Małgorzata KOŁOPIEŃCZYK
UNIWERSYTET ZIELONOGÓRSKI, Licealna 9, 65-417 Zielona Góra

Wykorzystanie zasobów sprzętowych w mikroprogramowanych układach sterujących z kodowaniem kolekcji mikrooperacji

Dr inż. Małgorzata KOŁOPIEŃCZYK

Absolwentka Wydziału Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego (1999 r.). Obecnie adiunkt w Instytucie Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zajmuje się następującymi zagadnieniami: projektowaniem i syntezą sterowników logicznych oraz inżynierią oprogramowania.

e-mail: m.kolopienczyk@iie.uz.zgora.pl

Streszczenie

W artykule omówiono wyniki prac nad wydajnym wykorzystaniem zasobów sprzętowych w mikroprogramowanych układach sterujących z wykorzystaniem metody współdzielenia kodów z kodowaniem kolekcji mikrooperacji. Porównano dwa typy układów mikroprogramowanych: układ z konwerterem adresu oraz układ bez konwertera. Badania wykazały, że zastosowanie metody współdzielenia kodów z kodowaniem mikrooperacji oraz wprowadzenie bloku konwertera adresu do mikroprogramowanego układu sterującego skutkuje, dla niektórych przypadków, co najmniej 30% zmniejszeniem rozmiaru pamięci jednostki sterującej implementowanej jako mikroprogramowany układ sterujący.

Słowa kluczowe: układ mikroprogramowany, konwerter adresu, mikrooperacje, współdzielenie kodów.

Use of hardware resources for compositional microprogram control units with microoperation collections encoding

Abstract

The paper presents results of efficient use of hardware resources for a microprogram control unit with code sharing and microoperation collections encoding. Two types of microprogram control unit are compared; structures with and without an address converter (Fig. 2). Xilinx ISE 8.2i package was used for synthesis and implementation of the microprogram control unit [7]. The target platform was the FPGA device Xilinx Virtex-II Pro xc2vp30-7ff896c. It can be concluded that implementation of the compositional microprogram control unit with codes sharing and microoperation collections encoding presented in this work results in decrease of the control memory size required for the control unit. In some cases the memory consumption drops even by 30% in comparison to the implementation without the address converter. This paper is divided into four parts. The first paragraph is a brief introduction to the issues of compositional microprogram control unit design [1, 8]. In the second and third paragraphs the results of resource utilisation are presented. The last – fourth – paragraph contains a summary.

Keywords: micoprogram control unit, address converter, microoperation, code sharing.

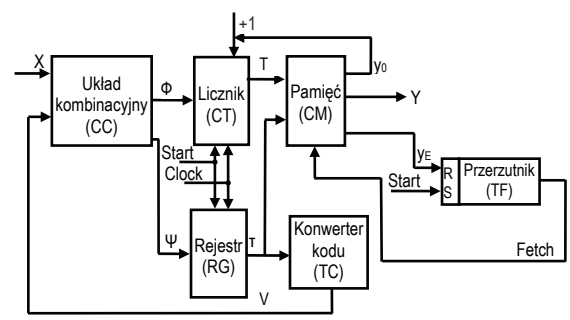
1. Wprowadzenie

W ostatnich latach można zaobserwować duże zmiany zarówno w metodach projektowania, jak i wytwarzania układów cyfrowych. Strukturalną reprezentację układu zastąpiono językami opisu sprzętu a płytkę drukowaną układami typu FPGA (ang. Field Programmable Gate Array). Szybki rozwój w dziedzinie techniki cyfrowej umożliwił implementację złożonych systemów cyfrowych przy użyciu pojedynczego zintegrowanego układu typu "System-On-a-Chip" (SoC) [1, 2, 10]. Ciągłe powiększanie funkcjonalności systemu cyfrowego w obrębie pojedynczego mikroobwodu SoC powoduje konieczność optymalizacji sprzętu. Bardzo ważnym elementem każdego systemu cyfrowego jest jednostka sterująca, która może być implementowana jako mikroprogramowany układ sterujący [1, 8]. Układy typu SoC zawierają matryce programowalne FPGA, które składają się z elementów LUT (ang. Look-Up Table) i dedykowanych bloków pamięci DMB (ang. Dedicated Memory Blocks). Liczba wejść elementu LUT jest ściśle ograniczona, co prowadzi do konieczności zmniejszenia ilości argumentów w obwodzie adresu mikroprogramowanego układu sterującego. Jednym ze sposobów rozwiązania tego problemu, jest zastosowanie metody współdzielenia kodów. Zastosowanie tej metody ma jednak sens tylko wówczas, gdy zostanie zachowana minimalna liczba bitów w adresie mikroinstrukcji – inaczej liczba DMB w pamięci ulegnie zwiększeniu.

