УДК 519.713:681.3

Л.В. ДЕРБУНОВИЧ, док. техн. наук, проф., ***И.Г. ЛИБЕРГ***,к.т.н., проф., ***Д.В. ЯКУБОВСКИЙ***, аспирант НТУ «ХПИ» (г. Харьков)

МЕТОД СИНТЕЗА ДЕТЕРМИНИРОВАННЫХ ТЕСТОВЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ НА ОДНОМЕРНЫХ СЕТЯХ КЛЕТОЧНЫХ АВТОМАТОВ

Пропонується метод синтезу тестових послідовностей, які є еволюцією. мережі клітинного автомату та можуть бути згенеровані на одномірних мережах клітинних автоматів. Це дозволяє досягти високої швидкості роботи генератора таких послыдовностей, а також низьких апаратних витрат на його реалізацію.

An approach for synthesis of test patterns which are evolution of cellular automata and can be generated by linear cellular automata is described. High speed of operating and low costs of hardware implementation can be achieved by using this scheme.

Широкое использование в современных вычислительных системах (ВС) микропроцессоров, ПЛИС и систем на одном кристалле (СОК) с числом выводов, достигающим 1000 на одну микросхему и функционирующих на тактовой частоте 5-10 ГГц, приводит к значительному возрастанию стоимости диагностического обеспечения (ДО) на всех этапах жизни ВС.

Известно, что затраты на генерацию тестов и моделирование неисправностей растут с увеличением размерности объекта диагностирования (ОД), рабочей частоты, числа выводов СБИС и печатных плат. Прогноз ряда ведущих фирм, занимающихся разработкой и производством современных СБИС и СОК, показывает, что к 2012 г. затраты на их проектирование и производство и стоимость средств их ДО будут практически равны. Для СБИС с 1000 выводами стоимость системы ДО составит ~$5·106, что эквивалентно стоимости разработки кристалла [1, 2].

Существующие системы ДО ориентированы на обнаружение класса устойчивых неисправностей константного типа, что неадекватно отражает множество возможных дефектов, возникающих в современных ВС. Повышение плотности интеграции электронных элементов привело к возрастанию числа дефектов типа "замыкание соседних линий", а с увеличением тактовой частоты становятся соизмеримыми задержки сигналов в линиях связи и элементах, что приводит к появлению неисправностей типа "задержка фронта и среза импульса", искажению функциональных характеристик устройств и системы в целом. На этапе эксплуатации ВС в условиях промышленного производства в большинстве случаев отказы в них обусловлены неустойчивыми неисправностями перемежающегося типа и сбоями, возникающими в процессе выполнения управляющих программ.

С целью снижения стоимости систем ДО и повышения эффективности современных систем управления и вычислительной техники ведущими зару-

бежными фирмами - производителями СОК, компьютеров и телекоммуникационных систем и сетей были предложены рекомендации, которые в настоящее время представлены международными стандартами проектирования сложных систем: *IEEE* 1149.1-4 "*Standard Test Access Port and Boundary-Scan Architecture*" (стандартный тест-порт и архитектура граничного сканирования) и IEEE P1500 "*Standard for Embedded Core Test*" - стандарт встроенных средств ТД [3-5]. Введение этих стандартов проектирования сложных аналого-цифровых устройств и систем определяет новые подходы к построению и организации функциональных и тестовых средств диагностирования этих систем.

В соответствии с ГОСТ 20911-75 "Техническая диагностика", под функциональным диагностированием (ФД) будем понимать диагностирование, осуществляемое во время функционирования ОД, на который поступают только рабочие воздействия, в отличие от тестового диагностирования (ТД), когда на ОД подаются тестовые воздействия, формируемые в процессе его проектирования с помощью специальных систем генерации тестов [6].

Полнота обнаружения неисправностей, время тестирования и аппаратные затраты являются основными критериями эффективности систем ДО[7].

