UDC 681.3

A.A. BARKALOV[1], A.A. KRASICHKOV[2], A.N. MIROSHKIN[2]

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

SYNTHESIS OF MICROPROGRAM CONTROL UNIT WITH CODE SHARING AND MODIFIED LINEAR CHAINS

The new design method for compositional microprogram control units with code sharing is proposed. The method targets on reduction in the number of PAL macrocells in the combinational part of control unit. Some additional control microinstructions containing codes of the classes of pseudoequivalent chains are used for operational linear chains modification. Proposed method is illustrated by an example. Various graph-scheme of algorithm (GSA) research results are illustrated with the diagrams. Most desirable GSA characteristics for using proposed method were obtained.

Key words: *control unit, algorithm, transition, hardware amount, code sharing.*

Introduction

One of the most important blocks of any digital system is its control unit [1] responsible for interplaying of all other system blocks. If an interpreted control algorithm is a linear one, it can be interpreted using the model of compositional microprogram control unit (CMCU) [2]. Recently, the complex programmable logic devices (CPLD) are widely used for implementation of logic circuits [3, 4].

One of the important tasks connected with control unit design is minimization of hardware amount. In case of CPLD, this task can be solved due to decrease of the number of Programmable Array Logic (PAL) macrocells. To solve this problem, the number of terms in sum-of-products (SOP) should be diminished for address functions of CMCU [3, 4]. In this article one of the ways for this problem solution is proposed. The method targets on CMCU with code sharing [2].

1. Peculiarities of CMCU with code sharing

Let a control algorithm to be interpreted be represented by a graph-scheme of algorithm (GSA) Γ [5].Let this GSA be characterized by the set of vertices $B = \{b_0, b_E\} \cup E_1 \cup E_2$ and the set of arcs E, where b_0 is an initial vertex, b_E is a final vertex, E_1 is a set of operator vertices, and E_2 is a set of conditional vertices. Each operator vertex $b_q \in E_1$ contains a collection of microoperations $Y(b_q) \subseteq Y$, where $Y = \{y_1, ..., y_N\}$ is a set of data-path microoperations. Each conditional vertex $b_q \in E_2$ contains some element $x_l \in X$, where $X = \{x_1, ..., x_L\}$ is a set of logical conditions (input signals). A GSA Γ is named a linear GSA [2] if the number of its operator vertices exceeds 75% of the total their number in GSA.

Let the set $C = \{\alpha_1, ..., \alpha_G\}$ be constructed for GSA Γ, where $\alpha_g \in C$ is an operational linear chain (OLC) [2]. Any component b_{gi} of OLC $\alpha_g \in C$ belongs to the set E_1 $(i = 1, ..., F_g)$. Each pair of adjacent components b_{gi}, b_{gi+1} corresponds to the arc $< b_{gi}, b_{gi+1} > \in E$, where $i = 1, ..., F_g - 1$, $g = 1, ..., G$. Each OLC $\alpha_g \in C$ has only one output O_g and the 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 the cell of control memory (CM) with address A_q. It is enough

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

bits for microinstruction addressing, where $M = |E_1|$. Let each OLC $\alpha_g \in C$ include F_g components and $Q = \max(F_1, ..., F_G)$. Let each OLC $\alpha_g \in C$ be encoded by binary code $K(\alpha_g)$ having

$$R_1 = \lceil \log_2 G \rceil \qquad (2)$$

bits and variables $\tau_r \in \tau$ be used for such encoding, where $|\tau| = R_1$. Let each component $b_q \in E_1$ be encoded by binary code $K(b_q)$ having

$$R_2 = \lceil \log_2 Q \rceil \qquad (3)$$

bits and variables $T_r \in T$ be used for this encoding, where $|T| = R_2$. Encoding of components is executed in

such a manner that condition

$$K(b_{gi+1}) = K(b_{gi}) + 1 \qquad (4)$$

takes place for each OLC $\alpha_g \in C$ $(i = 1, ..., F_g - 1)$. If condition

$$R_1 + R_2 = R \qquad (5)$$

takes place, then the model of CMCU with code sharing U_1 can be used for interpretation of GSA Γ (Fig. 1).

