Dariusz KANIA [1], **Waldemar GRABIEC** [2]
[1] POLITECHNIKA ŚLĄSKA, INSTYTUT ELEKTRONIKI
[2] WOJSKOWA AKADEMIA TECHNICZNA, WYDZIAŁ ELEKTRONIKI

Dekompozycja zespołu funkcji wykorzystująca elementy XOR

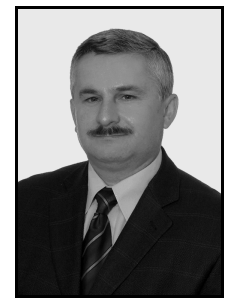
Dr hab. inż. Dariusz KANIA

Ukończył studia na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej. Pracę doktorską obronił w 1995, habilitacyjną 2004r. Jest profesorem w Instytucie Elektroniki Politechniki Śląskiej. Jego zainteresowania naukowe koncentrują się wokół programowalnych układów i systemów cyfrowych.

e-mail: dariusz.kania@polsl.pl

Mgr inż. Waldemar GRABIEC

Ukończył studia o specjalności telekomunikacja na Wydziale Elektroniki Wojskowej Akademii Technicznej. Aktualnie pracuje na stanowisku starszego wykładowcy w Instytucie Telekomunikacji Wydziału Elektroniki WAT. Jego zainteresowania naukowe dotyczą współczesnych systemów telekomunikacyjnych (głównie optycznych systemów transmisyjnych) oraz techniki cyfrowej w zakresie układów logiki programowalnej.



e-mail: waldemar.grabiec@wel.wat.edu.pl

Streszczenie

W artykule przedstawiono koncepcję syntezy logicznej dla matrycowych struktur CPLD. Rdzeniem układów CPLD jest blok logiczny typu PAL zawierający element XOR. Celem pracy jest zaprezentowanie metody syntezy, która umożliwia realizację zespołu funkcji za pomocą bloków logicznych typu PAL, zawierających określoną liczbę iloczynów i bramkę logiczną XOR.

Słowa kluczowe: synteza logiczna, dekompozycja, odwzorowanie technologiczne.

Decomposition of a multi-output function based on utilization of XOR gates

Abstract

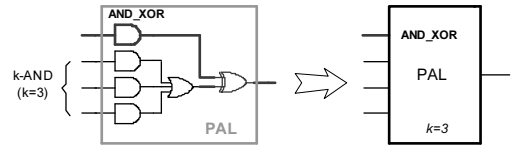
This paper presents logic synthesis for CPLD's. The core of CPLD's is a PAL-based structure with XOR gates. The aim of the work is to present the synthesis method, which enables implementation of the multi-output Boolean function by the means of the PAL-based logic blocks, containing a definite number of terms and XOR gates.

Keywords: logic synthesis, decomposition, technology mapping.

1. Wstęp

Struktury matrycowe CPLD (ang. *Complex Programmable Logic Devices*) obok układów FPGA typu tablicowego stanowią najbardziej znaną i popularną grupę programowalnych układów logicznych. Ze względu na krótkie czasy propagacji sygnałów przez strukturę oraz przewidywalne i stałe opóźnienia chętnie stosowane są w aplikacjach, w których właściwości czasowe stanowią parametr krytyczny.

Większość oferowanych obecnie struktur CPLD wykorzystuje architekturę charakterystyczną dla układów PAL (ang. *Programmable Array Logic*). Podstawowym jej elementem są bloki logiczne typu PAL (ang. *PAL-block*) zawierające pewną liczbę iloczynów k (najczęściej $k = 4 \div 8$) dołączonych na stałe do wejść bramki sumy logicznej. Bloki logiczne struktur CPLD oprócz wspomnianych bramek iloczynowych niejednokrotnie zawierają szereg dodatkowych elementów, do których należą m.in. konfigurowalne przerzutniki, wyjściowe bufory trójstanowe, czy bramki XOR (rys. 1) [1].



