Reduction of Hardware Amount for Control Unit with Address Transformer

Alexandr A. Barkalov, Larisa A. Titarenko, Alexandr S. Lavrik

***Abstract* — The method of hardware reduction is proposed oriented on control units and CPLD chips. The method is based on a wide fan-in of PAL macrocells allowing using more than one source of microinstruction address. The method of logical condition replacement is used for optimization of microinstruction addressing block. An example of proposed method application is given.**

***Index Terms*—Address transformer, CMCU, CPLD.**

I. Introduction

Complex programmable logic devices (CPLD) are widely used for implementation of logic circuits of control units [1]. As a rule, CPLD include macrocells of programmable array logic (PAL) [2], [3]. To design a logic circuit with optimal characteristics, some peculiarities of logic elements in use and a control algorithm to be interpreted should be taken into account. If a control algorithm is represented by a linear graph-scheme of algorithm (GSA), thin it can be interpreted using a model of compositional microprogram control unit (CMCU) [4]. One of the distinctive features of CPLD is the wide fan-in of macrocells [5], [6]. It can be used for increasing of the number of sources for classes of pseudoequivalent operational linear chains [7], [8]. The method is proposed in this article based on the abovementioned feature of CPLD, as well as on the replacement of logical conditions [1].

The aim of this research is reduction of the hardware amount in logic circuit of CMCU due to simultaneous use of more than one code source and the replacement of logical conditions. The task of research is the development of design method resulted in the hardware amount decrease for blocks of microinstruction addressing and microinstruction address transformer.

Manuscript received March 7, 2009.
A. A. Barkalov is with University of Zielona Gora, Poland.
e-mail: A.Barkalov@iie.uz.zgora.pl
L. A. Titarenko is with University of Zielona Gora, Poland.
e-mail: L.Titarenko@iie.uz.zgora.pl
A. S. Lavrik is with Donetsk National Technical University, Donetsk, Ukraine. e-mail: alexandrniordlavrik@gmail.com

II. Features of CMCU with Microinstruction 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 $Y=\{y_1,\dots,y_N\}$ is a set of data-path microoperations [1]. Each vertex $b_q\in E_2$ contains a single element of the set of logical conditions $X=\{x_1,\dots,x_L\}$. 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) $C=\{\alpha_1,\dots,\alpha_G\}$ 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 the arbitrary number of inputs. Formal definitions of OLC, its input and output can be found in [4]. Each vertex $b_q\in E_1$ corresponds to microinstruction MIq 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 $T_r\in T=\{T_1,\dots,T_R\}$. 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 $i=1,\dots,F_g-1$.

If outputs O_i,O_j are connected with an input of the same vertex, then OLC $\alpha_i,\alpha_j\in C$ are pseudoequivalent OLC (POLC) [2]. Let us construct the partition $\Pi_C=\{B_1,\dots,B_I\}$ 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 U1 with address transformer (Fig. 1).

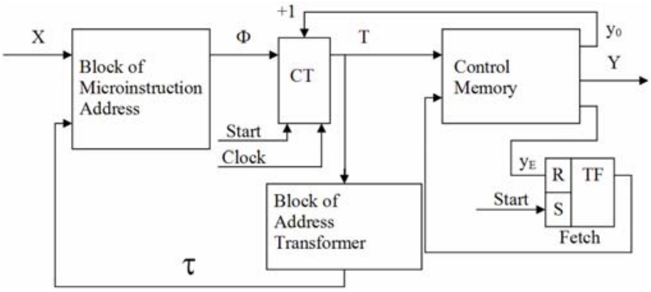

Fig. 1. Structural diagram of CMCU U_1

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 y0 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, a block of microinstruction address BMA generates functions

$$\Phi = \Phi(\tau, X) \quad (4)$$

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

$$\tau = \tau(T). \quad (5)$$

If the output of OLC $\alpha_g \notin C_1$ is reached, then $y_E = 1$. It causes reset of TF and operation of CMCU U1 is terminated.

Such an 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 Moore and Mealy finite state machines (FSM) respectively. But the 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 the 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.

