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

Wykorzystanie własności układu sterującego w układach CPLD

Prof. dr hab. inż. Alexander BARKALOV

Prof. Alexander A. Barkalov w latach 1976-1996 był pracownikiem dydaktycznym w Instytucie Informatyki Narodowej Politechniki Donieckiej. Współpracował aktywnie 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

Dr hab. inż. Larysa TITARENKO

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 2005 pracuje na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego.

e-mail: l.titarenko@iie.uz.zgora.pl

Streszczenie

W artykule przedstawiono metodę syntezy mikroprogramowalnego układu sterującego z użyciem wbudowanych bloków pamięci, która jest ukierunkowana 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.

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

Use of control unit properties in CPLD systems

Abstract

A method for decreasing the number of programmable array logic (PAL) macrocells in a logic circuit of the Moore finite-state-machine (FSM) is proposed. Programmable logic devices are nowadays widely used for implementation of control units (CU). The problem of CU optimization is still actual in computer science and its solution enables reduce the cost of the system. This method is based on 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 the digital system performance. An example of application of the proposed method is given. A 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 the 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 allows using different sources for representation of a current state code.

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

Mgr inż. Sławomir CHMIELEWSKI

Mgr inż. Sławomir Chmielewski absolwent Uniwersytetu Zielonogórskiego. Ukończył studia informatyczne o specjalizacji inżynieria komputerowa. Uczestnik czwartego roku na studiach doktoranckich.

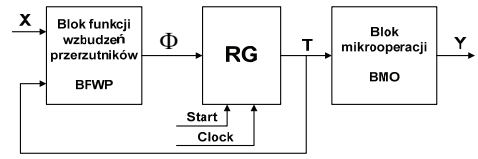
e-mail: S.Chmielewski@weit.uz.zgora.pl

1. Wstęp

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. Opis projektowania automatów Moore'a

Jednym z możliwych form opisu automatu Moore'a jest struktura tablicy [1], w której a_m jest początkowym stanem automatu, $a_m \in A$, gdzie $A = \{a_1,...,a_M\}$ jest zbiorem stanów automatu; $K(a_m)$ jest kodem stanu a_m zapisanym na $R = \lceil \log_2 M \rceil$ bitach; a_s jest następnym stanem automatu; $K(a_s)$ jest kodem stanu $a_s \in A$; X_h jest koniunkcją zmiennych ze zbioru wejściowych warunków cyfrowych $X = \{x_1,...,x_L\}$, wymuszając przejście $\langle a_m, a_s \rangle$; Φ_h jest zbiorem wejściowych funkcji wzbudzeń pamięci, które są równe 1 w celu przełączenia pamięć automatu z kodu $K(a_m)$ na kod $K(a_s)$, gdzie $\Phi = \{D_1,...,D_R\}$; h jest numerem przejścia automatu ($h = 1$, …, H). Stan a_m zawiera zbiór mikrooperacji $Y(a_m) \subseteq Y$, gdzie $Y = \{y_1,...,y_N\}$. Ustalona tablica automatu Moore'a U_1 pokazana jest na rysunku 1.



Rys. 1. Struktura diagramu Moore'a FSM U_1
Fig. 1. Structural diagram of Moore FSM U_1

Na tej podstawie wyprowadzany jest blok funkcji wzbudzeń przerzutników (BFWP)

$$\Phi = \Phi(T, X) \tag{1}$$

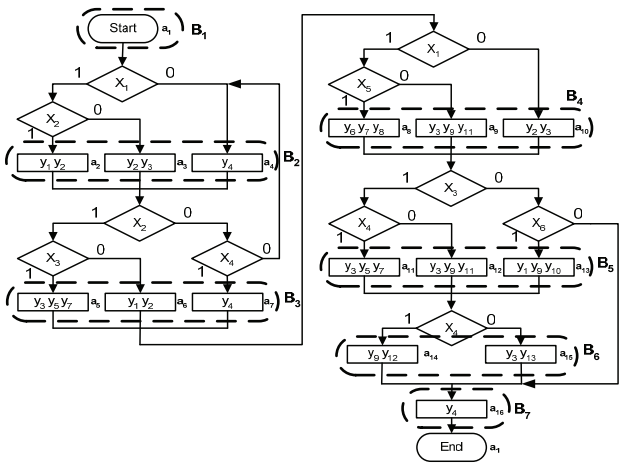
i mikrooperacji (BMO)