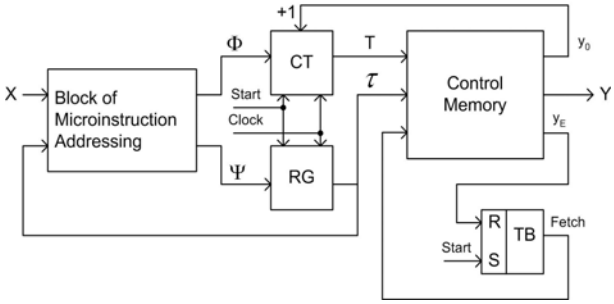

Fig. 1. Structure diagram of CMCU U_1

In CMCU U_1, a block of microinstruction addressing (BMA) implements the system of input memory functions for counter CT and register RG:

$$\Phi = \Phi(\tau, X), \quad \Psi = \Psi(\tau, X). \qquad (6)$$

Let us point out that in the case of CMCU U_1 an address of microinstruction is represented as the following one:

$$A(b_q) = K(\alpha_g) * K(b_q), \qquad (7)$$

where b_q is a component of OLC $\alpha_g \in C$ and "*" is a sign of concatenation. The CMCU U_1 operates in the following order.

If Start=1, then an initial address (all zeros) is loaded into RG and CT. In the same time a flip-flop TF is set up which causes Fetch=1, then microinstructions can be read out of control memory. Each cell of CM keeps microoperations $y_n \in Y$ and special variables y_0 and y_E. If $y_0 = 1$, then a current content of CT is incremented, otherwise both CT and RG are loaded from BMA. The first case corresponds to transition from any OLC component except of its output. The second case corresponds to transition from OLC output. If $y_E = 1$, then flip-flop TF is reset, signal Fetch=0 and operation of CMCU is terminated. It corresponds to transition from the vertex $b_q \in E_1$, where $< b_q, b_E > \in E$. Pulse Clock is used for timing of CMCU.

Let us point out that OLC $\alpha_i, \alpha_j \in C$ are pseudo-equivalent OLC [2] if their outputs are connected with input of the same vertex of GSA Γ. The hardware amount in logic circuit of BMA can be decreased due to introduction of a special block for transforming the OLC codes into the codes of the classes of pseudo-equivalent OLC (POLC) named as a code transformer (TC) [2]. But TC consumes some resources of the chip in use.

In this article we propose to use free cells of CM for such a transformation. It results in decrease of hardware amount in both blocks BMA and TC without increase of the number of PROM chips in CM.

2.Main idea of proposed approach

Let $C_1 \subset C$ be a set of OLC such that their outputs are not connected with the vertex b_E. Let us find the partition $\Pi_C = \{B_1, ..., B_I\}$ of the set C_1 by the classes of POLC. Let condition

$$2^{R_2} > F_g \qquad (8)$$

take place for each OLC $\alpha_g \in C_1$.

Let us encode the classes $B_i \in \Pi_C$ by binary codes $K(B_i)$ with

$$R_3 = \lceil \log_2 I \rceil \qquad (9)$$

bits and let us use elements of the set Z for this encoding, where $|Z| = R_3$. Let us insert an additional component corresponding to control microinstruction MC_g with $y_0 = 0$ and $K(B_i)$, where $\alpha_g \in B_i$. Now all microinstructions MI_q contain $y_0 = 1$.

In this case GSA Γ can be interpreted by CMCU U_2 (Fig. 2) proposed in this article.

