

Małgorzata KOŁOPIEŃCZYK
UNIwersytet Zielonogórski

Analiza zużycia zasobów sprzętowych w mikroprogramowanych układach sterujących ze współdzieleniem kodów

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 zaprezentowano wyniki prac nad zmniejszeniem zużycia zasobów sprzętowych w mikroprogramowanych układach sterujących ze współdzieleniem kodów. Porównano dwa typy układów mikroprogramowanych: układ z konwerterem adresu oraz układ bez konwertera. Do syntezy i implementacji struktur wykorzystano oprogramowanie Xilinx ISE 8.2i. Zastosowano cztery dostępne w pakiecie ISE strategie optymalizacji. Platformę docelową stanowił układ FPGA Xilinx Virtex-II Pro (xc2vp30-7ff896c). Badania wykazały, że wprowadzenie bloku konwertera adresu skutkuje, dla niektórych przypadków, co najmniej 50 % zmniejszeniem rozmiaru pamięci mikroprogramowanego układu sterującego ze współdzieleniem kodów.

Słowa kluczowe: sieć działań, mikroprogramowany układ sterujący, konwerter adresu

Analysis of resource utilization in compositional microprogram control unit with code sharing

Abstract

This paper presents results of decreasing hardware resources usage for microprogram control units with code sharing. Two types of microprogram control units are compared: a structure with address converter and without it. For synthesis and implementation of the microprogram control unit Xilinx ISE 8.2i package was used. Four optimization strategies available in the package ISE have been applied. Two strategies focus on the resources usage optimization (area level 1, area level 2); the other two - on the device maximum frequency (speed level 1, speed level 2). The target platform was the FPGA device Xilinx Virtex-II Pro xc2vp30-7ff896c). The test results show that for some examples memory consumption drops even by 50 % in comparison to the implementation without the address converter. It's important to notice that adding an additional block (address converter) to the microprogram control unit does not cause an increase in hardware resource use.

This paper has been divided into four parts. First chapter is a brief introduction to the issues of compositional microprogram control unit design. In second and third chapters the results of resource utilisation are presented. The last - fourth - chapter contains a summary.

Keywords: flow-chart, compositional microprogram control unit, address converter

1. Wstęp

Jednym z kryteriów branych pod uwagę przy projektowaniu systemów cyfrowych jest efektywne wykorzystanie zasobów sprzętowych oraz niski pobór energii. Jest to szczególnie ważne przy projektowaniu urządzeń przenośnych oraz przy produkcji masowej, gdzie redukcja kosztów jest jednym z kluczowych zadań.

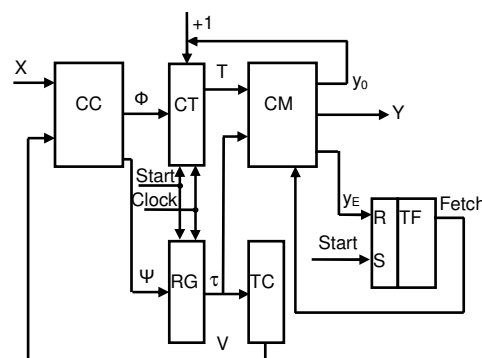
W artykule zaprezentowano wyniki prac nad zmniejszeniem zużycia zasobów sprzętowych w mikroprogramowanych układach

sterujących ze współdzieleniem kodów. W metodzie tej do opisu algorytmu sterowania wykorzystano sieć działań [1, 6].

Metoda współdzielenia kodów polega na przekształceniu adresu, reprezentowanego jako pary <kod łańcucha, kod elementu łańcucha>, w adres mikroinstrukcji.

Metoda ta umożliwia minimalizację rozmiaru pamięci niezależnie od charakterystyki implementowanego algorytmu sterowania. Szczegółowy opis metody współdzielenia kodów można znaleźć w pracy [5].

W niniejszym artykule porównano dwa typy układów mikroprogramowanych: układ z konwerterem adresu (ang. Compositional Microprogram Control Unit with Code Sharing and Address Converter, CMCU_AT) oraz układ bez konwertera (Compositional Microprogram Control Unit with Code Sharing, CMCU_CS) (Rys. 1).



