ISSN 0209-2069

ZESZYTY NAUKOWE NR 72
WYŻSZEJ SZKOŁY MORSKIEJ
SZCZECIN 2003

WYDZIAŁ INŻYNIERYJNO-EKONOMICZNY TRANSPORTU

Grzegorz Hołowiński
Krzysztof Małecki

Badanie algorytmu rozpoznawania symetrii argumentów funkcji logicznych

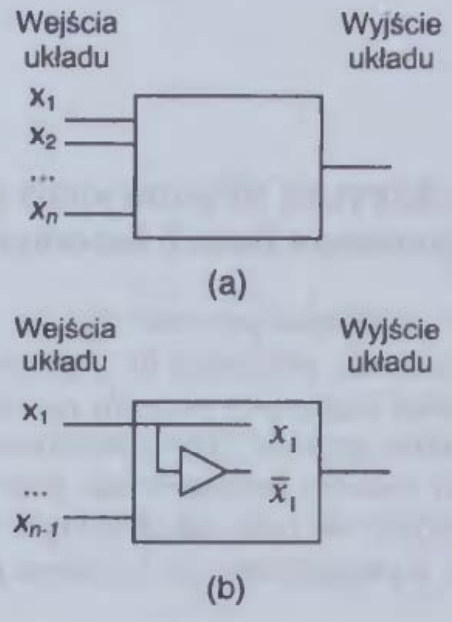
W praktyce istotnym problemem jest zmniejszenie liczby wejść układu logicznego, np. układu sterowania, procesora lub jego modułu, itp. Rozpoznanie symetrii argumentów funkcji logicznych pozwala na zmniejszenie liczby wejść takiego układu. W niniejszym artykule autorzy przedstawiają wyniki badań nad algorytmem poszukiwania symetrii zmiennych dla form wielomianowych. Proponowane rozwiązanie dotyczy nie tylko układów logicznych, ale także tych zjawisk, które można zapisać w postaci funkcji lub systemu funkcji logicznych.

Research on Symmetry Recognition Algorithm of Logical Functions Arguments

An essential practical problem is to decrease the number of inputs in the logical system, e.g. the control system, the processor or its module, etc. A recognition of the symmetry of logical functions arguments permits to decrease the input number of such a system. The present paper presents the results of research on an algorithm of searching for symmetry of variables for polynomial forms. The solution suggested concerns not only logical systems, but also those phenomena which can be recorded in the form of functions or a system of logical functions.

Wstęp

Symetria funkcji pozwala na zmniejszenie liczby wejść układu logicznego (rys. 1). Metody wykrywania symetrii funkcji boolowskich zostały opracowane w pracy [1]. Właściwości symetrii zastosowano również w syntezie diagramów decyzyjnych (ang. *Decision Diagrams* – DD) [2]. Wykrywanie symetrii zmiennych stanowi niezbędną część algorytmów minimalizacji funkcji logicznych (np. Espresso [3] i innych [4]), stosowanych obecnie przy projektowaniu układów FPGA. Etap wykrywania symetrii pozwala skrócić czas minimalizacji poprzez wyeliminowanie zmiennych symetrycznych w kolejnych krokach algorytmu.



Rys. 1. Układ logiczny bez symetrii wejść (a) i z symetrią zmiennych (b)

Fig. 1. Logical system without symmetry of inputs (a) and with symmetry of variables (b)

Rozpoznawanie symetryczności zmiennych, badanej funkcji logicznej, przedstawionych w postaci tablicy prawdy jest problemem NP-trudnym, gdyż zadane funkcje są, w większości przypadków, funkcjami zupełnymi. Oznacza to, że dla liczby n zmiennych wektor prawdy funkcji boolowskiej składa się z 2^n elementów. Aby zmniejszyć rozmiar naszej funkcji, należałoby przeprowadzić proces minimalizacji i podać zadaną funkcję w postaci np. wielomianu arytmetycznego. W artykule przedstawiono wyniki badań nad możliwością wykorzystania wielomianów arytmetycznych do rozpoznawania symetrii całkowitej oraz częściowej.

