УДК 004.3

А.А. Баркалов (д.т.н, профессор),
А.А. Красичков (к.т.н., доцент), **А.Н. Мирошкин** (ассистент)
Кафедра компьютерной инженерии ДонНТУ
A.Barkalov@iie.uz.zgora.pl
krasich@cs.dgtu.donetsk.ua
miroshkin@cs.dgtu.donetsk.ua

МОДИФИКАЦИЯ МЕТОДА СИНТЕЗА КОМПОЗИЦИОННОГО УСТРОЙСТВА УПРАВЛЕНИЯ ДЛЯ РЕАЛИЗАЦИИ В БАЗИСЕ FPGA

В статье предложен метод синтеза для композиционных микропрограммных устройств управления с разделением кодов и модификацией операторных линейных цепей граф-схемы алгоритма управления. Метод направлен на уменьшение аппаратурных затрат (LUT-элементов базиса FPGA) при реализации устройства. Уменьшение сложности схемы блока адресации микрокоманд достигается за счет введения дополнительных операторных вершин, содержащих коды классов псевдоэквивалентных операторных линейных цепей. Приведены условия целесообразности и пример применения данного метода.

Устройство управления, разделение кодов, алгоритм, FPGA

Введение

Использование любых элементных базисов для реализации схем устройств управления обуславливает необходимость учитывать не только особенность реализуемого алгоритма управления, но и особенности базиса. При необходимости реализовать комплекс операционных и управляющего устройств на одном кристалле, задача уменьшения аппаратурных затрат становится очень актуальной [1]. Среди возможных путей решения таких задач можно отметить уменьшение затрат на реализацию управляющего устройства за счет использования псевдоэквивалентных состояний реализуемого устройства для уменьшения разрядности входов схемы адресации микрокоманд, линейный характер ГСА, задающей алгоритм управления [2], а также наличие блоков встроенной памяти, что позволяет реализовать устройство управления на одном кристалле.

Для реализации алгоритмов управления, доля операторных вершин в которых больше 75%, целесообразно использовать УУ класса композиционных микропрограммных устройств управления (КМУУ) [2]. В

качестве элементного базиса для реализации схем устройств управления настоящее время широко используется базис FPGA (Field-Programmable Gate Arrays), элементом которого являются LUT-элементы (LUT – Look-Up Table, реализация любой функции от определенного количества переменных) [3, 4]. В настоящей работе предлагается один из путей решения задачи уменьшения аппаратурных затрат при реализации управляющего устройства в виде КМУУ с разделением кодов и модификацией элементарных операторных линейных цепей (ОЛЦ) в базисе FPGA.

Целью исследования является сокращение разрядности входа комбинационной схемы КМУУ за счет введения в ОЛЦ операторных вершин, содержащих коды классов псевдоэквивалентных состояний автомата. Задачей исследования является разработка модификации метода синтеза КМУУ, позволяющего уменьшить число LUT-элементов в схеме адресации микрокоманд. При этом алгоритм управления представляется в виде граф-схемы алгоритма (ГСА) [2].

Основные положения

Граф-схема алгоритма управления состоит из операторных и условных вершин, образующих множества E_1 и E_2 соответственно, а также множества дуг E. Начальную вершину ГСА обозначим b_0, конечную – b_E. Операторная вершина $b_q \in E_1$ содержит набор микроопераций $Y(b_q) \subseteq Y$, где $Y = \{y_1, ...y_N\}$ – множество микроопераций (выходных сигналов). Условная вершина $b_g \in E_2$ содержит один элемент $X(b_g)$ множества логических условий $X = \{x_1, ...x_L\}$. Если количество операторных вершин составляет 75% и более от общего количества вершин, речь идет о линейной ГСА.

Операторная линейная цепь является последовательностью операторных вершин граф-схемы алгоритма. Каждая ОЛЦ α_g имеет произвольное число входов I_g^i и только один выход Q_g. Формальные определения ОЛЦ, их входов и выходов можно найти в [5]. ОЛЦ, которая имеет один вход и один выход, называется элементарной [2].

Операторные линейные цепи, выходы которых связаны с входом одной вершины, называются псевдоэквивалентными (ПОЛЦ). Множество таких цепей образуют класс B_i псевдоэквивалентных ОЛЦ.