W artykule zaprezentowano wyniki prac nad zmniejszeniem zużycia zasobów sprzętowych w mikroprogramowanych układach sterujących z wykorzystaniem metody współdzielenia kodów z kodowaniem kolekcji mikrooperacji. Porównano dwa typy układów mikroprogramowanych: układ z konwerterem adresu oraz układ bez konwertera. Szczegółowy opis metody współdzielenia kodów oraz kodowania kolekcji mikrooperacji można znaleźć w pracy [5].

2. Mikroprogramowany układ sterujący

W niniejszym artykule omówiono dwa typy układów mikroprogramowanych: układ bez konwertera (ang. Compositional Microprogram Control Unit with Code Sharing, CMCU_CS) (rys. 1) oraz układ z konwerterem adresu i kodowaniem kolekcji mikrooperacji (rys. 2) (ang. Compositional Microprogram Control Unit with Encoding of Collections of Microoperations, CMCU_EM).



Rys. 1. Mikroprogramowany układ sterujący CMCU_CS
Fig. 1. Microprogram control unit CMCU_CS

Na rysunku 1 przedstawiono strukturę i przepływ sygnałów w mikroprogramowanym układzie sterującym. Mikroprogramowany układ sterujący jest rozwinięciem dwupoziomowej struktury skończonego automatu stanów.

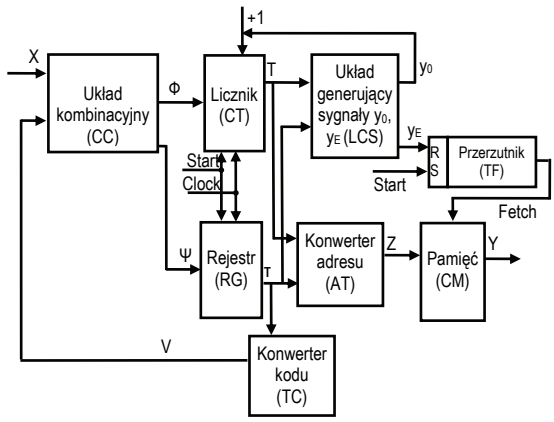
W tym przypadku jednostka sterująca zostaje poddana dekompozycji na dwa podstawowe bloki [2, 3]: układu zarządzającego oraz układu generującego i przechowującego mikroinstrukcje.

Układ zarządzający to uproszczony automat stanów, odpowiedzialny za wyznaczanie adresu mikroinstrukcji. Zbudowany jest z układu kombinacyjnego CC, generującego funkcje wzbudzeń oraz rejestru RG przechowującego wartości aktualnego stanu, w jakim znalazł się układ [4, 9].

Pamięć CM odpowiedzialna jest za przechowywanie mikroinstrukcji. Mikroinstrukcje są generowane na podstawie funkcji wyznaczonych przez licznik CT oraz rejestr RG. Konwerter kodu TC umożliwia przekształcenie kodu łańcucha w kod klasy pseudoekwiwalentnych łańcuchów. Przerzutnik TF wyznacza sygnał Fetch, co umożliwia pobranie mikroinstrukcji z pamięci.

Zaletą takiej struktury jest minimalna liczba wyjść układu CC oraz jego wejść, uzyskiwana dzięki konwerterowi TC.

Zasada działania mikroprogramowanego układu sterującego z konwerterem adresu (rys. 2) jest taka sama jak układu bez konwertera. Różnica polega na wprowadzeniu dodatkowego bloku - konwerter adresu, do struktury układu mikroprogramowanego. Konwerter adresu przekształca adres mikroinstrukcji na adres o minimalnej pojemności bitowej i wyznacza adres mikroinstrukcji (funkcja Z) na podstawie kodu stanu, w jakim znajduje się rejestr, oraz kodu bloku operacyjnego w danym łańcuchu. Układ LCS (ang. Local Control Signal) służy do generowania wewnętrznych zmiennych y_0 oraz y_E.



