Розробка засобів обчислювальної техніки та дослідження комп'ютерних мереж

УДК 004.274

А.А. Баркалов[1], д-р.техн. наук, проф.,
Л.А. Титаренко[1], д-р.техн. наук, проф.,
И.Я. Зеленева[2], канд. техн. наук, доц.
С.С. Грушко[2], аспирант
[1]Университет Зеленогурский, г. Зеленая Гура, Польша,
[2] Запорожский национальный технический университет, г. Запорожье, Украина
E-mail: A.Barkalov@iie.uz.zgora.pl

Использование псевдоэквивалентных состояний в совмещенном микропрограммном автомате

Предметом исследования в данной статье является логическая схема совмещенного микропрограммного автомата (СМПА), который объединяет функции автомата Мили и автомата Мура. Рассмотрена задача оптимизации логической схемы совмещенного микропрограммного автомата, реализованного в базисе FPGA. При этом можно использовать три внутренних компонента FPGA: элементы табличного типа LUT (look up table), встроенные блоки памяти EBM (embedded memory block) и программируемые межсоединения. Число компонентов FPGA, необходимое для реализации логической схемы автомата, зависит от параметров СМПА и характеристик элементного базиса. Уменьшение требуемого числа компонентов ведет к уменьшению площади кристалла FPGA, занимаемой схемой СМПА, за счет чего снижаются аппаратурные затраты в схеме, потребляемая энергия, и повышается экономическая эффективность проекта в целом. Для решения задачи оптимизации схемы СМПА по критерию аппаратурных затрат в статье предлагается использовать метод псевдоэквивалентных состояний с целью уменьшения разрядности входного кода схемы СМПА, а также структурные особенности элементного базиса FPGA. В результате достигается уменьшение общей площади схемы на кристалле. Приведен пример решения этой задачи по исходным данным в виде граф-схемы алгоритма управления.

Ключевые слова: совмещенный микропрограммный автомат, FPGA, LUT, встроенные блоки памяти, псевдоэквивалентные состояния, граф-схема алгоритма.

Введение

Цифровые системы, как правило, содержат в своей структуре устройство управления – неотъемлемую и важную часть, обеспечивающую корректное функционирование системы в целом [1]. На практике устройства управления часто представлены моделями микропрограммных автоматов (МПА), а их имплементация выполняется в современном базисе больших интегральных схем [3]. При этом актуальной задачей является оптимизация площади кристалла СБИС, занимаемой схемой МПА [2, 4]. Решение этой задачи позволяет увеличить частоту функционирования МПА и уменьшить потребляемую мощность. Методы решения данной задачи зависят от модели МПА и структуры элементного базиса, в котором реализуется схема автомата [4, 10 - 11].

В практике проектирования цифровых систем часто используется модель совмещенного микропрограммного автомата (СМПА) – имеется ввиду совмещение выходных сигналов автоматов Мили и Мура [2, 6]. В настоящей работе предлагается метод синтеза СМПА, ориентированный на широко используемый базис FPGA (field programmable gate arrays) [5 - 7]. Базовое решение задачи синтеза опубликовано ранее в статье [8], на основе чего далее рассматривается задача оптимизации площади кристалла, требуемой для размещения логической схемы СМПА. Для решения предлагается использовать метод псевдоэквивалентных состояний [4, 10] с целью уменьшения разрядности входного кода схемы СМПА, и, как следствие, уменьшения общей площади схемы на кристалле.

Особенности построения модели СМПА

Совмещенный микропрограммный автомат может быть представлен в виде восьмикомпонентного вектора:

$$S = <A, X, Y^1, Y^2, \delta, \lambda_1, \lambda_2, \alpha_1>. \quad (1)$$

Вектор (1) состоит из следующих компонентов:

$A=\{a_1, \dots, a_M\}$ – множество внутренних состояний;

$X=\{x_1, \dots, x_L\}$ – множество логических условий;

Y^1 – множество выходных переменных автомата Мили;

Y^2 – множество выходных переменных автомата Мура;

δ – функция переходов;

λ_1 – функция выходов автомата Мили;

λ_2 – функция выходов автомата Мура;

$a_1 \in A$ – начальное состояние СМПА.

