КІБЕРНЕТИКА

А.А. БАРКАЛОВ, Л.А. ТИТАРЕНКО, А.В. БАЕВ, А.В. МАТВИЕНКО

УДК 004.274

СОВМЕСТНОЕ ИСПОЛЬЗОВАНИЕ МЕТОДОВ СТРУКТУРНОЙ ДЕКОМПОЗИЦИИ ДЛЯ ОПТИМИЗАЦИИ СХЕМЫ МИКРОПРОГРАММНОГО АВТОМАТА МУРА

Аннотация. Предложен метод оптимизации аппаратурных затрат в схеме автомата Мура, реализуемой в базисе FPGA. Метод основан на одновременном использовании замены входов и преобразования кодов состояний в коды классов псевдоэквивалентных состояний. Такой подход приводит к трехуровневой схеме автомата. Приведен пример синтеза автомата Мура с использованием предложенного метода и выполнен анализ его положительных и отрицательных характеристик. Исследования на базе стандартных автоматов показали, что данный метод позволяет уменьшить аппаратурные затраты и потребляемую мощность при незначительной потере быстродействия.

Ключевые слова: автомат Мура, синтез, FPGA, LUT, структурная декомпозиция.

ВВЕДЕНИЕ

В состав цифровых систем входят различные последовательностные блоки, например, устройства управления [1, 2] или интерфейсы, обеспечивающие взаимодействие процессоров и аппаратных ускорителей встроенных систем [3, 4]. Для синтеза схем последовательностных блоков часто используется модель микропрограммного автомата (МПА) Мура [1, 5, 6].

Как правило, при реализации схем МПА возникает проблема уменьшения аппаратурных затрат [7–9], для решения которой необходимо найти схему с минимальным числом элементов и межсоединений [10, 11]. Однако эта задача имеет NP-сложность [6]. Поэтому на практике используются различные эвристики, позволяющие найти близкое к оптимальному решение за приемлемое время [12]. Для разработки таких эвристик нужно учитывать особенности модели МПА и элементного базиса [13].

В настоящей статье рассматривается задача реализации схем МПА Мура в базисе FPGA (field-programmable logic arrays) [14, 15]. Этот базис выбран, поскольку он широко применяется для реализации разнообразных цифровых систем [2, 9].

Предлагаемый в работе метод основан на совместном использовании двух методов структурной декомпозиции [13, 16] для уменьшения числа элементов табличного типа LUT (look-up table) в схеме МПА Мура и приводит к трехуровневой схеме микропрограммного автомата.

ОСОБЕННОСТИ АВТОМАТА И ЭЛЕМЕНТНОГО БАЗИСА

Автомат Мура можно представить вектором $S = \langle X, Y, A, \delta, \lambda, a_1 \rangle$ [1, 6], компонентами которого являются: множество входов $X = \{x_1, \dots, x_L\}$; множество выходов $Y = \{y_1, \dots, y_N\}$; множество внутренних состояний $A = \{a_1, \dots, a_M\}$; функция переходов δ; функция выходов λ; начальное состояние $a_1 \in A$. Функция δ по текущему состоянию и входному сигналу определяет следующее состояние, называемое состоянием перехода. Функция λ задает набор выходов $Y(a_m) \subseteq Y$ в каждом состоянии $a_m \in A$. Существует большое количество методов задания МПА [5, 6, 17]. В настоящей статье для этого выбран язык граф-схем алгоритма (ГСА) [1].

Как известно, ГСА Г состоит из операторных вершин, содержащих наборы выходов, и условных вершин, содержащих входы $x_i \in X$ [1]. Кроме того, ГСА Г включает начальную и конечную вершины, соответствующие состоянию $a_1 \in A$ автомата Мура (см. ГСА Γ_1 на рис. 1).

Согласно правилам [1] начальная (Start) и конечная (End) вершины отмечаются состоянием a_1. Каждая операторная вершина отмечается отдельным состоянием.

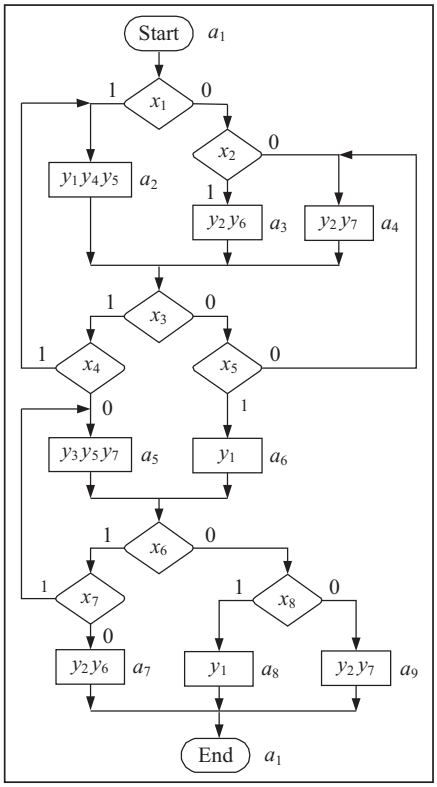

Рис. 1. Отмеченная граф-схема алгоритма Γ_1

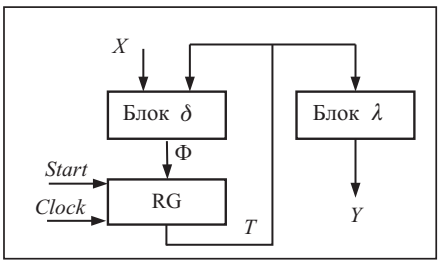

Рис. 2. Структурная схема МПА U_1

Анализ ГСА Γ_1 позволяет получить множества $X = \{x_1, \dots, x_8\}$, $Y = \{y_1, \dots, y_7\}$, $A = \{a_1, \dots, a_9\}$, имеющие параметры $L = 8$, $N = 7$ и $M = 9$ соответственно.

Для реализации схемы МПА Мура необходимо [1, 13]:

— выполнить кодирование состояний $a_m \in A$ кодами $K(a_m)$ и определить множества внутренних переменных T и функций возбуждения памяти (ФВП) Ф;

— построить прямую структурную таблицу (ПСТ) со столбцами a_m — исходное состояние, $K(a_m)$ — код состояния $a_m \in A$, a_s — состояние перехода, $K(a_s)$ — код состояния $a_s \in A_m$, X_h — входной сигнал, определяющий переход $\langle a_m, a_s \rangle$, Φ_h — набор функций возбуждения памяти, равных единице для перехода $\langle a_m, a_s \rangle$, h — номер перехода, $h \in \{1, \dots, H\}$. В столбцах a_m записываются наборы выходов $Y(a_m) \subseteq Y$;

— построить систему функций, задающих схему МПА Мура (на основе ПСТ)

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

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

Системы (1) и (2) являются основой для реализации схемы блоков δ и λ соответственно. Коды состояний хранятся в регистре (RG), состоящем из R триггеров:

$$R = \lceil \log_2 M \rceil. \tag{3}$$

Обозначим U_1 автомат Мура, схема которого показана на рис. 2.

