Двойное кодирование состояний в совмещенном автомате

А.А. Баркалов[1], Л.А. Титаренко[2], Я.Е. Визор[3], А.В. Матвиенко[4], С.А. Сабурова[5]

[1] – Institute of Informatics and Electronics. University of Zielona Gora, ul. Podgorna, 50, Zielona Gora, 65-246, POLAND, A.Barkalov@iie.uz.zgora.pl
[2] – Institute of Informatics and Electronics. University of Zielona Gora, ul. Podgorna, 50, Zielona Gora, 65-246, POLAND; ХНУРЭ, 61166, г. Харьков, проспект Науки, 14, L.Titarenko@iie.uz.zgora.pl,
[3, 4] – Институт кибернетики имени В.М. Глушкова НАН Украины, 03187, г. Киев, проспект Академика Глушкова, 40, yaviz@ukr.net, matv@online.ua,
[5] – ХНУРЭ, 61166, г. Харьков, проспект Науки, 14, sabsvet@gmail.com

A.A. Barkalov, L.A. Titarenko, Y.E. Vizor, A.V. Matvienko, S.A. Saburova

TWO FOLD STATE ASSIGNMENT FOR COMBINED AUTOMATION

Abstract. *The article proposes a method of reducing hardware costs in the development of control devices for digital systems. Reducing hardware costs can improve the quality of a digital system by reducing the size of the VLSI chip, reducing energy consumption and increasing speed. The methods for solving this problem largely depend on the model representing the control device and the characteristics of the elemental basis in which the circuit is implemented. The proposed method uses the model of combined microprogrammed automaton (CMPA) implemented in the FPGA (field-programmable logic arrays) basis. To set the behavior of the automaton, the language of the graph diagrams of the algorithm is used. A characteristic feature of the SMPA model is the presence of two types of output signals. The output signals of the Mealy machine exist during transitions between the states of the machine. The outputs of the Moore automaton are determined only by the states of the automaton. The main elements of FPGA that are used to implement CMPA are elements of the table type LUT (look-up table), programmable triggers and built-in memory blocks EMB (embedded memory blocks). EMBs are heavily used to implement the operational devices of digital systems. The proposed method allows to use only LUT elements in the design of the control device. The method is based on splitting the set of states of an automaton into classes, each of which corresponds to a separate block of the circuit. This approach leads to circuit with a regular structure having three levels of logic. The article provides an example of the synthesis of an automaton circuit using the proposed method. The conditions of its application are shown.*
Key words: *combined microprogrammed automaton, synthesis, FPGA, LUT, partitioning.*

Аннотация. *Предложен метод уменьшения аппаратурных затрат в схеме совмещенного микропрограммного автомата, реализуемого в базисе FPGA. Метод основан на разбиении множества состояний на классы, каждый из которых соответствует отдельному блоку схемы. Такой подход приводит к схемам с регулярной структурой и тремя логическими уровнями. Приведен пример синтеза схемы автомата с использованием предложенного метода. Показаны условия его применения.*
Ключевые слова*: совмещенный микропрограммный автомат, синтез, FPGA, LUT, разбиение.*

Анотація. *У статті запропоновано метод зменшення апаратурних витрат при розробці пристроїв управління цифрових систем. Зниження витрат апаратури дозволяє підвищити якість цифрової системи за рахунок зменшення площі кристала НВІС, зниження споживання енергії та підвищення швидкодії. Методи вирішення цього завдання в значній мірі залежать від моделі, що представляє пристрій управління, і характеристик елементного базису, в якому реалізується схема. Запропонований метод використовує модель суміщеного мікропрограмного автомата (СМПА), що реалізується в базисі FPGA (field-programmable logic arrays). Для завдання поведінки автомата використовується мова граф-схем алгоритму. Характерною рисою моделі СМПА є наявність двох типів вихідних сигналів. Вихідні сигнали автомата Мілі існують при переходах між станами автомата. Вихідні сигнали автомата Мура визначаються тільки станами автомата. Основними елементами FPGA, які використовуються для реалізації СМПА, є елементи табличного типу LUT (look-up table), програмовані тригери і вбудовані блоки пам'яті EMB (embedded memory blocks). Блоки EMB інтенсивно використовуються для реалізації операційних пристроїв цифрових систем. Пропонований метод дозволяє застосовувати при проектуванні пристрою управління тільки елементи LUT. Метод заснований на розбитті множини станів автомата на класи, кожен з яких відповідає окремому блоку схеми. Такий підхід призводить до схем з регулярною структурою і трьома логічними рівнями. У статті наведено приклад синтезу схеми автомата з використанням запропонованого методу. Показані умови його застосування.*
Ключові слова: *суміщений мікропрограмний автомат, синтез, FPGA, LUT, розбиття.*

Введение. Одним из главных блоков цифровых систем является устройство управления (УУ) [1, 2]. Для повышения качества цифровых систем (уменьшение площади кристалла, снижение потребления энергии, повышение быстродействия) необходимо уменьшать площадь кристалла СБИС, занимаемую схемой УУ [3, 4]. Методы решения этой задачи в значительной степени зависят от модели, представляющей УУ, и характеристик элементного базиса, в котором реализуется схема УУ [5].