Функция δ служит для нахождения состояния перехода $a_s \in A$ в зависимости от текущего состояния $a_m \in A$ и вектора входных переменных:

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

Множество выходных переменных $Y = Y^1 \cup Y^2$ состоит из $N_1=|Y^1|$ переменных автомата Мили и $N_2=|Y^2|$ переменных автомата Мура. При этом $Y^1 \cap Y^2 = \varnothing$ и $N_1+N_2 = N$.

Функция λ_1 определяет выходные переменные $y_n \in Y^1$:

$$y_n = \lambda_1(a_m, X) \quad (3)$$

Функция λ_2 определяет выходные переменные $y_n \in Y^2$:

$$y_n = \lambda_2(a_m) \quad (4)$$

В настоящей работе СМПА представляется граф-схемой алгоритма (ГСА) [1]. Для нахождения функций (2) - (4) необходимо построить по ГСА прямую структурную таблицу (ПСТ).

Реализация схемы СМПА в базисе FPGA

Для реализации логической схемы СМПА можно использовать три структурных компонента FPGA: элементы табличного типа LUT (look up table), встроенные блоки памяти EMB (embedded memory block) и программируемые межсоединения [8, 9]. Элементы LUT представляют собой ОЗУ, имеющее *s* входов и один выход. При этом число входов определяет максимальное число аргументов булевской функции, которая может быть реализована на одном LUT элементе. Как правило, число входов $S \le 6$ [8, 9].

Блок EMB представляет собой ОЗУ, имеющее S_A входов и t_F выходов. Общая емкость блока является константой и определяется следующей формулой:

$$V_0 = 2^{S_A} \cdot t_F. \quad (5)$$

Параметры S_A и t_F могут быть выбраны из следующего множества пар вида $\langle S_A, t_F \rangle$: ⟨15,1⟩, ⟨14,2⟩, ⟨12,8⟩, ⟨11,16⟩, ⟨10,32⟩ и ⟨9,64⟩.

Для реализации схемы СМПА по ГСА Г необходимо выполнить некоторые промежуточные этапы:

1. Отметить ГСА состояниями СМПА.

2. Закодировать состояния $a_m \in A$ двоичными кодами $K(a_m)$ разрядности *R*.

3. Построить прямую структурную таблицу автомата.

4. Получить системы функций, которые соответствуют (2) - (4).

Для кодирования состояний $a_m \in A$ используются внутренние переменные $T_r \in T$, где $|T|=R$. Закодируем состояния минимальным числом внутренних переменных (учитывая, что общее число состояний СМПА равно *M*):

$$R = \lceil \log_2 M \rceil. \quad (6)$$

Для хранения кодов состояний используется регистр (RG), содержащий *R* триггеров. Как правило, при синтезе схем в базисе FPGA используются D триггеры [6]. Для изменения содержимого RG необходимо задать функции возбуждения $D_r \in \varphi$, где $\varphi = \{D_1, \dots, D_r\}$. Изменение содержимого RG происходит по сигналу синхронизации Clock. Для установки в RG кода начального состояния $a_1 \in A$ используется импульс Start.

После выполнения этапов 1 - 2, прямая структурная таблица формируется с использованием правил [1]. ПСТ является основой для получения систем функций:

$$\varphi = \varphi(T, X); \quad (7)$$

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

$$Y^2 = Y^2(T). \quad (9)$$

Система (7) определяет функцию (2), система (8) – функцию (3) и система (9) – функцию (4).

Существуют две тривиальные модели СМПА для базиса FPGA. Недостатком обеих моделей является избыточность реализации для практических примеров. Это значит, что для автоматов реальной сложности [1] требуется больше логических элементов, чем для многоуровневых моделей [6].