Как видно из рис. 2, регистр RG управляется сигналами Ф, *Start* и *Clock*. Импульс *Start* устанавливает МПА в исходное состояние. Импульс синхронизации *Clock* разрешает изменение содержимого RG.

Для оптимизации блока δ можно использовать классы псевдоэквивалентных состояний (ПЭС) [13]. Состояния a_m, $a_s \in A$ являются ПЭС, если выполняется следующее условие:

$$\delta(a_m, X_h) = \delta(a_s, X_h), \ h \in \{1, \dots, H\}. \qquad (4)$$

Как следует из (4), для ПЭС совпадают функции переходов δ, однако функции выходов отличаются. Если условие (4) выполняется, то выходы операторных вершин ГСА, отмеченные состояниями a_m, $a_s \in A$ связаны с входом одной и той же вершины ГСА. Используя условие (4), можно найти разбиение Π_A множества состояний A на классы ПЭС: $\Pi_A = \{B_1, \dots, B_I\}$. Мощность множества Π_A находится в пределах $[M_0 + 1, M_0]$, где M_0 — число состояний эквивалентного автомата Мили [13].

Например, из ГСА Γ_1 можно найти разбиение $\Pi_A = \{B_1, \dots, B_4\}$, где $B_1 = \{a_1\}$, $B_2 = \{a_2, a_3, a_4\}$, $B_3 = \{a_5, a_6\}$, $B_4 = \{a_7, a_8, a_9\}$. В данном случае $I = M_0 + 1$, так как эквивалентный автомат Мили имеет $M_0 = 3$ состояния [13].

Автомат Мура имеет две особенности, которые можно использовать для уменьшения аппаратурных затрат [13, 18]: первая — наличие ПЭС, вторая — отсутствие непосредственной зависимости выходов от входов. Далее рассматривается, как можно использовать эти особенности.

Как показано в [13], первую особенность автомата Мура можно использовать для уменьшения числа аргументов в функциях системы (1), а вторую — для уменьшения числа аргументов в функциях системы (2), поскольку отсутствие зависимости выходов $y_n \in Y$ от входов $x_i \in X$ позволяет закодировать состояния так, чтобы исключить некоторые внутренние переменные из формул для системы (2).

Отметим, что внутренние переменные являются элементами множества $T = \{T_1, \dots, T_R\}$. Как правило, регистры состояния строят на синхронных триггерах типа D [19]. Поэтому множество ФВП имеет вид $\Phi = \{D_1, \dots, D_R\}$.

Рассмотрим особенности элементов LUT. В основном, ведущие производители микросхем FPGA [20, 21] используют элементы LUT на основе ячеек памяти типа SRAM. Элемент LUT состоит из 2^{S_L} ячеек и мультиплексора, имеющего S_L управляющих входов. Эти входы называются адресными, а входной вектор на входах рассматривается как адрес ячейки памяти.

Элементы LUT являются основой для построения различных конфигурируемых логических блоков (КЛБ) [15]. Архитектура этих блоков у разных производителей отличается. Для дальнейшего рассмотрения достаточно представить КЛБ в виде соединения элемента LUT, триггера и мультиплексора (рис. 3).

В зависимости от информации на адресных входах элемент LUT формирует функцию $f_C \in \{0, 1\}$. Выход LUT поступает на вход D-триггера. По сигналу *Clock* на выходе триггера формируется сигнал $f_R = f_C$. С использованием сигнала $f_S \in \{0, 1\}$ на выходе f формируется либо комбинационный ($f = f_C$), либо регистровый ($f = f_R$) выход КЛБ.

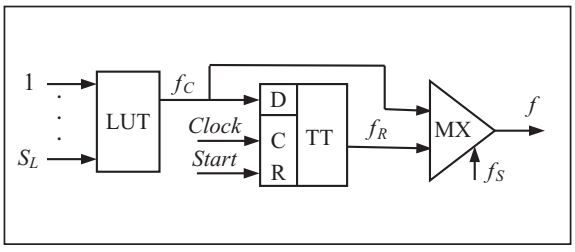

Рис. 3. Организация КЛБ на основе LUT

Основной особенностью элемента LUT является ограниченное число входов. На прак-

тике $S_L \le 6$ [20, 21]. Автоматы реальных цифровых систем могут иметь параметры $L \approx 30$, $R \approx 7$, $N \approx 50$. Таким образом, для большинства функций $f_i \in \Phi \cup Y$ может выполняться условие

$$NA(f_i) > S_L, \tag{5}$$

где $NA(f_i)$ — число аргументов функции $f_i \in \Phi \cup Y$.

Если (5) выполняется для некоторой функции, то для реализации соответствующей схемы используются методы функциональной декомпозиции (ФД) [22–24], в результате чего схемы МПА имеют много уровней логики и сложную систему межсоединений [22]. Побочным эффектом ФД является уменьшение быстродействия и увеличение энергопотребления [24] по сравнению с одноуровневыми схемами. Это обусловливает необходимость оптимизации характеристик схемы МПА.

МЕТОДЫ ОПТИМИЗАЦИИ СХЕМ МПА НА БАЗЕ ЭЛЕМЕНТОВ LUT

В настоящей работе рассматривается случай использования элементов LUT для реализации схем МПА Мура. Как известно, для этого можно применять конфигурируемые блоки памяти [20, 21]. Однако часто на практике на этих блоках реализуются табличные функции [12, 15]. В этом случае для реализации схем МПА можно использовать только элементы LUT и триггера.

Для оптимизации характеристик схем МПА Мура можно применять следующие методы:

— оптимальное кодирование состояний (ОКС);

— функциональная декомпозиция систем (1), (2);

— структурная декомпозиция (СД) схемы МПА.

Предназначением ОКС является нахождение таких кодов $K(a_m)$, которые позволяют улучшить одну (или несколько) характеристик схемы МПА [17], а именно площадь кристалла, занимаемую схемой МПА, быстродействие и потребляемую мощность. Первая характеристика определяется числом элементов и их межсоединений в схеме МПА. Как правило, площадь кристалла в значительной степени влияет на быстродействие и потребляемую мощность.