В настоящее время модели микропрограммных автоматов (МПА) широко используются для задания поведения УУ [6, 7]. К одной из таких моделей относится совмещенный МПА (СМПА), который имеет два типа выходных сигналов: выходные сигналы автомата Мили и выходные сигналы автомата Мура.

Наиболее популярным базисом для реализации цифровых систем являются микросхемы *FPGA* (*field-programmable logic arrays*) [8]. Основными элементами *FPGA* являются элементы табличного типа *LUT* (*look-up table*), программируемые триггера и программируемые межсоединения [5].

В данной работе мы предлагаем метод уменьшения числа *LUT* элементов и количества их уровней в схеме СМПА. Для задания поведения УУ используется язык граф-схем алгоритма [1].

Реализация СМПА в базисе FPGA. Для реализации схемы СМПА необходимо получить три системы булевых функций [6, 7]:

$$Y_{M1} = Y_{M1}(T, X); \quad (1)$$

$$Y_{M2} = Y_{M2}(T); \quad (2)$$

$$\Phi = \Phi(T, X). \quad (3)$$

В системах (1) – (3) используются следующие обозначения: $Y_{M1} \subseteq Y$ – множество микроопераций (МО) автомата Мили, $|Y_{M1}|=N_1$; $Y_{M2} \subseteq Y$ – множество МО автомата Мура, $|Y_{M2}| = N_2$; Ф – множество функций возбуждения памяти МПА; T – множество внутренних переменных, используемых для кодирования состояний $a_m \in A$; $A=\{a_1,\ldots, a_M\}$ – множество состояний. При этом $Y_{M1} \cup Y_{M2} = Y$, где Y – множество МО СМПА ($Y = \{y_1,\ldots, y_N\}$) и выполняется условие: $Y_{M1} \cap Y_{M2} = \varnothing$.

Множества Ф и T имеют одинаковое количество элементов определяющееся соотношением

$$R = \lceil \log_2 M \rceil. \quad (4)$$

При этом $\Phi = \{D_1,\ldots, D_R\}$ и $T=\{T_1,\ldots,T_R\}$. Состояния $a_m \in A$ кодируются кодами $K(a_m)$, хранящиеся во внутренней памяти СМПА. Эта память представлена регистром *RG*. Как правило, *RG* имеет информационные входы типа D [3].

Чтобы найти системы (1) – (3), необходимо построить прямую структурную таблицу (ПСТ) совмещенного МПА. Эта таблица формируется на основе исходной ГСА Г, отмеченной состояниями МПА Мура [6, 7].

Будем рассматривать проблему реализации схемы СМПА в базисе *LUT* элементов. Пусть *LUTer* означает блок, состоящий из элементов *LUT*, триггеров и программируемых межсоединений, показанных на рис. 1.

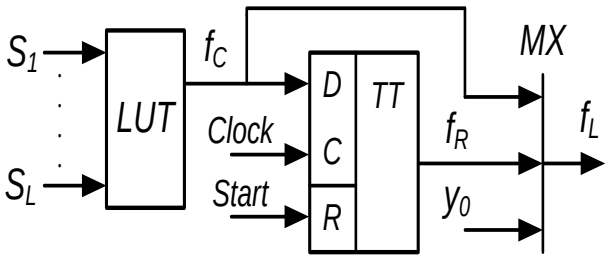

РИС. 1. Логический элемент блока *LUTer*

Элемент *LUT* реализует произвольную логическую функцию f_C, которая зависит от не более, чем S_L аргументов. Выход f_C подается на вход *D* двухтактного триггера. Импульс *Start* используется для операции $f_R := 0$. Импульс *Clock* инициирует операцию $f_R := f_C$. Выход логического элемента может быть либо комбинационным ($f_L = f_C$), либо нет ($f_L = f_R$). Для выбора типа выхода используетс мультиплексор *MX* и сигнал $y0$.