Пусть ГСА содержит G элементарных ОЛЦ α_g, которые составляют множество C. Для кодирования элементов этого множества достаточно

$$R_1 = \lceil \log_2 G \rceil \quad (1)$$

бит. Количество компонент в ОЛЦ α_g обозначим F_g. Максимальная длина $Q = \max(F_1, ...F_G)$ линейной цепи определяет разрядность кода компоненты ОЛЦ:

$$R_2 = \lceil \log_2 Q \rceil. \quad (2)$$

Используем для кодирования элементарных ОЛЦ элементы $\tau_r \in \tau$, а для кодирования компонент – элементы $T_r \in T$, где $|\tau| = R_1$, $|T| = R_2$. Кодирование компонент выполняется в естественном порядке, то есть

$$K(b_{gi}) = K(b_{gi-1}) + 1, \quad (3)$$

где $g = 1, ..., G$, $i = 1, ..., F_g$.

Отметим, что каждая вершина $b_q \in E_1$ соответствует микрокоманде MI_q, хранимой в управляющей памяти (УП) по адресу $A(b_q) = A_q$. Если адрес A_q получается путем конкатенации кода ОЛЦ и кода компоненты, то это называется разделением кодов [2]. Общая разрядность адреса в этом случае равна

$$R_A = R_1 + R_2. \quad (4)$$

Для интерпретации ГСА может быть использована модель композиционного микропрограммного устройства управления с элементарными ОЛЦ и разделением кодов (Рис. 1), обозначаемая в дальнейшем U_1.

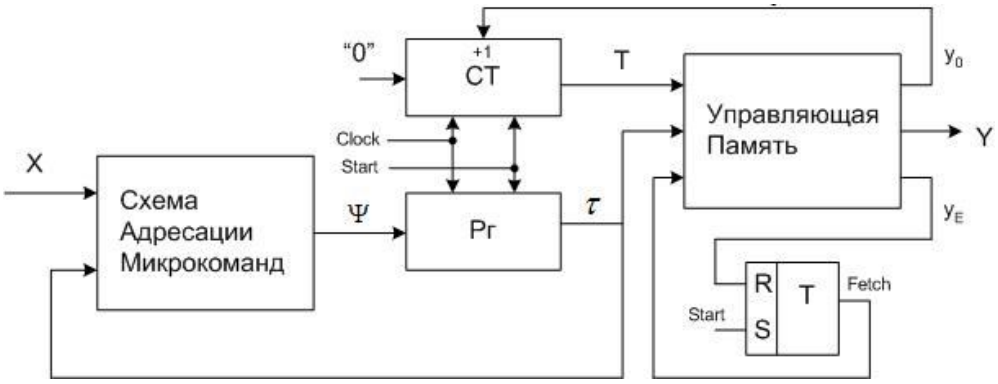


Рисунок 1 – Структурная схема КМУУ с элементарными ОЛЦ и разделением кодов

В КМУУ U_1 схема адресации микрокоманд (САМ) реализует систему функций возбуждения регистра Рг

$$\Psi = \Psi(X, \tau). \quad (5)$$

По сигналу Start в Рг заносится начальный адрес микропрограммы, счетчик переводится в нулевое состояние, а триггер выборки ТВ

устанавливается в состояние «1». При этом сигнал $\mathrm{Fetch} = 1$, что разрешает выборку микрокоманд из управляющей памяти (УП). Если считанная микрокоманда не соответствует выходу ОЛЦ, то одновременно с микрооперациями $Y(b_q)$ формируется сигнал y_0. Если $y_0 = 1$, то к содержимому СТ прибавляется единица и адресуется следующая компонента текущей ОЛЦ. Если выход ОЛЦ достигнут, то $y_0 = 0$. При этом адрес входа следующей ОЛЦ формируется схемой САМ. При достижении окончания микропрограммы формируется сигнал y_E, триггер ТВ обнуляется и выборка микрокоманд прекращается.

На вход R счетчика СТ следует подавать сигнал $\overline{y_0} \vee \mathrm{Start}$, что обеспечит перевод счетчика в нулевое состояние каждый раз при включении питания или при окончании очередной ОЛЦ.

Число термов в схеме САМ может быть уменьшено путем введения преобразователя кодов ОЛЦ в коды классов псевдоэквивалентных ОЛЦ (ПОЛЦ) [2]. Однако реализация такого преобразователя требует дополнительных LUT-элементов FPGA.

