МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ
УНИВЕРСИТЕТ РАДИОЭЛЕКТРОНИКИ

ISSN 0135-1710

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ УПРАВЛЕНИЯ И ПРИБОРЫ АВТОМАТИКИ

Всеукраинский межведомственный научно-технический сборник

Основан в 1965 г.

Выпуск 148

Харьков
2009

УДК 62-50

В сборнике представлены результаты исследований, касающихся компьютерной инженерии, управления, технической диагностики, автоматизации проектирования, оптимизированного использования компьютерных сетей и создания интеллектуальных экспертных систем. Предложены новые подходы, алгоритмы и их программная реализация в области автоматического управления сложными системами, оригинальные информационные технологии в науке, образовании, медицине.

Для преподавателей университетов, научных работников, специалистов, аспирантов.

У збірнику наведено результати досліджень, що стосуються комп'ютерної інженерії, управління, технічної діагностики, автоматизації проектування, оптимізованого використання комп'ютерних мереж і створення інтелектуальних експертних систем. Запропоновано нові підходи, алгоритми та їх програмна реалізація в області автоматичного управління складними системами, оригінальні інформаційні технології в науці, освіті, медицині.

Для викладачів університетів, науковців, фахівців, аспірантів.

Свидетельство о государственной регистрации
печатного средства массовой информации КВ № 12073-944ПР от 07.12.2006 г.

Адрес редакционной коллегии: Украина, 61166, Харьков, просп. Ленина, 14, Харьковский национальный университет радиоэлектроники, комн. 321, тел. 70-21-326

СОДЕРЖАНИЕ

УДК. 621.396.2: 625.316.2

В.А. ТИХОНОВ, Н.В. КУДРЯВЦЕВА

СПЕКТРАЛЬНЫЙ АНАЛИЗ КОМБИНИРОВАННЫХ МОДЕЛЕЙ ЛИНЕЙНОГО ПРЕДСКАЗАНИЯ НЕГАУССОВЫХ ПРОЦЕССОВ

Рассматривается синтез комбинированной модели линейного предсказания на примере обобщенной модели авторегрессии-скользящего среднего. Предлагаются выражения для параметрических спектральных оценок комбинированной модели негауссовых случайных процессов.

1. Введение

При решении ряда прикладных задач статистической радиотехники необходимо анализировать сложные случайные процессы с многомодовым спектром, состоящие из нескольких зависимых или независимых процессов. Хотя некоторые процессы с многомодовым спектром достаточно точно описываются моделями линейного предсказания, для анализа сложных составных процессов необходимы новые модели линейного предсказания. В статье рассмотрены комбинированные модели линейного предсказания, которые описывают составной процесс как целое или как смесь составляющих процессов. Они моделируют, в частности, процессы, которые состоят из различных комбинаций моделей гауссовых или негауссовых процессов [1]. Предложенные комбинированные модели описывают составляющие процессов классическими или обобщенными моделями линейного предсказания r-го ранга [2].

Целью исследования является разработка нового метода спектрального анализа комбинированных моделей линейного предсказания гауссовых и негауссовых процессов.

Задачи: вывод формул для параметрических оценок спектров комбинированных моделей; нахождение параметрических оценок спектров негауссовых процессов.

2. Комбинированные модели линейного предсказания

Возможны различные варианты применения комбинированных моделей для описания негауссовых процессов, у которых спектры второго порядка отличаются от спектров высших порядков. В качестве таких моделей используются модели авторегрессии-скользящего среднего (АРСС), авторегрессии (АР) или скользящего среднего (СС). Ошибка предсказания моделей АРСС, АР или СС может содержать информацию о негауссовых составляющих процесса, не учтенных этими моделями. Для описания ошибок предсказания можно присоединить к моделям АРСС, АР и СС обобщенные модели АРСС (ОАРСС), АР (ОАР) или СС (ОСС) [1]. Если построение комбинированных моделей начать с обобщенных моделей, т.е. моделей ОСС и ОАР, то можно выделить негауссовы и гауссовы составляющие процесса.

Ниже рассмотрен пример синтеза комбинированной модели обобщенной авторегрессии и скользящего среднего ($\text{ОАР}\times\text{СС}$), которая представляется комбинацией обобщенной модели ОАР третьего ранга и классической модели СС второго ранга. Комбинированная присоединенная модель $\text{ОАР}\times\text{СС}$ описывается разностным уравнением [1]:

$$x[t]=\sum_{i=1}^{p_3}\Phi_3^1[i]x[t-i]-\sum_{n=1}^{q_2}Q_2[n]a_2[t-n]+a_2[t], \qquad (1)$$

где $\Phi_3^1[i]$ – коэффициенты модели ОАР третьего ранга; p_3 – порядок модели ОАР; 1 – сдвиг моментной функции, по которой рассчитывались $\Phi_3^1[i]$; $Q_2[n]$ – коэффициенты модели СС второго ранга; q_2 – порядок модели СС; $a_2[t]$ – ошибки предсказания модели $\text{ОАР}\times\text{СС}$.