Поставим в соответствие системе (1) блок *LUTer* 1, системе (2) – блок *LUTer* 2 и системе (3) – блок *LUTer T*. Это приводит к СМПА U_1 (рис. 2).

В U_1 регистр *RG* распределен между триггерами блока *LUTer T*. Поэтому блок *LUTer T* имеет входы синхронизации (*Clock*) и обнуления (*Start*).

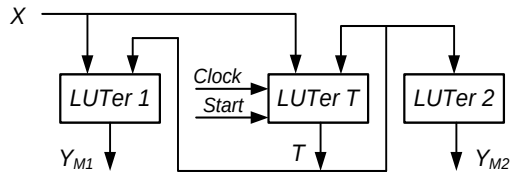

РИС. 2. Структурная схема СМПА U_1

Оптимизация схемы СМПА. При синтезе СМПА в базисе *FPGA* возникает проблема, связанная с ограниченным числом S_L входов *LUT* [9]. Обычно величина S_L не превышает 6 [10, 11].

Пусть $L(f_i)$ – число аргументов в функции $f_i \in \Phi \cup Y$. Если выполняется условие

$$L(f_i) \le S_L, \qquad (5)$$

то для реализации схемы соответствующей функции f_i, достаточно одного элемента *LUT*.

Если условие (5) выполняется для всех функций ΦUY, схема U_1 содержит $N+R$ элементов *LUT* и только один логический уровень. Такая схема имеет минимальное число межсоединений, потребляемую мощность и время задержки (максимальное быстродействие).

Однако анализ библиотеки [12] показывает, что функции (1) и (3) могут зависеть от $L+R \approx 30$ аргументов и условие (5) может выполняться только для ограниченного числа функций $f_i \in \Phi UY$. Поэтому соответствующие функции представляются в виде композиции подфункций, для каждой из которых выполняется условие (5). Для этой цели используются методы функциональной декомпозиции [13, 14]. Их приме нение приводит к многоуровневым схемам, имеющим гораздо худшие характеристики по сравнению с одноуровневыми схемами МПА [15].

Если условие (5) нарушается, то необходимо оптимизировать схему СМПА. Для этого могут быть использованы методы [5].

1. Оптимальное кодирование состояний. Под оптимальным понимается кодирование, уменьшающее число аргументов в функциях (1) – (3). Одним из лучших методов кодирования считается метод *JEDI* [16]. Однако для сложных автоматов (M >50) ни один из методов не гарантирует выполнения условия (5) [17].

2. Использование внутренних блоков памяти. Современные *FPGA* имеют в своем составе блоки памяти *EMB* (*embedded memory blocks*). Один блок *EMB* может заменить сотни элементов *LUT* [18]. Однако эти блоки широко используются для реализации операционных блоков цифровых систем [4]. Поэтому вполне вероятна ситуация, когда для реализации схемы УУ остаются только элементы *LUT*.

3. Структурная декомпозиция. Этот подход связан с увеличением числа структурных уровней схемы МПА [17]. Каждый уровень характеризуется собственными выходными переменными и выходными функциями. При этом выходные функции уровня i служат входными переменными уровня i+1. Известны следующие методы структурной декомпозиции: кодирование логических условий, кодирование наборов микроопераций, преобразование объектов [5].

В работах [20, 21] предлагается метод структурной декомпозиции, ведущий к схемам МПА с тремя структурными уровнями. В настоящей работе предлагается адаптировать этот подход к особенностям совмещенного МПА.

Основная идея предлагаемого метода. Пусть задана ГСА Г, представляющая поведение СМПА и отмеченная состояниями $a_m \in A$. Найдем разбиение $\Pi_A = \{A^1, \ldots, A^K\}$ множества A на классы, для каждого из которых выполняется условие

$$R_k + L_k \le S \quad (k \in \{1,\ldots,k\}). \qquad (6)$$

В условии (6) символ R_k означает число переменных, необходимых для кодирования состояний $a_m \in A^k$, L_k – число входных переменных $x_i \in X$, определяющих переходы из этих состояний и образующих множество $X^k \subseteq X$. Параметр R_k определяется формулой

$$R_k = \left\lceil \log_2(\left|A^k\right|+1) \right\rceil \quad (k \in \{1,\ldots,K\}). \qquad (7)$$

Единица в (7) прибавляется для кода, соответствующего отношению $a_m \notin A$. Используем для кодирования состояний $a_m \in A^k$ переменные из множества $\tau^k \subseteq \tau$, где

$$|\tau| = R_A = R_1 + R_2 + \ldots + R_K. \qquad (8)$$

Очевидно, каждый класс $A^k \in \Pi_A$ определяет множества $X^k \subseteq X$, $Y^k_{M1} \subseteq Y_{M1}$, $Y^k_{M2} \subseteq Y_{M2}$ и $\Phi^k \subseteq \Phi$. Смысл этих множеств ясен из последующего материала.

