КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

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

SYNTHESIS OF COMBINED FINITE STATE MASHINE WITH FPGAs

The method for synthesis of combined finite state machine (FSM) using field-programmable logic arrays is proposed. The FSM circuit is implemented with embedded memory blocks (EMBs).
Key words: combined FSM, FPGA, EMB, synthesis, graph-scheme of algorithm.

Запропоновано метод синтезу суміщеного мікропрограмного автомата в базисі НВІС типу FPGA. Схема автомата реалізується на вбудованих блоках пам'яті.
Ключові слова: суміщений автомат, FPGA, EMB, синтез, граф-схема алгоритму.

Предложен метод синтеза совмещенного микропрограммного автомата (МПА) в базисе СБИС типа FPGA. Схема автомата реализуется на встроенных блоках памяти.
Ключевые слова: совмещенный автомат, FPGA, EMB, синтез, граф-схема алгоритма.

УДК 004.274

А.А. БАРКАЛОВ, Л.А. ТИТАРЕНКО, Я.Е. ВИЗОР, А.В. МАТВИЕНКО

СИНТЕЗ СОВМЕЩЕННОГО МИКРОПРОГРАММНОГО АВТОМАТА В БАЗИСЕ FPGA

Введение. Практически любая цифровая система включает устройство управления, для синтеза которого часто используется модель микропрограммного автомата (МПА) [1, 2]. Одной из моделей МПА является совмещенный автомат, в котором существуют выходные сигналы двух типов [3]. Выходные сигналы типа Мили формируются при переходе между состояниями. Выходные сигналы типа Мура существуют в течение такта работы МПА [2, 3].

Для реализации схем цифровых систем в настоящее время широко используются СБИС типа *FPGA* (field-programmable logic arrays) [4, 5]. Два типа логических элементов, входящих в *FPGA*, могут использоваться для реализации схемы МПА. Первый из них – логические элементы типа *LUT* (look-up table), выходы которых могут быть связаны с входами триггеров. Элементы *LUT* имеют ограниченное число входов ($S \le 6$) и только один выход. Второй тип логических элементов – встроенные блоки памяти типа *EMB* (embedded memory blocks). Их важной характеристикой является реконфигурация, при которой меняется число выходов (t_F) и ячеек памяти (V). При этом общая емкость (V_0) является константой:

$$V_0 = 2^{SA} \cdot t_F, \qquad (1)$$

где S_A – число адресных входов при данном количестве выходов t_F. Как правило, существуют следующие конфигурации EMB: 32K×1, 16K×2, 8K×4, 4K×8, 2K×16, 1K×32512×64, (битов) [4, 5]. Это определяет

следующие пары вида $<S_A, t_F>$: <15, 1>, <14, 2>,<13, 4>, <12, 8>, <11, 16>, <10, 32> и <9, 64>.

При реализации МПА в базисе *FPGA* важно уменьшать площадь кристалла, занимаемого схемой. При этом улучшаются такие характеристики МПА, как время распространения сигналов и потребляемая мощность [6]. Один из подходов для решения данной задачи – это замена элементов *LUT* блоками *EMB* [7–12]. Однако до сих пор никто не рассматривал эту задачу применительно к совмещенному МПА. В настоящей работе мы предлагаем одно из возможных решений и анализируем условия его применения.

Особенности совмещенного МПА. Микропрограммный автомат задается шестикомпонентным вектором:

$$S = <A, X, Y, \delta, \lambda, a_1 >.$$

Здесь $A = \{a_1,\ldots,a_M\}$ – множество внутренних состояний, $X = \{x_1,\ldots,x_L\}$ – множество входных переменных, $Y = \{y_1,\ldots,y_N\}$ – множество выходных переменных, δ – функция переходов, λ – функция выходов, $a_1 \in A$ – начальное состояние. Функция δ служит для нахождения состояния перехода $a_s \in A$ на основе текущего состояния $a_m \in A$ и входных переменных:

$$a_s = \delta(a_m, X). \quad (2)$$

Для автомата Мили функция λ определяет выходную переменную $y_n \in Y$:

$$y_n = \lambda(a_m, X). \quad (3)$$

Для автомата Мура выходные переменные определяются только внутренними состояниями:

$$y_n = \lambda(a_m). \quad (4)$$

В совмещенном МПА множество $Y = Y^1 \cup Y^2$, где Y^1 – множество выходных переменных типа Мили и Y^2 – множество выходных переменных типа Мура. На рис. 1 показан граф, вершины которого соответствуют состояниям, а дуги – пе-