Уравнения для расчета параметров модели получают из условия оптимальности моделей. Оно заключается в статистической независимости ошибок предсказания для разных сдвигов времени:

$$E\{a_2[t]x[t-j]\} = E\{a_2[t]a_2[t-j]\} = 0,$$

$$E\{a_3^l[t]x[t-j]x[t-l]\} = E\{a_3^l[t]a_3^l[t-j]a_3^l[t-l]\} = 0 \ , \ 0 < j \ , \ l \geq 0 \ , \qquad (2)$$

где $a_3^l[t]$ – ошибка предсказания модели ОАР.

Параметры модели ОАР 3-го ранга вычисляются по системе уравнений, следующей из рекуррентного выражения [2]

$$m_3[j, j-l] = \sum_{i=1}^{p_3} \Phi_3^l[i] m_3[j-i, j-l], \ 0 < j \leq p_l, \qquad (3)$$

здесь $m_3[j, j-l]$ – моментные функции 3-го порядка при фиксированном сдвиге l.

После синтеза модели ОАР можно найти ошибку предсказания из выражения, следующего из (1):

$$x[t] - \sum_{i=1}^{p_3} \Phi_3^l[i] x[t-i] = a_3^l[t]. \qquad (4)$$

Широкополосная ошибка предсказания на выходе обеляющего ОАР фильтра будет коррелированной. Поэтому для расчета коэффициентов СС используется система уравнений [3]

$$r[j] = \left(\sum_{n=1}^{q_2} Q_2[n-j]Q_2[n] - Q_2[j] \right) \Bigg/ \left(\sum_{n=1}^{q_2} (Q_2[n])^2 + 1 \right), j = 1,2,...,q_2. \qquad (5)$$

Таким образом, уравнения (3) и (5) используются для расчета параметров комбинированной модели, а порядок моделей ОАР и СС определяется из условия их оптимальности (2).

3. Параметрическая оценка спектров комбинированной модели ОАР$\times$СС

Найдем выражения для параметрических спектров комбинированной модели ОАР$\times$СС. Уравнение (1) описывает формирующий фильтр с рациональной частотной характеристикой

$$H(\omega) = \frac{\sum_{n=0}^{q_2} Q_2[n] e^{-j\omega Tn}}{\sum_{i=0}^{p_3} \Phi_3^l[i] e^{-j\omega Ti}}, \qquad (6)$$

на вход которого подается негауссов белый шум. Формирующий фильтр комбинированной модели состоит из соединенных каскадно различных комбинаций фильтров. Амплитудно-частотная характеристика линейной системы, состоящей из n каскадно соединенных фильтров, описана в [4]. Спектр r-го порядка процесса на выходе такого формирующего фильтра равен произведению r-го центрального момента m_{ra} негауссова порождающего процесса на частотные характеристики r-го порядка $A_{ri}(\omega)$:

$$P_r(\omega) = A_r(\omega) m_{r,a} = A_{r1}(\omega) \cdot ... \cdot A_{rn}(\omega) m_{ra}, \qquad (7)$$

где

$$A_{ri}(\omega) = H_i(\omega_l) \cdot ... \cdot H_i(\omega_{r-1}) H_i(-\omega_l - ... - \omega_{r-1}). \qquad (8)$$

Нулевое сечение спектральной плотности r-го порядка, определяемое условием $\omega_2, ..., \omega_{r-1} = 0$, описывается выражением

$$P_r(\omega) = |H_r(\omega)|^2 m_{r,a} = |H_{r1}(\omega)|^2 \cdot ... \cdot |H_{rn}(\omega)|^2 m_{ra}. \qquad (9)$$

Из (6)-(9), используя ОАР параметрическую оценку спектральной плотности [5] и параметрическую оценку спектра третьего порядка модели СС [3], запишем выражение для оценки спектра второго порядка комбинированной модели ОАР$\times$СС в виде

$$P_2(\omega) = D_{a_2} \mid \sum_{n=0}^{q_2} Q_2[n] e^{-j\omega Tn} \mid^2 / \mid \sum_{i=0}^{p_3^l} \Phi_3^l[i] e^{-j\omega Ti} \mid^2. \qquad (10)$$

Спектральная плотность третьего порядка представляется параметрическим выражением

$$P_3(\omega_1, \omega_2) = m_{3a_2} \frac{\sum_{n=0}^{q_2} Q_2[n] e^{-j\omega_1 Tn} \sum_{n=0}^{q_2} Q_2[n] e^{-j\omega_2 Tn} \sum_{i=0}^{q_2} Q_2[n] e^{j(\omega_1+\omega_2)Tn}}{\sum_{i=0}^{p_3^l} \Phi_3^l[i] e^{-j\omega_1 Tk} \sum_{i=0}^{p_3^l} \Phi_3^l[i] e^{-j\omega_2 Ti} \sum_{i=0}^{p_3^l} \Phi_3^l[i] e^{j(\omega_1+\omega_2)Ti}},$$

а ее нулевое сечение имеет вид

$$P_3(\omega) = m_{3a_2} K \mid \sum_{n=0}^{q_2} Q_2[n] e^{-j\omega Tn} \mid^2 / \mid \sum_{i=0}^{p_3^l} \Phi_3^l[i] e^{-j\omega Ti} \mid^2, \; K = \sum_{n=0}^{q_2} Q_2[n] / \sum_{i=0}^{p_3^l} \Phi_3^l[i].$$

Спектральная плотность присоединенных комбинированных моделей линейного предсказания представляется произведением спектральных плотностей обобщенных или классических моделей линейного предсказания.

