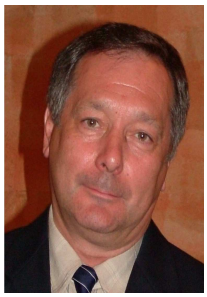
Alexander BARKALOV, Larysa TITARENKO, Olena HEBDA
UNIVERSITY OF ZIELONA GORA , Podgórna 50, 65-256, Zielona Góra

Synthesis of Moore FSM with encoding of collections of microoperations implemented with ASIC

Prof. dr hab. inż. Alexander BARKALOV, prof. UZ



Prof. Alexander A. Barkalov worked in Donetsk National Technical University (DNTU) from 1976 till 1996 as a tutor . He cooperated actively with Kiev Institute of Cybernetics (IC) named after Victor Glushkov. He got his degree of doctor of technical sciences (Informatics) in 1995 from IC. From 1996 till 2003 he worked as a professor of DNTU. From 2003 he has been working as a professor on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora.

e-mail: A.Barkalov@iie.uz.zgora.pl

Dr hab. inż. Larysa TITARENKO, prof. UZ

Prof. Larysa Titarenko has got her degree of doctor of technical sciences (Telecommunications) in 2005 from Kharkov National University of Radioelectronics (KNURE). Till September, 2003 she worked as a professor of KNURE. From 2005 she has been working as a professor on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora.

e-mail: L.Titarenko@iie.uz.zgora.pl

MA Olena Hebda

MA Olena Hebda studied at National aerospace university 'Kharkiv Aviation Institute' from 2003 till 2009 and has received higher education on speciality "Biotechnical and Medical Apparatuses and Systems" and qualification of the research engineer (electronics, telecommunications). In 2009 she has started PhD study on Department of Electrotechnics, Informatics and Telecommunications of University of Zielona Gora.

e-mail: O.Shapoval@weit.uz.zgora.pl

Abstract

The method is proposed for reduction of hardware amount in logic circuit of Moore finite state machine. The method is oriented on customized matrix technology. It is based on representation of the next state code as a concatenation of code for class of collection of microoperations and code of the vertex. Such an approach allows elimination of dependence among states and microoperations. As a result, both circuits for generation of input memory functions and microoperations are optimized.

Keywords:Moore FSM, graph-scheme of algorithm, pseudoequivalent states, customized matrices, logic circuit.

Syneza skończonego automatu stanu typu Moore'a z kodowaniem zbiorów mikrooperacji implementowanego w układach o strukturze matrycowej

Streszczenie

Zaproponowana metoda jest zorientowana na redukcję zasobów sprzętowych potrzebnych do implementacji skończonego automatu stanu typu Moore'a implementowanego w układach o strukturze matrycowej. W przypadku produkcji masowej szeroko stosowane są układy ASIC (ang. Application-Specified Integrated Circuits). W układach ASIC automaty skończone są projektowane przy użyciu struktur matrycowych. Jednym z głównych problemów syntezy automatów skończonych ze strukturami matrycowymi jest zmniejszenie powierzchni układu scalonego zajmowanej przez układ logiczny automatu Moore'a. W artykule proponowana jest metoda, która jest ukierunkowana na redukcję zasobów sprzętowych potrzebnych do implementacji skończonego automatu stanu typu Moore'a implementowanego w układach o strukturze matrycowej. Ta metoda jest oparta na przedstawieniu następnego kodu stanu jako konkatenacji kodu klas zbiorów mikrooperacji i kodów wierzchołków.

Słowa kluczowe: automat typu Moore'a, sieć działań, stany pseudorównoważne, układ logiczny.

1. Introduction

The model of Moore finite state machine (FSM) [1] is often used during the digital control systems realization [2, 3]. The development of microelectronics has led to appearance of different programmable logic devices [4,5], used for implementing FSM circuits. But in the case of mass production, they use ASIC (Application-Specified Integrated Circuits) [6]. In this case the circuit is implemented using customized matrices using the principle of distributed logic [7].

One of the important problems of FSM synthesis with ASIC is decrease of the chip area occupied by its logic circuit. One of the ways to solve this problem is optimal coding of FSM [2]. However this approach does not allow optimization of the circuit generated output signals. In this work some new optimization method is proposed.

