Synthesis of Moore FSM with Expanded Coding Space

Olena Hebda, Larysa Titarenko
Institute of Computer Engineering and Electronics
University of Zielona Góra
Zielona Góra, Poland
Email: O.Shapoval {at} weit.uz.zgora.pl

Abstract— **The proposed method is targeted on reduction of hardware amount in logic circuit of Moore finite-state machine implemented with programmable logic arrays (PLA). The method is based on using more than minimal amount of variables in codes of FSM internal states. The method includes two stages of state encoding. The second stage is connected with recoding of states inside each class of pseudoequivalent states. An example is given for proposed method application.**

Keywords- Moore FSM, graph-scheme of algorithm, pseudoequivalent states, PLA, logic circuit

I. INTRODUCTION

A control unit is a very important block of any digital system. The model of Moore finite state machine (FSM) [8] is often used during the digital control systems realization [1, 5]. One of the important problems of FSM synthesis is the decrease of chip space occupied by FSM logic circuit. Solution of this problem allows decreasing the power consumption and increasing the clock rate. Moreover it could help the developing industry, for example in Polish Lubuskie Province, because in this case the manufacturers will be able to produce much more competitive products.

The methods of solution of this problem depend strongly on logic elements used for implementing the FSM logic circuit [12, 18]. In this article we discuss the case when programmable logic arrays (PLA) are used for implementing Moore FSM logic circuit.

The PLAs were introduced in 1970s, they were a very popular base for the logic design [2]. As it is mentioned in [3], successful computation structures return back at the next round of the technological spiral. Now the return of PLA basis can be observed in the hybrid FPGAs [17], as well as in CoolRunner CPLD by Xilinx [22]. Also, the PLAs are very popular in the modern sublitographic technology [11]. In the nanoelectronics, these devices are named nano-PLAs. Nowadays, extensive research is conducted in the fields connected with nano-PLAs [3, 16]. In the case of application-specified integrated circuits (ASIC) [20], the combinational logic is very often implemented using so called matrix structures where the principle of distributed logic is used [14]. Two matrix structures combined together form a PLA [1]. So, the PLA basis is still popular in the logic design. It means that new design methods should be developed for area reduction of PLA-based control units. In this article, a new design method is proposed targeting the area reduction of PLA-based Moore FSM's logic circuit.

In this article a control algorithm to be implemented is represented by a graph-scheme of algorithm (GSA) [1]. The proposed method can be viewed as a development of the method [6].

II. PRELIMINARIES AND RELATED WORK

Let Moore FSM be represented by the structure table (ST) with columns [1]: a_m, $K(a_m)$, a_s, $K(a_s)$, X_h, Φ_h, h. Here a_m is the initial state of FSM; $K(a_m)$ is the code of state $a_m \in A$ have capacity $R = \lceil log_2 M \rceil$, to code the states the internal variables from the set $T = \{T_1,\ldots,T_R\}$ are used; a_s, $K(a_s)$ are the state of transition and its code respectively; X_h is the input, which determines the transition $\langle a_m, a_s \rangle$, and it is equal to conjunction of some elements (or their complements) of a set of input variables $X = \{x_1,\ldots,x_L\}$; Φ_h is the set of input memory functions for flip-flops of FSM memory, which are equal to 1 to change the content of the memory from $K(a_m)$ to $K(a_s)$, $\Phi_h \subseteq \Phi = \{\varphi_1,..,\varphi_R\}$; $h = \overline{1,H_1}$ is the number of transition. In the column a_m a collection of output variables Y_q is written. These output variables are generated in the state $a_m \in A$ ($Y_q \subseteq Y = \{y_1,\ldots,y_N\}$, $q = 1,\ldots Q$). This table is used to form the following system of functions

$$\Phi = \Phi(T, X), \tag{1}$$

$$Y = Y(T). \tag{2}$$

The systems (1) – (2) determine an FSM logical circuit. In the case of PLA-based Moore FSM, the logic circuit includes

two PLAs: a Core PLA (CPLA) and an Output PLA (OPLA).In this model, CPLA implements the system (1), a register RG keeps state codes, OPLA implements the system (2).