Издание журнала *IEEE Design Test of Computers, December*, 2004 посвящено проблеме верификации проектов сложных цифровых систем, создаваемых на основе современных субмикронных технологий. В публикациях ведущих специалистов крупнейших фирм производителей *Synopsis, Intel, IBM Inc, Mentor Graphics* и др., а также известных ученых в области технической диагностики *R. Gupta, S. Malik, M. Hsiao, G. Micheli, I. Koren* и др. обсуждается эффект синергизма при решении проблемы верификации цифровых систем. Эффект синергизма на этапе производства и эксплуатации достигается при создании ДО систем путем строгого соблюдения стандартов тестопригодного проектирования, сочетания функциональных и тестовых методов диагностирования неисправностей, использования статических и динамических методов верификации, встроенных средств самотестирования (*Built-in Self-Test, BIST*).

BIST позволяет проводить диагностирование на рабочих частотах дискретных устройств (ДУ), что не всегда позволяют сделать внешние средства диагностирования. Так, несмотря на возможности, которые предоставляет для внешнего диагностирования стандарт *JTAG*, для загрузки тестовых последовательностей требуется слишком большое время [8].

Развитие *BIST* также связано с развитием СОК, где средство диагностирования должно быть встроенным по определению.

Для проверки исправности ДУ используются различные методы компактного тестирования: исчерпывающее, псевдоисчерпывающее, псевдослучайное тестирование, тестовое диагностирование детерминированным множеством тестов, которые синтезируются на этапе проектирования ДУ программными системами генерации тестов и моделирования неисправностей.

Задача генератора тестовых последовательностей - подача тестовых воздействий на входы диагностируемой схемы. Выходные значения тестируемой схемы могут сжиматься в сигнатуру. В конце теста сигнатура сравнивается с эталонной – если они совпадают, схема считается исправной.

В работах [9-18] представлены различные методы синтеза генераторов псевдослучайных и псевдоисчерпывающих последовательностей на основе использования сдвиговых регистров с линейными и нелинейными обратными связями (СРЛОС И СРНОС соответственно).

Генерация псевдослучайных тестов имеет преимущество в виде небольшой площадии, требуемой для сдвигового регистра с линейной ОС. Однако может потребоваться значительное время для проведения теста, а также некоторая проблема заключается в подсчёте полноты обнаружения неисправностей.

Хранение тестовых наборов (полученных при помощи специальных методик заранее) в памяти приводит к резкому уменьшению времени тестирования и предсказуемому покрытию ошибок, однако это решение наиболее дорогостоящее из трёх в плане аппаратных затрат [8].

Большое число работ посвящено методам синтеза генераторных модулей *BIST* на одно- и двумерных сетях клеточных автоматов (СКА) [8, 19-22]. Показано преимущество схемной реализации генераторов на СКА благодаря отсутствию глобальных обратных связей в этих схемах и идентичностью структур конфигурируемых логических блоков ПЛИС типа *FPGA* и *CPLD*. За счет того, что клетки СКА работают параллельно, СКА обладают очень высоким быстродействием.

Поэтому разработка методов синтеза и проектирования эффективных технических средств *BIST* с использованием СКА, соответствующая перечисленным выше требованиям, является актуальной проблемой.

Цель статьи – описание метода синтеза детерминированных тестовых последовательностей **для СКА**, который позволяет достичь высоких показателей быстродействия при низких аппаратных затратах на реализацию генератора.

Структуры клеточных автоматов.

