Jacek TKACZ, Marian ADAMSKI
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI

Projektowanie sekwencyjnych układów cyfrowych z wykorzystaniem logiki sekwentów Gentzena

dr inż. Jacek TKACZ

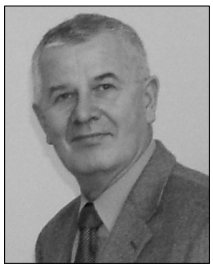
Absolwent Uniwersytetu Zielonogórskiego. Od 2001 roku pracownik zatrudniony na stanowisku asystenta w Zakładzie Inżynierii Komputerowej. Prowadzi badania nad symboliczną metodą dowodzenia twierdzeń i jej praktycznymi zastosowaniami w informatyce oraz elektronice. Interesuje się nowoczesnymi technologiami projektowania i wytwarzania aplikacji, w tym aplikacji mobilnych. W latach 1997-2005 zajmował się projektowaniem i rozwojem oprogramowania dla polskich bibliotek (PROLIB).

e-mail: J.Tkacz@iie.uz.zgora.pl

prof. dr hab. inż. Marian ADAMSKI

Dyrektor Instytutu Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zainteresowania badawcze obejmują projektowanie systemów cyfrowych realizowanych w postaci mikrosystemów cyfrowych oraz formalnych metod programowania sterowników logicznych. Członek IEEE, IEE, ACM, Polskiego Towarzystwa Elektrotechniki Teoretycznej i Stosowanej oraz Polskiego Towarzystwa Informatycznego.

e-mail: M.Adamski@iie.uz.zgora.pl



Streszczenie

Artykuł jest ilustracją możliwości zastosowania komputerowego wnioskowania symbolicznego w projektowaniu układów cyfrowych, a w tym do rozwiązywania skomplikowanych problemów logicznych. Wykorzystując przykład sieci działań zaczerpnięty z literatury, przedstawiono sposób uzyskiwania uproszczonego opisu bloku kombinacyjnego automatu cyfrowego na podstawie jego specyfikacji regułowej. Metoda polega na zastąpieniu sekwentami tablicy przejść-wyjść automatu i przeprowadzeniu syntezy logicznej metodą komputerowego wnioskowania.

Słowa kluczowe: Logika Gentzena, sekwent, wnioskowanie symboliczne, synteza, symulacja logiczna, VHDL, Verilog.

Synthesis of sequence digital circuits by means of using Gentzen reasoning method

Abstract

There is presented a new idea of an application of Gentzen logic symbolic reasoning for solving some combinational problems in the sequence circuit design in this paper. There is shown possibility of generating simplified behavioral description of combinatorial block of synchronous circuit based on flow chart example. The method rests on a replacement of the classical transition table by the sequents which describe relations between inputs and outputs of combinatorial block of sequential digital circuit. The group of rules of the type "if-then" is produced from sequents normalization process. Rules are equivalent to conjunction or disjunction form of logical functions. The designed and implemented reasoning tool is able to make a symbolic synthesis and logical simulation. By applying sequent specification conversion into hardware description languages there is opened the way for synthesis and simulations with use of commercial CAD tools. The collaboration with university CAD systems designed for synthesis digital circuit is also possible by transformation symbolic specification into ESPRESSO format.

Keywords: Gentzen logic, sequent, symbolic reasoning, synthesis, logic simulation, VHDL, Verilog.

1. Wstęp

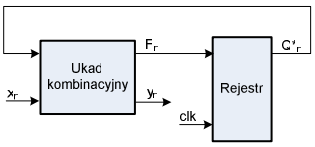
Dotychczasowe badania prowadzone nad komputerowym wnioskowaniem symbolicznym uwidoczniły jego przydatność do rozwiązywania różnorodnych problemów kombinatorycznych metodami formalnymi. Wykorzystanie rachunku sekwentów w projektowaniu układów cyfrowych wraz z obszerną bibliografią zostało przedstawione w monografii [1]. Monografia [1] skupiała się wokół zagadnień projektowania układów cyfrowych metodą systematyczną i dotyczyła głównie teorii automatycznej syntezy.

