ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996
Seria: AUTOMATYKA z. 118 Nr kol. 1338

Walery SOŁOWJEW, Marek GRUSZEWSKI
Politechnika Białostocka

SYNTEZA SWOBODNEJ LOGIKI REJESTROWEJ NA BAZIE PLD

Streszczenie. W pracy rozpatrywany jest problem syntezy logiki rejestrowej na bazie programowalnych układów logicznych - PLD (*Programmable Logic Devices*). Proponowana metoda polega na kolejnym formowaniu grup elementów rejestrowej logiki, realizowanej na jednym układzie PLD. W celu poprawy otrzymanego rozwiązania wykorzystuje się iteracyjny algorytm wzajemnej zamiany elementów z różnych grup. Głównym kryterium optymalizacji jest minimalizacja liczby PLD, dodatkowym - minimalizacja liczby zewnętrznych połączeń między poszczególnymi układami PLD. W pracy przedstawione są także analityczne metody oceny minimalnej liczby potrzebnych układów PLD.

SYNTHESIS REGISTERED LOGIC ON PROGRAMMABLE LOGIC DEVICES

Summary. Problem of synthesis on Programmable Logic Devices (PLDs) random registered logic - set of any elements of the project as flip-flops, latchs, registers and other is considered. Consecutive algorithm of formation of random logic element groups which will to implement on separate PLD is offered. For improvement of the received results iterative algorithm of mutual exchange of elements from different groups is used. Main criterion of optimization is minimization of number PLD, auxiliary - minimization of number of external nets of the circuit.

1. Wstęp

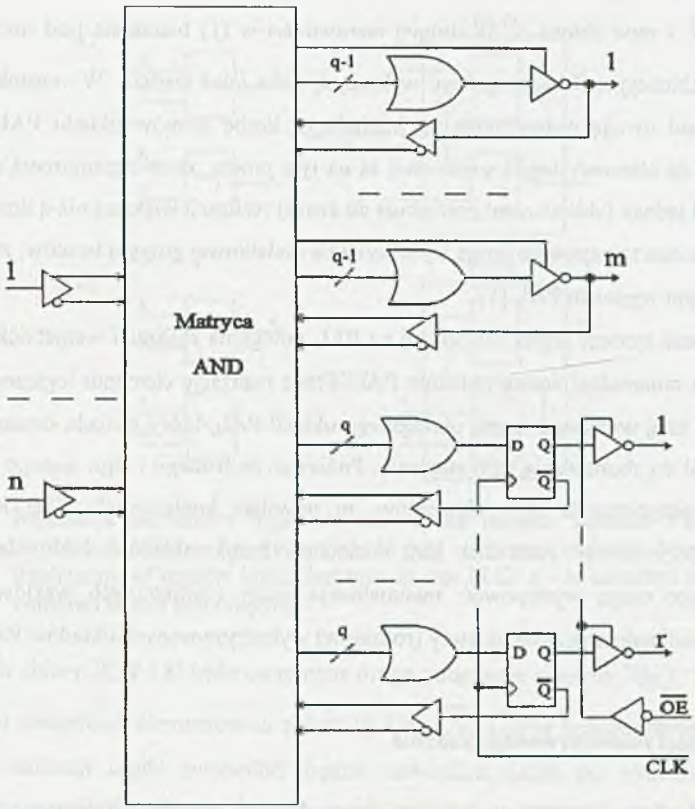
Obecnie systemy cyfrowe budowane są z oddzielnych bloków funkcjonalnych (mikroprocesorów, buforów szyn, podsystemów pamięci itd.). Tym nie mniej w schematach projektów często można zauważyć oddzielne elementy, nie wchodzące w skład żadnego z bloków funkcjonalnych. Jest to tak zwana logika swobodna (nieregularna, rozproszona), (random logic) [1]. Zwykle z pojęciem logiki swobodnej wiąże się elementy typu kombinacyjnego. Jednak w projektach systemów cyfrowych również często spotykane są oddzielne elementy i węzły funkcjonalne typu sekwencyjnego: przerzutniki, rejestry, liczniki i inne. Ogół podobnych elementów nazywany jest rejestrową logiką swobodną projektu. Tradycyjnie logika swobodna realizowana jest na elementach małej i średniej skali integracji. Dlatego potrzebne są elementy różnego typu, z których wiele nie jest w pełni wykorzystanych:

zostają, nie potrzebne do projektu, oddzielne sekcje mikroukładów. W rezultacie wzrasta liczba układów scalonych i całkowita liczba ich wyprowadzeń a więc i liczba otworów montażowych płytki drukowanej, zwiększa się moc pobierana przez układ, komplikuje się rozwiązywanie zadań projektowania konstruktorskiego i w skutek tego wydłuża się czas produkcji i zwiększa się koszt produktu. Programowalne układy logiczne (PLD - Programmable Logic Devices) stanowią nową bazę elementów [2], coraz szerzej wykorzystywaną do budowy różnorodnych urządzeń cyfrowych [3-5]. Wykorzystanie układów PLD do realizacji logiki swobodnej typu kombinacyjnego pokazało wysoką ich efektywność [6]. W pracy rozpatrywany jest problem syntezy swobodnej logiki rejestrowej na bazie układów PLD. W charakterze układów PLD na początku rozpatrzmy programowalne matryce logiczne (PAL - Programmable Array Logics - PALs) typu rejestrowego.

2. Struktura rejestrowego układu PAL

Uogólniona struktura rejestrowego układu PAL pokazana jest na rys.1. Składa się ona z programowalnej matrycy AND, zawiera n dwufazowych wejść, m kombinacyjnych i r rejestrowych wyjść. Matryca AND pozwala formować na swoich wyjściach dowolny iloczyn n zmiennych. Wyjściowe szyny matrycy AND przedstawiają sobą termy (product terms) układu PAL. Makrokomórka kombinacyjnego wyjścia układu PAL zawiera bramkę sumy logicznej - OR, do której podłączonych jest q-1 termów matrycy AND oraz inwerter wyjściowy, którego trzeci stan sterowany jest oddzielnym termem. Każde wyjście kombinacyjne ma pętlę sprzężenia zwrotnego z matrycą AND, dzięki czemu przy ustawieniu inwertera w trzeci stan wyjście PAL-u może być wykorzystywane jako wejście. Makrokomórki rejestrowych wyjść układu PAL zawierają dodatkowo przerzutnik typu D. W tym przypadku pętla sprzężenia zwrotnego zrealizowana jest poprzez podłączenie zanegowanego wyjścia przerzutnika na wejście matrycy AND. Wszystkie przerzutniki układu PAL synchronizowane są wspólnym sygnałem CLK, a inwertery wyjściowe sterowane są wspólnym sygnałem uaktywnienia wyjść $\overline{OE}$. Układ rejestrowy PAL (zobrazowany na rys. 1) będziemy oznaczać PAL(n,m,r,q).

W ogólnym przypadku przy syntezie logiki rejestrowej na układach PAL typ przerzutnika wyjściowego nie gra szczególnej roli, jakkolwiek rozwinięte architektury PAL pozwalają się ustawiać (programować) na dowolny typ przerzutnika wyjściowego, w tym z asynchronicznym ustawianiem i zerowaniem.