Tabela 1

Tablica prawdy systemu
dwóch niezupełnych funkcji logicznych

Truth table of a system
of two incompletely logical functions

x_0	x_1	x_2	x_3	x_4	f_1	f_0
0	0	0	1	1	0	1
1	0	0	0	0	0	0
1	1	1	0	1	1	1
1	0	0	1	0	0	0
1	1	0	1	0	1	1

1. Podstawy teoretyczne

Formą arytmetyczną funkcji boolowskiej $f(X) = f(x_1,x_2,...,x_n)$ n zmiennych jest zapis postaci:

$$P(X) = \sum_{i=0}^{2^n-1} p^{(i)} (x_1)^{i_1} (x_2)^{i_2} .. (x_n)^{i_n},$$

$$x_i^{i_j} = \begin{cases} 1 \text{ dla } i_j = 0 \\ 0 \text{ dla } i_j = 1 \end{cases}$$

gdzie $p^{(i)}$ są współczynnikami wielomianu $P(X)$, liczbami całkowitymi takimi, że $P(X) \in (0,1)$ [5], $i = i_1 i_2 ... i_n$ – binarna reprezentacja i.

Zaletą wielomianów arytmetycznych jest możliwość przedstawienia nie tylko pojedynczych funkcji logicznych, ale także systemów m funkcji logicznych $f_1(X), f_2(X), ..., f_m(X)$.

Wprowadzimy teraz pojęcia niezbędne do zdefiniowania funkcji ρ–symetrycznych (częściowo symetrycznych).

Definicja 1. Podziałem ρ zbioru n zmiennych, X nazywamy [6] ciąg $(\rho_1,\rho_2,...,\rho_s)$, $s \le n$ podzbiorów zbioru X $\rho_j \subseteq X$ takich, że $\bigcap_{j=1}^{s} \rho_j = X$, $\forall i, j \in \{1,...,s\}\ \rho_i \cap \rho_j = \varnothing$.

Przykład. Dla zbioru zmiennych $X = (x_1, x_2, x_3, x_4, x_5)$ przykładowymi podziałami są: $((x_1, x_2), (x_3, x_4, x_5))$, $((x_1, x_2), (x_3), (x_4, x_5))$

Definicja 2. Permutacją $\prod$ podziału ρ nazywamy działanie polegające na dowolnym przestawieniu zmiennych w każdym z podzbiorów ρ [6]:

$$\Pi(X) = (\Pi(\rho_1), \Pi(\rho_2), ..., \Pi(\rho_s))$$

Dysponując definicją podziału i permutacji ρ można wprowadzić pojęcie funkcji ρ-symetrycznej.

Definicja 3. Funkcją ρ-symetryczną nazywamy funkcję logiczną $f(X) = f(\rho_1, \rho_2, ..., \rho_s)$, dla której dla dwóch dowolnych różnych permutacji ρ wartość funkcji nie ulega zmianie:

$$\forall \Pi_1, \Pi_2 \; f(\Pi_1(X)) = f(\Pi_2(X))$$

Przykład. Funkcja $f = x_1x_3 \vee x_2x_4$ jest funkcją ρ-symetryczną, o podziale $((x_1, x_3), (x_2, x_4))$, a permutacje nie zmieniają jej wartości:

$$f(x_1, x_2, x_3, x_4) = f(x_1, x_4, x_3, x_2) = f(x_3, x_4, x_1, x_2) = f(x_3, x_2, x_1, x_4)$$

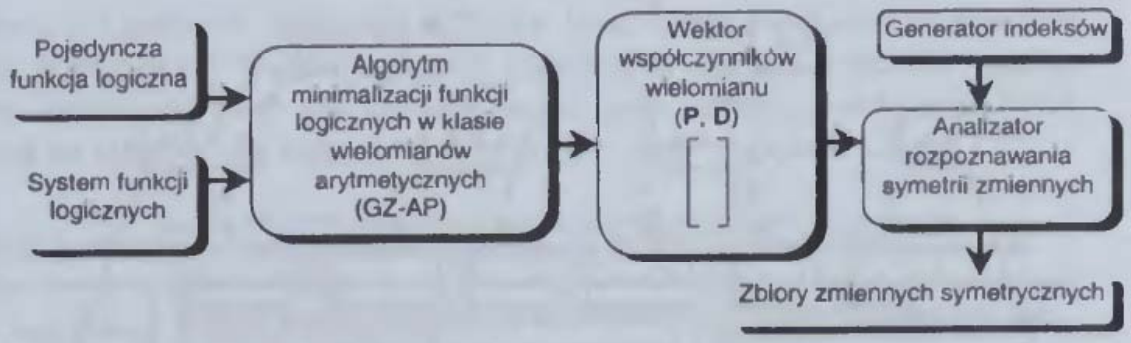
Definicja 4. Funkcją całkowicie symetryczną nazywamy funkcję ρ-symetryczną, dla której podział ρ jest zdefiniowany jako pojedynczy podzbiór obejmujący wszystkie zmienne $((x_1, x_2, ..., x_n))$:

$$f(X) \text{ jest calk. symetryczna} \Leftrightarrow \forall \Pi_1(X), \Pi_2(X) \; f(\Pi_1(X)) = f(\Pi_2(X))$$