W chwili obecnej system Gentzena [4][6] znajduje coraz większe zastosowanie w rozwiązywaniu trudnych teoretycznych problemów logicznych metodami komputerowymi [1][7][8][11][12]. Największą zaletą proponowanej metody jest jej elastyczność, nieoceniona zwłaszcza w badaniach naukowych.

2. Synteza sekwencyjnych układów cyfrowych opisanych siecią działań

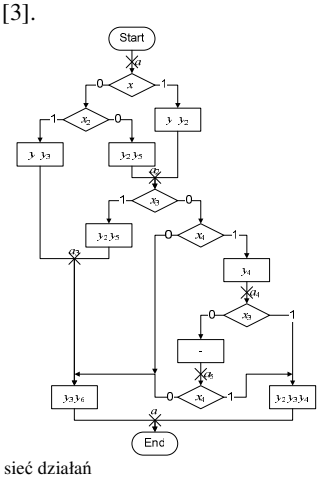
W badaniach przyjęto założenie, że synteza układów sekwencyjnych realizowana jest z wykorzystaniem układów rejestrowych. Układ rejestrowy składa się z układu kombinacyjnego i rejestru będącego zbiorem przerzutników (Rys. 1). Jest on realizacją sekwencyjnego bloku funkcjonalnego lub automatu sterującego, sekwencyjnego albo współbieżnego.

Specyfikacja sekwencyjnego układu cyfrowego polega na zastąpieniu funkcji przejść i funkcji wyjść regułami przedstawionymi w formie sekwentów [1][2][3]. Projektując układ sterujący wykorzystujący rejestr opierając się o jeden ze znanych typów automatów (Moore'a lub Mealey'ego) należy w bloku kombinacyjnym wyznaczającym funkcje wzbudzeń uwzględnić wejścia sterujące x_n oraz wyjścia y_n (Rys. 1).



Rys. 1. Układ sekwencyjny z wykorzystaniem rejestru
Fig. 1. Sequence digital circuits with register

Jako przykład współpracy systemu wnioskującego z językami HDL zrealizowany zostanie układ sekwencyjny przedstawiony za pomocą oznakowanej sieci działań [5] (Rys. 2). Zastosowanie sieci działań do opisu funkcjonowania układu sekwencyjnego opisano w literaturze [3].



Rys. 2. Oznakowana sieć działań
Fig.2. Signed flow chart

Oznakowanie (*a1..a5*) zostało wprowadzone w celu wskazania miejsc odpowiadających stanom w układzie sekwencyjnym. Wyjścia układu będą generowane przy przejściach i zależeć będą od warunku przejścia, więc rozpatrywany układ będzie automatem Mealy'go [9][10]. Celem przykładu jest pokazanie możliwości wstępnej weryfikacji formalnej metod opartych na intuicji, które następnie są uszczegóławiane i algorytmizowane, będąc podstawą do konstrukcji nowatorskich systemów CAD [2][3].

Sieć działań można przedstawić w postaci tabeli przejść-wyjść, a następnie tabelę zapisać w postaci sekwentów i poddać normalizacji. W przypadku zastosowania logiki sekwentów tworzenie tablicy może zostać pominięte, gdyż sekwenty można utworzyć bezpośrednio z sieci działań według następujących schematów (1) i (2):

$$\begin{array}{c} \textit{stan aktualny} * \textit{warunek} \mid\!- \textit{stan następny} \\ \textit{stan aktualny} \mid\!- \textit{wyjścia} \end{array} \quad (1)$$

lub

$$\textit{stan aktualny} * \textit{warunek} \mid\!- \textit{stan następny} * \textit{wyjścia} \quad (2)$$

W praktyce, dla sieci działań reguły przejść i wyjść zapisuje się łącznie w postaci blokowej (3):

$$\textit{stan aktualny} \mid\!- \prod(\textit{warunek} \rightarrow (\textit{stan następny} * \textit{wyjścia})) \quad (3)$$

Stan aktualny będzie iloczynem formuł zbudowanych na podstawie przyjętej tabeli kodowania stanów (Tab. 1).