Rys. 1. Ogólna struktura rejestrowego układu PAL
Fig. 1. General structure registered PAL

2. Postawienie zadania

Niech U = $\{u_1,...,u_S\}$ będzie zbiorem elementów logiki swobodnej projektu. Niech każdy element u_i, u_i U, ma zbiór $X(u_i)$ zmiennych wejściowych, zbiór $Y(u_i)$ zmiennych wyjściowych, których wartości formowane są na kombinacyjnych wyjściach PAL, oraz zbiór $D(u_i)$ zmiennych wyjściowych, których wartości formowane są na rejestrowych wyjściach PAL. Warunki realizacji jednego elementu u_i na PAL(n,m,r,q) określone są spełnieniem nierówności:

$$\begin{aligned} &| Y(u_i) | \le m; \\ &| X(u_i) | + | Y(u_i) | \le n + m; \\ &| D(u_i) | \le r, \end{aligned} \qquad (1)$$

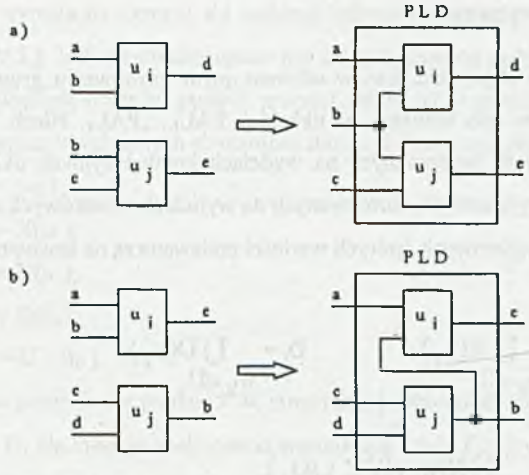
gdzie $|A|$ - moc zbioru A. W drugiej nierówności w (1) bierze się pod uwagę fakt, że wyjścia kombinacyjne PAL mogą być wykorzystywane jako wejścia. W warunkach (1) nie bierze się pod uwagę ograniczenia ze względu na liczbę termów układu PAL, ponieważ zakłada się, że elementy logiki swobodnej są na tyle proste, że te ograniczenia spełniane są zawsze. Jeśli jednak jakiś element potrzebuje do swojej realizacji większej niż q liczby termów, to zawsze można to zapewnić drogą wykorzystania dodatkowej grupy q termów, związanych z jakimś wolnym wyjściem PAL [1].

Zadanie syntezy logiki swobodnej na PAL polega na realizacji wszystkich elementów zbioru U na minimalnej liczbie układów PAL. Przez realizację elementu logicznego na PAL rozumie się tutaj wyznaczenie mu określonego układu PAL, który posiada dostatecznie dużo wejść i wyjść do zbudowania tego elementu. Ponieważ do jednego i tego samego układu PAL może być przypisanych kilka elementów, to powstaje konieczność takiej ich wspólnej realizacji, przy której potrzebna jest minimalna liczba układów PAL. Jako kryteria wspomagające mogą występować: minimalizacja liczby zewnętrznych węzłów (połączeń) schematu, minimalizacja nomenklatury (rodzajów) wykorzystywanych układów PAL i inne.

3. Osobliwości rozwiązywanego zadania

Jeżeli dwa elementy u_i i u_j ze zbioru U mają wspólne zmienne wejściowe, tj. $X(u_i) \cap X(u_j) \neq \varnothing$, $u_i, u_j \in U$, $i \neq j$, gdzie $\varnothing$ - zbiór pusty, to w celu wykluczenia dublowania się zmiennych wejściowych, racjonalne jest realizowanie ich na jednym układzie PAL (rys. 2a). Może się zdarzyć, że zmienne formowane na wyjściach elementu u_j są zmiennymi wejściowymi elementu u_i, tj. $X(u_i) \cap (Y(u_j) \cup D(u_j)) \neq \varnothing$. Takie elementy również należy starać się realizować razem na jednym układzie PAL, ponieważ pozwoli to wykorzystać pętle sprzężenia zwrotnego PAL do przekazania wartości zmiennych wyjściowych elementu u_j na wejście elementu u_i (rys. 2b). Przy tym wartości zmiennych zbioru $Y(u_j)$ brane są bezpośrednio z wyjść kombinacyjnych układu PAL, a zbiory $D(u_j)$ - z odwracających wyjść przerzutników (rys. 1).