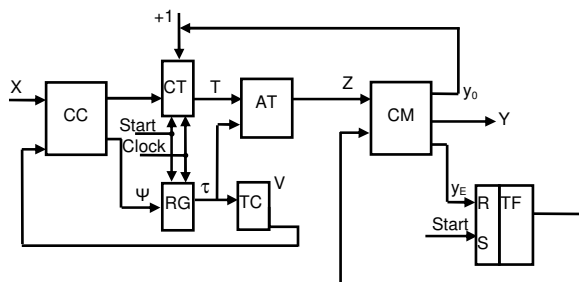
Rys. 1. Mikroprogramowany układ sterujący ze współdzieleniem kodów
Fig. 1. Structural diagram of microprogram control unit with code sharing

Mikroprogramowany układ sterujący ze współdzieleniem kodów działa w następujący sposób: w chwili aktywacji układu (aktywny impuls Start), do licznika CT oraz rejestru RG wczytywany jest kod zero, który odpowiada adresowi pierwszej mikroinstrukcji realizowanego algorytmu. W tym samym czasie ustawiany jest przerzutnik TF i sygnał Fetch=1 umożliwia pobranie mikroinstrukcji z pamięci CM. Bieżąca mikroinstrukcja zostaje odczytana z pamięci. Jeżeli kolejny blok w sieci działań należy do tego samego łańcucha, to zmienna $y_0=1$. Oznacza to zwiększenie wartości licznika ($CT=CT+1$), natomiast zawartość rejestru RG nie ulega zmianie. Taka sytuacja odpowiada adresowaniu kolejnego elementu bieżącego łańcucha sieci działań. Jeżeli zostanie osiągnięte wyjście łańcucha, to zmienna $y_0=0$. Wówczas do rejestru RG ładowana jest wartość funkcji Ψ określająca kod następnego łańcucha, a do licznika CT wartość funkcji Φ określająca kod jego wejścia [5].

Zawartość licznika CT oraz rejestru RG zostaje zmieniona za pomocą impulsu synchronizującego Clock. Jeżeli osiągnięty został blok końcowy sieci działań, to wówczas wewnętrzna zmienna $y_E=1$, co powoduje, zresetowanie przerzutnika TF (Fetch=0) a tym samym zakończenie działania mikroprogramowanego układu sterującego.

Zasada działania mikroprogramowanego układu sterującego z konwerterem adresu (Rys. 2) jest taka sama jak układu bez konwertera.

Różnica polega na wprowadzeniu dodatkowego bloku - konwertera adresu, do struktury układu mikroprogramowanego. Konwerter adresu przekształca adres mikroinstrukcji na adres o minimalnej pojemności bitowej.



Rys. 2. Mikroprogramowany układ sterujący ze współdzieleniem kodów i konwerterem adresu

Fig. 2. Structural diagram of microprogram control unit with code sharing and address converter

Do syntezy i implementacji struktur wykorzystano oprogramowanie Xilinx ISE 8.2i [8]. Do testowania rozmiaru pamięci omawianych struktur mikroprogramowanych wykorzystano oprogramowanie fca2cmcu [6].

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.

2. Rozmiar pamięci

Zestaw testowy zawierał 20, zróżnicowanych pod względem ilości bloków operacyjnych, bloków warunkowych oraz mikroinstrukcji, sieci działań, co daje 40 różnych struktur układów mikroprogramowanych. Poniżej przedstawione zostały rozmiary pamięci mikroprogramowanych układów sterujących wygenerowanych przez program fca2cmcu na podstawie zestawu sieci testowych.

Tab. 1. Rozmiar pamięci jednostki sterującej [bity]

Tab. 1. The size of control memory of control unit [bits]

Sieć testowa	Układ mikroprogramowany	
	bez konwertera adresu	z konwerterem adresu
we_00	512	256
we_01	2304	1152
we_02	2304	1152
we_03	1280	640
we_04	2560	640
we_05	2816	704
we_06	1664	832
we_07	2560	1280
we_08	1152	576
we_09	2304	576
we_10	3328	1664
we_11	1280	640
we_12	2560	1280
we_13	1152	576
we_14	1152	576
we_15	1664	1664
we_16	3328	1664
we_17	1152	1152
we_18	1664	832
we_19	1152	576

Na podstawie tabeli 1 można stwierdzić, że zastosowanie konwertera adresu w strukturze mikroprogramowanego układu sterującego skutkuje zmniejszeniem rozmiaru pamięci jednostki sterującej implementowanej jako mikroprogramowany układ sterujący. Należy zauważyć, że dla niektórych przypadków pamięć została zmniejszona aż czterokrotnie (sieć testowa we_4) w stosunku do układu bez konwertera adresu.