Tab. 1. Tabela kodowania stanów
Tab. 1. Table of states coding

Stan	Q1	Q2	Q3
a1	0	0	0
a2	0	0	1
a3	0	1	0
a4	0	1	1
a5	1	0	0

Stany następne należy opisać wyprowadzając je z równania przerzutnika. W omawianym przykładzie zastosowany zostanie przerzutnik D, dla którego równanie dla stanu następnego ma postać (4):

$$Q' = D \quad (4)$$

Budując sekwenty według zaproponowanej metody uzyska się reprezentację sekwentową dla rozpatrywanej sieci działań (Tab. 2). Niezbędne jest określenie typu formuł (Wejściowe/Wyjściowe) dla uporządkowania i składania sekwentów znormalizowanych w złożone opisy. Brak definicji typów formuł uniemożliwi transformację na języki opisu sprzętu, a także na format ESPRESSO.

Tab. 2. Reprezentacja sekwentowa sieci działań
Tab. 2. Sequents representation of flow chart

Specyfikacja układu

```
IN:Q1,Q2,Q3,x1,x2,x3,x4;
OUT:y1,y2,y3,y4,y5,y6,D1,D2,D3;
#----------------------------------
/Q1*/Q2*/Q3|-(x1->/D1*/D2*D3*y1*y2)*
(/x1*/x2->/D1*/D2*D3*y2*y5)*
(/x1*x2->/D1*D2*/D3*y1*y3);
#----------------------------------
/Q1*/Q2*Q3|-(x3->/D1*D2*/D3*y2*y5)*
(/x3*/x4->/D1*/D2*/D3*y3*y6)*
(/x3*x4->/D1*D2*D3*y4);
#----------------------------------
/Q1*Q2*/Q3|-/D1*/D2*/D3*y3*y6;
#----------------------------------
/Q1*Q2*Q3|-(x3->/D1*/D2*/D3*y2*y3*y4)*
(/x3->D1*/D2*/D3);
#----------------------------------
Q1*/Q2*/Q3|-(x4->/D1*/D2*/D3*y2*y3*y4)*
(/x4->/D1*/D2*/D3*y3*y6);
```

W wyniku normalizacji specyfikacji wyznaczone zostaną sekwenty definiujące równania wzbudzeń przerzutników oraz ich afirmacje, a także równania dla poszczególnych wyjść układu. Dzięki wyznaczeniu równań wzbudzeń przerzutników, a także ich afirmacji projektant może wybrać lepsze rozwiązanie poprzez wybór tych funkcji, które składają się z mniejszej liczby termów lub termów o mniejszej liczbie zmiennych. W przypadku realizacji układu sekwencyjnego konieczne jest dołączenie do wygenerowanego układu rejestru (Rys. 1) analogicznie jak przy projektowaniu układów sekwencyjnych z wykorzystaniem tabeli przejść-wyjść.

W przypadku układów kombinacyjnych wygenerowany układ można bezpośrednio syntezować do struktury FPGA, bez definiowania bloku rejestrowego.

3. Symulacja i synteza z wykorzystaniem komercyjnych narzędzi CAD.

W wyniku normalizacji opisu układu w postaci sekwentowej i wprowadzeniu spójników logicznych powstałe złożone sekwenty odpowiadające funkcjom realizowanego układu stosunkowo łatwo można przekształcić do postaci języka VHDL. Nadmiarowości w postaci wyznaczonych F^0 lub w specyficznych przypadkach wartości nieokreślonych F^- należy pominąć.

Dla sieci działań (Rys. 2), specyfikacji (Tab. 2) oraz po odrzuceniu ewentualnych elementów nadmiarowych (F^0 i F^-), w wyniku normalizacji i wprowadzania spójników logicznych, powstały sekwenty reprezentujące poszczególne funkcje układu (Tab. 3).