Będziemy także uważać, że zbiory zmiennych wyjściowych różnych elementów logicznych nie przecinają się: $Y(u_i) \cap Y(u_j) = \varnothing$ i $D(u_i) \cap D(u_j) = \varnothing$, przy $i \neq j$ dla wszystkich $u_i, u_j \in U$, tj. nie dopuszcza się formowania sygnałów drogą łączenia montażowego. Jeżeli jednak w jakimś schemacie zdarza się to, przy rozwiązywaniu zadania wspólne zmienne wyjściowe należy oznaczyć w różny sposób.



Rys.2. Realizacja elementów logiki rejestrowej na jednym układzie PLD: a) przy wspólnych zmiennych wejściowych; b) przy wspólnych zmiennych wyjściowych
Fig. 2. Realization of register logic elements on one PLD: a - at common inputs; b - at common inputs and outputs

Niech zbiory X, Y i D będą otrzymane drogą połączenia zbiorów $X(u_i)$, $Y(u_i)$ i $D(u_i)$ odpowiednio wszystkich elementów u_i zbioru U, $i = \overline{1,S}$. Liczba wykorzystywanych wejść PAL przy realizacji logiki swobodnej będzie minimalna, jeżeli po pierwsze, w pełni wykluczone zostanie dublowanie zmiennych wejściowych, a po drugie, jeżeli wszystkie logiczne elementy, mające wspólne zmienne wyjściowe i wejściowe będą realizowane na jednym układzie PAL. W drugim ostatnim przypadku liczba potrzebnych wejść PAL zmniejsza się o wartość $X \cap (Y \cup D)$. W żaden sposób nie można zmniejszyć liczby wykorzystywanych wyjść PAL, ponieważ do formowania wartości każdej zmiennej wyjściowej potrzebne jest oddzielne wyjście PAL. Na podstawie przedstawionego rozumowania dolną granicę liczby układów PAL przy realizacji logiki swobodnej określa się następującym wyrażeniem:

$$T_1 = \max\left(\left\lceil\frac{|D|}{r}\right\rceil, \left\lceil\frac{|Y|}{m}\right\rceil, \left\lceil\frac{|X\setminus(Y\cup D)|+|Y|}{n+m}\right\rceil\right); \qquad (2)$$

gdzie]A[- najmniejsza liczba całkowita, większa lub równa A. Pierwszy element w nawiasach okrągłych wyrażenia (2) określa dostateczność wyjść rejestrowych PAL, drugi - wyjść kombinacyjnych, a trzeci - łączną liczbę wejść i wyjść kombinacyjnych. Uwzględnia się tutaj, że $|X \cap (Y \cup D)|$ wartości zmiennych wejściowych będą podane na wejście matrycy AND za pomocą pętli sprzężeń zwrotnych układu PAL.

4. Sekwencyjny algorytm syntezy

Sedno danego algorytmu tkwi w sekwencyjnym formowaniu grup elementów logiki swobodnej $U_1,...,U_T$ w celu realizacji na układach $PAL_1,...,PAL_T$. Niech Y_t będzie zbiorem zmiennych wyjściowych, formowanych na wyjściach kombinacyjnych układu PAL_t, a D_t - zbiorem zmiennych wyjściowych, formowanych na wyjściach rejestrowych układu PAL_t, i X_t - zbiorem zmiennych wejściowych, których wartości podawane są na zewnętrzne wejścia układu PAL_t, $t = \overline{1,T}$, tj.:

$$Y_t = \bigcup_{u_k \in U_t} Y(u_k); \qquad D_t = \bigcup_{u_k \in U_t} D(u_k); \tag{3}$$

$$X_t = \left(\bigcup_{u_k \in U_t} X(u_k) \right) \setminus (Y_t \cup D_t).$$

Na podstawie (1) można określić warunki możliwości dołączenia danego elementu u_i, $u_i \in U$, do grupy U_t:

$$\begin{aligned} &|Y_t \cup Y(u_i)| \le m; \\ &|(X_t \cup X(u_i)) \setminus (Y_t \cup Y(u_i) \cup D_t \cup D(u_i))| + |Y_t \cup Y(u_i)| \le n + m; \\ &|D_t \cup D(u_i)| \le r. \end{aligned} \tag{4}$$

