Valery SALAUYOU, Adam KLIMOWICZ, Tomasz GRZEŚ,
Teodora DIMITROVA-GREKOW, Irena BUŁATOWA
POLITECHNIKA BIAŁOSTOCKA, WYDZIAŁ INFORMATYKI

Badania efektywności metod syntezy automatów skończonych zaimplementowanych w pakiecie ZUBR

Dr hab. inż. Valery SALAUYOU

Ukończył w 1978 r. studia na wydziale Matematyki Stosowanej w Białoruskim Państwowym Uniwersytecie w Mińsku. Obronił pracę doktorską w 1986 r. o specjalności „Informatyka Techniczna" i uzyskał tytuł doktora habilitowanego w 2003 r. Od 25 lat pracuje w dziedzinie projektowania logicznego systemów cyfrowych.

e-mail: walsol@ ii.pb.bialystok.pl

Mgr inż. Adam KLIMOWICZ

Ukończył studia w Instytucie Informatyki Politechniki Białostockiej. Jest asystentem na Wydziale Informatyki Politechniki Białostockiej. Jego zainteresowania naukowe to synteza układów kombinacyjnych i automatów skończonych na bazie programowalnych układów logicznych, ze szczególnym uwzględnieniem układów o strukturze PLD/CPLD.

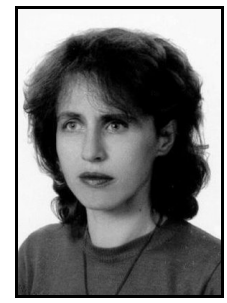
e-mail: aklim@ ii.pb.bialystok.pl

Mgr inż. Tomasz GRZEŚ

Ukończył w 1999 r. studia w Instytucie Informatyki Politechniki Białostockiej. Obecnie jest asystentem na Wydziale Informatyki Politechniki Białostockiej. W pracy naukowej zajmuje się problemem minimalizacji mocy pobieranej przez układy programowalne.

e-mail: grzes@ chilan.com

Dr inż. Teodora DIMITROVA-GREKOW



Ukończyła studia na Wydziale Elektroniki Politechniki Sofijskiej w Bułgarii w 1991r., obroniła pracę doktorską na Uniwersytecie Technicznym w Wiedniu w 1997 r. Jest adiunktem na Wydziale Informatyki Politechniki Białostockiej. Jej zainteresowania naukowe to przemysłowe systemy programowalne, synteza układów cyfrowych na PLD, systemy równoległe.

e-mail: teodora@ ii.pb.bialystok.pl

Dr inż. Irena BUŁATOWA

Ukończyła studia na Wydziale Systemów Komputerowych Uniwersytetu Informatyki i Radioelektroniki w Mińsku na Białorusi, gdzie też obroniła pracę doktorską w 1992r. Jest adiunktem na Wydziale Informatyki Politechniki Białostockiej. Jej zainteresowania naukowe to metody syntezy automatów skończonych na bazie programowalnych układów logicznych.

e-mail: irena@ ii.pb.bialystok.pl

Streszczenie

W artykule przedstawiono pakiet programów ZUBR automatyzacji projektowania logicznego systemów cyfrowych na programowalnych układach logicznych. Opisano metody syntezy automatów skończonych zaimplementowane w pakiecie ZUBR. Wyniki badań eksperymentalnych potwierdzają efektywność opracowanych metod w porównaniu do metod stosowanych w pakietach przemysłowych pod względem kosztu realizacji i szybkości działania.

Słowa kluczowe: automat skończony, metoda syntezy, programowalne układy logiczne.

Experimental Researches of Fnite State Machines Synthesis Methods Realized in Package ZUBR

Abstract

In this paper the software package ZUBR for logical design of digital devices on programmable logic devices is presented. The methods of synthesis of finite state machines (FSM) implemented in package ZUBR are described. Experimental results show the higher efficiency (lower cost and higher device speed) of proposed synthesis methods in comparison with industrial design systems.

