Int. J. Appl. Math. Comput. Sci., 2007, Vol. 17, No. 4, 565–575
DOI: 10.2478/v10006-007-0046-8

amcs

REDUCTION IN THE NUMBER OF PAL MACROCELLS IN THE CIRCUIT OF A MOORE FSM

ALEXANDER BARKALOV, LARYSA TITARENKO, SŁAWOMIR CHMIELEWSKI

Institute of Computer Engineering and Electronics, University of Zielona Góra
ul. Podgórna 50, 65-246 Zielona Góra, Poland
e-mail: {A.Barkalov, L.Titarenko@iie.uz.zgora.pl}, S.Chmielewski@weit.uz.zgora.pl

Optimization methods of logic circuits for Moore finite-state machines are proposed. These methods are based on the existence of pseudoequivalent states of a Moore finite-state machine, a wide fan-in of PAL macrocells and free resources of embedded memory blocks. The methods are oriented to hypothetical VLSI microcircuits based on the CPLD technology and containing PAL macrocells and embedded memory blocks. The conditions of effective application of each proposed method are shown. An algorithm to choose the best model of a finite-state machine for given conditions is proposed. Examples of proposed methods application are given. The effectiveness of the proposed methods is also investigated.

Keywords: Moore finite-state machine, complex programmable logic devices, design, logic circuit, pseudoequivalent states

1. Introduction

A control unit is a very important block of any digital system (De Micheli, 1994). A model of a Moore finite-state machine (FSM) is used very often to represent the control unit (Baranov, 1994). One of the most important steps in the design of FSM logic circuits is the encoding of its internal states. This step is known as the state assignment problem (De Micheli, 1994). In this step binary codes are assigned to FSM internal states. The quality of the resulting combinational part of the FSM (cost/area, power consumption, maximum frequency) depends heavily on the of outcome this step. Because of their importance, state assignment methods are continually being developed. There are effective state assignment methods based on symbolic minimization (Devadas *et al.,* 1988; Kam *et al.*, 1998; Villa *et al.*, 1990; 1998). Genetics algorithms (Chattopadhyay, 2005; Micheli *et al.*, 1985; Xia and Almaini, 2002) and other heuristics (Barkalov, 1998; 2005; Kania, 2004) are used for this problem solution, too. Let us point out that there is no universal effective state assignment algorithm fitting to any kind of control algorithm to be interpreted and logic elements to be used for the implementation of FSM logic circuits. This means that the peculiarities of components such as an FSM model, a control algorithm and logic elements should be taken into account to optimize the main characteristics of FSM circuits. Rapid evolution in semiconductor technology has resulted in the appearance of sophisticated VLSI circuits such as complex programmable logic devices (CPLDs) and field-programmable gate arrays (FPGAs) (Maxfield, 2004; Altera, 2007; Xilinx, 2007; Latticesemi, 2007). Such devices have enough resources to implement a complex digital system using only a single chip (Maxfield, 2004). One of the issues of the day in this area is a decrease in the hardware amount in FSM logic circuits (Adamski and Barkalov, 2006; Barkalov and Węgrzyn, 2006). The solution to this problem would permit to decrease the chip area occupied by an FSM circuit and give the potential possibility to increase the amount of digital system functions within the bounds of a single chip. In this article we are going to discuss the methods of Moore FSM design using a CPLD, which are popular to implement complex controllers (Barkalov and Węgrzyn, 2006; Kania, 2004). Unfortunately, in contrast to the FPGA, modern CPLDs have no embedded memory blocks, which can be used to implement the system of data-path microoperations. Therefore, in this article we deal with hypothetic CPLD chips, where programmable array logic (PAL) macrocells are used to implement the systems of Boolean functions and embedded memory blocks are used to implement the table functions of the digital system (Barkalov and Węgrzyn, 2006). The peculiarities of PAL macrocells are a wide fan-in and a very limited number of conjunctions (terms) per cell (Kania, 2004). A peculiarity of the known embedded me-

mory blocks is their configurability (Maxfield, 2004). For example, an embedded memory block of FLEX 10K can be configured as a memory block with the following characteristics: 256×8, 512×4, 1024×2, 2048×1 (Xilinx, 2007). This means that the number of embedded memory block outputs belongs to the set $\{1, 2, 4, 8\}$. The peculiarities of the Moore FSM are the existence of pseudoequivalent states (Barkalov, 1998) and the regular character of the system of output functions (microoperations) that makes its effective implementation possible using embedded memory blocks (Barkalov and Wegrzyn, 2006). In this article, we propose methods to optimize the amount of PAL macrocells in the logic circuit of the Moore FSM based on the above mentioned peculiarities.

2. Background of Moore FSM Design

Let the control algorithm of a digital system be specified by a graph scheme of algorithm (Baranov, 1994) $\Gamma = (B, E)$, where $B = \{b_0, b_E\} \cup E_1 \cup E_2$ is a set of the vertices and E is a set of edges. Here b_0 is an initial vertex, b_E is a final vertex, E_1 is a set of operational vertices, and E_2 is a set of conditional vertices. The vertex $b_q \in E_1$ contains a collection of microoperations $Y(b_q) \subseteq Y$, where $Y = \{y_1, \ldots, y_N\}$ is a set of microoperations of the digital system data-path (De Micheli, 1994). The vertex $b_q \in E_2$ contains some logic condition $x_e \in X$, where $X = \{x_1, \ldots, x_L\}$ is a set of logic conditions (flags) (Adamski, 2006). The initial and final vertices of the graph scheme of algorithm correspond to an initial state $a_1 \in A$, where $A = \{a_1, \ldots, a_M\}$is a set of internal states of a Moore FSM. Each operational vertex $b_q \in E_1$ corresponds to a unique state $a_m \in A$. The logic circuit of the Moore FSM U_1 is represented by the following systems of Boolean functions:

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

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

where $T = \{T_1, \ldots, T_R\}$ is a set of internal variables encoding the states $a_m \in A$, $R =]\log_2 M[$; $\Phi = \{D_1, \ldots, D_R\}$ is the set of the FSM input memory functions. The systems (1) and (2) are formed on the basis of a structure table with columns (Baranov, 1994): a_m is the current FSM state, $K(a_s)$ is the code of the state a_m, a_s is the next state, $K(a_s)$ is the code of the state a_s, X_h is the conjunction of some elements of the set X (or their complements) determining the transition $< a_m, a_s >$, Φ_h is the collection of input memory functions that are equal to 1 to switch the memory from $K(a_m)$ into $K(a_s)$, and $h = 1, \ldots, H_1(\Gamma)$ is the line number. The column a_m contains the collection of the microoperations $Y(a_m) \subseteq Y$ that are generated in the state $a_m \in A$. It is clear that $Y(b_q) = Y(a_m)$, where the vertex $b_q \in E_1$ is marked by the internal state $a_m \in A$. The structure diagram of a Moore FSM U_1 is shown in Fig. 1.

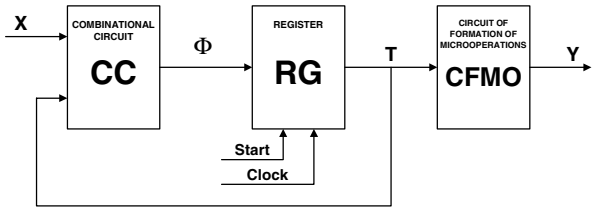

Fig. 1. Structure diagram of the Moore FSM U_1.