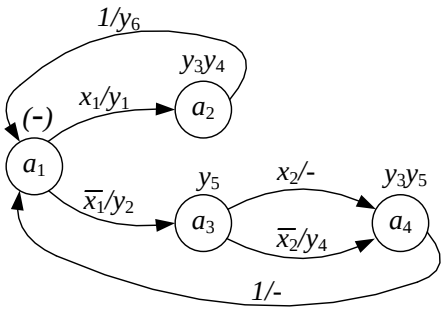

РИС. 1. Граф переходов совмещенного автомата S_1

реходам между ними. Выходы типа Мура показаны рядом с вершинами, а выходы типа Мили – под дугами. Над дугами показаны входные сигналы, вызывающие переход. Как следует из рис. 1, $A=\{a_1,\ldots, a_4\}$, $X=\{x_1, x_2\}$, $Y^1=\{y_1, y_2, y_4, y_6\}$, $Y^2=\{y_3, y_4, y_5\}$. Это дает $M=4$, $L=2$, $N_1=4$ и $N_2=3$, где $N_1=|Y^1|$ и $N_2=|Y^2|$. Анализ множеств Y^1 и Y^2 позволяет получить соотношение $Y^1 \cap Y^2 = \varnothing$.

Закодируем состояния $a_m \in A$ двоичными кодами $K(a_m)$, имеющими R разрядов:

$$R=\lceil \log_2 M \rceil. \quad (5)$$

Для кодирования состояний используем переменные $T_r \in T$, где $T=\{T_1,\ldots,T_R\}$.

Коды состояний хранятся в памяти МПА, которая обычно представляется регистром *RG* с *D*-входами [2]. Для переключения памяти используются функции возбуждения $D_r \in \Phi$, где $\Phi=\{D_1,\ldots, D_R\}$.

Для синтеза схемы совмещенного МПА необходимо получить функции δ и λ. Функция (2) определяется системой булевских функций

$$\Phi = \Phi(T, X). \quad (6)$$

Системы (7) – (8) соответствуют функциям (3) – (4):

$$Y^1 = Y^1(T, X); \quad (7)$$
$$Y^2 = Y^2(T). \quad (8)$$

Системы (6) – (8) определяют структурную схему (рис. 2).

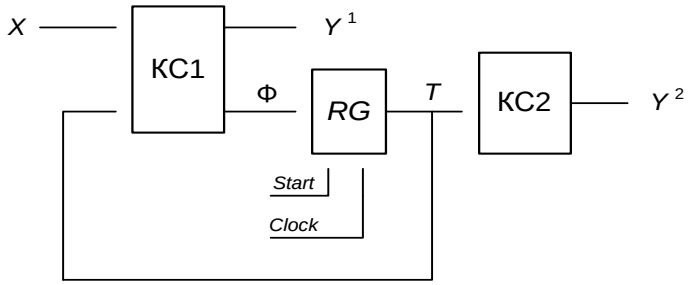

РИС. 2. Структурная схема совмещенного МПА

Блок КС1 генерирует функции (6) – (7), блок КС1 – функции (8). Сигнал *Start* устанавливает в *RG* нулевой код начального состояния $a_1 \in A$. Импульс *Clock* вызывает переключение *RG*, что соответствует переходам МПА.

Реализация совмещенного МПА в базисе FPGA. В настоящей работе предложена структурная схема и метод синтеза совмещенного МПА в базисе *FPGA*. Для реализации схем КС1 и КС2 (рис. 3) предлагается использовать блоки *EMB*.