The problem of FSM synthesis with PLAs was thoroughly investigated in 70s [19]. Two main directions of design were discussed. The first of them was the minimization of a number of standard PLA chips implementing an FSM circuit [4]. The second group of methods targeted the minimization of the CMOS VLSI area occupied by an FSM logic circuit [2]. To solve the first problem, the algorithms of state assignment were used [9, 21], as well as different methods of decomposition [13]. To solve the second problem, the encoding of input and output variables were used [2]. Nowadays, the last group of methods is used in optimizing the nano-PLA designs [3]. Their approach leads to the Six Matrix (SM) implementation of Mealy FSM discussed in [3]. The same approach can be used for the case of Moore FSM.

The systems (1) – (2) can be implemented using only two matrices. The first of them (AND-matrix) implements the product terms of the systems. The second matrix (OR-matrix) implements the functions. But this solution leads to the FSM circuits with the highest possible area. The six matrix solution [3] leads to the rather slow FSM circuits. So, in this article we discuss the Four Matrix (FM) implementation of a Moore FSM. This model is shown in Fig. 1. It can be treated as a compromise between the economical but slow SM model and the fast but redundant two matrices model. Let us denote the FM model of Moore FSM as an FSM U_1.

Let us analyze the components of matrix circuit shown in Fig. 1. A conjunctive matrix MT_1 implements the system of terms $F=\{F_1,\ldots,F_{H_1}\}$; a disjunctive matrix MT_2 implements system (1); a conjunctive matrix MT_3 implements terms A_m $m=\overline{1,M}$, corresponding to FSM states; a disjunctive matrix MT_4 implements system (2). The register RG keeps state codes; it is controlled by pulses Start (clearing) and Clock (changing content depending on functions Φ). The matrices MT_1 and MT_2 determine the CPLA, whereas the matrices MT_3 and MT_4 determine the OPLA.

Unfortunately, as a rule, Moore FSM has more states then an equivalent Mealy FSM [1]. It leads to the fact that the number H_1 is much more than the number of rows H_0 of ST for an equivalent Mealy FSM. Let M_0 be the number of states of equivalent Mealy FSM and R_0 is defined as

$$R_0=\lceil log_2 M_0 \rceil. \tag{3}$$

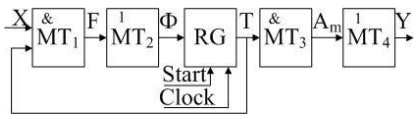

Figure 1. Matrix circuit of Moore FSM U_1

Then the following relations are true for practical cases:

$$H_0<H_1;\ R_0=\lceil log_2 M_0 \rceil < R. \tag{4}$$

The hardware amount is determined for the matrix circuits as a chip area occupied by an FSM's circuit [2]. Complexity of each matrix is defined by the area $S(MT_i)$ of a chip demanded for its implementation ($i=\overline{1,4}$). In theoretical articles this area is defined in conventional units [2]. The following estimations can be obtained for FSM U_1:

$$\begin{aligned} &S(M_1)=2(L+R)H_1;\ S(M_2)=H_1R;\\ &S(M_3)=2R\cdot M;\ \ S(M_4)=M\cdot N. \end{aligned} \tag{5}$$

The area $S(U_1)$ that is occupied by logic circuit of FSM U_1 is defined as a sum of the areas (5). Because of relations (4), the chip area occupied by the circuit of Moore FSM always exceeds this value for an equivalent Mealy FSM.

There are a lot of methods targeting hardware reduction of Moore FSM circuit [12, 18]. But they deal with either CPLD or FPGA chips. Because of it, they cannot be directly used in the case of PLA-based Moore FSM. Taking it into account, we propose new method targeting PLA-based Moore FSM. The proposed method can be viewed as a development of the method [6].

One of Moore FSM features is existence of pseudoequivalent states [5], which are the states with the same transitions by the effect of the same inputs [1]. There are two main optimization methods [5] based on the existence of pseudoequivalent states: method of optimal state assignment and method of transformation of the states codes.

Let us find the classes of pseudoequivalent states B_i ($i=\overline{1,I}$) for a given set A. These classes define a partition $\Pi_A=\{B_1,\ldots,B_I\}$ for the set A. Let us point out that $I=M_0$ [5].

Method of optimal state assignment. Let us construct the following system of equations for some Moore FSM:

$$B_i=\bigvee_{m=1}^{M} C_{im}A_m\ \ (i=\overline{1,I}). \tag{6}$$

In (10), the Boolean variable $C_{im}=1$ if $a_m\in B_i$. Let us execute the state assignment for a given FSM. In the case of the optimal state assignment, the system (6) has exactly I product terms. It is possible if the following statement is true for each class $B_i\in A$: the codes $K(a_m)$ belong to a single generalized interval of R-dimensional Boolean space. It leads to Moore FSM U_2. The structures of both U_1 and U_2 are the same. But in U_2 the matrix MT_1 produces only $H_0=H_2<H_1$ terms.