Keywords: finite state machine (FSM), synthesis method, programmable logic devices.

1. Wstęp

Programowalne układy logiczne (PLD) są szeroko stosowane do realizacji układów sekwencyjnych, których modelem matematycznym jest automat skończony. W wyniku ciągłego rozwoju architektur układów programowalnych, rosną ich możliwości do realizacji systemów cyfrowych. Z tego powodu, ważnym zadaniem jest opracowanie efektywnych metod syntezy automatów skończonych, które uwzględniają cechy szczególne architektur współczesnych układów programowalnych.

Pakiet programów ZUBR opracowany został w celu optymalizacji metod projektowania logicznego układów kombinacyjnych i automatów skończonych. Wyższa efektywność metod syntezy osiąga się przede wszystkim dzięki wykorzystaniu specyficznych właściwości architektur współczesnych układów programowalnych już na etapie projektowania logicznego.

2. Podstawowe zadania projektowe systemu ZUBR

Pakiet ZUBR pozwala na podwyższenie efektywności metod syntezy układów cyfrowych, która wyraża się zwiększeniem szybkości działania i obniżeniem kosztów realizacji systemów. Przy pomocy pakietu ZUBR można wykonywać następujące zadania projektowe:

- synteza logiczna układów kombinacyjnych i automatów skończonych;
- weryfikacja syntezy układów kombinacyjnych i automatów skończonych;
- przedstawienie wyników syntezy w językach projektowania przemysłowych pakietów;
- automatyczny dobór najbardziej odpowiednich modeli automatów skończonych dla konkretnej architektury układu programowalnego i wymagań systemowych;
- konwersja opisów układów kombinacyjnych i automatów skończonych z formatów systemu SIS na języki projektowania VHDL, Verilog, Abel, AHDL;

- określenie zależności pobieranej mocy w układach automatów skończonych od częstotliwości przełączenia sygnałów wejściowych;
- automatyczne generowanie sekwencji testowych dla weryfikacji kolejnych etapów projektowania układów cyfrowych i inne.

Pakiet ZUBR wykorzystuje się wspólnie z przemysłowymi pakietami automatyzowanego projektowania systemów cyfrowych na PLD, np. firm Altera, Xilinx, Mentor Graphics: podstawowe etapy projektowania (wprowadzenie danych, kompilacja, modelowanie i inne) wykonują się przy pomocy pakietu przemysłowego, natomiast synteza logiczna – przy pomocy pakietu ZUBR. Specjalne programy konwerterów zapewniają zgodność pakietu ZUBR z pakietami przemysłowymi, przedstawiając opisy wejściowe oraz wyniki syntezy w wymaganych formatach. Dodatkowo opracowane zostały programy wspomagające projektowanie, które umożliwiają wybór najbardziej odpowiedniego modelu automatu skończonego i najlepszej metody syntezy dla konkretnego zastosowania.

3. Metody syntezy automatów skończonych

Metody syntezy automatów skończonych zaimplementowane w pakiecie ZUBR bazują się na klasyfikacji modeli automatów [1], zgodnie z którą wyróżniamy następujące klasy automatów: A, B, C, D, E i F. Modele te uwzględniają specyficzne cechy budowy układów PLD (przerzutniki buforów wejściowych, wyjściowych, przerzutniki w pętlach sprzężeń zwrotnych, makrokomórki z dwoma sprzężeniami zwrotnymi, różna liczba termów makrokomórek i inne) i wykorzystują je do podwyższenia efektywności realizacji automatów.

Automaty klas A i B – są to standardowe modele automatów Mealy'ego i Moore'a. W automacie klasy C (Moore'a) przerzutniki buforów wyjściowych PLD wykorzystywane są jako elementy pamięci automatu. W automacie klasy D (Mealy'ego) przerzutniki pętli sprzężeń zwrotnych wykorzystywane są w jakości elementów pamięci automatu. W modelach klas E (Mealy'ego) i F (Moore'a) elementami pamięci automatu są przerzutniki buforów wejściowych PLD. Opracowano także wspólne modele automatów klas ADE, AD, AE, BF, które łączą w sobie zalety i niwelują wady poszczególnych klas automatów.

