УДК 681.3

А.А. БАРКАЛОВ, Л.А. ТИТАРЕНКО, А.Н. МИРОШКИН

ЭЛЕМЕНТАРИЗАЦИЯ ОПЕРАТОРНЫХ ЛИНЕЙНЫХ ЦЕПЕЙ УПРАВЛЯЮЩЕГО АЛГОРИТМА ПРИ РЕАЛИЗАЦИИ КОМПОЗИЦИОННЫХ УСТРОЙСТВ УПРАВЛЕНИЯ НА FPGA-МИКРОСХЕМАХ

Рассматривается влияние элементаризации операторных линейных цепей алгоритма управления на количество аппаратурных затрат при реализации композиционного устройства управления на современных микросхемах программируемой логики (Spartan3). Определяются области параметров исходных ГСА, при которых целесообразно применение методики элементаризации операторных линейных цепей алгоритма. Приводятся результаты исследований.

1. Введение

Цифровая система, реализующая принцип микропрограммного управления [1], состоит из операционного автомата (ОА) и устройства управления (УУ). Операционный автомат содержит аппаратные средства, способные выполнять конечное множество операций над данными, а устройство управления интерпретирует алгоритм обработки данных, вырабатывая последовательность управляющих сигналов, под воздействием которых операционный автомат выполняет необходимые микрооперации [2]. Для задания последовательности необходимых микроопераций в данной работе используется язык граф-схем алгоритма (ГСА). Функционирование УУ, реализующего ГСА, может быть описано моделью цифрового автомата [3]. Реализация устройства управления в базисе современных программируемых логических интегральных схем (ПЛИС) характеризуется такими показателями как количество аппаратурных затрат, тактовая частота микросхемы, потребляемая и рассеиваемая мощности. Современные ПЛИС позволяют реализовывать все более сложные схемы, однако уменьшение аппаратурных затрат остается актуальной задачей, поскольку позволяет увеличить скорость работы, а также уменьшить потребляемую и рассеиваемую мощности устройства.

Данная статья посвящена исследованию одного из подходов, направленных непосредственно на уменьшение аппаратурных затрат в схеме композиционного микропрограммного устройства управления (КМУУ) с разделением кодов в базисе ПЛИС типа программируемых вентильных матриц (Field Programmable Gate Array, FPGA).

Множество подходов к уменьшению аппаратурных затрат в устройствах управления уже известны [4-8]. Однако использование новых элементных базисов и модификация известных подходов в некоторых случаях могут давать лучшие результаты.

Базис FPGA содержит все необходимые средства для реализации сложных цифровых систем. Наличие специализированных элементов, таких как синхронные блоки встроенной памяти (Embedded Memory Block, EMB), делают базис FPGA очень удобным для реализации схем с памятью, таких как композиционные управляющие автоматы. Целый ряд работ посвящен реализации конечных автоматов с памятью в базисе FPGA [5–9].

Целью данной работы является уменьшение комбинационной части КМУУ благодаря применению методики элементаризации операторных линейных цепей алгоритма управления.

Задачей исследования является разработка методики синтеза КМУУ для ГСА с элементарными операторными линейными цепями. Вторая часть статьи содержит некоторые сведения о синтезе исследуемых структур КМУУ. Результаты исследований и некоторые графики представлены в третьей части. Выводы и направления дальнейших исследований завершают данную статью.

2. Исследуемые структуры устройств управления

Абстрактный автомат задается пятью компонентами: (X, Y, S, f, g), где $X = \{x_1, ..., x_L\}$ – входной алфавит, $Y = \{y_1, ..., y_N\}$ – выходной алфавит, S – множество внутренних состояний автомата, $f : X \times S \to S$ – функции перехода автомата, $g : X \times S \to Y$ – функции выхода. Автомат, имеющий шестую компоненту, s_0 – начальное состояние, называется детерминированным. Конечным называется автомат, имеющий конечное множество внутренних состояний. Цифровое устройство управления является детерминированным конечным цифровым автоматом.