В статье [10] предложен способ совмещения элементов LUT и EMB для построения более компактной схемы СМПА. Примем эту структуру в качестве базовой для последующей оптимизации. Ориентируясь на структурные особенности базиса FPGA, будем обозначать как LUTer схему, состоящую из элементов LUT, а схему из блоков EMB будем называть EMBer. Эти обозначения используются в структурной схеме СМПА U_1 (рис.1).

В автомате U_1 блок EMBer реализует системы (7) - (8), а LUTer — систему (9). Этот подход целесообразен, если выполняются следующие условия:

$$2^{L+R}(N_1 + R) \le V_0; \quad (10)$$

$$2^{L+R}(N + R) > V_0; \quad (11)$$

$$R \le S. \quad (12)$$

При выполнении (10) системы (7) - (8) реализуются на одном блоке EMB. Условие (11) свидетельствует, что нельзя реализовать на одном EMB системы (7) - (9). Условие (12) свидетельствует, что каждая из функций $y_n \in Y^2$ реализует-

ся на одном элементе LUT. Модель U_1 – это пример гетерогенной реализации схемы МПА, когда разные логические элементы используются для реализации разных систем функций [5].

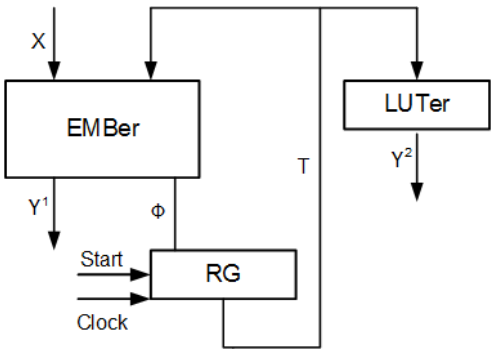

Рисунок 1 - Структурная схема СМПА U_1

В настоящей работе предлагается метод синтеза для случая, когда условие (10) не выполняется. Данный метод основан на использовании классов псевдоэквивалентных состояний (ПЭС) автомата Мура [9].

Метод синтеза СМПА с использованием псевдоэквивалентных состояний

Начнем с принципа отметки состояний СМПА. Рассмотрим граф-схему алгоритма (ГСА) Γ_1 (рис. 2). На дугах ГСА Γ_1 показаны переменные $y_n \in Y^1$, а в операторных вершинах – переменные $y_n \in Y^2$.

Из-за наличия системы (9) операторные вершины должны быть отмечены состояниями автомата Мура [1]. Предлагается обозначать вершины одинаковыми состояниями, если вершины не содержат переменных $y_n \in Y^2$ и при этом выходы таких вершин связаны с входом одной и той же вершины ГСА. Это условие позволяет использовать по два раза отметки a_3 и a_6 в данном примере (рис. 2).