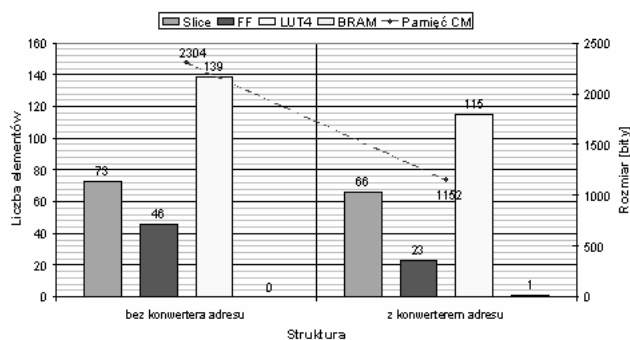
3. Analiza zużycia zasobów sprzętowych

Do syntezy i implementacji układów mikroprogramowanych wykorzystano oprogramowanie Xilinx ISE 8.2i. Zastosowano cztery dostępne w pakiecie ISE strategie optymalizacji. Dwie z nich jako podstawowe kryterium wykorzystują ilość zużytych zasobów (area level 1, area level 2), dwie kolejne – maksymalną częstotliwość pracy układu (speed level 1, speed level 2). Platformę docelową stanowił układ FPGA Xilinx Virtex-II Pro (xc2vp30-7ff896c).

Układ xc2vp30 zawiera w swojej strukturze następujące zasoby sprzętowe [8]:

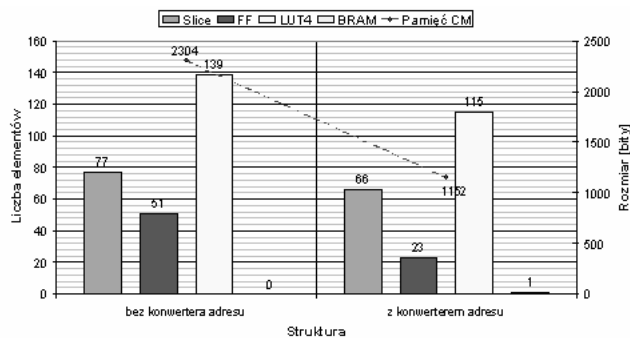
- 13696 „plastrów” (Slice),
- 27392 przerzutników,
- 27392 czterowjęściowych elementów LUT,
- 136 dedykowanych bloków pamięci (Block RAM).

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



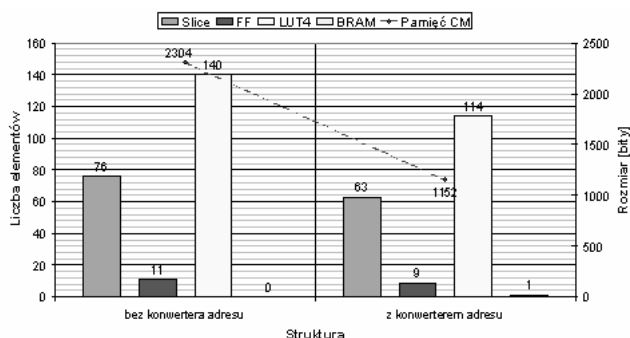
Rys. 3. Zużycie zasobów sprzętowych dla sieci testowej we_01 (speed level 1)

Fig. 3. Resource utilization for flow-chart we_01 (speed level 1)



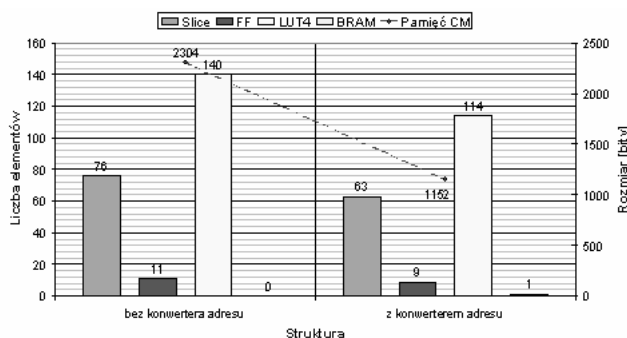
Rys. 4. Zużycie zasobów sprzętowych dla sieci testowej we_01 (speed level 2)

Fig. 4. Resource utilization for flow-chart we_01 (speed level 2)



Rys. 5. Zużycie zasobów sprzętowych dla sieci testowej we_01 (area level 1)

Fig. 5. Resource utilization for flow-chart we_01 (area level 1)



Rys. 6. Zużycie zasobów sprzętowych dla sieci testowej we_01 (area level 2)
Fig. 6. Resource utilization for flow-chart we_01 (area level 2)

Na podstawie wykresów przedstawionych na powyższych rysunkach można stwierdzić, że ilość elementów LUT (ang. look-up table) w mikroprogramowanym układzie z konwerterem adresu jest nie większa niż w układzie bez bloku konwertera adresu.

Wykorzystanie identycznej liczby bloków BRAM spowodowane jest 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.

Warto zwrócić uwagę na fakt, że dodanie kolejnych bloków (konwertera adresu) 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.