Rys. 1. Struktura bloku logicznego typu PAL zawierającego bramkę XOR
Fig. 1. Structure of PAL-based logic block consisting of XOR gate

Podstawowym problemem związanym z syntezą logiczną dla układów CPLD typu PAL jest efektywne wykorzystanie dostępnych zasobów iloczynowych. Głównym elementem klasycznej metody syntezy dedykowanej dla struktur CPLD jest minimalizacja, zwykle wykonywana dla każdej funkcji oddzielnie. Okazuje się jednak, że możliwe jest również wykorzystanie elementów dekompozycji, dotychczas rezerwowanych do wspierania procesu syntezy układów realizowanych w strukturach FPGA typu tablicowego (ang. *LUT-based FPGA*). W przypadku struktur matrycowych CPLD istota dekompozycji sprowadza się do dopasowania projektowanego układu cyfrowego do struktury bloku logicznego typu PAL [2].

W artykule zaprezentowano pomysł dekompozycji zespołu funkcji logicznych, umożliwiający wykorzystanie elementu XOR wchodzącego w skład bloków logicznych większości struktur CPLD. Przedstawiona koncepcja dekompozycji wykorzystuje elementy dekompozycji kolumnowej, będącej rozszerzeniem klasycznego modelu dekompozycji Curtisa [2, 3, 4]. Niniejszy artykuł jest kontynuacją pracy [1] w której przedstawiono koncepcję wykorzystania elementu XOR dla pojedynczych funkcji.

2. Istota dekompozycji kolumnowej zespołu funkcji

Od pewnego czasu obserwuje się dynamiczny rozwój metod syntezy logicznej wykorzystującej elementy dekompozycji. Taki stan rzeczy spowodowany jest głównie wprowadzeniem na rynek układów FPGA typu tablicowego. W syntezie logicznej przeznaczonej dla tej grupy układów programowalnych, przede wszystkim ze względu ich architekturę, dekompozycja odgrywa bardzo istotną rolę.

Wyniki szeregu eksperymentów przedstawionych między innymi w [2] pokazują, że niezwykle korzystne jest prowadzenie dekompozycji dla zespołu funkcji. Funkcja $f : B^n \rightarrow B^m$ podlega dekompozycji $F[g_1(\mathbf{X}_1), g_2(\mathbf{X}_1),..., g_p(\mathbf{X}_1), \mathbf{X}_2]$ wtedy i tylko wtedy, gdy złożoność kolumnowa matrycy podziałów (siatki Karnaugha) spełnia zależność $\nu(\mathbf{X_2} \mid \mathbf{X_1}) \leq 2^p$. Pomiędzy zbiorem $\mathbf{X_1}$ (związanym) i zbiorem $\mathbf{X_2}$ (wolnym) zachodzą następujące relacje: $\mathbf{X_1} \cup \mathbf{X_2} = \{i_n,...,i_2,i_1\}$ oraz $\mathbf{X_1} \cap \mathbf{X_2} = \phi$ [2, 3, 4].

Niech $\mathbf{Y} = \{y_m,...,y_2,y_1\}$ będzie zbiorem m-funkcji $y_i = f_i(i_n,...,i_2,i_1)$.

Przykład 1:

Poszukiwana jest realizacji funkcji *rd53.pla* ($f : B^5 \rightarrow B^3$) opisanej w formacie Espresso [5]. Układ testowy (*benchmark*) rd53 jest zespołem trzech funkcji (m=3): f_1, f_2, f_3. Poszukajmy dwóch realizacji układu testowego rd53 na blokach logicznych zawierających tylko trzy lub tylko cztery iloczyny (k=3 lub 4). Siatkę Karnaugha dla rozpatrywanego zespołu funkcji przedstawiono na rys. 2.

ab \ cde	000	001	011	010	110	111	101	100
00	000	001	010	001	010	011	010	001
01	001	010	011	010	011	100	011	010
11	010	011	100	011	100	101	100	011
10	001	010	011	010	011	100	011	010

f_1,f_2,f_3

Rys. 2. Siatka Karnaugha funkcji testowej *rd53*
Fig. 2. Karnaugh map of the *rd53* benchmark

W wyniku minimalizacji programem Espresso każdej funkcji oddzielnie (wywołanie z linii poleceń: *Espresso –Dso*) uzyskano następujące wartości parametrów (liczba iloczynów dla poszczególnych funkcji wyjściowych Δ_{fi}, $i=1,2,3$): Δ_{f1}=5, Δ_{f2}=16, Δ_{f3}=11.