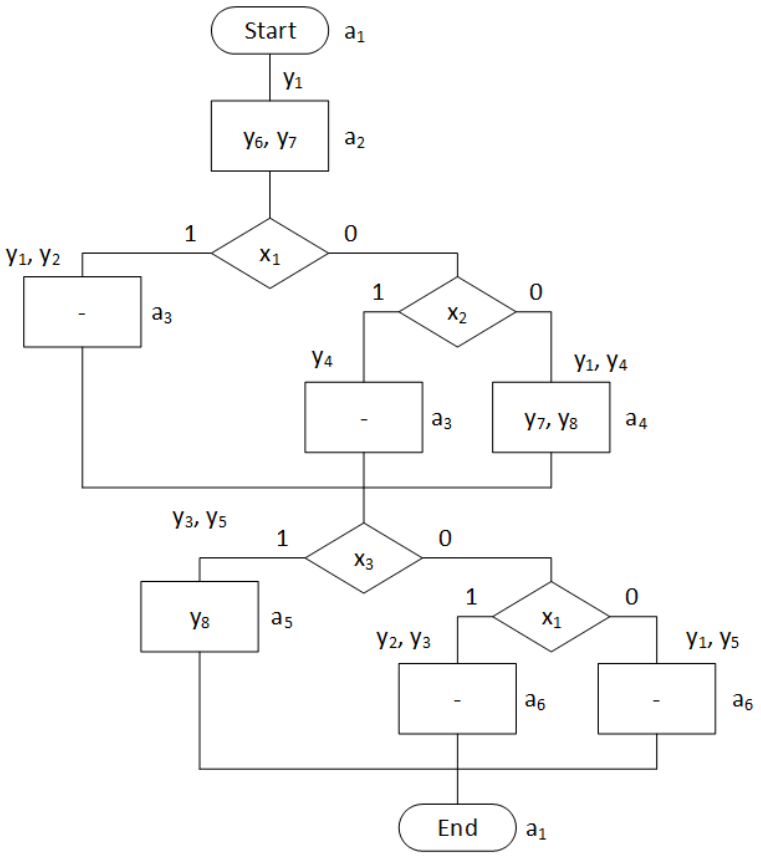

Рисунок 2 - ГСА Γ_2, отмеченная состояниями совмещенного автомата

Из ГСА Г1 имеем следующие множества и их параметры: $A = \{a_1, \dots, a_6\}$, M = 6, $X = \{x_1, x_2, x_3\}$, L = 3, $Y^1 = \{y_1, \dots, y_5\}$, N_1 = 5, $Y^2 = \{y_6, y_7, y_8\}$, N_2 = 3, N = 8.

Используя (6), получаем: R = 3, $T = \{T_1, T_2, T_3\}$ и $\Phi = \{D_1, D_2, D_3\}$.

Состояния a_m, $a_s \in A$ называются псевдоэквивалентными (ПЭС), если отмеченные ими вершины связаны со входом одной и той же вершины ГСА [9]. Это определение позволяет найти разбиение Π_A множества А на классы ПЭС B_1, …, B_I. Для ГСА Г1 имеем $\Pi_A = \{B_1, \dots, B_4\}$, где $B_1 = \{a_1\}$, $B_2 = \{a_2\}$, $B_3 = \{a_3, a_4\}$ и $B_4 = \{a_5, a_6\}$.

Закодируем классы $B_i \in \Pi_A$ кодами $K(B_i)$, имеющими R_I разрядов, где:

$$R_I = \lceil \log_2 I \rceil. \quad (13)$$

Для ГСА Г1 имеем количество классов ПЭС I = 4, а значит R_I = 2. Используем для кодирования переменные $\tau_r \in \tau$, где $|\tau| = R_I$. В нашем примере имеем множество переменных $\tau = \{\tau_1, \tau_2\}$. Закодируем классы тривиальным образом:

$K(B_1)$ = 00, …, $K(B_4)$ = 11.

Пусть выполняются следующие условия:

$$2^{L+R_I}(N_1 + R) \le V_0; \quad (14)$$
$$2^{L+R}(N_1 + R) > V_0. \quad (15)$$

В этом случае предлагается реализовать схему СМПА, используя модель U_2 (рис. 3).

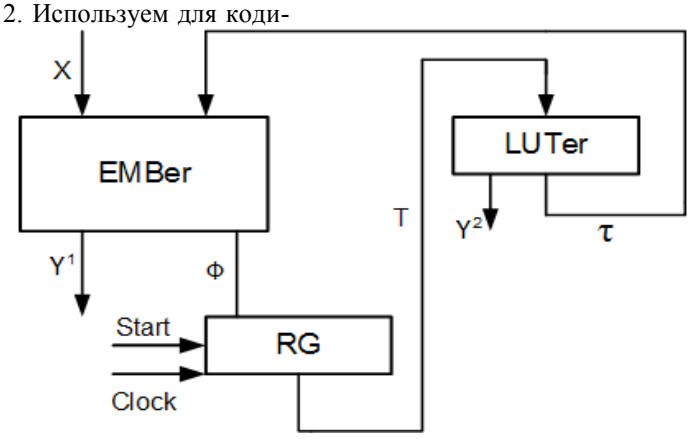

Рисунок 3 - Структурная схема СМПА U_2

В автомате U_2 EMBer реализует системы функций:

$$\Phi = \Phi(\tau, X); \quad (16)$$
$$Y^1 = Y^1(\tau, X). \quad (17)$$

Блок LUTer реализует системы (9) и (18), где

$$\tau = \tau(T). \quad (18)$$

Если выполняется (14), то блок EMBer реализуется на одном EMB. Если выполняется (12), то блок LUTer включает N_2+R_I элементов LUT. Это на R_I больше, чем в автомате U_1 и целесообразно в том случае, если в блоке LUTer есть свободные ресурсы.

В настоящей работе предлагается метод синтеза СМПА U_2 по исходной ГСА Г. Метод включает следующие этапы:

1. Формирование множества состояний A.
2. Кодирование состояний $a_m \in A$.
3. Формирование множества классов псевдоэквивалентных состояний Π_A.
4. Кодирование классов $B_i \in \Pi_A$.
5. Формирование ПСТ автомата U_2.
6. Формирование таблицы блока EMBer.
7. Формирование таблицы блока LUTer.
8. Реализация схемы СМПА в заданном элементном базисе.

Рассмотрим пример синтеза СМПА U2 для ГСА Г1.

Пример синтеза СМПА U2

Этапы 1, 3, 4 уже выполнены. Если условие (12) выполняется, то состояния $a_m \in A$ кодируются тривиальным образом (этап 2). Это дает следующие коды: $K(a_1)$ = 000, … , $K(a_6)$ = 101.

В ПСТ автомата U_2 рассматриваются переходы из классов ПЭС. Поэтому ПСТ имеет следующие столбцы: B_i, $K(B_i)$, a_s, $K(a_s)$, X_h, Y_h^1, Φ_h, h. Здесь B_i – класс ПЭС, включающий исходное состояние $a_m \in A$; $K(B_i)$ – код класса $B_i \in \Pi_A$; a_s - состояние перехода; $K(a_s)$ – код состояния $a_s \in A$; X_h – входной сигнал, определяющий переход <a_m, a_s>; Y_h^1 – выходные переменные $y_n \in Y^1$, формируемые на переходе <a_m, a_s>; Φ_h – набор функций

возбуждения памяти, принимающих единичные значения для записи в RG кода $K(a_s)$; h – номер перехода ($h = \overline{1,H}$). Для рассматриваемого примера $H = 7$ (табл. 1).

Таблица 1 - Прямая структурная таблица СМПА U_1

B_i	$K(B_i)$	a_s	$K(a_s)$	X_h	Y_h^1	Φ_h	h
B_1	00	a_2	001	1	y_1	D_3	1
B_2	01	a_3	010	x_1	y_2y_3	D_2	2
		a_3	010	$\overline{x_1}x_2$	-	D_2	3
		a_4	011	$\overline{x_1}\cdot\overline{x_2}$	y_1y_4	D_2D_3	4
B_3	10	a_5	100	x_3	y_3y_5	D_1	5
		a_6	101	$\overline{x_2}x_1$	y_2y_3	D_1D_3	6
		a_6	101	$\overline{x_2}\cdot\overline{x_1}$	y_1y_5	D_1D_3	7

Фрагмент таблицы блока EMBer для рассматриваемого примера представлен в табл. 2. Эта таблица задает переходы из состояний класса $B_2 \in \Pi_A$. В табл. 2 столбец q показывает номер ячейки, а столбец h – соответствующий номер строки ПСТ.