Закодируем состояния $a_m \in A$ кодами $K(a_m)$ разрядности R. Теперь каждое состояние имеет два кода. Код $K(a_m)$ определяет $a_m \in A$, а код $C(a_m)$ – $a_m \in A^k$. Основываясь на таком подходе, мы предлагаем СМПА U_2 (рис. 3).

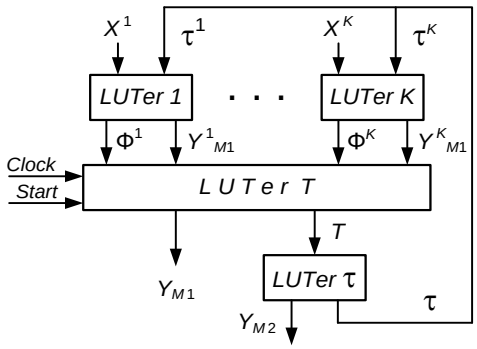

РИС. 3. Структурная схема СМПА U_2

В автомате U_2 блок *LUTer k* формирует функции

$$\Phi^k = \Phi^k(\tau^k, x^k); \quad (9)$$

$$Y^k = Y^k_{M1}(\tau^k, x^k). \quad (10)$$

Блок *LUTer T* формирует микрооперации $y_n \in Y_{M1}$ и внутренние переменные $T_r \in T$. Сигналы *Start* и *Clock* управляют распределенным регистром *RG*. Для формирования функций используются следующие уравнения:

$$D_r = D^1_r \vee D^2_r \vee .... \vee D^K_r, \ (r \in \{1,\ldots,R\}); \quad (11)$$

$$y_n = y^1_n \vee y^2_n \vee .... \vee y^K_n \quad (y_n \in Y_{M1}). \quad (12)$$

Блок *LUTer* τ формирует систему (2) и переменные $\tau_r \in \tau$:

$$\tau_r = \tau_r(T), \ (r \in \{1,\ldots, R_A\}). \quad (13)$$

В силу справедливости (6) только один *LUT* для реализации любой функции блоками *LUTer* 1 – *LUTer K* требуется только один *LUT*. Поэтому число *LUT* NL_k в блоке *LUTer k* определяется формулой $NL_k = \left|\Phi^k\right| + \left|Y^k_{M1}\right|$, ($k \in \{1,\ldots, K\}$). Следовательно, для уменьшения числа *LUT* необходимо находить такое разбиение Π_A, для которого:

$$\left|\Phi^i \cap \Phi^j\right| \to \min; \quad (14)$$

$$\left|Y^i_{M1} \cap Y^j_{M1}\right| \to \min. \quad (15)$$

Для параметров i, j из (14), (15) выполняется следующее: $i, j \in \{1,\ldots, K\}$, $i \neq j$.

Метод поиска разбиения Π_A, удовлетворяющего (14), (15), предложен в [21], поэтому мы его не будем рассматривать.

Предлагаемый метод синтеза схемы СМПА U_2 включает следующие этапы:

- формирование множества состояний по исходной ГСА;
- кодирование состояний $a_m \in A$, оптимизирующее систему (2);
- нахождение разбиения Π_A удовлетворяющего (14), (15);
- кодирование состояний $a_m \in A^k$;
- формирование систем функций (9), (10);
- формирование систем функций (11), (12);
- формирование систем функций (2) и (13);
- реализацию схемы СМПА в заданном базисе.

Пример синтеза автомата U_2. Пусть $U_i(\Gamma_j)$ означает, что автомат U_i реализуется по ГСА Γ_j. Рассмотрим пример синтеза СМПА $U_2(\Gamma_1)$. На рис. 4 показана ГСА Γ_1. Пусть $S = 4$.

