Alexander BARKALOV[1], Larysa TITARENKO[1], Łukasz SMOLIŃSKI[1]

University of Zielona Góra(1)

CMCU model with base structure dedicated for CPLD systems

Abstract. *The method of hardware reduction presented in this work is intended for the compositional microprogram control unit (CMCU) implemented in the complex programmable logic device (CPLD). This method is based on applying more than one data source in generating the CMCU states and the microinstruction address.*

Streszczenie. *W artykule przedstawiona została metoda zmniejszenia powierzchni sterowników sprzętowych realizowanych w układach typu CPLD. Metoda bazuje na wykorzystaniu więcej niż jednego źródła danych przy generowaniu stanu układu oraz adresu mikroinstrukcji.* ***(Model CMCU o bazowej strukturze dedykowany dla układów CPLD)***

Keywords: CPLD, CMCU, UFM, PAL
Słowa kluczowe: CPLD, CMCU, UFM, PAL

doi:10.12915/pe.2014.12.06

Introduction

Control units are crucial parts of digital systems [1, 2, 3, 4, 5, 6, 7, 8]. Nowadays CPLD are widely used for implementing logic circuits of control units [9, 10, 11, 12]. They include macrocells of programmable array logic (PAL) having the limited number of terms. The problem of reducing the size of a control unit is still a subject of current interest [10, 11, 13, 14]. To optimize the amount of hardware in the logic circuit of a control unit, the peculiarities of CPLD and the features of a control algorithm to be implemented should be taken into account. If a control algorithm is represented by a linear graph-scheme of algorithm (GSA), then a model of the compositional microprogram control unit (CMCU) can be used for its interpretation [15]. It is assumed that the control memory (CM) of the CMCU can be implemented as external PROM/ROM memory. Some of the CPLD family devices are equipped with integrated memory. Altera CPLD devices are equipped with user flesh memory (UFM) [16]. Cypress CPLD devices are equipped with cluster memory blocks (CMB) [17]. This article presents a method of hardware reduction based on two sources of code. The main idea presented in [18, 19] has been adapted to CMCU model with a base structure [15], bringing as a result the new mathematical model and design algorithm. The article presents the implementation results for proposed model.

Background for CMCU with base structure

It is assumed that the GSA is represented by sets of vertices **B**, where

$$B = B_1 \cup B_2 \cup \{b_0, b_E\} \tag{1}$$

and a set of arcs **E**, where

$$E = \{\langle b_t, b_q \rangle\}. \tag{2}$$

It is further assumed that

$$b_t, b_q \in B \tag{3}$$

and $\mathbf{b_0}$ is an initial vertex, $\mathbf{b_E}$ is a final vertex, $\mathbf{B_1}$ is a set of operator vertices, $\mathbf{B_2}$ is a set of conditional vertices. A vertex contains a microinstruction

$$m_i \in M(i = 1, \ldots, |B_1|) \tag{4}$$

and $\mathbf{m_i}$ is a set of data-path microoperations [20, 21, 22]:

$$Y = \{y_1, \ldots y_N\}. \tag{5}$$

Each vertex $b_t \in B_2$ contains a single element $\mathbf{x_i}$ from a set of logical conditions

$$X = \{x_1, \ldots, x_L\}. \tag{6}$$

A set **C** of operational linear chains (OLC) for the GSA shall be formed, where each OLC is a sequence of operator vertices and each pair of its adjacent components corresponds to an arc of the GSA:

$$C = \{\alpha_1, \ldots, \alpha_i\}. \tag{7}$$

Each OLC α_i has only one output $\mathbf{O_i}$ and arbitrary number of inputs $\mathbf{I_i^j}$. Formal definitions of OLC, its input and output can be found in [15]. Each vertex from $b_t \in B_1$ corresponds to microinstruction $m_i \in M$ kept in a control memory (CM) of CMCU and it has an address $A(b_t)$. The microinstructions can be addressed using **R** bits, where

$$R = \lceil \log_2 |M| \rceil, |M| = |B_1|, \tag{8}$$

and the bits are represented by variables from the set **T**:

$$T = \{T_1, \ldots, T_R\}. \tag{9}$$

Assuming that the OLC $\alpha_g \in C$ include $\mathbf{F_g}$ components, the following condition takes place:

$$A(b_{g_{i+1}}) = A(b_{g_i}) + 1(i = 1, \ldots, F_g - 1). \tag{10}$$

In equation 10 variable $\mathbf{b_{g_i}}$ is the i-th component of OLC $\alpha_g \in C$. The current state of the CMCU is held in the register RG [23]. The value of used registers needed to implement the RG is:

$$R_1 = \lceil \log_2 M_1 \rceil, M_1 = |C|. \tag{11}$$

The registers outputs are represented by variables from the set τ:

$$\tau = \{\tau_1, \ldots, \tau_{R_1}\}. \tag{12}$$

The combinational circuit (CC) generates data Φ for the counter CT and data Ψ for the register RG:

$$\Psi = \Psi(X, \tau), \tag{13}$$

$$\Phi = \Phi(X, \tau). \tag{14}$$

It is assumed that the OLC outputs $O_i \in \Pi_C$ are encoded by binary codes $K_C(O_i)$ using variables from the set τ. The excitation functions for the counter CT and register RG in equations 13 and 14 are built with the use of logical terms. The terms are created by joining two parts: the set of conjuctions

$x_i \subseteq X$ and the code $K_C(O_i)$. The synthesis of the CMCU with a base structure includes the following steps [23]:

1. Formation of the set of OLCs.
2. Formation of the control memory content.
3. Formation of the transition table of the CMCU.
4. Formation of the excitation functions for the counter
5. Formation of the excitation functions for the register
6. Synthesis of the logic circuit of CMCU.

On figure 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 $\mathbf{y_0}$ is formed together with microoperations $Y_q \subseteq Y$. If $y_0 = 0$, then content of the CT is incremented according to the addressing mode. Otherwise, block of CC generates functions Φ and Ψ. If $\mathbf{y_E}$ equals 1, then CMCU stops and new data from CM will be not loaded.

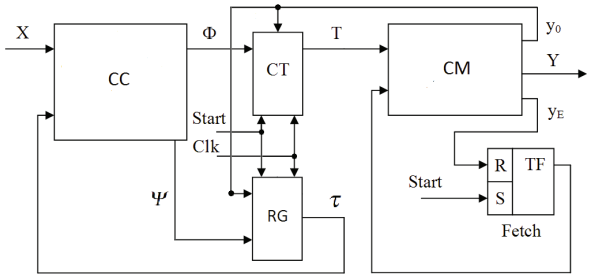

Fig. 1. Structural diagram of CMCU U_1

Adaptation the approach of the two sources of code to CMCU model with base structure

This chapter presents a method of hardware reduction based on two sources of code. The main idea presented in [18, 19] has been adapted to CMCU model with a base structure [15], bringing as a result the following new mathematical model and design algorithm.

It shall be pointed out that the logic for the CC, CT, RG is implemented as parts of the CPLD. An external PROM chip or memory integrated with the CPLD may be applied to implement the CM [24]. The memory has t outputs, where $t = 2, 4, 8, 16, 32$ [24]. Some information can be implemented using free outputs of the CM. it is assumed that one-hot encoding of microoperations is used [25, 26, 27, 28]. The word of CM has

$$R_0 = N + 2 \tag{15}$$

bits. The R_2 outputs of the CM are free.

$$R_2 = t - R_0. \tag{16}$$

The R_2 bits are represented by variables from the collection **P**:

$$P = \{p_1, \ldots, p_{R_2}\}. \tag{17}$$

If condition

$$R_2 > 0 \tag{18}$$

takes place, the method can be used. The number of occurrence $O_i \in \Pi_C$ in excitation functions from equations 13 and 14 shall be counted. R_2 most used O_i items in excitation functions shall be taken. Next, these items shall be moved to a new collection $O_i \in \Pi_B$. Following the above assumptions, an equation

$$R_2 \geq |\Pi_B| \tag{19}$$

takes place.