Rys. 2. Mikroprogramowany układ sterujący CMCU_EM
Fig. 2. Microprogram control unit CMCU_EM

3. Analiza zużycia zasobów sprzętowych

Prezentowane struktury zaimplementowano z wykorzystaniem oprogramowania Xilinx ISE 8.2i. Zastosowano cztery dostępne w Pakiecie strategie optymalizacji. Platformę docelową stanowił układ FPGA Xilinx Virtex–II Pro (xc2vp30-7ff896c) [3, 6, 7].

Do testowania rozmiaru pamięci struktur układów mikroprogramowanych wykorzystano oprogramowanie fca2cmcu [5].

Wejściem dla programu fca2cmcu jest plik z opisem tekstowym sieci działań. Podstawowym zadaniem programu jest translacja tekstowej postaci sieci działań do struktur mikroprogramowanych układów sterujących opisanych w języku VHDL.

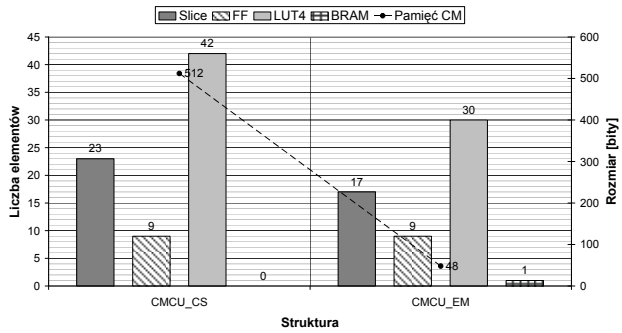
Zestaw testowy zawierał 40 różnych struktur układów mikroprogramowanych. W tabeli 1 przedstawione zostały rozmiary pamięci mikroprogramowanych układów sterujących wygenerowanych przez program fca2cmcu na podstawie zestawu sieci testowych.

Tab. 1. Rozmiar pamięci [bity]
Tab. 1. Memory size [bits]

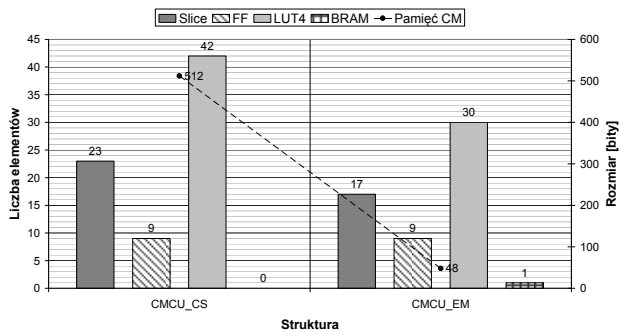
Sieć testowa	Układ bez konwertera adresu	Układ z konwerterem adresu
we_00	512	48
we_01	2304	12
we_02	2304	12
we_03	1208	128
we_04	2560	128
we_05	2816	144
we_06	1664	352
we_07	2560	128
we_08	1152	112
we_09	2304	112
we_10	3328	352
we_11	1280	128

Na podstawie tabeli 1 można stwierdzić, że zastosowanie konwertera adresu skutkuje zmniejszeniem rozmiaru pamięci jednostki sterującej implementowanej jako mikroprogramowany układ sterujący.

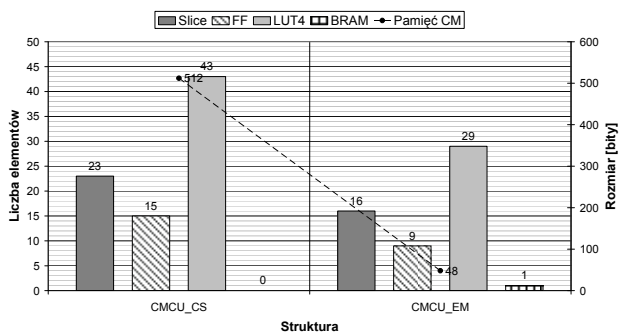
Na rysunkach 3, 4, 5 oraz 6 przedstawiono zużycie zasobów sprzętowych dla jednej z testowych sieci działań (we_00) dla różnych strategii optymalizacji.