Для автомата Мура ОКС заключается в нахождении кодов ПЭС, позволяющих представить любой класс $B_i \in \Pi_A$ минимальным числом кодов R-мерного булева пространства [11, 13]. Для этого используется, например, алгоритм JEDI, распространяемый с системой SIS [10, 5].

По мере уменьшения размеров транзисторов увеличивается влияние межсоединений на быстродействие и потребляемую мощность [26, 27]. В связи с этим необходимо уменьшать число межсоединений в схеме МПА. Этого можно достичь минимизацией систем функций (1), (2) [7, 28]. Каждый аргумент функции соответствует одному межсоединению. Например, для оптимизации схемы блока λ (см. рис. 2) можно использовать кодирование состояний, основанное на методах [16, 28].

Функциональная декомпозиция является одним из наиболее эффективных средств реализации схем МПА в базисе FPGA [22–24]. В результате ФД начальная функция разбивается на все более и более простые части. Процесс завершается, когда любая часть функции имеет не больше, чем S_L аргументов. Существует много публикаций, посвященных этой проблеме. Укажем только работы [5, 23, 24], в которых рассмотрены классические и новые методы ФД.

Практически любая САПР, ориентированная на FPGA, включает в свой состав средства для ФД, которые можно найти, например, в академических системах ABC [29], SIS [30], PKmin [31], DEMAIN [32]. Программные средства для выполнения ФД включены в пакеты крупнейших производителей FPGA Xilinx [33] и Intel (Altera) [34].

Недостатком метода ФД является то, что трудно предугадать количество уровней элементов LUT и предвидеть степень сложности их межсоединений.

В настоящей работе предлагается метод, приводящий к трехуровневым схемам автоматов с регулярной системой межсоединений и основанный на СД схемы МПА.

Методы СД предназначены для устранения прямой зависимости функций $D_r \in \Phi$ (для автоматов Мили и Мура) и $y_n \in Y$ (для автоматов Мили) от входов $x_i \in X$ [13]. Для этого в структурную схему вводятся дополнительные блоки, реализующие отличные от (1) функции. Эти методы основаны на микропрограммных устройствах управления, описанных М. Уилксом еще в 1951 г.

В настоящее время известны следующие методы СД [13]:

— замена входов $x_i \in X$ дополнительными переменными $p_g \in P$, где $|P| \ll |X|$;

— кодирование наборов выходов;

— преобразование кодов объектов;

— кодирование термов ПСТ.

Рассмотрим первый из этих методов. Пусть $X(a_m)$ — множество входов, определяющих переходы из состояния $a_m \in A$. Найдем минимальное число $G = \max(|X(a_1)|, \ldots, |X(a_M)|)$ переменных $p_g \in P$, достаточное для замены входов [7].

Построим таблицу замены входов, имеющую столбцы $a_m \in A$ и строки $p_g \in P$. Переменная $x_i \in X$ записывается на пересечении строки p_g и столбца a_m, если вход $x_i \in X$ заменяется переменной $p_g \in P$ в состоянии $a_m \in A$. Эта таблица позволяет найти систему функций

$$P = P(T, X). \tag{6}$$

Замена $X \to P$ позволяет превратить систему (1) в систему

$$\Phi = \Phi(T, P). \tag{7}$$

При этом система (2) не изменяется.

Применение замены входов превращает МПА U_1 в автомат Мура U_2 (рис. 4), в котором блок P реализует систему (6), блок δ — систему (7), а блок λ — систему (2).

Такой подход применяется в автоматах, реализуемых на программируемых логических матрицах и заказных матрицах [1]. Однако он никогда не использовался в МПА Мура, схемы которых реализуются в базисе LUT. В настоящей работе предлагается использовать замену входов и в этом случае.

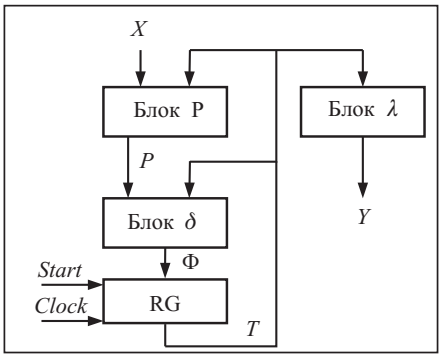

Рис. 4. Структурная схема МПА U_2

Кодирование наборов выходов сводится к замене наборов $Y_q \subseteq Y$ переменными $z_r \in Z$. При этом $|Z| = R_Q$, $R_Q = \lceil \log_2 Q \rceil$, где Q — число попарно различных наборов выходов [13]. Для использования этого метода необходимо найти системы функций

$$Z = Z(T, X), \tag{8}$$

$$Y = Y(Z). \tag{9}$$

Как следует из (6), (7), замена входов исключает прямую зависимость Ф от X, что приводит к уменьшению числа элементов LUT в блоке δ (по сравнению с U_1). Из систем (8), (9) следует, что кодирование выходов упрощает систему $Y(T, X)$, однако этот метод нецелесообразно использовать для МПА Мура, где нет прямой зависимости выходов Y от входов X.

Преобразование кодов объектов [13] основано на установлении зависимости, например, выходов Y от состояний A и некоторых дополнительных переменных, заменяющих входы $x_i \in X$. В настоящей работе предлагается использовать метод преобразования кодов состояний в коды классов ПЭС [13]. Этот подход применялся при реализации МПА в базисе программируемых логических интегральных схем [1] для уменьшения числа термов в системе функций возбуждения памяти. Для автоматов на базе элементов LUT такое преобразование целесообразно, если оно приводит к уменьшению числа аргументов функций $D_r \in \Phi$.

Кодирование термов ПСТ сводится к введению в схему МПА блока, формирующего коды строк таблицы переходов. При этом функции Φ и Y зависят от дополнительных переменных [1, 13]. Однако этот метод нецелесообразно применять для МПА Мура, у которого нет прямой зависимости функций $y_n \in Y$ от переменных $x_i \in X$.

Таким образом, в настоящей работе предлагается одновременно использовать методы замены входов X и преобразования входов состояний, причем этот подход применяется впервые для автоматов Мура, схемы которых реализуются в базисе FPGA.

ОСНОВНАЯ ИДЕЯ ПРЕДЛАГАЕМОГО МЕТОДА

Рассмотрим ГСА Γ_1 (см. рис. 1). Использовав определение ПЭС [13], найдем разбиение $\Pi_A = \{B_1, \ldots, B_I\}$, где $B_i \subseteq A$ — класс ПЭС. В данном случае имеем $I = 4$ класса ПЭС: $B_1 = \{a_1\}$, $B_2 = \{a_2, a_3, a_4\}$, $B_3 = \{a_5, a_6\}$, $B_4 = \{a_7, a_8, a_9\}$.

