UDC 681.234

A.A. BARKALOV[1], I.Y. ZELENYOVA[2], A.S. LAVRIK[2]

[1]*University of Zielona Gora, Zielona Gora, Poland*
[2]*DonNTU, Donetsk, Ukraine*

OPTIMIZATION OF MICROPROGRAM CONTROL UNIT BASED ON CPLD FEATURES

The method of hardware reduction is proposed which is oriented on compositional microprogram control units and PAL-based CPLD chips. The method is based on a wide fan-in of PAL macrocells allowing using more than one source of microinstruction address. Such approach permits to minimize the number of PAL macrocells used for transformation of microinstruction address. Conditions for this method' application and example of its application are given. The results of experiments are shown.

Keywords*: compositional microprogram control unit, operational linear chain, microinstruction, address transformation.*

Introduction

A control unit is one of the very important parts of any digital system [1]. If a control algorithm to be interpreted is a linear one, then it can be implemented using the model of compositional microprogram control unit (CMCU) [2]. The complex programmable logic devices (CPLD) with programmable array logic (PAL) macrocells are widely used for implementation of logic circuits of control units [3, 4]. The high cost of such devices requires optimization of hardware amount in the circuit of CMCU. One of the ways for this task solution is decrease of the number of conjunctive terms in the sum-of-product (SOP) forms of the functions to be Implemented [5, 6]. This article proposes an approach for this problem solution based on a wide fan-in of PAL macrocells [3, 4]. This approach aims at CMCU with address transformer [2] and a control algorithm is represented as graph-scheme of algorithm (GSA) [7].

1. Peculiarities of CMCU with address transformer

Let GSA Γ be represented by sets of vertices B and arcs E. Let $B = \{b_0, b_E\} \cup E_1 \cup E_2$, where b_0 is an initial vertex, b_E is a final vertex, E_1 is a set of operator vertices, where $|E_1| = M$, and E_2 is a set of conditional vertices. A vertex $b_q \in E_1$ contains a microinstruction $Y(b_q) \subseteq Y$, where $Y = \{y_1, ..., y_N\}$ is a set of datapath microoperations [7]. Each vertex $b_q \in E_2$ contains a single element of a set of logical conditions $X = \{x_1, ..., x_L\}$. Let GSA Γ be a linear GSA, that is a GSA with more than 75% of operator vertices.

Let us form a set of operational linear chains (OLC) $C = \{\alpha_1, ..., \alpha_G\}$ for GSA Γ, where each OLC $\alpha_g \in C$ is a sequence of operator vertices and each pair of its adjacent components corresponds to some arc of the GSA. Each OLC $\alpha_g \in C$ has only one output O_g and arbitrary number of inputs. Formal definitions of OLC, its input and output can be found in [2]. Each vertex $b_q \in E_1$ corresponds to microinstruction MI_q kept in a control memory (CM) of CMCU and it has an address $A(b_q)$. The microinstructions can be addressed using

$$R = \lceil \log_2 M \rceil \quad (1)$$

bits, represented by variables $T_r \in T = \{T_1, ..., T_R\}$. Let OLC $\alpha_g \in C$ include F_g components and the following condition takes place:

$$A(b_{gi+1}) = A(b_{gi}) + 1. \quad (2)$$

In equation (2) b_{gi} is the i-th component of OLC $\alpha_g \in C$, where $i = 1, ..., F_g - 1$.

If outputs O_i, O_j are connected with an input of the same vertex, then OLC $\alpha_i, \alpha_j \in C$ are pseudo-equivalent OLC (POLC) [2]. Let us construct the partition $\Pi_C = \{B_1, ..., B_I\}$ of the set $C_1 \subseteq C$ on the classes of POLC. Let us point out that $\alpha_g \in C_1$ if $\langle O_g, B_E \rangle \notin E$. Let us encode the classes $B_i \in \Pi_C$ by binary codes $K(B_i)$ with

$$R_1 = \lceil \log_2 I \rceil \quad (3)$$

bits and use the variables $\tau_r \in \tau = \{\tau_1, ..., \tau_{R_1}\}$ for the

encoding. In this case a GSA Γ can be interpreted using the model of CMCU U_1 with address transformer (Fig. 1).