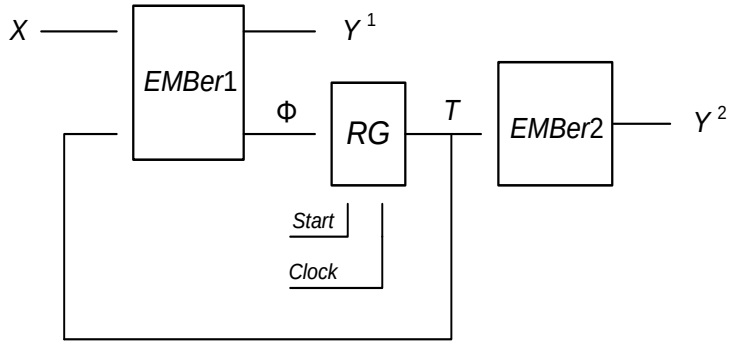

РИС. 3. Структурная схема совмещенного МПА в базисе *FPGA*

Символ *EMBer* определяет блок, состоящий из встроенных блоков памяти. Блок *EMBer*1 (*EMBer*2) соответствует блоку КС1 (КС2). Регистр *RG* реализуется на триггерах, входящих в состав *LUT*-элементов. Если блоки *EMB* являются синхронными, то регистр *RG* не нужен [6]. При этом импульсы *Start* и *Clock* поступают в блок *EMBer*1.

Пусть алгоритм управления представлен граф-схемой алгоритма (ГСА) [2]. Для реализации предлагаемого метода синтеза совмещенного МПА нужно выполнить следующие шаги:

- формирование множества состояний A;
- кодирование состояний $a_m \in A$;
- формирование прямой структурной таблицы МПА;
- формирование таблицы блока *EMBer*1;
- формирование таблицы блока *EMBer*2;
- реализацию схемы МПА в заданном элементном базисе.

Пример применения предложенного метода. Рассмотрим пример синтеза МПА по ГСА Γ_1 (рис. 4). Правила отметки состояний мы обсудим позже.

Выходные переменные $y_n \in Y^1$ показаны на дугах ГСА. Например, при переходе из a_1 в a_2 формируется $y_3 \in Y^1$. Выходные переменные $y_n \in Y^2$ показаны в операторных вершинах ГСА. Например, если МПА находится в состоянии, то формируются y_1 и y_2.

Для отметки состояний предлагается следующая процедура. Если операторная вершина включает переменные $y_n \in Y^2$, то эта вершина отмечается отдельным состоянием. Если две вершины не содержат переменных $y_n \in Y^2$, то они отмечаются одинаковым состоянием, если их выходы связаны с входом одной и той же вершины ГСА.

Применение этой процедуры для ГСА Γ_1 (рис. 4) позволяет найти множество $A = \{a_1, \ldots, a_5\}$ с $M = 5$. Используя (5), найдем $R = 3$, что дает множества $T = \{T_1, T_2, T_3\}$ $\Phi = \{D_1, D_2, D_3\}$. Закодируем состояния $a_m \in A$ тривиальным образом: $K(a_1) = 000, \ldots, K(a_5) = 100$. Найдем множество $Y^1 = \{y_3, y_4, y_5,\}$, и $Y^2 = \{y_1, y_2, y_6, y_7\}$.

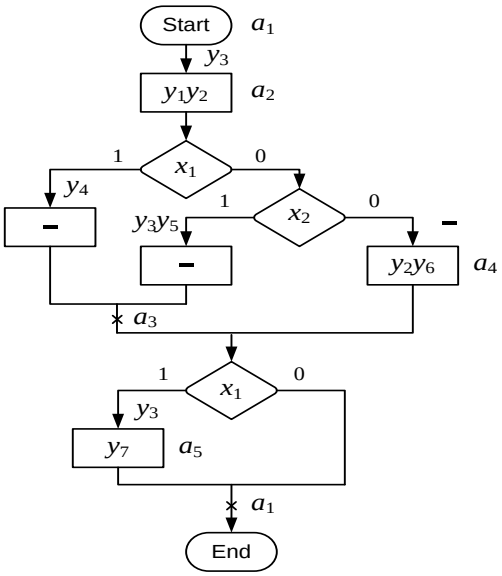

РИС. 4. Отмеченная ГСА Γ_1

Прямая структурная таблица (ПСТ) строится по правилам [2] и включает следующие столбцы: a_m – исходное состояние; $K(a_m)$ – код состояния $a_m \in A$; a_S – состояние перехода; $K(a_S)$ – код состояния $a_S \in A$; X_h – входной сигнал, определяющий переход $< a_m, a_S >$; $Y^1{}_h$ – набор выходных переменных, формируемых при переходе $< a_m, a_S >$; Φ_h – функции возбуждения памяти, принимающие единичное значение для переключения регистра RG из $K(a_m)$ в $K(a_S)$; h – номер перехода. В столбце a_m записываются переменные $y_n \in Y^2$, формируемые в состоянии $a_m \in A$. Для рассматриваемого примера ПСТ представлена табл. 1.