4. Синтез комбинированной модели ОАР$\times$СС

Методом статистического моделирования исследовалась комбинированная модель ОАР$\times$СС негауссова процесса, полученного в виде аддитивной смеси негауссова узкополосного ОАР и гауссова широкополосного СС процессов в отношении 5/1 по мощности. Центральная частота СС составляющей процесса выбиралась равной $f_1 = 50$, а для ОАР составляющей центральная частота равнялась $f_2 = 10$ при интервале дискретизации $T = 0{,}01$. Коэффициент асимметрии негауссовой составляющей был равен $\gamma_1 = 1{,}066$, а для смеси он составлял $\gamma_1 = 0{,}8173$. На рис. 1 показан график теоретической спектральной плотности, рассчитанный по формуле (10), в предположении, что смесь является комбинированным процессом линейного предсказания с указанными выше частотами пиков составляющих.

Покажем, как комбинированная модель линейного предсказания применяется для моделирования смеси, состоящей из гауссовой и негауссовой составляющих. При построении комбинированной модели полученной негауссовой смеси вначале была построена модель ОАР(2) третьего ранга, параметры которой рассчитывались из системы уравнений (3). Так как у гауссова процесса моментная функция третьего порядка равна нулю, то коэффициенты ОАР(2) третьего ранга не зависят от гауссовой составляющей, являющейся процессом СС.

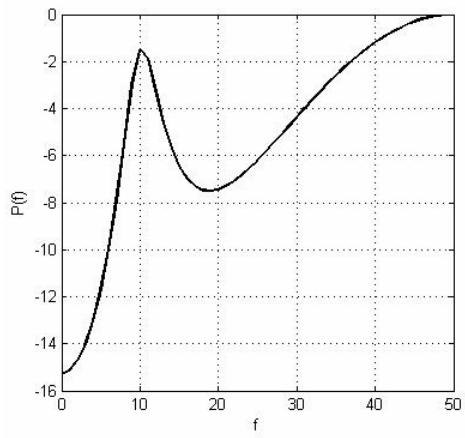


Рис. 1. Теоретический спектр комбинированной модели смеси негауссова процесса ОАР (2) и гауссова процесса СС (8)

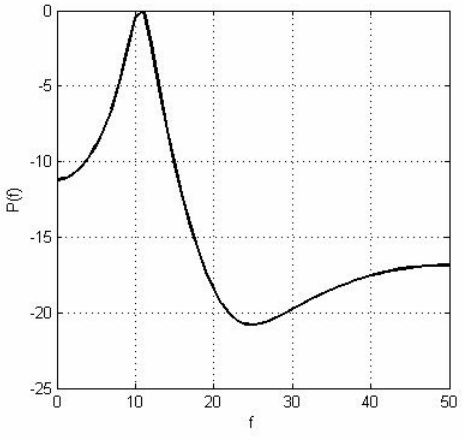


Рис. 2. Выборочный спектр смеси негауссова процесса комбинированной модели ОАРСС

Модель ОАР(2) третьего ранга использовалась для выделения ошибки предсказания $a_3^1[t]$. Для этого применялось соотношение (4). Затем для ошибки предсказания строилась модель СС(3) второго ранга, параметры которой рассчитывались с использованием (5). На рис. 2 показан график выборочной параметрической спектральной плотности, рассчитанной по комбинированной модели ОАР $\times$ СС смеси при сдвиге $1=1$. Сравнение графиков, представленных на рис. 1 и 2, показывает, что они соответствуют заданным спектрам негауссовой и гауссовой составляющих смеси. Отметим, что график на рис. 2 показывает не только спектральные характеристики смеси, но и указывает на гауссову и негауссову составляющие спектра смеси. Отличие уровней спектральных пиков от теоретических вызвано тем, что аддитивная смесь не является истинным комбинированным процессом ОАР $\times$ СС. Наличие аддитивного гауссова процесса несколько смещает выборочные оценки коэффициентов ОАР негауссова процесса, найденные по смеси. Кроме этого, необходимо учитывать, что негауссова составляющая была мощнее в 5 раз гауссовой составляющей, что также повлияло на форму выборочного спектра.

5. Заключение

Практическая значимость: предложенные модели могут быть полезны для моделирования негауссовых сигналов и помех, для выделения полезных сигналов на фоне помех и шумов, для спектрального анализа случайных процессов в присутствии помех и шумов, при решении задач линейного прогнозирования.