Realizując rozpatrywaną funkcję $f: B^5 \rightarrow B^3$ **metodą klasyczną** należałoby użyć 15 bloków logicznych (k=3), uzyskując strukturę 3-warstwową. W przypadku użycia bloków 4-iloczynowych (k=4), uzyskujemy rozwiązanie wykorzystujące 11 bloków i będące strukturą 2-warstwową. Ze względu na ograniczoną objętość artykułu nie przedstawiono szerzej sposobu realizacji funkcji metodą klasyczną. Aparat matematyczny związany z określeniem podstawowych parametrów (liczba bloków logicznych oraz liczba warstw logicznych) dla metody klasycznej przedstawiono w [2].

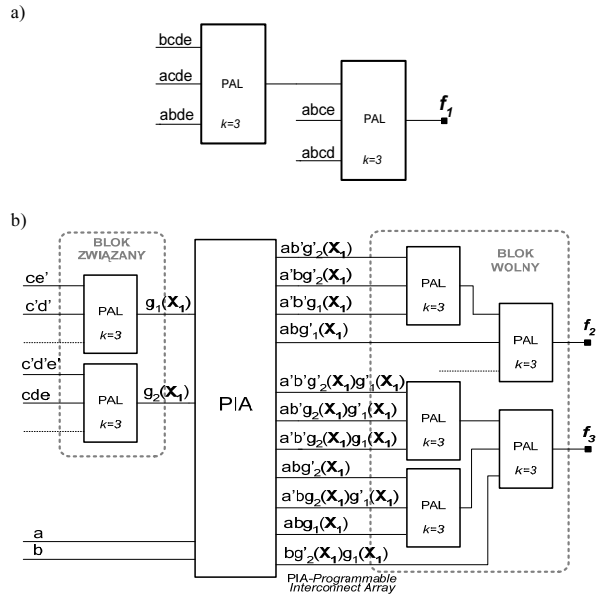
Okazuje się, że istnieje szereg efektywniejszych metod syntezy od metody klasycznej [2]. Jedną z nich, najefektywniejszą pod względem powierzchni, jest metoda wykorzystująca elementy dekompozycji, ukierunkowana na struktury CPLD typu PAL, tzw. dekompozycja kolumnowa. Istota tej metody sprowadza się do odpowiedniego kodowania wzorców kolumn matrycy podziału zapewniającego równomierne wykorzystywanie jak najmniejszej liczby iloczynów. Precyzyjny opis dekompozycji kolumnowej można znaleźć w pracy [2].

Przykład 2:

Poszukajmy realizacji układu *rd53* na blokach typu PAL zawierających 3-iloczyny bazując na metodzie dekompozycji kolumnowej. W wyniku minimalizacji (*Espresso-Dso*) uzyskaliśmy wartości wyróżników Δ_{f1}=5, Δ_{f2}=16, Δ_{f3}=11, określające liczby implikantów potrzebnych do realizacji poszczególnych funkcji. Korzyści stosowania dekompozycji mogą wystąpić dla funkcji, dla której spełniony jest warunek $\Delta_{fi}>2k$. W przeciwnym przypadku poszczególne funkcje realizowane są metodą klasyczną. Dla rozpatrywanego układu *rd53* taka sytuacja występuje dla funkcji f_1 ($\Delta_{f1}<2k$), stąd poszukiwania odpowiedniej dekompozycji kolumnowej ograniczone są do zespołu dwóch funkcji f_2,f_3.