Поставим в соответствие каждому классу ПЭС $B_i \in \Pi_A$ двоичный код $K(B_i)$ разрядности R_A:

$$R_A = \lceil \log_2 I \rceil. \tag{10}$$

Используем для кодирования классов элементы $\tau_r \in \tau$, где $|\tau| = R_A$.

Очевидно, если $a_m, a_s \in B_i$, то $X(a_m) = X(a_s)$. Например, для класса $B_3 \in \Pi_A$ имеем $X(a_5) = X(a_6) = \{x_6, x_7, x_8\}$. Таким образом, замену входов можно выполнить не для состояний $a_m \in B_i$, а для классов $B_i \in \Pi_A$. Если выполняется условие

$$R_A < R, \tag{11}$$

то уменьшается число аргументов в функциях $p_g \in P$. При таком подходе функции (6) преобразуются в функции

$$P = P(\tau, X). \tag{12}$$

Очевидно, переменные $\tau_r \in \tau$ заменяют переменные $T_r \in T$ в функциях $D_r \in \Phi$. Теперь эти функции представляются системой

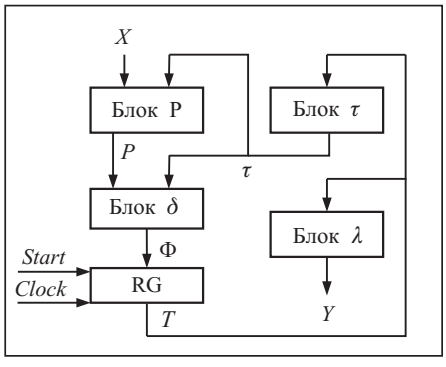

Рис. 5. Структурная схема МПА Мура U_3

$$\Phi = \Phi(\tau, P). \tag{13}$$

Для преобразования $K(a_m)$ в $K(B_i)$ необходимо построить систему функций

$$\tau = \tau(T). \tag{14}$$

Для реализации системы (14) требуются некоторые ресурсы кристалла FPGA.

В результате проведенного анализа можно предложить структурную схему МПА Мура U_3 (рис. 5).

В МПА U_3 блоки P, δ, τ реализуют системы (12)–(14) соответственно. Функ-

ции блока λ не изменяются для автоматов U_1–U_3.

В случае использования элементов LUT регистр RG распределен между триггерами КЛБ блока δ. Состоящий из элементов LUT блок назовем блоком LUTer.

На рис. 6 представлена структурная схема МПА U_3 на элементах LUT.

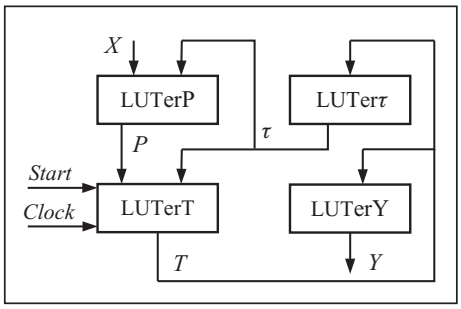

Рис. 6. Структурная схема МПА Мура U_3 на элементах LUT

Блок LUTerP реализует систему (12), генерируя переменные $p_g \in P$. Блок LUTerT объединяет в себе блок δ и регистр RG из рис. 5. Поэтому импульсы *Start* и *Clock* поступают непосредственно в блок LUTerT, а выходами этого блока являются переменные $T_r \in T$. Блок LUTerτ реализует систему (14), а LUTerY — систему (2).

В настоящей работе предлагается метод синтеза МПА U_3, который включает следующие этапы:

1) отметка граф-схемы алгоритма Г состояниями автомата Мура и формирование множества $A = \{a_1, \ldots, a_M\}$;

2) формирование разбиения Π_A множества состояний A на классы ПЭС;

3) кодирование классов $B_i \in \Pi_A$;

4) формирование таблицы замены входов $x_i \in X$ дополнительными переменными $p_g \in P$;

5) кодирование состояний $a_m \in A$, оптимизирующее системы (2) и (14);

6) формирование ПСТ автомата U_3;

7) формирование систем функций, задающих схему МПА;

8) реализация схемы в заданном элементном базисе.

Обозначим $U_i(\Gamma_j)$ тот факт, что МПА U_i реализуется по ГСА Γ_j. Рассмотрим пример синтеза МПА $U_3(\Gamma_1)$. Используем для реализации схемы МПА элементы LUT, имеющие $S_L = 5$ входов.

ПРИМЕР СИНТЕЗА МПА U_3

Два первых этапа предложенного метода синтеза выполнены ранее. В результате получены множества $A = \{a_1, \ldots, a_9\}$ и $\Pi_A = \{B_1, \ldots, B_4\}$. Начнем описание примера с выполнения этапа 3.

Использовав (10), получим $R_A = 2$ и $\tau = \{\tau_1, \tau_2\}$. Закодируем классы следующим образом: чем больше состояний включает класс, тем меньше единиц содержит его код, и получим коды $K(B_1) = 11$, $K(B_2) = 00$, $K(B_3) = 10$, $K(B_4) = 01$. Полагаем, что такой подход позволит оптимизировать систему (14).

Построим множества $X(B_i) \subseteq X$, включающие входы, которые определяют переходы из состояний $a_m \in B_i$. Из анализа ГСА Γ_1 можно найти, что $X(B_1) = \{x_1, x_2\}$, $X(B_2) = \{x_3, x_4, x_5\}$, $X(B_3) = \{x_6, x_7, x_8\}$, $X(B_4) = \varnothing$. Очевидно, имеем $G = \max(2, 3, 3, 0) = 3$, что дает множество дополнительных переменных $P = \{p_1, p_2, p_3\}$.

Таблица 1. Таблица замены входов МПА $U_3(\Gamma_1)$

p_g \ B_i	B_1	B_2	B_3	B_4
p_1	x_1	x_3	x_6	—
p_2	x_2	x_4	x_7	—
p_3	—	x_5	x_8	—

Таблица замены входов имеет столбцы $B_i \in \Pi_A$ и строки $p_g \in P$. Распределение входов для рассматриваемого примера выполняется тривиальным образом (табл. 1).