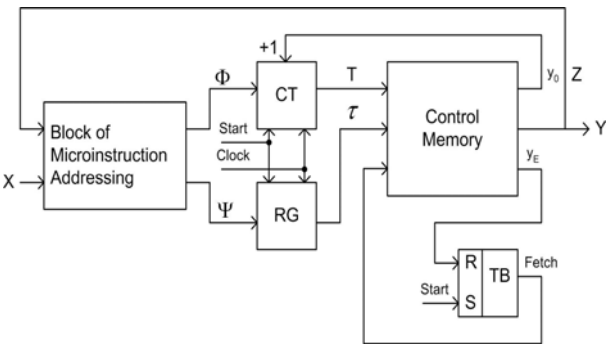

Fig. 2. Structural diagram of CMCU U_2

In CMCU U_2, the block BMA implements functions

$$\Phi = \Phi(Z, X), \qquad (10)$$

$$\Psi = \Psi(Z, X), \qquad (11)$$

all other components of U_2 have the same meaning as their counterparts in U_1.

Functions (10)–(11) are generated if contents of RG and CT represent an address of control microinstruction. In this case a data-path of controlled system is in the idle state. It can be achieved, for example, if data-path synchronization is stirred by variable y_0.

In this article we propose the following method of CMCU U_2 design:

1. Construction of the sets C, C_1, Π_C for GSA Γ.
2. Including of additional components into OLC $\alpha_g \in C_1$.
3. Encoding of OLC, their components and classes.
4. Construction of control memory content.
5. Construction of transition table of CMCU.
6. Implementation of CMCU logic circuit.

3. Example of proposed method application

Let GSA Γ_1 be characterized by sets $C = \{\alpha_1,...,\alpha_7\}$, $\alpha_6 \notin C_1$, $\Pi_C = \{B_1, B_2, B_3\}$, where $B_1 = \{\alpha_1\}$, $B_2 = \{\alpha_2, \alpha_3\}$, $B_3 = \{\alpha_4..\alpha_6\}$, $\alpha_1 =< b_1..b_3 >$, $\alpha_2 =< b_4, b_5 >$, $\alpha_3 =< b_6.. b_8 >$, $\alpha_4 =< b_9.. b_{11} >$, $\alpha_5 =< b_{12}.. b_{14} >$, $\alpha_6 =< b_{15}, b_{16} >$, $\alpha_7 =< b_{17}.. b_{20} >$. It means that G=7, R_1=3, $\tau = \{\tau_1, \tau_2, \tau_3\}$, Q=4, R_2=2, $T = \{T_1, T_2\}$, M=20, R=5 and conditions (5) and (8) take places. Therefore, application of proposed method has sense. Let us point out that R_3=2, $Z = \{z_1, z_2\}$. After introducing of additional components into OLC $\alpha_g \in C_1$ we have:

$$\alpha_1 =< b_1, b_2, b_3, MC_1 >,\ \alpha_2 =< b_4, b_5, MC_2 >,$$
$$\alpha_3 =< b_6, b_7, b_8, MC_3 >,\ \alpha_4 =< b_9, b_{10}, b_{11}, MC_4 >,$$
$$\alpha_5 =< b_{12}, b_{13}, b_{14}, MC_5 >,\ \alpha_6 =< b_{15}, b_{16}, MC_6 >.$$

Let us encode OLC $\alpha_g \in C$ in arbitrary manner, namely: $K(\alpha_1) = 000$, …, $K(\alpha_7) = 110$. Let code 00 be assigned to the first component of any OLC $\alpha_g \in C_1$, code 01 to the second, code 10 to the third, and code 11 to the fourth. Now microinstruction addresses are shown in table 1.

Table 1

Microinstruction addresses for CMCU $U_2(\Gamma_1)$

$\tau_1\tau_2\tau_3$ / T_1T_2	000	001	010	011	100	101	110
00	b_1	b_4	b_6	b_9	b_{12}	b_{15}	b_{17}
01	b_2	b_5	b_7	b_{10}	b_{13}	b_{16}	b_{18}
10	b_3	MC_2	b_8	b_{11}	b_{14}	MC_6	b_{19}
11	MC_1	*	MC_3	MC_4	MC_5	*	b_{20}

The symbol $U_i(\Gamma_j)$ stands for the case when GSA Γ_j is interpreted by CMCU U_i. We can derive from Table 1, for example, $A(b_1)$=00000, $A(b_8)$=01010, $A(MC_2)$=00110.