However, such an encoding that leads to $H_2 = H_0$ is not always possible [1]. The well-known algorithm JEDI [15] can be used for the optimal state assignment.

Method of transformation of the states codes. The classes $B_i \in A$ are encoded by the binary codes $K(B_i)$ with $R_B = \lceil log_2 I \rceil$ bits. The variables $\tau_r \in \tau$ are used for such an encoding, where $|\tau| = R_B$. Next system of functions is formed

$$\tau = \tau(T). \quad (7)$$

The system (7) specifies the law of transformation of the codes $K(a_m)$ into the codes $K(B_i)$. The structure table is transformed in such a way that the states $a_m \in B_i$ are replaced by the classes $B_i \in \Pi_A$ in the column of present state. Such an approach allows to decrease the number of rows of ST to H_0, whereas the number of feedback variables is reduced to $R_B < R$.

This approach leads to a Moore FSM U_3 with a code transformer (CT). The number of transitions of the Moore FSM U_3 is equal to H_0. The drawback of U_3 is the existence of a block of the code transformer that consumes additional resources of a chip.

III. The Basic Idea of a Proposed Method

Let us execute the state assignment in the following way. If $a_m \in B_i$, then the codes $K(a_m)$ include the codes $K(B_i)$. It means that the codes $K(B_i)$ are represented by some variables $T_r \in T'$, where $T' \subset T$. It results in Moore FSM U_4 whose structure is the same as the structure of Moore FSM U_1 (Fig. 1).

The proposed approach guarantees the reduction of the number of terms in the system Φ up to H_0. It leads to reducing area of matrix MT_1, as well as the number of inputs for matrix MT_2 MT2. Unfortunately, this approach does not permit the terms reduction in system (2).

Let us represent the coding space as a Karnaugh map having I_K columns, where

$$I_K = 2^{R_B}. \quad (8)$$

Let $|B_i| = M_i$ and $M_B = max(M_1,...,M_I)$, then the map should have

$$O_K = 2^{R_M} \quad (9)$$

rows, where

$$R_M = \lceil log_2 M_B \rceil. \quad (10)$$

If condition (9) is true, then the states for any class $B_i \in \Pi_A$ are placed in the same column of the map and the code $K(B_i)$ is determined by the values of variables marking this column. If the following condition

$$R_B + R_M > R \quad (11)$$

takes place, then the expansion of encoding space occurs.

The value of R_M can be decreased if the following is condition true:

$$I_K > I. \quad (12)$$

In this case up to $(I_K - I)$ classes B_i can be placed in adjacent cells of the map.

Let us discuss the following example with the partition $\Pi_A = \{B_1,...,B_5\}$, where $B_1 = \{a_1\}$, $B_2 = \{a_2,a_3,a_4,a_5,a_6\}$, $B_3 = \{a_7,a_8,a_9,a_{10}\}$, $B_4 = \{a_{11},a_{12},a_{13},a_{14},a_{15}\}$, $B_5 = \{a_{16},a_{17}\}$. Now, there is $R_B = 3$, $M_B = 5$ and $R_M = 3$. Therefore, it is enough $R_B + R_M = 6$ variables to encode the states. But the map includes $I_K = 8$ columns and condition (12) is true. Obviously, those classes $B_i \in \Pi_A$ having the maximum number of states should be transformed. Let us represent the class B_2 as $B_2^1 = \{a_2,...a_5\}$ and $B_2^2 = \{a_6\}$, whereas the class B_4 as $B_4^1 = \{a_{11},...a_{14}\}$ and $B_4^2 = \{a_{15}\}$. Now there are $M_B = 4$, $R_M = 2$ and variables $T_1,...,T_5$ are used for state encoding (Fig. 2). As follows from Fig. 2, there is one code for each class $B_i \in \Pi_A$, namely: $K(B_1) = *00$, $K(B_2) = 0*1$, $K(B_3) = 010$, $K(B_4) = 11*$, $K(B_5) = 10*$. In this case there is $R_B + R_M = 5$, so there is no code space expansion.

