Alexander BARKALOV, Larysa TITARENKO, **Sławomir CHMIELEWSKI**
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI, ul. licealna 9, 65-417 Zielona Góra

Metody syntezy mikroprogramowalnego układu sterującego z wykorzystaniem pseudorównoważnych stanów

prof. zw. dr hab. inż. Alexander BARKALOV

Prof. Alexander A. Barkalov w latach 1976 - 1996 był pracownikiem dydaktycznym w Instytucie Informatyki Narodowej Politechniki Donieckiej. Współpracował aktywno z Instytutem Cybernetyki im. V. M. Glushkova w Kijowie, gdzie uzyskał tytuł doktora habilitowanego ze specjalnością informatyka. W latach 1996 - 2003 pracował jako profesor w Instytucie Informatyki Narodowej Politechniki Donieckiej. Od 2004 pracuje jako profesor na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: A.Barkalov@iie.uz.zgora.pl

mgr inż. Sławomir CHMIELEWSKI

Mgr inż. Sławomir Chmielewski absolwent Uniwersytetu Zielonogórskiego. Ukończył studia informatyczne o specjalizacji inżynieria komputerowa. Zainteresowania naukowe koncentrują się wokół nowoczesnych metod projektowania specjalistycznych układów cyfrowych..

e-mail: S.Chmielewski@weit.uz.zgora.pl

Streszczenie

W artykule przedstawiono metody syntezy mikroprogramowalnego układu sterującego z użyciem wbudowanych bloków pamięci. Metody są ukierunkowane na zmniejszenie rozmiaru układu sterującego poprzez zastosowanie transformacji kodów klas pseudorównoważnych w pamięci. Podejście takie pozwala uzyskać uproszczoną formę funkcji przejścia części adresowej układu, dzięki któremu możliwa jest redukcja zasobów sprzętowych potrzebnych do implementacji jednostki sterującej w układach programowalnych typu CPLD bez zmniejszenia wydajności systemu cyfrowego. W artykule zamieszono wprowadzenie teoretyczne, przykład oraz wyniki badań uzyskanych podczas syntezy testowych sieci działań przy użyciu oprogramowania Xilinx ISE 10.2 dla układu XC9500 CPLD (xc9536-5PC44).

Słowa kluczowe: automat Moore'a, mikroprogramowalny układ sterujący.

Methods of synthesis of control unit with using pseudoequivalent states

Abstract

Method of decrease in the number of programmable array logic (PAL) macrocells in logic circuit of Moore finite-state-machine (FSM) is proposed. Programmable logic devices are nowadays widely used for implementation of Control Units (CU). The problem of the optimization of CU is still actual in computer science and it solution permits to decrease the cost of the system. This method is based on the use of free outputs of embedded memory blocks to represent the code of the class of pseudoequivalent states. The proposed approach allows minimizing the hardware without decreasing of the digital system performance. An example of application of the proposed method is given. Control unit of any digital system can be implemented as the Moore FSM. Recent achievements in semiconductor technology have resulted in development of such sophisticated VLSI chips as field-programmable logic arrays (FPGA) and complex programmable logic devices (CPLD). Very often CPLD are used to implement complex controllers. In CPLD, logic functions are implemented using programmable array logic macrocells. One of the issues of the day is decrease in the number of PAL macrocells required for implementation of FSM logic circuit. A proper state assignment can be used to solve this problem. The peculiarities of Moore FSM are existence of pseudoequivalent states and dependence of microoperations only on FSM internal states. The peculiarity of CPLD is a wide fan-in of PAL macrocell. It permits to use different sources for representation of a current state code.

Keywords: Moore finite-state-machine, programmable logic device.

dr hab. inż. Larysa TITARENKO, prof. UZ

Dr hab. Larysa Titarenko w 2004 roku obroniła rozprawę habilitacyjną i uzyskała tytuł doktora habilitowanego ze specjalnością telekomunikacja. W latach 2004 - 2005 pracowała jako profesor w Narodowym Uniwersytecie Radioelektroniki w Charkowie. Od 2007 pracuje jako profesor na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: L.Titarenko@iie.uz.zgora.pl

1. Wprowadzenie

Jednostka sterująca (ang. Control Unit, CU) we wszystkich systemach cyfrowych może zostać zaprojektowana, jako skończony automat stanów (ang. Finite State Machine, FSM) [1, 2]. Aktualnie programowalne układy cyfrowe są bardzo często stosowane do realizacji układu logicznego automatu FSM [3]. Postęp w technologii półprzewodnikowej powoduje pojawienie się coraz to bardziej złożonych układów cyfrowych (ang. Very Large Scale Integration, VLSI), takich jak złożone programowalne układy cyfrowe (ang. Complex Programmable Logic Devices, CPLD), gdzie funkcje logiczne są implementowane przy użyciu programowalnych bloków logicznych (ang. Programmable Array Logic, PAL) [4, 5]. Obecnie jedną z istotnych kwestii w przypadku implementowania automatów FSM przy zastosowaniu układów CPLD jest zmniejszenie liczby zużycia makrokomórek PAL [6].