$$Y = Y(T), \tag{2}$$

gdzie $T = \{T_1,...,T_R\}$ jest zbiorem zmiennych wewnętrznych kodujących stany $a_m \in A$. Sygnał „Start" jest zastosowany do załadowania stanu początkowego $a_1 \in A$ do rejestru (ang. ReGister, RG), który służy do zapamiętania kodu $K(a_m)$ stanu $a_m \in A$. Sygnał „Clock" przełącza stan RG.

Stan a_s, gdzie $a_s \in A$ jest pseudorównoważnym stanem, jeżeli stan ten jest kolejnym stanem automatu dla wyjść poprzedzających go stanów. W takim przypadku mamy udoskonaloną metodę wykorzystania liczby stanów [2], każda mikrooperacja $y_n \in Y$ jest reprezentowana przez *R*-wymiarową przestrzeń boolowską.

Na rysunku 2 bloki pseudorównoważne zaznaczono linią przerywaną.



Rys. 2. Sieć działań Γ
Fig. 2. Graph scheme of algorithm Γ

3. Idea proponowanej metody

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

$$R_1 = \lceil \log_2 I_B \rceil \tag{3}$$

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

Specyficzną cechą układu typu PAL jest duża ilość wejść makrokomórek i ilość termów na mikrokomórkę [2, 3], dlatego też możliwe jest użycie więcej niż jednego źródła kodowania stanów dla bloku BFWP [4].

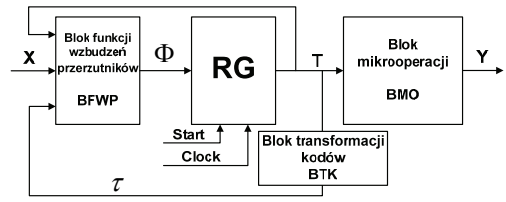
Blok BFWP jest optymalizowany dzięki istnieniu pseudorównoważnych stanów, przez co dany stan automatu przypisywany jest do kodu odpowiedniej klasy obiektu przy użyciu makrokomórek PAL. Jako obiekt rozumiany jest wewnętrzny stan automatu i zbiór mikrooperacji. To jest możliwe dla automatów Moore'a U_2 o strukturze pokazanych na rysunku 3.

Dla skończonego automatu U_2, blok BFWP implementuje funkcje

$$\Phi = \Phi(T, \tau, X), \tag{4}$$

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

$$\tau = \tau(T). \tag{5}$$



Rys. 3. Struktura diagramu Moore'a U_2
Fig. 3. Structural diagram of Moore U_2

Dzięki takiemu podejściu możliwe jest zmniejszenie liczby użytych makrokomórek w obu blokach BMO i BTK. Pozwala to również na to, aby była ona przedstawiona za pomocą sieci działań algorytmu GSA [1].

Metoda składa się z następujących etapów projektowania:

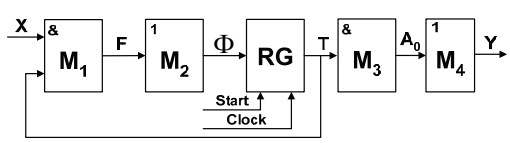
- Utworzenie tablicy przejścia automatu Moore'a (bez kodowania stanów).
- Przydzieleniu stanów do odpowiednich klas Π_A, Π_B i Π_C.
- Kodowanie poszczególnych klas $B_i \in \Pi_B$.
- Minimalizacja stanów w funkcji (2) i (5).
- Utworzenie tabeli transformacji, gdzie stan obecny $a_m \in B_i$ jest zastąpiony przez odpowiednią klasę $B_i \in \Pi_A$, a kod stanu $K(a_m)$ jest zastąpiony przez odpowiedni kod $K(B_i)$.
- Utworzenie tabeli dla bloku BTK i funkcji (5).
- Utworzenie funkcji (2) i (4).
- Zaimplementowanie automatu Moore'a U_2 w układzie cyfrowym przy użyciu funkcji (2), (4) i (5) oraz zastosowaniu dedykowanego układ CPLD.