Here the combinational circuit (CC) forms the functions (1) and the circuit of formation of microoperations (CFMO) forms the functions (2). The register (RG) keeps the code $K(a_m)$. The pulse "Start" is used to load the code of the initial state $a_1 \in A$ into the register. The pulse "Clock" is used to change the content of the register. In this article we discuss the case when the CPLD technology is used in some SoPC. In this case the combinational circuit is implemented using PAL macrocells and the circuit of formation of microoperations is implemented using embedded memory blocks.

As a rule, the number of transitions $H_1(\Gamma)$ exceeds the number of transitions $H_0(\Gamma)$ of the equivalent Mealy FSM (Barkalov and Węgrzyn, 2006). It leads to an increase in the number of PAL macrocells in the circuit of the Moore FSM compared with the equivalent Mealy FSM. The value $H_1(\Gamma)$ can be decreased taking into account the pseudoequivalent states of the Moore FSM (Barkalov, 1998). The states $a_m, a_s \in A$ are pseudoequivalent states if identical inputs result in identical next states for both $a_m, a_s \in A$. This is possible if the outputs of the operational vertices marked by these states are connected with the input of the same vertex of the graph scheme of algorithm Γ. Let $\Pi_A = \{B_1, \cdots, B_I\}$ be a partition of the set A by the classes of pseudoequivalent states $(I \leq M)$. There are two main methods of Moore FSM optimization based on pseudoequivalent states (Barkalov, 1998; Barkalov and Węgrzyn, 2006):

- optimal encoding of the states;
- transformation of the codes of states into the codes of classes of pseudoequivalent states.

In the first case, the states $a_m \in A$ are encoded so that the codes of the states $a_m \in B_i$ $(i = 1, \ldots, I)$ belong to a single generalized interval of the R-dimensional Boolean space. This leads to a Moore FSM U_2 that has the same structure as the Moore FSM U_1. The algorithm from (De Micheli, 1994) can be used for such an encoding. In (Barkalov, 1998) it is shown that the number of transitions $H_2(\Gamma)$ of U_2 is decreased to $H_0(\Gamma)$. But such an encoding is not always possible (Adamski and Barkalov, 2006). In the second case, the classes $B_i \in \Pi_A$ are encoded by the binary codes $K(B_i)$ with $R_1 =]\log_2 I[$ bits. The variables $\tau_r \in \tau$ are used for such an encoding, where $|\tau| = R_1$. Let us point out that $I = M_0$, where

M_0 is the number of the states of the equivalent Mealy FSM. This approach leads to a Moore FSM U_3, with a code transformer (TC) (Fig. 2). In the Moore FSM U_3 the

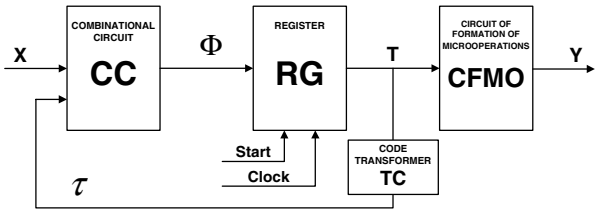

Fig. 2. Structure diagram of the Moore FSM U_3.

combinational circuit implements the functions

$$\Phi = \Phi(\tau, X) \tag{3}$$

and the code transformer implements the functions

$$\tau = \tau(T). \tag{4}$$

The number of transitions of the Moore FSM U_3 is equal to $H_0(\Gamma)$. The drawback of U_3 is the existence of a block of the code transformer that consumes additional resources of embedded memory blocks (in comparison with U_1).

In our article we propose to combine the application of an optimal encoding of the states and the transformation of the states codes. In this case the block of the code transformer can be even eliminated if some condition holds. The proposed method is based on the following features of the hypothetical CPLD in use:

- the fan-in of PAL macrocells exceeds significantly the maximal possible number of literals in terms of the system (1),
- the number of the outputs of the embedded memory block can be chosen from some restricted area.

The first feature permits us to use more than one source to represent the code of the current state $a_m \in A$. The second feature permits us to use some bits of the embedded memory block to represent the codes of the classes of pseudoequivalent states.

3. Main Ideas of the Proposed Method

Let the embedded memory block have q words if the number of its outputs $t_F = 1$. If $q \geq M$, then the embedded memory block should be configured in such a manner that it has

$$t_{\max} =]q/M[\tag{5}$$

outputs. The final value of the number of the outputs t_F is chosen from the set S_p that contains the possible fixed numbers of outputs. For example, if $t_{\max} = 6$ and $S_p = \{1, 2, 4, 8\}$, then $t_F = 4$.

The total amount of the outputs t_s of all embedded memory blocks of the circuit of formation of microoperations is determined as

$$t_s =]\frac{N}{t_F}[t_F. \tag{6}$$

In this case,

$$\Delta_t = t_s - N \tag{7}$$

outputs are free and they can be used to represent the codes of the classes of pseudoequivalent states.

If

$$\Delta_t \geq R_1, \tag{8}$$

then the graph scheme of algorithm Γ can be interpreted by a Moore FSM U_4 (Fig. 3). In the Moore FSM U_4

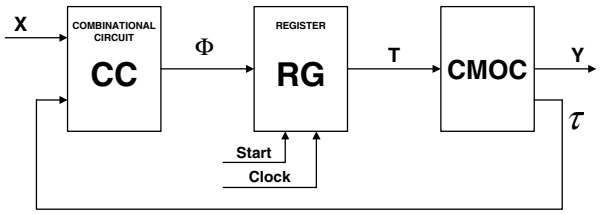

Fig. 3. Structure diagram of the Moore FSM U_4.

the combinational circuit forms the functions (3), and the circuit of formation of microoperations and the codes of the classes (CMOC) implements both the systems (2) and (4). In this case the block of code transformer is eliminated and the FSM states can be encoded in an arbitrary manner.

If (8) is violated, then we propose the following approach. Let us represent the set Π_A as $\Pi_A = \Pi_B \cup \Pi_C$, where $B_i \in \Pi_B$

$$|B_i| > 1, \tag{9}$$

otherwise $B_i \in \Pi_C$.

It is clear that the circuit of the code transformer should generate only the codes $K(B_i)$, where $B_i \in \Pi_B$. Let us encode the states $a_m \in A$ in an optimal way (Barkalov, 1998), and let us represent the set Π_B as $\Pi_B = \Pi_D \cup \Pi_E$. Here $B_i \in \Pi_D$ if the codes of the states belong to a single generalized interval of the Boolean space. Now only the codes of the states $a_m \in A(\Pi_E)$ should be transformed, where $A(\Pi_j)$ is a set of the states, where $B_i \in \Pi_j$ $(j = A, B, C, D, E)$. It is to take enough $R_2 =]\log_2(|\Pi_E| + 1)[$ binary variables to encode the classes $B_i \in \Pi_E$. Let these variables form a set Z, where $|Z| = R_2$. If

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

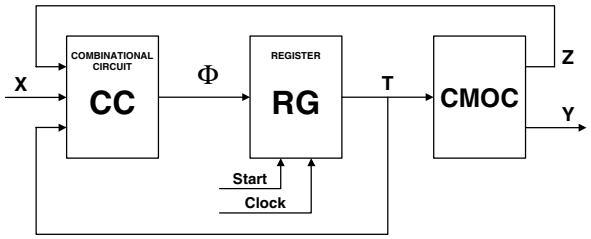

Fig. 4. Structure diagram of the Moore FSM U_5.