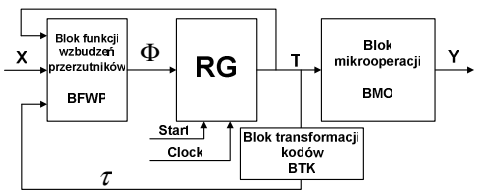
2. Proponowane metody

Niech $\Pi_A = \{B_1,...,B_I\}$ będzie zbiorem części A klasy pseudorównoważnych stanów. Tworząc zbiór $\Pi_A = \Pi_B \cup \Pi_C$, gdzie $B_i \in \Pi_B$, jeżeli $|B_i| \geq 2$, a w przeciwnym wypadku $B_i \in \Pi_C$. Następnie kodujemy $B_i \in \Pi_B$ kodem binarnym dwójkowym $K(B_i)$ używając

$$R_1 = \lceil \log_2 I_B \rceil \quad (1)$$

bitów $\tau_r \in \tau$, gdzie $I_B = |\Pi_B|$.

Blok BFWP jest optymalizowany dzięki istnieniu pseudorównoważnych stanów, przez co dany stan automatu przypisywany jest do kodu odpowiedniej klasy obiektu. Jako obiekt rozumiany jest wewnętrzny stan automatu i zbiór mikrooperacji. Schemat blokowy automatu Moore'a U_2 pokazana jest na rysunku 1, gdzie U_1 przyjmuje się jako klasyczny automat Moore'a.



Rys. 1. Schemat blokowy automatu Moore'a U_2
Fig. 1. Structural diagram of Moore U_2

Dla skończonego automatu U_2, blok BFWP implementuje funkcje

$$\Phi = \Phi(T, \tau, X), \quad (2)$$

blok transformacji kodów (BTK) przekształca kody pseudorównoważnych stanów $a_m \in B_i$ w kod klasy $K(B_i)$. Blok BTK generuje funkcję

$$\tau = \tau(T). \quad (3)$$

Kodując stany $a_m \in A$ w optymalny sposób możemy podzielić zbiór $\Pi_B = \Pi_D \cup \Pi_E$. Gdzie $B_i \in \Pi_D$, jeżeli kod stanu $a_m \in B_i$ należy do pojedynczego argumentu logicznego przestrzeni Boolowskiej, pozostałe kody stanów z zbioru Π_B przypisujemy do zbioru Π_E.

Następnie obliczamy liczbę bitów potrzebnych do zakodowania stanów dla zbioru $B_i \in \Pi_E$:

$$R_2 = \lceil \log_2(I_E + 1) \rceil, \quad (4)$$

gdzie $I_E = |\Pi_E|$.

Oznaczmy t_F jako ustaloną liczbą wyjść bloku wewnętrznej pamięci (ang, Embedded Memory Blocks, EMB*)* oraz niech q jest ilością jego słów. Wartość t_F dla FSM U_1 jest określona przez

$$t_F = \lceil M / q \rceil. \quad (5)$$

Całkowita liczba wyjść t_S całego bloku EMB w układzie BMO jest zdefiniowana jako

$$t_S = \lceil N / t_F \rceil * t_F. \quad (6)$$

W tym przypadku

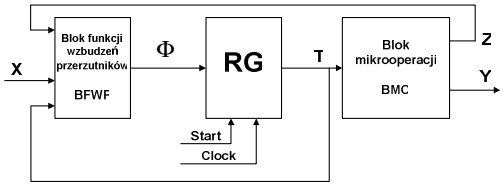
$$\Delta_t = t_S - N, \quad (7)$$

wyjścia nie są w użyciu, w celu reprezentowania mikrooperacji $y_n \in Y$. Oznaczmy $z_r \in Z$ dla takiego kodowania, gdzie $|Z| = R_2$.

W przypadku, kiedy warunek

$$\Delta_t \geq R_2, \quad (8)$$

będzie spełniony, wtedy sieć działań może być interpretowana przez proponowany automat Moore'a U_3 o strukturze pokazanej na rysunku 2.