Закодируем состояния $a_m \in A$ так, чтобы максимально уменьшить число

литералов в функциях (2) и (14). Поскольку $M=9$, из (3) находим $R=4$, $T=\{T_1,\dots,T_4\}$, $\Phi=\{D_1,\dots,D_4\}$.

Построим системы функций $Y=Y(A)$ и $B=B(A)$, где $B=\Pi_A$. Например, выход y_1 формируется в состояниях a_2, a_6, a_8, откуда получаем формулу $y_1=A_2\vee A_6\vee A_8$. Далее, класс $B_1=\{a_1\}$, что дает $B_1=A_1$. Здесь A_m означает конъюнкцию внутренних переменных $T_r\in T$, соответствующую коду $K(a_m)$, где $a_m\in A$. На основе содержимого операторных вершин ГСА Γ_1 и классов $B_i\in\Pi_A$ получаем следующие системы функций:

$$y_1=A_2\vee A_6\vee A_8,\ y_2=A_3\vee A_4\vee A_7\vee A_9,\ y_3=A_5,$$
$$y_4=A_2,\ y_5=A_2\vee A_5,\ y_6=A_3\vee A_7,\ y_7=A_4\vee A_5\vee A_9; \tag{15}$$

$$B_1=A_1,\ B_2=A_2\vee A_3\vee A_4,\ B_3=A_5\vee A_6,\ B_4=A_7\vee A_8\vee A_9. \tag{16}$$

Преобразуем систему (16) в систему $\tau=\tau(B)$. Для этого необходимо использовать коды $K(B_i)$ следующим образом: $K(B_1)=11$, следовательно, $\tau_1=\tau_2=B$; $K(B_3)=10$, следовательно, $\tau_1=B_3$; $K(B_4)=01$, следовательно, $\tau_2=B_4$. Используя этот анализ и систему (16), получаем следующую систему функций:

$$\tau_1=B_1\vee B_3=A_1\vee A_5\vee A_6,$$
$$\tau_2=B_1\vee B_4=A_1\vee A_7\vee A_8\vee A_9. \tag{17}$$

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

Рис. 7. Коды состояния МПА $U_3(\Gamma_1)$

Совместная минимизация систем (15) и (17) является сложной комбинаторной задачей. Для ее решения можно применить методы из [6]. Один из вариантов кодирования состояний показан на рис. 7.

Для построения ПСТ МПА U_3 необходимо построить систему обобщенных формул перехода (ОФП) [13]. Система ОФП реализуется на основе ГСА Г. Например, для класса $B_1\in\Pi_A$ можно построить формулы переходов $a_7\to a_1$, $a_8\to a_1$ и $a_9\to a_1$ и заменить их одной ОФП: $B_4\to a_1$. Использовав такой подход, получим следующую систему ОФП для МПА U_3:

$$B_1\to x_1a_2\vee\overline{x}_1x_2a_3\vee\overline{x}_1\overline{x}_2a_4;$$
$$B_2\to x_3x_4a_2\vee x_3\overline{x}_4a_5\vee\overline{x}_3x_5a_6\vee\overline{x}_3\overline{x}_5a_4;$$
$$B_3\to x_6x_7a_5\vee x_6\overline{x}_7a_7\vee\overline{x}_6x_8a_8\vee\overline{x}_6\overline{x}_8a_9;\ B_4\to a_1.$$

Эта система определяет зависимость $\Phi=\Phi(\tau,X)$. Для получения системы (13) необходимо заменить входы $x_i\in X$ переменными $p_g\in P$, использовав табл. 1, что приведет к системе

$$B_1\to p_1p_2\vee\overline{p}_1p_2a_3\vee\overline{p}_1\overline{p}_2a_4;$$
$$B_2\to p_1p_2a_2\vee p_1\overline{p}_2a_5\vee\overline{p}_1p_3a_6\vee\overline{p}_1\overline{p}_3a_4;$$
$$B_3\to p_1p_2a_5\vee p_1\overline{p}_2a_7\vee\overline{p}_1p_3a_8\vee\overline{p}_1\overline{p}_3a_9;\ B_4\to a_1.$$

Эта система однозначно определяет столбцы ПСТ: B_i, $K(B_i)$, a_s, $K(a_s)$, P_h, Φ_h, h. Для МПА $U_3(\Gamma_1)$ ПСТ имеет $H=11$ строк (табл. 2). Коды состояний $a_s\in A$ приведены на рис. 7.

Таблица 2. Прямая структурная таблица автомата $U_3(\Gamma_1)$

B_i	$K(B_i)$	a_s	$K(a_s)$	P_h	Φ_h	h
B_1	11	a_2	1001	p_1	D_1D_4	1
		a_3	0101	$\overline{p_1}p_2$	D_2D_4	2
		a_4	1101	$\overline{p_1}\,\overline{p_2}$	$D_1D_3D_4$	3
B_2	00	a_2	1001	p_1p_2	D_1D_4	4
		a_5	0001	$p_1\overline{p_2}$	D_4	5
		a_6	0010	$\overline{p_1}p_3$	D_3	6
		a_4	1101	$\overline{p_1}\,\overline{p_3}$	$D_1D_2D_4$	7
B_3	10	a_5	0001	p_1p_2	D_4	8
		a_7	0100	$p_1\overline{p_2}$	D_2	9
		a_8	1000	$\overline{p_1}p_3$	D_1	10
		a_9	1100	$\overline{p_1}\,\overline{p_3}$	D_1D_2	11

Отметим, что переходы из состояний $a_m \in B_4$ в табл. 2 не включены, поскольку переходы происходят в состояние a_1 с кодом 0000 и для их выполнения функции $D_1 - D_4$ равны нулю.

Схема блока LUTerP строится на основе системы (12). Для рассматриваемого примера система $P(\tau, X)$ формируется на основе табл. 1 и кодов классов $K(B_i)$:

$$p_1 = B_1x_1 \vee B_2x_3 \vee B_3x_4 = \tau_1\tau_2x_1 \vee \overline{\tau_1}x_3 \vee \tau_1\overline{\tau_2}x_4;$$

$$p_2 = B_1x_2 \vee B_2x_4 \vee B_3x_7 = \tau_1\tau_1x_2 \vee \overline{\tau_1}x_4 \vee \tau_1\overline{\tau_2}x_7;$$

$$p_3 = B_2x_5 \vee B_3x_8 = \overline{\tau_1}x_5 \vee \tau_1\overline{\tau_2}x_8.$$

Для синтеза блока LUTerT используется система (13), формируемая на основе ПСТ. Например, из табл. 2 можно найти уравнение (с учетом минимизации)

$$D_4 = \tau_1\tau_2 \vee \overline{\tau_1}p_1 \vee \overline{\tau_1}\,\overline{p_1}\,\overline{p_3} \vee \tau_1\overline{\tau_2}p_1p_2.$$

Остальные уравнения системы (13) имеют аналогичный вид.