Одним из способов реализации управляющего алгоритма в виде конечного автомата является композиционное микропрограммное устройство управления [10, 11]. Принцип организации КМУУ основан на выделении в ГСА операторных линейных цепей (ОЛЦ).

Операторной линейной цепью α_g ГСА Γ называется конечная упорядоченная последовательность $< b_{g_1}, ..., b_{g_{F_g}} >$ из F_g операторных вершин ГСА Γ, для любой пары соседних компонент которой существует дуга, соединяющая их.

КМУУ является композицией автомата с жесткой и программируемой логикой. Автомат с жесткой логикой выполняет переходы между различными ОЛЦ, а автомат с программируемой логикой адресует состояния в пределах каждой ОЛЦ. Прямая структурная таблица (ПСТ) КМУУ описывает все возможные переходы между различными ОЛЦ. Строка ПСТ включает следующие элементы: текущее состояние a_m, его двоичный код $k(a_m)$, следующее состояние a_s и его код $k(a_s)$, множество логических сигналов X, обеспечивающих переход из состояния a_m в состояние a_s, функции D возбуждения памяти регистра состояния автомата и номер h строки.

Функции перехода задаются формулой:

$$D_i = \bigvee_{h=1}^{H} (P_i^h \wedge (\bigwedge_{r=1}^{R} Q_r^h) \wedge (\bigwedge_{l=1}^{L} C_l^h x_l^h)), \; i = \overline{1, R} \qquad (1)$$

и реализуются автоматом с жесткой логикой, где D_i – функция возбуждения i-го разряда регистра состояния автомата; P_i^h – переменная, определяющая наличие функции D_i в строке h ПСТ ($P_i^h = 1$, если функция D_i присутствует в строке h, и $P_i^h = 0$ в обратном случае); Q_r^h – выход r-го разряда регистра состояния автомата в строке h ПСТ автомата ($Q_r^h = Q_r$, если на выходе r-го разряда регистра находится значение логической «1», и $Q_r^h = \overline{Q_r}$ в обратном случае); x_l^h – переменная, определяющая значение логического условия x_l в строке h ПСТ автомата ($x_l^h = x_l$, если переход в строке h выполняется при выполнении условия, и $x_l^h = \overline{x_l}$ – при невыполнении); C_l^h – переменная, определяющая наличие логического условия x_l в строке h ПСТ автомата ($C_l^h = 1$ при наличии сигнала x_l и $C_l^h = 0$ при его отсутствии); H – количество строк ПСТ автомата; R – разрядность регистра состояния автомата; L – количество сигналов во входном алфавите автомата.

Функции выхода КМУУ определяются только для каждого состояния автомата с программируемой логикой, причем выходные функции не зависят от логических условий и формируются при помощи ПЗУ, следовательно, они формально описываются как $g : S \to Y$.

Операторная вершина b_m называется *входом* ОЛЦ α_g, если существует дуга $\langle b_t, b_m \rangle$, или *выходом*, если существует дуга $\langle b_m, b_t \rangle$, где $b_t \notin \alpha_g$.

Функционирование КМУУ подразумевает естественную адресацию вершин в пределах каждой ОЛЦ α_g, т.е. для перехода $\langle b_i \to b_{i+1} \rangle$ справедливо равенство

$$A(b_{i+1}) = A(b_i) + 1, \qquad (2)$$

где $A(b_i), A(b_{i+1})$ – адреса вершин b_i и b_{i+1} соответственно, $b_i, b_{i+1} \in \alpha_g$.

Для программной реализации более удобным является алгоритм формирования ОЛЦ, основанный на математической модели произвольной операторной вершины b_m ГСА. Для описания алгоритма введем следующие обозначения:

O_m^i – количество переходов $< b_g \to b_m >$, где $b_g \in B_1$, B_1 – множество операторных вершин граф-схемы алгоритма управления;

O_m^o – количество переходов $< b_m \to b_g >$, где $b_g \in B_1$.

Алгоритм формирования ОЛЦ заключается в следующем:

1. Поиск главного входа очередной ОЛЦ по признаку $O_m^i = 0$. Отметка найденной вершины как входа ОЛЦ. Назначение ей очередного кода ОЛЦ и нулевого кода компоненты. При отсутствии вершины с $O_m^i = 0$ – переход на п. 4.