III. 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 $t \in \{1,2,4,8,16\}$. Let us address the components of OLC $\alpha_g \in C_1$ in such a manner that condition (2) takes place and the maximal possible amount of classes $B_i \in \Pi_C$ is represented by a single generalized interval of R-dimensional Boolean space. Such an 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

$$\Pi_B = \emptyset \quad (6)$$

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

$$R_2 = \lceil \log_2(I_B + 1) \rceil \quad (7)$$

bits for such an 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 the hot-one encoding of microoperations [2] when CM word has N+2 bits. In this case the CM can be implemented using

$$R_0 = \left\lceil \frac{N+2}{t} \right\rceil \quad (8)$$

chips with enough amount of cells (not less than M). Obviously, that R3 outputs of PROM are free, where

$$R_3 = R_0 * t - N - 2. \quad (9)$$

If condition

$$R_3 \geq R_2 \quad (10)$$

takes place, then the CM is a source of the codes for $B_i \in \Pi_B$ and the block BAT is absent. This approach permits to decrease the number of PAL macrocells in the logic circuit of block BMA, as well as the number of PROM chips used for the address transformation.

Further optimization of the block BMA logic circuit is possible due to the logical condition replacement [1]. In this case the set X is replaced by some set $P = \{P_1,\dots P_Q\}$, where $Q << L$. The structural diagram of CMCU U_2 based on this principle is shown in Fig. 2.

In CMCU U_2, codes $K_A(B_i)$ of the classes $B_i \in \Pi_A$ are represented by variables $T_r \in T$, whereas codes $K_B(B_i)$ of the classes $B_i \in \Pi_B$ by variables $v_r \in V$, where $|V| = R_2$. In contrast to CMCU U_1, there is no block BAT, and the block BMA implements functions

$$\Phi = \Phi(T, V, P). \quad (11)$$

Variables $p_q \in P$ are generated by a block of logical conditions (BLC) as the following system

$$P = P(T, V, X). \quad (12)$$

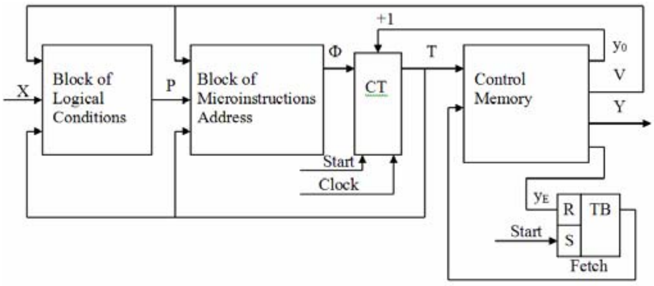

Fig. 2. Structural diagram of CMCU U_2

Let the symbol $U_i(\Gamma_j)$ mean that CMCU U_1 interprets Γ_j, and symbol $Q_i(\Gamma_j)$ determine the number of macrocells in the logic circuit of BMA for CMCU $U_i(\Gamma_j)$, where $i=1,2$. Let each macrocell have S inputs, where G_q variables $p_q \in P$ are connected with inputs of the macrocell number q. The proposed method can be applied if the following condition

$$Q_q + R + R_2 \le S \tag{13}$$

takes place, where $q=1,\dots,Q_1(\Gamma_j)$. If condition (13) is violated, the number $Q_2(\Gamma_j)$ exceeds tremendously the number $Q_1(\Gamma_j)$.

The following method is proposed in our article for synthesis of CMCU U_2:

1. Constructions of sets C, C_1, and Π_C for GSA Γ.
2. Microinstruction addressing.
3. Constructions of sets Π_A and Π_B.
4. Encoding of classes $B_i \in \Pi_B$.
5. Construction of control memory content.
6. Replacement of logical conditions.
7. Construction of CMCU transition table.
8. Specification of block BLC.
7. Implementation of CMCU logic circuit.