СКА характеризуется четырьмя параметрами: геометрией сети, окрестностью, которая влияет на каждую клетку, количеством состояний клетки и функцией, с помощью которой клеточный автомат вычисляет своё последующее состояние (правилом настройки клетки). Варьируя эти свойства, можно получить самые разнообразные СКА. На рис. 1 показана одномерная линейная СКА длины n с нулевыми граничными условиями:

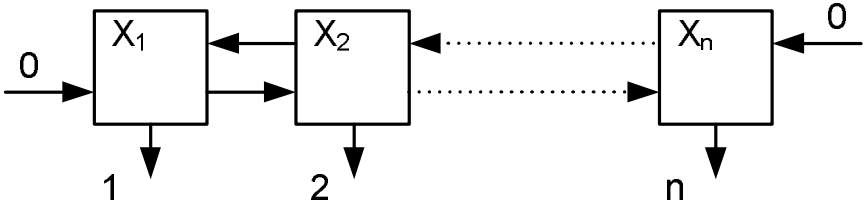

Рис. 1. Одномерная n-клеточная СКА с нулевыми граничными условиями

В данной статье рассматривается линейная СКА, каждая клетка которой в любой момент времени находится в одном из двух состояний – «0» или «1». Каждая клетка СКА содержит своё правило настройки. Такая СКА, где правила настройки клеток не одинаковы, называется гибридной. В СКА используется неймановская окрестность, то есть на клетку влияют только её ближайшие соседи и она сама.

Пусть СКА состоит из n клеток (n – размерность СКА), которые расположены так, как показано на рис. 1. Пусть СКА имеет нулевые граничные условия, то есть на соответствующие входы первой и последней клеток всегда подаётся нулевое значение.

Если определить текущее состояние i-й клетки на шаге t как x_i^t, то последующее состояние x_i^{t+1} определяется следующей функцией:

$$x_i^{t+1} = f(x_{i-1}^t, x_i^t, x_{i+1}^t), i = 1,...,n. \quad (1)$$

Пример типовой структуры клетки показан на рис. 2.

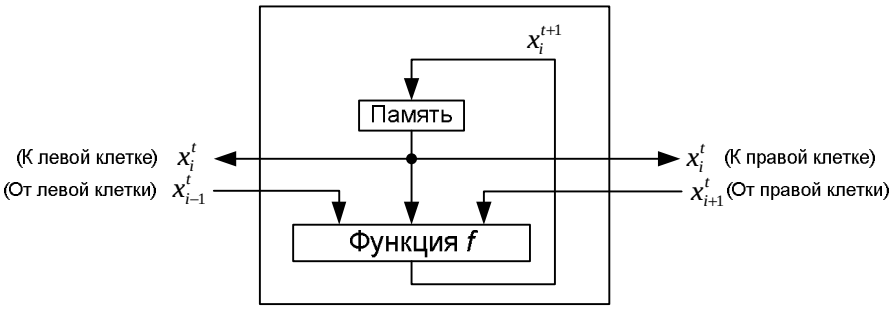

Рис. 2. Типовая структура клетки

Клетка состоит из двух основных блоков. Элементом памяти задаётся начальное состояние клетки и сохраняется ткущее состояние x_i^t. Функция f (правило настройки клетки) определяет следующее состояние клетки x_i^{t+1} в зависимости от окрестности клетки x_i^t, то есть в зависимости от x_{i-1}^t, x_i^t и x_{i+1}^t.

На табл. 1 приведён пример получения численного значения правила настройки для клетки СКА с окрестностью 1 и числом состояний 2.

Таблица 1 – Пример получения численного значения правила настройки

Состояние окрестности -> Правило	111	110	101	100	011	010	001	000
72	0	1	0	0	1	0	0	0
215	1	1	0	1	0	1	1	1

Верхняя строка представляет все возможные состояния окрестности клетки как восемь трехзначных двоичных чисел. Каждое такое число определяет состояние данной клетки и клеток слева и справа от неё. Под каждым числом записано состояние, в которое перейдёт данная клетка. Этим определяется правило настройки клетки. Полученные таким образом восемь бит образуют двоичное число от 0 до 255. Это число, записанное в десятичной системе, используется, как название правила настройки. В табл. 1 приведен пример для правил настройки 72 и 215.