Istota dekompozycji kolumnowej sprowadza się do poszukiwania odpowiedniego podziału zbioru argumentów zespołu funkcji, dla którego liczba wzorców kolumn (złożoność kolumnowa) matrycy podziałów zapewnia podział analizowanego układu na blok związany i wolny o porównywalnej złożoności, określanej jako iloczyn liczby wejść i wyjść. Złożoność (krotność) kolumnową wyznacza się wykorzystując algorytm kolorowania wzorców kolumn ukierunkowany na struktury typu PAL [2]. Dla rozpatrywanego układu testowego *rd53* wybrany zostaje podział argumentów na dwa podzbiory $\mathbf{X_1}$={*c*,*d*,*e*}; $\mathbf{X_2}$={*a*,*b*}, dla którego odpowiednie kodowanie wzorców kolumn [2] prowadzi do rozwiązania przedstawionego na rys. 3b. Rysunek 3a przedstawia natomiast klasyczną realizację funkcji f_1 rozpatrywanego zespołu funkcji. W metodzie tej wykorzystano dwa 3-iloczynowe bloki logiczne typu PAL, uzyskując strukturę 2-warstwową.

Widać więc, że uzyskujemy rozwiązanie znacznie efektywniejsze pod względem liczby bloków od rozwiązania uzyskiwanego metodą klasyczną. Wykorzystujemy tylko 9 bloków, uzyskując strukturą 3-warstwową, nie gorszą pod względem dynamicznym od rozwiązania klasycznego. Podobne oszczędności układowe występują dla większych bloków zawierających cztery iloczyny (k=4). Uzyskujemy wtedy rozwiązanie wykorzystujące siedem bloków logicznych typu PAL i będące również strukturą 3-warstwową.



Rys. 3. Wynik dekompozycji kolumnowej układu testowego *rd53*
Fig. 3. Results of column decomposition for the *rd53* benchmark

3. Dekompozycja zespołu funkcji ukierunkowana na wykorzystanie elementów XOR

Opracowany model dekompozycji kolumnowej można rozszerzyć poszukując rozwiązań wykorzystujących powszechnie dostępne w strukturach CPLD elementy XOR. Złożoność (krotność) kolumnowa siatki Karnaugha zespołu funkcji f_2,f_3 wynosi 4 (rys. 4a). Analizując siatkę Karnaugha przedstawioną na rys. 4a można spostrzec, iż dokonując negacji funkcji f_2 dla wzorca A (0010→1101) uzyskamy identyczną kolumnę (w sensie kombinacji 0 i 1) jak kolumny skojarzone z wzorcem C. Analogicznie, dokonując negacji funkcji f_2 dla wzorca D (1000→0111) uzyskujemy wzorzec B.

Na rys. 4b przedstawiono siatkę Karnaugha zmodyfikowaną w stosunku do siatki pierwotnej zgodnie z powyższymi spostrzeżeniami. Nową funkcję f_2 powstałą po wykonaniu opisanych wyżej operacji negacji oznaczono jako f_2'.

a)

ab \ cde	000	001	011	010	110	111	101	100
00	00	01	10	01	10	11	10	01
01	01	10	11	10	11	00	11	10
11	10	11	00	11	00	01	00	11
10	01	10	11	10	11	00	11	10
	A	B	C	B	C	D	C	B f_2,f_3

b)

ab \ cde	000	001	011	010	110	111	101	100
00	10	01	10	01	10	01	10	01
01	11	10	11	10	11	10	11	10
11	00	11	00	11	00	11	00	11
10	11	10	11	10	11	10	11	10
	C	B	C	B	C	B	C	B f_2',f_3

Rys. 4. Siatka Karnaugha zespołu funkcji (f_2,f_3) przed (a) oraz po (b) modyfikacji
Fig. 4. Karnaugh map of f_2 , f_3 function before (a), after (b) modification

W efekcie tych zmian siatka Karnaugha z rys. 4b posiada tylko dwa wzorce kolumn (B i C), stąd w myśl twierdzenia Curtisa [4] blok związany może mieć tylko jedno wyjście (p=1). W sytuacji, w której nie ma potrzeby dokonywania ekspansji liczby iloczynów, istnieje możliwość realizacji funkcji g($\mathbf{X_1}$) wykorzystując tylko jeden blok logiczny zawierający trzy iloczyny.