It shall be highlighted that the collection Π_C has been divided into two collections Π_A and Π_B, where $\Pi_C = \Pi_A \cup \Pi_B$. The control memory CM is a source of the variables $p_i \in P$ applied to encode $O_i \in \Pi_B$ by code $K_B(O_i)$. The register RG is a source of the variables $\tau = \{\tau_1, \ldots, \tau_{R_1}\}$ applied to encode $O_i \in \Pi_A$ by code $K_A(O_i)$.

Figure 2 depicts a modified structural diagram of the CMCU. Provided that the assumption presented in equation 19 is fulfilled, it is possible to apply the one-hot encoding to the elements from the collection $K_B(O_i)$. The excitation functions generating data for the counter CT and register RG in equations 23 and 24 are built with the use of logical terms. The terms are created by joining two parts: the conjunction of conditions $x_i \subseteq X$ and the code $K_B(O_i)$ or $K_A(O_i)$. The length of the these terms has a direct influence on the number of logical gates used in implementing the CMCU model, and the number of the gates used has, in turn, a direct influence on the size of the realized system. The next chapter presents an example of how to build terms for an excitation function in the CMCU with base structure model. In the case where the one-hot encoding is applied to the elements from the collection Π_B, the length of the terms connected in excitation functions is reduced. It shall be highlighted that the maximum length of a term built in the base method with the use of the information from Π_C equals:

$$L_C = |X| + R_1. \tag{20}$$

For the proposed model the maximum length of a term built with the use of the information from the collection Π_A does not change and equals:

$$L_A = |X| + R_1. \tag{21}$$

The maximum length of a term built with the use of information from the collection Π_B is reduced and equals:

$$L_B = |X| + 1. \tag{22}$$

The CC unit generates the functions:

$$\Psi = \Psi(P, X, \tau), \tag{23}$$

$$\Phi = \Phi(P, X, \tau). \tag{24}$$

The application of unused memory areas as a source of information about the state of the system and the presentation of this information with the use of a smaller set of variables makes it possible to reduce the size of the designed system.

An example to presented model

The algorithm to realization is shown in the flow-charts Γ_1 and Γ_2.

We have following sets: $B_1 = \{b_1, \ldots, b_{11}\}, B_2 = \{x_1, x_2, x_3\}, C = \{\alpha_1, \ldots, \alpha_4\}, |M| = 11, R = 4, \alpha_1 = \langle b_1, b_2\rangle, \alpha_2 = \langle b_3, \ldots, b_7\rangle, \alpha_3 = \langle b_8, b_9\rangle, \alpha_4 = \langle b_{10}, b_{11}\rangle$, $\Pi_C = \{O_4, O_3, O_2, O_1\}$, $Y = \{y_1, \ldots, y_5\}$, $N = 5$, $t = 8$. First, the OLC items (tab. 1) and addresses of microinstruction are encoded (tab. 2). Next, a transition table is created (tab. 3). In base method calculation is as follows: Based on the current state from the transition table and sets B_2, the excitation functions generating data for the CT and

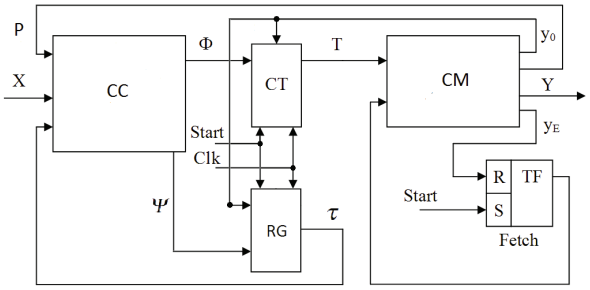

Fig. 2. Structural diagram of CMCU U_2

Table 1. OLC items

OLC outputs O_i	$K_C(O_i)$ τ_2, τ_1
O_1	00
O_2	01
O_3	10
O_4	11

RG are formed:

$$\tau_2{}' = \overline{\tau_2} * \overline{\tau_1} * x_2 * \overline{x_1} + \overline{\tau_2} * \overline{\tau_1} * \overline{x_2} * \overline{x_1} = \overline{\tau_2} * \overline{\tau_1} * \overline{x_1},$$