Правило настройки клетки может быть также представлено в виде булевой функции. Каждая функция является суммой минтермов трёх переменных. Для правила настройки 72 функция будет иметь следующий вид:

Правило настройки 72: $x_{i+1}^t = x_{i-1}^t * x_i^t + x_i^t * x_{i+1}^t$,

где «*» соответствует логическому «И», а «+» – логическому «ИЛИ».

Эволюции. Эволюция данной СКА, состоящей из n клеток, как показано на рис. 1, с известным начальным состоянием, определяется отображением $F : \{0,1\}^n \to \{0,1\}^n$, которое задаёт её поведение на каждом такте функционирования СКА.

Эволюция обычно представляется в виде автоматной диаграммы состояний. Пусть

$$X_i = (x_1^{t_1},...,x_i^{t_j},...,x_n^{t_m})^T,$$

где T обозначает транспонирование матрицы. Выражение (2) представляет эволюцию n-размерной СКА для m шагов (итераций):

$$\text{эволюция СКА} = (X_1,...,X_i,...,X_n). \quad (2)$$

Эволюция каждой клетки СКА также может быть представлена в виде диаграммы состояний. Эволюция i-й клетки состоит из последовательности трёх двоичных столбцов, (X_{i-1}, X_i, X_{i+1}), где $X_i = (x_i^{t_1},...,x_i^{t_j},...,x_i^{t_m})^T$. Состояния i-той клетки соответствуют центральному столбцу, X_i.

Определение 1. Эволюция для данной структуры КА разрядностью n описывается набором из n троек столбцов, каждая из которых представляет эволюцию одной клетки со связями, соответствующими неймановской окрестности:

$$\text{эволюция КА} = \{(X_{i-1}, X_i, X_{i+1}) / 1 \le i \le n\}. \quad (3)$$

Каждая последовательность столбцов (X_{i-1}, X_i, X_{i+1}) представляет эволюцию i-й клетки. Следует заметить, что столбцы X_0 и X_{n+1} должны быть нулевыми, что соответствует нулевым граничным условиям.

Детерминированность переходов состояний клетки. В данной статье рассматриваются только детерминированные правила настройки клеток, которые соответствуют детерминированным моделям конечных автоматов. Это означает, что каждая пара идентичных состояний клетки переведёт клетку в одно и то же состояние.

Пусть состояние окрестности i-й клетки на шаге t будет определено как $N(x_i^t) = (x_{i-1}^t, x_i^t, x_{i+1}^t)$. Тогда свойство детерминированности переходов состояний клетки можно записать следующим образом:

Для произвольных двух шагов t_i и t_k, где $t_i \neq t_k$,

$$\text{Если } N(x_i^{t_j}) = N(x_i^{t_k}) \text{ Ю } x_i^{t_{k+1}} = x_i^{t_{j+1}}. \quad (4)$$

В предлагаемом алгоритме (4) используется для определения, является ли эволюцией последовательность состояний клетки, определённая тремя столбцами заданной тестовой последовательности.

Синтез тестовых последовательностей. Детерминированная тестовая последовательность может содержать тройки столбцов, которые не являются эволюцией клетки в соответствии с (4), то есть не являться эволюцией СКА в соответствии с (3). Такую последовательность невозможно сгенерировать на линейной СКА с неймановской окрестностью. Для генерации такой последовательности тестов структурой КА его следует преобразовать.

Детерминированная тестовая последовательность размерностью n и длиной m представляется как набор столбцов $C = \{c_1, c_2, ..., c_n\}$ длиной m бит каждый. Алгоритм позволяет получить такую последовательность C', для которой выполняются следующие условия:

1) $C' = \{(c'_{i-1}, c'_i, c'_{i+1}) / 1 \le i \le n'\}, n \le n'$. Каждый столбец C' может быть либо столбцом из C, либо связующим столбцом. Столбцы c'_0 и $c'_{n'+1}$ будут нулевыми, они соответствуют нулевым граничным условиям.

2) Любая последовательность трёх столбцов $(c'_{i-1}, c'_i, c'_{i+1})$ из C' удовлетворяет свойству, описанному в (4), т.е. является эволюцией клетки.