Научная новизна: впервые получены спектральные оценки комбинированных моделей линейного предсказания негауссовых процессов, полностью описывающихся одним разностным уравнением.

Список литературы: 1. *Тихонов В. А., Кудрявцева Н. В.* Присоединенные комбинированные модели линейного предсказания-обобщенного линейного предсказания негауссовых процессов // Радиотехника. 2008. №154. С. 152–155. **2.** *Тихонов В.А.* Обобщенная модель авторегрессии негауссовых процессов // Радиотехника. 2003. №132. С. 78–82. **3.** *Бокс Дж., Дженкинс Г.* Анализ временных рядов: Пер. с англ. М.: Мир, 1974. Вып. 1. 406 с. **4.** *Гольденберг Л.М.* и др. Цифровая обработка сигналов. М.: Радио и связь, 1985. 312 с. **5.** *Тихонов В.А*. Параметрические оценки спектров высших порядков негауссовых статистически связанных процессов // Радиоэлектроника. 2005. № 11. С. 27–39.

Поступила в редколлегию 11.09.2009

Тихонов Вячеслав Анатольевич, д-р физ.-мат.наук, проф. кафедры РЭС ХНУРЭ. Научные интересы: статистическая радиофизика, статистические модели, негауссовы процессы, экономическая статистика. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 702-15-87.

Кудрявцева Наталья Валериевна, стажер-исследователь кафедры РЭС ХНУРЭ. Научные интересы: теория линейного предсказания, статистическая обработка сигналов, негауссовы процессы. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 702-15-87.

УДК 621.38

А.А. БОРИСЕНКО, В.В. ПЕТРОВ

СИНТЕЗ УНИТАРНЫХ БИНОМИАЛЬНЫХ СЧЕТЧИКОВ

На основании исследований, проведенных в данной работе, находятся обобщенные аналитические выражения, необходимые при построении быстродействующих унитарных биномиальных счетчиков любой разрядности. Полученные счетчики имеют регулярную структуру и удобны для реализации на ПЛИС. Также описывается программная модель и производится оценка быстродействия.

1. Постановка задачи

На практике перспективным является применение биномиальных счетных схем, построенных на основе унитарной биномиальной системы счисления [1]. Эти схемы содержат естественную избыточность, которую можно использовать для повышения помехоустойчивости, и обладают повышенным быстродействием. Отличием унитарных биномиальных счетчиков (УБС) от других, работающих в унитарных кодах, является расширенная функциональность – возможность установки коэффициента пересчета [2]. Счетчик содержит k дополнительных входов, на которые подается биномиальное число с параметрами $n' = k' = k$, количество единиц в нем определяет число состояний. Для практической реализации указанных счетных схем необходимо разработать алгоритм их синтеза. Для построения такого алгоритма необходимо получение простых аналитических выражений на основе существующих алгоритмов счета, позволяющих строить биномиальные счетчики любой разрядности.

Текущий коэффициент пересчета определяется числом сочетаний $(k-i)$ единиц из $m = (n+1-i)$ элементов:

$$N_{T.} = C_m^{k-i} = C_{n+1-i}^{k-i} = \frac{(n+1-i)!}{(k-i)!(n+1-i-k+i)!} = k+1-i, \qquad (1)$$

где i – количество единиц биномиального числа на входе УБС.

Работа УБС с параметрами $n = k = 4$ приведена в табл. 1.

Таблица 1. Состояния УБС с параметрами $n = k = 4$

№ п/п	Номер состояния	Бин. число на входе счетчика	Разряды счетчика
		4321	4321
1	0	0000	0000
2	1		1000
3	2		1100
4	3		1110
5	4		1111
6	0	1000	0000
7	1		0100
8	2		0110
9	3		0111
10	0	1100	0000
11	1		0010
12	2		0011
13	0	1110	0000
14	1		0001
15	0	1111	0000

Алгоритм работы УБС имеет вид:

1. Все разряды счетчика установлены в нуль.

2. Заносится 1 в $(k-q-i)$-й разряд, где q – число единиц в счетчике, i - число единиц биномиального числа на входе счетчика.

3. Если младший разряд не заполнен единицей, то происходит переход к пункту 2.

4. Если младший разряд заполнен единицей или количество единиц биномиального числа на входе равно k, то цикл счета окончен.

Как видно из табл. 1, унитарный биномиальный код с параметрами $n = k$ имеет некоторые свойства рефлексных кодов, поскольку при переходе к младшему или старшему биномиальному чис-

лу происходит изменение цифры только в одном разряде числа. Исключение составляет переход от максимального числа к нулю. Это свойство позволяет получить значительно большее быстродействие по сравнению с двоичными счетчиками, без улучшения характеристик элементной базы.

2. Логический синтез

В качестве элемента памяти УБС выбираем D –триггеры из-за простоты их управления и возможности применения регистров на их основе.

Воспользовавшись табл. 1, составим таблицу истинности суммирующего двухразрядного унитарного биномиального счетчика с параметрами $n = k = 2$ (табл. 2).