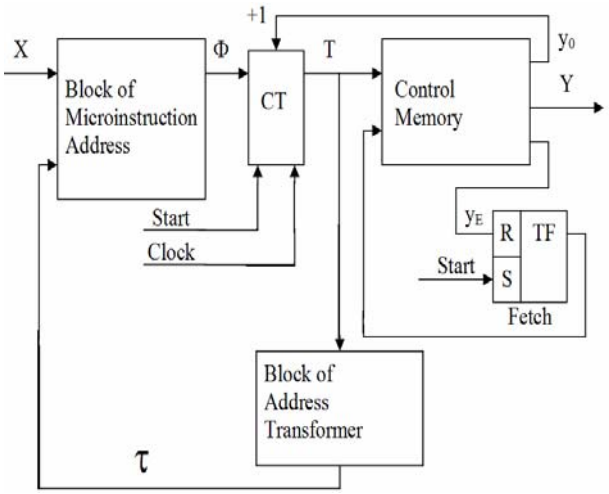

Fig. 1. Structural diagram of CMCU U_1

The pulse Start causes loading of the first microinstruction address into a counter CT and set up of a fetch flip-flop TF. If Fetch = 1, then microinstructions can be read out the control memory CM. If a current microinstruction does not correspond to an OLC output, then a special variable y_0 is formed together with microoperations $Y_q \subseteq Y$. If $y_0 = 1$, then content of the CT is incremented according to the addressing mode (2). Otherwise, block of microinstruction address BMA generates functions

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

to load the next microinstruction address into the CT. In the same time, a block of address transformer BAT generates functions

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

If the output of OLC $\alpha_g \notin C_1$ is reached, then $y_E = 1$. It causes cleaning of TF and operation of CMCU U_1 is terminated.

Such organization of CMCU permits decrease of the number of terms in functions Φ from H_1 till H_0, where H_1, H_0 is the number of terms for equivalent finite state machines (FSM) Moore and Mealy respectively. But block BAT consumes some macrocells or cells of PROM used for implementation of CM. In this article we propose some CMCU U_2, where $H_2 = H_0$ and block BAT consumes less hardware then its counterpart in U_1. Here H_2 means the number of terms in functions Φ for CMCU U_2.

2. Main idea of proposed method

Let us point out that logic circuits for BMA, CT, TF and BAT are implemented as the parts of CPLD. To implement the CM one should use PROM chips with t outputs, where t = 1, 2, 4, 8, 16 [3, 4]. Let us address the components of OLC $\alpha_g \in C_1$ in such a manner that condition (2) takes place and maximal possible amount of classes $B_i \in \Pi_C$ was represented by a single generalized interval of R-dimensional Boolean space. Such addressing needs a special algorithm which should be developed.

Let $\Pi_C = \Pi_A \cup \Pi_B$, where $B_i \in \Pi_A$ if this class is represented by one interval, and $B_i \in \Pi_B$ otherwise. The counter CT is a source of the codes for $B_i \in \Pi_A$. If condition

$$\Pi_B = \emptyset \tag{6}$$

takes place, then block BAT is absent. Otherwise, only output addresses for OLC from classes $B_i \in \Pi_B$ should be transformed. It is enough

$$R_2 = \lceil \log_2(I_B + 1) \rceil \tag{7}$$

bits for such encoding, where $I_B = |\Pi_B|$ and 1 is added to take into account the case when $B_i \in \Pi_A$. Let us point out that some part of these codes can be implemented using free outputs of PROM. Let us use hot-one encoding of microoperations [2] when CM word has N+2 bits. In this case CM can be implemented using

$$R_0 = \lceil (N+2)/t \rceil \tag{8}$$

chips with enough amount of cells (not less than M). Obviously, that R_3 outputs of PROM are free, where

$$R_3 = R_0 * t - N - 2 \tag{9}$$

If condition

$$R_3 \geq R_2, \tag{10}$$

takes place, then CM is a source of the codes for $B_i \in \Pi_B$ and block BAT is absent. Otherwise, we can represent Π_B as $\Pi_E \cup \Pi_D$, where $I_E = |\Pi_E|$, $I_D = |\Pi_D|$. In this case

$$I_E = 2^{R_3} - 1, \tag{11}$$

$$R_4 = \lceil \log_2(I_D + 1) \rceil. \tag{12}$$

The value I_E is decremented to represent the situation that $B_i \in \Pi_E$. The value of I_D is incremented to show the case that $B_i \in \Pi_D$. Thus, only the outputs of OLC $\alpha_g \in B_i$ should be transformed, where $B_i \in \Pi_D$. In common case when $\Pi_i \neq \varnothing (i = A, E, D)$ the following CMCU U_2 is proposed for interpretation of GSA Γ (Fig. 2).

