćwiczeń w ramach omawianej tematyki nauczania, w tym prezentacji przykładów i sprawdzania wiedzy dla danych dowolnie wybranych przez użytkownika, oprogramowanie powinno uwzględniać algorytmy:

- komputerowej syntezy i analizy układów,
- symulacji układów cyfrowych,
- komputerowego sprawdzania wiedzy.

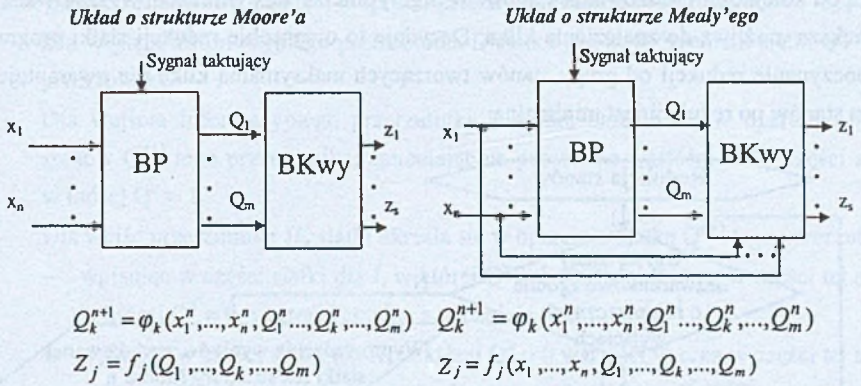
W Instytucie Informatyki Politechniki Śląskiej od wielu lat prowadzone są prace w zakresie opracowania metod syntezy i analizy układów cyfrowych oraz algorytmów dla komputerowego wspomagania nauczania i syntezy układów cyfrowych. Wyniki prac są wykorzystywane między innymi w programach dydaktycznych [4].

W referacie przedstawiono rozwiązania zastosowane w z programach:

- **Synchroniczne** - program przeznaczony do prezentacji kolejnych etapów syntezy układów synchronicznych dla dowolnych danych wprowadzonych przez użytkownika w postaci siatki przejść.
- **Digilab** - program przeznaczony do nauczania głównie liczników, umożliwiający:
  - prezentacje kolejnych etapów syntezy liczników równoległych,
  - symulację liczników i innych układów cyfrowych realizowanych zarówno w „strukturze szytej", jak i w wersji mikroprogramowanej,
  - tworzenie schematów układów cyfrowych.

2. Podstawy teoretyczne syntezy układów synchronicznych

Rys. 1 przedstawia często stosowane struktury synchronicznego układu sekwencyjnego.



$$Q_k^{n+1} = \varphi_k(x_1^n,...,x_n^n,Q_1^n...,Q_k^n,...,Q_m^n) \qquad Q_k^{n+1} = \varphi_k(x_1^n,...,x_n^n,Q_1^n...,Q_k^n,...,Q_m^n)$$

$$Z_j = f_j(Q_1,...,Q_k,...,Q_m) \qquad Z_j = f_j(x_1,...,x_n,Q_1,...,Q_k,...,Q_m)$$

Rys. 1. Schemat blokowy układu synchronicznego
Fig. 1. Block diagram of synchronous circuit

Blok Pamięci (BP) jest zrealizowany w oparciu o przerzutniki sterowane Blokiem Kombinacyjnym wejściowym (Bkwe).

Algorytm syntezy układów synchronicznych oparty o siatki przejść/wyjść

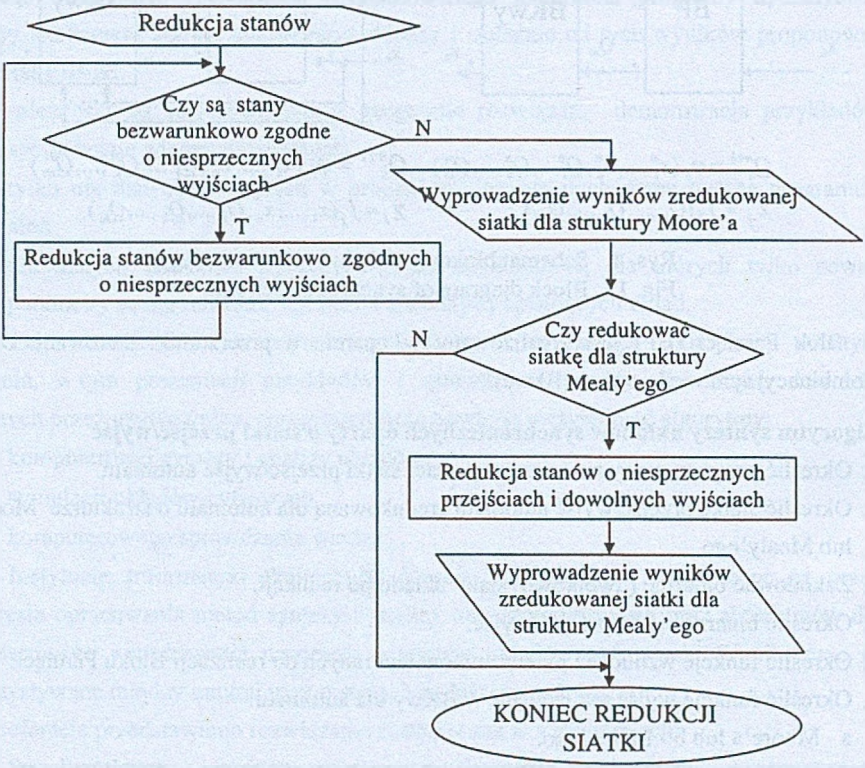
1. Określić program pracy automatu w postaci siatki przejść/wyjść automatu.
2. Określić siatkę przejść/wyjść automatu zredukowaną dla automatu o strukturze Moore'a lub Mealy'ego.
3. Zakodować binarnie (dwójkowo) stany układu po redukcji.
4. Określić binarną siatkę przejść/wyjść.
5. Określić funkcje wzbudzeń przerzutników wybranych do realizacji Bloku Pamięci.
6. Określić funkcje wyjść realizowane w BKwy dla automatu:
   a - Moore'a lub b - Mealy'ego.
7. Określić minimalne wyrażenia dla funkcji wzbudzeń przerzutników i funkcji wyjść.