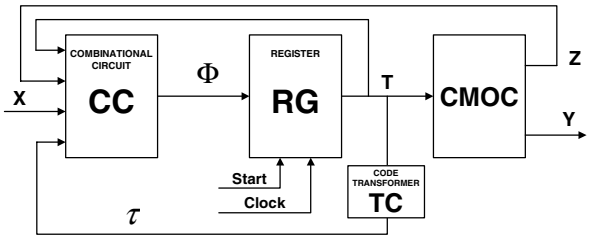

Fig. 5. Structure diagram of the Moore FSM U_6.

then the graph scheme of algorithm Γ can be interpreted by a Moore FSM U_5 (Fig. 4).

Here the combinational circuit forms the functions

$$\Phi = \Phi\left(T, Z, X\right), \tag{11}$$

the CMOC forms both functions (2) and the functions

$$Z = Z(T). \tag{12}$$

In the FSM U_5 the block of the code transformer is missing and the variables $T_r \in T$ represent both the states $a_m \in A(\Pi_C)$ and the classes $B_i \in \Pi_D$. The classes $B_i \in \Pi_E$ are represented by the CMOC. In this case the number of inputs in the PAL macrocells is increased from $L + R_1$ (the FSM U_3) to $L + R + R_2$ (the FSM U_5,) but it does not increase the hardware amount in the CC in comparison with the FSM U_3. The cycle times of U_1 and U_5 are the same in the worst case. In the best case, the combinational circuit of U_5 has fewer levels than the combinational circuit of U_1. This means that the cycle time of U_5 can be less than that of U_1. Therefore, the proposed approach permits us to decrease the hardware amount without the decrease in the performance of the digital system. Let us point out that the cycle times of U_2, U_3, U_4, U_5 are the same.

If (8) and (10) are violated, then we propose to represent the set Π_E as $\Pi_E = \Pi_F \cup \Pi_G$. The set Π_F includes n_F classes, where

$$n_F = 2^{\Delta_t} - 1. \tag{13}$$

The codes of the classes $B_i \in \Pi_F$ are kept in the CMOC and the variables $z_r \in Z$ are used for their representation, where $|Z| = \Delta_t$. The set Π_G includes

$$n_G = I - n_C - n_D - n_F \tag{14}$$

classes, where $n_C = |\Pi_C|$, $n_D = |\Pi_D|$. These classes can be encoded using the variables $\tau_r \in \tau$, where $|\tau| = R_3$ and

$$R_3 =]\log_2(n_G + 1)[. \tag{15}$$

In this case we propose to interpret the graph scheme of algorithm Γ by a Moore FSM U_6 (Fig. 5).

Here the combinational circuit forms the functions

$$\Phi = \Phi(T, Z, \tau, X), \tag{16}$$

the CMOC forms both the functions (2) and (12), and the circuit of the code transformer forms the functions (4). In the FSM U_6 the number of the inputs of the PAL macrocells is equal to $L + R + \Delta_t + R_3$, but the combinational circuit has the same hardware amount as in the case of the FSM U_3. The block of the code transformer of U_6 has less hardware than that of U_3.

The Moore FSM U_6 has the most complex structure and its design method includes the biggest amount of steps in comparison with the FSM $U_1 - U_5$. In our article we propose the design method of the FSM U_6 including the following steps:

1. Construction of a marked graph scheme of the algorithm Γ and the construction of the set of internal states $A = \{a_1, \ldots, a_M\}$ of Moore FSM.
2. Construction of the partition $\Pi_A = \Pi_B \cup \Pi_C$.
3. Optimal encoding of the states and the construction of the sets Π_D and Π_E.
4. Calculation of Δ_t t and the construction of the sets Π_F and Π_G.
5. Encoding the classes $B_i \in \Pi_F \cup \Pi_G$.
6. Construction of the table of the CMOC.
7. Construction of the modified structure table of the FSM.
8. Construction of the table of the code transformer.
9. Implementation of the FSM logic circuit.

The choice of a particular model depends on some conditions. In this article we propose the algorithm given in (Fig. 6).

If the condition (8) holds, then the model U_4 should be chosen. Otherwise the optimal encoding of the states should be executed. If all classes $B_i \in \Pi_A$ are represented by unique generalized intervals of the Boolean space ($\Pi_E = \varnothing$), then the model U_5 should be chosen.

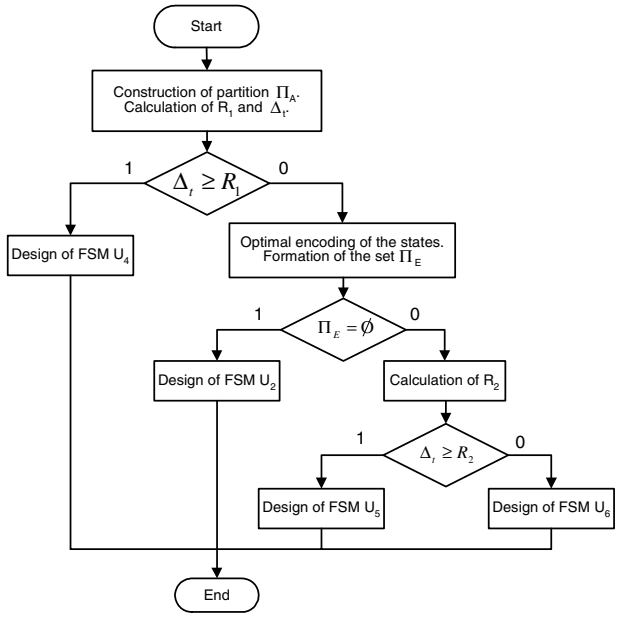

Fig. 6. Choice of the Moore FSM model.

If $\Delta_t < R_1$ and $\Pi_E = \varnothing$, then the condition (10) determines the optimal model of the Moore FSM for the interpretation of the graph scheme of algorithm Γ using the hardware of an SoPC with the CPLD technology.

4. Application Examples of the Proposed Methods

Let us discuss some examples in the case when the control algorithm is represented by the marked graph scheme of algorithm Γ_1 (Fig. 7). The design method will be found from Fig. 6 using the parameter q of the embedded memory block in use.

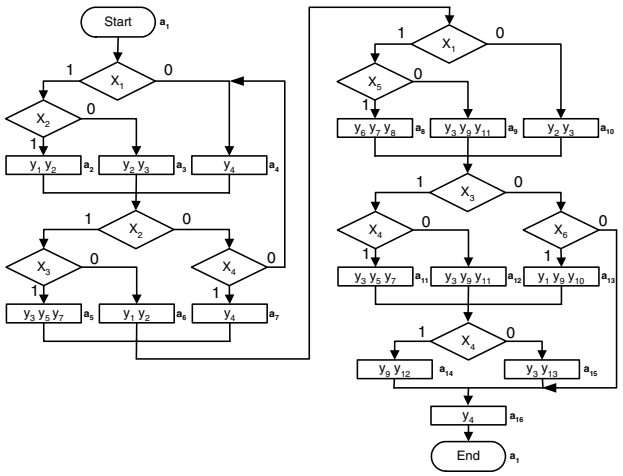

Fig. 7. Marked graph scheme of algorithm Γ_1.

