UDC 621.391

VLSI-BASED SYNTHESIS OF MOORE FINITE-STATE-MACHINES TARGETING TELECOMMUNICATIONS SYSTEMS

A. BARKALOV
*University of Zielona Góra,
Institute of Computer Engineering and Electronics*
L. TITARENKO
Kharkiv National University of Radio Electronics, University of Zielona Góra, Institute of Computer Engineering and Electronics
S. CHMIELEWSKI
*University of Zielona Góra,
Institute of Computer Engineering and Electronics*

Abstract *– The optimization methods of the logic circuit of Moore finite-state-machine are proposed. These methods are based on the existence of pseudo equivalent states of Moore finite-state-machine, wide fan-in of PAL macrocells, and free resources of embedded memory blocks. The methods are oriented on hypothetical VLSI microcircuits based on CPLD technology and containing PAL macrocells and embedded memory blocks. The conditions of effective application of each proposed method are shown. An algorithm of choice of the best model of finite-state-machine for given conditions is proposed. The examples of proposed methods application are given. The effectiveness of the proposed methods is also investigated. The analysis of the effectiveness of proposed methods showed that optimal in the given conditions method always permits a decrease of the hardware amount in comparison with earlier known methods of the Moore finite-state-machine design. This decrease in hardware does not lead to a decrease in the performance of the control unit. Moreover, there are some special cases, when some other models of Moore finite-state-machine are more effective. The proposed methods can be modified for real CPLDs, where embedded memory blocks are absent. In this case, the system of microoperations is implemented using PAL macrocells too. The same effectiveness of proposed methods should be tested for both cases of FPGA with embedded memory blocks and for CPLD CoolRunner based on PLA technology. The proposed methods should be modified to meet the specific requirements of these chips.*

Анотація *– Запропоновано методи оптимізації логічної схеми кінцевого автомата Мура. Ці методи засновані на існуванні псевдоеквівалентних станів кінцевого автомата Мура, широкому коефіцієнту об'єднання по входу макроелементів PAL та вільних ресурсах вбудованих блоків пам'яті. Методи орієнтовані на гіпотетичні мікросхеми VLSI на основі технології CPLD і містять макроелементи PAL та вбудовані блоки пам'яті. Показано умови ефективного застосування кожного запропонованого методу. Запропоновано алгоритм вибору найкращої моделі кінцевого автомата для даних умов. Наведені приклади застосування запропонованих методів. Ефективність запропонованих методів також було досліджено. Аналіз ефективності запропонованих методів показав, що оптимальний у даних умовах метод завжди дозволяє зменшити апаратну кількість порівняно з раніше відомими методами проєктування кінцевого автомата Мура. Це зменшення апаратних засобів не призводить до зниження продуктивності блоку управління. Більше того, існують особливі випадки, коли деякі інші моделі кінцевого автомата Мура є більш ефективними. Запропоновані методи можуть бути модифіковані для реальних CPLD, де вбудовані блоки пам'яті відсутні. У цьому випадку система мікрооперацій також реалізована з використанням макроелементів PAL. Слід перевірити однакову ефективність запропонованих методів як для випадків FPGA із вбудованими блоками пам'яті, так і для CPLD CoolRunner на основі технології PLA. Запропоновані методи необхідно модифікувати відповідно до конкретних вимог цих мікросхем.*

Introduction

Practically any modern telecommunications system includes a control unit [1]. It is a "brain" of the system responsible for interplay of all its blocks. Very often, a model of Moore finite state machine (FSM) is used for representing the control unit [2]. Nowadays, the field programmable logic devices (CPLDs) are mostly used for implementing logic circuits of FSMs [3,4] Mostly, FPGAs are used for implementing complex FSMs, whereas CPLDs target rather fast FSMs [5,6,7].

The majority of CPLDs are based of PAL-cells connected with programmable flip-flops [8,9]. The generalized structure of PAL-based cell is represented in many works [10,11]. There are two specifics of PAL-cells which should be used under the development of design methods. First of all, PAL-cells have a limited number of product terms, q. In most cases q=5 [11]. It means that logic functions having more than 5 product terms should be implemented using more than one cell. It leads to decomposition of Boolean functions representing an FSM circuit [12,13,14]. In turn, it increases the propagation time of FSM. So, it is very important to diminish the numbers of terms in functions to be implemented. The second, specific of PAL-cells is a considerable amount of inputs (up to 30) [8,9]. In this article, we show how to use this peculiarity. One of the most important steps of FSM synthesis is the state assignment [1]. During this set, each internal state of FSM is replaced by a binary code.