ТАБЛИЦА 1. Прямая структурная таблица совмещенного МПА

a_m	$K(a_m)$	a_S	$K(a_S)$	X_h	$Y^1{}_h$	Φ_h	h
a_1(–)	000	a_2	001	1	y_3	D_3	1
$a_2(y_1y_2)$	001	a_3	010	x_1	y_4	D_2	2
		a_3	010	$\overline{x_1}\, x_2$	$y_3\, y_5$	D_2	3
		a_4	011	$\overline{x_1}\, \overline{x_2}$	–	$D_2\, D_3$	4
a_3(–)	010	a_5	100	x_1	y_3	D_1	5
		a_1	000	$\overline{x_1}$	–	–	6
$a_4(y_2y_6)$	011	a_5	100	x_1	y_3	D_1	7
		a_1	000	$\overline{x_1}$	–	–	8
$a_5(y_7)$	100	a_1	000	*1*	–	–	9

В рассматриваемом примере оба блока МПА реализуются на *EMB*, поэтому нет необходимости в формировании систем (6) – (8).

Блок *EMBer*1задается таблицей со столбцами: $K(a_m)$, X (адрес ячейки), Ф, Y^1 (содержимое ячейки), q (номер ячейки). В рассматриваемом примере количество строк таблицы *EMBer*1 равно 32, так как $R + L = 5$. Часть содержимого *EMBer*1 – задана в табл. 2.

ТАБЛИЦА 2. Фрагмент таблицы блока *EMBer*1

$K(am)$	X	Ф	Y^1	q	h
$T_1T_2T_3$	x_1x_2	$D_1D_2D_3$	$y_3y_4y_5$		
000	00	001	100	1	1
000	01	001	100	2	1
000	10	001	100	3	1
000	11	001	100	4	1
001	00	011	000	5	4
001	01	010	101	6	3
001	10	010	010	7	2
001	11	010	010	8	2

Столбец h табл. 2 добавлен для того, чтобы показать соответствие между ПСТ (табл. 1) и таблицей блока *EMBer*1. Как видно из табл. 2, первые четыре строки соответствуют безусловному переходу из состояния $a_1 \in A$. Второй строке табл. 1 соответствуют две строки табл. 2. Очевидно, таблица блока *EMBer*1 является таблицей истинности, соответствующей ПСТ совмещенного МПА.

Таблица блока *EMBer*2 формируется на основе столбца a_m исходной ПСТ. Она включает столбцы: $K(a_m)$, Y^2, m. Первый столбец определяет адрес ячейки, второй – ее содержимое. Так как $R = 3$, таблица *EMBer*2 имеет 8 строк (табл. 3).

ТАБЛИЦА 3. Таблица блока *EMBer*2

$K(a_m)$	Y^2	m	$K(am)$	Y^2	m
$T_1T_2T_3$	$y_1y_2y_6y_7$		$T_1T_2T_3$	$y_1y_2y_6y_7$	
000	0000	1	100	0000	5
001	1100	2	101	0000	6
010	0110	3	110	0000	7
011	0001	4	100	0000	8

Для реализации схемы МПА необходимо запрограммировать *EMB* на основе таблицы вида табл. 2 и табл. 3 [6]. Для этого используются стандартные промышленные пакеты.

Анализ предложенного метода. Реализация блока *EMBer*1 возможна, если выполняется условие

$$2^{R+L} \le V_0 . \tag{9}$$

При этом для реализации схемы *EMBer*1 необходимо n_1 блоков *EMB*:

$$n_1 = \left\lceil \frac{R + N_1}{t_F} \right\rceil . \tag{10}$$

В формуле (10) символ t_F соответствует числу выходов *EMB*, при котором выполняется (8).