We can get the following characteristics of the control unit from Fig. 7: $A = \{a_1, \ldots, a_{16}\}$, $M = 16$, $R = 4$, $T = \{T_1, \ldots, T_4\}$, $\Phi = \{D_1, \ldots, D_4\}$, $Y = \{y_1, \ldots, y_{13}\}$, $N = 13$. Let us encode the states $a_m \in A$ in a trivial way: $K(a_1) = 0000$, $K(a_2) = 0001, \ldots, K(a_{16}) = 1111$. Let the symbol $U_i(\Gamma_i)$ mean that the Moore FSM U_i interprets the graph scheme of algorithm Γ_j. Let us find a system of transition formulas (Baranov, 1994) for the states $a_m \in A$. If the outputs of the vertices marked by $a_i, a_j \in A$ are connected with the input of the same vertex of the graph scheme of algorithm Γ, then we will combine the transition formulas for these states into a single formula of transition. In the case of the graph scheme of algorithm Γ_1, we can form the following system:

Table 1. Fragment of the structure table of the Moore FSM $U_1(\Gamma_1)$.

a_m	$K(a_m)$	a_s	$K(a_s)$	X_h	ϕ_h	h
$a_2(y_1y_2)$	0001	a_5	0100	x_2x_3	D_2	4
		a_6	0101	$x_2\bar{x}_3$	D_2D_4	5
		a_7	0110	$\bar{x}_2x_4$	D_2D_3	6
		a_4	0011	$\bar{x}_2\bar{x}_4$	D_3D_4	7

$$
\begin{aligned}
&a_1 \rightarrow x_1x_2a_2 \vee x_1\bar{x}_2a_3 \vee \bar{x}_1a_4,\\
&a_2, a_3, a_4 \rightarrow x_2x_3a_5 \vee x_2\bar{x}_3a_6 \vee \bar{x}_2x_4a_7 \vee \bar{x}_2\bar{x}_4a_4,\\
&a_5, a_6, a_7 \rightarrow x_1x_5a_8 \vee x_1\bar{x}_5a_9 \vee x_1a_{10},\\
&a_8, a_9, a_{10} \rightarrow x_3x_4a_{11} \vee x_3\bar{x}_4a_{12} \vee \bar{x}_3x_6a_{13} \vee \bar{x}_3\bar{x}_6a_{16},\\
&a_{11}, a_{12}, a_{13} \rightarrow x_4a_{14} \vee \bar{x}_4a_{15},\\
&a_{14}, a_{15} \rightarrow a_{16},\ a_{16} \rightarrow a_1.
\end{aligned} \tag{17}
$$

It is clear that the states from the left-hand side of each transition formula are pseudoequivalent states. Thus, in the case of the FSM $U_1(\Gamma_1)$ we can form the partition $\Pi_A = \{B_1, \ldots, B_7\}$, where $B_1 = \{a_1\}$, $B_2 = \{a_2, a_3, a_4\}$, $B_3 = \{a_5, a_6, a_7\}$, $B_4 = \{a_8, a_9, a_{10}\}$, $B_5 = \{a_{11}, a_{12}, a_{13}\}$, $B_6 = \{a_{14}, a_{15}\}$, $B_7 = \{a_{16}\}$ and $I = 7$. Let $|B_i| = n_i$ and H_i be the number of the terms in the transition formula for the class $B_i \in \Pi_A$. The number $H_1(\Gamma)$ of the lines in the structure table of the Moore FSM $U_1(\Gamma)$ can be found as

$$H_1(\Gamma) = \sum_{i=1}^{I} n_i H_i. \tag{18}$$

In the case of the FSM $U_1(\Gamma_1)$ we can get $H_1(\Gamma_1) = 45$. This means that the structure table of the Moore FSM $U_1(\Gamma_1)$ has 45 lines. Some part of this table is shown in Table 1.

This table is a basis to form the system (1). For example, from Table 1 we can get part of the Boolean equation for the function $D_4 \in \Phi$:

$$D_4 = \bar{T}_1\bar{T}_2\bar{T}_3T_4x_2\bar{x}_3 \vee \bar{T}_1\bar{T}_2\bar{T}_3T_4\bar{x}_2\bar{x}_4.$$

Let us discuss the case when the system (2) is implemented using embedded memory blocks with $q = 64$ if $t_F = 1$, and $S_p = \{1, 2, 4, 8\}$. From (5) we can get $t_{\max} = 4$ and $t_{\max} = t_F$, because $t_{\max} \in S_p$. This means that the circuit of formation of microoperations of the Moore FSM can be implemented using $]N/t_F[= 4$ embedded memory blocks. From (6) we have $t_s = 16$ and from (7) we have $\Delta_t = 3$. In the case of the FSM $U_1(\Gamma_1)$ we have $I = 7$. This means that $R_1 = 3$ and $\tau = \{\tau_1, \tau_2, \tau_3\}$. The condition (8) holds, and according to the choice algorithm (Fig. 6) we should use the model U_4 for the interpretation of the graph scheme of algorithm Γ_1.

Let us encode the classes $B_i \in \Pi_A$ in a trivial way: $K(B_1) = 000$, $K(B_2) = 001, \ldots, K(B_7) = 110$. The CMOC table has the following columns: a_m, $K(a_m)$, $Y(a_m)$, $K(B_i)$, m. The m-th line of this table contains both the microoperations $y_n \in Y(a_m)$ and the code $K(B_i)$, where $a_m \in B_i$ $(m = 1, \ldots, M)$. This table is formed in a trivial way. To save space, let us show the content of the CMOC as Table 2.

Table 2. Content of the CMOC of the Moore FSM $U_4(\Gamma_1)$.

T_1T_2 \ T_3T_4	00	01	11	10
00	-	$y_1y_2z_3$	$y_2y_3z_3$	y_4z_3
01	$y_3y_5y_7z_2$	$y_1y_2z_2$	y_4z_2	$y_6y_7y_8z_2z_3$
11	$y_3y_9y_{11}z_2z_3$	$y_2y_3z_2z_3$	$y_3y_5y_7z_1$	$y_3y_9y_{11}z_1$
10	$y_1y_9y_{10}z_1$	$y_9y_{12}z_1z_3$	$y_3y_{13}z_1z_3$	$y_4z_1z_2$

For example, the cell 0111 corresponds to the state a_8 with $Y(a_8) = (y_6, y_7, y_8)$. Because $a_8 \in B_4$ with $K(B_4) = 011$, then the cell 0111 contains y_6, y_7, y_8, z_2 and z_3. The other cells from Table 2 are filled in the same manner.