Таблица 2 - Фрагмент таблицы блока EMBer автомата U_2

$K(B_i)$	X	Y^1	Φ	q	h
	$x_1x_2x_3$	$y_1y_2y_3y_4y_5$	$D_1D_2D_3$		
01	000	10010	011	9	4
01	001	10010	011	10	4
01	010	00000	010	11	3
01	011	00000	010	12	3
01	100	01100	010	13	2
01	101	01100	010	14	2
01	110	01100	010	15	2
01	111	01100	010	16	2

Далее, таблица блока LUTer имеет столбцы $K(a_m)$, Y^2, τ, m. Она соответствует N_2+R_I таблицам, определяющим функции (9) и (18). Для рассматриваемого примера эта таблица имеет M = 6 строк (табл. 3).

Таблицы для отдельных функций $y_n \in Y^2$ и $\tau_r \in \tau$ формируются тривиальным образом, используя информацию из табл. 3. Если $R \leq S$, то каждая из таблиц выходных функций однозначно определяет содержимое соответствующих элементов LUT.

Таблица 3 - Таблица блока LUTer автомата U_2

$K(a_m)$	Y^2	τ	m
$T_1T_2T_3$	$y_6y_7y_8$		
000	000	00	1
001	110	01	2
010	000	10	3
011	011	10	4
100	001	11	5
101	000	11	6

Предложенный метод псевдоэквивалентных состояний для СМПА целесообразно использовать, если условие (10) не выполняется, а условие (15) – выполняется.

При выполнении (15) блок EMBer реализуется в виде одного блока встроенной памяти EMB. Анализ параметров некоторых реальных алгоритмов управления, а также библиотеки [11] показывает, что условие (15) выполняется для 86% стандартных примеров.

Заключение

Метод псевдоэквивалентных состояний, как показали исследования авторов, целесообразен и эффективен при выполнении еще одного условия $R \leq S$, обозначенного выше под номером (12), однако на практике как раз это условие часто нарушается. Если же $R > S$, то необходимо выполнить ряд действий по декомпозиции [2, 5] исходной системы функций СМПА:

1. Получить формулы для отдельных функций из таблицы блока LUTer и выполнить их оптимизацию.
2. Выполнить декомпозицию оптимальных функций.
3. Построить таблицы для всех частей оптимальных функций, причем каждая из этих таблиц соответствует отдельному элементу LUT блока LUTer.

Выполнение перечисленных выше действий существенно осложняет процесс синтеза, занимает дополнительное время проектирования, поэтому имеет смысл продолжить поиск альтернативных методов оптимизации.

Направление дальнейших исследований авторов связано с разработкой методов синтеза СМПА при нарушении условий (12) и (15). Для этого могут быть использованы различные методы [4, 9], адаптированные к условиям совмещенного автомата.