In CMCU U_2, codes $K_A(B_i)$ of the classes $B_i \in \Pi_A$ are represented by variables $T_r \in T$; codes $K_E(B_i)$ of the classes $B_i \in \Pi_E$ are represented by variables $v_r \in V$, where $|V| = R_3$; codes $K_D(B_i)$ of the classes $B_i \in \Pi_D$ are represented by variables

$z_r \in Z$, where $|Z| = R_4$. In CMCU U_2, block BMA implements functions

$$\Phi = \Phi(T, Z, V, X), \tag{13}$$

and block BAT implements functions

$$Z = Z(T). \tag{14}$$

Let symbol $U_i(\Gamma_j)$ stand for implementation of GSA Γ_j by CMCU U_i, and symbol $Q_i(\Gamma_j)$ for the number of macrocells in logic circuit of block BAT for CMCU $U_i(\Gamma_j)$, where i = 1, 2. Let inputs of q-th macrocell of BAT receive L_q logical conditions and let each macrocell have S inputs. Application of the proposed method has sense if condition

$$L_q + R + R_3 + R_4 \le S \tag{15}$$

takes place, where q = 1, … , $Q_2(\Gamma_j)$.

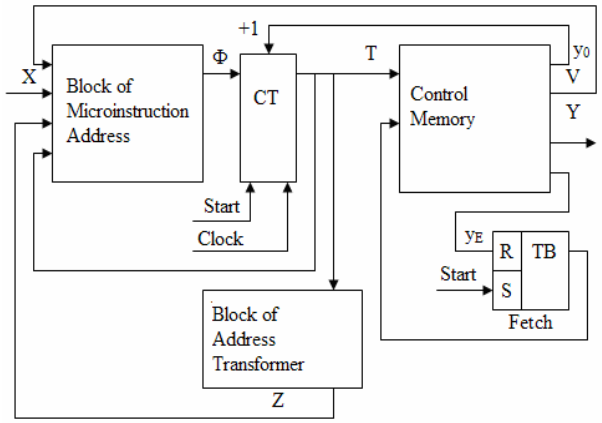

Fig. 2. Structure diagram of CMCU U_2

In this article the method of CMCU U_2 synthesis is proposed with the following steps:

1. Construction of the sets C, C_1 and Π_C for GSA Γ.
2. Microinstruction addressing.
3. Construction of the sets Π_A, Π_E and Π_D.
4. Encoding of the classes $B_i \in \Pi_E \cup \Pi_D$.
5. Construction of control memory content.
6. Construction of transition table for CMCU.
7. Construction of the table for address transformer.
8. Synthesis of the logic circuit of CMCU.

3. Example of application of proposed method

Let the sets $C = \{\alpha_1, ..., \alpha_9\}$, $C_1 = \{\alpha_1, ..., \alpha_8\}$ and $\Pi_C = \{B_1, ..., B_5\}$ are formed for GSA Γ_1, where $\alpha_1 = \langle b_1, b_2 \rangle$, $\alpha_2 = \langle b_3, ..., b_6 \rangle$, $\alpha_3 = \langle b_7, b_8 \rangle$, $\alpha_4 = \langle b_5, ..., b_{13} \rangle$, $\alpha_5 = \langle b_4, ..., b_{17} \rangle$, $\alpha_6 = \langle b_{18}, ..., b_{21} \rangle$, $\alpha_7 = \langle b_{22}, ..., b_{25} \rangle$, $\alpha_8 = \langle b_{26}, ..., b_{28} \rangle$, $\alpha_9 = \langle b_{29}, ..., b_{31} \rangle$, $B_1 = \{\alpha_1\}$, $B_2 = \{\alpha_2, \alpha_3\}$, $B_3 = \{\alpha_4, \alpha_5\}$, $B_4 = \{\alpha_6, \alpha_7\}$, $B_5 = \{\alpha_8\}$. Thus, I = 5, $R_1 = 3$, $\tau = \{\tau_1, \tau_2, \tau_3\}$, M = 31, R = 5.