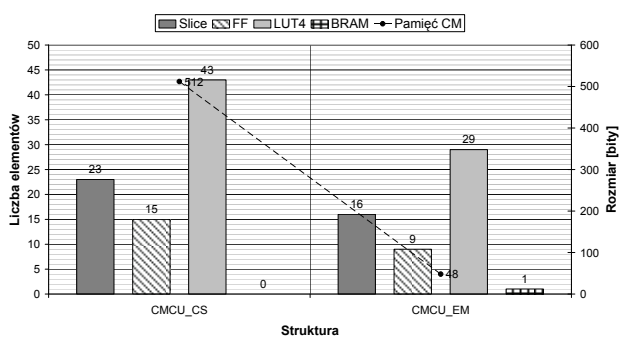
Rys. 3. Zużycie zasobów sprzętowych dla sieci testowej we_00 (area level 1)
Fig. 3. Resource utilization for flow-chart we_00 (area level 1)



Rys. 4. Zużycie zasobów sprzętowych dla sieci testowej we_00 (area level 2)
Fig. 4. Resource utilization for flow-chart we_00 (area level 2)



Rys. 5. Zużycie zasobów sprzętowych dla sieci testowej we_00 (speed level 1)
Fig. 5. Resource utilization for flow-chart we_00 (speed level 1)



Rys. 6. Zużycie zasobów sprzętowych dla sieci testowej we_00 (speed level 2)
Fig. 6. Resource utilization for flow-chart we_00 (speed level 2)

Na podstawie wykresów przedstawionych na powyższych rysunkach można stwierdzić, że liczba elementów LUT w mikroprogramowanym układzie z konwerterem adresu jest mniejsza niż w układzie bez bloku konwertera adresu.

W prezentowanych przykładach wykorzystano identyczną liczbę bloków BRAM. Jest to spowodowane faktem, że sieci testowe zawierały około 100 bloków operacyjnych, natomiast pojemność bloków BRAM w układzie Xilinx Virtex II Pro wynosi 18 kilobitów. Dla bardziej rozbudowanych sieci działań wykorzystana zostałaby większa liczba bloków BRAM [5].

Tab. 2. Minimalny czas trwania cyklu [ns]
Tab. 2. Minimal period [ns]

Sieć testowa	Struktura CMCU	Minimalny czas trwania cyklu [ns]			
		speed 1	speed 2	area 1	area 2
we_00	CS	5,443	5,443	5,965	5,965
	EM	3,687	3,687	4,495	4,495
we_01	CS	6,967	6,967	9,073	9,073
	EM	5,610	5,610	9,514	9,514
we_02	CS	7,293	7,293	9,466	9,466
	EM	5,715	5,715	7,147	7,147
we_03	CS	5,743	5,473	10,076	10,076
	EM	5,022	5,022	7,516	7,516
we_04	CS	7,897	7,846	13,158	13,158
	EM	5,965	5,965	9,231	9,231
we_05	CS	8,260	8,260	12,122	12,122
	EM	4,729	4,699	6,094	6,094
we_06	CS	3,304	3,237	3,402	3,402
	EM	5,340	5,340	6,756	6,756
we_07	CS	6,692	6,692	13,307	13,307
	EM	5,424	5,424	7,497	7,497
we_08	CS	6,615	6,679	7,344	7,344
	EM	6,386	6,386	6,605	6,605
we_09	CS	6,488	6,488	9,626	9,626
	EM	6,213	6,213	6,669	6,669
we_10	CS	9,674	9,674	11,485	11,485
	EM	5,266	5,266	6,765	6,765

Tab. 3. Częstotliwość maksymalna [MHz]
Tab. 3. Maximum frequency [MHz]

Sieć testowa	Struktura CMCU	Częstotliwość maksymalna [MHz]			
		speed 1	speed 2	area 1	area 2
we_00	CS	183,736	183,736	167,636	167,636
	EM	271,201	271,201	202,220	202,220
we_01	CS	143,532	143,532	110,222	110,222
	EM	178,263	178,263	105,13	105,13
we_02	CS	137,120	137,120	105,641	105,641
	EM	174,981	174,981	139,910	139,910
we_03	CS	182,725	182,725	99,248	99,248
	EM	199,136	199,136	133,042	133,042
we_04	CS	126,624	127,455	75,998	75,998
	EM	167,645	167,645	108,327	108,327
we_05	CS	121,060	21,060	82,497	82,497
	EM	211,452	212,802	164,083	164,083
we_06	CS	302,686	308,966	293,966	293,966
	EM	187,269	187,269	146,020	146,021
we_07	CS	149,423	149,423	75,148	75,148
	EM	184,369	184,369	133,390	133,390
we_08	CS	144,605	149,714	136,168	136,168
	EM	156,602	156,602	151,410	151,410
we_09	CS	154,128	154,128	103,888	103,888
	EM	160,966	160,966	149,947	149,947
we_10	CS	103,371	103,371	87,074	87,074
	EM	189,879	189,879	147,829	147,829