Przykład. Funkcja $f = x_1x_2x_3 \vee x_2x_3x_4 \vee x_1x_3x_4 \vee x_1x_2x_4$ jest całkowicie symetryczna – dowolne przestawianie zmiennych nie zmienia jej wartości.

2. Struktura i zasada działania pakietu programowego do rozpoznawania symetryczności

W niniejszym podrozdziale został przedstawiony model systemu rozpoznawania symetryczności funkcji. Poszczególnymi składowymi tego systemu są (rys. 2.): algorytm minimalizacji funkcji logicznych w klasie wielomianów arytmetycznych [7], generator indeksów oraz analizator sprawdzający symetryczność.



Rys. 2. Struktura pakietu programowego do rozpoznawania symetryczności

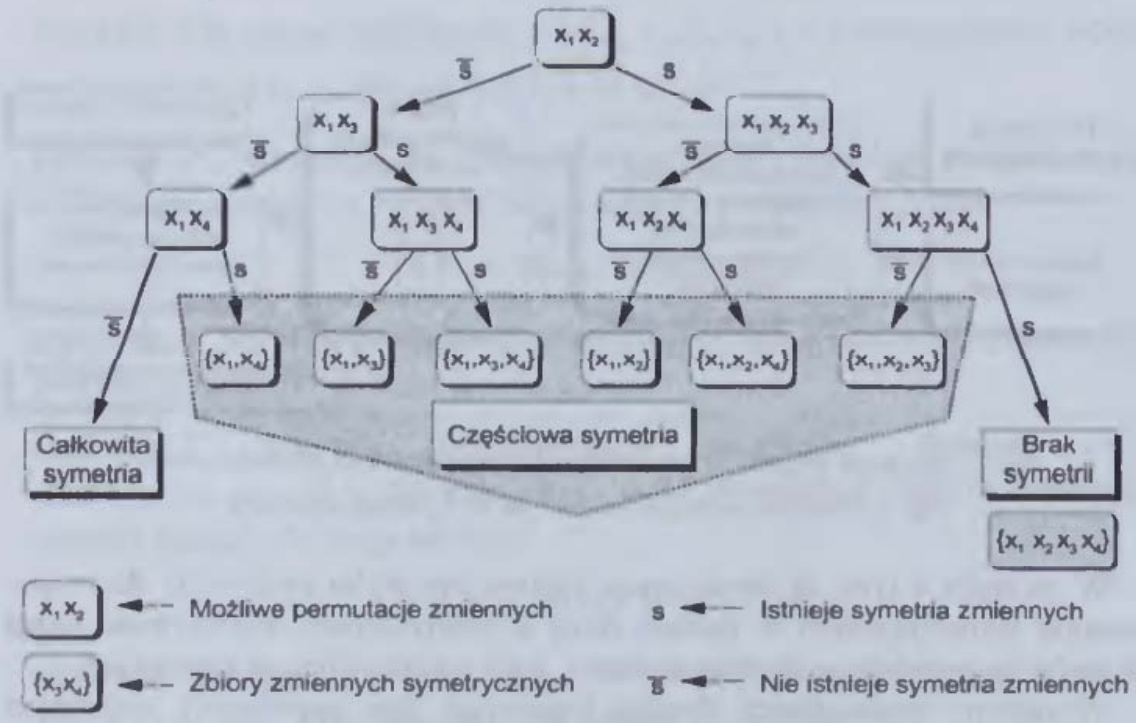
Fig. 2. Program package structure for recognizing symmetry

W związku z tym, że opracowany system jest wykorzystywany do rozpoznawania symetryczności w postaci form arytmetycznych, analizowane mogą być zarówno pojedyncze funkcje logiczne, jak i systemy funkcji logicznych.