Таблица 2. Таблица истинности счетчика с параметрами n = k = 2

№	x1	x2	t		t^{+1}		D1	D2
			Q1	Q2	Q1	Q2		
0	0	0	0	0	1	0	1	0
1	0	0	1	0	1	1	1	1
2	0	0	1	1	0	0	0	0
3	1	0	0	0	0	1	0	1
4	1	0	0	1	0	0	0	0
5	1	1	0	0	0	0	0	0

В табл.2 x1 – x2 являются входами для подачи биномиального числа; Q1 – Q2 – разряды счетчика; D1 – D2 являются функциями возбуждения триггеров Q1 – Q2.

Карты Карно функций возбуждения D1, D2 триггеров Q1, Q2 приведены на рис.1.

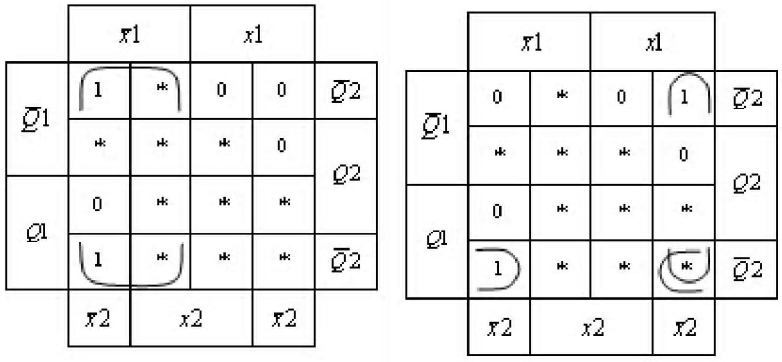


Рис. 1. Карты Карно для функции возбуждения D1, D2 триггеров Q1, Q2

Неиспользуемые наборы в табл. 3 являются факультативными и оптимально доопределяются для получения минимальной дизъюнктивной нормальной формы (МДНФ). МДНФ функций имеют вид

$$D1 = \bar{x}1\bar{Q}2,$$

$$D2 = \bar{x}2Q1\bar{Q}2 + x1\bar{x}2\bar{Q}2 = \bar{x}2\bar{Q}2(x1 + Q1). \qquad (2)$$

Для установления закономерностей формирования функций возбуждения D –триггеров n – разрядного унитарного биномиального счетчика установим эти функции для параметров $n = k = 3$. Граф состояний счетчика показан на рис. 2.

Воспользовавшись табл. 1, 2, составим таблицу истинности работы счетчика (табл. 3).

В табл. 3 x1 – x3 являются входами для подачи биномиального числа, определяющего разрядность счетчика; Q1 – Q3 являются разрядами счетчика; D1 – D3 – функции возбуждения триггеров Q1 – Q3.

Карта Карно функции возбуждения D1 первого триггера Q1 приведена на рис.3.

Таблица 3. Таблица истинности УБС с параметрами n = k= 3

№	x1	x2	x3	t			t^{+1}			D1	D2	D3
				Q1	Q2	Q3	Q1	Q2	Q3			
0	0	0	0	0	0	0	1	0	0	1	0	0
1	0	0	0	1	0	0	1	1	0	1	1	0
2	0	0	0	1	1	0	1	1	1	1	1	1
3	0	0	0	1	1	1	0	0	0	0	0	0
4	1	0	0	0	0	0	0	1	0	0	1	0
5	1	0	0	0	1	0	0	1	1	0	1	1
6	1	0	0	0	1	1	0	0	0	0	0	0
7	1	1	0	0	0	0	0	0	1	0	0	1
8	1	1	0	0	0	1	0	0	0	0	0	0
9	1	1	1	0	0	0	0	0	0	0	0	0