W programie *Synchroniczne* zaimplementowano podany algorytm uwzględniając:

Ad 1. Spośród możliwych sposobów opisu działania automatu: wykresem czasowym, grafem przejść, tablicą (siatką) przejść/wyjść, przyjęto opis w postaci siatki przejść/wyjść.

Ad 2. Celem zmniejszenia liczby elementów pamięci potrzebnych do realizacji układu oraz uproszczenia struktury dla Bkwe należy maksymalnie uprościć siatkę przejść/wyjść. Algorytm redukcji siatki przejść/wyjść stanowi kluczową część algorytmu syntezy i jest przedstawiony blokowo na rys. 2. Do określenia minimalnej liczby maksymalnych grup stanów zgodnych i redukcji kilku stanów do jednego wykorzystany jest algorytm

znajdowania maksymalnej kliki w grafie (największego pełnego podgrafu odpowiadającego grupie stanów połączonych „każdy z każdym"). Maksymalna klika (grupa stanów) zostaje zastąpiona stanem zredukowanym (w siatce: wierszem zredukowanym). Wyniki redukcji zależą od kolejności redukowania stanów. W algorytmie za każdym razem wyszukiwana jest największa możliwa do znalezienia klika. Decyduje to o sposobie redukcji siatki programu. Rozpoczynanie redukcji od grupy stanów tworzących maksymalną klikę nie gwarantuje, że liczba stanów po redukcji jest minimalna.



Rys. 2. Algorytm redukcji siatki przejść/wyjść
Fig. 2. Algorithm of reduction of transition/output map

Ad 3. Ponieważ stany automatu synchronicznego zmieniają się w takt sygnału zegarowego i w czasie równym okresowi tego sygnału stan układu musi się ustabilizować (układ musi nadążać za zmianami wejść), to problem wyścigu należy pominąć. Można więc wybrać dowolny sposób kodowania stanów. Praktycznie trudno przewidzieć, jaki sposób kodowania będzie najbardziej korzystny. W programie ***Synchroniczne*** przyjęto kod Graya do kodowania stanów automatu.

Ad 4. Proces określenia binarnej siatki stanów jest zadaniem prostym i sprowadza się do zamiany dziesiętnych indeksów stanów na indeksy binarne (dwójkowe).

Ad 5. Funkcje wzbudzeń przerzutników określa się w oparciu o siatki binarne dla poszczególnych elementów pamięci w sposób zależny od rodzaju przerzutnika:

- Dla wejścia informacyjnego przerzutnika D siatka jest powtórzeniem siatki Q^{n+1} tego przerzutnika.
- Dla wejścia informacyjnego przerzutnika T siatkę określa się w oparciu o siatkę stanów Q^{n+1} tego przerzutnika, zmieniając na przeciwne wartości w tej części siatki, w której $Q^n = 1$.
- Dla wejść przerzutnika JK siatki określa się w oparciu o siatkę Q^{n+1} tego przerzutnika:
  - wpisując w części siatki dla J, w której $Q^n = 1$ wartości Φ oraz w części tej siatki, w której $Q^n = 0$, wartości zgodne z tą częścią siatki dla Q^{n+1},
  - wpisując w części siatki dla K, w której $Q^n = 0$ wartości Φ oraz w części tej siatki, w której $Q^n = 1$ wartości przeciwne do zawartych w siatce dla Q^{n+1}.

Ad 6. W celu wyznaczenia funkcji wyjść realizowanych w BKwy na podstawie binarnych siatek przejść/wyjść określane są:

- dla struktury Moore'a - tablica zależności sygnałów wyjściowych od sygnałów stanu,
- dla układu o strukturze Mealy'ego - siatki zależności sygnałów wyjściowych od sygnałów stanu i sygnałów wejściowych.

Ad 7. W celu określenia minimalnych wyrażeń opisujących układ, program *Synchroniczne* umożliwia wymianę danych zawierających wyniki rozwiązania z programami, przeznaczonymi do komputerowego określania minimalnych wyrażeń.

Synteza liczników równoległych metodą tablic kolejnych stanów

Dla licznika równoległego można przyjąć szczególny przypadek struktury Moore'a podanej na rys. 1, gdzie wyjścia licznika odpowiadają wyjściom Bloku Pamięci: Q_1 .. Q_m, a wejścia programujące licznika odpowiadają wejściom układu synchronicznego: x_1 .. x_n .

Synteza sprowadza się wtedy do zaprojektowania Bloku Kombinacyjnego wejściowego sterującego wejściami informacyjnymi przerzutników wybranych do realizacji licznika.