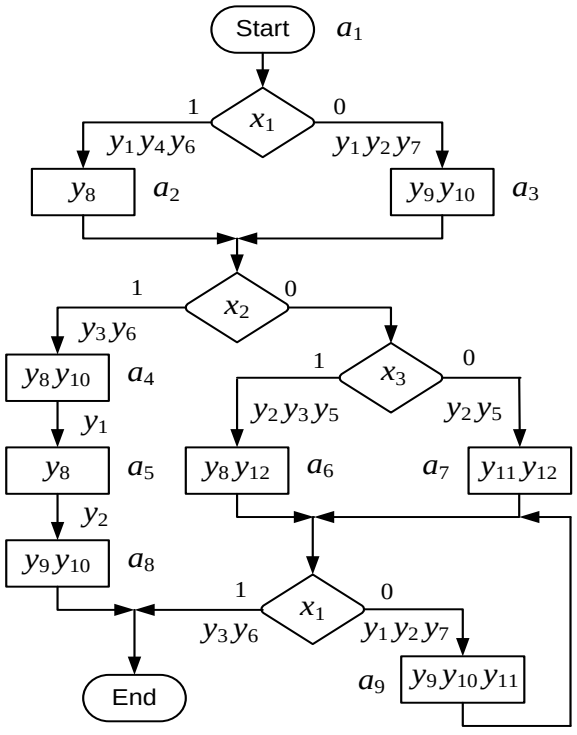

РИС. 4. Исходная ГСА Γ_1

Анализ ГСА Γ_1 показывает, что $M = 9$, $L = 3$, $N_1 = 7$, $N_2 = 5$, $N = 12$. Из (4), получим $R = 4$, что определяет множества $T = \{T_1,...,T_4\}$ и $\Phi = \{D_1,..., D_4\}$.

Закодируем состояния $a_m \in A$ кодами $K(a_m)$ так, чтобы уменьшить число литералов в системе (2). Для этих целей используем алгоритм из работы [19]. Отметим, что такое кодирование может нарушить условие (14). Однако алгоритм [19] приводит к минимизации числа элементов *LUT* и их межсоединений в схеме блока *LUTer* τ. Один из вариантов кодирования показан на рис. 5.

T_3T_4 \ T_1T_2	00	01	11	10
00	a_1	a_2	*	a_3
01	*	a_4	*	a_8
11	*	a_5	*	a_9
10	a_7	a_6	*	*

РИС. 5. Коды состояний автомата $U_2(\Gamma_1)$

Используя ГСА Γ_1 и коды состояний (рис. 5), можно получить следующую систему функций:

$$y_8 = A_2 \vee A_4 \vee A_5 \vee A_6 = T_2,\ y_9 = A_3 \vee A_8 \vee A_9 = T_1,$$
$$y_{10} = A_3 \vee A_4 \vee A_8 \vee A_9 = T_1 \vee \overline{T_3}T_4, \qquad (16)$$
$$y_{11} = A_7 \vee A_9 = \overline{T_2}T_3,\ y_{12} = A_6 \vee A_7 = T_3\overline{T_4}.$$

В функциях (16) символ A_m означает конъюнкцию переменных $T_r \in T$, соответствующая коду $K(a_m)$ состояния $a_m \in A$.

Если блок *LUTer* k имеет $|x^k| = L_k$, то в класс A^k можно поместить до $NS_k = 2^{S-L_k} - 1$, ($k \in \{1,\ldots, K\}$) состояний $a_m \in A$. При $S = 4$ имеем следующие пары: $<L_k, NS_k>$: <0, 15>, <1, 7>, <2, 3>, <3, 1>.

Используя подход [21], найдем следующее разбиение: $\Pi_A = \{A^1, A^2\}$, где $A^1=\{a_1, a_4, a_5, a_6, a_7, a_8, a_9\}$ и $A^2=\{a_2, a_3\}$. При этом $X^1 = \{x_1\}$, $X^2 = \{x_2, x_3\}$, то есть выполняются условия (6) и $X^1 \cap X^2 = \varnothing$.

Используя (7), (8), получим $R_1 = 3$, $R_2 = 2$, $R_3 = 5$. Это дает множества $\tau^1=\{\tau_1, \tau_2, \tau_3\}$ $\tau^2=\{\tau_4, \tau_5\}$ и $\tau = \{\tau_1,\ldots,\tau_5\}$. Закодируем состояния авто- мата так, как приведено в табл. 1.

Для формирования систем (9), (10) необходимо построить прямые структурные таблицы для каждого класса $A^k \in \Pi_A$. Для нашего примера это табл. 2 и 3.