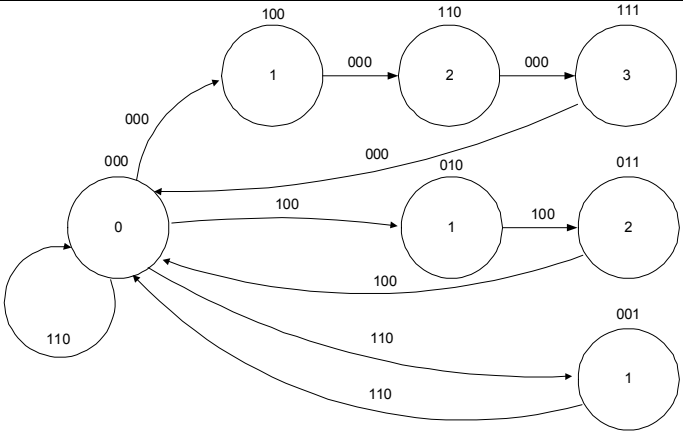


Рис. 2. Граф состояний УБС с параметрами n = k = 3

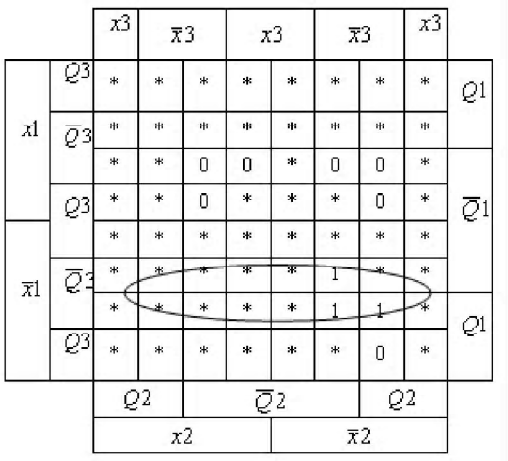


Рис. 3. Карта Карно для функции D1 возбуждения триггера Q1

МДНФ функции D1 имеет вид

$$D1 = \bar{x}1\bar{Q}3\,. \qquad (3)$$

Карты Карно функции возбуждения D2 и D3 триггеров Q2, Q3 приведены на рис.4, 5 соответственно.

МДНФ функций D2, D3 имеет вид

$$D2 = x1\bar{x}2\bar{Q}3 + \bar{x}2Q1\bar{Q}3 = \bar{x}2\bar{Q}3(x1 + Q1)\,, \qquad (4)$$

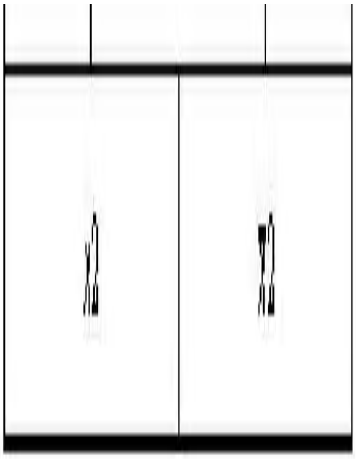


Рис. 4. Карта Карно для функции D2 возбуждения триггера Q2

$$)3 = \overline{x}3\overline{Q}3(x2 + Q2)\,. \qquad (5)$$

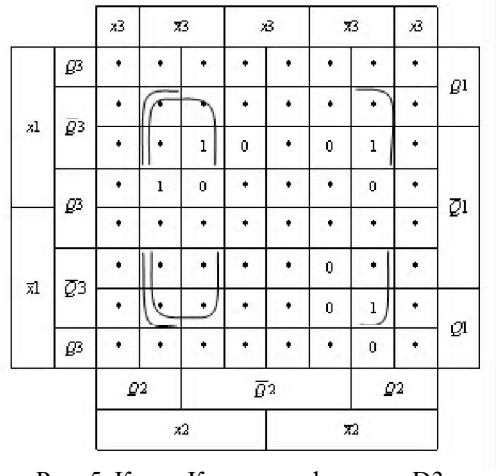


Рис. 5. Карта Карно для функции D3 возбуждения триггера Q3

3. Аналитический синтез функций возбуждения

Установим закономерности в алгоритмах формирования функций возбуждения D – триггеров для n-разрядных УБС. Для этого сведем функции возбуждения триггеров счетчиков с $n = 2$, $n = 3$ в табл. 4

Таблица 4. Функции возбуждения унитарных биномиальных счетчиков с n = k

Разрядность счетчика, n	Функции возбуждения D – триггеров
2	$D1 = \overline{x}1\overline{Q}2$
	$D2 = \overline{x}2\overline{Q}2(x1 + Q1)$
3	$D1 = \overline{x}1\overline{Q}3$
	$D2 = \overline{x}2\overline{Q}3(x1 + Q1)$
	$D3 = \overline{x}3\overline{Q}3(x2 + Q2)$

Закономерность построения функций возбуждения триггеров является очевидной. По аналогии запишем функции для счетчика с $n = 4$. Для получения однородной структуры перепишем функцию D1 в виде

$$D1 = \overline{x}1\overline{Q}4 \cdot 1, \qquad (6)$$

$$D2 = \overline{x}2\overline{Q}4(x1 + Q1), \qquad (7)$$

$$D3 = \overline{x}3\overline{Q}4(x2 + Q2), \qquad (8)$$

$$D4 = \overline{x}4\overline{Q}4(x3 + Q3)\,. \qquad (9)$$

На основании табл. 4 и соотношений (6) – (9) запишем в общем виде функцию возбуждения D-триггера n-разрядного унитарного биномиального счетчика:

$$\begin{cases} D_1 = \overline{x}1\overline{Q}_{n_{max}} \cdot 1, j = 1 \\ D_j = \overline{x}_j\overline{Q}_{n_{max}}(x_{j-1} + Q_{j-1}), 1 < J \le n_{max}\,, \end{cases} \qquad (10)$$

где D_j – функция возбуждения j-го триггера; $Q_{n_{max}}$ – единичный выход самого старшего триггера x_{j-1}, x_j - (j-1) и j-й входы биномиального числа, $j = \overline{1,2...n_{max}}$.