Let us address the microinstructions using some modification of algorithm from [2]. Now we have $A(b_1) = 00000, \ldots, A(b_{25}) = 110000$, $A(b_{26}) = 11100, \ldots$, $A(b_{28}) = 11110$, $A(b_{29}) = 11001, \ldots, A(b_{31}) = 11011$. Let us construct the Karnaugh map marked by the variables $T_r \in T = \{T_1, ..., T_5\}$ (Fig. 3). This map contains outputs of OLC $\alpha_g \in C$ and code space intervals corresponding to the classes $B_i \in \Pi_C$.

The sign * in this map stands for the case when a vertex $b_q \in E_1$ with address $A(b_q)$ is not the output of OLC $\alpha_g \in C_1$. The following code intervals can be derived from Fig. 3: class B_1 corresponds to interval 0000*, class B_2 to 001**, class B_3 to 01*** and 10000, B_4 to 101** and 11000, class B_5 to 111**. Let us point out that $\alpha_9 \notin C_1$ and class $B_6 = \{\alpha_9\}$ is not considered here.

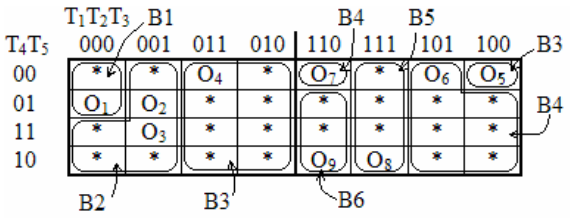

Fig. 3. Karnaugh map for outputs of OLC

The obtained intervals determine the sets $\Pi_A = \{B_1, B_2, B_5\}$ and $\Pi_B = \{B_3, B_4\}$. Let N=13 for GSA Γ_1 and t=4 for PROM chips in use. In this case we can get R_3 = 1 (from (9)) and R_2 = 2 (from (7)). Thus, condition (10) is violated and block BAT should be used in CMCU $U_2(\Gamma_1)$. Let $\Pi_E = \{B_3\}$, then $\Pi_D = \{B_4\}$. Thus, the sets Π_A, Π_E and Π_D are constructed.

Obviously, $V = \{v_1\}$, $Z = \{z_1\}$ let $K_E(B_3) = 1$, $K_D(B_4) = 1$. As it was found, $K_A(B_1) = 0000*$, $K_A(B_2) = 001**$, $K_A(B_5) = 111**$.

The content of control memory is constructed in a trivial way [2] and this step is here omitted. Let us point out, that the cells with addresses 10100 and 11000 include the variable v_1 = 1. Let transitions for classes B2, B3, B4 are described by the following system of generalized transition formulae [7]:

$$\begin{aligned} B_2 &\to x_3 b_9 \vee \overline{x_3} b_{26}; \\ B_3 &\to x_1 b_{18} \vee \overline{x_1} x_2 b_{20} \vee \overline{x_1}\,\overline{x_2} b_{26}; \\ B_4 &\to x_5 b_{27} \vee \overline{x_5} b_5. \end{aligned} \tag{16}$$

The system (16) determines the fragment of transition table with 7 lines (Table 1).

Table 1

Fragment of transition table for CMCU $U_2(\Gamma_1)$

B_i	$K_A(B_i)$	$K_E(B_i)$	$K_D(B_i)$	b_q	$A(b_q)$	X_h	Φ_h	h
	$T_1 \dots T_5$	v_1	z_1					
B_2	001**	0	0	b_9	01000	x_3	D_2	1
				b_{26}	11100	$\overline{x_3}$	$D_1D_2D_3$	2
B_3	*****	1	0	b_{18}	10001	x_1	D_1D_5	3
				b_{20}	10011	$\overline{x_1}x_2$	$D_1D_4D_5$	4
				b_{26}	11100	$\overline{x_1}\,\overline{x_2}$	$D_1D_2D_3$	5
B_4	*****	0	1	b_{27}	11101	x_5	$D_1D_2D_3D_5$	6
				b_5	00100	$\overline{x_5}$	D_3	7

Connection of this table and system (16) is a transparent one. Let us point out that the case $v_1 = z_1 = 0$ corresponds to classes $B_i \in \Pi_A$.