Let us encode classes $B_i \in \Pi_C$ by the following codes: $K(B_1)$=00, $K(B_2)$=01, $K(B_3)$=10. Let microoperations $y_n \in Y$ are distributed among the operator vertices in the following manner:

$$Y(b_1) = Y(b_5) = \{y_1, y_2\},$$
$$Y(b_2) = Y(b_6) = Y(b_{14}) = \{y_3\},$$
$$Y(b_3) = Y(b_7) = Y(b_{12}) = \{y_1, y_4\},$$
$$Y(b_4) = Y(b_9) = Y(b_{13}) = \{y_2, y_3\},$$
$$Y(b_8) = Y(b_{15}) = Y(b_{17}) = \{y_5\},$$
$$Y(b_{10}) = Y(b_{16}) = Y(b_{19}) = \{y_4\},$$
$$Y(b_{11}) = Y(b_{18}) = \{y_1, y_5\},\ Y(b_{20}) = \{y_6\}.$$

In this case the control memory content for CMCU $U_2(\Gamma_1)$ is shown in table 2.

Table 2

Content of control memory for CMCU $U_2(\Gamma_1)$

$\tau_1\tau_2\tau_3$ / T_1T_2	000	001	010	011	100	101	110
00	y_0 y_1 y_2	y_0 y_2 y_3	y_0 y_3	y_0 y_2 y_3	y_0 y_1 y_4	y_0 y_5	y_0 y_5
01	y_0 y_3	y_0 y_1 y_2	y_0 y_1 y_4	y_0 y_4	y_0 y_2 y_3	y_0 y_4	y_0 y_1 y_5
10	y_0 y_1 y_4	z_2	y_0 y_5	y_0 y_1 y_5	y_0 y_3	z_1	y_0 y_4
11	–	*	z_2	z_1	z_1	*	y_E y_6

Let us point out that transition from table 2 to control memory implementation is a straightforward one.

Let transitions from OLC outputs of GSA Γ_1 be represented by the following system of generalized transition formulae [2]:

$$B_1 \to x_1 b_4 \vee \overline{x_1} x_2 b_6 \vee \overline{x_1}\,\overline{x_2} b_{11};$$
$$B_2 \to x_2 x_3 b_9 \vee x_2 \overline{x_3} b_{13} \vee \overline{x_2} x_4 b_{12} \vee \overline{x_2}\,\overline{x_4} b_{15}; \quad (12)$$
$$B_3 \to x_4 b_{17} \vee \overline{x_4} x_5 b_{11} \vee \overline{x_4}\,\overline{x_5} b_{20}.$$

Such a system is the base to construct the transition table of CMCU U_2 with the following columns: $B_i, K(B_i), b_q, A(b_q), X_h, \Psi_h, \Phi_h, h$. Here X_h is a conjunction of some elements $x_l \in X$, determining the transition from $B_i \in \Pi_C$ into microinstruction MI_q; Ψ_h is a set of input memory functions to form the code

$K(\alpha_g)$ into RG, where $\alpha_g \in B_i$; Φ_h is a set of input memory functions to form the code $K(b_q)$ into CT; h is the number of transition, where $h = 1, ..., H_2(\Gamma_j)$. The subscript 2 underlines that we deal with CMCU U_2. The number $H_2(\Gamma_j)$ is equal to the number of terms in the system of the type (12). In our case, $H_2(\Gamma_1) = 10$ and some part of the table is shown in table 3.