2. Если следующая вершина является условной или конечной, то текущая помечается как вершина-выход текущей ОЛЦ и выполняется переход на п. 1. Иначе переход на следующую вершину.

3. Если для текущей операторной вершины выполняется неравенство $O_m^i \geq 1$, то она может войти в одну из нескольких ОЛЦ. Такая вершина помечается для дальнейшего анализа, затем выполняется переход на п. 1. Иначе назначение текущей вершине очередного кода компоненты. Переход на п. 2.

4. При наличии вершины, которая может войти в одну из нескольких ОЛЦ, определить ее в ОЛЦ-претендент с меньшей длиной. Переход на п. 2. При отсутствии такой вершины выполнить переход на п. 5.

5. Конец.

Описанный алгоритм позволяет получить множество $C = \{\alpha_1, ..., \alpha_G\}$ минимальной мощности.

Принцип разделения кодов (Code Sharing, CS-структура КМУУ) [11, 12] реализуется следующим образом: адрес любой операторной вершины определяется как конкатенация кода ОЛЦ, которой она принадлежит, и кода данной вершины в пределах ОЛЦ (кода компоненты, порядкового номера вершины).

Такая адресация позволяет ввести в схему регистр для хранения кода текущей ОЛЦ и счетчик для хранения кода текущей компоненты ОЛЦ.

Под длиной ОЛЦ будем понимать количество входящих в нее операторных вершин: $F_g = |\alpha_g|$. Пусть $F_{max} = \max(F_1, ..., F_G)$, где G – количество ОЛЦ в ГСА, тогда разрядность регистра равна

$$R_1 = \lceil \log_2 G \rceil, \tag{3}$$

а счетчик имеет

$$R_2 = \lceil \log_2 (F_{max}) \rceil \tag{4}$$

разрядов.

Система функций выходов КМУУ с разделением кодов реализуется на ПЗУ тривиально: в памяти расположена микропрограмма из

$$M = (G-1) \cdot 2^{R_2} + F_{min} \tag{5}$$

слов, где $F_{min} = \min(F_1, ..., F_G)$. Для обеспечения равенства (5) необходимо ОЛЦ минимальной длины закодировать максимальным кодом ОЛЦ, что отнесет соответствующее содержимое управляющей памяти (УП) в область с максимальными адресами.

При унитарном кодировании микрокоманд [11] разрядность слова УП равна

$$N = |Y| + 2, \tag{6}$$

где константа «2» определяет разряды для хранения внутренних переменных y_0 и y_E.

На рис. 1, а представлена структурная схема КМУУ CS. Схема формирования адреса (СФА) реализует систему функций переходов (1), а управляющая память содержит микро-

программу, в которой находятся закодированные одним из известных способов [13] выходные функции.

По сигналу Start в регистр Рг и счетчик СТ заносится начальный адрес микропрограммы, а триггер выборки ТВ разрешает чтение из управляющей памяти. Если считанная из УП микрокоманда не соответствует выходу ОЛЦ, то одновременно с микрооперациями Y формируется сигнал y_0. Если $y_0 = 1$, то к содержимому СТ прибавляется единица и адресуется следующая компонента текущей ОЛЦ, содержимое Рг при этом остается неизменным. Если выход ОЛЦ достигнут, то $y_0 = 0$. При этом адрес входа следующей ОЛЦ вырабатывается схемой формирования адреса, после чего новый код ОЛЦ записывается в Рг, а номер компоненты – в СТ. При достижении окончания микропрограммы формируется сигнал y_E, триггер ТВ обнуляется и выборка микрокоманд прекращается.