Wynikiem minimalizacji funkcji logicznych jest pojedynczy wielomian arytmetyczny w polaryzacji zerowej zapisany w postaci wektora współczynników (dla pojedynczej funkcji logicznej – **P** oraz dla systemu funkcji logicznych – **D**). Następnie, dane te (w formie pojedynczych produktów występujących w końcowej formie rozwiązania) i dodatkowo indeks opisujący dany produkt są przekazywane do specjalnego analizatora, gdzie jest wykrywana symetria dla poszczególnych zmiennych. W przypadku znalezienia symetrii pomiędzy argumentami jest wyświetlane rozwiązanie w postaci zbiorów zmiennych symetrycznych i są pobierane: kolejny produkt wielomianu arytmetycznego oraz indeks tego produktu. Następuje proces rozpoznawania symetrii, itd., aż do momentu przeanalizowania całej formy arytmetycznej.

2.1. Zasada działania generatora indeksów

Generator indeksów działa na zasadzie przeszukiwania drzewa wszystkich możliwych permutacji. Rysunek 3 przedstawia, na przykładzie uproszczonego drzewa, zasadę działania generatora indeksów dla przypadku poszukiwania występowania symetrii względem zmiennej x_1. Dla pozostałych zmiennych procedura poszukiwania jest taka sama.



Rys. 3. Uproszczony algorytm, w postaci drzewa, poszukiwania zmiennych symetrycznych dla funkcji boolowskiej czterech (n = 4) zmiennych

Fig. 3. Simplified tree-shaped algorithm of searching for symmetric variables for Boolean function of four variables (n = 4)

Generator indeksów argumentów funkcji całkowicie symetrycznych

Program generuje wszystkie możliwe kombinacje indeksów dla argumentów całkowicie symetrycznych funkcji boolowskich. Oznacza to, że generowane są grupy indeksów dla poszczególnych grup argumentów, mających ten sam współczynnik wielomianu arytmetycznego.

Dla funkcji boolowskiej n zmiennych generator podaje na swoim wyjściu indeksy następujących grup argumentów funkcji: (a) wszystkich możliwych argumentów x_1; x_2; ...; x_n. (b) wszystkich możliwych par x_1–x_2; x_1–x_3; ...; x_1–x_n; x_2–x_3; x_2–x_4; ...; x_2–x_n; ...; x_{n-1}–x_n. (c) wszystkich możliwych trójek x_1–x_2–x_3; x_1–x_2–x_4; ...; x_2–x_3–x_4; x_2–x_3–x_5; ...; x_{n-2}–x_{n-1}-x_n. (d) itd., aż do wszystkich możliwych kombinacji n–1 elementowych, np. x_1–x_2–...–x_{n-1}.

Poszczególne grupy indeksów przekazywane na wyjściu generatora trafiają do modułu, który sprawdza, czy wartości elementów wektora prawdy o indeksach uzyskanych za pomocą generatora są dla rozpatrywanej funkcji boolowskiej takie same. Jeżeli dla którejś z grup indeksów elementy wektora prawdy nie będą przyjmowały tej samej wartości, to oznacza, że funkcja nie jest całkowicie symetryczna.

Generator indeksów argumentów funkcji częściowo symetrycznych

Program generuje wszystkie możliwe kombinacje indeksów argumentów funkcji boolowskich w celu wykrycia istniejących dla nich symetrii. Analiza wartości wektora prawdy dla rozpatrywanych grup argumentów umożliwia odpowiedź na pytanie: czy argumenty te są symetryczne względem siebie?

Zasada działania generatora indeksów argumentów funkcji częściowo symetrycznych

Dane wejściowe	Wektor współczynników wielomianu arytmetycznego opisującego pojedynczą funkcję logiczną lub system funkcji logicznych
Dane wyjściowe	Zbiory zmiennych symetrycznych
Krok 1:	Pobierz dwa pierwsze argumenty, dla których ma być rozpatrzona częściowa symetryczność.
Krok 2:	Rozpoznawanie symetrii dla pobranych argumentów.
Krok 3:	Jeżeli rozpoznawanie zostało przeprowadzone dla wszystkich zmiennych zadanej funkcji logicznej, to idź do 5. Gdy symetryczność zmiennych nie została rozpoznana, to weź kolejną parę argumentów badanej funkcji logicznej i idź do 2. Gdy symetryczność zmiennych została rozpoznana, to idź do 4.
Krok 4:	Dobierz kolejny argument i idź do 2.
Krok 5:	Wyświetl zbiory zmiennych symetrycznych.