$$\tau_1{}' = \overline{\tau_2}*\overline{\tau_1}*x_1+\overline{\tau_2}*\overline{\tau_1}*\overline{x_2}*\overline{x_1}+\tau_2*\overline{\tau_1}*x_3+*\tau_2*\tau_1*x_3 = = \overline{\tau_2} * \overline{\tau_1} * \overline{x_2} + \tau_2 * x_3 + \overline{\tau_2} * \overline{\tau_1} * x_1,$$

$$T_4 = \overline{\tau_2} * \overline{\tau_1} * \overline{x_2} * \overline{x_1},$$

$$T_3 = \overline{\tau_2} * \overline{\tau_1} * x_2 * \overline{x_1} + \tau_2 * \overline{\tau_1} * x_3 + \tau_2 * \tau_1 * x_3 = = \overline{\tau_2} * \overline{\tau_1} * x_2 * \overline{x_1} + *\tau_2 * x_3,$$

$$T_2 = \overline{\tau_2} * \overline{\tau_1} * x_1 + \overline{\tau_2} * \overline{\tau_1} * x_2 * \overline{x_1},$$

$$T_1 = \overline{\tau_2}*\overline{\tau_1}*x_2*\overline{x_1}+\overline{\tau_2}*\overline{\tau_1}*\overline{x_2}*\overline{x_1}+\tau_2*\overline{\tau_1}*x_3+\tau_2*\tau_1*x_3 = = \overline{\tau_2} * \overline{\tau_1} * \overline{x_1} + \tau_2 * x_3.$$

According to presented modification calculation is as follows: $R_0 = N + 2 = 7$ and $R_2 = t - R_0 = 1$. R_2 is a value of free CM outputs and shall be represented by variables $P = \{p_1\}$. Next, the collection Π_C shall be divided into two collections Π_A and Π_B. According to formula 19, it is possible to determine the size of collection Π_B. R_2 the most commonly used outputs (tab 4) from Π_C shall be taken out and moved to Π_B, placing the other elements in Π_A. As a result: $\Pi_B = \{O_1\}$ and $\Pi_A = \{O_4, O_3, O_2\}$. Next, the elements from Π_B shall be encoded using the variables $p_i \in P$. Modified OLC encoding is shown on table 5. Modified transition table is shown on table 6. The new encoding is $K_B(0_1) = p_1$. The second source of code is marked with a boldface and an underlining in table 2.

Finally, excitation functions shall be prepared using a new code for element $\Pi_B = \{O_1\}$:

$$\tau_2{}' = p_1 * \overline{x_1},$$

$$\tau_1{}' = p_1 * \overline{x_2} + \tau_2 * x_3 + p_1 * x_1,$$

$$T_4 = p_1 * \overline{x_2} * \overline{x_1},$$

$$T_3 = p_1 * x_2 * \overline{x_1} + *\tau_2 * x_3,$$

Table 2. Control memory content

Vertex b_i	Address $A(b_t)$ T_4,T_3,T_2,T_1	Microinstruction $y_0,y_1,y_2,y_3,y_4,y_5,y_E,p_1$	Comment
b_1	0000	0 1 0 0 0 1 0 0	I_1^1
b_2	0001	1 0 0 1 1 0 0 **1**	O_1
b_3	0010	0 0 1 1 0 0 0 0	I_2^1
b_4	0011	0 1 0 0 1 0 0 0	–
b_5	0100	0 0 0 0 1 0 0 0	–
b_6	0101	0 1 0 1 0 0 0 0	I_2^2
b_7	0110	1 0 1 1 1 0 1 0	O_2
b_8	0111	0 1 1 0 0 0 0 0	I_3^1
b_9	1000	1 1 0 1 0 0 0 0	O_3
b_{10}	1001	0 0 0 1 1 0 0 0	I_4^1
b_{11}	1010	1 1 0 1 0 0 0 0	O_4