Table 3

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

B_i	$K(B_i)$	b_q	$A(b_q)$	X_h	Ψ_h	Φ_h	h
B_3	10	b_{17}	11000	x_4	D_1 D_2	-	1
		b_{11}	01110	$\overline{x_4}x_5$	D_2 D_3	D_4	2
		b_{20}	11011	$\overline{x_4 x_5}$	D_1 D_2	D_4 D_5	3

The connections between Table 3 and system (12) as well as with Table 1 are obvious. After minimization we can get the parts of system (10) and (11) from nable 3:

$$D_1 = z_1\overline{z_2}x_4 \vee z_1\overline{z_2 x_4 x_5};$$
$$D_2 = z_1\overline{z_2}; \qquad D_3 = z_1\overline{z_2 x_4}x_5; \qquad (13)$$
$$D_4 = z_1\overline{z_2 x_4}; \qquad D_5 = z_1\overline{z_2 x_4 x_5}.$$

Implementation of logic circuit of CMCU $U_2(\Gamma_j)$ is reduced to implementation of systems (10) – (11) using PAL macrocells and implementation of control memory using PROM chips.

In our case Table 2 is used to implement the control memory.

Let us point out that for GSA Γ_1 we have $H_1(\Gamma_1)$=20, and $\eta = H_1(\Gamma_1) / H_2(\Gamma_1) = 2$.

As some experiments show [2], the number of terms in systems (10) – (11) is η times less than its number for system (6).

Obviously, the number of PAL macrocells in logic circuit of block BMA can be found if we know the number of terms per cell. But the ratio of the numbers of macrocells is approximately equal to η [6].

4. Investigation of effectiveness of proposed method

Let us find the range of CMCU U_2 effective using with the help of probabilistic approach from [2]. According to this approach each GSA is characterized by operational vertices percentage p_1.

In the case of linear GSA $p_1 \geq 0,75$. In our researches matrix models of CMCU are used [5] instead of schemes in certain basis. Hardware amount is characterized by used matrix area. Given method effectiveness depends on research of ratio

$$f = \frac{S(U_2)}{S(U_1)}, \qquad (14)$$

where $S(U_i)$ is the CMCU U_i (i=1,2) matrix realization area. CMCU U_1 matrix scheme is represented in Fig. 3.

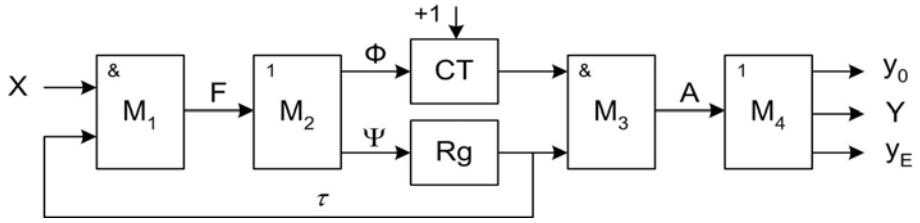

Fig. 3. Compositional microprogram control unit U_1 matrix realization

Some signals (Clock, Start, TB, Fetch) and connections that don't affect on the research results are skipped in Fig. 3. Conjunctive matrix M_1 realizes terms system $F = \{F_1, ..., F_H\}$, that conform to CMCU transition table rows. Disjunctive matrix M_2 realizes functions (6), that depends on terms $F_h \in F$. Conjunctive matrix M_3 realizes variables $A_q \in A$, where A is set of microinstruction addresses. Disjunctive matrix M_2 realizes microoperations $y_n \in Y$ and additional variables y_0, y_E. Areas $S(M_j)_1$ of CMCU U_1 (j=1,…,4) matrixes M_j:

$$S(M_1)_1 = 2(L + R_1)H; \qquad (15)$$

$$S(M_2)_1 = HR; \qquad (16)$$

$$S(M_3)_1 = 2R * 2^R; \qquad (17)$$

$$S(M_4)_1 = 2^R(N + 2). \qquad (18)$$

CMCU U_2 matrix realization is the same as in the case of CMCU U_1.

But feedback systems in these structures of control unit are different. Here areas $S(M_j)_2$ are determined as

$$S(M_1)_2 = 2(L + R_3)H_0; \qquad (19)$$

$$S(M_2)_2 = H_0R; \qquad (20)$$

$$S(M_3)_2 = S(M_3)_1; S(M_4)_2 = S(M_4)_1. \qquad (21)$$

In formulae (19) and (20) variable H_0 is equal to transition table rows number of Mealy FSM, equivalent to Moore FSM that determines BMA.