Tab. 3. Znormalizowany opis układu
Tab. 3. Normalized description of digital circuit

Sekwenty dla wzbudzeń przerzutników oraz wyjść układu

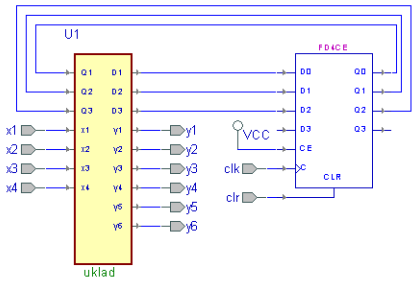
```
/Q1*Q2*Q3*/x3|-D1;
/Q1*/Q2*/Q3*/x1*x2 + /Q1*/Q2*Q3*x3 +
/Q1*/Q2*Q3*x4|-D2;
/Q1*/Q2*/Q3* x1 + /Q1*/Q2*Q3*/x3*x4 +
/Q1*/Q2*/Q3*/x2|-D3;

/Q1*/Q2*/Q3*x1 + /Q1*/Q2*/Q3*x2|-y1;
/Q1*/Q2*/Q3*x1 + Q1*/Q2*/Q3*x4 + /Q1*/Q2*/Q3*/x2 +
/Q1*Q3*x3|-y2;
/Q1*/Q2*Q3*/x3*/x4 + /Q1*Q2*/Q3 + /Q1*/Q3*/x1*x2 +
/Q1*Q2*x3 + Q1*/Q2*/Q3|-y3;
/Q1*/Q2*Q3*/x3*x4 + /Q1*Q2*Q3*x3 + Q1*/Q2*/Q3*x4 |-y4;
/Q1*/Q2*/Q3*/x1*/x2 + /Q1*/Q2*Q3*x3 |-y5;
/Q1*/Q2*Q3*/x3*/x4 + /Q1*Q2*/Q3 + Q1*/Q2*/Q3*/x4|-y6;
```

Opracowane narzędzie wnioskujące umożliwia automatyczną transformację funkcji z postaci sekwentowej na języki opisu sprzętu VHDL, Verilog oraz na format stosowany w rozwiązaniach uniwersyteckich ESPRESSO (np. DEMAIN) [9]. Wybór języka opisu sprzętu jest uzależniony od indywidualnych preferencji inżyniera-projektanta.

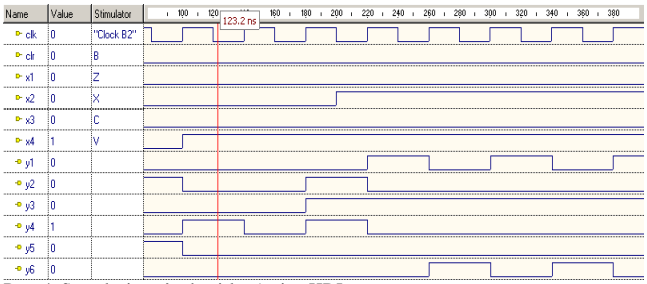
Opis behawioralny rejestrowego bloku funkcyjnego przedstawia się w postaci zbioru sekwentów, które po normalizacji automatycznie przedstawiane są w formie akceptowalnej przez język opisu sprzętu. Przed symulacją behawioralną można także dokonać symulacji symbolicznej, podobnie jak w przypadku układów kombinacyjnych [12].

Do wykonania symulacji projektowanego układu wykorzystano narzędzie ActiveHDL v7.3 firmy ALDEC. W celu lepszego zobrazowania sposobu projektowania układu zdecydowano się na przedstawienie projektu w formie diagramu blokowego (Rys. 3), gdzie element „U1-uklad" jest opisany wygenerowanym w języku VHDL blokiem funkcyjnym, a element FD4CE rejestrem złożonym z czterech przerzutników typu D.



Rys. 3. Układ rejestrowy
Fig. 3. Digital circuit with register

Taki sposób projektowania układu nie jest jednak wymogiem, ponieważ specyfikacja całego układu mogła zostać przygotowana na przykład tylko w języku HDL, bez użycia diagramu blokowego. Stosując narzędzia do symulacji z pakietu Active-HDL przeprowadzono badanie poprawności funkcjonowania projektowanego układu. Na Rys. 4 przedstawiono fragment przebiegu symulacji.