Для доказательства выражения (10) используем метод математической индукции. В соответствии с первым шагом метода $D_2 = \overline{x}_2\overline{Q}_{n_{max}}(x_1 + Q_1)$. На втором шаге допустим правильность выражения (10) при $j = k$, тогда $D_k = \overline{x}_k\overline{Q}_{n_{max}}(x_{k-1} + Q_{k-1})$. На третьем шаге метода докажем, что выражение (10) будет верно при $j = k + 1$, тогда

$D_{k+1} = \overline{x}_{k+1}\overline{Q}_{n_{max}}(x_{k+1-1} + Q_{k+1-1}) = \overline{x}_{k+1}\overline{Q}_{n_{max}}(x_k + Q_k)$, заменим в этом выражении $k+1$ на a, в результате получим – $D_a = \overline{x}_a\overline{Q}_{n_{max}}(x_{a-1} + Q_{a-1})$. Полученное выражение полностью соответствует гипотезе (10), гипотеза подтверждена.

На основе соотношений (6) – (9) получим схему, представленную на рис. 6. Выделим в ней однотипные блоки, названные ячейками памяти (ЯП), которые являются универсальными элементами для построения УБС.

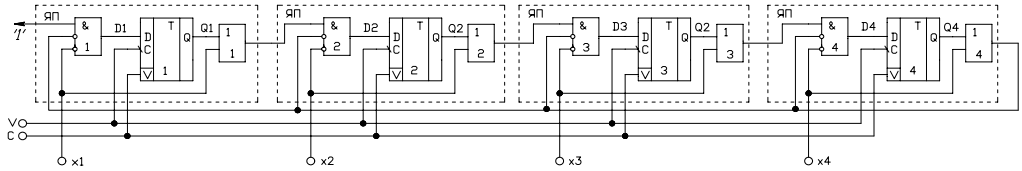


Рис. 6. Биномиальный счетчик с параметрами $n = k = 4$

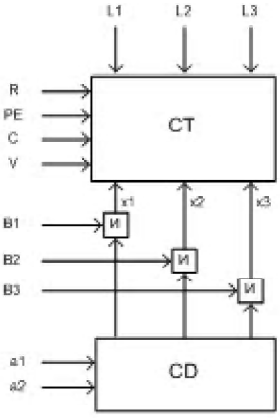


Рис. 7. Функциональная схема УБС с преобразователем кода

Для определения коэффициента пересчета УБС двоичным числом необходим преобразователь кода (рис. 7). Также выбор коэффициента пересчета возможен при помощи биномиального счетчика такого же типа.

В случае использования биномиального счетчика его собственный коэффициент пересчета выбирается равным $N = k+1$, а его информационные выходы заводятся на входы биномиального числа управляемого счетчика. При таком включении коэффициент пересчета управляемого счетчика согласно (1) будет определяться унитарным кодом, поданным на вход управляющего счетчика.

На рис. 8 приведена схема УБС с параметрами $n = k = 3$.

Приведенная схема отличается от рис. 6 тем, что в ней добавлена схема асинхронного сброса и параллельной загрузки, выполненная на элементах 1.1 - 1.3, .2.1 – 2.3, 3.1 – 3.3, преобразователь кодов на элементах 4.1 – 4.2, на группу входов a1 – a2 которого подается двоичное число, определяющее коэффициент пересчета счетчика. Согласно (1)

$$N_{T.} = k + 1 - G_2,$$

где G_2 – число в двоичной форме, поданное на входы a1 – a2.

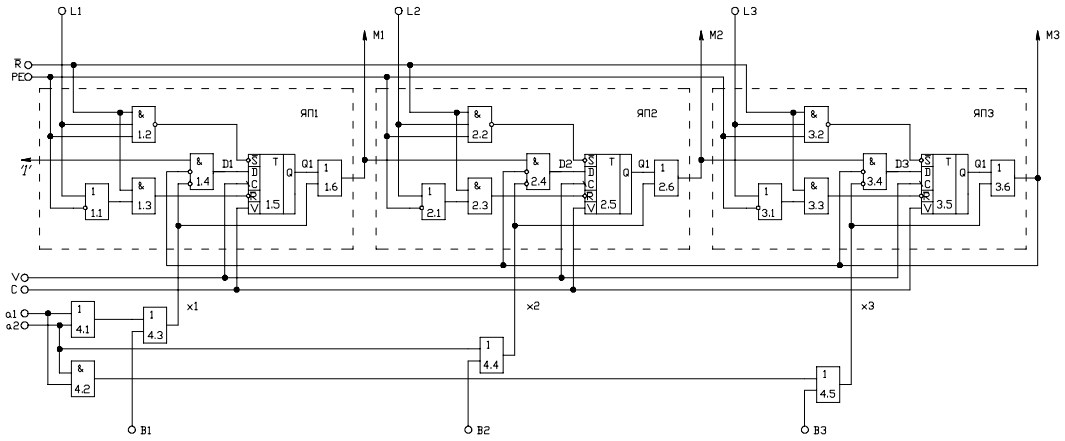


Рис. 8. Унитарный биномиальный счетчик с параметрами $n = k = 3$

Для подачи числа, определяющего коэффициент пересчета в биномиальной форме, предназначены входы $B1 - B3$. Вход разрешения параллельной загрузки и асинхронного сброса соответственно PE и $\overline{R}$, $L1 - L3$ – входы параллельной загрузки. Выходы $M1 - M3$ предназначены для контроля ячеек памяти.