Warto zwrócić uwagę na fakt, że dodanie kolejnych bloków (konwertera adresu, bloku LCS) do struktury mikroprogramowanego układu sterującego nie pociąga za sobą zwiększenia zużycia zasobów sprzętowych. Również zastosowanie różnych strategii optymalizacji nie wpływa znacząco na zużycie zasobów sprzętowych.

W tabeli 2 przedstawiono czasy trwania cyklu dla wybranych struktur zaimplementowanych w układzie FPGA.

W tabeli 3 przedstawiono częstotliwości maksymalne dla struktur zaimplementowanych w układzie FPGA.

Analizując tabelę 2 oraz 3 można stwierdzić, że zastosowanie konwertera adresu, powoduje wydłużenie czasu trwania cyklu, a co za tym idzie zmniejszenie częstotliwości maksymalnej.

4. Podsumowanie

W artykule zaprezentowano wyniki prac nad efektywnym wykorzystaniem zasobów sprzętowych w mikroprogramowanych układach sterujących z kodowaniem kolekcji mikrooperacji. Porównano dwa typy układów mikroprogramowanych: układ z konwerterem adresu oraz układ bez konwertera.

Przeprowadzone badania wykazały, że wprowadzenie bloku konwertera adresu do struktury mikroprogramowanego układu sterującego skutkuje, zmniejszeniem rozmiaru pamięci jednostki sterującej implementowanej jako mikroprogramowany układ sterujący. Warto zwrócić uwagę na fakt, że dodanie kolejnych bloków (konwertera adresu, układu LCS) do struktury mikroprogramowanych układów sterujących nie pociąga za sobą zwiększenia zużycia zasobów sprzętowych. Natomiast, zgodnie z oczekiwaniami, zastosowanie konwertera adresu, powoduje wydłużenie czasu trwania cyklu, a co za tym idzie zmniejszenie częstotliwości maksymalnej.

5. Literatura

[1] Barkalov A., Titarenko L., Kołopieńczyk M.: Optymalizacja jednostki sterującej poprzez zastosowanie metody współdzielenia kodów, Pomiary Automatyka Kontrola, nr 7, 2006.

[2] Bomar B.W.: Implementation of microprogrammed control in FPGAs, IEEE Transactions on Industrial Electronics, Vol. 49, 2002.

[3] Grushnitsky R., Mursaev A., Ugrjumov E.: Development of systems on chips with programmable logic, SPb: BHV, Petersburg, 2002.

[4] Gorzałczany M. B.: Układy cyfrowe metody syntezy. Tom II Układy sekwencyjne układy mikroprogramowane, Wydawnictwo Politechniki Świętokrzyskiej, Kielce, 2000.

[5] Kołopieńczyk M.: Application of Address Converter for Decreasing Memory Size of Compositional Mikroprogram Control Unit with Code Sharing: University of Zielona Góra Press, Zielona Góra, 2008.

[6] Łuba T.: Synteza układów cyfrowych, Praca zbiorowa pod redakcją prof. Tadeusza Łuby, WKŁ, Warszawa, 2003.

[7] Xilinx, Products and Services: www, Xilinx Corporation, http://www.xilinx.com/products.

[8] Barkalov A., Titarenko L.: Logic synthesis for FSM-based control units, Springer-Verlag, Berlin, 2009.

[9] Barkalov A., Titarenko L.: Logic synthesis for compositional microprogram control units, Springer-Verlag, Berlin, 2008.

[10] Adamski M, Barkalov A.: Architectural and sequential synthesis of digital devices, University of Zielona Góra, Zielona Góra, 2006.

otrzymano / received: 13.05.2010
przyjęto do druku / accepted: 01.09.2010 *artykuł recenzowany*