Otherwise, $B_i \in \Pi_E \cup \Pi_D$ and content of the column $K_A(B_i)$ is ignored. This table is a base for construction of the system (13). For example, the following parts of SOP can be derived from Table 1:

$$D_1 = \overline{T_1}\,\overline{T_2}T_3\,\overline{v_1}\,\overline{z_1}\,\overline{x_3} \vee v_1\overline{z_1} \vee \overline{v_1}z_1x_5;$$

$$D_2 = \overline{T_1}\,\overline{T_2}T_3\,\overline{v_1}\,\overline{z_1} \vee v_1\overline{z_1}\,\overline{x_1}\,\overline{x_2} \vee \overline{v_1}z_1x_5.$$

The table of BAT is constructed for the classes $B_i \in \Pi_D$. In our particular case this table includes 2 lines (Table 2).

Table 2

Table of the block BAT for CMCU $U_2(\Gamma_1)$

α_g	$A(O_g)$	B_i	$K_D(B_i)$	Z_j	j
α_6	10100	B_4	1	z_1	1
a_7	1000				2

This table is the base to construct the system (14). In our case this system represented as the following one:

$$z_1 = T_1\overline{T_2}T_3\,\overline{T_4}\,\overline{T_5} \vee T_1T_2\,\overline{T_3}\,\overline{T_4}\,\overline{T_5}.$$

Synthesis of logic circuit of CMCU $U_2(\Gamma_j)$ is reduced to implementation of systems (13)-(14) using PAL macrocells and control memory using PROM chips.

These problems are well-known [2, 6] and they are not discussed in our article.

Let us point out that table of block BAT for CMCU $U_1(\Gamma_1)$ includes 8 lines and $R_1 = 3$. Let macrocell PAL have q=3 terms, then block BAT of CMCU $U_2(\Gamma_1)$ is implemented using $Q_2(\Gamma_1) = 1$ macrocells. If the classes $B_i \in \Pi_C$ of CMCU $U_1(\Gamma_1)$ are encoded in the following way: $K(B_1) = 000,\dots$, $K(B_5) = 100$, then $Q_1(\Gamma_1) = 4$ (if q = 3). In both cases the block BMA is implemented with the same amount of macrocells.

4. Research of proposed method' efficiency

Let's find area of effective application CMCU U_2, using probabilistic approach considered in [2]. According to this approach each GSA Γ is characterized by a share of operational vertex P_1. In case of linear GSA $P_1 \geq 0,75$. In research CMCU matrix models [7] are used, instead their implementations with some PAL macrocells. Thus, hardware amount are characterized by the area of the matrixes, occupied schemes. Conclusion about efficiency of the offered method is made on the basis of investigation for the following relation

$$f = 1 - \frac{S(U_2)}{S(U_1)}, \qquad (17)$$

where $S(U_i)$ - area of the matrix realization of CMCU U_i (i=1 x_1,2).

CMCU U_1 matrix model is shown on Fig. 4.

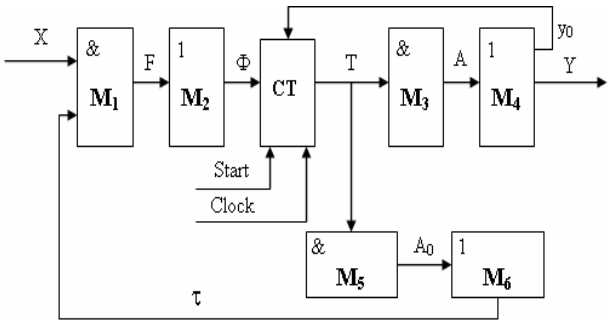

Fig. 4. CMCU U_1 matrix model

In this circuit conjunctive matrix M_1 realizes system F of terms of functions Φ, and disjunctive matrix M_2 realizes functions (4). Conjunctive matrix M_3 realizes the complete decoder having R inputs which outputs represent the addresses of microinstructions from a set A. Disjunctive matrix M_4 realizes functions from a set $\{Y, y_0, y_E\}$. Conjunctive matrix M_5 realizes the terms corresponding to addresses of outputs OLC and

forming set A_0. Disjunctive matrix M_6 realizes functions of system (5).