В настоящей работе предлагается уменьшить сложность преобразователя кодов путем введения дополнительных вершин, содержащих коды класса псевдоэквивалентных состояний автомата, в ОЛЦ алгоритма. Для хранения дополнительных слов микропрограммы используются свободные ресурсы встроенных блоков памяти.

Основная идея предлагаемого метода

В исходной ГСА множество C_1, состоящее из элементарных ОЛЦ α_g, не связанных с конечной вершиной, разбивается на классы $B_i \in \Pi_C$ псевдоэквивалентных ОЛЦ. Каждому классу B_i в соответствие ставится двоичный код $K(B_i)$ разрядности

$$R_3 = \lceil \log_2 I \rceil, \qquad (6)$$

где I – количество классов ПОЛЦ. После вершины-выхода каждой ОЛЦ $\alpha_g \in C_1$ вводится дополнительная вершина, содержащая код $K(B_i)$ текущей ОЛЦ. Для кодирования компонент модифицированных ОЛЦ достаточно

$$R'_2 = \lceil \log_2 Q' \rceil \qquad (7)$$

разрядов, где Q' - максимальное количество вершин в ОЛЦ после модификации. Причем, $R'_2 \geq R_2$.

Реконфигурируемые блоки памяти в микросхемах FPGA могут быть сконфигурированы для выполнения определенных задач. Так, в

микросхемах Stratix III блоки встроенной памяти могут быть настроены на одну из следующих конфигураций: 16K×8, 8K×16, 4K×32, 2K×64, 16K×9, 8K×18, 4K×36, 2K×72 [4]. Общее количество контактов схемы памяти является величиной постоянной, следовательно, существуют ограничения на применение предложенного метода модификации ОЛЦ алгоритма управления. Если обозначить количество контактов одного блока EMB через N_c, то

$$n_1 = N_c - R_A \tag{8}$$

свободных выходов каждого блока могут быть использованы для генерации микрокоманд. Под R_A подразумевается разрядность адреса микрокоманды для КМУУ с общей памятью [5]:

$$R_A = \lceil \log_2(|E_1| + 1) \rceil, \tag{9}$$

где E_1 - общее количество состояний автомата, константа «1» вводится для обозначения начального (конечного) состояния. Для реализации всех выходных сигналов автомата необходимо и достаточно

$$N_{EMB} = \left\lceil \frac{N+2}{n_1} \right\rceil \tag{10}$$

блоков встроенной памяти, где N - количество разрядов для кодирования микроопераций согласно унитарной стратегии кодирования [2], а константа 2 учитывает дополнительные переменные y_0 и y_E. Общее количество незадействованных выходов памяти определяется как

$$n_2 = n_1 * \left\lceil \frac{N+2}{n_1} \right\rceil - (N+2). \tag{11}$$

При модификации ОЛЦ может быть увеличена разрядность адреса компоненты ОЛЦ, в этом случае для реализации выходных функций будут задействованы $(n_1 - 1)$ выходов блока EMB. Если при уменьшении количества разрядов для реализации выходных функций количество используемых при этом блоков памяти останется неизменным, т.е.:

$$\left\lceil \frac{N+2}{n_1} \right\rceil * n_1 = \left\lceil \frac{N+2}{n_1 - 1} \right\rceil * (n_1 - 1), \tag{12}$$

то применение предложенного метода целесообразно. Следовательно, условием возможности применения предложенного метода является выполнение одного из условий:

$$\begin{cases} R'_2 = R_2; \\ \left\lceil \frac{N+2}{n_1} \right\rceil * n_1 = \left\lceil \frac{N+2}{n_1 - 1} \right\rceil * (n_1 - 1). \end{cases} \tag{13}$$

Первое равенство выполняется, если модификация ОЛЦ не увеличивает разрядность адреса их компонент, второе – при невыполнении первого обеспечивает размещение всех «вытесненных» выходных сигналов в последнем блоке ЕМВ без увеличения количества таких блоков. Если выполняется одно и/или два отношения из (13), то можно выполнить модификацию ОЛЦ, что ведет к КМУУ U_2 (Рис.2).