Алгоритм синтеза C' состоит в итеративном нахождении таких последовательностей из трёх столбцов в C, которые удовлетворяют свойству (4). В случае, если невозможно найти такие столбцы в C, в C' добавляется связующий столбец. Далее алгоритм рассмотрен более подробно.

Алгоритм синтеза. Пусть c_0 будет крайним левым столбцом в C', соответствующим нулевым граничным условиям СКА, X^0 – текущим столбцом, и X^{-1} – столбцом, находящимся слева от текущего.

Шаг 1. Добавить в C' нулевой столбец c_0. Выбрать из C произвольный столбец c_i, пометить его в C как обработанный, и добавить его в C'. Принять этот столбец за текущий столбец X^0.

Шаг 2. Найти последовательным перебором всех необработанных столбцов в C ($c_i / 1 \le i \le n$) такой столбец c_i, что три столбца (X^{-1}, X^0, c_i) является эволюцией клетки в соответствии с (4). Если такой столбец существует, добавить его в C' и пометить в C как обработанный. Принять этот столбец за текущий столбец X^0. Если такой столбец не существует, перейти к шагу 4.

Шаг 3. Если в C существуют необработанные столбцы, перейти к шагу 2, иначе перейти к шагу 5.

Шаг 4. Добавить к C' справа связующий столбец c_l, подобранный таким образом, чтобы три столбца (X^{-1}, X^0, c_l) являлись эволюцией клетки в соответствии с (4). Принять этот столбец c_l за текущий столбец X^0. Перейти к шагу 3.

Шаг 5. Добавить в C' нулевой столбец c_0. Если столбцы (X^{-1}, X^0, c_0) не являются эволюцией клетки в соответствии с (4), удалить c_0 и перейти к шагу 4, иначе перейти к шагу 6.

Шаг 6. Конец.

В результате работы алгоритма на основе заданной тестовой последовательности C синтезируется новая тестовая последовательность C', которая представляет собой эволюцию СКА. Каждой эволюции клетки соответствует определённое правило настройки клетки.

Таким образом, линейная СКА, собранная из клеток, генерирующих найденные эволюции, генерирует последовательность C', которая содержит все наборы исходной тестовой последовательности C. На входы тестируемой схемы подаются значения на выходах клеток, соответствующих столбцам исходной последовательности C.

На рис. 3 приведена блок-схема алгоритма.

Пример работы алгоритма. Пусть имеется заданная последовательность C (рис. 4). Разрядность последовательности $n = 8$, длина последовательности $m = 5$.

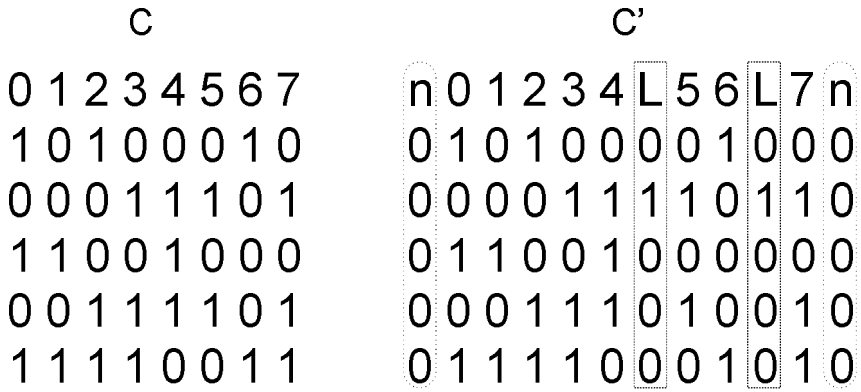

Рис. 4. Заданная последовательность

После выполнения алгоритма будет синтезирована последовательность C' (рис. 4).

Буквами «n» обозначены нулевые столбцы, обеспечивающие нулевые граничные условия, буквами «L» обозначены связующие столбцы.