Список литературы

1. Baranov S. Logic and System Desing of Digital Systems/S. Baranov.–Tallinn:TUT Press, 2008.–267 pp.
2. De Micheli G. Synthesis and Optimization of Digital Circuits / G. De Micheli. – New York: McGraw-Hill, 1994. – 636 pp.
3. Соловьев В.В. Логическое проектирование цифровых систем на основе программируемых логических интегральных схем /В.В. Соловьев, А. Климович.–М.:Горячая Линия–Телеком, 2008. – 376 с.
4. Sklyarov V. Synthesis and Optimization of FPGA – based Systems / V. Sklyarov, I. Sklyarova, A. Barkalov, L. Titarenko. – Berlin: Springer, 2014. – 432 pp.
5. Sklyarova I. Design of FPGA – based circuits using Hierarchical Finite State Machines / I. Sklyarova, V. Sklyarov, A. Sudnitson. – Tallinn: TUT Press, 2012. – 240 pp.
6. Intel FPGAs – Режим доступу: https://www.altera.com/products/fpga/overview.html (дата звернення 29.11.2016 р). – Назва з екрана.
7. FPGAs & 3D ICs – Режим доступу: https://www.xilinx.com/products/silicon-devices/fpga.html (дата звернення 29.11.2016 р). – Назва з екрана.
8. Баркалов А.А. Реализация совмещенного микропрограммного автомата в базисе FPGA / А.А. Баркалов, Л.А. Титаренко, И.Я. Зеленева // Наукові праці ДонНТУ. Серія «Інформатика, кібернетика та обчислювальна техніка» (ІКОТ) №2(21). – Красноармійськ: ДВНЗ «ДонНТУ», 2015. – С. 84-88.
9. Баркалов А.А. Оптимизация схемы совмещенного автомата в базисе FPGA способом замены входных переменных / А.А. Баркалов, И.Я. Зеленева, С.С. Грушко // Науковий вісник Чернівецького національного університету імені Юрія Федьковича. Серія: Комп'ютерні системи та компоненти. Том 6, Випуск 2. – Чернівці: ЧНУ, 2015. – с. 49 -54.
10. Баркалов А.А. Принципы оптимизации логической схемы микропрограммного автомата Мура / А.А. Баркалов // Кибернетика и системный анализ. – 1998, №1. – С.65 – 72.
11. Yang S. Logic Synthesis and optimization benchmarks user guide / S. Yang– North Carolina: Microelectonics Center of North Carolina, 1991. – 43 pp.

Надійшла до редакції 15.03.2017

О.О. БАРКАЛОВ[1], Л.О. ТИТАРЕНКО[1], І.Я. ЗЕЛЕНЬОВА[2], С.С. ГРУШКО[2]
[1]Університет Зеленогурський (Польща), [2]ЗНТУ (Україна)

ВИКОРИСТАННЯ ПСЕВДОЕКВІВАЛЕНТНИХ СТАНІВ У СУМІЩЕНОМУ МІКРОПРОГРАМНОМУ АВТОМАТІ

Предметом дослідження в даній статті є логічна схема суміщеного мікропрограмного автомата (СМПА), який об'єднує функції автомата Мілі і автомата Мура. Розглянуто задачу оптимізації логічної схеми суміщеного мікропрограмного автомата, реалізованого в базисі FPGA. Число компонентів FPGA, необхідне для реалізації логічної схеми автомата, залежить від параметрів СМПА і характеристик елементного базису. Очевидно, що зменшення необхідного числа компонентів веде до зменшення площі кристала FPGA, що займає схема СМПА, за рахунок чого знижуються апаратурні витрати в схемі, споживана енергія, і підвищується економічна ефективність проекту в цілому. Для вирішення завдання оптимізації схеми СМПА за критерієм витрат апаратури в статті пропонується використовувати метод псевдоеквівалентних станів з метою зменшення розрядності вхідного коду схеми СМПА, а також структурні особливості елементного базису FPGA. В результаті досягається зменшення загальної площі схеми на кристалі. Наведено приклад вирішення цього завдання за вихідними даними у вигляді ГСА.

Ключові слова: суміщений мікропрограмний автомат, FPGA, LUT, вбудовані блоки пам'яті, псєвдоеквівалентні стани, граф-схема алгоритму.

A.A. BARKALOV[1], L.A. TITARENKO[1], I.J. ZELENEVA [2], S.S. HRUSHKO[2]
[1]University of Zielona góra (Poland), [2] Zaporizhzhia National Technical University (Ukraine)

USE OF PSEUDOEQUIVALENT STATES IN THE COMBINED FINITE STATE MACHINE

The subject of the research in this article is the logic circuit of the combined finite state machine (CFSM), which combines the functions of the both FSM Mealy and Moore. In practice, such a model of control automata is used widely, but in the literature there is almost no theoretical description CFSM models and ways to optimize them. The article considers the problem of optimizing the logic of the combined finite state machine implemented in FPGA basis. In this case, you can use three internal FPGA components: the elements of a table-type LUT (look up table), embedded memory blocks EBM (embedded memory block) and programmable interconnects.