A proper state assignment can decrease the number of PAL-cells in the FSM logic circuit [15] or reduce the power consumption [16]. Let us point out that these very goals can be achieved due to implementing the system of microoperations using embedded memory blocks (EMB) [6]. But such blocks can be found only in CPLD Delta 39K [17].

The peculiarity of EMBs is their ability for tuning. It means that they have the constant size, whereas both numbers of cells and outputs can be different [6]. As for delta 39K, the blocks can be configured as 2048x1, 1024x2, 512x4, or 256x8 devices [17]. Nowadays, there are no methods targeting CPLDs with EMB in the respect to FSM synthesis. The only exception is our article [18]. We think that these methods should be developed, because other families of CPLDs with EMBs can be produced.

The proposed methods discussed in this article are based on the classes of pseudoequivalent states (PS) of Moore FSM [19]. The wide fan-in of PAL-cells allows using more than one source for representing codes of PS [18]. So, in this article three main specifics are used for optimizing the logic circuit of CPLD-based Moore FSM: the pseudoequivalent states of FSM; the wide fan-in of PAL-macrocells the configurability of EMBs.

In this article, we deal with hypothetic CPLD chips having embedded memory blocks. The control algorithm of a digital system is represented by a graph-scheme of algorithm (GSA) [2].

1. Background of Moore FSM design

Let control algorithm of digital system be specified by graph-scheme of algorithm [2] $\Gamma = (B, E)$, where $B = \{b_0, b_E\} \cup E_1 \cup E_2$ is a set of the vertices and E is set of the edges. Here b_0 is initial vertex, b_E is final vertex, E_1 is a set of operational vertices, 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 $Y = \{y_1, ..., y_N\}$ is a set of microoperations of data-path [1] of digital system. The vertex $b_q \in E_2$ contains some logic condition $x_e \in X$, where $X = \{x_1, ..., x_L\}$ is a set of logic conditions (flags) [6]. The initial and final vertices of graph-scheme of algorithm correspond to initial state $a_1 \in A$, where $A = \{a_1, ..., a_M\}$ is a set of internal states of Moore finite-state-machine. Each operational vertex $b_q \in E_1$ corresponds to unique state $a_m \in A$. The logic circuit of Moore finite-state-machine U_1 is represented by the following systems of Boolean functions:

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

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

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

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

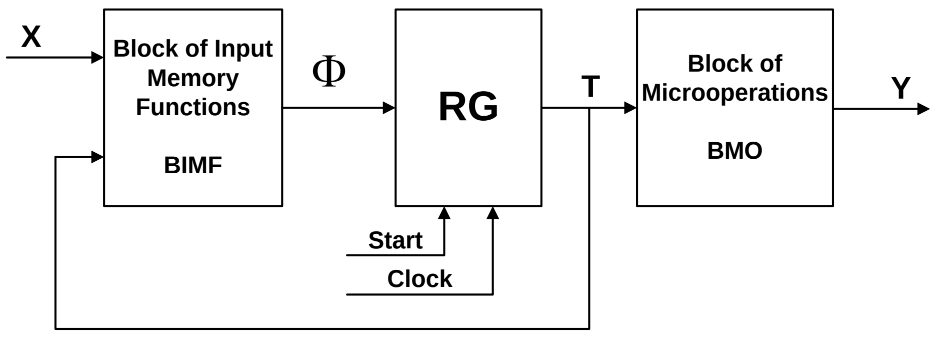

Fig. 1. Structure diagram of Moore FSM U1

As a rule, the number of transitions $H_1(\Gamma)$ exceeds number of transitions $H_0(\Gamma)$ of an equivalent Mealy finite-state-machine [6]. It leads to an increase of the number of PAL macrocells in the circuit of Moore finite-state-machine in comparison with the equivalent