IV. Example of Application of Proposed Method

Let the sets $C=\{\alpha_1,\dots,\alpha_9\}$, $C_1=\{\alpha_1,\dots,\alpha_8\}$ and $\Pi_C=\{B_1,\dots,B_5\}$ be formed for a GSA Γ_1, where $\alpha_1=\langle b_1,b_2\rangle$, $\alpha_2=\langle b_3,\dots,b_6\rangle$, $\alpha_3=\langle b_7,b_8\rangle$, $\alpha_4=\langle b_5,\dots,b_{13}\rangle$, $\alpha_5=\langle b_4,\dots,b_{17}\rangle$, $\alpha_6=\langle b_{18},\dots,b_{21}\rangle$, $\alpha_7=\langle b_{22},\dots,b_{25}\rangle$, $\alpha_8=\langle b_{26},\dots,b_{28}\rangle$, $\alpha_9=\langle b_{29},\dots,b_{31}\rangle$, $B_1=\{\alpha_1\}$, $B_2=\{\alpha_2,\alpha_3\}$, $B_3=\{\alpha_4,\alpha_5\}$, $B_4=\{\alpha_6,\alpha_7\}$, $B_5=\{\alpha_8\}$. 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 the algorithm from [4]. Now we have $A(b_1)=00000,\dots$, $A(b_{25})=110000$, $A(b_{26})=11100,\dots$, $A(b_{28})=11110$, $A(b_{29})=11001,\dots$, $A(b_{31})=11011$. Let us construct the Karnaugh map marked by the variables $T_r \in T=\{T_1,\dots,T_5\}$ (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: the class B_1 corresponds to interval 0000*, the class B_2 to 001**, the class B_3 to 01*** and 10000, the class B_4 to 101** and 11000, the class B_5 to 111**. Let us point out that $\alpha_9 \notin C_1$ and the class $B_6=\{\alpha_9\}$ is not considered here.

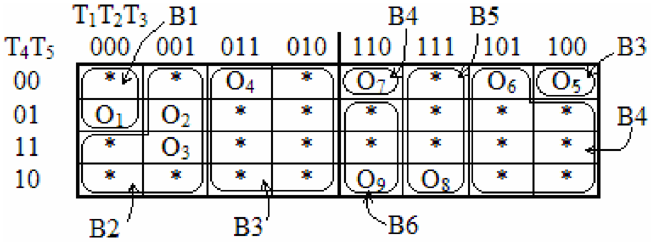

Fig. 3. Karnaugh map for outputs of OLC

The obtained intervals determine the sets $\Pi_A=\{B_1,B_2,B_5\}$ and $\Pi_B=\{B_3,B_4\}$. Let $N=12$ for the GSA Γ_1 and $t=4$. In this case we have $R_3=2$, condition (10) takes place, because $R_2=\lceil \log_2(2+1)\rceil = R_3$. Therefore, the model of CMCU $U_2(\Gamma_1)$ can be applied and the block BAT is absent.

Let us encode the classes $B_i \in \Pi_B$ in the following way: $K_B(B_3)=01$, $K_B(B_4)=10$. Now the code 00 corresponds to the case, when $B_i \in \Pi_A$. The code 11 can be used for optimization of other codes. Finally we get $K_B(B_3)=*1$ and $K_B(B_4)=1*$. Besides, the following codes can be derived from the Karnaugh map shown in Fig. 3: $K_A(B_1)=0000*$, $K_A(B_2)=001**$, $K_A(B_5)=111**$.

The following procedure is proposed for construction of the control memory content, which can be viewed as some modification of the known method [4]:

1. $q=1$.
2. If $b_q \in E_1$, then the memory cell with address $A(b_q)$ contains $Y(b_q)$. Otherwise, go to point 6.
3. If b_q is not the output of OLC $\alpha_g \in C$, then the memory cell with address $A(b_g)$ contains y_o.
4. If b_q is the output of OLC $\alpha_g \notin C_1$, then the memory cell with address $A(b_q)$ contains y_E.
5. If b_q is the output of OLC $\alpha_g \in C_1$, where $a_g \in B_i$, then the memory cell $A(b_q)$ contains code $K_B(B_i)$.
6. If all vertices of GSA Γ are analyzed, then go to step

7. Otherwise, $q := q+1$, go to point 2.

7. End.

There are no problems in this procedure application. So, this step in our article is omitted.