For decreasing number of parameters in equations (15)–(21) let's use research results [2]. Let K is the number of GSA Γ vertexes. In this case

$$R = \lceil \log_2 (p_1 K) \rceil; \quad (22)$$

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

$$H = 17,4 + 1,7 k_1 p_1 K; \quad (24)$$

$$R_1 = \lceil \log 2 (k_1 p_1 K) \rceil. \quad (25)$$

As follows from (25), $G = k_1 M$, where $k_1 \le 1$ – coefficient that is reciprocally proportional to the average OLC length (number of its components) in GSA Γ.

For determination R_3 it is necessary to find equivalent Mealy FSM states number:

$$I = 2,75 + 0,34 k_1 p_1 K. \quad (26)$$

Formula (26) can be used now in (9). The value of H_0 can be estimated as

$$H_0 = 4,4 + 1,1 * I. \quad (27)$$

To determine effectiveness of CMCU U_2, there is a necessity to investigate a function

$$f = \frac{S(M_1)_2 + ... + S(M_4)_2}{S(M_1)_1 + ... + S(M_4)_1}. \quad (28)$$

This function depends on variables K, p_1, k_1, N. Some of our research results are shown below.

As follows from Fig. 4 and Fig. 5, in case of fulfillment at condition (8) CMCU U_2 is more effective in hardware amount then CMCU U_1.

With the rising up of variable k_1 gain from using CMCU U_2 rise up too. Average gain with p_1=0.75, k_1=0.1 и N=15 is near 47%.

As research results show, using CMCU U_2 structure for GSA with greater value of k_1 gives less effect.

Average gain with p_1=0.75, k_1=0.1 и N=15 is near 33%.

Conclusions

The proposed method of modification of OLC targets on decrease in hardware amount (the number of PAL macrocells) in the block of microinstructions addressing of CMCU with code sharing. This optimization does not affect the number of PROM chips used for implementation of the control memory of CMCU.

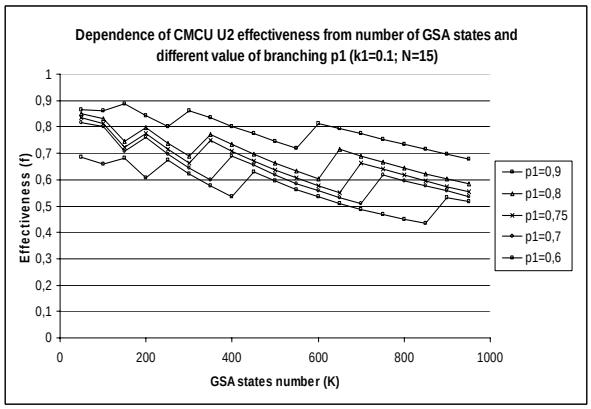

Fig. 4. Dependence of CMCU U_2 effectiveness from number of GSA states and value p_1 (k1=0.1; N=15)

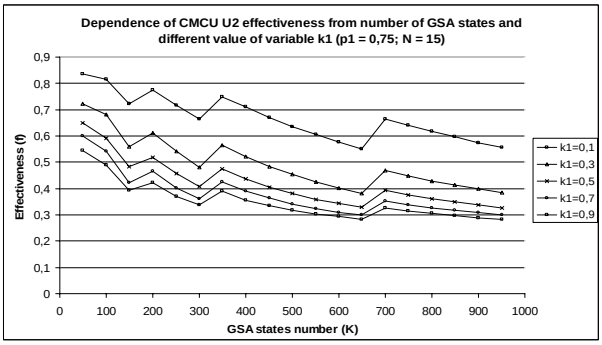

Fig. 5. Dependence of CMCU U_2 effectiveness from number of GSA states and different value of variable k_1 (p_1=0.75; N=15)

The method is based on encoding of the classes of pseudoequivalent OLC permitting decrease of the transition table lines in comparison with equivalent CMCU U_1 without modification of OLCThe drawback of this method is increase in the number of cycles needed for execution of control algorithm in comparison with CMCU U_1. But decrease in the number of macrocells can lead to decrease in the number of layers in combinational part of CMCU.