Przy tym zmniejszenie liczby zewnętrznych węzłów schematu w rezultacie realizacji elementu u_i na PAL_t określa się wartością:

$$\begin{aligned} C_t(u_i) = &|X_t \cap X(u_i)| + |X_t \cap (Y(u_i) \cup D(u_i))| + \\ &+ |(Y_t \cup D_t) \cap X(u_i)|. \end{aligned} \tag{5}$$

Pierwsza składowa w (5) określa zmniejszenie liczby zewnętrznych węzłów schematu jako wynik wykluczenia dublowania się zmiennych wejściowych; druga - jako wynik wykorzystania pętli sprzężeń zwrotnych PAL w celu podania na wejście matrycy AND wartości zmiennych wejściowych elementu u_i formowanych przez inne elementy grupy U_t.

W ogólnym przypadku sekwencyjny algorytm syntezy logiki swobodnej na PAL wygląda następująco:

1. Przyjmuje się t:=0.
2. Przyjmuje się t:=t+1. Zaczyna się formowanie grupy elementów logicznych U_t. Jako element bazowy grupy U_t włącza się element u_i, $u_i \in U$, dla którego spełnione jest:

   $|X(u_i)| + |Y(u_i)| + + |D(u_i)| = \max$

(jako pierwszy wybiera się element, do realizacji którego potrzebna jest maksymalna liczba wyprowadzeń PAL). Jeśli takich elementów jest kilka, to spośród nich wybiera się element u_i, który ma największą liczbę nie pustych przecięć zbioru $X(u_i)$ ze zbiorami zmiennych wejściowych i wyjściowych innych elementów zbioru U. Przyjmuje się:

$U_t := \{u_i\}$;

$X_t := X(u_i)$;

$Y_t := Y(u_i)$;

$D_t := D(u_i)$;

$U := U \setminus \{u_i\}$.

3. Jeżeli $U = \varnothing$, to przejście do punktu 5. W innym przypadku do zbioru U_t włącza się element u_i, $u_i \in U$, dla którego spełnione są warunki (3) i cena $C_t(u_i)$ włączenia elementu u_i do grupy U_t jest maksymalna. Przyjmuje się:

$U_t := U_t \cup \{u_i\}$;

$X_t := (X_t \cup X(u_i)) \setminus (Y(u_i) \cup D(u_i))$;

$Y_t := Y_t \cup Y(u_i)$;

$D_t := D_t \cup D(u_i)$;

$U := U \setminus \{u_i\}$.

Punkt 3 powtarza się dopóty, dopóki do zbioru U_t można włączyć choćby jeden element zbioru U.

4. Jeżeli $U = \varnothing$, przejście do punktu 5, w innym przypadku przejście do pkt. 2.
5. Koniec.

5. Algorytm iteracyjny

Formowanie grup elementów logicznych $U_1,...,U_t$ za pomocą algorytmu sekwencyjnego ma na celu maksymalne wykorzystanie wyprowadzeń każdego układu PAL. W rezultacie może się okazać, że ostatni układ PAL_T nie jest w pełni wykorzystany, a poprzednie zapełnienia układów $PAL_1,..,PAL_{T-1}$ nie zawsze są optymalne z punktu widzenia minimalizacji liczby zewnętrznych połączeń schematu. W tym przypadku można zoptymalizować rozwiązanie za pomocą algorytmu iteracyjnego (permutacyjnego).