Table 3. Transition table of CMCU

Actual state $K_C(O_i)$ τ_2,τ_1	Next state $\tau_2{}',\tau_1{}'$	Conditions B_2	Address $A(b_t)$ T_4,T_3,T_2,T_1
00	01	x_1	0010
00	10	$x_2, \overline{x_1}$	0111
00	11	$\overline{x_2}, \overline{x_1}$	1001
10	00	$\overline{x_3}$	0000
10	01	x_3	0101
11	00	$\overline{x_3}$	0000
11	01	x_3	0101

Table 4. Occurrence $O_i \in \Pi_C$ in forming excitation functions

OLC outputs O_i	Occurrence
O_1	10
O_3	3
O_4	3
O_2	0

Table 5. Modified OLC items

OLC outputs O_i	$K_A(O_i), K_B(O_i)$ τ_2,τ_1,p_1
O_1	--1
O_2	01-
O_3	10-
O_4	11-

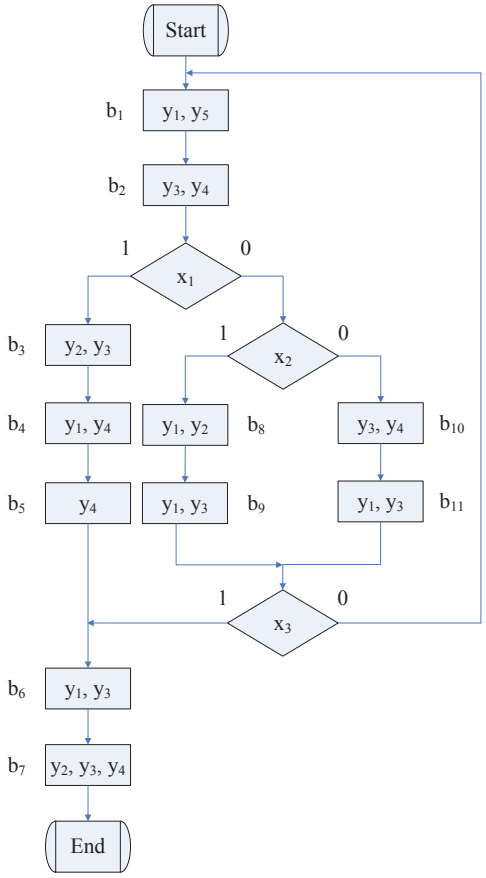

Fig. 3. Flow-chart Γ_1

$$T_2 = p_1 * x_1 + p_1 * x_2 * \overline{x_1},$$

$$T_1 = p_1 * \overline{x_1} + \tau_2 * x_3.$$

It shall be stressed that the size of the block CC is reduced due to the application of information from the collection Π_B and size of block CM in real hardware does not change. For example an excitation function τ_1 is reduced from seven to five logic gates.

Conclusions and results

Figure 5 presents the implementation results in real hardware. When the number of free bits in figure 5 equals 0, it means that the implementation takes place without the application of the presented method. Altera family MAX II de-

Table 6. Modified transition table of CMCU

Actual state	Next state	Conditions	Address
$K_A(O_i), K_B(O_i)$			$A(b_t)$
τ_2, τ_1, p_1	τ_2', τ_1'	B_2	T_4, T_3, T_2, T_1
--1	01	x_1	0010
--1	10	$x_2, \overline{x_1}$	0111
--1	11	$\overline{x_2}, \overline{x_1}$	1001
10-	00	$\overline{x_3}$	0000
10-	01	x_3	0101
11-	00	$\overline{x_3}$	0000
11-	01	x_3	0101