Рассмотрим принцип элементаризации ОЛЦ. Схема формирования адреса вырабатывает код следующей ОЛЦ и номер ее компоненты. В том случае, если адресацию операторных вершин выполнить таким образом, что каждая ОЛЦ имеет ровно один вход и один выход, то переход всегда будет осуществляться на компоненту с нулевым кодом, что избавит от необходимости его формирования. При этом количество ОЛЦ может увеличиться, и СФА будет формировать адрес, разрядность которого вместо $R_1 + R_2$ (3), (4) будет составлять

$$R_1' = \lceil \log_2(G') \rceil, \qquad (7)$$

где G' – количество ОЛЦ после применения методики элементаризации. Структурная схема КМУУ с разделением кодов и элементаризацией ОЛЦ приведена на рис. 1, б.

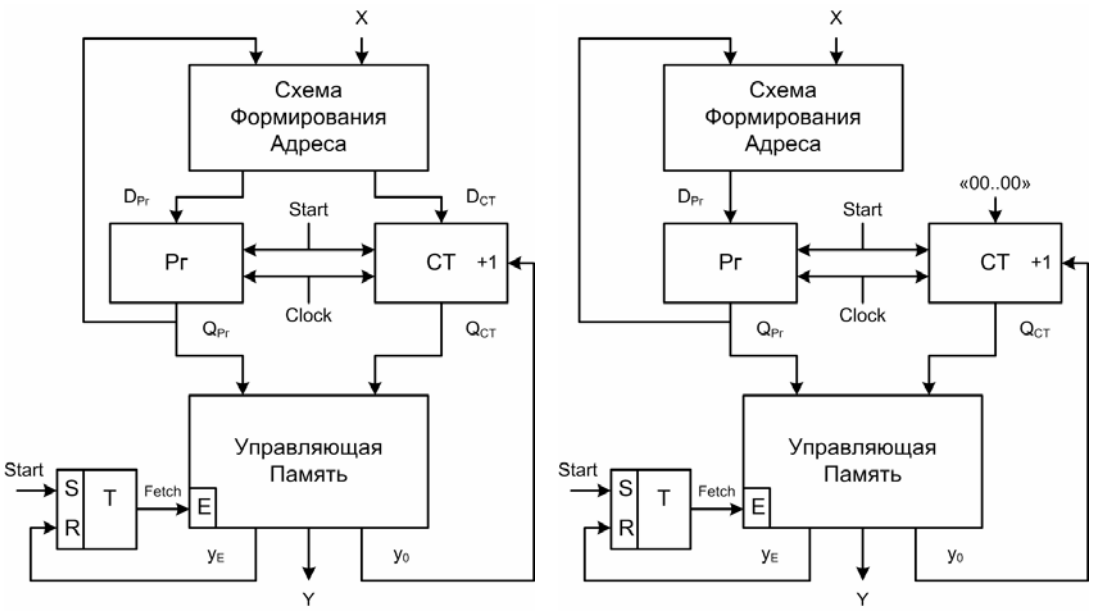


Рис. 1. Структурная схема КМУУ: а – с разделением кодов (CS);
б – с разделением кодов и элементаризацией ОЛЦ (ECS)

3. Результаты экспериментов

Для эксперимента были выбраны ГСА со следующими параметрами:
– количество вершин от 10 до 500 с шагом 10;
– доля операторных вершин от 50 до 90% с шагом 10%;
– количество микроопераций = 15;
– количество логических условий = 5.

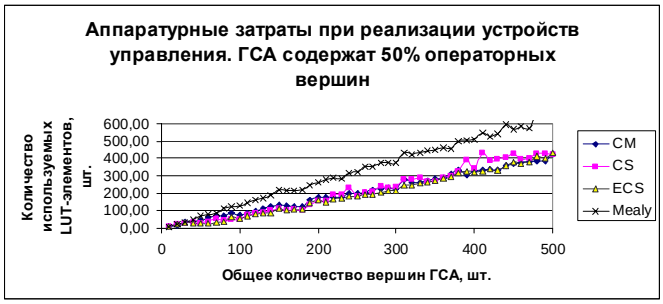
Для указанных ГСА были получены результаты имплементации КМУУ с разделением кодов без применения элементаризации ОЛЦ и с применением таковой (рис. 2-4). Также

приводится сравнительная характеристика результатов реализации тех же ГСА в виде автоматов Мили и в виде КМУУ с общей памятью [11] (СМ-структура). В качестве элементного базиса выбраны FPGA-микросхемы Spartan-6 фирмы Xilinx.