4. Analiza proponowanej metody

Analizując proponowaną metodę można wykazać, że zajętość zasobów sprzętowych w proponowanym układzie U_2 (rys. 3) jest mniejsza niż w układzie U_1 (rys. 1). Korzystając z opisu jaki jest w [2], możemy określić trzy główne podziały:

1. Rozpatrywanie klas sieci działań algorytmu Γ zamiast poszczególnych jego stanów. Każda klasa charakteryzuje się przez parametr p_1, który określa wartość z przedziału od 0 do 1 wystąpienia stanów operacyjnych w porównaniu do wystąpienia stanów warunkowych.
2. Użycie automatów stanów FSM [1] zamiast implementacji w standardowych układach VLSI.
3. Zbadanie zależności $S(U_2)/S(U_1)$, gdzie $S(U_1)$, $S(U_2)$ jest parametrem określającym odpowiedni układ U_1 lub U_2. Takie podejście pozwala zaimplementować rozwiązanie w układach programowalnych [2] takich jak: PAL, PLA lub PROM.

Struktura automatu Moore'a U_1 została przedstawiona na rysunku 4.



Rys. 4. Struktura automatu Moore'a FSM U_1
Fig. 4. Matrix realization of Moore FSM U_1

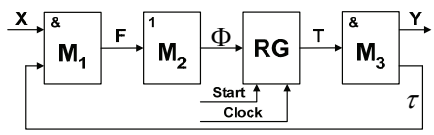
M_1 jest zbiorem funkcji logicznych AND implementująca system F wyrażenia systemu (1); M_2 jest zbiorem funkcji logicz-

nych OR implementująca system (1); M_3 jest funkcją logiczną AND implementująca system A_0, gdzie każda funkcja koresponduje z koniunkcją $A_m(m=1,...,M)$, poszczególny kod $K(a_m)$ stanu $a_m \in A$; M_4 jest dysjunkcją dla implementacji funkcji (2). Bloki M_1 i M_2 reprezentują układ BFWP, natomiast bloki M_3 i M_4 reprezentują układ BMO. Złożoność tych układów mogą być przedstawione w następujący sposób jako:

$$S(BIMF)_1 = 2(L+R)\cdot H + H\cdot R\,; \tag{6}$$

$$S(BMO)_1 = 2\cdot R\cdot M + M\cdot N\,. \tag{7}$$

Struktura automatu Moore'a U_2 została przedstawiona na rysunku 5.



Rys. 5. Struktura automatu Moore'a FSM U_2
Fig. 5. Matrix realization of Moore FSM U_2

W przypadku U_2, parametr F zawiera H_0 elementów, które określają liczbę tranzycji dla równoważnego automatu Mealy'ego [7]. Zakładając, że każda funkcja (2) i (5) jest implementowana przy użyciu średniej k - PAL makrokomórek. Oba bloki BMO i BTK reprezentują blok M_3. Złożoność takiego układu może być wyrażona jako:

$$S(BIMF)_2 = 2(L+R_1)\cdot H_0 + H_0\cdot R\,; \tag{8}$$

$$S(BMO)_2 = 2R\left(N+R_1\right)\cdot k\,. \tag{9}$$

Analizując przedstawione funkcje otrzymujemy zależność:

$$f = \frac{S(BIMF)_2 + S(BMO)_2}{S(BIMF)_1 + S(BMO)_1}\,. \tag{10}$$

Biorąc pod uwagę funkcję f (10), gdzie parametry L, R, H, H_0, R_1 określone są jako funkcje [2]:

$$L = (1-p_1)\cdot K/1{,}3\,; \tag{11}$$

$$R = \lceil \log_2 p_1\cdot K \rceil; \tag{12}$$

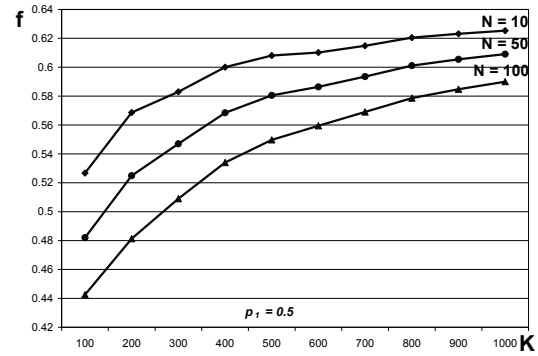
$$H = 17{,}4 + 1{,}7\cdot p_1\cdot K\,; \tag{13}$$

$$H_0 = 4{,}4 + 1{,}1\cdot p_1\cdot K\,; \tag{14}$$

$$R_1 = \lceil \log_2(2{,}75 + 0{,}34\cdot p_1\cdot K) \rceil. \tag{15}$$

W wyrażeniu (11) - (15) parametr K jest równy liczbie stanów w sieci działań algorytmu Γ. Przykładowe wyniki zostały zaprezentowane na rysunku 5.

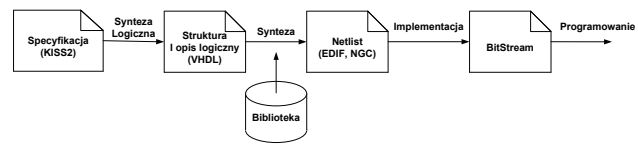
Analizując rysunek 6 dostrzec można, że proponowana metoda daje obiecujące wyniki, które są nie gorsze niż w klasycznych automatach Moore'a. W raz ze wzrostem liczby stanów w sieci działań algorytmu Γ dostrzega się zysk w zużyciu zasobów sprzętowym. Średnie wykorzystanie zasobów sprzętowych dla sieci działań, gdzie $K(\Gamma)=500$ wynosi 40%.



Rys. 6. Porównanie modelu U_2 i U_1
Fig. 6. Comparison of models U_2 and U_1

5. Badania eksperymentalne

Badania skuteczności zaproponowanej metody wykonano na podstawie 30 przykładów sieci działań, które zostały porównane z standardowym automatem Moore'a. Badania zostały przeprowadzone na układzie CPLD rodziny XC9500 (xc9536-5PC44). W celu przeprowadzenia badań napisany został program w języku C#, który w pierwszej kolejności przekształca automat Mealy'ego reprezentowane w formacie KISS2 do automatów Moore'a w tym samym formacie. Kolejnym etapem jest utworzenie programu w języku VHDL i podaniu go syntezie przy użyciu oprogramowania Xilinx ISE 10.2 do konkretnego układu. Każdy przykład jest wykonywany w sposób automatyczny (rys. 7).



Rys. 7. Schemat działania algorytmu
Fig. 7. Diagram of the algorithm

W tabeli 3 przedstawione zostały 10 przykładowych otrzymanych rezultatów, gdzie reprezentowane są one przez więcej niż 60 przejść zapisanych w formacie KISS2.

W tabeli 3 symbol Q_i określa liczbę użytych makrokomórek w automacie Moore'a dla konkretnego przykładu wyszczególnionego w kolumnie „FILE". We wszystkich przypadkach zastosowano algorytm ESPRESSO [6], który został wykorzystany do kodowania stanów. Średni zysk dla optymalnego kodowania stanów wynosi 15,1%.

Tab. 3. Wyniki eksperymentu
Tab. 3. Results of experiments

File	p-01	p-02	p-03	p-04	p-05	p-06	p-07	p-08	p-09	p-10
Q_1	216	276	124	197	255	168	346	164	240	507
Q_2	198	213	86	196	208	149	261	163	191	421
Q_1/Q_2 [%]	8,3	22,8	30,6	0,5	18,4	11,3	24,6	0,6	20,4	17,0

6. Wnioski

Zaprezentowana metoda pozwala 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 pseudorówno-

ważnych stanów automatu Moore'a, która prowadzi 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 oscyluje w granicach od 1% 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].

7. Literatura

[1] Baranov S.: Logic and System Design of Digital Systems. Tallinn: TUT Press, 2008.