Rys. 2. Schemat blokowy automatu Moore'a U_3
Fig. 2. Structural diagram of Moore U_3

Dla skończonego automatu U_3, blok BFWP realizuje następujący system funkcji

$$\Phi = \Phi(T, Z, X), \quad (9)$$

działanie układu jest zgodne z funkcją $Y = Y(T)$ oraz

$$Z = Z(T). \quad (10)$$

W przypadku kiedy warunek

$$\Delta_t < R_2, \quad (11)$$

należy zbiór Π_E podzielić na dwa zbiory $\Pi_E = \Pi_F \cup \Pi_G$, gdzie Π_F zawiera n_F liczbę klas:

$$n_F = 2^{\Delta_t} - 1. \quad (12)$$

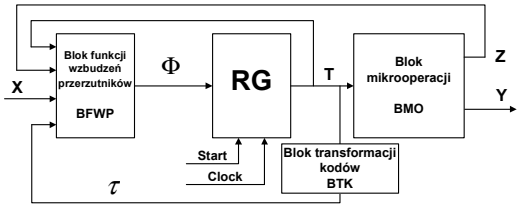
Kody z zbioru Π_F będą zapisane razem z mikrooperacjami $y_n \in Y$, które będą reprezentowane przez zmienną $z_r \in Z$, gdzie $|Z| = \Delta_t$. Zbiór stanów Π_G zawiera

$$n_G = I - n_C - n_D - n_F \quad (13)$$

liczbę klas, gdzie $n_C = |\Pi_C|$, $n_D = |\Pi_D|$, które będą zakodowane na zmiennych $\tau_r \in \tau$, gdzie $|\tau| = R_3$

$$R_3 = \lceil \log_2(n_G + 1) \rceil \quad (14)$$

Dla spełnionego warunku (11) sieć działań może zostać zaproponowana dla automatu Moore'a U_4 o strukturze pokazanej na rysunku 3.



Rys. 3. Schemat blokowy automatu Moore'a U_4
Fig. 3. Structural diagram of Moore U_4

Dla skończonego automatu U_4, blok BFWP realizuje następujący system funkcji

$$\Phi = (T, \tau, Z, X), \quad (15)$$

działanie układu jest zgodne z funkcją $Y = Y(T)$ oraz

$$Z = Z(T). \quad (16)$$

3. Podsumowanie

Zaprezentowane metody pozwalają zredukować zużycie zasobów sprzętowych wymaganych do realizacji automatu Moore'a, dzięki lepszemu wykorzystaniu wbudowanych bloków pamięci. Metoda ta bazuje na transformacji kodu stanu w kod klasy pseudorównoważnych stanów automatu Moore'a oraz wykorzystaniu wolnej przestrzeni pamięci dla zakodowania stanów, które prowadzą do zmniejszenia kosztów realizowanego układu.

Przeprowadzone badania eksperymentalne dla układów typu CPLD serii XC9500 pokazały, że proponowana metoda wykorzystuje mniejszą liczbę makrokomórek PAL, niż ma to miejsce w klasycznych metodach projektowania automatów Moore'a. Redukcja rozmiaru układu może sięgać ponad 35%, natomiast najczęściej oscyluje w granicach od 15% do 30% i uzależniona jest od własności sieci działań.

Zmniejszenie zużycia zasobów sprzętowych wiąże się z zmniejszeniem układów kombinacyjnych w danym systemie. Dzięki czemu uzyskujemy mniejszą liczbę cykli automatu, a co za tym idzie zwiększenie wydajności.

Dalsze prace autorów będą skupiały się nad sprawdzeniem zaproponowanej metody dla układów typu FPGA oraz nad sprawdzeniem kolejnych przykładów sieci działań z bazy testowej [7].

4. Literatura

[1] S. Baranov, Logic and System Design of Digital Systems. Tallinn: TUT Press, 2008.
[2] A. Barkalov and M. Węgrzyn, Design of Control Units with Programmable Logic, Zielona Gora: University of Zielona Gora Press, 2006.
[3] A. Barkalov, L. Titarenko, Logic Synthesis for FSM – based Control Units. Berlin: Springer, str. 233, 2009.
[4] V. Solovjev, A. Klimowicz, Logic design of digital Systems with programmable logic devices, Moscow: Hotline – Telecom, str. 376, 2008.
[5] G. De Micheli, Synthesis and Optimization of Digital Circuits, N.Y.: McGraw Hill, 1994.
[6] D. Kania, Synteza logiczna przeznaczona dla matrycowych struktur programowalnych typu PAL, Gliwice: Silesian Technical University, 2004.
[7] S. Yang, Logic Synthesis and Optimization Benchmarks User Guide, Microelectronics Center of North Carolina, Research Triangle Park, North Carolina, 1991.

otrzymano / received: 00.00.0000
przyjęto do druku / accepted: 00.00.0000 *artykuł recenzowany*