Let the transitions for classes $B_i \in \Pi_C$ be specified by the following system of generalized formulae of transitions (GFT) [4]:

$$
\begin{aligned}
&B_1 \to x_1 b_3 \vee \bar{x}_1 x_4 b_5 \vee \bar{x}_1 \bar{x}_4 b_7;\\
&B_2 \to x_3 b_9 \vee \bar{x}_3 b_{26};\\
&B_3 \to x_1 b_{18} \vee \bar{x}_1 x_2 b_{20} \vee \bar{x}_1 \bar{x}_2 b_{26};\\
&B_4 \to x_5 b_{27} \vee \bar{x}_5 b_5;\\
&B_5 \to x_6 b_{24} \vee \bar{x}_6 x_7 b_{29} \vee \bar{x}_6 \bar{x}_7 b_{19}.
\end{aligned}
\tag{14}
$$

A GFT is a some modification of transition formulae using for finite-state-machines (FSM) [1]. The fact that OLCs replace FSM states is taken into account. Because all transitions are the same for OLC $\alpha_g \in B_i$, then the classes $B_i \in \Pi_C$ are written into GFT. The expression "GFT" underlines the fact of OLC outputs replacement by corresponding classes. Obviously, the transition for class $B_i \notin \Pi_C$ are not considered, because CMCU operation is terminated after generation of the variable y_E.

Let $X(B_i)$ be a set of logical conditions determing transitions from the class $B_i \in \Pi_C$, where $|X(B_i)| = Q_i$. Let $Q = \max(Q_i | B_i \in \Pi_C)$, then logical conditions $x_l \in X$ can be replaced by Q elements of the set P.

Using system (14), the following sets can be obtained: $X(B_1) = \{x_1, x_4\}$, $X(B_2) = \{x_3\}$, $X(B_3) = \{x_1, x_2\}$, $X(B_4) = \{x_5\}$, $X(B_5) = \{x_6, x_7\}$, $Q_1 = Q_3 = Q_5 = 2$, $Q_2 = Q_4 = 1$ и $Q = 2$. Now we have the set $P = \{p_1, p_2\}$. Let us form a table for logical condition replacement having columns $B_i \in \Pi_C$ and rows $p_q \in P$. If logic condition $x_l \in X$ is replaced by variable $p_q \in P$ for class $B_i \in \Pi_C$, then the symbol x_l is written on intersection of the row p_q and column B_i. The replacement is executed in a way minimizing appearance of the same variable x_l in the different rows of the table (Table I).

TABLE I

LOGICAL CONDITION REPLACEMENT FOR CMCU $U_2(\Gamma_1)$

B_i	B_1	B_2	B_3	B_4	B_5
p_1	x_1	x_3	x_1	–	x_6
p_2	x_4	–	x_2	x_5	x_7

Let us transform the system of GFT by replacement of conditions $x_l \in X$ by variables $p_q \in P$ (using Table I in our case):

$$
\begin{aligned}
&B_1 \to p_1 b_3 \vee \bar{p}_1 p_2 b_5 \vee \bar{p}_1 \bar{p}_2 b_7;\\
&B_2 \to p_1 b_9 \vee \bar{p}_1 b_{26};\\
&B_3 \to p_1 b_{18} \vee \bar{p}_1 p_2 b_{20} \vee \bar{p}_1 \bar{p}_2 b_{26};\\
&B_4 \to p_2 b_{27} \vee \bar{p}_2 b_5;\\
&B_5 \to p_1 b_{24} \vee \bar{p}_1 p_2 b_{29} \vee \bar{p}_1 \bar{p}_2 b_{19}.
\end{aligned}
\tag{15}
$$

Such a system is used to construct the CMCU U_2 transition table (Table II) having columns $B_i, K(B_i), b_q, A(b_q), P_h, \Phi_h, h$. The system (15) is used to construct such a table for the CMCU $U_2(\Gamma_1)$. The table includes $H = 13$ lines, it is determined by the number of terms in system (15). In this table there are two columns, $K_A(B_i)$ and $K_B(B_i)$, to represent the codes corresponding to the classes Π_A and Π_B.