So, matrixes M_1 and M_2 represent the block of BMA, matrix M_3 and M_4 - the control memory, and matrixes M_5 and M_6 - block BAT. Areas $S(M_j)$ of these matrixes can be defined as follows:

$$S(M_1)_1 = 2(L + R_1) * H_0;$$
$$S(M_2)_1 = H_0 * R;$$
$$S(M_3)_1 = 2R * 2^R;\ S(M_4)_1 = 2^R (N+2); \quad (18)$$
$$S(M_5)_1 = 2R * G;$$
$$S(M_6)_1 = G * R_1$$

Matrix implementation CMCU U_2 has the same appearance, as well as for CMCU U_1. Owing to condition (15) and equalities of capacities of CM for both CMCU fairly following condition:

$$S(M_j)_1 = S(M_j)_2 \quad (j = 1,\ldots,4), \quad (19)$$

In formulas for $S(M_j)_i$ the index *i* defines type of CMCU U_i (i=1,2). For variables quantity reduction in (17) we use results of operation [2]. Let K – number of vertices of ГСА Г, then

$$R = \lceil \log_2 p_1 K \rceil. \quad (20)$$

Block BMA represents Moore machine with

$$G = k_1 p_1 K \quad (21)$$

states. Parameter $k_1 \le 1$ defines average OLC length (number of components) in ГСА Г. Number of classes $B_i \in \Pi_C$ can be found as number of states of the equivalent Mile machine.

$$I = 2,75 + 0,43 k_1 p_1 K. \quad (22)$$

Parameters L and H_0 can be found as:

$$L = (1 - p_1) K / 1,3; \quad (23)$$

$$H_0 = 4,4 + 1,1I. \quad (24)$$

Let $k_2 < 1$ defines number of $B_i \in \Pi_B$ classes, then

$$I_B = k_2 (2,75 + 0,34 k_1 p_1 K); \quad (25)$$

$$R_2 = \lceil \log_2 \lceil k_2 (2,75 + 0,34 k_1 p_1 K) \rceil \rceil. \quad (26)$$

Areas $S(M_5)_2$ and $S(M_6)_2$ can be found as:

$$S(M_5)_2 = 2R * I_B; \quad (27)$$

$$S(M_6)_2 = I_B * R_2. \quad (28)$$

For definition of CMCU U_2 effective area application it is necessary to investigate the function

$$f = 1 - \frac{S(M_1)_2 + \cdots + S(M_6)_2}{S(M_1)_1 + \cdots + S(M_6)_1}, \quad (29)$$

depending on parameters K, p_1, k_1, k_2, N. Some results of our experiments are shown in Fig. 5.

As it follows from Fig 5, the CMCU U_2 consumes less hardware, than the equivalent CMCU U_1 if condition (10) take place. Thus the gain is increased with growth of coefficient k_1. Influence of coefficient k_2 is not so obvious, but its reduction yields in the gain increase. The average gain at K=300, p_1=0.75, k_1=0.5, k_2=0.5 and N = 5, is around 13 %.

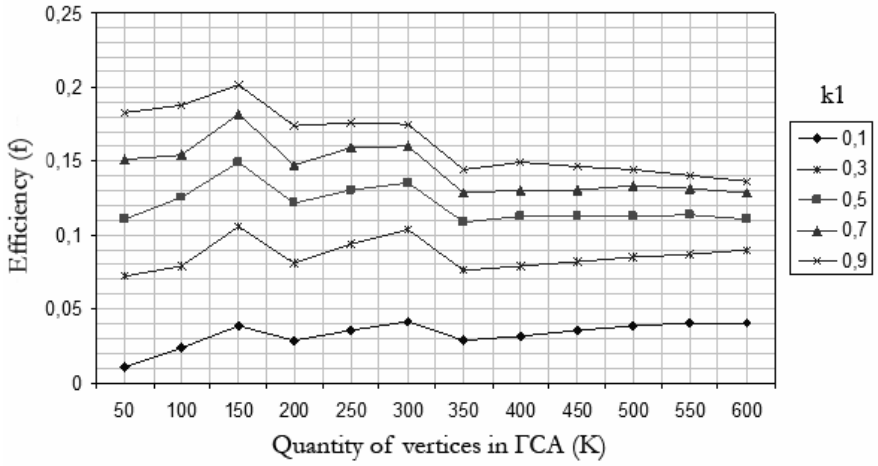

Fig. 5. Dependence of the offered application structure efficiency on quantity of ГСА vertices at different value k_1 (p_1 = 0,75, k_2 = 0,5, N = 5)

Conclusion

The proposed method targets on decrease in hardware amount (the number of macrocells) in the circuit of address transformer. In this case the number of macrocells in the block of microinstruction address as well as the number of PROM chips in the block of control memory does not change. The method is based on use of three sources of the codes of pseudoequivalent OLC classes. It is possible due to a wide fan-in of industrial PAL macrocells. Let us point out that block BAT can be eliminated if one of the conditions (6) or (10) takes place. Our experiments

show that the number of macrocells in block BAT is decreased up to 60-70 % in comparison with known methods of CMCU design. The total decrease in hardware amount is up to 10% in comparison with CMCU $U_1(\Gamma_1)$.