Mealy finite-state-machine. The value $H_1(\Gamma)$ can be decreased taking into account the pseudoequivalent states of Moore finite-state-machine [19]. The states $a_m, a_s \in A$ are pseudoequivalent states, if identical inputs result in identical next states for both $a_m, a_s \in A$. It is possible, if outputs of operational vertices marked by these states are connected with the input of the same vertex of graph-scheme of algorithm Γ. Let $\Pi_A = \{B_1, \dots, B_I\}$ be a partition of the set A by the classes of pseudoequivalent states $(I \le M)$.There are two main methods of Moore finite-state-machine optimization based on pseudoequivalent states [19]: 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 in such a manner that codes of the states $a_m \in B_i$ $(i = 1, \dots, I)$ belong to single generalized interval of R-dimensional Boolean space. It leads to Moore finite-state-machine U_2 that has the same structure as Moore finite-state-machine U_1. The algorithm from work [1] can be used for such encoding. It is shown in the work [19] that the number of transitions $H_2(\Gamma)$ of U_2 is decreased up to $H_0(\Gamma)$. But such encoding is not always possible [6].

In the second case the classes $B_i \in \Pi_A$ are encoded by binary codes $K(B_i)$ with $R_1 =]\log_2 I[$ bits. The variables $\tau_r \in \tau$ are used for such 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 finite-state-machine. This approach leads to the Moore finite-state-machine U_3 with BCT (Fig.2).

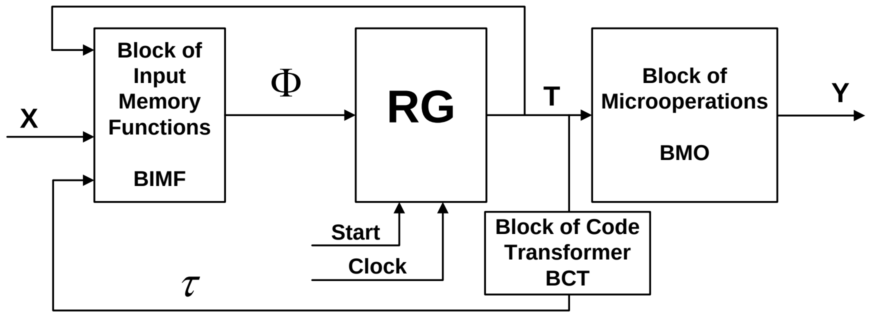

Fig. 2. Structure diagram of Moore FSM U3

In the Moore finite-state-machine U_3 BIMF implements the functions

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

and block of code transformer (BCT) implements the functions

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

The number of transitions of Moore finite-state-machine U_3 is equal to $H_0(\Gamma)$. The drawback of U_3 is existence of block of 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 optimal encoding of the states and transformation of the codes of the states. In this case block of code transformer can be even eliminated if some condition holds. The proposed method is based on the following features of 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 embedded memory block can be chosen from some restricted area.

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

2. Main ideas of proposed method

Let embedded memory block have q words if the number of its outputs $t_F = 1$. If $q \geq M$, then 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 outputs t_F is chosen from 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 BMO is determined as

$$t_s = \left]N / t_F\right[\cdot t_F. \tag{6}$$

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 the condition

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

holds, then graph-scheme of algorithm Γ can be interpreted by Moore finite-state-machine U_4 (Fig. 3).

In the Moore finite-state-machine U_4 BIMF forms functions (3) and BMO and codes of the classes BMO implements both the systems (2) and (4). In this case block code transformer is eliminated and states of finite-state-machine can be encoded in an arbitrary manner.

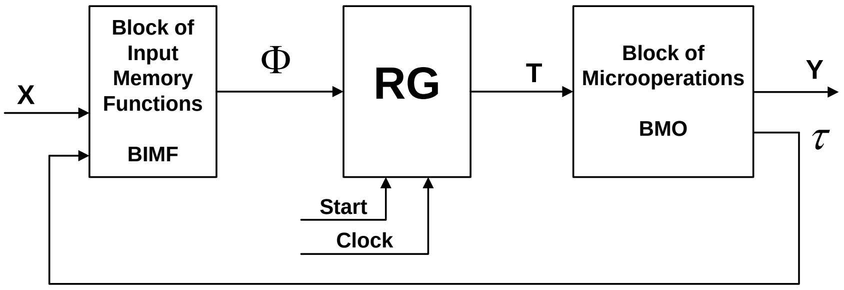

Fig. 3. Structure diagram of Moore FSM U4

If condition (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 $B_i \in \Pi_B$, if condition