W celu ułatwienia określenia rozwiązania Bloku Kombinacyjnego wejściowego opisuje się go taką tablicą zależności, w której kombinacje wartości argumentów funkcji: x_1, .., x_n, Q_1, .., Q_m zawarte w lewej części tablicy są zapisane w kolejności odpowiadającej kolejnym stanom licznika Q_1 .. Q_m dla różnych wartości wejść programujących: x_1 .. x_n. Taka tablica nazywana jest „tablicą kolejnych stanów".

Wartości wejść informacyjnych w prawej części tablicy można łatwo określić na podstawie porównania stanów licznika: aktualnego i następnego, z uwzględnieniem tablicy wzbudzeń przerzutnika.

3. Opis programów wspomagających nauczanie w zakresie układów synchronicznych

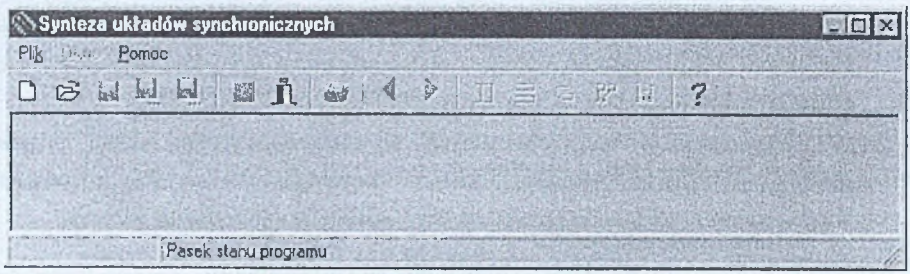
3.1. Opis programu *Synchroniczne*

W programie ***Synchroniczne*** został zaimplementowany algorytm syntezy układów synchronicznych oparty o siatki przejść/wyjść z wizualizacją kolejnych etapów. Pozwala to na prezentację przebiegu syntezy dla dowolnych danych wprowadzonych przez użytkownika.

Program[1] zrealizowany został [3] w postaci aplikacji dla środowiska Windows 95/98 z wykorzystaniem narzędzia programistycznego systemu Delphi 3.0 firmy Borland z językiem programowania Object Pascal. Możliwość jednoczesnej obserwacji wszystkich kroków syntezy została zrealizowana poprzez wykorzystanie techniki *MDI - Multiple Document Interface* - dostępnej w systemie Windows.

Interfejs użytkownika

Okno główne programu przedstawia rys. 3. Można w nim odnaleźć elementy typowe występujące standardowo dla w wielu innych aplikacji systemu Windows.



Rys. 3. Okno główne programu
Fig. 3. Main window of the program

Pasek narzędzi umożliwia szybki dostęp do wszystkich funkcji programu:

 - przyciski odpowiadające poleceniom w menu *Plik*

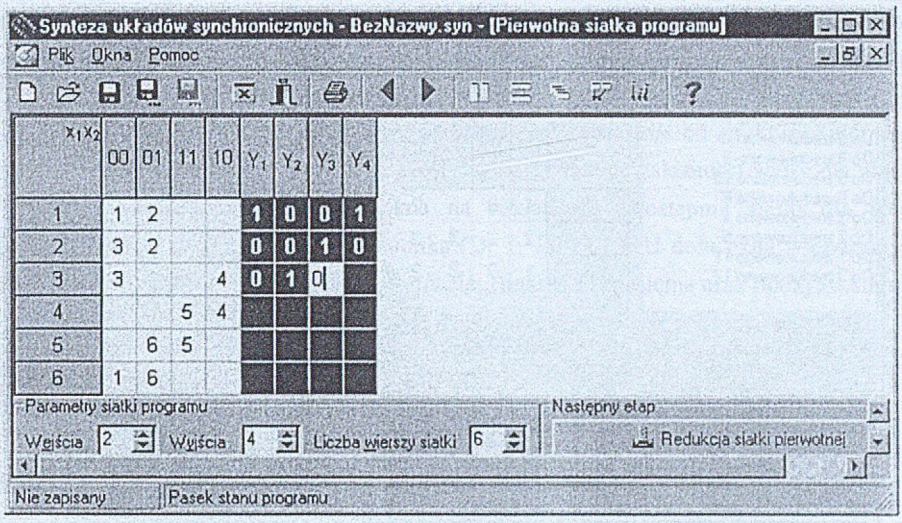
 - przyciski odpowiadające poleceniom w menu *Okna*

 - wyświetla indeks haseł dostępnych w pliku pomocy.