ТАБЛИЦА 1. Коды состояний $A_M \in A^K$

a_m	$\tau_1\tau_2\tau_3$	$\tau_4\tau_5$	a_m	$\tau_1\tau_2\tau_3$	$\tau_4\tau_5$	a_m	$\tau_1\tau_2\tau_3$	$\tau_4\tau_5$
a_1	0 0 1	0 0	a_4	0 1 0	0 0	a_7	1 0 1	0 0
a_2	0 0 0	0 1	a_5	0 1 1	0 0	a_8	1 1 0	0 0
a_3	0 0 0	1 1	a_6	1 0 0	0 0	a_9	1 1 1	0 0

В табл. 2 и 3 используются коды $C(a_m)$ из табл. 1, коды $K(a_s)$ – из рис. 5. При этом учтен факт, что состояния a_6, a_7 – псевдоэквивалентны [5], как и состояния a_6, $a_7 \in A^2$. Из табл. 2 имеем множества $X^1=\{x_1\}$, $Y^1=\{y_1, y_2, y_3, y_4, y_6, y_7\}$ и $\Phi^1 = \Phi$. Из табл. 3 имеем множества $X^2 = \{x_2, x_3\}$, $Y^2 = \{y_2, y_3, y_5, y_6\}$, $\Phi^2 = \{D_2, D_3, D_4\}$. Таким образом, $|X^1 \cap X^2| = 0$, $|Y^1 \cap Y^2| = 3 < N_1$, $|\Phi^1 \cap \Phi^2| = 3 < R$. *LUTerT* не имеет элементов *LUT* для реализации функций y_1, y_4, y_5, y_7, D_1.

ТАБЛИЦА 2. ПСТ для класса A^1

a_m	$C(a_m)$	a_s	$K(a_s)$	X_h^1	Y_h^1	Φ_h^1	h
a_1	001	a_2	0100	x_1	$y_1y_4y_6$	D_2	1
		a_3	1000	$\overline{x_1}$	$y_1y_2y_7$	D_1	2
a_4	010	a_5	0111	1	y_1	$D_2\,D_3\,D_4$	3
a_5	011	a_8	1001	1	y_2	$D_1\,D_4$	4
a_6 a_7	10*	a_1	0000	x_1	y_3y_6	-	5
		a_9	1011	$\overline{x_1}$	$y_1y_2y_7$	$D_1\,D_3\,D_4$	6
a_8	110	a_1	0000	1	-	-	7
a_9	111	a_1	0000	x_1	y_3y_6	-	8
		a_9	1011	$\overline{x_1}$	$y_1y_2y_7$	$D_1\,D_3\,D_4$	9

ТАБЛИЦА 3. ПСТ для класса A^2

a_m	$C(a_m)$	a_s	$K(a_s)$	X_h^2	Y_h^2	Φ_h^2	h
a_2 a_3	*1	a_4	0101	x_2	y_3y_6	$D_2\,D_4$	1
		a_6	0110	$\overline{x_2}x_3$	$y_2y_3y_5$	$D_2\,D_3$	2
		a_7	0010	$\overline{x_2}\,\overline{x_3}$	y_2y_5	D_3	3

Функции (9), (10) формируются тривиальным образом. Например, из табл. 2 можно получить функции: $y_3^1 = \tau_1\overline{\tau_2}x_1 \vee \tau_1\tau_2\tau_3x_1$, $D_2^1 = \overline{\tau_1}\,\overline{\tau_2}\tau_3x_1 \vee \overline{\tau_1}\tau_2\,\overline{\tau_3}$. Из табл. 3 можно получить следующие функции: $y_3^2 = \tau_5x_2 \vee \tau_5x_3$, $D_2^1 = y_3^2$.

Функции (11), (12) также формируются тривиально: $y_3 = y_3^1 \vee y_3^2$ и $D_2 = D_2^1 \vee y_3^2$. Функции (13) формируются по табл. 1. Например, переменная $\tau^1 = 1$ для состояний a_6, a_7 a_8 и a_9. Это дает $\tau_1 = A_6 \vee A_7 \vee A_8 \vee A_9 = T_3\overline{T_4} \vee T_1$. Аналогичным образом формируются остальные уравнения системы (13). Функции (2) уже были получены ранее и представлены в виде (16).

Теперь имеются все системы функций, задающие схемы блоков СМПА $U_2(\Gamma_1)$.

Последний этап предложенного метода связан с применением стандартных индустриальных пакетов [10, 11].

Выводы. Предложенный метод дает наилучшие результаты при выполнении условий

$$R \le S, \qquad (17)$$

$$K \le S. \qquad (18)$$

При выполнении (17) блок *LUTer* τ имеет только один уровень логики. При этом число элементов *LUT* не превышает N_2+R_A. При выполнении (18) блок *LUTer T* имеет один уровень логики, состоящий из не более, чем N_1+R элементов.

Предложенный метод позволяет получить схемы с тремя уровнями логики. Схема имеет регулярный характер межсоединений. Например, переменные $x_l \in X$ поступают только на первый уровень, а переменные $T_r \in T$ – только на третий.