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

holds, otherwise $B_i \in \Pi_C$.
It is clear that circuit of 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 the optimal way [19] and let us represent the set Π_B as $\Pi_B = \Pi_D \cup \Pi_E$. Here $B_i \in \Pi_D$, if codes of the states $a_m \in B_i$ belong to single generalized interval of Boolean space. Now only codes of the states $a_m \in A(\Pi_E)$ should be transformed where $A(\Pi_j)$ is a set of the states $a_m \in B_i$, where $B_i \in \Pi_j\,(j = A,B,C,D,E)$. It is enough $R_2 = \left]\log_2(|\Pi_E|+1)\right[$ binary variables to encode the classes $B_i \in \Pi_E$. Let these variables form the set Z, where $|Z| = R_2$.

If the condition

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

holds, then graph-scheme of algorithm Γ can be interpreted by the Moore finite-state-machine U_5 (Fig. 4).

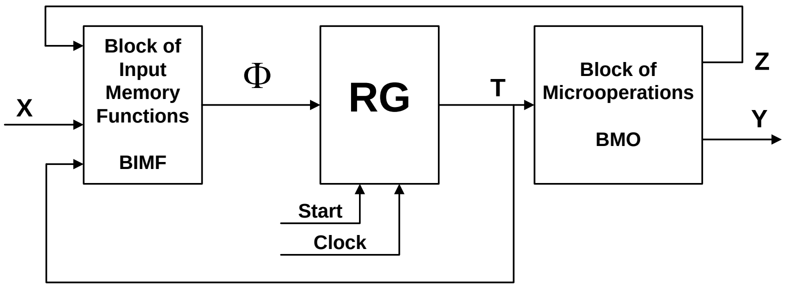

Fig. 4. Structure diagram of Moore FSM U5

Here BIMF forms functions

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

circuit BMO forms both functions (2) and functions

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

In the finite-state-machine U_5 the block of code transformer is absent and 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 circuit BMO.

In this case the number of inputs in the PAL macrocells is increased from $L + R_1$ (finite-state-machine U_3) to $L + R + R_2$ (finite-state-machine U_5), but it does not increase the hardware amount in the BIMF in comparison with finite-state-machine U_3. The cycle time of both U_1 and U_5 is the same in the worst case. In the best case the BIMF of U_5 has less amount of levels than BIMF of U_1. It means that the time of cycle of U_5 can be less than the time of cycle of U_1. Therefore, the proposed approach permits to decrease the hardware amount without a decrease of performance of digital system. Let us point out that the cycle times of U_2, U_3, U_4, U_5 are the same.

If conditions (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 circuit BMO 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 the Moore finite-state-machine U_6 (Fig.5).

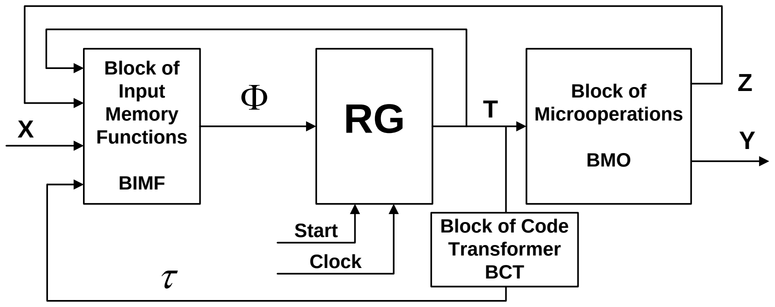

Fig. 5. Structure diagram of Moore FSM U6

Here BIMF forms the functions

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

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

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

- Construction of the marked graph-scheme of algorithm Γ and construction of the set of internal states $A=\{a_1,...,a_M\}$ of the Moore finite-state-machine.
- Construction of the partition $\Pi_A=\Pi_B\cup\Pi_C$.
- Optimal encoding of the states and construction of the sets Π_D and Π_E.
- Calculation of Δt and construction of the sets Π_F and Π_G.
- Encoding of the classes $B_i\in\Pi_F\cup\Pi_G$.
- Construction of the table of circuit BMO.
- Construction of the modified structure table of finite-state-machine.
- Construction of the table of code transformer
- Implementation of the logic circuit of finite-state-machine.