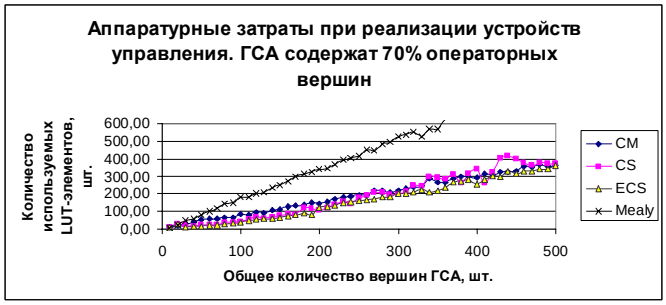
Для автоматизации процесса синтеза управляющих устройств была разработана система Генератор Автоматов (ГенА), которая по описанию алгоритма управления в формате XML генерирует VHDL-модель управляющего устройства заданной структуры. Под моделью понимается описание структурных элементов устройства на языке описания аппаратуры VHDL и при необходимости файлы прошивки ПЗУ. Далее модель автомата поступает в САПР Xilinx ISE, где проходит процесс имплементации в одну из доступных микросхем по выбору пользователя. Отчет об имплементации содержит подробную количественную информацию об использовании ресурсов микросхемы, которая и подлежит дальнейшему анализу.

При получении набора экспериментальных данных был выполнен синтез и имплементация пяти ГСА для каждой измеряемой точки.

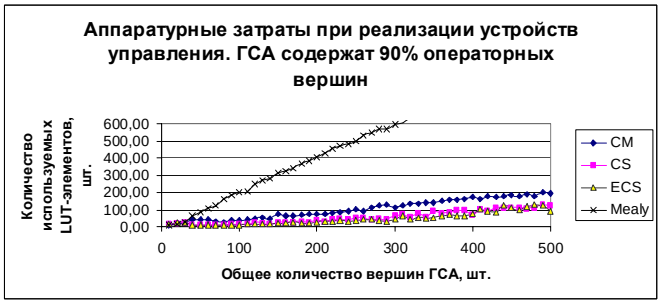
Анализ графиков на рис. 2 позволяет сделать вывод о том, что для реализации любой ГСА, параметры которой находятся в исследуемых пределах, в виде КМУУ необходимо количество аппаратуры одного порядка. Рост аппаратурных затрат для реализации алгоритма в виде автомата Мили имеет линейный характер, причем угол наклона прямой, аппроксимирующей график затрат, с ростом процента операторных вершин увеличивается, а у композиционных устройств уменьшается.



а



б



в

Рис. 2. Аппаратурные затраты при реализации автомата Мили, КМУУ с общей памятью, с разделением кодов, с элементаризацией ОЛЦ. Процент операторных вершин в ГСА: а – 50%; б – 70%; в – 90%

Для сравнения структур КМУУ между собой рассмотрим график на рис. 3, который был нормирован относительно аппаратурных затрат для КМУУ с общей памятью.

Как видно из рис. 3, структура КМУУ с разделением кодов требует меньшие аппаратурные затраты для ГСА с количеством вершин от 10 до 380, после чего определить, даст ли данная структура более выгодную в сравнении с КМУУ с общей памятью реализацию, можно лишь выполнив синтез обеих структур для каждой конкренной ГСА.

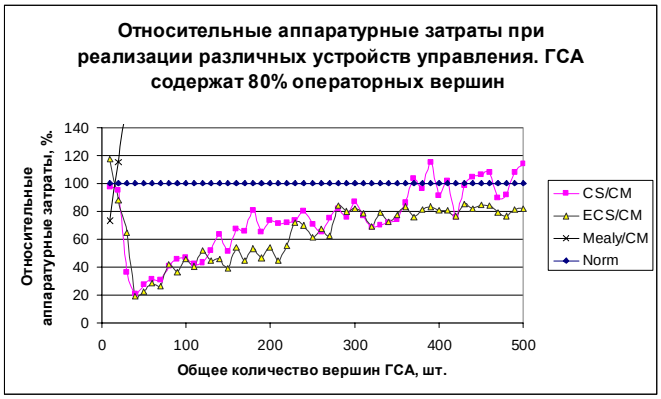


Рис. 3. Относительные аппаратурные затраты при реализации автомата Мили, КМУУ с разделением кодов, с элементаризацией ОЛЦ по отношению к значениям для КМУУ с общей памятью. ГСА содержат 80% операторных вершин