Регулярность связей позволяет упростить решение задачи размещения и трассировки [22]. Оптимизация системы (2) позволяет уменьшить число межсоединений в схеме *LUTer* τ. Максимальное число связей определяется произведением $N_2 \times R$. В рассмотренном примере это 5×4 = 20. Однако анализ системы (16) показывает, что для реализации микроопераций требуется только 7 межсоединений.

Анализ библиотеки [12] показал, что условия (17), (18) имеют место для 78 % всех тестовых примеров. Исследования проводились для микросхем семейства Virtex-6 (S = 6). При этом схемы СМПА U_2 отличались большим быстродействием, чем их аналоги, имеющие структуру U_1. Для оставшихся 22 % тестовых примеров выигрыш был значительно меньше, так как блоки *LUTer T* и *LUTer* τ были реализованы в виде многоуровневых схем.

Дальнейшие направления наших исследований связано с: 1) заменой некоторых *LUT* блоками *EMB* и 2) использованием методов кодирования логических условий для уменьшения параметра *K*.

СПИСОК ЛИТЕРАТУРЫ

1. Grout I. Digital Systems Design with FPGAs and CPLDs. Amsterdam: Elseveit, 2008. 784 p.
2. Баркалов А.А., Титаренко Л.А., Визор Я.Е., Матвиенко А.В., Горина В.В. Уменьшение числа LUT элементов в схеме совмещенного автомата. *Управляющие системы и машины.* 2016. № 3. С. 16 – 22.
3. Грушницкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем с использованием микросхем программируемой логики. СПб: БХВ. Петербург, 2002. 608 с.
4. Skliarova I., Sklyarov V., Sudnitson *A.* Design of FPGA–based circuits using Hierarchical Finite State Machines. Tallinn: TUT Press, 2012. 240 p.
5. Jozwiak L., Chojnski A. Effective and efficient FPGA synthesis through general functional decomposition. – *Journal of System Architecture*. 2003. N 4. P. 247 – 265.
6. Баркалов А.А., Титаренко Л.А., Визор Я.Е., Матвиенко А.В. Уменьшение аппаратурных затрат в совмещенных автоматах. *Управляющие системы и машины*. 2017. № 4. С. 43–50.
7. Баркалов А.А., Титаренко Л.А., Визор Я.Е., Матвиенко А.В. Реализация схемы совмещенного автомата в базисе FPGA. *Комп'ютерні засоби, мережі та системи*. К.: Ін-т кібернетики ім. В.М.Глушкова НАН України. 2016. № 15. С. 10 – 19.
8. Baranov S. Logic Synthesis for Control Automata. Dordrecht: Kluwer Academic Publishers. 1994. 312 p.
9. www.altera.com.
10. www.xilinx.com.
11. Yang S. Logic Synthesis and optimization benchmarks user guide. Microelectronics Center of North Carolina. 1991. 43 p.
12. Kubica M., Kania D., Kulisz J. A technology mapping of FSMs based on a graph of excitations and outputs. IEEE Access. 2018. N 6. P. 16123 – 16131.
13. Rawski M., Tomaszewicz P., Borowski G. and Łuba T. Logic Synthesis Method of Digital Circuits Designed for Implementation with Embedded Memory Blocks on FPGAs, *Design of Digital Systems and Devises. LNEE* 70. Springer, Berlin. 2011. P. 121 – 144.
14. Barkalov A., Titarenko L. Logic Synthesis for FSM – based Control Units. Berlin: Springer, 2009. 233 p.
15. Barkalov A., Titarenko L., Mazurkiewicz M. Foundations of embedded systems. Berlin: Springer, 2019. 196 p.
16. Баркалов А.А., Титаренко Л.А., Визор Я.Е., Матвиенко А.В. Реализация схемы совмещенного микропрограммного автомата в базисе FPGA. *Проблеми інформатизації та управління.* Збірник наукових праць. Національний авіаційний університет. Київ. 2015. Вип. 3(51). С. 5–13.
17. Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. М.: Горячая линия – ТЕЛЕКОМ, 2001. 636 с.