The choice of particular model depends on some conditions. In this article we propose the following algorithm of such choice (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 Boolean space ($\Pi_E=R$), then the model U_5 should be chosen. If $\Delta_t<R_1$ and ($\Pi_E=R$), then condition (10) determines the optimal model of Moore finite-state-machine for interpretation of graph-scheme of algorithm Γ using the hardware of a SoPC with the CPLD technology.

Conclusions

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

- Existence of the pseudoequivalent states (P_1).
- Wide fan-in of the PAL macrocells (P_2).

Existence of the set of fixed numbers for outputs of embedded memory block (P_3) remind that such blocks exist only for our hypothetical CPLD.

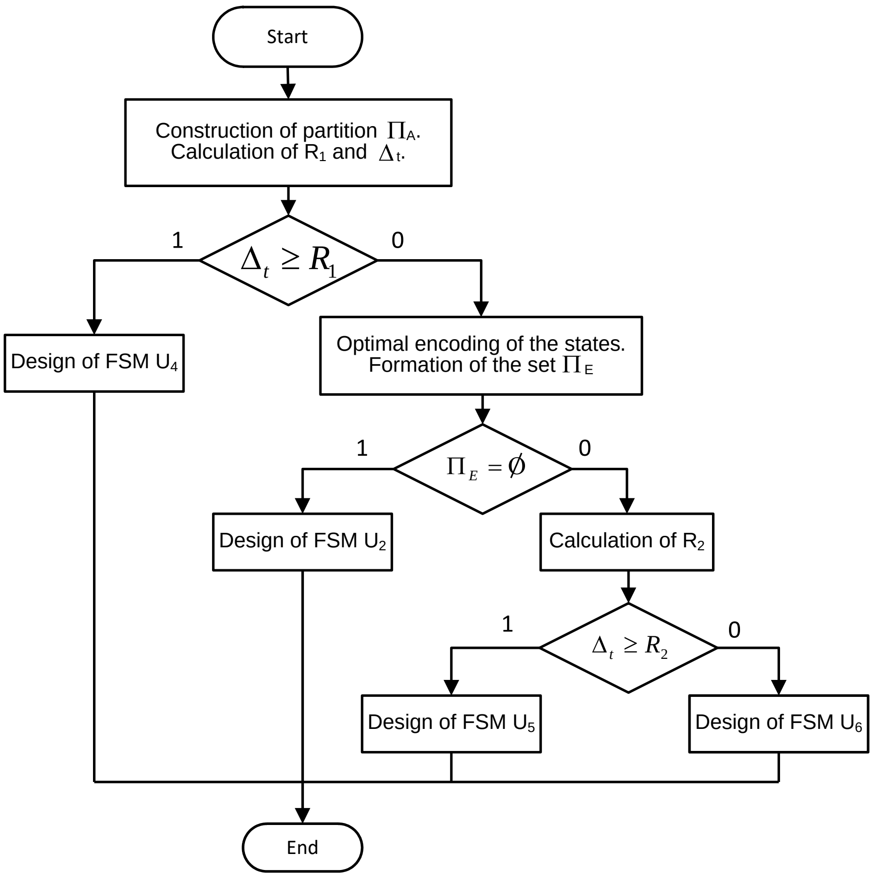

Fig. 6. Choice of the Moore FSM model

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

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

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

Our analysis of effectiveness of proposed methods showed that optimal in the given conditions method always permits decrease of the hardware amount in comparison with earlier known methods of the Moore finite-state-machine design. This decrease of hardware does not lead to a decrease of the performance of the control unit. There are some special cases such as $\Delta_t = 0$ or $\Pi_i = \varnothing$ $(i = B, C, \ldots, G)$, where some other models of Moore finite-state-machine are more effective. These cases are the subjects of our further research. The proposed methods can be modified for real CPLDs, where embedded memory blocks are absent. In this case the system of microoperations is implemented using PAL macrocells too. The same effectiveness of proposed methods should be tested for both cases of FPGA with embedded memory blocks and for CPLD CoolRunner [9] based on PLA technology. Of course, the proposed methods should be modified to meet specific requirements of these chips.