Proponowane podejście [3] polega na wykonaniu następujących kroków:

- w zależności od cech układu sekwencyjnego, architektury PLD oraz wymagań dotyczących szybkości i kosztów, dla każdego podsystemu wybierany jest najbardziej odpowiedni model automatu skończonego;
- na podstawie charakterystyk wewnętrznych automatu skończonego oraz wybranego modelu, wybierana jest optymalna metoda syntezy;
- wykonuje się syntezę automatu skończonego;
- buduje się zbiór funkcji boolowskich, odpowiadający kombinacyjnej części automatu (w razie potrzeby wykonuje się ich minimalizację);
- wykonuje się syntezę zbioru funkcji boolowskich jedną z wybranych metod syntezy układów kombinacyjnych na bazie PLD;
- w razie konieczności wykonuje się odwzorowanie układu logicznego w układ PLD.

W pakiecie ZUBR zaimplementowane zostały następujące metody syntezy automatów skończonych:

- szybkich automatów klas A i B (metoda A1);
- złożonych automatów klas A i B (metoda A2);
- automatów klasy C (metoda A3);
- automatów klasy D (metoda A4);
- automatów klasy E (metoda A5);
- automatów klasy F (metoda A6);
- wspólnego modelu automatów klas ADE (metoda A7);
- wspólnego modelu automatów klas AD (metoda A8);
- wspólnego modelu automatów klas AE (metoda A9);
- wspólnego modelu automatów klas BF (metoda A10).

Metoda syntezy A1 szybkich automatów klas A i B pozwala budować automaty skończone o bardzo dużej szybkości działania, dla których częstotliwość przełączania elementów pamięci jest równa maksymalnej częstotliwości pracy układów PLD. Większą szybkość osiąga się dzięki temu, że każda funkcja przejść jest realizowana na jednej makrokomórce PLD. Aby ograniczyć złożoność funkcji przejść wykorzystane jest rozszczepienie stanów wewnętrznych tak, żeby liczba przejść w każdym stanie nie przewyższała liczby termów podłączonych do jednej makrokomórki. W procesie kodowania stanów wewnętrznych kontroluje się złożoność funkcji przejść i dla funkcji, których nie można zrealizować na jednej makrokomórce, zwiększa się liczba R bitów kodu stanów wewnętrznych. Metoda A1 pozwala efektywnie wykorzystać makrokomórki PLD z podwójnym sprzężeniem zwrotnym poprzez jednoczesną realizację funkcji przejść i wprowadzanie wartości zmiennych wejściowych. Główną wadą tej metody jest wąski obszar zastosowania z powodu rozchodzenia (dążenia do nieskończoności) się algorytmu rozszczepienia stanów wewnętrznych.

Metoda syntezy A2 automatów skończonych klas A i B pozwala budować na PLD automaty o praktycznie nieograniczonej złożoności. Nie gwarantuje jednak osiągnięcia maksymalnej szybkości działania automatu. Dla zwiększenia szybkości działania w metodzie A2 rozczepienie stanów wewnętrznych stosuje się tylko w wybranych przypadkach. W obu metodach syntezy A1 i A2 automatów skończonych klas A i B przewiduje się możliwość programowania poziomu logicznego sygnałów PLD, w celu obniżenia kosztów realizacji.