TABLE II

TABLE OF TRANSITIONS FOR CMCU $U_2(\Gamma_1)$

B_i	$K_A(B_i)$	$K_B(B_i)$	b_q	$A(b_q)$	P_h	Φ_h	h
B_1	0000*	00	b_3	00010	p_1	D_4	1
			b_5	00100	$\bar{p}_1 p_2$	D_3	2
			b_7	00110	$\bar{p}_1 \bar{p}_2$	$D_3 D_4$	3
B_2	001**	00	b_9	01000	p_1	D_2	4
			b_{26}	11100	$\bar{p}_1$	$D_1 D_2 D_3$	5
B_3	*****	*0	b_{18}	10001	p_1	$D_1 D_5$	6
			b_{20}	10011	$\bar{p}_1 p_2$	$D_1 D_4 D_5$	7
			b_{26}	11100	$\bar{p}_1 \bar{p}_2$	$D_1 D_2 D_3$	8
B_4	*****	1*	b_{27}	11101	p_2	$D_1 D_2 D_3 D_5$	9
			b_5	00100	$\bar{p}_2$	D_3	10
B_5	111**	00	b_{24}	10111	p_1	$D_1 D_3 D_4 D_5$	11
			b_{29}	11001	$\bar{p}_1 p_2$	$D_1 D_2 D_5$	12
			b_{19}	10010	$\bar{p}_1 \bar{p}_2$	$D_1 D_4$	13

The table of transitions is used to derive equations (11), having the following terms:

$$
F_h = \left(\bigwedge_{r=1}^{R} T_r^{l_{rh}} \right) * \left(\bigwedge_{r=1}^{R_2} V_r^{E_{rh}} \right) * P_h. \tag{16}
$$

In (16), the symbol $l_{rh} \in \{0,1,*\}$ stands for the value of bit r of the code $K_A(B_i)$ from the line h of the table, where $T_r^0 = \overline{T}_r, T_r^1 = T_r, T_r^* = 1$ $(r = 1,\ldots,R)$. The symbol $E_{rh} \in \{0,1,*\}$ stands for the value of bit r of the code $K_B(B_i)$ from the line h of the table, where $V_r^0 = \overline{V}_r, V_r^1 = V_r, V_r^* = 1$ $(r = 1,\ldots,R_2)$. In both cases we have $h = 1,\ldots,H$. For example, the following sum-of-the-products (SOP) can be derived from Table II:

$D_1 = F_5 \vee \ldots \vee F_9 \vee F_{11} \vee F_{12} \vee F_{13} =$
$= \overline{T_1}\,\overline{T_2}T_3\overline{V_1}\,\overline{V_2}\,\overline{P_1} \vee V_2 \vee V_1P_2 \vee T_1T_2T_3\overline{V_1}\,\overline{V_2}$ (after minimizing of initial SOP).

The block BLC is specified by its table having the following columns: $B_i, K_A(B_i), K_B(B_i)$, $P_1,\ldots,P_Q, i$ (Table III). The table is constructed using the initial table of logical replacement (see Table I).

TABLE III

SPECIFICATION OF BLOCK BLC FOR CMCU $U_2(\Gamma_1)$

B_i	$K_A(B_i)$	$K_B(B_i)$	p_1	p_2	i
B_1	0000*	00	x_1	x_4	1
B_2	001**	00	x_3	–	2
B_3	*****	*1	x_1	x_2	3
B_4	*****	1*	–	x_5	4
B_5	111**	00	x_6	x_7	5

This table is used to derive system (12). For example, the following equation can be derived from Table III: $P_1 = \overline{T_1T_2T_3T_4V_1V_2}x_1 \vee \overline{T_1T_2}T_3\overline{V_1V_2}x_3 \vee V_2x_1 \vee T_1T_2T_3\overline{V_1V_2}x_6$.

Implementation of CMCU U_2 logic circuit is reduced to the implementation of systems (12)-(13) with PAL macrocells and control memory with PROM chips. There are many effective methods for solution of these tasks [3], because of it we do not discuss this step in our article.