Блок LUTerτ задается системой (14). Для приведенного примера можно получить следующую систему:

$$\tau_1 = \overline{T_1}\,\overline{T_2},\ \tau_2 = \overline{T_3}\,\overline{T_4}. \tag{18}$$

Эта система получена из функций (17) с использованием кодов $K(a_m)$ из рис. 7 и правил минимизации [6].

Блок LUTerY задается системой (2). В рассматриваемом случае из системы (15) можно получить следующую систему функций:

$$y_1 = T_1\overline{T_2} \vee T_3;\ y_2 = T_2;\ y_3 = \overline{T_1}\,\overline{T_2}T_4;$$

$$y_4 = T_1\overline{T_2}T_4;\ y_5 = \overline{T_2}T_4;\ y_6 = \overline{T_1}T_2; \tag{19}$$

$$y_7 = T_1T_2 \vee \overline{T_1}\,\overline{T_2}T_4.$$

Последний этап метода — реализация схемы в виде сети элементов LUT и их межсоединений. Для каждого элемента формируется таблица истины, преобразуемая в общий поток данных (bit-stream) [2]. Решается задача размещения и трассировки, формируется поток данных для программируемых межсоединений. Общий

поток данных для элементов и межсоединений представляет собой конфигурацию [2], которая записывается во внешнюю или внутреннюю память. На этом этапе применяются САПР, такие как Vivado [33] фирмы Xilinx. (В настоящей работе этот этап для приведенного примера не рассматривается.)

АНАЛИЗ ПРЕДЛОЖЕННОГО МЕТОДА

Замена входов позволяет уменьшить число аргументов в функциях возбуждения памяти автомата. В приведенном примере при $S_L = 5$ и $G + R_A = 5$ выполняется условие $G + R_A \le S_L$, которое определяет наилучшую ситуацию, когда для реализации любой функции $D_r \in \Phi$ требуется только один LUT. При этом блок LUTerT состоит из R элементов.

Однако такое решение связано с определенными накладными расходами. Сравнение автоматов на рис. 2 и рис. 5 показывает, что:

— схема МПА U_3 включает дополнительные блоки (блок P и блок τ), которые используют некоторые ресурсы кристалла (элементы LUT и межсоединения);

— блок P добавляет дополнительную задержку в цикл работы МПА U_3 (по сравнению с МПА U_1).

Очевидно, использование предложенного метода целесообразно, если суммарное число элементов в блоках LUTerP и LUTerτ будет меньше, чем их число в схеме блока δ, эквивалентного МПА U_1. Кроме того, данный подход нельзя использовать, если он не обеспечивает требуемого быстродействия МПА.

Предложенный метод позволяет уменьшить число входов в блоке LUTerP. Если замена $X \to P$ выполняется для состояний, то LUTerP имеет $R + L$ входов. В описанном случае замена $X \to P$ выполняется для классов ПЭС, при этом LUTerP имеет $R_A + L$ входов. Если выполняется условие (11), то предложенный подход приводит к уменьшению числа входных переменных в блоке P по сравнению с эквивалентным МПА U_2.

Сравним эквивалентные автоматы U_2 и U_3. Пусть L_g — число входов $x_i \in X$, заменяемых переменной $p_g \in P$. Пусть для некоторой функции $p_g \in P$ выполняются условия

$$L_g + R > S_L; \tag{20}$$

$$L_g + R_A \le S_L. \tag{21}$$

В этом случае для реализации схемы, соответствующей функции $p_g \in P$, в МПА U_3 необходим только один LUT, а в МПА U_2 — несколько элементов. При этом в МПА U_2 необходимо несколько уровней логики для реализации функции $p_g \in P$. Это приводит к увеличению числа межсоединений и времени такта МПА U_2 по сравнению с МПА U_3.

Для рассматриваемого примера имеем $R = 4$, $R_A = 2$, $L_1 = L_2 = 3$, $L_3 = 2$ (см. табл. 2). Поскольку $S_L = 5$, условия (20) и (21) выполняются для всех функций $p_g \in P$. При этом LUTerP автомата $U_3(\Gamma_1)$ имеет $G = 3$ элемента LUT и один уровень логики. Очевидно, что такое решение оптимально. Оно достигнуто за счет предложенного перехода от состояний $a_m \in A$ к их классам $B_i \in \Pi_A$.

Если выполняется условие $R \le S_L$, то для реализации любой функции $y_n \in Y$ требуется только один LUT. Это условие выполняется для МПА $U_1(\Gamma_1)$–$U_3(\Gamma_1)$. Для всех этих МПА блок LUTerY имеет $y_n \in Y$ элементов LUT. При этом для улучшения качества схемы блока LUTerY необходимо уменьшить число аргументов в функциях (2). Для автомата U_1 это не всегда возможно, так как приоритетом при кодировании состояний является оптимизация блока δ [13].

Если выполняется условие (20), то состояния $a_m \in A$ автомата U_2 необходимо закодировать так, чтобы уменьшить число уровней схемы блока P. Для авто-

мата U_3 задачи оптимизации блоков LUTerP, LUTerT и LUTerY не являются взаимосвязанными. В МПА U_3 оптимизация схем блоков LUTerP и LUTerT связана с оптимальным кодированием классов $B_i \in \Pi_A$ [13]. При этом кодирование состояний должно быть направлено на уменьшение числа аргументов в функциях (2) и (14).

В общем случае блоки LUTerτ и LUTerY имеют $R \times (R_A + N)$ входов. В приведенном примере имеем $R_A = 2$, $N = 7$. Таким образом, для МПА $U_3(\Gamma_1)$ блоки LUTerτ и LUTerY могут иметь до $4 \times (2 + 7) = 36$ межсоединений. Как следует из (18), схема блока LUTerτ имеет четыре межсоединения. Из (19) следует, что схема блока LUTerY имеет 17 межсоединений, т.е. в рассматриваемом примере предлагаемый подход позволяет уменьшить число межсоединений в $36 / (4 + 17) = 1.71$ раза по сравнению с возможным максимумом.

Отметим, что межсоединения потребляют до 70 % электрической мощности [10]. Поэтому предложенный метод позволяет значительно уменьшить мощность, потребляемую схемой МПА. Кроме того, по мере уменьшения размеров транзисторов межсоединения существенно влияют на общую временную задержку, что приводит к уменьшению рабочей частоты.