Cechą charakterystyczną metod syntezy A3 i A4 automatów skończonych klasy C i D jest wykorzystywanie przerzutników wyjściowych makrokomórek układów PLD jako elementów pamięci automatu w przypadku, gdy wektory zmiennych wyjściowych są identyczne z częścią kodu stanów wewnętrznych automatu [2]. Pozwala to na znaczne obniżenie kosztów realizacji i równoczesne podwyższenie szybkości działania automatów skończonych w porównaniu z automatami klas A i B. Obniżenie kosztów realizacji uzyskuje się przez zmniejszenie liczby wykorzystywanych makrokomórek wyjściowych PLD. Oprócz tego, uproszczona będzie część kombinacyjna automatu, ponieważ odpada konieczność realizacji funkcji wzbudzeń elementów pamięci, które są identyczne z funkcjami wyjściowymi.

Zwiększoną, w porównaniu do tradycyjnych metod, szybkość działania automatów skończonych klasy C i D można wytłumaczyć tym, że w automatach klasy C i D realizowane są przeważnie, funkcje wyjściowe, które są z reguły prostsze niż funkcje przejść, co prowadzi do zmniejszenia liczby poziomów logicznych przy syntezie części kombinacyjnej automatu skończonego.

Kodowanie stanów automatu klasy C do rozwiązania zadania ortogonalizacji wierszy macierzy, które wykorzystuje się w charakterze kodów stanów wewnętrznych automatu. Zaproponowany algorytm pozwala minimalizować liczbę wartości znaczących w macierzy, co prowadzi do zmniejszenia liczby argumentów realizowanych funkcji.

Wadą metod A3 i A4 jest konieczność rozszczepiania stanów wewnętrznych przy przejściu od automatu typu Mealy'ego do automatu typu Moore'a, dla metody A3 i przy przejściu od automatu klasy A do automatu klasy D, dla metody A4. Oprócz tego przy budowie automatu klasy D, układ PLD powinien dopuszczać konfigurację wyjściowych makrokomórek z przerzutnikami w pętli sprzężenia zwrotnego.

Główną cechą wyróżniającą metod syntezy A5 i A6 automatów skończonych klasy E i F [1] jest wykorzystanie przerzutników wejściowych makrokomórek układów PLD jako elementów pamięci automatu w przypadku, gdy wektory zmiennych wejściowych mają identyczne wartości z częścią kodu stanów wewnętrznych. Pozwala to obniżyć koszt realizacji i jednocześnie zwiększyć szybkość działania automatów w porównaniu z automatami klas A i B.

W metodach A5 i A6 rozszczepienie stanów wewnętrznych wykorzystuje się w celu przekształcenia automatów klas A i B odpowiednio w automaty klas E i F. Kodowanie stanów wewnętrznych automatów klas E i F prowadzi do zadania pokrycia grafu ortogonalności wierszy macierzy kodów minimalną liczbą pełnych podgrafów. W ogólnym przypadku, aby możliwe było zbudowanie automatów skończonych klasy E i F, każdy bufor wejściowy układu PLD powinien posiadać dwa sprzężenia z logiką wewnętrzną układu PLD: rejestrowe i kombinacyjne. Do

wad metod A5 i A6 można odnieść konieczność rozszczepienia stanów wewnętrznych przy przejściu od automatów klas A i B do automatów klas E i F. Ponadto istnieje konieczność rozszczepienia stanów wewnętrznych do przekształcenia automatu typu Mealy'ego w automat Moore'a (przy syntezie automatów klasy F).

Metoda syntezy A7 wspólnego modelu automatów skończonych klas ADE [3], pozwala najbardziej efektywnie wykorzystać możliwości architektur układów PLD: przerzutniki buforów wejściowych i makrokomórek wyjściowych układów PLD używane są w charakterze elementów pamięci automatu. Metoda A7 maksymalnie wykorzystuje modele automatów klasy D i E, co pozwala zminimalizować liczbę R elementów pamięci. W metodzie A7 stosuje się rozszczepienie stanów wewnętrznych względem wektorów zmiennych wejściowych i wejściowych. Wadą metody A7 są podwyższone wymagania do możliwości architektury układów PLD: bufory wejściowe muszą mieć dwa rodzaje sprzężeń z częścią kombinacyjną oraz konieczność umożliwienia konfiguracji makrokomórek wyjściowych z przerzutnikami w pętli sprężenia zwrotnego.