T_4T_5 \ $T_1T_2T_3$	000	001	011	010	110	111	101	100
00	a_1	a_2	a_6	a_7	a_{11}	a_{15}	a_{16}	*
01	*	a_3	*	a_8	a_{12}	*	a_{17}	*
11	*	a_4	*	a_9	a_{13}	*	*	*
10	*	a_5	*	a_{10}	a_{14}	*	*	*

Figure 2. The codes of states for Moore FSM $U_4(\Gamma_1)$

IV. PROPOSED DESIGN METHOD

The proposed design method includes the following steps:

1. Constructing the marked GSA and the set of states A.
2. Construction of the partition Π_A.
3. Primary state assignment.
4. Secondary state assignment.
5. Construction of the transformed structure table.
6. Implementation of FSM matrix circuit.

Let us discuss some steps of the proposed method.

Primary state assignment. This step is connected with finding values of I_K and O_K. Here the possibility of decreasing R_M under keeping R_B is analyzed. This step is reduced to the placement of states $a_m \in B_i$ inside the columns of Karnaugh map having the size $I_K \times O_K$. It allows finding the set $T' \subset T$ and codes $K(B_i)$.

Secondary state assignment. This step is connected with optimizing system (2). Let us represent equations of system (2) as the following ones:

$$y_n = \bigvee_{m=1}^{M} C_{nm} A_m \ (n = \overline{1,N}). \tag{13}$$

In (13) the symbol C_{nm} stands for the Boolean variable equal to 1 if the output variable $y_n \in Y$ is generated in the state $a_m \in A$. States $a_m \in B_i$ are rearranged in the column $K(B_i)$ to decrease the number of terms in system (13). The "don't care" cells of the map are used for this optimization.

Construction of transformed structure table. This table is constructed using the system of generalized formulae of transitions [1], which is the following one:

$$B_i \to \bigvee_{h=1}^{H_0} C_{ih} X_h A_s \ (i = \overline{1,I}). \tag{14}$$

In (14), the Boolean variable $C_{ih} = 1$, if the term $X_h A_s$ belongs to the system of transitions for the class $B_i \in \Pi_A$. This system is constructed using initial GSA Γ. The transformed structure table includes the columns B_i, $K(B_i)$, a_S, $K(a_S)$, X_k, Φ_k, h. The rows of this table correspond to the terms $F_h \in F$:

$$F_h = (\bigwedge_{r=1}^{R_B} T_r^{l_{ri}}) * X_h . (h = \overline{1,H_0}). \tag{15}$$

In (15), the first part is determined by the code $K(B_i)$ from the row h, whereas the value of $l_{ri} \in \{0,1,*\}$ is the value of code bit r; $T_r^0 = \overline{T_r}, T_r^1 = T_r, T_r^* = 1$, where $r = \overline{1,R_B}$.

The terms (15) belong to functions $D_r \in \Phi$, where:

$$D_r = \bigvee_{h=1}^{H_0} C_{rh} F_h \ (r = \overline{1,R_B + R_M}). \tag{16}$$

In (16), the Boolean variable $C_{rh} = 1$, iff the row h contains function $D_r \in \Phi$ $(h = \overline{1,H_0})$.

Implementation of FSM matrix circuit. Let us analyze the matrices from the circuit shown in Fig. 1. The matrix MT_1 implements terms (15); it has $I_1 = 2(L + R_B)$ inputs and $O_1 = H_0$ outputs. The matrix MT_2 implements functions (16); it has $I_2 = H_0$ inputs and $O_2 = R_B + R_M$. The matrix MT_3 implements terms (13); it has I_3 inputs and O_3 outputs depending on outcome of the secondary encoding stage ($I_3 \le 2(R_B + R_M), O_3 \le M$). If some functions $y_n \in Y$ are represented by a single term, then they are formed directly by the matrix MT_3. It decreases the number of inputs for MT_4 from $N-1$ to 0. The lower bound corresponds to the situation when all output variables are formed by the matrix MT_3. In the general case the matrix MT_4 has $I_4 \le M$ inputs and $O_4 \le N$ outputs.

Therefore we can find for matrices MT_1 and MT_2:

$$\begin{aligned} S(MT_1) &= 2(L + R_B)H_0; \\ S(MT_2) &= (R_B + R_M)H_0. \end{aligned} \tag{17}$$

In the worst case the area of matrices MT_3 and MT_4 is defined as

$$\begin{aligned} S(MT_3) &= 2(R_B + R_M)M; \\ S(MT_4) &= M \cdot N. \end{aligned} \tag{18}$$

Thus the proposed approach guarantees reduction of area that occupied by matrices MT_1 and MT_2 due to equality of terms in the system of input memory functions of Moore FSM to the equivalent Mealy FSM.