Применение элементаризации ОЛЦ управляющего алгоритма делает ECS-структуру КМУУ более экономной в плане необходимых ресурсов в сравнении с КМУУ с общей памятью на всем исследуемом отрезке параметров ГСА

Для сравнения CS- и ECS-структур КМУУ рассмотрим рис. 4, где выполнено нормирование аппаратурных затрат по отношению к КМУУ без элементаризации ОЛЦ управляющего алгоритма.

График на рис. 4 показывает, что в 26% из общего числа ГСА с 80% операторных вершин структура КМУУ без элементаризации ОЛЦ алгоритма управления оказывается эффективнее. Общее количество вершин этих ГСА сосредоточены на отрезках [10; 120] и [270; 360]. На указанных отрезках необходимо выполнять синтез обеих структур для выяснения той, которая даст меньшие аппаратурные затраты в каждом конкретном случае. Для ГСА, параметры которых не попадают в указанные отрезки, достаточно синтезировать лишь КМУУ с элементаризацей ОЛЦ.

Рис. 4. Относительные аппаратурные затраты при реализации КМУУ с использованием методики элементаризации ОЛЦ алгоритма управления и без таковой

4. Выводы

Предлагаемая методика элементаризации ОЛЦ алгоритма при реализации КМУУ с разделением кодов в базисе ПЛИС типа FPGA направлена на уменьшение числа LUT-

элементов, используемых в схеме формирования адреса микрокоманд. Методика заключается в формировании ОЛЦ с единственным входом, что позволяет уменьшить количество формируемых разрядов адреса при переходе между различными ОЛЦ с величины $R_1 + R_2$ (3), (4) до R_1' (7).

Проведенные исследования показали, что применение методики элементаризации ОЛЦ исходных ГСА может снизить количество аппаратурных затрат в среднем на 10-15% в сравнении с базовой структурой КМУУ при реализации схем устройств управления в базисе современных FPGA-микросхем. Если количество вершин ГСА принадлежит отрезкам [10; 120] или [270; 360], применение данной методики может оказаться неэффективным, однако проверить это можно лишь после синтеза всех сравниваемых структур КМУУ для конкретной ГСА.

Научная новизна предлагаемой методики заключается в установлении аналитической зависимости между параметрами входного алгоритма и количеством необходимых для реализации схемы ресурсов микросхемы FPGA. Для микросхемы Spartan-3 фирмы Xilinx аналитическая зависимость имеет вид $Q = (-0{,}043p^2 + 4{,}554p - 40{,}8) \cdot N$, где Q – количество LUT-элементов, необходимых для реализации схемы, N – общее количество вершин ГСА, p – доля операторных вершин в ней ($p \in [0.5;0.9]$). Данная зависимость может быть использована для других FPGA-микросхем фирмы Xilinx с четырехвходовыми LUT-элементами. Погрешность формулы для ГСА с количеством вершин более 40 не превышает 30%, в то время как для малых ГСА статистическая погрешность может достигать величины аппаратурных затрат.

Период синхросигнала схемы КМУУ составляет 70-110% соответствующего значения для автомата Мили. Схема автомата Мили может функционировать быстрее для интерпретации более разветвленных ГСА, а при увеличении доли операторных вершин она уступает в скорости схеме КМУУ.

Длительность формирования выходных сигналов для рассматриваемых структур КМУУ одинакова и составляет 3,6-5 нс против 7-11 нс для автомата Мили, причем для КМУУ эти значения постоянны, а в автомате Мили задержки растут при увеличении сложности схемы.