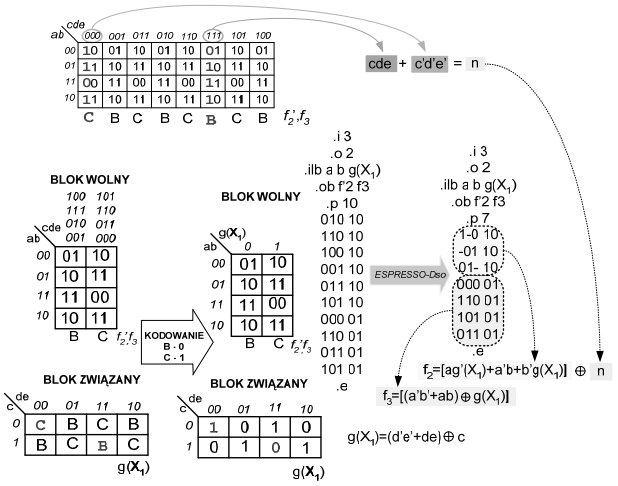
Zachodzi jednak potrzeba negacji funkcji f_2 dla dwóch kombinacji wektorów związanych c,d,e = {000, 111}. Zwykle w blokach

logicznych struktur CPLD występuje tylko jedna linia iloczynowa dołączona do elementu XOR. Nie ma więc bezpośredniej możliwości negacji funkcji f_2 dla wymienionych powyżej dwóch kombinacji wektorów związanych. Zachodzi więc potrzeba użycia dodatkowego bloku typu PAL (oznaczmy go literą ***n***) wytwarzającego na wyjściu sygnał odpowiedzialny za negację funkcji f_2. Blok ten powinien realizować funkcję $n(c,d,e)=cde+c'd'e'$.

W tej sytuacji funkcję f_2 można wyrazić zależnością $f_2 = f_2' \oplus n$, przy czym funkcja f_2' jest opisana siatką Karnaugha z rys. 4b.

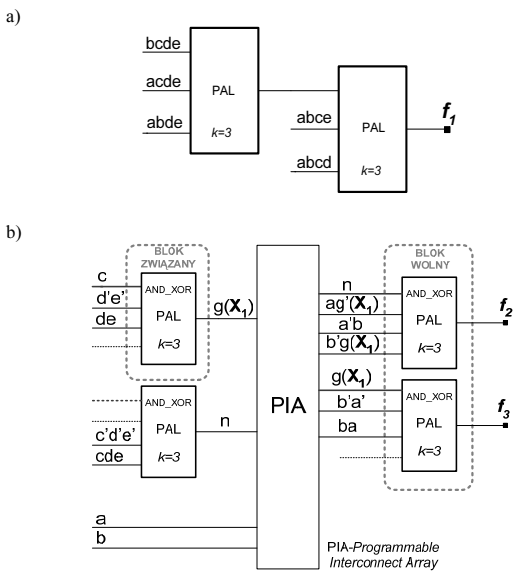
Tego typu przekształcenie pozwala na ograniczenie liczby wyjść bloku związanego wynikające z redukcji liczby wzorców kolumn. Redukcja ta jest efektem negacji odpowiednich wzorców, możliwej do zrealizowania za pomocą elementów XOR.

Poszczególne etapy dekompozycji zespołu funkcji ukierunkowanej na wykorzystanie elementów XOR przedstawione są na rys. 5.



Rys. 5. Etapy dekompozycji zespołu funkcji ukierunkowanej na wykorzystanie elementów XOR

Fig. 5. Decomposition steps of multi-output function based on utilization of XOR gates



Rys. 6. Implementacja układu *rd53* za pomocą bloków logicznych typu PAL z bramką XOR

Fig. 6. Implementation of *rd53* by means of the PAL-based logic blocks with XOR gate

W końcowym efekcie, wykorzystując bramki XOR do realizacji zespołu funkcji $f: B^5 \rightarrow B^3$ wykorzystano **sześć bloków logicznych** typu PAL zawierających trzy iloczyny wraz z bramką XOR. Dwa spośród nich wykorzystano do realizacji funkcji f_1 metodą klasyczną. Uzyskane rozwiązanie przedstawione na rys. 6 stanowi strukturę 2-warstwową. Linią przerywaną oznaczono niewykorzystane iloczyny w bloku logicznym.