2. The general aspects and the basic idea of proposed method

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 an initial state of FSM; $K(a_m)$ is a code of state $a_m \in A$ of capacity $R = \lceil \log_2 M \rceil$, to code the states the internal variables from the set $T = \{T_1,..,T_R\}$ are used; a_s, $K(a_s)$, are a state of transition and its code respectively; X_h is an input, which determines the transition $\langle a_m, a_s \rangle$, and equal to conjunction of some elements (or their complements) of a logic conditions set $X = \{x_1,..,x_L\}$; Φ_h is a set of input memory functions for flip-flops of FSM memory, which are equal to 1 for memory switching from $K(a_m)$ to $K(a_s)$, $\Phi_h \subseteq \Phi = \{\varphi_1,..,\varphi_R\}$; $h = 1, \ldots, H$ is a number of transition. In the column a_m a set of microoperations Y_q is written, which is generated in the state $a_m \in A$, where $Y_q \subseteq Y = \{y_1,..,y_N\}$, $q = 1,...,Q$. This table is a basis to form the system of functions.

$$\Phi = \Phi(T, X), \quad (1)$$

$$Y = Y(T), \quad (2)$$

which determines an FSM logic circuit. Systems (1)-(2) describe the matrix model of Moore FSM U_1, shown in Fig.1.

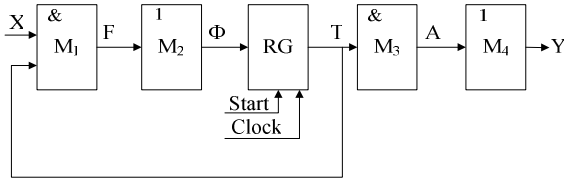


Fig. 1. Matrix implementation of FSM U_1

Rys. 1. Matrycowy układ automatu Moore'a U_1

One of Moore FSM features is existence of pseudoequivalent states [2], which are the states with the same transitions by the effect of the same inputs. Such states correspond to the control algorithm operator vertices [1], outputs of which are connected with an input of the same vertex.

Let $\Pi_A = \{B_1,..,B_I\}$ be a partition of a set A on classes of pseudoequivalent states. Let us code classes $B_i \in \Pi_A$ by binary codes $K(B_i)$ having $R_B = \lceil \log_2 I \rceil$ bits.

Let initial GSA Γ include Q different collections of microoperations (CMO) $Y_q \subseteq Y$. Let us code set Y_q with binary code $K(Y_q)$ having $R_Y = \lceil \log_2 Q \rceil$ bits.

Let $E_1 = \{b_1,\ldots,b_D\}$ be a set of operator vertices from GSA Γ. Let us use the following relation α on this set E_1

$$b_i \alpha b_j \leftrightarrow Y(b_i) = Y(b_j). \quad (3)$$

In (3), the symbols $Y(b_i), Y(b_j) \subseteq Y$ stand for collections of MO from vertices b_i and b_j $(i, j \in \{1,...,D\})$.The relation α determines the partition $\Pi_\alpha = \{C_1,...,C_\eta\}$. Let us encode each vertex $b_q \in C_j$ by the binary code $K(b_q)$ having $R_\alpha = \lceil \log_2 G \rceil$ bits. In (6), $G = \max(|C_1|,...,|C_\eta|)$. Let us use variables $z_r \in Z_1$ for this encoding, where $|Z_1| = R_\alpha$. In this case, the code for state $a_m \in A$ can be represented as:

$$K(a_m) = K(Y_q) * K(b_q), \quad (4)$$

where $b_q \in E_1$ is the operator vertex marked by state $a_m \in A$, $Y_q = Y(b_q)$, and * is the sign of concatenation.

Let us construct the system

$$B = B(A), \quad (5)$$

which describes the dependence among the classes $B_i \in \Pi_A$ from the states $a_m \in A$. Each function $B_i \in B$ is represented as the following

$$B_i = \bigvee_{i=1}^{I} C_{im} A_m \, (i = 1,...,I), \quad (6)$$

where the symbol C_{im} stands for Boolean variable equal to 1, $a_m \in B_i$. The proposed matrix implementation of Moore FSM U_2 is shown in Fig. 2.