References:

1. *Popovskij, V., Barkalov, A., Titarenko, L.* (2011), Control and adaptation in telecommunication systems: Mathematical foundations, Berlin: Springer, 176 p.

2. *Baranov, S.* (1994), Logic Synthesis for Control Automata, Springer Science & Business Media, 404 p.

3. *Maxfield, C.* (2004), The Design Warrior's Guide to FPGA, NJ: Elsevier, 560 p.

4. *Grout, I.* (2008), Digital Systems Design with FPGAs and CPLDs, Oxford: Elsevier, 784 p.

5. *Anderson, J., Brown, S.* (1998), "Technology mapping for large complex PLDs", Proceedings 1998 Design and Automation Conference. 35th DAC. (Cat. No.98CH36175), P. 698-703. **DOI**: https://doi.org/10.1145/277044.277220

6. *Barkalov, A., Titarenko, L., Mielcarek, K., Chmielewski, S.* (2020), Logic Synthesis for FPGA-Based Control Units, Springer International Publishing, 257 p.

7. *Kania, D.* (2004), Logic Synthesis Oriented on Programmable Logic Devices of the PAL type, Scientific Notebooks of Silesian Univ. Technology 1619, CD-ROM (in Polish).

8. *Kania, D.* (2015), "Logic decomposition for PAL-based CPLDs", Journal of Circuits, Systems and Computers, No. 24(03), p.1550042. **DOI**: https://doi.org/10.1142/S0218126615500425

9. *Kallaher, B.* "PAL vs. CPLD vs. FPGA", Digilent blog, available at: https://blog.digilentinc.com/pal-vs-cpld-vs-fpga/

10. *Salauyou, V., Klimowicz, A.* (2010), Logic Synthesis for Digital Devices with programmable structures, Bialystok: OWPB, 480 p.

11. *Kania, D., Czerwinski, R.* (2012), "Area and Speed oriented synthesis of FSMs for PAL-based CPLDs", Microprocessors and Microsystems, No. 36(1), P. 45-61. **DOI**: https://doi.org/10.1016/j.micpro.2011.06.004

12. *Czerwinski, R., Kania, D.* (2009), "Synthesis of FSMs for CPLDs", International Journal of Applied Mathematics and Computer Science, No. 9(4), P. 647-659. **DOI**: https://doi.org/10.2478/v10006-009-0052-0

13. *Opara, A., Kania, D.* (2010), "Decomposition – based logic synthesis for PAL-Based CPLDs", International Journal of Applied Mathematics and Computer Science, No. 20(2), P. 367-384. **DOI**: https://doi.org/10.2478/v10006-010-0027-1

14. *Kania, D., Milek, A.* (2010), "Logic Synthesis Based on decomposition for CPLDs", Microprocessors and microsystems, No.34(1), P. 25-38. **DOI**: https://doi.org/10.1016/j.micpro.2009.11.002

15. *Villa, T., Sangiovanni-Vincentelli, A.* (1990), "NOVA: State Assignment of Finite State Machines for Optimal Two-Level Logic Implementation", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, No.9(9), P. 905-924. DOI: https://doi.org/10.1109/43.59068

16. *Salauyou, V., Grzes, T.* (2007), "FSM state assignment methods for low-power design", Proceedings of the 6th International Conference on Computer Information Systems and Industrial Management Applications (CISIM'07), P. 345-350. **DOI**: https://doi.org/10.1109/CISIM.2007.32

17. Cypress Semiconductor Product Qualification Report (2002), available at: https://www.cypress.com/file/39936/download

18. *Barkalov, A., Titarenko, L., Chmielewski, S.* (2007), "Reduction in the number of PAL macrocells in the circuit of a Moore FSM", International Journal of Applied Mathematics and Computer Science, No. 17(4), P. 565-575. **DOI**: https://doi.org/10.2478/v10006-007-0046-8

19. *Barkalov, A.* (1998), "Principles of logic optimization for Moore microprogrammed automaton", Cybernetics and System Analysis, No. 34(1), P. 54-60. **DOI**: https://doi.org/10.1007/BF02911262

20. *Yang, S.* (1991), Logic Synthesis and Optimization Benchmarks User Guide. Microelectronics Center of North Carolina, Research Triangle Park, North Carolina, 45 p.