Rys. 4. Symulacja w środowisku Active-HDL
Fig. 4. Simulation in Active-HDL environment

Wyniki symulacji są zgodne z założeniami zdefiniowanymi przy pomocy sieci działań. Na wygenerowanym przebiegu można zaobserwować przejście ze stanu *a1* do *a2* i wygenerowanie wyjść *y2* i *y5* dla sygnałów sterujących będących zerami. Następnie po pojawieniu się sygnału sterującego *x4* układ przechodzi do stanu a4 generując wyjście *y4*. Dalej przechodzi do stanu *a5*, nie generując żadnych sygnałów wyjściowych. Ze stanu *a5* system wraca do stanu początkowego *a1* i generuje wyjścia *y2*,*y3* i *y4*. Od momentu pojawienia się kolejnego sygnału sterującego *x2*, układ przełącza stany *a1* i *a3*, generując na przemian wyjścia *y1* i *y3* oraz *y3* i *y6*.

Po dodaniu do schematu blokowego wymaganych buforów wejściowych i wyjściowych oraz globalnego bufora dla sygnału zegara przeprowadzona została przykładowa synteza i implementacja projektowanego układu. W procesie syntezy i implementacji wykorzystany został pakiet Xilinx ISE v9.1. Jako układ docelowy wybrano posiadany układ firmy Xilinx z rodziny spartan2e. W wyniku przeprowadzonej syntezy i implementacji uzyskano plik konfiguracyjny *bitstream*, który posłużył do zaprogramowania rzeczywistego układu FPGA.

4. Analiza i symulacja logiczna

Znormalizowane klauzule odzwierciedlające prace poszczególnych przerzutników mogą być w sposób automatyczny transformowane na języki opisu sprzętu, symulowane w komfortowym środowisku projektowym, a następnie jednoznacznie odwzorowane w strukturze matrycowej rekonfigurowanego układu cyfrowego. Stosując założenie o otwartym świecie (*OWA*) można wprowadzać do znormalizowanej specyfikacji układu sekwencyjnego dodatkowe założenia i badać jego zachowanie. Analiza jest bardzo przejrzysta dla człowieka gdyż odzwierciedla naturalny tok jego myślenia. Przykładowo, chcąc zbadać jak zachowa się układ w stanie początkowym *a0* zakodowanemu jako */Q3/Q2/Q1*, gdy sygnały sterujące *x1* i *x2* przyjmą wartości „*1*", należy dodać do uprzednio znormalizowanej specyfikacji wyrażenie (5).

$$|{-}x1{*}x2{*}/Q1{*}/Q2{*}/Q3; \qquad (5)$$

W wyniku ponownej normalizacji systemem wnioskującym i uporządkowaniu wyrażeń ze względu na typy zmiennych otrzymuje się równania opisujące zachowanie układu (Tab. 4).

Tab. 4. Symulacja logiczna dla poprawnie zdefiniowanej akcji
Tab. 4. Logical simulation for correctly defined action

Wyznaczone zachowanie układu				
`|-/D1;`	`|-/D2;`	`|-D3;`	`|-y1;`	`|-y2;`

Przyjęte założenie otwartego świata spowodowało wyznaczenie wszystkich możliwych przypadków wynikających z pierwotnej specyfikacji bloku kombinacyjnego układu sekwencyjnego. Pierwszy sekwent informuje o rozpatrywanym warunku. Sekwenty *|-/D1*, *|-/D2* i *|-D3* informują o stanie *a1*, w jakim znajdzie się układ. Natomiast sekwenty *|-y1* i *|-y2* informują, jakie wyjścia układu będą aktywne. Sygnał *x2* został pominięty, gdyż nie miał on znaczenia przy opuszczaniu stanu *a0*, gdy *x1* miał wartość „*1*".

Ciekawym przykładem będzie zdefiniowanie sytuacji nieokreślonej w specyfikacji, gdzie w stanie *a0* zadany zostanie tylko sygnał sterujący *x2*, który nie doprowadzi do opuszczenia stanu *a0*.

$$|{-}x2{*}/Q1{*}/Q2{*}/Q3; \qquad (6)$$

W wyniku wtórnej normalizacji systemem wnioskującym i uporządkowaniu wyrażeń ze względu na typy zmiennych otrzymuje się równania opisujące zachowanie układu (Tab. 5).