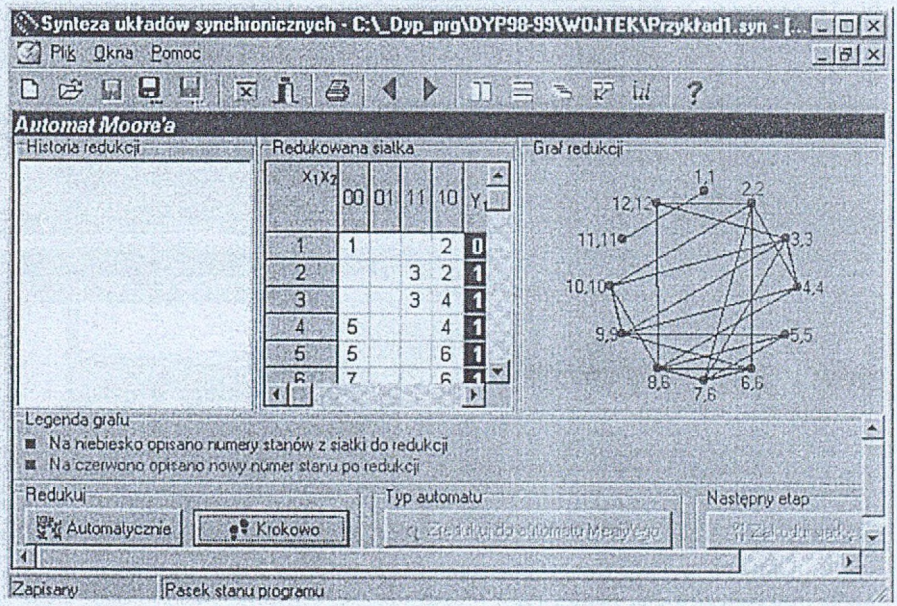
Proces syntezy prowadzony jest w oknach-dzieciach MDI wyposażonych w mechanizm nie pozwalający przejść do następnego etapu syntezy przed zakończeniem bieżącego etapu. Rys. 4 przedstawia okno podczas tworzenia siatki przejść/wyjść. Program sprawdza, czy

[1] Program *Synchr-s* dostępny w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl

wprowadzane wartości są poprawne. W przypadku błędu usuwana jest błędna wartość i generowany odpowiedni komunikat. Rys. 5 przedstawia okno po rozpoczęciu redukcji stanów (redukcji wierszy siatki).



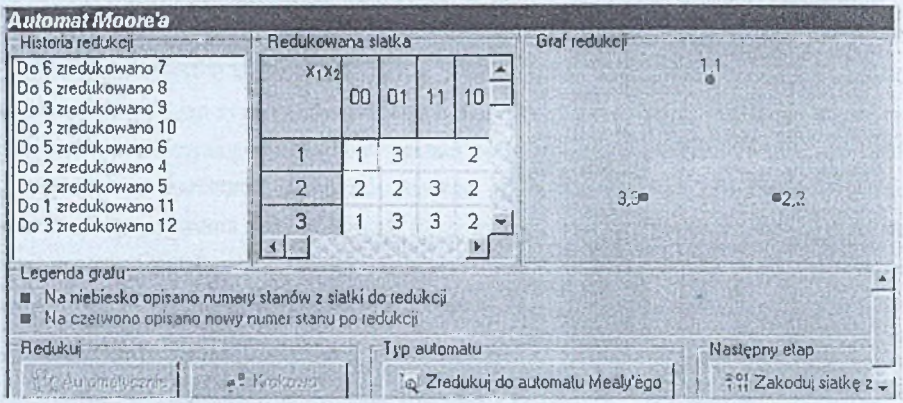
Rys. 4. Okno podczas tworzenia pierwotnej siatki programu
Fig. 4 . Window shown during the primary flow map generation



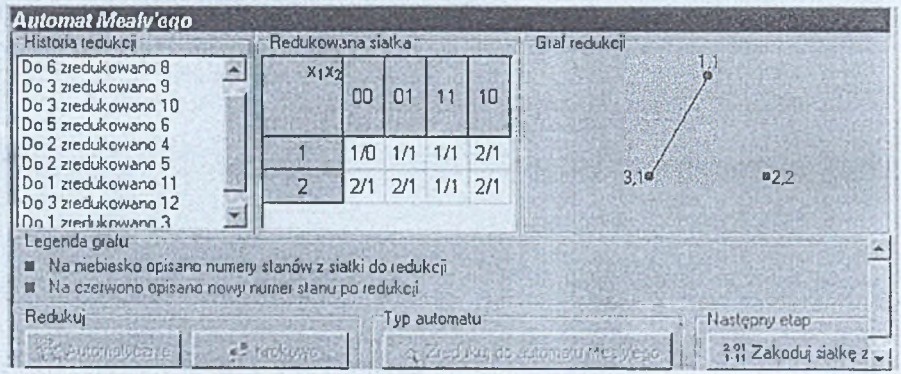
Rys. 5. Okno MDI po rozpoczęciu redukcji
Fig. 5. MDI Window after start of reduction

Wyniki redukcji siatki udostępniane są: „Krokowo" po kolejnych etapach lub „Automatycznie" tylko wyniki końcowe. Po każdym kroku uzupełniana jest lista „Historia redukcji" (z opisem redukowanych stanów), uaktualniany jest graf redukcji obrazujący możliwe złączenia redukowanych stanów oraz siatka przejść/wyjść.

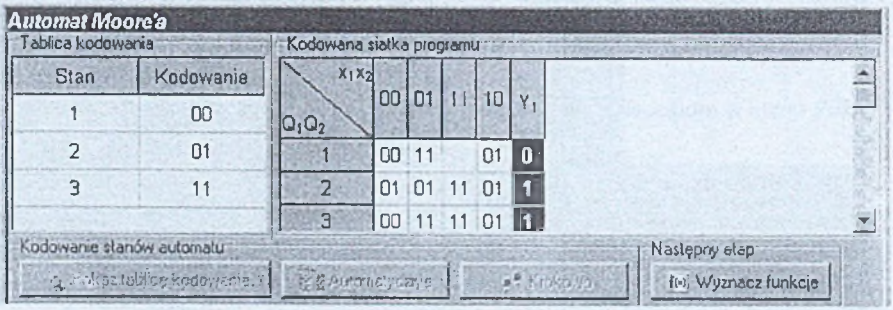
a)



b)



c)