Rys. 4. Algorytm opisujący zasadę działania generatora indeksów argumentów (zmiennych) dla funkcji częściowo symetrycznych

Fig. 4. Algorithm describing the working principle of an argument index (variables) for partially symmetric functions

Powyższy algorytm jest wykonywany dla wszystkich par argumentów. W przypadku, gdy dla wygenerowanych indeksów argumentów moduł porównujący wartości wektora prawdy rozpatrywanej funkcji potwierdzi symetryczność zmiennych, to do takiej pary argumentów dobiera się po kolei – pojedynczo – wszystkie z pozostałych argumentów w celu wykrycia zjawiska symetrii trzech argumentów, dla których generator szuka odpowiednich indeksów. Gdy symetria zostanie wykryta, to dobiera się kolejny z możliwych argumentów funkcji w celu wykrycia symetrii poczwórnej, itd. Możliwym krańcowym stanem jest wykrycie całkowitej symetrii lub jej całkowitego braku.

3. Badania eksperymentalne

Poniższe badania eksperymentalne zostały wykonane na standardowych funkcjach porównawczych MCNC na komputerze PC Pentium 200 MMX z systemem operacyjnym LINUX RED HAT.

Celem przeprowadzonych badań eksperymentalnych dotyczących rozpoznawania symetrii zmiennych było: (a) wyznaczenie pewnych charakterystyk opracowanego systemu, (b) potwierdzenie i udowodnienie metody rozpoznawania symetrii zmiennych dla zespołów funkcji boolowskich, (c) porównanie uzyskanych wyników z wynikami uzyskanymi w klasie wielomianów Reeda-Mullera.

Tabela 2

Wyniki minimalizacji i rozpoznawania symetrii dla standardowych funkcji porównawczych – opracowany pakiet programowy

Minimization results and symmetry recognition for standard comparative functions – using a worked out program package

Funkcja	WE	*T/L*	Rozpoznana symetria	t_{gen} [s]	t_c [s]
bw11	5	24/68	x_1–x_3–x_5; x_2–x_4	0,00	0,00
bw13	5	7/25	x_2–x_4; x_3–x_5	0,00	0,00
bw2	5	4/13	x_4–x_5	0,01	0,01
5x01	7	18/81	x_3–x_4	0,00	0,01
5x10	7	7/34	x_1–x_7; x_2–x_3–x_4; x_5–x_6	0,00	0,00
5x5	7	17/51	x_1–x_4; x_5–x_6	0,00	0,00
5x6	7	9/21	x_5–x_6–x_7	0,00	0,00
z41	7	40/189	x_1–x_4–x_7; x_2–x_5; x_3–x_6	0,00	0,01
z42	7	55/236	x_1–x_4–x_7; x_2–x_5; x_3–x_6	0,00	0,01
f53	8	27/98	x_1–x_2	0,00	0,01
f55	8	9/21	x_1–x_2–x_3–x_4	0,01	0,01
f57	8	3/4	x_1–x_2–x_3–x_4–x_5–x_6; x_7–x_8	0,00	0,00
Newtag	8	24/116	x_1–x_3; x_5–x_6; x_7–x_8	0,00	0,02
sao21	10	380/1868	x_6–x_{10}	0,03	0,85
sao22	10	768/4096	x_6–x_{10}	0,06	2,11
sao23	10	577/3057	nie wykryto symetrii	0,03	1,73
sao24	10	936/4536	x_6–x_{10}	0,06	2,67
Sym10	10	848/4660	x_1–x_2–x_3–x_4–x_5–x_6–x_7–x_8–x_9–x_{10}	0,36	2,46

T – liczba produktów, t_c – czas całkowity,
L – liczba literałów, WE – liczba argumentów.
t_{gen} – czas generatora,

Należy zaznaczyć, że wszystkie funkcje porównawcze (tab. 2) wykorzystane do przeprowadzenia badań są funkcjami zupełnymi. Dlatego też, wyniki uzyskane dla funkcji powyżej 8 zmiennych są wynikami mało zadowalającymi, jeżeli chodzi o stopień minimalizacji. Dzieje się tak, ponieważ algorytm GZ-AP został opracowany z myślą o minimalizacji funkcji niezupełnych. Zwróćmy uwagę na wyniki dotyczące rozpoznania symetrii argumentów dla poszczególnych funkcji. Prawie wszystkie wymienione w powyższej tabeli funkcje są funkcjami częściowo symetrycznymi. Wyjątek stanowi *sym10* – która jest funkcją całkowicie symetryczną – i *sao23*, dla której nie została rozpoznana żadna symetria.