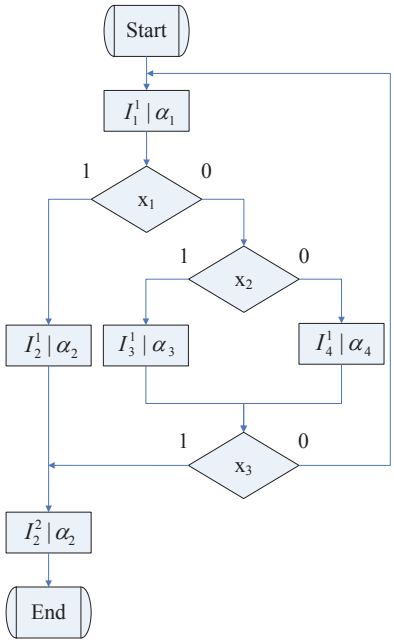

Fig. 4. OLC flow-chart Γ_2

vice EPM1270 F256C5 equipped with UFM has been used for tests. The analysis shows that the number of gates applied to implement the excitation functions can be lowered using more then one source of code. The main factor allowing for this reduction is the natural redundancy of PROM/ROM/UFM/CMB, since their numbers of outputs belong to a limited set. The application of unused memory areas as a source of information about the state of the system and the presentation of this information with the use of a smaller set of variables makes it possible to reduce the size of the designed system. The above-presented design algorithm makes it possible to implement the Boolean functions through free PROM/ROM/UFM /CMB resources. The algorithm is compatible with the CPLD systems equipped with integrated memory resources. Such a system is, for example, the Delta 39K Cypress family or MAX II Altera family.

When the CPLD does not contain dedicated elements, the implementation part serving as "read only" should be moved beyond the CPLD. This part of the system can be implemented with the use of very low-cost external PROM chips.

REFERENCES

[1] V. Salauyou, T. Grześ: FSM State Assignment Methods for Low-Power Design, IEEE Computer Society, 2007.
[2] T. Łuba: Synthesis of Logic Devices, Warsaw University of Technology Press (in Polish), 2004.
[3] S. Baranov: Logic Synthesis for Control Automata, Kluwer Academic Publishers, 1994.
[4] G. DeMicheli: Synthesis and Optimization of Digital Circuits, McGraw-Hill, 1994.
[5] A. Milik, E. Hrynkiewicz: Synthesis and implementation of reconfigurable PLC on FPGA platform, International Journal of Electronics and Telecommunications, 2012.
[6] A. Barkalov, M. Węgrzyn: Design of control units with programmable logic, University of Zielona Góra Press, 2006.
[7] A. Klimowicz, V. Salauyou: The synthesis of combined mealy and moore machines structural model using values of output variables as codes of states, Digital System Design, 2012.
[8] E. Monmasson, L. Idkhajine, M.N. Cirstea: FPGAs in Industrial Control Applications, Industrial Informatics, IEEE Transactions on, 2011.
[9] D. Kania: The Logic Synthesis for the PAL-based Complex

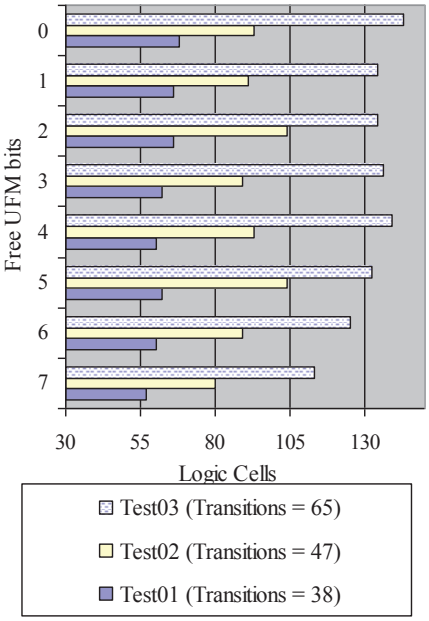

Fig. 5. Reduction results of used CPLD logic cells

Programmable Logic Devices, Lecture Notes of the Silesian University of Technology (in Polish), 2004.

[10] R. Czerwinski, D. Kania: Area and Speed Oriented Synthesis of FSMs for PAL-based CPLDs, Microprocess. Microsyst., 2012.

[11] M. Kubica, D. Kania: Synteza logiczna zespołu funkcji ukierunkowana na minimalizację liczby wykorzystywanych bloków logicznych PAL w oparciu o zmodyfikowany graf wyjść, Pomiary Automatyka Kontrola, 2011.