The area of matrices MT_3 and MT_4 can be optimized due to secondary state encoding. It can be done using "don't care cells" of Karnaugh map.

V. Analysis of the Proposed Method

Hardware amount for implementation of Moore FSM logic circuit depend on characteristics of GSA such as number of condition vertices, output variables and other. Let us use the probabilistic approach suggested by [10] and developed in [7]. There are three key points in such an approach:

1. The use of a class of flow charts instead of a particular flow chart. This class is characterized by the parameters p_1 and p_2. Let Q be the number of vertices in GSA Γ; Q_1 be the number of operator vertices in GSA Γ; Q_2 be the number of conditional vertices in GSA Γ. $p_1 = Q_1/(Q-2)$ (resp. $p_2 \approx 1-p_1$) is the probability of the event that a particular node of the flow chart Γ is an operational (resp. conditional) one.

2. The use of matrix realization for the circuit of the control unit [2] instead of the standard VLSI. In this case we can determine a hardware amount as the volume of matrices for a given circuit of the control unit.

3. The use of relative characteristic instead of absolute one. Let us study of the relation $\eta = S(U_i)/S(U_j)$, where $S(U_{i,j})$ is the volume of matrices for the implementation of the circuit of the control unit $U_{i,j}$.

Let us use the results of [2, 7], where estimation of the main FSM parameters are expressed as function of GSA features and some coefficients:

$$\begin{aligned} H_0 &= 4.44 + 1.44\,p_1 Q/p_3; \\ H &= 12.6 + 2.16\,p_1 Q/p_3; \\ M &= p_1 Q + 1;\ L = (1-p_1)Q/p_4. \end{aligned} \tag{19}$$

Here Q_0 is the number of COV, $p_3 = p_1 Q/Q_0 \in \{1.1;1.2\}$, $p_4 = p_2 Q/L \in \{1.1;1.2\}$.

Let us analyze method of optimal state encoding. The area of U_2 is defined as

$$\begin{aligned} S(U_2) &= S(MT_1) + S(MT_2) + S(MT_3) + S(MT_4) = \\ &= kH_0(3R + 2L) + M(2R + N). \end{aligned} \tag{20}$$

Here $k \in \{1,2;\ldots;1,8\}$ is efficient factor of optimal state encoding method [5]. Using equations (19) we can find $S(U_2) = f(p_1, Q, p_3, p_4, N, k)$. Final equation is lengthy and low-informative for readers therefore we don't show it here.

In the same way we can find $S(U_4) = f(p_1, Q, p_3, p_4, N, p_5)$, where p_5 is a factor of distribution states inside classes of pseudoequivalent states (M_B will possess a minimal possible value if $p_5 = 0$).

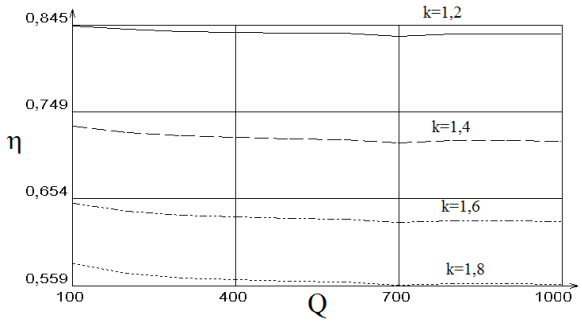

Figure 3. Comparison of model U_2 and U_4

The analyses of different GSA shows that $Q \in \{100,\ldots,1000\}$ and $N \le 100$. Let $N \in \{20;50;100\}$. Let us research function $\eta = S(U_4)/S(U_2)$. Some results of investigation are shown in Fig. 3 ($p_1 = 0,2$, $p_3 = 1,1$; $p_4 = 1.1$; $N = 20$; $p_5 = 0,6$).

Our investigations show that Moore FSM U_4 always offers gains in the area compared with Moore FSM U_2. This gain goes up with increasing the number of the vertices of the initial GSA. The best results (up to 44.24%) are achieved for flow charts with the number of vertices $500 \le Q \le 1000$. However efficiency of method decreases by 2–8 % with rising values of parameters N, p_1, p_5.