Metody syntezy A8 i A9 wspólnych modelu automatów skończonych klas AD i AE są zgodne odpowiednio z metodą A7, kiedy nie istnieje możliwości wykorzystywania buforów wejściowych układu PLD (metoda A8) lub wyjściowych makrokomórek w charakterze elementów pamięci (metoda A9). Metoda A10 jest identyczna z metodą A9 dla automatów klas AE, z tą różnicą, że zamiast automatu Mealy'ego rozpatruje się automat Moore'a.

4. Badania eksperymentalne

Badania eksperymentalne zostały przeprowadzone na przykładach testowych opracowanych w MCNC [4]. Metody pakietu ŻUBR zostały porównane z metodami wykorzystywanymi w następujących pakietach przemysłowych: MAX PLUS II 10.1, firmy Altera, WebPack 5.2, firmy Xilinx, oraz FPGA Advantage 5.2 firmy Mentor Graphics.

Przy syntezie każdego przykładu za pomocą pakietów przemysłowych były wykonywane kolejno następujące czynności:

- opis układu wejściowego w języku systemu SIS za pomocą konwertera pakietu ZUBR został przekonwertowany na język VHDL (AHDL dla systemu MAX+PLUS II);
- wykonywano syntezę układu za pomocą pakietu przemysłowego na PLD odpowiedniej rodziny, przy czym parametry syntezy były ustawiane na maksymalną minimalizację kosztów (powierzchni układu);
- rezultaty syntezy oceniano ze względu na koszt (liczbę wykorzystanych elementów logicznych PLD) i szybkość działania (maksymalne opóźnienie w nanosekundach przy przejściu sygnałów z wejść na wyjścia) za pomocą pakietu przemysłowego.

Przy syntezie każdego testowego przykładu z wykorzystaniem pakietu ZUBR wykonywano kolejno następujące czynności:

- wykonywano syntezę układu za pomocą pakietu ZUBR na PLD odpowiedniej rodziny otrzymano wyniki w języku VHDL (AHDL dla systemu MAX+PLUS II);
- wykonywano syntezę otrzymanego układu za pomocą pakietu przemysłowego na PLD odpowiedniej rodziny, przy parametrach ustawionych na minimalizację kosztów;
- rezultaty syntezy oceniano ze względu na koszt (liczbą wykorzystanych elementów logicznych PLD) i szybkość działania (maksymalne opóźnienie w nanosekundach przy przejściu sygnałów z wejść na wyjścia) za pomocą pakietu przemysłowego.

Rezultaty badań eksperymentalnych są pokazane w tabeli 1, gdzie przyjęto następujące oznaczenia: C_{mid} - średni spadek kosztu realizacji układów syntezowanych za pomocą pakietu ŻUBR; C_{max} – maksymalny spadek kosztu realizacji układów syntezowanych za pomocą pakietu ŻUBR; D_{mid} – średni wzrost szybkości działania układów syntezowanych za pomocą pakietu ŻUBR; D_{max} – maksymalny wzrost szybkości działania układów syntezowanych za pomocą pakietu ŻUBR.

Przeprowadzone badania pokazały wysoką efektywność metod syntezy zaimplementowanych w pakiecie ŻUBR. Wykorzystanie pakietu ŻUBR do syntezy automatów skończonych pozwala obniżyć koszt realizacji średnio od 1,78 do 7,5 razy, a dla niektórych przykładów aż 23 razy. Przy czym średnia szybkość działania może wzrosnąć 2,63 razy, a dla oddzielnych przykładów 5,98 razy.