V. CONCLUSION

The proposed method is oriented on hardware amount decrease in the logic circuit of microinstruction address transformer, which is the part of CMCU. This task solution is based on use of more than one source of codes for classes of pseudoequivalent OLC. As a limit, three sources can be used. Optimization of the logic circuit of block of microinstruction addressing is reached due to usage of the know method of logical condition replacement. It allows decrease for the number of required inputs of PAL macrocells. It gives an additional possibility in use of these free inputs for receiving of variables used for OLC classes encoding.

Unfortunately, the gain in hardware is accompanied by decrease of CMCU performance because of the propagation time of additional block BLC. Besides, this block consumes some recourses of the chip. Therefore, the proposed method can be applied if total hardware amount for BMA and BLC is less than hardware amount for BMA of equivalent CMCU U_1.

The scientific novelty of proposed method is determined by simultaneous use the PLA macrocells wide fan-in and logical condition replacement. It leads to hardware amount decrease for blocks BMA and BAT. If condition (10) takes place, then the block BAT is absent. The practical significance of this method is determined in decrease for the number of macrocells in CMCU logic circuit. It allows getting logic circuits with less hardware than for control units known from literature. Our investigation shows that the number of macrocells is decreased up to 10% for CMCU $U_2(\Gamma_j)$ in comparison with equivalent CMCU $U_1(\Gamma_j)$.

There are two directions of our further research. The first, we should develop an algorithm permitting decrease for the number of OLC with outputs addresses to be transformed. The second, we should try this approach for CPLD based on programmable logic arrays [9], as well as on FPGA [10].

REFERENCES

[1] *Baranov S.* Logic Synthesis for Control Automata. Kluwer Academic Publishers, 1994. 312 pp.

[2] *Grushvitskiy R.I., Mursaev A.H., Ugrumov E.P. System's Design by Using Programmable Logic Microchips* – St. Petersburg: BHV, 2002. 608 p. (in Russian).

[3] *Solov'ev V.V.* Digital System Design based on Programmable Logic Integral Schemas – Moscow: Hot Line-TELEKOM, 2001. 636 p. (in Russian).

[4] *Barkalov A., Titarenko L.* Logic Synthesis for Compositional Microprogram Control Units. – Berlin: Springer, 2008. 272 p.

[5] Altera devices overview. http://www.altera.com/products/devices/common/dev-family_overview.html.

[6] Xilinx CPLDs http://www.xilinx.com/products/silicon_solutions/cplds/index.htm.

[7] *Barkalov A.A., Zeleneva I.Y., Lavrik A.S.* Using PLIS peculiarities for control device schema optimization / Scinse Trans. of Donetsk Nationality Technical University. Series «Informatics, Cybernetics and Calculation Techniques». Issue 9 (132) – Donetsk: DonNTU. – 2008. P. 178-182. (in Russian).

[8] Barkalov A.A., Kovalev S.A., Krasichkov A.A., Lavrik A.S. Control Devise Optimization with transformation of microcommands address c / Proc. of IX International Semnar.– Taganrog. Book. 3. 2008. P. 12-20.

[9] CoolRunner CPLD Datasheet. http://www.xilinx.com/support/documentation/coolrunner-ii.htm

[10] *Maxfield C.* The Design Warrior's Guide to FPGAs. – Amsterdam: Elseveir, 2004. – 541 pp.

Alexandr A. Barkalov – Doctor of Science, Professor of the Donetsk National Technical University (Ukraine), Professor of the Uniwersytet Zielonogórski (Poland).
Scientific interests: digital control units, SoPC
Address: Campus A, Budynek Dydaktyczny / A-2
prof. Z. Szafrana str. 2, 65-516 Zielona Góra
E-mail: A.Barkalov@iie.uz.zgora.pl

Larisa A. Titarenko – Doctor of Science, Professor of the Kharkov National University of Radioelectronics (Ukraine), Professor of the Uniwersytet Zielonogórski (Poland). Scientific interests: digital control units, telecommunication systems. Address: Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Góra
E-mail: L.Titarenko@iie.uz.zgora.pl

Alexandr S. Lavrik – Post graduate Student of the Donetsk National Technical University.