Если условие (9) нарушается, то число *EMB* в схеме резко увеличивается. При этом предложенный метод теряет смысл. Анализ стандартных примеров из библиотеки [13] показал, что условие (9) выполняется для 82 % стандартных автоматов.

При выполнении условия

$$2^{R+L} \times (R+L) \le V_0,$$

совмещенный МПА реализуется тривиальным образом на одном *EMB*. Анализ [13] показал, что это условие выполняется для 76 % всех примеров. Отметим, что библиотека [13] была разработана в 1991 году, поэтому она содержит примеры недостаточной сложности для современных *FPGA*. Однако автоматы из [13] используются всеми авторами [7–12] для сравнения их методов с уже известными.

При выполнении условия

$$2^{R+L} \times N_2 \le V_0 , \tag{11}$$

блок *EMBer*2 реализуется на одном *EMB*. Анализ библиотеки [13] показал, что условие (11) выполняется для 100 % стандартных примеров.

Заключение. В данной работе предложен метод синтеза совмещенного МПА в базисе *FPGA*. При этом схема МПА реализуется на встроенных блоках памяти *EMB*. В работе получены условия, при которых данный МПА может быть реализован на конкретных *EMB*. Число *EMB* в схеме зависит от параметров МПА (L, N, R, N_1, N_2) и *EMB* (S_A, t_F).

Анализ библиотеки [13] показал, что предложенный метод не может быть использован только для 18 % стандартных примеров. Разработка методов синтеза совмещенных МПА при нарушении условия (11) является направлением наших дальнейших исследований. Мы планируем адаптировать методы замены

логических условий [1, 6] и оптимизации автомата Мура [14] к особенностям, как модели совмещенного МПА, так и базиса *FPGA*.

1. *Baranov S.* Logic Synthesis for Control Automata. – Dordrecht: Kluwer Academic Publishers, 1994. – 312 p.
2. *DeMicheli G.* Synthesis and Optimization of Digital Circuits. – New York: McGraw-Hill, – 1994. – 636 p.
3. *Соловьев В.В.* Проектирование цифровых схем на основе программируемых логических интегральных схем. – М.: Горячая линия – ТЕЛЕКОМ, 2001. – 636 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. *Грушницкий Р.И., Мурсаев А.Х., Угрюмов Е.П.* Проектирование систем с использованием микросхем программируемой логики. – СПб: БХВ. – Петербург, 2002. – 608 с.
6. *Sklyarov V., Skliarova I., Barkalov A., Titarenko L.* Synthesis and Optimization of FPGA-based Systems. – Berlin: Springer, 2014. – 432 p.
7. *Cong J. and Yan. K.* Synthesis for FPGAs with Embedded Memory Blocks // Proceeding of the 2000 ACM/SIGDA 8th International Symposium on FPGAs. – 2000. – P. 75 – 82.
8. *Garcia–Vargas L., Senhadji–Navarro R., M. Civit–Balcells A. and Guerra–Gutierrezz P.* ROM–Based Finite State Machine Implementation in Low Cost FPGAs // IEEE International Simposium on Industrial Electronics, Vigo. – 2007. – P. 2342–2347.
9. *Nowicka M., Łuba T. and Rawski V.* FPGA–based decomposition of boolean functions: algorithms and implementations // Advanced Computer Systems. – 1999. – P. 502 – 509.
10. *Rawski M., Selvaraj H. and Łuba T.* An application of functional decomposition in ROM–based FSM implementation in FPGA devices // Journal of System Architecture 51(6–7). – 2005. – P. 424 – 434.
11. *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.
12. *Tiwari A. and Tomko K.* Saving power by mapping finite state machines into embedded memory blocks in FPGAs, Proceedings of Design Automation and Test in Europe. – 2004. – Vol. 2. – P. 916 – 921.
13. *Yang S.* Logic Synthesis and optimization benchmarks user guide // Microelectronics Center of North Carolina. – 1991. – 43 p.
14. *Баркалов А.А.* Принципы оптимизации логической схемы микропрограммного автомата Мура // Кибернетика и системный анализ. – 1998. – №1. – С. 65 – 72.

Получено 20.07.2015