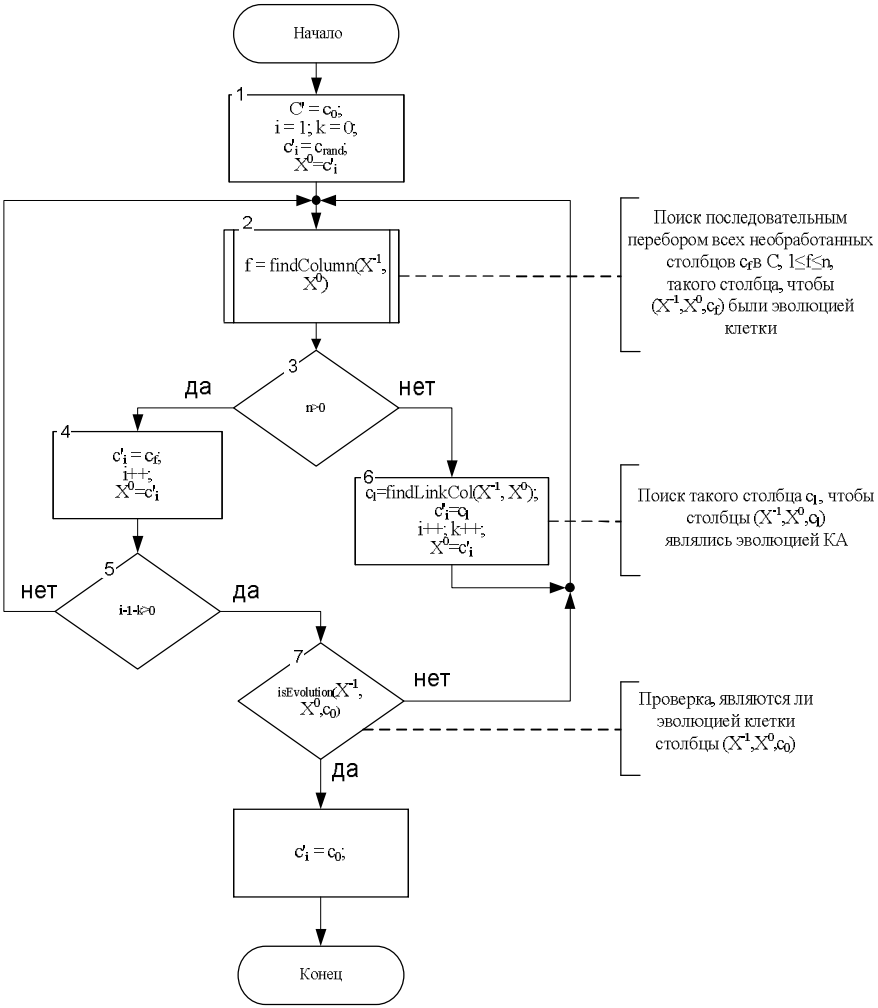

Рис. 3. Блок-схема алгоритма синтеза

Как видно, в данном примере для получения заданной последовательности размерностью 8 потребуется СКА размерностью 10, так как использовано 2 соединяющих столбца.

Вывод. Предложен метод синтеза детерминированных тестовых последовательностей, которые генерируются линейной СКА с неймановской окрестностью при определённых правилах настройки её клеток. В соответствии с

данным методом и алгоритмом на основании произвольной детерминированной тестовой последовательности синтезируется детерминированная тестовая последовательность для СКА. Решение задачи нахождения правил настройки для каждой клетки позволит синтезировать генератор на СКА с высоким быстродействием, низкими аппаратными затратами и отсутствием глобальных обратных связей в схеме генератора.