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
	AT	2,528	2,528	3,685	3,685
we_01	CS	6,967	6,967	9,073	9,073
	AT	3,269	3,269	4,205	4,205
we_02	CS	7,293	7,293	9,466	9,466
	AT	3,248	3,248	4,562	4,562
we_03	CS	5,473	5,473	10,076	10,076
	AT	2,462	2,462	3,498	3,498
we_04	CS	7,897	7,846	13,158	13,158
	AT	2,802	2,802	3,758	3,758
we_05	CS	8,260	8,260	12,122	12,122
	AT	2,679	2,679	4,220	4,220
we_06	CS	3,237	3,237	3,402	3,402
	AT	3,370	3,370	3,635	3,635
we_07	CS	6,692	6,692	13,307	13,307
	AT	3,598	3,598	6,026	6,026
we_08	CS	6,915	6,679	7,344	7,344
	AT	2,533	2,533	3,841	3,841
we_09	CS	6,488	6,488	9,626	9,626
	AT	3,124	3,124	4,676	4,676
we_10	CS	9,674	9,674	11,485	11,485
	AT	4,219	4,219	5,825	5,825

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

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
	AT	395,515	395,515	271,37	271,370
we_01	CS	143,532	143,532	110,222	110,222
	AT	305,941	305,941	237,821	237,821
we_02	CS	137,120	137,120	105,641	105,641
	AT	307,926	307,926	219,194	219,194

we_03	CS	182,725	182,725	99,248	99,248
	AT	406,248	406,248	285,874	285,874
we_04	CS	126,624	127,455	75,998	75,998
	AT	356,837	356,837	266,098	266,098
we_05	CS	121,060	121,060	82,497	82,497
	AT	373,336	373,336	236,965	236,965
we_06	CS	308,966	308,966	293,966	293,966
	AT	296,718	296,718	275,126	275,126
we_07	CS	149,423	149,423	75,148	75,148
	AT	277,947	277,947	165,937	165,937
we_08	CS	144,605	149,714	136,168	136,168
	AT	394,851	394,851	260,348	260,348
we_09	CS	154,128	154,128	103,888	103,888
	AT	320,061	320,061	213,861	213,861
we_10	CS	103,371	103,371	87,074	87,074
	AT	237,043	237,043	171,683	171,683

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

4. Wnioski

Klasyczna metoda implementacji jednostki sterującej w postaci skończonego automatu stanów często pochłania zasoby układów programowalnych, które mogą być w efektywniejszy sposób wykorzystane przez inne bloki projektowanego systemu [2]. Jednym ze sposobów rozwiązaniem tego problemu jest implementacja jednostki sterującej jako mikroprogramowany układ sterujący [3, 4, 7].

W artykule zaprezentowano wyniki prac nad zmniejszeniem zużycia zasobów sprzętowych w mikroprogramowanych układach sterujących ze współdzieleniem kodów. 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 ze współdzieleniem kodów skutkuje, co najmniej dwukrotnym zmniejszeniem rozmiaru pamięci jednostki sterującej implementowanej jako mikroprogramowany układ sterujący w stosunku do układu bez konwertera adresu.

Należy podkreślić fakt, że wprowadzenie bloku konwertera adresu 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] A. Barkalov, L. Titarenko, M. Kołopieńczyk: Optymalizacja jednostki sterującej poprzez zastosowanie metody współdzielenia kodów. Pomiary Automatyka Kontrola 2006, nr 7, wyd. spec., s. 29-31.
- [2] B.W. Bomar: Implementation of microprogrammed control in FPGAs, IEEE Transactions on Industrial Electronics, 2002, Vol. 49, Issue 2, s. 415-422.
- [3] R. Grushnitsky, A. Mursaev, E. Ugrjumov: Development of systems on chips with programmable logic. – Spb: BHV – Petersburg, 2002 (in Russian)– 626 p.
- [4] M. B. Gorzałczany: Układy cyfrowe metody syntezy. Tom II Układy sekwencyjne układy mikroprogramowane. Wydawnictwo Politechniki Świętokrzyskiej, Kielce 2000.
- [5] M. Kołopieńczyk: Zastosowanie konwertera adresów do zmniejszenia rozmiaru pamięci mikroprogramowanego układu sterującego ze współdzieleniem kodów, Rozprawa doktorska, Uniwersytet Zielonogórski, Zielona Góra 2007.
- [6] M. Kołopieńczyk: Program konwertujący tekstową sieć działań do struktur mikroprogramowanych układów sterujących ze współdzieleniem kodów. Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne 2008, nr 6, s. 809-810.
- [7] T. Łuba: Synteza układów cyfrowych, Praca zbiorowa pod redakcją prof. Tadeusza Łuby Warszawa: WKŁ 2003, s. 296.
- [8] Xilinx: Products and Services. www. Xilinx Corporation, Available at <http://www.xilinx.com/products>.