Ostatnim parametrem wchodzącym w skład charakterystyki jest czas (*i*) wykonywania generatora indeksów opracowanego pakietu programowego, (*ii*) czas całkowity, na który składa się czas minimalizacji funkcji i czas generatora. Uzyskane czasy są wielkościami oscylującymi wokół jednej sekundy, z wyjątkiem funkcji 10. zmiennych, dla którym czas całkowity przekroczył dwie sekundy.

Sprawdźmy teraz efektywność opracowanego pakietu w przypadku rozpoznawania symetrii dla zespołów funkcji boolowskich.

Tabela 3

Wyniki minimalizacji i rozpoznania symetrii dla standardowych funkcji porównawczych – algorytm GZ-AP

Minimization results and symmetry recognition for standard comparative functions – using the GZ-AP algorithm

Funkcja	WE	WY	*T/L*	Rozpoznana symetria	t_{gen} [s]	t_c [s]
F2	4	4	9/20	–	0,00	0,01
F2_1,2	4	2	7/18	x_1–x_4	0,00	0,03
F2_3,4	4	2	7/18	x_2–x_3	0,00	0,00
Bw26_27	5	2	28/75	x_2–x_4	0,00	0,00
Rd53	5	3	11/30	x_1–x_2–x_3–x_4–x_5	0,00	0,00
5x	7	8	60/222	–	0,01	0,02
5x_5,6,7,10	7	4	28/90	x_5–x_6	0,00	0,01
Rd73	7	3	127/448	x_1–x_2–x_3–x_4–x_5–x_6–x_7	0,01	0,03
Z4	7	4	64/250	x_1–x_4–x_7; x_2–x_5; x_3–x_6	0,00	0,02
F5	8	7	94/400	–	0,00	0,04
Rd84	8	3	255/1024	x_1–x_2–x_3–x_4–x_5–x_6–x_7–x_8	0,03	0,15
Sao2	10	3	1020//5116	x_6–x_{10}	0,05	3,04

T – liczba produktów,
L – liczba literałów,
t_{gen} – czas generatora,
t_c – czas całkowity,
WE – liczba argumentów,
WY – liczba funkcji.

Okazuje się, że nie zawsze system funkcji w danej funkcji porównawczej ma rozpoznawalną symetrię. Przeanalizujmy sytuację na przykładzie funkcji *f2* (tab. 3). W przypadku rozważania całego zespołu nie została rozpoznana żadna symetria. Ale okazało się, że funkcje *f2_1* i *f2_2* oraz *f2_3* i *f2_4* mają wspólne symetrie, co zostało uwidocznione w powyższej tabeli.

3.1. Porównywanie uzyskanych wyników badań eksperymentalnych

Sprawdźmy, czy rzeczywiście rozpoznawanie symetrii argumentów funkcji logicznych przedstawionych w postaci form arytmetycznych daje nam konkretne korzyści. Tabela 4 przedstawia fragment badań eksperymentalnych dotyczących procesu porównania uzyskanych wyników. Podano w niej trzy wielowyjściowe funkcje testujące. W przypadku rozpoznawania symetrii dla poszczególnych pojedynczych składowych występujących w rozpatrywanych systemach, wyniki są takie same. Oznacza to, że korzystanie z logiki arytmetycznej w celu przeprowadzenia procesu rozpoznawania symetrii jest możliwe a uzyskiwane wyniki są poprawne.

W klasie wielomianów RM niemożliwe było rozpoznanie symetrii dla całego systemu funkcji logicznych. Natomiast w klasie wielomianów arytmetycznych jest to możliwe, ale należy zaznaczyć, że nie zawsze istnieje symetria dla wielowyjściowej funkcji logicznej.