Список литературы: **1.** *Aitken R.C.* Nanometer technology effects on fault models for IC testing // Computer. – 1999. – N 11. – P. 46-51. **2.** *Cheng K.T., Kristic A.* Current Directories in Automatic Test-Pattern Generation // Computer. – 1999. – N 11. – P. 58-64. **3.** JTAG Boundary Scan Architecture Standard Proposal, Version 2.0. Technical Subcommittee of the Joint Test Action Group. – March 30, 1988. **4.** *Maunder C, Beenker F., Vivier C. A* Standard Boundary Scan Architecture. – June 1987. **5.** *Zorian Y., Marinissen E.J.* Testing Embedded-Core based System Chips // Computer. – 1999. – N6. – P. 52-60. **6.** ГОСТ 20417-75. Техническая диагностика. Общие положения о принципах разработки систем диагностирования // М.: Изд-во стандартов, 1979. **7.** *W. Daehn, J. Mucha* A hardware approach to self-testing of large programmable logic arrays // IEEE Trans. Computers. – 1981. – Vol. 30, N 11. – P. 829-833. **8.** *Neebe lD.J., Kime* C.*R.* Cellular Automata for Weighted RandomPattern Generation // IEEE Trans. Computers. – 1997. – Vol. 46, N 11. – P. 1219-1229. **9.** *Bardell P.H., McAnney W.H., Savir J.* Built-in Test for VLSI: Pseudorandom techniques. – New York: John Wiley & Sons, 1987. – 274 p. **11.** Генератор псевдослучайных чисел: А.с. 1347167 СССР, МКИ НОЗ К 3/84 / *Л.В. Дербунович, В.Ф. Бохан, И.Г. Либерг* (СССР). – N4022981/21; Заявлено 07.02.86; Опубл. 23.10.87, Бюл. N 39. – 1с. **12.** *Boubezari S., Kaminska B.*A Deterministic Built-In Self-Test Generator Based on Cellular Automata Structures // IEEE Trans. Computers. – 1995. – Vol.44, N 6. – P. 805-816. **13.** *J. van Sas, Cattoor F., H. De Man* Optimized BIST strategies for programmable data paths based on cellular automata // IEEE Int'l Test Conf. – 1992. – P. 119. **14.** *Derbunovich L., Berezna M., Ryzhykova M., Tatarenko D.* Pseudoexhaustive tpg based on nonlinear feedback shift registers // Информационно-управляющие системы на железнодорожном транспорте – 2005. – N5 – С. 54-59. **15.** *Fredricksen H.* A Survey of full length nonlinear shift register cycle algorithms // SIAM Review. – 1982. – Vol 24, N 2. – P. 195-221. **16.** *Дербунович Л.В., Темников И.Н., Татаренко Д.А.* Генераторы тестов для дискретных устройств с самотестированием // Информационно-управляющие системы на железнодорожном транспорте. – 2004. – N1. – С. 42-46. **17.** *Tang D.T., Chen C.L.* Logic test pattern generation using linear codes // IEEE Trans. Comput. – 1984. – Vol. C-33, N 9. – P. 845-850. **18.** *Chatterjee M., Pradhan D.K.* A BIST pattern generator design for near-prefect fault coverage // IEEE Trans. Computer-Aided Design – 1999 – Vol 18 – N2 – P. 238-247. **19.** *Hortensius P.D., McLeod R.D., Pries W., Miller D.M., Card H.C.* Cellular automata-based pseudorandom number generators for built-in self-test // IEEE Trans. CAD. – 1989. – N 8 – P. 842-859. **20.** *Gloster C.R., Borglez F.* Boundary scan with cellular-based built-in self-test // Proc. Intern. Test Conference. – 1988. – P. 138-145. **21.** *Gortensius P.D., Card H.C, McLeod R.D.* Parallel random-number generation for VLSI systems using cellular automata. // IEEE Trans. Comput. – 1989. – Vol. C-38, N 10. – P. 1466-1473. **22.** *Boubezari S., Kaminska B.* Cellular automata synthesis based on precomputed test vectors for built-in self-test // IEEE Int'l Conf. on Computer-Aided Design in Santa Clara. – 1993. – P.578-583

Поступила в редколлегию 04.11.08