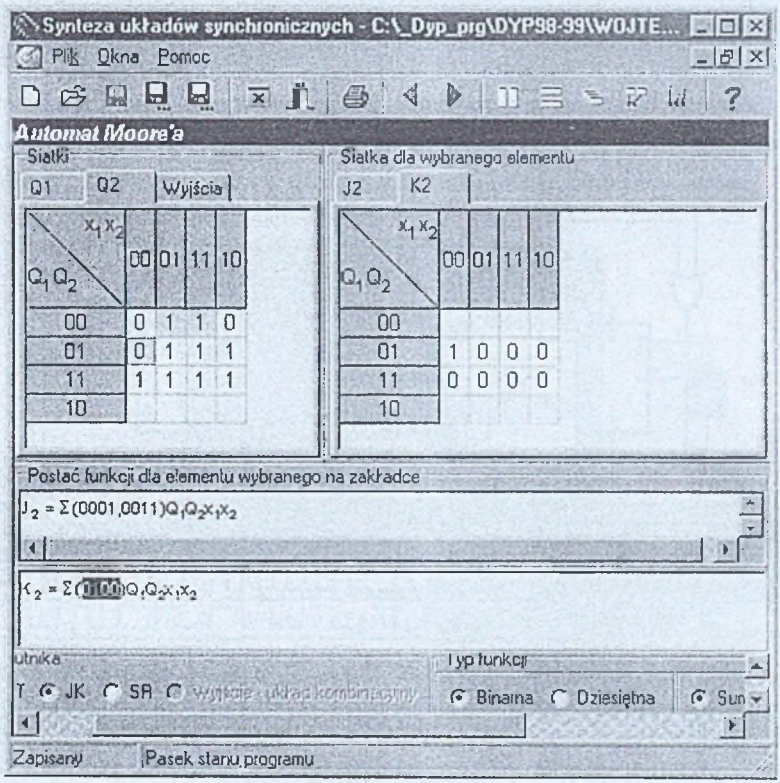


Rys. 6. Okno po redukcji stanów dla struktury Moore'a (a), Maely'ego (b), po zakodowaniu stanów (c)

Fig. 6. Window after reduction states for Moore's machine(a), Mealy's machine (b), after encoding states (c)

Po zakończeniu redukcji siatki dla automatu Moore'a (rys. 6a) uaktywniają się przyciski: „Zredukuj do automatu Mealy'ego" oraz „Zakoduj siatkę zredukowaną". Dają one możliwość dalszej redukcji siatki dla automatu Mealy'ego (rys. 6b) albo przejścia do etapu kodowania siatki zredukowanej dla automatu w strukturze Moore'a (rys. 6c).

Przycisk „Wyznacz funkcje" umożliwia przejście do etapu wyznaczania funkcji wejść informacyjnych przerzutników i funkcji wyjść. Lewa część okna (rys. 7) na kolejnych zakładkach udostępnia siatki dla elementów pamięci oraz (zależnie od struktury automatu): siatki dla wyjść w układzie o strukturze Mealy'ego albo tablicę zależności wyjść dla układu o strukturze Moore'a. Prawa część okna na zakładkach udostępnia siatki dla wejść informacyjnych wybranego typu przerzutnika (D, T, JK i SR). U dołu okna widoczna jest kanoniczna postać funkcji w wybranym formacie: Binarna / Dziesiętna oraz Iloczyn / Suma.



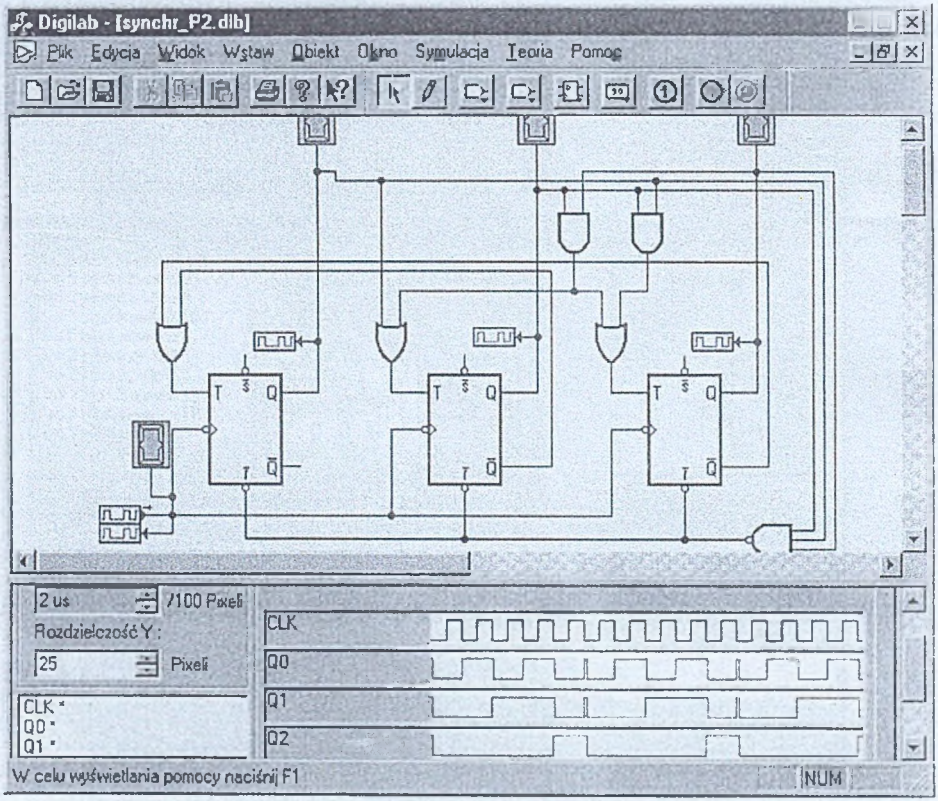
Rys. 7. Okno funkcji wejść przerzutników i funkcji wyjść
Fig. 7. Window of flip-flop excitation inputs and output functions

Wyniki można eksportować do pliku z aktualnie wybranej zakładki w określonej przez użytkownika postaci rozwiązania.