Tabela 4

Fragment badań eksperymentalnych dotyczących porównania wyników uzyskanych w procesie rozpoznawania symetrii argumentów w klasie wielomianów arytmetycznych i w klasie wielomianów RM

Fragment of experimental research concerning a comparison of results obtained in the process of recognizing symmetry of arguments in the class of arhythmical polynomials and in the class of RM polynomials

Funkcja	Nr w systemie	Rozpoznana symetria				
		wielomiany arytmetyczne			wielomiany RM	
		pojedyncze funkcje	system funkcji		pojedyncze funkcje	system funkcji
f2	f2_1	x_1–x_4	x_1–x_4	brak	x_1–x_4	
	f2_2	x_1–x_4	x_1–x_4	brak	x_1–x_4	
	f2_3	x_2–x_3	x_2–x_3	brak	x_2–x_3	
	f2_4	x_2–x_3	x_2–x_3	brak	x_2–x_3	
rd53	rd53_1	x_1–x_2–x_3–x_4–x_5	x_1–x_2–x_3–x_4–x_5		x_1–x_2–x_3–x_4–x_5	brak
	rd53_2	x_1–x_2–x_3–x_4–x_5			x_1–x_2–x_3–x_4–x_5	
	rd53_3	x_1–x_2–x_3–x_4–x_5			x_1–x_2–x_3–x_4–x_5	
rd73	rd73_1	x_1–x_2–x_3–x_4–x_5–x_6–x_7	x_1–x_2–x_3–x_4–x_5–x_6–x_7		x_1–x_2–x_3–x_4–x_5–x_6–x_7	brak
	rd73_2	x_1–x_2–x_3–x_4–x_5–x_6–x_7			x_1–x_2–x_3–x_4–x_5–x_6–x_7	
	rd73_3	x_1–x_2–x_3–x_4–x_5–x_6–x_7			x_1–x_2–x_3–x_4–x_5–x_6–x_7	

W takim przypadku należy podzielić rozpatrywaną funkcję na mniejsze części i dla nich przeprowadzić proces ponownie. W szczególnym przypadku sprowadzi się to do rozbicia zespołu funkcji na pojedyncze funkcje logiczne.

Podsumowanie

Autorzy w niniejszym artykule zaprezentowali pakiet programowy do rozpoznawania argumentów funkcji logicznych w postaci form arytmetycznych. Przeprowadzone doświadczenia potwierdziły możliwość rozpoznawania symetrii argumentów pojedynczych funkcji logicznych przedstawionych w postaci wielomianów arytmetycznych oraz możliwości zastosowania logiki arytmetycznej do rozpoznawania symetrii dla zespołów funkcji logicznych.

Literatura

1. Chien-Chang Tsai, Marek-Sadowska M., *Generalized Reed-Muller Forms as a Tool to Detect Symmetries*, IEEE Trans. on Computers, 1996, no.1, vol. C-45, pp. 33–40.
2. Moller D., Molitor J., Drechsler R., *Symmetry Based Variable Ordering for ROBDDs*, Proc. IFIP Workshop on Logic and Architecture Synthesis, 1994, pp. 47–53.
3. Brayton R.K., Hatchel G.D., McMullen C.T., Sangiovanni-Vincentelli A.L., *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, Boston, 1984.
4. Łuba T., Jasiński K., Zbierzchowski B., *Specjalizowane układy cyfrowe w strukturach PLD i FPGA*, Wydawnictwa Komunikacji i Łączności, Warszawa, 1997.
5. Falkowski B., Hołowiński G., Małecki K., *Effective Minimization of Logic Functions in RM Domain*, Proc. Int. Conf. Applications of Computer Systems, Szczecin, Poland, 1997, pp. 248–255.
6. Davio M.J., Deschamps P., Thayse A., *Discrete and Switching Functions*, McGraw-Hill Int. Book Co., 1978.
7. Małecki K., *Nowy algorytm do minimalizacjji systemów funkcji boolowskich w klasie wielomianów arytmetycznych*, XX Międzynarodowe Sympozjum Naukowe Studentów i Młodych Pracowników Nauki, Zielona Góra, 1998, pp. 319–323.

Wpłynęło do redakcji w listopadzie 2002 r.

Recenzent

prof. dr hab. inż. Evgeny Ochin

Adresy Autorów

dr inż. Grzegorz Hołowiński
Wyższa Szkoła Morska w Szczecinie
Wydział Inżynieryjno-Ekonomiczny Transportu
Instytut Zarządzania Transportem
70-507 Szczecin
ul. Henryka Pobożnego 11
gholowinski@wsm.szczecin.pl

dr inż. Krzysztof Małecki
Politechnika Szczecińska
Wydział Informatyki
Szczecin
ul. Żołnierska 49
kmalecki@wi.ps.pl