References

1. De Micheli G. Synthesis and Optimization of Digital Circuits / G. De Micheli – NY: McGraw-Hill, 1994. – 636 p.

2. Баркалов А.А. Синтез устройств управления на программируемых логических устройствах / А.А. Баркалов. – Донецк: ДНТУ, 2002. – 262 с.

3. Altera devices [Електронний ресурс] – Режим доступу: http://www.altera.com.

4. Xilinx CPLDs [Електронний ресурс] – Режим доступу: http://www.xilinx.com.

5. Грушницкий Р.И. Проектирование систем с использованием микросхем программируемой логики / Р.И. Грушницкий, А.Х. Мурсаев, Е.П. Угрюмов. – СПб.: БХВ. – Петербург, 2002. – 608 с.

6. Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. / В.В. Соловьев. – М.: ТЕЛЕКОМ, 2001. – 636 с.

7. Baranov S. Logic Synthesis for Control Automata / S. Baranov. – Kluwer Academic Publishers, 1994. – 312 p.

Поступила в редакцию 16.02.2009

Рецензент: д-р техн. наук, ст. наук. співр. В.М. Опанасенко, Інститут кібернетики ім. В.М. Глушкова, Київ, Україна.

ВИКОРИСТАННЯ ОСОБЛИВОСТЕЙ ПЛІС ДЛЯ ОПТИМІЗАЦІЇ СХЕМИ МІКРОПРОГРАМНОГО ПРИСТРОЮ КЕРУВАННЯ

О.О. Баркалов, І.Я. Зеленьова, О.С. Лаврік

В роботі запропоновано метод зменшення апаратурних витрат у логічній схемі композиційного мікропрограмного пристрою керування при реалізації на ПЛІС. Метод базується на великому коефіцієнту об'єднання за входом у макроосередків ПМЛ, що дозволяє використовувати більше ніж одне джерело для адреси мікрокоманди. Такий підхід дозволяє мінімізувати кількість макроосередків ПМЛ що використовуються для перетворювання адреси мікрокоманди. Наведено умови та приклад використання запропонованого методу, а також результати експериментів.

Ключові слова: композиційний мікропрограмний пристрій керування, операторний лінійний ланцюг, мікроінструкція, перетворення адреси.

ИСПОЛЬЗОВАНИЕ ОСОБЕННОСТЕЙ ПЛИС ДЛЯ ОПТИМИЗАЦИИ СХЕМЫ МИКРОПРОГРАММНОГО УСТРОЙСТВА УПРАВЛЕНИЯ

А.А. Баркалов, И.Я. Зеленёва, А.С. Лаврик

В работе предложен метод уменьшения аппаратурных затрат в логической схеме композиционного микропрограммного устройства управления при реализации на ПЛИС. Метод базируется на большом коэффициенте объединения по входу макроячеек ПМЛ, что позволяет использовать больше одного источника для адреса микрокоманды. Такой подход позволяет минимизировать количество макроячеек ПМЛ, которые используются для преобразования адреса микрокоманды. Приведены условия и пример использования предложенного метода, а также результаты экспериментов.

Ключевые слова: композиционное микропрограммное устройство управления, операторная линейная цепь, микроинструкция, преобразование адреса.

Баркалов Александр Александрович – д-р техн. наук, профессор, профессор кафедры электронных вычислительных машин Донецкого Национального технического университета, Донецк, Украина, профессор Зеленогурского университета, Польша, e-mail: A.Barkalov@iie.uz.zgora.pl.

Зеленёва Ирина Яковлевна – канд. техн. наук, доцент, доцент кафедры электронных вычислительных машин Донецкого национального технического университета, Донецк, Украина.

Лаврик Александр Сергеевич – ассистент кафедры электронных вычислительных машин Донецкого национального технического университета, Донецк, Украина.