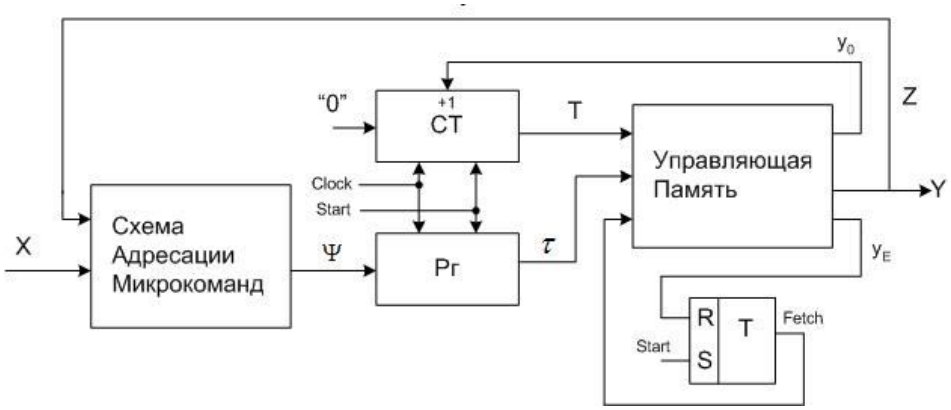


Рисунок 2 – Структурная схема КМУУ с модификацией элементарных ОЛЦ и разделением кодов

В КМУУ U_2 переменные $z_r \in Z$, где $|Z| = R_3$, представляют собой разряды кода $K(B_i)$. Схема САМ реализует функцию

$$\Psi = \Psi(Z, X). \qquad (14)$$

Остальные блоки КМУУ U_2 выполняют те же функции, что и одноименные блоки КМУУ U_1. Отметим, что блоки САМ, СТ, РГ, ТВ реализуются в составе микросхемы FPGA. Для реализации блока УП используются блоки встроенной памяти (embedded ROM), входящие в состав микросхемы FPGA.

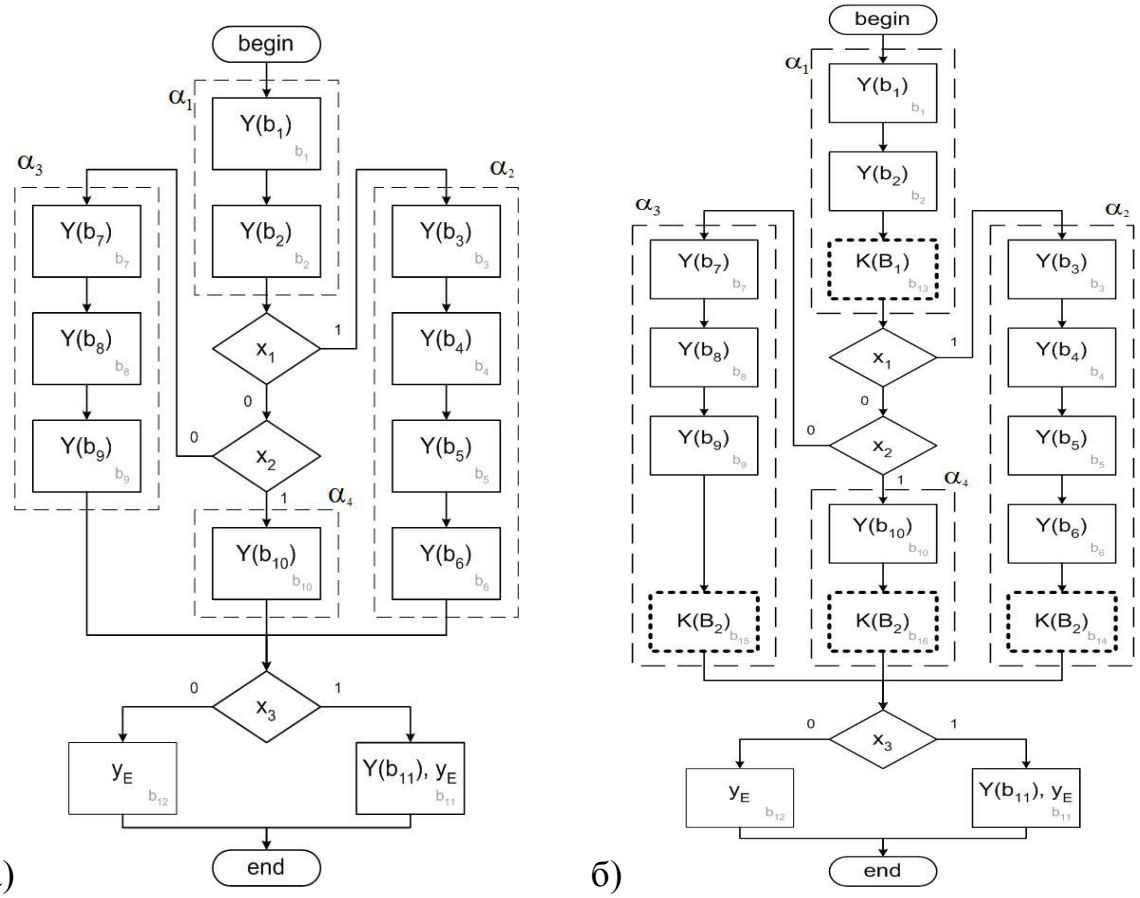
В настоящей работе предлагается метод синтеза КМУУ U_2, включающий следующие этапы:

1. Формирование множеств C, C_1 и Π_C для ГСА Γ.
2. Включение дополнительных вершин, содержащих коды $K(B_i)$, в ОЛЦ α_g.
3. Кодирование ОЛЦ, их компонент и классов $B_i \in \Pi_C$.
4. Формирование содержимого управляющей памяти.
5. Формирование таблицы переходов КМУУ и функции $\Psi = \Psi(Z, X)$.
6. Синтез логической схемы КМУУ.

Пример применения метода

Рассмотрим применение метода на примере ГСА Γ_1 (рис. 3). Множество $C = \{\alpha_1,...,\alpha_6\}$ – элементарные ОЛЦ алгоритма, $C_1 = C \setminus \{\alpha_5, \alpha_6\}$, $\Pi_C = \{B_1, B_2\}$, где $B_1 = \{\alpha_1\}$, $B_2 = \{\alpha_2, \alpha_3, \alpha_4\}$. Следовательно, количество ОЛЦ $G = 6$, для кодирования которых используются $R_1 = 3$ переменных их множества $\tau = \{\tau_1, \tau_2, \tau_3\}$. Максимальная длина ОЛЦ $Q = 4$, для кодирования которых достаточно $R_2 = 2$ переменных из множества $T = \{T_1, T_2\}$. Общее количество операторных вершин $|E_1| = 12$, что требует разрядности $R_A = 4$ адреса слова в УП. Для кодирования $I = 2$ классов $B_i \in \Pi_C$ достаточно $R_3 = 1$ переменной.

Кодирование ОЛЦ $\alpha_g \in C$ и их компонент выполним с учетом ограничения (3). Адреса $A(b_q)$ микрокоманд КМУУ $U_2(\Gamma_1)$ приведены в табл. 1. Здесь и далее символ $U_i(\Gamma_j)$ означает, что КМУУ U_i интерпретирует ГСА Γ_j.



а) б)

Рисунок 3 – Исходная ГСА Γ_1 до (а) и после (б) введения дополнительных вершин, содержащих коды ПОЛЦ

Таблица 1. Адреса микрокоманд устройства управления согласно ГСА Γ_1

τ_1,τ_2,τ_3 / $(T_0)T_1T_2$	000	001	010	011	100	101
(0)00	b_1	b_3	b_7	b_{10}	b_{11}	b_{12}
(0)01	b_2	b_4	b_8	b_{16}	–	–
(0)10	b_{13}	b_5	b_9	–	–	–
(0)11	–	b_6	b_{15}	–	–	–
(1)00	–	b_{14}	–	–	–	–

Адрес образуется путем конкатенации адреса ОЛЦ и адреса компоненты, т.е., из табл. 1, например, $A(b_2) = 00001$, $A(b_9) = 01010$ и так далее. Модификация ОЛЦ заключается во введении дополнительных операторных вершин (на рис. 3,б они выделены полужирной штриховой границей, а в табл. 1 – цветом) в конец каждой ОЛЦ, не связанной с конечной вершиной. В каждую такую вершину записывается код класса текущей ПОЛЦ. После преобразования длина максимальной ОЛЦ равна $Q' = 5$, что обуславливает необходимость дополнительного разряда для кодирования компонент ОЛЦ.

Коды классов $B_i \in \Pi_C$ зададим следующим образом: $K(B_1) = 0$, $K(B_2) = 1$. Формат микрокоманд КМУУ U_2 включает поля y_0, y_E и FY или y_0 и FB, где поле FY содержит код набора микроопераций, FB – код класса $B_i \in \Pi_C$. Если $y_0 = 0$, то содержимое микрокоманды интерпретируется как код класса ПОЛЦ (поле FB).