To form a modified structure table of the Moore FSM $U_4(\Gamma_1)$, replace the states $a_m \in B_i$ and the left-hand side of each transition formula by the corresponding class $B_i \in \Pi_A$. This leads to the system

$$
\begin{aligned}
&B_1 \to x_1x_2a_2 \vee x_1\bar{x}_2a_3 \vee \bar{x}_1a_4,\\
&B_2 \to x_2x_3a_5 \vee x_2\bar{x}_3a_6 \vee x_2x_4a_7 \vee \bar{x}_2\bar{x}_4a_4,\\
&B_3 \to x_1x_5a_8 \vee x_1\bar{x}_5a_9 \vee \bar{x}_1a_{10},\\
&B_4 \to x_3x_4a_{11} \vee x_3\bar{x}_4a_{12} \vee \bar{x}_3x_6a_{13} \vee \bar{x}_3\bar{x}_6a_{16},\\
&B_5 \to x_4a_{14} \vee \bar{x}_4a_{15},\\
&B_6 \to a_{16}, B_7 \to a_1.
\end{aligned} \tag{19}
$$

The modified structure table corresponds to a system similar to (19) and it has the columns B_i, $K(B_i)$, a_s, $K(a_s)$, X_h, Φ_h and h. Moreover, it has

$$H_4(\Gamma) = \sum_{i=1}^{I} H_i \tag{20}$$

Table 3. Fragment of the modified structure table of the Moore FSM $U_4(\Gamma_1)$.

B_i	$K(B_i)$	a_s	$K(a_s)$	X_h	ϕ_h	h
B_1	000	a_2	0001	x_1x_2	D_4	1
		a_3	0010	$x_1\bar{x}_2$	D_3	2
		a_4	0011	$\bar{x}_1$	D_3D_4	3
B_2	001	a_5	0100	x_2x_3	D_2	4
		a_6	0101	$x_2\bar{x}_3$	D_2D_4	5
		a_7	0110	$\bar{x}_2x_4$	D_2D_3	6
		a_4	0011	$\bar{x}_2\bar{x}_3$	D_3D_4	7

lines. It is clear that $H_4(\Gamma) = H_0(\Gamma)$, where $H_0(\Gamma)$ is the number of lines in the structure table of the equivalent Mealy FSM. In case of the FSM $H_4(\Gamma_1)$, its modified structure table has $H_4(\Gamma_1) = 18$ lines. The part of this table for classes $B_1, B_2 \in \Pi_A$ is shown in Table 3.

This table is a basis to form the system (3). For example, from Table 3 we can form part of the Boolean equation of the function D_4:

$$D_4 = \bar{\tau}_1\bar{\tau}_2\bar{\tau}_3x_1 \vee \bar{\tau}_1\bar{\tau}_2\tau_3x_2\bar{x}_3 \vee \bar{\tau}_1\bar{\tau}_2\tau_3\bar{x}_2\bar{x}_4.$$

The implementation of the logic circuit of the FSM U_4 is reduced to the implementation of the system (3) using PAL macrocells and the implementation of the systems (2) and (4) using embedded memory blocks. There are effective methods for such implementation (Barkalov and Węgrzyn, 2006;). We therefore exclude this step from our deliberations.

Let $H_i(D_r)$ be the number of the terms in the function D_r $(r = 1, \ldots, R)$ for the FSM U_i $(i = 1, \ldots, 6)$. An analysis of the complete structure table of the FSM $U_1(\Gamma_1)$ shows that $H_1(D_1) = 26$, $H_1(D_2) = H_1(D_3) = H_1(D_4) = 25$. An analysis of the complete modified structure table of the FSM $U_4(\Gamma_1)$ shows that $H_4(D_1) = H_4(D_2) = 9$, $H_4(D_3) = H_4(D_4) = 10$. Let $Q_i(D_r, S)$ be the number of PAL macrocells with S terms to implement the function $D_r \in \Phi$ for the FSM U_i $(i = 1, \ldots, 6)$. Using the results from (Barkalov and Węgrzyn, 2006), the value of $Q_i(D_r, S)$ can be calculated as

$$Q_i(D_r, S) = \left]\frac{H_i(D_r) - 1}{S - 1}\right[. \tag{21}$$

If, e.g., $S = 6$, then $Q_1(D_r, 6) = 5$ and $Q_4(D_r, 6) = 2$ $(r = 1, \ldots, 4)$. This means that the combinational circuit of $U_1(\Gamma_1)$ includes $Q_1(\Gamma_1) = 20$ PAL macrocells and the combinational circuit of $U_4(\Gamma_1)$ includes $Q_4(\Gamma_1) = 8$ PAL macrocells. Therefore, in this case the hardware amount in the combinational circuit is decreased to 60%. The numbers of embedded memory blocks in both the CMOC of $U_4(\Gamma_1)$ and the circuit of formation of microoperations of $U_1(\Gamma_1)$ are the same. The cycle

times of both $U_1(\Gamma_1)$ and $U_4(\Gamma_1)$ are the same. Let us point out that in the case of the graph scheme of algorithm Γ_1 we have

$$\frac{Q_1(\Gamma_1)}{Q_4(\Gamma_1)} = \frac{H_1(\Gamma_1)}{H_4(\Gamma_1)}. \tag{22}$$

Now let us discuss the case when $q = 32$, if $t_F = 1$, and $S_p = \{1, 2, 4, 8\}$. From (5) we can get $t_{\max} = t_F = 2$. This means that the circuit of formation of microoperations of the Moore FSM $U_1(\Gamma_1)$ is implemented using $]N/t_F[= 7$ embedded memory blocks.

From (6) we have $t_S = 14$ and from (7) we have $\Delta_t = 1$. This means that the condition (8) is violated and an optimal encoding of the states should be applied. Using an algorithm from (De Micheli, 1994) we can get the following result regarding the optimal encoding of states of the FSM $U_1(\Gamma_1)$ (Table 4). From the Karnaugh

Table 4. Optimal encoding of the states of the Moore FSM $U_1(\Gamma_1)$.

T_1T_2 \ T_3T_4	00	01	11	10
00	a_1	a_2	a_3	a_4
01	a_5	a_6	a_7	a_{14}
11	a_8	a_9	a_{10}	a_{15}
10	a_{11}	a_{12}	a_{13}	a_{16}

map of Tab. 4 we get $\Pi_C = \{B_1, B_7\}$, $\Pi_D = \{B_6\}$, $\Pi_E = \{B_2, \ldots, B_5\}$, $|\Pi_E| = 4$. From (9) we have $R_2 = 3$ and $\Delta_t < R_2$. This means that the condition (10) is violated and the Moore FSM U_6 should be applied to interpret the graph scheme of algorithm Γ_1. From (13) we get $n_F = 1$, which implies $n_G = 3$. Now we have the following sets of classes $B_i \in \Pi_A : \Pi_C = \{B_4, B_7\}$, $\Pi_D = \{B_6\}$, $\Pi_F = \{B_2\}$, $\Pi_G = \{B_3, B_4, B_5\}$. According to Fig. 5, the codes of the classes $B_i \in \Pi_C \cup \Pi_D$ are represented by a register, the codes of the classes $B_i \in \Pi_F$ are represented by the CMOC and the codes of the classes $B_i \in \Pi_G$ are represented by the code transformer.