4. Wyniki eksperymentów

Jak widać wykorzystanie występującego powszechnie w strukturach CPLD elementu XOR prowadzi do najlepszego rozwiązania. Trzeba przyznać, że od pewnego czasu systemy komercyjne są w stanie wykorzystywać elementy XOR zawarte w strukturach programowalnych, uzyskując rozwiązania lepsze od rozwiązań klasycznych. Dalekie są one jednak od rozwiązań uzyskiwanych metodami wspieranymi elementami dekompozycji. Szereg różnorodnych wyników eksperymentów można znaleźć w [2]. Zbiorcze zestawienie wyników dla analizowanego układu zamieszczono w tabeli 1. W przypadku komercyjnego systemu syntezy firmy ALTERA przedstawiono rozwiązanie dla układów rodziny MAX 5000, w których bloki logiczne zawierają tylko trzy iloczyny i bramkę XOR. W wierszu ALTERA zawarto wyniki optymalizowane pod kątem liczby bloków (*area*) oraz liczby warstw logicznych (*speed*).

Tab. 1. Wyniki eksperymentów dla układu testowego *rd53*

Tab. 1. Experimental results for *rd53* benchmark

METODA	BLOKI/ WARSTWY PAL (K=3) TYLKO 3 X AND	BLOKI/ WARSTWY PAL (K=4) TYLKO 4 X AND	BLOKI/ WARSTWY 3 X AND + XOR
Klasyczna	15/3	11/2	-
Oparta na dekompozycji kolumnowej	9/3	7/3	-
ALTERA (*area*)	15/3	-	11/3
ALTERA (*speed*)	15/3	-	11/3
Dekompozycja + XOR	-	-	6/2

5. Podsumowanie

W artykule przedstawiono początki prac związanych z koncepcją rozszerzenia modelu dekompozycji kolumnowej, których celem jest opracowanie efektywnych modeli dekompozycji pozwalających do wykorzystanie elementów XOR. Istota prac sprowadza się do opracowania algorytmów dekompozycji dopasowanej do architektury układów CPLD.

Wyniki wstępnych eksperymentów potwierdzają możliwość znaczącej poprawy efektywności znanych metod syntezy dedykowanych dla struktur CPLD typu PAL. Nadchodzi czas na opracowanie prototypowych narzędzi programowych pozwalających na przeprowadzenie szeregu badań eksperymentalnych. Część algorytmów można bezpośrednio zaadoptować z programu PALDec [2]. Konieczne jest jednak zmodyfikowanie sposobu poszukiwania dopełnień wzorców kolumn dla zespołu funkcji poprzez odpowiednie użycie grafu niezgodności i dopełnień [2]. Mamy nadzieję, że uzyskane wyniki potwierdzą właściwy kierunek poszukiwań.

6. Literatura

[1] D. Kania, W. Grabiec: Synteza logiczna przeznaczona dla struktur CPLD z elementami XOR. X Konferencja Naukowa: Reprogramowalne Układy Cyfrowe, Szczecin 2007

[2] D. Kania: Synteza logiczna przeznaczona dla matrycowych struktur logicznych typu PAL. Zeszyty Naukowe Politechniki Śląskiej, Gliwice 2004

[3] R.L. Ashenhurst: The decomposition of switching functions, Proceedings of an International Symposium on the Theory of Switching, April 1957

[4] H.A. Curtis: The Design of switching Circuits, D.van Nostrand Company Inc., Princeton, New Jersey, Toronto, New York 1962

[5] Collaborative Benchmarking Laboratory, Department of Computer Science at North Carolina State University, http://www.cbl.ncsu.edu.

Artykuł recenzowany