[12] Altera: Max II device handbook. [web page] http://www.altera.com/literature/hb/max2/max2_mii5v1.pdf. [Accessed on 5 May. 2014.].

[13] R. Czerwinski, D. Kania: Synthesis of finite state machines for CPLDs, International Journal of Applied Mathematics and Computer Science, 2009.

[14] B. Prem Anand, C.G. Saravanan: Development of research engine control unit using fpga - based embedded control system, Journal of KONES Powertrain and Transport, 2012.

[15] A. Barkalov, L. Titarenko: Logic Synthesis for Compositional Microprogram Control Units, Springer, 2008.

[16] Altera: Using the UFM in MAX II devices. [web page] www.altera.com/literature/an/an489.pdf. [Accessed on 5 May. 2014.].

[17] Cypress: Delta39K ISR CPLD Family. [web page] http://pdf.datasheetcatalog.com/datasheet2/9/0pfaeyx4ushkk0zzjksaycgxhqky.pdf. [Accessed on 5 May. 2014.].

[18] A. Barkalov, L. Titarenko, L. Smoliński: Optimization of control unit based on construction of CPLD, Pomiary, Automatyka, Kontrola, 2012.

[19] A. Barkalov, L. Titarenko, L. Smoliński: Optimization of microprogram control unit with code sharing, Proceedings of IEEE East-West Design and Test Symposium - EWDTS, 2011.

[20] S. Baranov: Logic and system design of digital systems, TUT Press, 2008.

[21] M. Adamski, A. Barkalov: Architectural and Sequential Synthesis of Digital Devices, University of Zielona Góra Press, 2006.

[22] M. Wiśniewska, R. Wiśniewski, M. Adamski: Reduction of the microinstruction lenght in the designing process of microprogrammed controllers, Przegląd Elektroniczny, 2009.

[23] R. Wiśniewski: Synthesis of compositional microprogram control units for programmable devices, University of Zielona Gora Press, 2009.

[24] C. Maxfield: The Design Warrior's Guide to FPGAs, Elseveir, 2004.

[25] A. Barkalov, L. Titarenko: Logic Synthesis for FSM - based Control Units, Springer, 2009.

[26] M. Kołopieńczyk: Application of adress converter for decreasing memory size of CMCU with code sharing, Lecture Notes in Control and Computer Science, 2008.

[27] J. Bieganowski: Synthesis of microprogram control units oriented toward decreasing the number of macrocells of addressing circuit, Lecture Notes in Control and Computer Science, 2011.

[28] V. Salauyou, A. Klimowicz: Synteza logiczna układów cyfrowych w strukturach programowalnych, Oficyna Wydawnicza Politechniki Białostockiej, 2010.

***Authors**:*

***Prof. Alexander Barkalov**: born in Russia in 1954. Since 2003 he has been a professor of computer engineering at the Institute of Computer Engineering and Electronics, University of Zielona Góra, Poland, and he is also a professor at the Institute of Computers, Donetsk National Technical University, Ukraine. His current research interests include the theory of digital automata, especially the methods for synthesis and optimization of control units implemented with field programmable logic devices.*
Email: A.Barkalov@iie.uz.zgora.pl

***Prof. Larysa Titarenko**: born in Ukraine in 1971. Since 2007 she has been a professor of telecommunications at the Institute of Computer Engineering and Electronics, University of Zielona Góra, Poland, and she is also a professor at the Institute of Telecommunications Systems, Kharkov National University of Radioelectronics, Ukraine. Her current research interests include telecommunication systems, antennas and digital automata, as well as their applications.*
Email:L.Titarenko@iie.uz.zgora.pl

***MA Łukasz Smoliński**: born in Poland in 1984. In 2009 obtained a master's degree in computer science. Currently a fourth year student in the doctoral program at the University of Zielona Gora.*
Email: L.Smolinski@weit.uz.zgora.pl

Mr. Łukasz Smoliński is a scholar within Sub-measure 8.2.2 Regional Innovation Strategies, Measure 8.2 Transfer of knowledge, Priority VIII Regional human resources for the economy Human Capital Operational Programme co-financed by European Social Fund and state budget.