From the Karnaugh map (Tab. 4) we get the following codes: $K(B_1) = K(a_1) = 0000$, $K(B_6) = *110$, $K(B_7) = K(a_{16}) = 1010$. Since $\Delta_t = 1$, we have $Z = \{z_1\}$. Let $K(B_2) = 1$ and let $z_1 = 0$ means that the codes of the classes $B_i \in \Pi_F$ are not used to form the current transition of the FSM. The number of variables in the set τ can be determined using (15). In our example we have $R_3 = 2$ and $\tau = \{\tau_1, \tau_2\}$. Let us encode the classes $B_i \in \Pi_G$ in the following manner: $K(B_3) = 01$, $K(B_4) = 10$, $K(B_5) = 11$. The input assignment $\tau_1 = \tau_2 = 0$ means that the codes of the classes $B_i \in \Pi_G$ are not used to form the current FSM transition.

The CMOC of the Moore FSM $U_6(\Gamma_1)$ is represented by Tab. 5.

Table 5. Content of the CMOC of the Moore FSM $U_6(\Gamma_1)$.

T_1T_2 \ T_3T_4	00	01	11	10
00	-	$y_1y_2z_1$	$y_2y_3z_1$	y_4z_1
01	$y_3y_5y_7$	y_1y_2	y_4	y_9y_{12}
11	$y_6y_7y_8$	$y_3y_9y_{11}$	y_2y_3	y_3y_{13}
10	$y_3y_5y_7$	$y_3y_9y_{11}$	$y_1y_9y_{10}$	y_4

The modified structure table of the Moore FSM U_6 is constructed based on a modified system of the formulae of transitions. In the case of the FSM $U_6(\Gamma_1)$ this system is represented by (19). This table has the same columns as the modified structure table of the Moore FSM U_4. The column $K(B_i)$ contains the code

$$K(B_i) = [K(B_i)^C \vee K(B_i)^D] * K(B_i)^F * K(B_i)^G \tag{23}$$

where $K(B_i)^j$ is the code of the class $B_i \in \Pi_j$ $(j = C, D, F, G)$, '$*$' signifies concatenation. The number of lines $H_6(\Gamma)$ is determined as $H_4(\Gamma)$. In the case of the FSM $U_6(\Gamma_1)$ we have $H_6(\Gamma_1) = 18$. The transitions for the classes $B_1, B_2, B_3 \in \Pi_A$ are shown in Table 3.

The code $K(B_i)$ is represented by the variables T_1, $T_2, T_3, T_4, \tau_1, \tau_2, z_1$. If $\tau_1 \vee \tau_2 \vee z_1 = 1$, then $B_i \in \Pi_F$ or $B_i \in \Pi_G$. In this case the code of $a_m \in A$ is ignored and it is represented by the signs '$*$' in the column $K(B_i)$. This table is a basis to form the system (16). From Table 3 we can get, e.g.,

$$D_4 = \bar{T}_1\bar{T}_2\bar{T}_3\bar{T}_4\bar{\tau}_1\bar{\tau}_2\bar{z}_1x_1 \vee \bar{\tau}_1\bar{\tau}_2z_1x_2\bar{x}_3 \vee \bar{\tau}_1\bar{\tau}_2z_1\bar{x}_2x_4 \vee \bar{\tau}_1\tau_2\bar{z}_1x_1\bar{x}_5 \vee \tau_1\tau_2\bar{z}_1\bar{x}_1.$$

The table of the circuit of the code transformer contains the columns a_m, $K(a_m)$, B_i, $K(B_i)$, τ_m, m, where $a_m \in A(\Pi_G)$. In the case of the FSM $U_6(\Gamma_1)$ this table includes 6 lines (Table 6).

If some line of this table includes more than one state, then the column $K(a_m)$ contains the generalized interval corresponding to the codes of these states. The table of the code transformer is a basis to form the functions (4). The codes of the states $a_m \notin A(\Pi_G)$ can be treated as "don't care" input assignments (McCluskey, 1986) and they can be used to minimize the functions (4). The Karnaugh map for the function $\tau_1 \in \tau$ is shown in Tab. 8.

From this map we can get $\tau_1 = T_1$. Using the same approach, we can get $\tau_2 = \bar{T}_1 \vee \bar{T}_2$. Implementation of the logic circuit of the finite-state machine U_6 is reduced to the implementation of systems (4) and (16) using PAL macrocells and to the implementation of the systems (2) and (12) using embedded memory blocks.

In the case of the Moore FSM $U_6(\Gamma_1)$ we have $H_6(D_1) = 9$, $H_6(D_2) = H_6(D_4) = 10$, $H_6(D_3) =$

Table 6. Fragment of the modified structure table of the Moore FSM $U_6(\Gamma_1)$.

B_i	$K(B_i)$	a_s	$K(a_s)$	X_h	ϕ_h	h
B_1	0000000	a_2	0001	x_1x_2	D_4	1
		a_3	0011	$x_1\bar{x}_2$	D_3D_4	2
		a_4	0010	$\bar{x}_1$	D_3	3
B_2	****001	a_5	0100	x_2x_3	D_2	4
		a_6	0101	$x_2\bar{x}_3$	D_2D_4	5
		a_7	0111	$\bar{x}_2x_4$	$D_2D_3D_4$	6
		a_4	0010	$\bar{x}_2\bar{x}_4$	D_3	7
B_3	****000	a_8	1100	x_1x_5	D_1D_2	8
		a_9	1101	$x_1\bar{x}_5$	$D_1D_2D_4$	9
		a_{10}	1111	$\bar{x}_1$	$D_1D_2D_3D_4$	10

Table 7. Table of the code transformer of the Moore FSM $U_6(\Gamma_1)$.

a_m	$K(a_m)$	B_i	$K(B_i)$	τ_m	m
a_5, a_6	010*	B_3	01	τ_2	1
a_7	0111	B_3	01	τ_2	2
a_8, a_9	110*	B_4	10	τ_1	3
a_{10}	1111	B_4	10	τ_1	4
a_{11}, a_{12}	100*	B_5	11	$\tau_1\tau_2$	5
a_{13}	1011	B_5	11	$\tau_1\tau_2$	6

Table 8. Karnaugh map for the function τ_1.

T1T2 \ T3T4	00	01	11	10
00	*	*	*	*
01	0	0	0	*
11	1	1	1	*
10	1	1	1	*

10. If PAL macrocells have $S = 6$, then from (20) we get $Q_6(\Gamma_1) = 8$. To implement the circuit of the code transformer of the FSM $U_6(\Gamma_1)$, it is enough to take only $TC_6(\Gamma_1) = 1$ macrocell. Here $TC_i(\Gamma_j)$ means the amount of hardware to implement the circuit of code transformer of the FSM U_i that interprets the graph scheme of the algorithm Γ_j. Thus, only $Q_6(\Gamma_1)+TC_6(\Gamma_1) = 9$ macrocells should be used to implement an arbitrary logic of the FSM $U_6(\Gamma_1)$. Therefore, in this case the number of PAL macrocells is decreased to 55% in comparison with the FSM $U_1(\Gamma_1)$. The other characteristics of both $U_1(\Gamma_1)$ and $U_6(\Gamma_1)$ are the same (the cycle time and the number of embedded memory blocks).