Tab. 5. Symulacja logiczna dla niepoprawnie zdefiniowanej akcji
Tab. 5. Logical simulation for not correctly defined action

Wyznaczone zachowanie układu			
`|-/D1;`	`x1|-/D2;`	`/x1|-/D3;`	`/x1|-y3;`
`x1|-D3;`	`x1|-y2;`	`/x1|-D2;`	`|-y1;`

W sytuacji niedokładnego sprecyzowania akcji system Gentzena wyznaczy wszystkie możliwe warianty. W rozpatrywanej sytuacji sygnał wyjściowy *y1* generowany jest zawsze. Jeżeli sygnał sterujący *x1* ma wartość „*1*", to generowane jest wyjście *y2*, a także układ przechodzi do stanu *a1* opisanego sekwentami (*|-/D1*, *x1|-/D2* i *x1|-D3*). W przypadku, gdy sygnał *x1* przyjmie wartość „*0*", to na wyjściu pojawi się oprócz sygnału *y1* sygnał *y3*, a układ przejdzie do stanu *a3*, któremu odpowiadają sekwenty (*|-/D1*, */x1|-D2* i */x1|-/D3*). Zachowanie takie odpowiada specyfikacji opisanej siecią działań.

5. Podsumowanie

Artykuł jest ilustracją możliwości zastosowania komputerowego wnioskowania symbolicznego Gentzena do rozwiązywania skomplikowanych problemów logicznych, jako formalna procedura dowodzenia. Zawiłe zależności między parametrami mogą być przekształcone do prostszych, łatwiejszych do weryfikacji relacji binarnych. Dokumentacja w postaci drzewa jest równocześnie dowodem formalnym wykazującym poprawność wykonanych przekształceń. Opracowane eksperymentalne oprogramowanie może być wykorzystywane samodzielnie lub stanowić nakładkę do systemu ESPRESSO. Rezultaty projektowania w postaci regułowej w łatwy sposób transformowane są na opisy w językach VHDL i Verilog.

6. Literatura

[1] Adamski M.: *Projektowanie układów cyfrowych systematyczną metodą strukturalną*, Wydawnictwo Wyższej Szkoły Inżynierskiej w Zielonej Górze, Zielona Góra 1990.
[2] Adamski M, Barkalov A.: *Architectural and Sequential Synthesis of Digital Devices*. University of Zielona Góra Press. Zielona Góra 2006.
[3] Barkalov A., Węgrzyn M.: *Design of control units with programmable logic*. University of Zielona Góra Press, Zielona Góra 2006.
[4] Ben-Ari M.: *Logika matematyczna w informatyce*. Wydawnictwa Naukowo-Techniczne, Warszawa 2005.
[5] Bukowiec A.: *Automata Synthesis System – Sample FSMs*. Publikacja internetowa: http://willow.iie.uz.zgora.pl/~abukowie/AS/as.htm
[6] Gallier J.H.: *Logic for computer science. Foundations of Automatic Theorem Proving*. Harper & Row 1986
[7] Indrzejczak A.: *Wprowadzenie do rachunku sekwentów- zagadnienia metodologiczne, zastosowania*. Książka internetowa. http://www.filozof.uni.lodz.pl/prac/ai/Gentzen.pdf
[8] Indrzejczak A.: *Elementy logiki*. Wyższa Szkoła Humanistyczno-Ekonomiczna w Łodzi, Łódź, 2005
[9] Łuba T., Rawski M., Tomasiewicz P., Zwierzchowski B.: *Synteza układów cyfrowych*. Praca zbiorowa pod redakcją prof. Tadeusza Łuby. Wydawnictwa Komunikacji i Łączności, Warszawa 2003.
[10] Kamionka-Mikuła H., Małysiak H., Pochopień B.: *Synteza i analiza układów cyfrowych*. Wydawnictwo Pracowni Komputerowej Jacka Skalmierskiego, Gliwice 2006.
[11] Szajna J.: *Projektowanie układów cyfrowych z wykorzystaniem programowania logicznego*. Wydawnictwo Politechniki Zielonogórskiej, Zielona Góra 1996.
[12] Tkacz J., Adamski M.: *Wykorzystanie komputerowego wnioskowania w projektowaniu kombinacyjnych układów sterowania*. Przegląd Telekomunikacyjny nr 6, 2008, s 728-730.

Artykuł recenzowany