Пусть количество контактов одного блока EMB равно $N_C = 25$ [4], тогда согласно формуле (8), в каждом блоке свободны по $n_1 = 25 - 5 = 20$ разрядов. Предположим, что устройство вырабатывает $N = 30$ различных микроопераций, для чего задействовано $N_{EMB} = \left\lceil \frac{30+2}{20} \right\rceil = 2$ блока встроенной памяти (10). При этом, согласно (11), в последнем блоке $n_2 = 8$ контактов остаются свободными. Второе неравенство условия (13) выполняется, что делает применение рассматриваемой модификации метода целесообразным.

Содержимое УП КМУУ $U_2(\Gamma_1)$ показано в табл. 2.

Таблица 2. Содержимое управляющей памяти КМУУ $U_2(\Gamma_1)$

$A(b_q)$	y_0	FY		y_E	$A(b_q)$	y_0	FY		y_E
		FB	*	0			FB	*	0
$A(b_1)$	1	$Y(b_1)$		0	$A(b_7)$	1	$Y(b_7)$		0
$A(b_2)$	1	$Y(b_2)$		0	$A(b_8)$	1	$Y(b_8)$		0
$A(b_{13})$	0	$K(B_1)$	*	0	$A(b_9)$	1	$Y(b_9)$		0
$A(b_3)$	1	$Y(b_3)$		0	$A(b_{15})$	0	$K(B_2)$	*	0
$A(b_4)$	1	$Y(b_4)$		0	$A(b_{10})$	1	$Y(b_{10})$		0
$A(b_5)$	1	$Y(b_5)$		0	$A(b_{16})$	0	$K(B_2)$		*
$A(b_6)$	1	$Y(b_6)$		0	$A(b_{11})$	*	$Y(b_{11})$		1
$A(b_{14})$	0	$K(B_2)$	*	0	$A(b_{12})$	*	-		1

Как можно увидеть из табл. 2, если вершина $b_q \in E_1$ не является выходом ОЛЦ $\alpha_g \in C_1$, то в ячейку с адресом $A(b_q)$ записывается микрооперация y_0. В противном случае в эту ячейку записывается код $K(B_i)$, где $\alpha_g \in B_i$. Если вершина $b_q \in E_1$ связана с конечной вершиной ГСА, то в ячейку с адресом $A(b_q)$ заносится микрооперация y_E.

Переходы из выходов ОЛЦ $\alpha_g \in C_1$ представлены следующей системой обобщенных формул переходов [2]:

$$B_1 \to x_1 b_3 \vee \overline{x_1} x_2 b_{10} \vee \overline{x_1}\,\overline{x_2} b_7;$$
$$B_2 \to x_3 b_{11} \vee \overline{x_3} b_{12}. \qquad (15)$$

Подобная система является основой для формирования таблицы переходов КМУУ U_2 со столбцами: B_i, $K(B_i)$, b_q, $A(b_q)$, X_h, Ψ_h, h. Назначение столбцов ясно из табл. 3, задающей переходы для класса $B_1 \in \Pi_C$.

Таблица 3. Фрагмент таблицы переходов КМУУ $U_2(\Gamma_1)$

B_i	$K(B_i)$	b_q	$A(b_q)$			X_h	Ψ_h	h
	z_1		τ_1	τ_2	τ_3			
B_1	0	b_3	0	0	1	x_1	D_3	1
		b_7	0	1	0	$\overline{x_1}\,\overline{x_2}$	D_2	2
		b_{10}	0	1	1	$\overline{x_1} x_2$	D_2, D_3	3

Адреса микрокоманд берутся из табл. 1. Отметим, что $\Psi = \{D_1, D_2, D_3\}$. Общее число строк $H_2(\Gamma_j)$ в таблице переходов КМУУ

$U_2(\Gamma_j)$ совпадает с числом термов в системе обобщенных формул переходов. В нашем примере, $H_2(\Gamma_1) = 5$.

Система (15) формируется по таблице переходов. Так, из табл. 3 можно построить фрагменты ДНФ:

$$\begin{aligned} D_2 &= \overline{z_1}\,\overline{x_1}; \\ D_3 &= \overline{z_1}\,x_1 \vee z_1\,\overline{x_1}\,x_2. \end{aligned} \qquad (16)$$

Для минимизации числа термов в системе (15) классы $B_i \in \Pi_C$ могут быть закодированы с использованием, например, алгоритма ESPRESSO [2]. Реализация логической схемы КМУУ U_2 сводится к реализации системы (16) на LUT-элементах и к реализации УП на встроенных блоках памяти микросхемы FPGA. Для этой цели могут быть использованы известные методы [1, 2] или стандартные пакеты САПР.