Tab. 1. Porównanie efektywności metod syntezy automatów skończonych
Tab. 1. Comparison of effichiency of FSM synthesis methods

Metoda	PLD	C_{mid}	C_{max}	D_{mid}	D_{max}
MAX+PLUSII Altera					
A2	MAX7000	1,78	2,75	1,27	1,98
A2	MAX9000	1,80	2,75	0.99	1.33
A3	MAX5000	3,37	17,00	-	-
A4	FLEX10K	7,50	23,00	-	-
A5	FLEX10K	3,63	8,00	1,43	1,74
A7	FLEX10K	3,74	5,00	-	-
A7	MAX9000	2,04	5,00	-	-
A8	FLEX10K	4,22	7,67	1,48	1,69
A8	MAX9000	2,38	5,33	1,10	1,31
A9	FLEX10K	4,16	8,00	1,50	1,74
WebPack Xilinx					
A2	XC9500	2,04	3,00	-	-
A5	VIRTEXII	3,46	9,00	1,29	1,42
A5	MAX5000	3,37	17.00	-	-
A7	XC9500	2,24	4,50	-	-
A7	VIRTEXII	3,14	5,67	-	-
A8	XC9500	2,06	3,00	2,09	5,98
A8	VIRTEXII	3,65	8,67	1,29	1,43
A9	XC9500	2,13	3,00	2,63	5,98
A9	VIRTEXII	4,07	9,00	1,20	1,42
FPGA Advantage Mentor Graphics					
A2	MAX7000	1,80	5,33	0,95	1,53
A2	XC9500	1,96	3,63	-	-
A7	MAX9000	2,29	6,75	0,94	1,00
A7	FLEX10K	2,56	6,67	1,41	3,00
A7	VIRTEXII	3,95	13,67	0,97	1,20

5. Podsumowanie

Metody zaimplementowane w pakiecie ZUBR w dużym stopniu wykorzystują specyficzne cechy architektur układów PLD, co prowadzi do znacznego zmniejszenia kosztów realizacji oraz zwiększenia szybkości działania projektowanych układów. Przeprowadzone badania eksperymentalne potwierdziły efektywność opracowanych metod syntezy automatów skończonych oraz pozwoliły określić zakres zastosowań poszczególnych modeli automatów i metod syntezy.

Metoda A1 pozwala budować na PLD względnie proste automaty skończone o bardzo wysokiej szybkości działania. Metoda A2 pozwala budować na PLD automaty skończone o praktycznie nieograniczonej złożoności i nominalne szybkości działania i koszty. Metody A3, A4, A5 i A6 w szczególnych przypadkach są zależne od wewnętrznych charakterystyk skończonego automatu, pozwalają budować automaty skończone o bardzo niskie koszty i wysokiej szybkości działania. Metody A7, A8, A9 i A10 pozwalają na maksymalne wykorzystywanie architektury układów PLD, co rzutuje na budowanie automaty skończone o bardzo niskie koszty i wysokiej szybkości działania.

6. Literatura

[1] Sołowiew W.W.: Projektirowanije cifrowych sistiem na osnowie programmirujemych logiczeskich integralnych schiem. Moskwa, Hot Line-Telekom, 2001, 636 s

[2] Klimowicz A., Sołowjew W. Wykorzystywanie wyjściowych makrokomórek układu PLD w charakterze elementów pamięci automatu skończonego // Materiały VI Krajowej Konferencji Naukowej Reprogramowalne Układy Cyfrowe (RUC'2003), (Szczecin, 8-9 maja 2003).

[3] Sołowjew W., Klimowicz A. Synteza wspólnych modeli automatów skończonych na PLD // Materiały V Krajowej Konferencji Naukowej Reprogramowalne Układy Cyfrowe (RUC'2002), (Szczecin, 9-10 maja 2002) s. 35-42.

[4] Yang S. Logic synthesis and optimization benchmarks user guide. Version 3.0. - Technical Report, Microelectronics Centre of North Carolina, 1991. - 43p.

Artykuł recenzowany