Автомат U_1 имеет два уровня логики, а автомат U_3 — три уровня. Однако уменьшение числа межсоединений позволяет избежать значительной потери быстродействия. Кроме того, с использованием предлагаемого подхода уменьшается количество элементов LUT в схеме блока LUTerT, а также их уровней. Последнее еще больше нивелирует разницу в быстродействии эквивалентных автоматов U_1 и U_3.

Теоретические выводы авторов подтвердились результатами исследований, для которых использовалась библиотека стандартных автоматов [35], микросхемы семейства Virtex-7 фирмы Xilinx [36] и система проектирования Vivado 19.1 [33]. Изучались модели автоматов U_1, U_2 и U_3, для которых при кодировании состояний использовался алгоритм JEDI.

В библиотеке [35] представлены 48 автоматов Мили. Для проведения исследований осуществлялось преобразование тестовых автоматов в автоматы Мура, представленные в форме файлов KISS2 [35]. Далее разрабатывались модели автоматов на языке VHDL, которые использованы для входа в систему Vivado. В результате работы этой системы были получены рапорты, содержащие информацию о числе элементов LUT ($S_L = 6$), максимальной рабочей частоте и потребляемой мощности.

Результаты исследований показали, что предлагаемый метод позволяет обеспечить в среднем экономию числа LUT — 24 % и потребляемой мощности для эквивалентных автоматов U_2 — 32 %. Для автомата U_1 эти показатели равны 36 % и 51 % соответственно. При этом потери быстродействия составляют 6 % для автомата U_1 и 9 % — для автомата U_2.

Очевидно, что эти результаты получены только для микросхем Virtex-7 и автоматов из библиотеки [35]. Однако можно утверждать, что предложенный метод позволяет значительно уменьшить аппаратурные затраты и потребляемую мощность при незначительной потере быстродействия.

ЗАКЛЮЧЕНИЕ

В настоящей работе предлагается метод уменьшения аппаратурных затрат в схеме МПА Мура. Этот метод основан на совместном использовании замены методов входных переменных и преобразования кодов состояний в коды классов псевдоэквивалентных состояний. Кроме того, состояния МПА кодируются так, чтобы уменьшить число межсоединений между различными блоками схемы.

Как показали исследования с использованием стандартных автоматов [35], предлагаемый подход позволяет уменьшить число элементов LUT и потребляемую мощность по сравнению с известными методами. Однако увеличение числа уровней в схеме МПА приводит к уменьшению быстродействия. Если полученное быстродействие является недостаточным для какой-либо цифровой системы, то этот метод нельзя применять.

Отметим, что существуют методы оптимизации схем МПА без использования блока преобразователя кодов [13]. Поэтому в дальнейших исследованиях будет применен подобный подход для оптимизации трехуровневой схемы МПА Мура, реализуемой в базисе FPGA. Кроме того, планируется использовать эти подходы для оптимизации схем СМПА в базисе ASIC и PLD [37], а также при синтезе нейроподобных сетей [38].

СПИСОК ЛИТЕРАТУРЫ

1. Baranov S. Logic synthesis for control automata. Dordrecht: Kluwer Academic Publishers, 1994. 312 p.
2. Sklyarov V., Skliarova I., Barkalov A., Titarenko L. Synthesis and optimization of FPGA-based systems. Berlin: Springer, 2014. 432 p.
3. Barkalov A., Titarenko L., Mazurkiewicz M. Foundations of embedded systems. Springer Nature Switzerland, 2019. 180 p.
4. Marwedel P. Embedded systemdesign: Embedded systems, foundations of cyber-physical systems and the Internet of things. Berlin: Springer, 2018. 423 p.
5. Czerwinski R., Kania D. Finite state machines logic synthesis for complex programmable logic devices. Berlin: Springer, 2013. 172 p.
6. DeMicheli G. Synthesis and optimization of digital circuits. New York: McGraw-Hill, 1994. 576 p.
7. Баркалов А.А., Титаренко Л.А., Баев А.В., Матвиенко А.В. Смешанное кодирование наборов микроопераций в микропрограммном автомате. *Кибернетика и системный анализ*. 2020. Т. 56, № 3. С. 3–16.
8. Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. Москва: Горячая линия — ТЕЛЕКОМ, 2001. 636 с.
9. Skliarova I., Sklyarov V., Sudnitson A. Design of FPGA-based circuits using hierarchical finite state machines. Tallinn: TUT Press, 2012. 240 p.
10. Machado L., Cjrtadella J. Support - reducing decomposition for FPGA mapping. *IEEE Transactions on Computer — Aided Design of Integrated Circuits and Systems*. 2020. Vol. 39, N 1. P. 213–224.
11. Barkalov A., Titarenko L. Logic synthesis for FSM-based control units. Berlin: Springer, 2009. 233 p.
12. Grout I. Digital systems design with FPGAs and CPLDs. Amsterdam: Elsevier, 2008. 784 p.
13. Barkalov A., Titarenko L., Mielcatek K., Chmielewski S. Logic synthesis for FPGA-based control units: Structural decomposition in Logic design. Berlin: Springer, 2020. 247 p.
14. Грушницкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем с использованием микросхем программируемой логики. СПб: БХВ-Петербург, 2002. 608 с.
15. Maxfield C. The design warrior's guide to FPGAs. Orlando: Academic Press, 2004. 542 p.
16. Баркалов А.А., Титаренко Л.А. Преобразование кодов в композиционных микропрограммных устройств управления. *Кибернетика и системный анализ*. 2011. № 5. С. 107–118.
17. Kubica M., Kania D. Area-oriented technology mapping for LUT-based logic blocks. Int. Journal of Applied Mathematics and Computer Science. 2017. Vol. 27, N 1. P. 207–222.
18. Kolopienczyk M., Titarenko L., Barkalov A. Design of EMB-based Moore FSMs. *Journal of Circuits, Systems and Computers*. 2017. Vol. 21, N 7. P. 1–23.
19. Tiwari A., Tomko K. Saving power by mapping finite state machines into embedded memory blocks in FPGAs. *Proc. Design, Automation and Test in Europe Conference and Exhibition* (Paris, France, 6–20 Feb. 2004). 2004. Vol. 2. P. 916–921.