VI. Conclusion

The proposed method of expansion of encoding space targets on decrease in the chip space occupied by the matrix circuit of Moore FSM. This approach guarantees the decrease for the number of terms in the system of input memory functions of Moore FSM up to this value of the equivalent Mealy FSM. It allows decreasing for the chip space used for implementing the output variables. But to improve the outcome of this approach, it is necessary to work out an efficient method of the secondary state assignment.

The results of our investigations show that the proposed method is more effective for GSA with number of vertices more than 500. However gain is slightly decreased with rising numbers of output variables, parameters p_1 and p_5

The further direction of our research is connected with application of proposed methods for the cases when FSM logic circuits are implemented using such standard elements as CPLD and FPGA.

The co-author Olena Hebda 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.

REFERENCES

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

[2] S.Baranov, Logic synthesis for control automata, Norwell, MA, USA: Kluwer Academic Publishers, 1994.

[3] S. Baranov, I. Levin, O. Koren, M. Karpovski "Designing fault tolerant FSM by nano-PLA," in Proceedings of the 15th IEEE International On-Line Testing Symposium; 2009, Sembra-Lisbon, Portugal, pp.229-234.

[4] A. Barkalov, D. Das "Optimization of logic circuit of microprogrammed Mealy automaton on PLA," Automatic Control and Computer Sciences; vol. 25, No. 3, pp. 90–94, 1991.

[5] A. Barkalov, L. Titarenko, Logic synthesis for FSM-based control units. Lecture notes in electrical engineering. Berlin: Springer Verlag Heidelberg, 2009, no. 53.

[6] A. Barkalov, L. Titarenko, O. Hebda "Matrix implementation ofMoore FSM with expansion of coding space," Measurement, Automation and Monitoring,vol. 56, No. 7, pp. 694–696, 2010.

[7] A. Barkalov, Synthesis of Control Units on Programmable Logic Devices. DNTU, Donetsk, 2002, in Russian.

[8] G. DeMicheli, Synthesis and optimization of digital circuits. NY: McGraw-Hill, 1994.

[9] G. DeMicheli, R. Brayton, A. Sangiovanni-Vincentelli "Optimal state assignment for finite state machines," IEEE Trans. on CAD, vol. 4, pp. 269–284, 1985.

[10] G. Novikov "About one approach for finite state machines research," Contr. Syst. Mach., No. 2, pp. 70–75, 1974 (in Russian).

[11] A. D. Hon, M. Wilson "Nanowire-based sublithographic programmable logic arrays," International Journal of AMCS, vol. 17, No. 4, pp. 565–575, 2007.

[12] D. Kania, R.Czerwiski "Area and speed oriented synthesis of FSM for PAL-based CPLDs," Microprocessors and Microsystems, vol. 36, No. 1, pp. 45–61, 2012.

[13] I. Levin "Decomposition design of automata based on PLA with memory," Automatic Control and Computer Sciences, vol. 20, No. 2, pp. 61–68, 1986

[14] Z. Navabi, Embedded core design with FPGA, NY: McGraw-Hill, 1997.

[15] E. Sentovich, K. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanh, H. Savoj, P. Stephan, R. Brayton, A. Sangiovanni-Vincentelli, "SIS: a system for sequential circuit synthesis," in Proc. of International Conference on Computer Design (ICCD'92), Cambridge, MA, USA, pp. 328–333, 1992.

[16] A. Shrestha, A.Takaota, S. Tayu, S. Ueno "On two problems of nano-PLA design," IEEE Trans. on Informatics and Systems, vol. E94, No. 1, pp. 25–41, 2011.

[17] S. Singh, R. Singh, M. Bhatia "Performance evaluation of hybrid reconfigurable computing architecture over symmetrical FPGAs," International Journal of Embedded Systems & Applications, vol. 2, No. 3, pp. 107-116, 2012.

[18] I. Skliarova, V. Sklyarov, A. Sudnitson, Design of FPGA-based circuits using hierarchical finite state machines, Tallinn: TUT Press, 2008.

[19] V. Sklyarov, Synthesis of automata with matrix VLSIs, Minsk: Nauka i Technika, 1984, (in Russian).

[20] M. Smith, Application specific integrated circuits, Boston: Addison-Wesley, 1997.

[21] T. Villa, A. Sangiovanni-Vincentelli "NOVA: state assignment of finite state machines for optimal two-level logic implementation," IEEE Trans. on CAD,vol. 9, No. 9, pp. 905–924, 1990.

[22] XILINX. Website of the Xilinx Corporation;2012. http://www.xilinx.com