3.2. Opis programu *Digilab*

Program ***Digilab*** uwzględnia: prezentację przebiegu algorytmu syntezy liczników równoległych metodą „tablic kolejnych stanów", tworzenie schematów układów cyfrowych oraz symulację dowolnych układów cyfrowych realizowanych zarówno w „strukturze szytej" jak i w wersji mikroprogramowanej.

Program[1] zrealizowany został [2] w postaci aplikacji dla środowiska 32-bitowego systemu operacyjnego Windows 95. Do zaimplementowania programu wybrano język C++ oraz środowisko programowania Visual C++ w wersji 5.0 z użyciem bibliotek MFC (Microsoft Foundation Class) w wersji 4.21.



Rys. 8. Okno główne programu
Fig. 8. Main window of the program

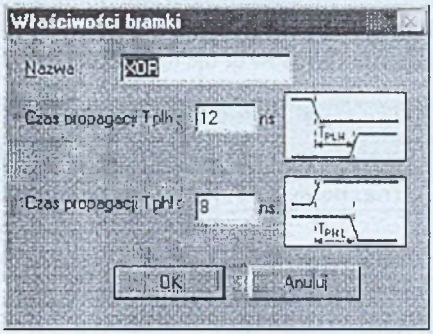
Budowę modelu symulacji układów cyfrowych oparto na liniowej liście zdarzeń posortowanej względem parametru „czas". Obiekt symulowany wstawia do listy zdarzenia, które mają zajść po upływie określonego czasu. Upływ czasu jest symulowany na podstawie

[1] Program *Sym-licz* dostępny w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl

sygnału zwrotnego generowanego po wykonaniu zdarzenia do obiektu, który wstawił to zdarzenie do listy.

Okno główne programu przedstawia rys. 8. Można w nim wyróżnić kolejno od góry: Pasek tytułu, Menu główne, Pasek narzędzi, Obszar roboczy, Okno wykresów czasowych, Pasek statusu. Pasek narzędzi składa się z dwóch części: typowego Paska głównego i Rozwijanych pasków wyboru elementów:

Druga część udostępnia kolejno od lewej: narzędzie wyboru, narzędzie rysowania linii służące do łączenia elementów składowych układu, bramki 1-, 2- i 3-we:, przerzutniki, elementy 74xx (komutatory, liczniki, rejestry i pamięci RAM) oraz dodatkowe elementy potrzebne do symulacji np.: przełącznik, definiowany przebieg, element rejestracji wykresów czasowych, próbnik logiczny. Utworzony schemat można modyfikować: wstawiać i usuwać elementy, zmieniać ich położenie (również kątowe), zmieniać połączenia między elementami, a także ich właściwości. Właściwości elementów można ustawić za pomocą odpowiedniego dla danego typu elementów okna. Na rys. 9 przedstawiono okno ustawiania właściwości bramki XOR.

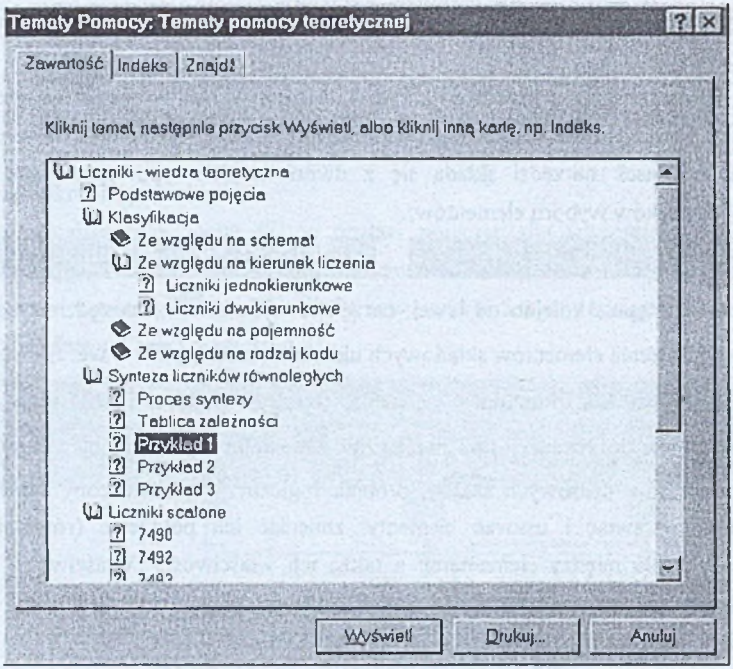


Rys. 9. Okno ustawiania właściwości bramki
Fig. 9. Window of gate property setting