Niech elementy u_i i u_j są przypisane do realizacji na różnych układach PAL, tj. należą do różnych grup: $u_i \in U_t$, $u_j \in U_h$, $t \neq h$. Przez U_t^* i U_h^* oznaczmy zbiory, które utworzone

są ze zbiorów U_t i U_h odpowiednio drogą wzajemnej wymiany elementów u_i i u_j, tj. $U_t^* = (U_t \setminus \{u_i\}) \cup \{u_j\}$, $U_h^* = (U_h \setminus \{u_j\}) \cup \{u_i\}$. Warunki realizacji grup U_t^* i U_h^* na PAL określone są spełnieniem nierówności:

$$|Y_t^*| \le m;\ |X_t^*| + |Y_t^*| \le n + m;\ |D_t^*| \le r; \qquad 6)$$

$$|Y_h^*| \le m;\ |X_h^*| + |Y_h^*| \le n + m;\ |D_h^*| \le r,$$

gdzie zbiory X_t^*, Y_t^*, D_t^*, X_h^*, Y_h^*, D_h^* określone są na podstawie (3) (tylko w miejsce elementów zbioru U_t rozpatruje się elementy zbiorów U_t^* i U_h^*). Zmianę liczby zewnętrznych węzłów schematu jako wynik wzajemnej wymiany elementów u_i i u_j określa się wartością:

$$C_{th}(u_i, u_j) = \begin{matrix} |X_t^*| + |Y_t^*| + |D_t^*| + |X_t^*| + |Y_h^*| + |D_h^*| - \\ -|X_t| - |Y_t| - |D_t| - |X_h| - |Y_h| - |D_h| \end{matrix} \qquad (7)$$

Iteracyjny algorytm ulepszenia rozwiązania zadania syntezy logiki swobodnej na układach PAL wygląda następująco:

1. Spośród wszystkich elementów zbioru U znajdowana jest para u_i i u_j, $u_i \in U_t$ i $u_j \in U_h$, $t \ne h$, $t = \overline{1,T-1}$, $h = \overline{t+1,T}$, dla której spełnione są warunki (6) i cena $C_{th}(u_i,u_j)$ (obliczona zgodnie z (7)) jest ujemna. Przyjmuje się: $U_t := U_t^*$; $U_h := U_h^*$.
2. Punkt 1 powtarza się dopóty, dopóki może być znaleziona choćby jedna para elementów nadająca się do wymiany.
3. Koniec.

6. Przykład

Rozpatrzmy przykład realizacji na PAL(4,2,2,4) elementów $u_1,\ldots,u_5$ swobodnej logiki rejestrowej projektu, określonych następującymi zbiorami:

$X(u_1) = \{x_1,x_2,x_3,x_4,y_1,y_2,d_1\}$; $Y(u_1) = \varnothing$; $D(u_1) = \{d_2\}$;

$X(u_2) = \{x_1,x_2,x_3,x_4,y_2,d_1,d_2\}$; $Y(u_2) = \{y_1\}$; $D(u_2) = \varnothing$;

$X(u_3) = \{d_1,d_2\}$; $Y(u_3) = \{y_2\}$; $D(u_3) = \varnothing$;

$X(u_4) = \{y_1,y_2\}$; $Y(u_4) = \varnothing$; $D(u_4) = \{d_1\}$;

$X(u_5) = \{x_5,x_6,x_7,y_3\}$; $Y(u_5) = \{y_4\}$; $D(u_5) = \varnothing$;

$X(u_6) = \{x_6, x_7, x_8, y_4\}$; $Y(u_6) = \{y_3\}$; $D(u_6) = \varnothing$;

$X(u_7) = \{x_8, y_3, y_4, d_3\}$; $Y(u_7) = \varnothing$; $D(u_7) = \{d_4\}$;

$X(u_8) = \{x_5, y_3, y_4, d_4\}$; $Y(u_8) = \varnothing$; $D(u_8) = \{d_3\}$.

Ponieważ mamy tutaj $|X| = 16$, $|D| = |Y| = 4$ i $|X\setminus(Y \cup D)| = 8$, to dolna granica T_1 liczby układów PAL, niezbędnych do realizacji logiki swobodnej projektu (obliczona zgodnie z (2)), wynosi 2.