Заключение

Предложенный метод модификации ГСА для интерпретации ее композиционным микропрограммным устройством управления за счет введения дополнительных микрокоманд с кодом класса ПОЛЦ ориентирован на уменьшение числа LUT-элементов в схеме адресации микрокоманд. При этом число блоков встроенной памяти в управляющей памяти устройства управления совпадает с соответствующим числом для базовой структуры U_1 КМУУ с разделением кодов. Для анализа дополнительных вершин алгоритма затрачивается время, что увеличивает количество тактов автомата для прохода одного цикла алгоритма. В то же время снижается сложность схемы адресации микрокоманд (количество уровней схемы), что позволяет уменьшить период синхросигнала. Вывод об изменении быстродействия автомата необходимо делать для каждого конкретного случая, поскольку временные характеристики косвенно зависят от реализуемой ГСА.

Недостатком предлагаемого метода является ограничение (13) на возможность его применения.

Научная новизна предложенного метода состоит в использовании классов ПОЛЦ и свободных ресурсов блоков встроенной памяти для уменьшения числа LUT-элементов в схеме адресации микрокоманд устройства управления для ГСА с элементарными ОЛЦ. Практическая значимость метода заключается в уменьшении перематров необходимых для реализации устройства микросхем, что позволит реализовать схемы, обладающие меньшей стоимостью, чем известные аналоги.

Дальнейшие направления наших исследований связаны с разработкой САПР для синтеза схем КМУУ [5].

Список литературы

1. Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. – М.: Горячая линия-ТЕЛЕКОМ, 2001. – 636 с.
2. Баркалов А.А. Синтез устройств управления на программируемых логических устройствах. – Донецк: ДНТУ, 2002. – 262 с.
3. Virtex-6 FPGAs. Lowest Power High-Performance FPGAs // Електронний ресурс. http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf
4. Stratix III FPGA: Lowest Power, Highest Performance 65-nm FPGA // http://www.altera.com/products/devices/stratix-fpgas/stratix-iii/st3-index.jsp
5. Баркалов А.А., Титаренко Л.А. Синтез микропрограммных автоматов на заказных и программируемых СБИС. – Донецк: УНИТЕХ, 2009. – 336 с.
6. Hardware Reduction for FPGA-based compositional microprogram control units / A.A. Barkalov, L.Titarenko, A.Miroshkin // Proceedings of IEEE East-West Design & Test Symposium - EWDTS '08. Moscow, Russia, 18 -21.09.2009 .- Kharkov, 2009, pp. 21-26.

Надійшла до редколегії 03.10.2009 р Рецензент: д.т.н., проф. Скобцов Ю.О.

О.О. Баркалов, О.О. Красічков, О.М. Мірошкін
Донецький національний технічний університет

Модифікація метода синтезу композиційного пристрою керування для реалізації в базисі FPGA. У статті запропонований метод синтезу для композиційних мікропрограмних пристроїв керування з розділенням кодів та модифікацією операторних лінійних ланцюгів граф-схеми алгоритму керування. Метод спрямований на зменшення апаратних витрат (LUT-елементів базису FPGA) при реалізації пристрою. Зниження складності блоку адресації мікрокоманд забезпечується шляхом включення додаткових операторних вершин, що містять код класу поточного псевдоеквівалентного операторного ланцюгу. Приведені умови доцільності використання методу та приклад його застосування.
Пристрій керування, розділення кодів, алгоритм, FPGA

A.A. Barkalov, A.A. Krasichkov, A.N. Miroshkin
Donetsk National Technical University

Modification of compositional control unit synthesis method for realization if FPGA basis. Synthesis method for compositional microprogram control unit with code sharing and modification of operational linear chains is proposed in the article. Method is directed at hardware amount (LUT-elements of FPGA basis) decrease when realizing control unit. Complexity decrease of block of microinstruction addressing is reached by additional vertices with pseudoequivalent operational linear chain class code implementation. Conditions of usage appropriateness of proposed method are given. Method usage description is conducted by the example.
Control unit, code sharing, algorithm, FPGA