[2] Barkalov A. and Węgrzyn M.: Design of Control Units with Programmable Logic, Zielona Gora: University of Zielona Gora Press, 2006.

[3] Barkalov A., Titarenko L.: Logic Synthesis for FSM – based Control Units. Berlin: Springer, str. 233, 2009.

[4] Solovjev V., Klimowicz A.: Logic design of digital Systems with programmable logic devices, Moscow: Hotline – Telecom, str. 376, 2008.

[5] De Micheli G.: Synthesis and Optimization of Digital Circuits, N.Y.: McGraw Hill, 1994.

[6] Kania D.: Synteza logiczna przeznaczona dla matrycowych struktur programowalnych typu PAL, Gliwice: Silesian Technical University, 2004.

[7] Yang S.: Logic Synthesis and Optimization Benchmarks User Guide, Microelectronics Center of North Carolina, Research Triangle Park, North Carolina, 1991.

otrzymano / received: 13.05.2011
przyjęto do druku / accepted: 04.07.2011 *artykuł recenzowany*

INFORMACJE

Szanowni Autorzy artykułów publikowanych w PAK,

W trosce o jak najwyższy poziom punktacji miesięcznika PAK zwracam się z prośbą o cytowanie artykułów opublikowanych w PAK w innych artykułach, zwłaszcza tych publikowanych w czasopismach z listy filadelfijskiej. Ma to bezpośredni wpływ na współczynnik IF (Impact Factor) miesięcznika PAK.

W algorytmach oceny czasopism współczynnik IF ma największą wagę. Na zwiększenie wartości współczynnika IF redakcja czasopisma nie ma żadnego wpływu, ale wszystko zależy od Autorów cytujących. W przypadku miesięcznika PAK aktualnie każde cytowanie zwiększa IF o około 0,002. Oczywiście cytowanie artykułu tylko wtedy jest uzasadnione, jeżeli jest on tematycznie związany z artykułem cytującym, a autor korzystał z niego przy przygotowaniu pracy.

Aby ułatwić Autorom korzystanie z artykułów opublikowanych w PAK (a także możliwość cytowania) została opracowana przez redakcję PAK „Wyszukiwarka", umożliwiająca wyszukiwanie artykułów według nazwiska autora, słowa tytułu artykułu, albo frazy kluczowej.

Aby skorzystać z „Wyszukiwarki" należy:
- wejść na stronę: www.pak.info.pl
- w menu „Wyszukiwarka" (po lewej stronie ekranu) wybrać „Artykuły".

Strona zawiera również szereg innych łatwo dostępnych funkcjonalności, m.in. wykazy artykułów opublikowanych w PAK, a cytowanych w artykułach opublikowanych w czasopismach z listy filadelfijskiej.

Zdaję sobie sprawę, że redakcje niektórych czasopism usuwają cytowania artykułów publikowanych w czasopismach spoza listy filadelfijskiej, np. argumentując, że są one mało dostępne. Taka argumentacja będzie mniej uzasadniona, jeżeli tytuł naszego miesięcznika oraz tytuły artykułów będą podane w cytowaniach w języku angielskim. Proszę zauważyć, że oficjalny tytuł anglojęzyczny miesięcznika PAK (występujący na okładce) ma formę: Measurement, Automation and Monitoring (MA&M), a wszystkie artykuły naukowe publikowane w PAK są napisane albo w języku angielskim, albo mają rozszerzone abstrakty w tym języku.

Tadeusz SKUBIS
Redaktor naczelny Wydawnictwa PAK

Informacja redakcji dotycząca artykułów współautorskich

W miesięczniku PAK od numeru 06/2010 w nagłówkach artykułów współautorskich wskazywany jest autor korespondujący (Corresponding Author), tj. ten z którym redakcja prowadzi wszelkie uzgodnienia na etapie przygotowania artykułu do publikacji. Jego nazwisko jest wyróżnione drukiem pogrubionym. Takie oznaczenie nie odnosi się do faktycznego udziału współautora w opracowaniu artykułu. Ponadto w nagłówku artykułu podawane są adresy korespondencyjne wszystkich współautorów.

Wprowadzona procedura wynika z międzynarodowych standardów wydawniczych.

Redakcja