Практическая значимость методики состоит в уменьшении количества необходимых ресурсов микросхемы, а также в возможности определения этого количества до начала процесса синтеза схемы, что позволит выбрать подходящую микросхему.

Дальнейшие направления исследований связаны с анализом эффективности применения различных подходов к кодированию вершин ГСА, использования незанятых участков управляющей памяти для уменьшения аппаратурных затрат при реализации устройства управления.

Список литературы: 1. *Wilkes M.V.* The Best Way to Design an Automatic Calculating Machine (1951, July), Rept. Manchester Univ. Computer lnaug. Conf., Manchester, U.K. **2.** *Baranov S.* Logic and System Design of Digital Systems. Tallinn: TUT Press, 2008. 266 p. **3.** *Глушков В.М.* Синтез цифровых автоматов. М.: Физматгиз, 1962. 476 с. **4.** *ROM*-based finite state machine implementation in low cost FPGAs / I. Garcна-Vargas, R. Senhadji-Navarro, G. Jimйnez-Moreno, A. Civit-Balcells, P. Guerra-Gutiйrrez // (2007) IEEE International Symposium on Industrial Electronics, art. no. 4374972. P. 2342-2347. **5.** *ROM*-based FSM implementation using input multiplexing in FPGA devices / R. Senhadji-Navarro, I. Garcна-Vargas, G. Jimйnez-Moreno , A. Civit-Ballcels // (2004) Electronics Letters, 40 (20). P. 1249-1251. **6.** *An application* of functional decomposition in ROM-based FSM implementation in FPGA devices / M. Rawski, H. Selvaraj, T. Juba // (2005) Journal of Systems Architecture, 51 (6-7). P. 424-434. **7.** *Synthesis* and Implementation of RAM-Based Finite State Machines in FPGAs. / V. Sklyarov // 10th International Conference «Field-Programmable Logic and Applications: The Roadmap to Reconfigurable Computing» Proceedings, FPL 2000 Villach, Austria, August 27–30, 2000. P. 718-727. **8.** *Saving* power by mapping finite-state machines into embedded memory blocks in FPGAs. / A. Tiwari, K.A. Tomko // (2004) Proceedings - Design, Automation and Test in Europe Conference and Exhibition, 2. P. 916-921. **9.** *Garcia E.* Creating Finite State Machines Using True Dual-Port Fully Synchronous SelectRAM Blocks // Xcell Journal, 2000, Issue 38. P. 36-38. **10.** *Баркалов А.А.* Микропрограммное устройство управления как композиция автоматов с программируемой и жесткой логикой // Автоматика и вычислительная техника. 1983. №4. С.36-41. **11.** *Баркалов А.А.* Синтез устройств управления на программируемых логических устройствах. Донецк: ДонНТУ, 2002. 262 с. **12.** *Оптими-*

зация устройства управления с разделением кодов / А.А. Баркалов, Л.А. Титаренко, А.Н. Мирошкин / / Управляющие системы и машины. 2010. N 1. С. 45-50. **13.** *Синтез* микропрограммных устройств управления. Баркалов А.А., Палагин А.В. К.: ИК НАН Украины, 1997. 135 с.

Поступила в редколлегию 21.03.2011

Баркалов Александр Александрович, д-р техн. наук, проф. кафедры компьютерной инженерии ДонНТУ, проф. университета Зеленогурского (Польша). Научные интересы: цифровые устройства управления. Хобби: научная работа, спорт. Адрес: Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Gora. E-mail: A.Barkalov@iie.uz.zgora.pl.

Титаренко Лариса Александровна, д-р техн. наук, проф. кафедры ТКС ХНУРЭ, проф. университета Зеленогурского (Польша). Научные интересы: системы телекоммуникаций, цифровые устройства управления. Хобби: научная работа, спорт. Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Gora E-mail: L.Titarenko@iie.uz.zgora.pl.

Мирошкин Александр Николаевич, аспирант кафедры компьютерной инженерии ДонНТУ. Научные интересы: цифровая схемотехника. Хобби: научная работа, спорт. Адрес: Украина, Донецкая обл., Макеевка, ул. Курская, д. 15, кв. 45.