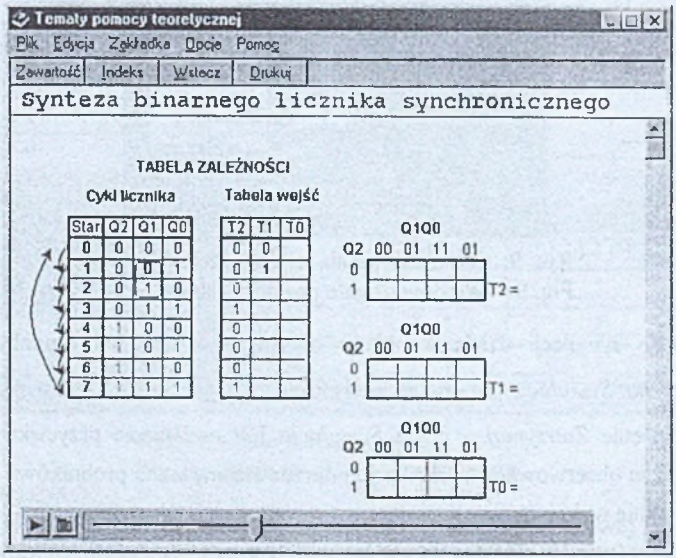
Rozpoczęcie symulacji działania układu opisanego schematem umożliwia polecenie *Uruchom* z menu *Symulacja* lub przez naciśnięcie przycisku. Zakończenie symulacji umożliwia polecenie *Zatrzymaj* z menu *Symulacja* lub naciśnięcie przycisku. Wyniki z symulacji można obserwować na schemacie poprzez zmianę stanu próbników i na wykresach czasowych w oknie wykresów.

Opcja Teoria (rys. 10) udostępnia wiedzę teoretyczną w zakresie liczników i prezentację etapów syntezy liczników równoległych metodą tablic kolejnych stanów.

[1] Program *Sym-licz* dostępny w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl



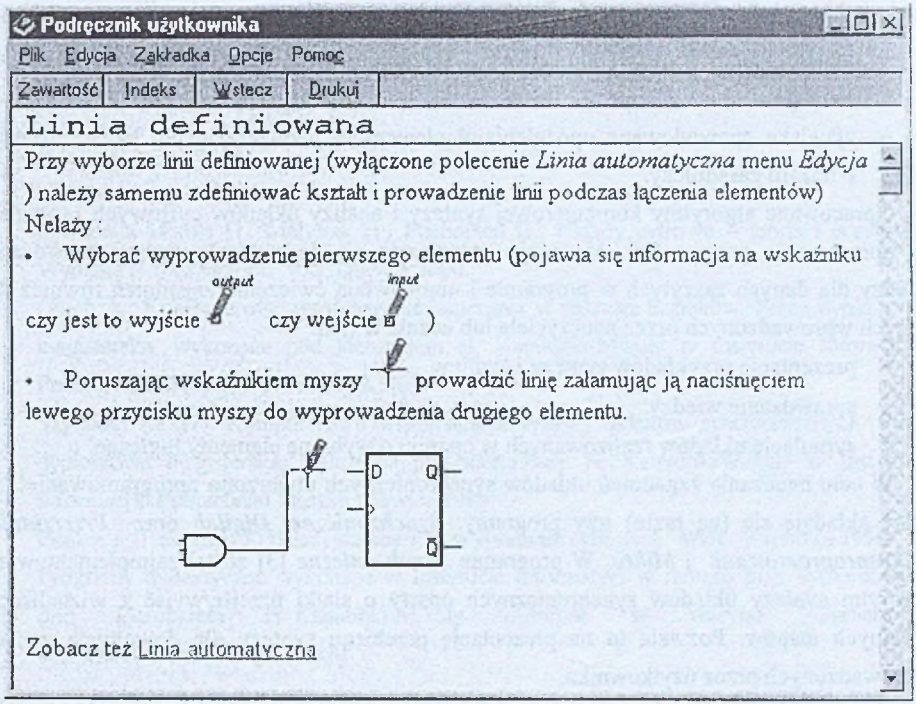
Rys. 10. Okno opcji TEORIA
Fig. 10. Window of the option: THEORY



Rys. 11. Okno prezentacji etapów syntezy licznika równoległego
Fig. 11. Window of the parallel counter synthesis presentation

Rys. 11 przedstawia przykładowe okno podczas symulacji kolejnych etapów syntezy licznika równoległego. Narzędzie: widoczne w dolnej części okna uaktywnia proces symulacji przebiegu syntezy. Wybranie pozycji: **Symulacja działania** umożliwia prezentację schematu i symulację pracy projektowanego układu.

Opcja Pomoc (rys. 12) udostępnia informacje na temat obsługi programu.



Rys. 12. Przykładowe okno Pomocy
Fig. 12. Sample Help window

4. Podsumowanie

W przypadku zastosowania sieci komputerowych w nauczaniu, w tym nauczaniu na odległość, ważną grupę narzędzi programowych stanowią programy wspomagające nauczanie. Warunkiem powodzenia procesu edukacji na odległość jest uwzględnienie w oprogramowaniu dydaktycznym możliwości interaktywnego uczenia się, symulacji procesów i przebiegu projektowania oraz atrakcyjnej interaktywnej formy motywującej do samokształcenia.

W Instytucie Informatyki Politechniki Śląskiej prowadzone są prace w zakresie komputerowego wspomagania nauczania zagadnień w ramach przedmiotów: teoria automatów cyfrowych i podstawy techniki cyfrowej. Tworzone oprogramowanie obejmuje następujące zagadnienia:

- podstawy teoretyczne opisu układów cyfrowych,
- synteza układów cyfrowych różnymi metodami,
- analiza układów cyfrowych,
- minimalizacja wyrażeń logicznych z uwzględnieniem różnych metod i zagadnień hazardu,
- zjawiska spowodowane opóźnieniami elementów przełączających: hazard, wyścig i hazard zasadniczy.