5. Analysis of the Proposed Method

Let us find an area where the FSM $U_i(i = 4, 5, 6)$ has less hardware amount than the FSM $U_j(j = 1, 2, 3)$. Let us use the probabilistic approach described in (Barkalov and Barkalov, 2005). There are three key points in such an approach:

1. The use of the class of graph schemes of algorithm instead of a particular graph scheme of algorithm Γ. Each class is characterized by the parameters
$$p_1 = |E_1| / |B|, \quad p_2 = |E_2| / |B|. \tag{24}$$
It is clear that
$$\lim_{K(\Gamma)\to\infty} (p_1 + p_2) = 1, \tag{25}$$
where $K(\Gamma) = |B|$. Therefore p_1 (resp. p_2) can be treated as the probability of the event that a particular vertex of the graph scheme of algorithm Γ is an operational (resp. conditional) one.

2. The use of the matrix realization of the FSM circuit (Baranov, 1994) instead of the implementation using some standard VLSI. In this case we can find a hardware amount as the area of the matrices for a given structure of the logic circuit of the finite-state machine.

3. To study the relations $S(U_i)/S(U_j)$, where $S(U_i)$ an $S(U_j)$ are the areas of the matrices for the FSMs U_i and U_j, respectively. In (Barkalov and Wegrzyn, 2006) it is proved that such relations for the cases of the matrix realization are the same as for circuits implemented with standard programmable logic devices, such as PAL, PLA or PROM.

A matrix realization of the Moore FSM U_1 is shown in Fig. 8. Here M_1 is a conjunctive matrix that implements the system F of the terms of the system (1). M_2 is a disjunctive matrix that implements the functions of the system (1). M_3 is a conjunctive matrix that implements the

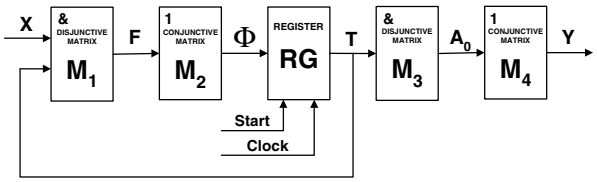

Fig. 8. Matrix realization of the Moore FSM U_1.

system A_0, where each function corresponds to the conjunction A_m $(m = 1, \ldots, M)$ to the code $K(a_m)$ of the state $a_m \in A$; M_4 is a disjunctive matrix that implements the functions (2). It is clear that the matrices M_1 and M_2 represent the combinational circuit, and the matrices M_3 and M_4 represent the circuit of formation of microoperations. The complexity of these circuits can be expressed as

$$S(CC)_1 = 2(L + R) \cdot H_1(\Gamma) + H_1(\Gamma) \cdot R,$$
$$S(CFMO)_1 = 2^R 2R + 2^R N. \qquad (26)$$

A matrix realization of the finite-state machine U_4 is shown is Fig. 9.

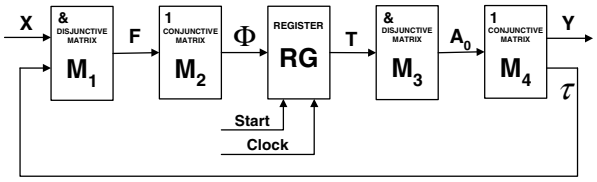

Fig. 9. Matrix realization of Moore FSM U_4.

Here the set F includes $H_0(\Gamma)$ elements, the set τ includes R_0 elements, where R_0 is the number of internal variables of the equivalent Mealy finite-state machine. It means that the complexity of the combinational circuit can be calculated as

$$S(CC)_4 = 2(L + R_0) \cdot H_0(\Gamma) + H_0(\Gamma) \cdot R. \qquad (27)$$

It is clear from the method of design of the finite-state machine U_4 that

$$S(CFMO)_1 = S(CMOC)_4. \qquad (28)$$

To find the range of effective application of the Moore finite-state machine U_4 we should examine the functions:

$$f_1 = S(CC)_4 / S(CC)_1;$$
$$f_2 = [S(CC)_4 + S(CFMO)_1] \,/\, [S(CC)_1 + S(CFMO)_1]. \qquad (29)$$

The function f_1 shows the decrease in the total area occupied by in matrices M_1 and M_2 due to the application of the model U_4 instead of the model U_1. The function f_2 shows the total decrease in the hardware amount in this case.

To reduce the number of variables in the expressions (26)–(31) we can use the results of (Barkalov and Wegrzyn, 2006), where the parameters L, R_0, R, $H_0(\Gamma)$, $H_1(\Gamma)$ are expressed as functions of $K(\Gamma)$ and some coefficients:

$$L = [(1 - p_1) \cdot K(\Gamma)] / p_4;$$
$$R_0 =]\log_2 (3,55 + 0,44 \cdot p_1 \cdot K(\Gamma)) / p_3[;$$
$$R =]\log_2 p_1 \cdot K(\Gamma)[;$$
$$H_0(\Gamma) = [4,44 + 1,44 \cdot p_1 \cdot K(\Gamma)] / p_3;$$
$$H_1(\Gamma) = 17,4 + [2,16 \cdot K(\Gamma) \cdot p_1] / p_3. \qquad (30)$$

Here $p_3 = |E_1|/Q$, where Q is the number of microinstructions of a graph-scheme of algorithm Γ, $p_3 = \{1,3; 1,4\}$; $p_4 = |E_2|/L$, $p_4 \leq 1,3$ (Barkalov and Węgrzyn, 2006). Now the functions f_1 and f_2 can be expressed as functions depending on $K(\Gamma)$, p_1, p_3, p_4 and N. Some results of investigation are shown in Fig. 10 and 11. Let us point out that these results are correct only if the condition (8) holds. Otherwise some other models of the Moore finite-state machine should be used for the interpretation of a graph-scheme of algorithm Γ. It is clear

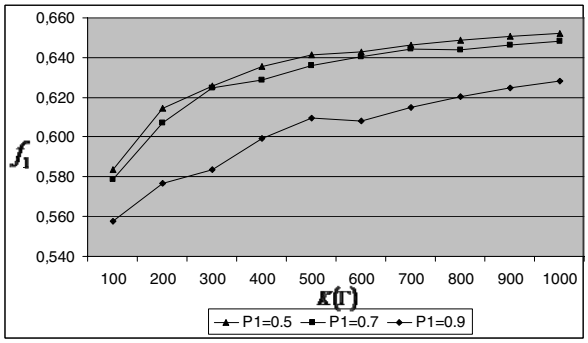

Fig. 10. Function f_1 for $p_2 = 1 - p_1$, $p_3 = 1.3$; $p_4 = 1.2$ and $p_5 = 0.5$.

from Fig. 10 that the application of the proposed method always gives less amount of hardware than the known methods. This gain is increased with a decrease in the number of the vertices of a graph-scheme of the algorithm Γ and an increase the number of operational vertices of graph-schemes of the algorithm Γ (increase in the parameter p_1). The average gain for the graph-scheme of algorithm with $K(\Gamma) = 500$ is equal to 39%. It follows from Fig. 11 that the Moore FSM with the proposed structure always requires less hardware amount than the known models of finite-state machines. This gain is increased with a decrease in the number of microoperations N. The average gain for graph-schemes of the algorithm with $K(\Gamma) = 500$ is near 32%.