It results in decrease of the cycle time. Thus, the final conclusion about algorithm execution time should be made after implementation of logic circuits for $U_1(\Gamma_j)$ as well as for $U_2(\Gamma_j)$. Our experiments show that the number of macrocells is decreased up to 30% and the number of layers is decreased up to 3. Of course, application of proposed method is possible only for interpretation of linear GSA when condition (8) takes place.

The next steps in our research are development of CAD tools for CMCU design and exploration of possibility for given method application in case of FPGA [7].

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

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

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

6. Баркалов А.А. Исследование аппаратурных характеристик автомата Мили с кодированием фрагмента микроопераций по VHDL- моделям / А.А Баркалов, Р.М Бабаков, Бадер Ахмад // Исскуственный интеллект. – 2007, №1. - С. 117-122.

7. Maxfield C. The Design Warrior's Guide for FPGA / C. Maxfield. – Amsterdam: Elseteir, 2004. – 541 p.

Поступила в редакцию 12.02.2009

Рецензент: д-р техн. наук, проф., проф. кафедры компьютерных интеллектуальных систем и сетей А.В. Дрозд, Одесский национальный политехнический университет, Одесса, Украина.

СИНТЕЗ МИКРОПРОГРАММНОГО УСТРОЙСТВА УПРАВЛЕНИЯ С РАЗДЕЛЕНИЕМ КОДОВ И МОДИФИКАЦИЕЙ ОПЕРАТОРНЫХ ЛИНЕЙНЫХ ЦЕПЕЙ

А.А. Баркалов, А.А. Красичков, А.Н. Мирошкин

В статье предложен метод синтеза композиционных микропрограммных устройств управления с разделением кодов. Метод нацелен на уменьшение количество макроячеек ПАЛ в комбинационной части управляющего устройства. Для модификации операторных линейных цепей используются дополнительные микрокоманды, содержащие коды классов псевдоэквивалентных операторных цепей. Приведен пример использования предложенного метода. Приведены различные результаты исследований граф-схем алгоритмов в виде диаграмм. Определены наиболее значимые характеристики граф-схем для использования предложенного метода.

Ключевые слова: управляющее устройство, алгоритм, переход, аппаратурные затраты, разделение кодов.

СИНТЕЗ МІКРО ПРОГРАМНОГО ПРИСТРОЯ КЕРУВАННЯ З РОЗДІЛЕННЯМ КОДІВ ТА МОДИФІКАЦІЄЮ ОПЕРАТОРНИХ ЛІНІЙНИХ ЛАНЦЮГІВ

О.О. Баркалов, О.О. Красічков, О.М. Мірошкін

У статті запропонований метод синтезу композиційних мікропрограмних пристроїв керування з розділенням кодів. Метод спрямований на зменшення кількості макроосередків ПАЛ у комбінаційній частині пристрою керування. Для модифікації операторних лінійних ланцюгів використовуються додаткові мікрокоманди, які містять коди класів псевдо еквівалентних операторних ланцюгів. Наведений приклад використання запропонованого методу. Наведені результати досліджень граф-схем алгоритмів у вигляді діаграм. Визначені найбільш вагомі характеристики граф-схем для використання запропонованого методу.

Ключові слова: пристрій керування, алгоритм, перехід, апаратурні витрати, розділення кодів.

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.

Красичков Алексей Александрович – канд. техн. наук, доцент, доцент кафедры электронных вычислительных машин Донецкого национального технического университета, Донецк, Украина, e-mail: krasich@cs.dgtu.donetsk.ua.

Мирошкин Александр Николаевич – аспирант кафедры электронных вычислительных машин Донецкого национального технического университета, Донецк, Украина, e-mail: MiroshkinAN@gmail.com.