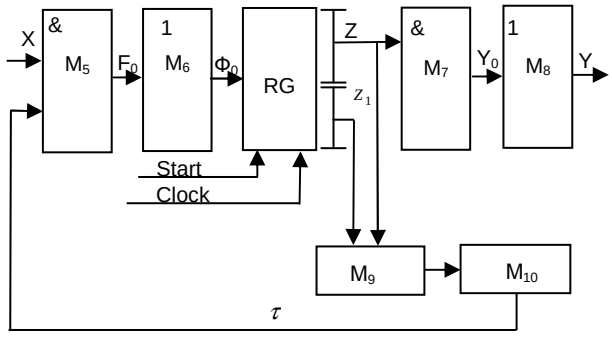


Fig. 1. Matrix implementation of FSM U_2

Rys. 1. Matrycowy układ automatu Moore'a U_2

In FSM U_2, the matrix M_5 implements the system of terms F_0 corresponding to rows of transformed table of transitions and depending on logical conditions $x_l \in X$ and additional variables $\tau_r \in \tau$, used for encoding the classes $B_i \in \Pi_A$, where $|\tau| = R_B$. The matrix M_6 implements the input memory functions

$$\Phi_0 = \Phi_0(\tau, X), \quad (7)$$

The system (10) includes $R_Y + R_\alpha$ functions; it is the number of flip-flops from RG. The matrix M_7 implements terms Y_0, entering the system $y_n \in Y$ and depending from variables $z_r \in Z$, where $|z| = R_Y$. The matrix M_8 implements functions $y_n \in Y$, depending on terms $\Delta_q \in Y_0$. The matrix M_9 implements the terms A_0 from (9), whereas the matrix M_{10} functions $\tau_r \in \tau$, used for encoding classes $B_i \in \Pi_A$, where $|\tau| = R_B$.

Matrices M_5 and M_6 form the block BIMF, the matrices M_7 and M_8 form the block BMO implementing the functions

$$Y = Y(Z). \quad (8)$$

Matrices M_9 and M_{10} form the block of code transformer (BCT) generating functions

$$\tau = \tau(z, z_1). \quad (12)$$

There are some positive features in the proposed method. Now codes of collections of microoperations do not depend on state codes. It allows encoding of collections $Y_q \subseteq Y$ minimizing the area of BMO. The number of rows in the table of transitions for FSM U_2 is always equal to H_0. It allows such their encoding that diminishes the area occupied by BIMF. As it was mentioned, it is enough

$$R_A = \lceil \log_2 M \rceil \quad (13)$$

variables for state encoding in case of FSM U_1. The main drawback of U_2 is increase of the number of inputs for BIMF if the following condition is true:

$$R_Y + R_\alpha > R_A. \quad (14)$$

Besides, the model U_2 includes the block BCT, which requires some area of the chip. But these drawbacks are compensated by area decrease for blocks BIMF and BMO in comparison with the model U_1.

3. Conclusion

The proposed method of state code presentation targets on area decrease under implementation of Moore FSM logic circuit with customized matrices. This approach allows decreasing the number of terms in the system of input memory functions up to corresponding value of the equivalent Mealy FSM. Besides, this method permits decreasing the number of terms in the system of microoperations due to the lack of dependence among the state codes and codes of collections of microoperations.

Investigation for effectiveness of proposed method was conducted on the standard examples [7]. It shows that the proposed method permits to decrease the average chip area occupied by FSM circuit up to 52% in comparison with the standard FSM implementation. In the same time, it was the increase for FSM performance in 86% of examples. The further direction of our research is application of proposed method for case of FPGA.

4. Literatura

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

[2]] A. Barkalov, L. Titarenko, Logic Synthesis for FSM-Based Control Units. Springer - Berlin Verlag Heidelberg, Lectures Notes in Electrical Engineering, 2009, №53.

[3] DeMicheli G. Synthesis and Optimization of Digital Circuits, McGraw-Hill, NY, 1994.

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

[5] Smith M. Application-Specific Integrated Circuits, Addison-Wesley, Boston, 1997.

[6] Nababi Z. Emvedded Core Design with FPGA, McGraw-Hill, NY, 2008.

[7] Yang S. Logic Synthesis and Optimization Benchmarks user guide. Technical report, №1991 – IWLS-UG-Saryang.-Microelectronics center of North Carolina.