Opracowane algorytmy komputerowej syntezy i analizy układów cyfrowych pozwalają na opracowanie programów[1], które nie ograniczają się do prezentowania i sprawdzania wiedzy dla danych zaszytych w programie i umożliwiają ćwiczenie zagadnień również dla danych wprowadzonych przez nauczyciela lub ucznia w tym:

- prezentację przykładów syntezy i analizy,
- sprawdzanie wiedzy,
- symulację układów realizowanych w oparciu o wybrane elementy logiczne.

W celu nauczania zagadnień układów synchronicznych utworzono oprogramowanie[2], na które składają się (na razie) trzy programy: ***Synchroniczne***, ***Digilab*** oraz ***Przerzutniki Mikroprogramowane*** i ***Mikro***. W programie ***Synchroniczne*** [3] został zaimplementowany algorytm syntezy układów synchronicznych oparty o siatki przejść/wyjść z wizualizacją kolejnych etapów. Pozwala to na prezentację przebiegu syntezy dla dowolnych danych wprowadzonych przez użytkownika.

Program ***Digilab*** [2] uwzględnia: prezentację przebiegu algorytmu syntezy liczników równoległych metodą „tablic kolejnych stanów", tworzenie schematów układów cyfrowych oraz symulację dowolnych układów cyfrowych realizowanych zarówno w „strukturze szytej", jak i w wersji mikroprogramowanej. Możliwość symulacji układów jest ważnym elementem nie tylko służącym do prezentacji działania układu, ale również do weryfikacji wyników syntezy układów cyfrowych.

Program ***Przerzutniki*** (nie omawiany w niniejszym referacie) przeznaczony jest do nauczania zagadnień przerzutników (synchronicznych i asynchronicznych). Uwzględnia on zagadnienia teoretyczne, przykłady budowy, działania i opisu różnych typów przerzutników

[1] Programy dostępne w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl

[2] Programy dostępne pod nazwą odpowiednio: *Synchr-s, Sym-licz, Prze-nau, Mikroprogramowane, Mikro* w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl

oraz możliwość sprawdzenia nabytej wiedzy poprzez samodzielne rozwiązywanie zadań i odpowiedzi na pytania testowe. Programy ***Mikroprogramowane*** i ***Mikro*** służą do nauczania i syntezy układów synchronicznych realizowanych w wersji mikroprogramowanej.

Prowadzone są dalsze prace celem uzupełnienia biblioteki programów wspomagających nauczanie w zakresie syntezy i analizy układów cyfrowych. Odnośnie do układów synchronicznych tworzone oprogramowanie ma na celu uzupełnić oprogramowanie w zakresie: możliwości szczegółowego ćwiczenia zasad obowiązujących w procesie syntezy tych układów, sposobów korekcji błędnych stanów oraz sprawdzania nabytej wiedzy.

LITERATURA

1. Kamionka-Mikuła H., Małysiak H., Pochopień B.: Układy cyfrowe – teoria i przykłady. Wydanie II-rozszerzone, PJS, Gliwice 2000.
2. Głuch A.: Komputerowe wspomaganie nauczania w zakresie liczników. Praca dyplomowa magisterska, wykonana pod kierunkiem H. Kamionki-Mikuły w Instytucie Informatyki Politechniki Śląskiej, Gliwice 1999.
3. Szymkiewicz W.: Komputerowe wspomaganie syntezy układów synchronicznych. Praca dyplomowa magisterska, wykonana pod kierunkiem H. Kamionki-Mikuły w Instytucie Informatyki Politechniki Śląskiej, Gliwice 1999.
4. Pieńkos J., Turczyński J.: Układy scalone TTL w systemach cyfrowych. WkiŁ. Warszawa 1986.
5. Programy dydaktyczne wykonane w Instytucie Informatyki w ramach prac dyplomowych pod kierunkiem H. Kamionki-Mikuły dostępne w witrynie internetowej: http://zmitac.iinf.polsl.gliwice.pl

Recenzent: Dr inż. Ryszard Winiarczyk

Wpłynęło do Redakcji 15 marca 2001 r.

Abstract

Computer networks are used more and more often for teaching, including remote teaching technique. In this case, structure of programs for computer aided teaching is very important.

The didactic programs make possible to teach in different ways dependent on the considered topics. In the case of teaching the problems connected with digital circuits

synthesis there is a need of presenting theoretical knowledge, demonstrating examples of synthesis, simulation of circuits and checking the new knowledge of students.

To make possible to cover a wide area of exercises, to present examples and to check the new knowledge the software should include proper algorithms for:

- computer aided synthesis of logical circuits,
- simulations of logical circuits
- computer aided performing of exercises and knowledge checking.

In the paper the methods used in computer aided teaching of synchronous digital circuits are presented. These methods are included in the following programs:

Synchroniczne – this program [3] presents subsequent stages of synchronous switching circuit synthesis with the use of transition maps (Fig. 3 – Fig. 7). Program accepts any proper data input by the user

Digilab - this program [2] makes possible the visual presentation of parallel counter synthesis stages (Fig. 11), generation of diagrams of digital circuits (Fig. 8), simulation of counters and other digital circuits implemented both as hard wired logic, as well as in microprogrammable version.