18. Jozwiak L. Using FPGAs in Cyber-Physical Synthesis. *Journal of System Architecture*. 2013. N 2. P. 124 – 1365.
19. Opara A., Kubica M., Kania D. Method of improving time efficiency of decomposition dedicated at FPGA structures and using BDD in process of cyber-physical synthesis. IEEE Access. 2019. N 1. P. 18101 – 18113.
20. Barkalov A., Titarenko L., Mielcatek K. Twofold state assignment for FPGA – based Mealy FSMs. In: Proceedings of International Conference MOCAST-18. Thessaloniki, Greece. New York, IEEE Explore. 2018. P. 1 – 4.
21. Barkalov A., Titarenko L., Mielcatek K. Hardware Reduction for LUT-based Mealy FSMs. *International Journal of Applied Mathematics and Computer Science*. 2018. Vol. 28, N 3. P. 28–41.
22. Sklyarov V., Skliarova I., Barkalov A., Titarenko L. Synthesis and Optimization of FPGA-based Systems. Berlin: Springer. 2014. 432 p.

REFERENCES

1. Grout I. Digital Systems Design with FPGAs and CPLDs. Amsterdam: Elseveit, 2008. 784 p.
2. Barkalov A.A., Titarenko L.A., Vizor Y.E., Matvienko A.V., Gorina V.V. Reducing the number of LUT elements in a combined automaton circuit. Control systems and computers. 2016. № 3. P. 16 – 22.
3. Grushnitsky R.I., Mursaev A.Kh., Ugryumov E.P. Designing systems using programmable logic chips. St. Petersburg: BHV. Petersburg, 2002. 608 p.
4. Skliarova I., Sklyarov V., Sudnitson A. Design of FPGA–based circuits using Hierarchical Finite State Machines. Tallinn: TUT Press, 2012. 240 p.
5. Jozwiak L., Chojnski A. Effective and efficient FPGA synthesis through general functional decomposition. Journal of System Architecture. 2003. N 4. P. 247 – 265.
6. Barkalov A.A., Titarenko L.A., Vizor Y.E., Matvienko A.V. Reducing hardware amount for combined automata. Control systems and computers. 2017, N 4. P. 43 – 50.
7. Barkalov A.A., Titarenko L.A., Vizor Y.E., Matvienko A.V. Synthesis of combined finite state machine with FPGAs. Computer means, networks and systems. K.: V.M. Glushkov Institute of cybernetics NAS of Ukraine. 2016. P. 10 – 19.
8. Baranov S. Logic Synthesis for Control Automata. – Dordrecht: Kluwer Academic Publishers. 1994. 312 p.
9. www.altera.com.
10. www.xilinx.com.
11. Yang S. Logic Synthesis and optimization benchmarks user guide. Microelectronics Center of North Carolina. 1991. 43 p.
12. Kubica M., Kania D., Kulisz J. A technology mapping of FSMs based on a graph of excitations and outputs. IEEE Access, 2018, N 6. P. 16123 – 16131.
13. Rawski M., Tomaszewicz P., Borowski G. and Łuba T. (2011). Logic Synthesis Method of Digital Circuits Designed for Implementation with Embedded Memory Blocks on FPGAs, Design of Digital Systems and Devises. LNEE 70. Springer, Berlin. P. 121 – 144.
14. Barkalov A., Titarenko L. Logic Synthesis for FSM –based Control Units. Berlin: Springer, 2009. 233 p.
15. Barkalov A., Titarenko L., Mazurkiewicz M. Foundations of embedded systems. Berlin: Springer, 2019. 196 p.
16. Barkalov A.A., Titarenko L.A., Vizor Y.E., Matvienko A.V. Implementing circuit of combined finite state machine with FPGAs. Information and management problems. Collection of scientific works. National Aviation University. Kyiv. 2015. Issue 3 (51). 5 – 13 p.
17. Soloviev V.V. Designing digital circuits based on programmable logic integrated circuits. M.: Hot line – TELECOM. 2001. 636 p.
18. Jozwiak L. Using FPGAs in Cyber-Physical Synthesis. Journal of System Architecture. 2013. N 2. P. 124 – 1365.
19. Opara A., Kubica M., Kania D. Method of improving time efficiency of decomposition dedicated at FPGA structures and using BDD in process of cyber-physical synthesis. IEEE Access. 2019. N 1. P. 18101 – 18113.
20. A.Barkalov, L.Titarenko, K.Mielcatek. Twofold state assignment for FPGA - based Mealy FSMs. In: Proceedings of International Conference MOCAST – 18. Thessaloniki , Greece. – New York, IEEE Explore, 2018 - pp. 1 - 4.
21. Barkalov A., Titarenko L., Mielcatek K. Hardware Reduction for LUT-based Mealy FSMs. International Journal of Applied Mathematics and Computer Science. 2018. Vol. 28, N 3. P. 28 – 41.
22. Sklyarov V., Skliarova I., Barkalov A., Titarenko L. Synthesis and Optimization of FPGA-based Systems. Berlin: Springer. 2014. 432 p.

Получено 12.09.2019