W pierwszym kroku algorytmu sekwencyjnego jako bazowy element grupy U_1 wybrany zostaje element u_1, ponieważ dla niego $|X(u_1)| + |Y(u_1)| + |D(u_1)| = \max = 8$. Jako drugi element grupy U_1 dołączany jest u_2, ponieważ $C_1(u_2) = \max = 8$. Następnie do grupy U_1 kolejno włączane są elementy u_3 i u_4. Więcej do grupy U_1 nie da się dołączyć żadnego elementu, dlatego też formowanie grupy U_1 zostaje zakończone. Analogicznie jako bazowy element grupy U_2 wybrany zostaje element u_5, następnie kolejno włączane są elementy u_6, u_7 i u_8. Wykonanie algorytmu iteracyjnego nie daje ulepszenia otrzymanego rozwiązania. W ten sposób, w tym przykładzie dzięki zastosowaniu układów PAL do realizacji logiki swobodnej, udało się zmniejszyć liczbę zewnętrznych węzłów schematu z 42 do 16.

Należy zauważyć, że elementy u_1 i u_2 (każdy oddzielnie) nie mogą być realizowane na PAL z zadanymi parametrami, ponieważ dla nich naruszone są warunki (1). Ale jednak łącznie z elementami u_3 i u_4 jest to możliwe dzięki temu, że na rejestrowych wyjściach PAL formowane są wartości zmiennych wejściowych elementów u_1 i u_2.

LITERATURA

1. Sołowiew W.W.: Projektowanie węzłów funkcjonalnych systemów cyfrowych na programowalnych układach logicznych . Bestprint, Mińsk 1996, s. 252.
2. PAL Device Data Book and Design Guide. Advanced Micro Devices, 1995.
3. Sołowiew W.W.: Synteza mikroprogramowalnych automatów na programowalnych matrycach logiki. Wiadomości Akademii Nauk Białorusi. Ser. fiz.-techn. nauk, nr 1, 1994, s.68-72.
4. Sołowiew W.W.: Wykorzystanie programowalnych matryc logiki do syntezy schematów kombinacyjnych. AWT, nr 4/5, 1995, s. 53 - 56.
5. Sołowiew W.W.: Złożoność realizacji urządzeń sterowania logicznego na PAL. Wiadomości Rosyjskiej Akademii Nauk, Teoria i Systemy Sterowania, nr 5, 1995, s.248-256.
6. Solovjev V., Vasiljev A.: Synthesis of any logic on Programmable Logic Devices. International Conference on Computer-Aided Design of Discrete Devices. CAD'95, Minsk-Szczecin, November 15-17, 1995, pp.193-198.

Recenzent: Prof. dr hab. inż. Józef Ober

Wpłynęło do Redakcji do 30.06.1996 r.

Abstract

The heaviest difficulties at designing of digital robot systems and automatics are caused with synthesis of the sequential circuits, containing elements memories in a kind of flip-flops, latchs, registers and other. In a number of cases application of formal methods of synthesis, for example, theory of automatic devices, not always probably or results in the inefficient decisions. Therefore the sequential circuits are frequently designed on the basis of experience and intuition of the developers. But, as the majority desiners operate with categories of elements of a small and average degree of integration, and the sequential circuit are under construction on their basis.

In the given work is offered the sequential circuits to build on new element base - Programmable Logic Devices (PLDs). Thus a way of thinking of the designer does not change: he develops any the sequential circuit from gates, latchs and flip-flops, and then this circuit "is put" in a given structure PLD. Any set of connected among themselves gates and triggers refers to as by register logic. In work algorithm of synthesis of register logic on PLD is considered, which consists of consecutive formation of groups of elements of register logic, sold on one PLD. For improvement of the received decision iterative algorithm of mutual exchange of elements from different groups is used. Main criterion of optimization is minimization of number PLD, auxiliary - minimization of number of external communications(connections) between PLD. Programm realization of a method and its(her) communication(connection) with known packages of automated designing (CUPL and PCAD) is represented also.