From the analysis of these figures it is clear that Moore finite-state machines offer gains in the cost. This gain is increased with reducing the number of vertices in the

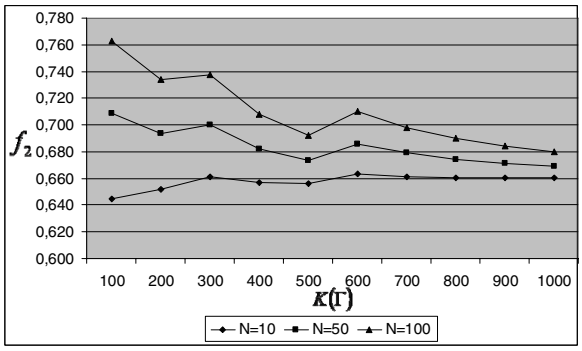

Fig. 11. Function f_2 for $p_1 = p_2 = p_5 = 0.5$, $p_3 = 1.3$ and $p_4 = 1.2$.

initial graph-scheme of algorithm (resp. decreasing the parameter $K(\Gamma)$) and decreasing the length of the codes of sets of microoperations in the initial graph-scheme of algorithm (resp. increasing the parameter P_3). The maximal gain is achieved for graph-schemes of algorithm with the number of vertices $100 \leq K(\Gamma) \leq 200$.

The correctness of these results was checked in the following way for the case of an industrial CPLD with PAL macrocells. Some software was written for the design of all FSM models discussed in this article. This software uses the standard package WebPack of Xilinx (www.xilinx.com) and VHDL models of Moore finite-state machines. A separate program is used to set up the main parameters of embedded memory blocks to estimate their amount and to choose a particular FSM model. Our software permits the estimation of the number of PAL macrocells in the combinational part of the FSM. Experiments conducted with the use of the software confirm the correctness of the tendencies shown in Fig. 10 and 11. But the total average gain was a bit less than it follows from these theoretical curves, and it was equal to, on average, near 28%.

Similar results were obtained for the comparison of the base models $U_1 - U_3$ and the proposed models $U_4 - U_6$.

6. Conclusion

The proposed methods of the implementation of the Moore finite-state machine using PAL macrocells and embedded memory blocks allow decreasing the cost of the logic circuit of the control unit in comparison with the known methods of Moore finite-state-machine design. In this article the proposed methods are based on the following peculiarities of both the Moore finite-state machine and CPLD:

1. Existence of pseudoequivalent states (P_1).
2. Wide fan–in of PAL macrocells (P_2).
3. Existence of the set of fixed numbers for the outputs of the embedded memory block (P_3). Let us remind, that such blocks exist only for our hypothetical CPLD.

There following structures of the logic circuit of Moore finite–state machine are proposed in this article:

1. Moore finite-state machine U_4 based on the properties P_1 and P_3.
2. Moore finite-state machine U_5 based on the optimal encoding of the pseudoequivalent states and properties P_2 and P_3.
3. Moore finite-state machine U_6 based on the optimal encoding of the pseudoequivalent states, the properties P_2 and P_3 and the use of the code transformer.

Each of the proposed methods can be applied only if some conditions hold, which are different for different methods. The choice of a particular method is supported by a special algorithm proposed in this article. Let us point out that these methods cannot be applied in the case of the Mealy finite-state machine, because it has no pseudoequivalent states.

Our analysis of the effectiveness of the proposed methods showed that the method optimal in the given conditions always permits a decrease in the hardware amount in comparison with earlier known methods of Moore finite-state machine design. This decrease in hardware does not lead to a decrease in the performance of the control unit. There are some special cases such as $\Delta_t = 0$ or $\Pi_i = \emptyset$ $(i = B, C, \ldots, G)$, where some other models of the Moore finite-state machine are more effective. These cases are the subject of our further research. The proposed methods can be modified for real CPLD, where embedded memory blocks are absent. In this case the system of microoperations is implemented using PAL macrocells, too. The same effectiveness of the proposed methods should be tested for both cases of the FPGA with embedded memory blocks and for the CPLD CoolRunner (www.xilinx.com) based on the PLA technology. Of course, the proposed methods should be modified to meet specific requirements of these chips.

References

Adamski M. and Barkalov A. (2006): *Architectural and Sequential Synthesis of Digital Devices.* Zielona Góra: University of Zielona Góra Press.

www.altera.com

Baranov S. (1994): *Logic Synthesis for Control Automata.* Boston: Kluwer.

Barkalov A. (1998): *Principles of Optimization of Logical Circuit of Moore FSM*. Cybernetics and system analysis, No. 1, pp. 65–72 (in Russian).

Barkalov A. and Barkalov A. (2005): *Design of Mealy Finite-State-Machines with Transformation of Object Codes.* International Journal of Applied Mathematics and Computer Science, Vol. 15, No. 1, pp. 151–158.

B arkalov A. and Węgrzyn M. (2006): *Design of Control Units with Programmable Logic.* Zielona Góra: University of Zielona Góra Press.

Chattopadhyay S. (2005): *Area Conscious State Assignment with Flip–Flop and Output Polarity Selection for Finite State Machine Synthesis – A Genetic Algorithm Approach,* The Computer Journal, Vol. 48, No. 4, pp. 443–450.

De Micheli G. (1994): *Synthesis and Optimization of Digital Circuits.* New York: McGraw Hill.

Devadas S., Ma H.–K., Newton R., Sangiovanni-Vincentelli A. (1988): *State Assignment of Finite State Machines Targeting Multilevel Logic Implementations,* IEEE Transactions on Computer-Aided Design, pp. 1290–1300.

Kam T., Villa T., Brayton R., Sangiovanni-Vincentelli A. (1998): *Synthesis of Finite State Machines: Functional Optimization,* Boston/London/Dordrecht: Kluwer Academic Publishers.

Kania D. (2004): *Logic Synthesis Oriented on Programmable Logic Devices of the PAL type.* Gliwice: Silesian University of Technology (in Polish).

www.latticesemi.com

Maxfield C. (2004): *The Design Warrior's Guide to FPGA*. NJ: Elsevier.

Mc Cluskey E. (1986): *Logic Design Principles.* Englewood Cliffs: Prentice Hall.

Micheli, G. D., Brayton, R. K. and Vincentelli, A. S. (1985): *Optimal state assignment for finite state machines.* IEEE Transactions on Computer-Aided Design, pp. 269–284.

Villa T., Kam T., Brayton R., Sangiovanni-Vincentelli A. (1998): *Synthesis of Finite State Machines: Logic Optimization,* Kluwer Academic Publishers, Boston/London/Dordrecht.

Villa T., Sangiovanni–Vincentelli A. (1998): *State Assignment of Finite State Machines for Optimal Two-Level Logic Implementation,* IEEE Transactions on Computer-Aided Design, pp. 905–924.

Xia, Y. and Almaini, A. (2002): *Genetic algorithm based state assignment for power and area optimization,* IEEP Comput. Dig. T., Vol. 149, No. 4, pp. 128–133.

www.xilinx.com

Yang S. (1991): *Logic Synthesis and Optimization Benchmarks User Guide,* Microelectronics Center of North Carolina, Research Triangle Park, North Carolina.

Received: 6 March 2007
Revised: 30 August 2007
Re-revised: 27 October 2007