The distributed register memory inside of FPGA is used to store the codes of CFSM states. D-flip-flops are commonly used in the synthesis of circuits in FPGA. The number of FPGA components, required to implement the logic of the automaton circuit, depends on the CFSM parameters and characteristics of element basis. Obviously, the reduction of necessary number of components leads to a decrease area occupied CFSM scheme in FPGA, thereby leads to reducing the hardware amount and power consumption in the circuit, and as result, increases the efficiency of the whole project.

To solve the problem of CFSM optimization for a criterion of hardware expenses in this article it's proposed to use the structural features of the basis FPGA, as well as the method of pseudoequivalent states. The states are called as a pseudoequivalent, if they mark some vertices linked with the input of the same next vertex in flow-chart.

The proposed method includes the following steps: forming a plurality of CFSM states; encoding of states; forming a set of classes pseudoequivalent states; formation EMBer and LUTer blocks and tables; implementation of the CFSM scheme in a given element basis.

As a result, it's possible to reduce the necessary number of EMB to only single for implementing CFSM circuit in FPGA. As a general result is a decreasing of the total area of CFSM circuit on a chip. The advisability conditions of this method are discussed. An example of the solution of this problem on the initial data in the form of a flow-chart of the control algorithm is explained.

Keywords: combined finite state machine, FPGA, LUT, embedded memory blocks, pseudoequivalent state, flow-chart of algorithm.

REFERENCES

1. Baranov, S. (2008), *Logic and System Desing of Digital Systems*, TUT Press, Tallinn, 267 p.
2. De Micheli, G. (1994), *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, New York, 636 p.
3. Solovev, V.V., Klimovich, A. (2008), *The logical design of digital systems based on programmable logic integrated circuits* [Logicheskoe proektirovanie tsifrovyih sistem na osnove programmiruemyih logicheskih integralnyih shem], Goryachaya Liniya, Telekom, Moscow, 376 p.
4. Sklyarov, V., Sklyarova, I., Barkalov, A., Titarenko, L. (2014), *Synthesis and Optimization of FPGA - based Systems*, Springer, Berlin, 432 p.
5. Sklyarova, I., Sklyarov, V., Sudnitson, A. (2012), *Design of FPGA – based circuits using Hierarchical Finite State Machines, TUT Press*, Tallinn, 240 p.
6. "Intel FPGAs" available at: https://www.altera.com/products/fpga/overview.html
7. "FPGAs & 3D ICs", available at: https://www.xilinx.com/products/silicon-devices/fpga.html
8. Barkalov, A.A., Titarenko, L.A. & Zeleneva, I.J. (2015), *Implementing of combined finite state machine with FPGA.* [Realizatsiya sovmeschennogo mikroprogramnogo avtomata v bazise FPGA]. *Naukovi pratsi DonNTU: Informatyka, Kybernetyka ta obchysliuvalna teknika*, №2(21), pp. 84-88.
9. Barkalov, A.A., Zeleneva, I.J. & Hrushko, S.S. (2015), Optimization of combined automaton circuit on base FPGA using the method of input variables changing [Optimizatsiya shemy sovmeschennogo avtomata v bazise FPGA sposobom zamenyi vhodnyih peremennyih], *Naukovyi visnyk Chernivetskogo Natsionalnogo Universytetu,* Volume 6, Issue 2, pp. 49-54.
10. Barkalov, A.A. (1998), Optimization principles of a logic circuit of Moore FSM [Printsipyi optimizatsii logicheskoy shemyi mikroprogrammnogo avtomata Mura], Kibernetika i sistemnyiy analiz, No. 1, pp. 65 – 72.
11. Yang, S. (1991), *Logic Synthesis and optimization benchmarks user guide*, Microelectonics Center of North Carolina, North Carolina, 43 p.