20. White paper FPGA architecture. URL: www.altera.com.
21. UG473 (v1.14) July 3, 2019. URL: www.xilinx.com.
22. Rawski M., Tomaszewicz P., Borowski G., Luba T. Logic synthesis method of digital circuits designed for implementation with embedded memory blocks on FPGAs. *Design of Digital Systems and Devises*. Adamski M., Barkalov A., Wegrzyn M. (Eds.). *Lecture Notes in Electrical Engineering.* Vol. 79. Berlin: Springer, 2011. P. 121–144.
23. Rawski M., Selvaraj H., Luba T. An application of functional decomposition in ROM-based FSM implementation in FPGA devices. *Journal of System Architecture.* 2005. Vol. 51, Iss. 6–7. P. 424–434.
24. Nowicka M., Luba T., Rawski M. FPGA-based decomposition of boolean functions: Algorithms and implementations. *Proc. of the 6th International Conference on Advanced Computer Systems* (Szczecin, 1999). P. 502–509.
25. Barkalov A., Titarenko L., Chmielewski S. Mixed encoding of collections of output variables for LUT-based FSMs. *Journal of Circuits, Systems and Computers*. 2019. Vol. 28, N 8. P. 1–21.
26. Garcia-Vargas I., Senhadji-Navarro R., JimБnez-Moreno G., Civit-Balcells A., Guerra-Gutierrezz P. ROM-based finite state machines implementation in low cost FPGAs. *IEEE Intern. Simp. on Industrial Electronics (ISIE'07)* (Vigo, 2007). 2007. P. 2342–2347.
27. Garcia-Vargas L., Senhaji-Navarro R. Finite state machines with input multiplexing: A performance study. I*EEE Transactions on CAD of Integrated Circuits and Systems.* 2015. Vol. 34, Iss. 5. P. 867–871.
28. Баркалов А.А., Титаренко Л.А., Ефименко К.Н. Оптимизация схем композиционных микропрограммных устройств управления. *Кибернетика и системный анализ*. 2011. № 1. С. 179–188.
29. ABC System (n.d.). URL: https://people.eecs.berkeley.edu/~alanmi/abc/. Accessed: January, 2020.
30. Sentowich E.M., Singh K.J., Lavango L., Moon C., Murgai R., Saldanha A., Savoj H., Stephan P.R., Bryton R.K., Sanjiovanni-Vincentelli A. SIS: A system for sequential circuit synthesis. Technical Report. Berkely: University of California, 1992.
31. Michalski T., Kokosinski Z. Functional decomposition of combionational logic circuits with PKmin, Technical Transactions. *Electrical Engeneering = Czasopismo Technicze. Elektrotechnika* (Iss 2-E). 2016. P. 191–202.
32. DEMAIN. URL: www.zpt.tele.pw.edu.pl/oprogramowanie/demain.html.
33. Vivado (2020). URL: https://www.xilinx.com/design sub tools/vivado.html. Accessed January, 2020.
34. QuartusII (2020). URL: https://www.intel.com/content/www/us/en/programmable/downloads/software/quartus-ii-we/121.html. Accessed January, 2020.
35. Yang S. Logic synthesis and optimization benchmarks user guide. Version 3.0. Techn. Rep. Microelectronics Center of North Carolina, 1991. 43 p.
36. Xilinx (2020). Virtex-5 Family Overview, PDF, Xilinx Corporation. URL: https://www.xilinx.com/support/documentation/data sub sheets/ds100.pdf/.
37. Palagin A.V., Opanasenko V.N., Kryvyi S.L. Resource and energy optimization oriented development of FPGA-based adaptive logical networks for classification problem. *Green IT Engineering: Components, Networks and Systems Implementation*. 2017. Vol. 105. P. 195–218. DOI: 10.1007 978-3-319-55595-9_10.
38. Opanasenko V.N., Kryvyi S.L. Synthesis of neural-like networks on the basis of conversion of cyclic Hamming codes. *Cybernetics and Systems Analysis.* 2017. Vol. 53, N 4. P. 627–635. https://doi.org/10.1007/s10559-017-9965-z.

Надійшла до редакції 30.09.2020

О.О. Баркалов, Л.О. Тітаренко, А.В. Баєв, О.В. Матвієнко
СПІЛЬНЕ ВИКОРИСТАННЯ МЕТОДІВ СТРУКТУРНОЇ ДЕКОМПОЗИЦІЇ ДЛЯ ОПТИМІЗАЦІЇ СХЕМИ МІКРОПРОГРАМНОГО АВТОМАТА МУРА

Анотація. Запропоновано метод оптимізації апаратурних витрат в схемі автомата Мура, що реалізується в базисі FPGA. Метод ґрунтується на одночасному використанні заміни входів і перетворення кодів станів у коди класів псевдоеквівалентних станів. Такий підхід призводить до трирівневої схеми автомата. Наведено приклад синтезу автомата Мура з використанням запропонованого методу і виконано аналіз позитивних і негативних його характеристик. Дослідження на базі стандартних автоматів показали, що запропонований метод дає змогу зменшити апаратурні витрати і споживану потужність із незначною втратою швидкодії.

Ключові слова: автомат Мура, синтез, FPGA, LUT, структурна декомпозиція.

A.A. Barkalov, L.A. Titarenko, A.V. Baiev, A.V. Matviienko
JOINT USING METHODS OF STRUCTURAL DECOMPOSITION FOR OPTIMIZING CIRCUIT OF MOORE FSM

Abstract. A method is proposed for optimizing hardware amount in the circuit of Moore FSM implemented with FPGA. The method is based on joint using replacement of inputs and transformation of state codes into codes of classes of pseudoequivalent states. This approach leads to a three-level circuit of FSM. There is shown an example of synthesis of Moore FSM with application of the proposed method. Analysis of positive and negative features of the proposed method is conducted. The researches on the base of standard benchmark FSM show that the proposed method allows reducing hardware amount and consumed power with insignificant degradation of FSM performance.

Keywords: Moore FSM, synthesis, FPGA, LUT, structural decomposition.

Баркалов Александр Александрович,
доктор техн. наук, профессор Университета Зеленогурского (Польша); профессор Донецкого национального университета имени Василия Стуса, Винница, e-mail: A.Barkalov@iie.uz.zgora.pl.

Титаренко Лариса Александровна,
доктор техн. наук, профессор Университета Зеленогурского (Польша); профессор Харьковского национального университета радиоэлектроники, e-mail: L.Titarenko@iie.uz.zgora.pl.

Баев Артем Викторович,
кандидат физ. мат. наук, исполняющий обязанности декана Донецкого национального университета имени Василия Стуса; фирма Peoly, Винница, e-mail: a.baev@donnu.edu.ua.

Матвиенко Александр Владимирович,
научный сотрудник Института кибернетики им. В.М. Глушкова НАН Украины, Киев,
e-mail: avmatv@ukr.net.