Таким образом, было получено простое аналитическое выражение, позволяющее получать функции возбуждения триггеров n-разрядного унитарного биномиального счетчика с переменным коэффициентом пересчета. Также был получен универсальный элемент ЯП, позволяющий строить быстродействующие унитарные биномиальные счетчики любой разрядности и быстродействующие каскадные биномиальные счетчики на их основе. Такие счетчики имеют регулярную структуру и удобны для реализации на ПБИС.

4. Оценка быстродействия

Для оценки быстродействие схемы рис. 8 найдем аналитические выражения, определяющие максимальную тактовую частоту f_{max}, которую можно подать на вход счетчика.

Как упоминалось выше, биномиальный код с параметрами $n = k$ имеет свойства рефлексных кодов. Это значит, что при переходе к младшему или старшему биномиальному числу происходит изменение только в одном разряде числа, исключение составляет переход от максимальной кодовой комбинации к нулевой. Как следствие, быстродействие всего счетчика равно быстродействию одного разряда. Счетчик, приведенный на рис. 8, разбит на однотипные элементы – ячейки памяти. Для оценки быстродействия счетчика необходимо найти максимальное время задержки распространения сигнала в одной ЯП. Из рис. 8 очевидным является то, что максимальное время задержки ЯП1 равно

$$t_{зд.р.}^{ЯП1} = t_{з.р.ср.}^{1.4} + t_{з.р.ср.}^{1.5} + t_{з.р.ср.}^{1.6},$$

где $t_{зд.р.}^{ЯП1}$ – время задержки ЯП1; $t_{з.р.ср.}^{1.4}, t_{з.р.ср.}^{1.6}$ – среднее время задержки распространения сигнала в элементах И1.4, ИЛИ1.6 соответственно; $t_{з.р.ср.}^{1.5}$ – среднее время задержки распространения сигнала в триггере 1.5.

Согласно сказанному выше

$$f_{max} \le \frac{1}{t_{зд.р.}^{ЯП}}.$$

Для создания имитационной модели описанного счетчика был использован САПР MAX+plus II. В качестве элементной базы была выбрана микросхема EPM3032A семейства MAX3000A. Результаты моделирования полностью совпали с теоретическими выкладками, приведенными выше. Было получено время задержки ЯП $t_{зд.р.}^{ЯП} = 3\,нс$. Максимальная частота на входе счетчика ограничена выбранной элементной базой и составила $f_{max} = 220\,МГц$.

5. Выводы

На основании исследований, проведенных в данной работе, найдены обобщенные аналитические выражения, необходимые при построении быстродействующих унитарных биномиальных счетчиков любой разрядности. Полученные счетчики имеют регулярную структуру и удобны для реализации на ПЛИС. Также была получена программная модель и произведена оценка быстродействия. Максимальная частота на входе биномиального счетчика ограничена выбранной элементной базой и составила $f_{max} = 220$ МГц.

Список литературы: 1. *Борисенко А.А.* Введение в теорию биномиального счета: Монография. Сумы: ИТД «Университетская книга», 2004. 88 с. **2.** А.С. СССР 1298906. Счетчик импульсов. *Борисенко А.А., Воронов В.Г., Володченко Г.С., Куно Г.В.*

Поступила в редколлегию 29.04.2009

Борисенко Алексей Андреевич, д-р техн. наук, профессор, заведующий кафедрой электроники и компьютерной техники Сумского государственного университета. Научные интересы: теория информации и кодирования, сжатие данных. Адрес: Украина, 40000, Сумы, ул. Римского-Корсакова, 2, тел. 392-322, e-mail: electron@sumdu.edu.ua.

Петров Владислав Викторович, аспирант кафедры электроники и компьютерной техники Сумского государственного университета. Научные интересы: теория информации и кодирования. Адрес: Украина, 40000, Сумы, ул. Римского-Корсакова, 2, тел. 392-322, e-mail: electron@sumdu.edu.ua.

УДК 631.3.037.37

А.Е. ГОРЯЧЕВ

ПРЕОБРАЗОВАНИЕ ДВОИЧНЫХ И ФАКТОРИАЛЬНЫХ ЧИСЕЛ С ПОМОЩЬЮ СЧЁТНЫХ УСТРОЙСТВ

Рассматривается один из способов преобразования двоичных чисел в факториальные и факториальных чисел в двоичные как промежуточный шаг при генерации либо нумерации перестановок. Данный способ предполагает для осуществления преобразования использование счётных устройств. Разрабатывается структура реализующей данный способ системы, а также исследуются её основные характеристики.

1. Постановка задачи

Преобразование двоичных чисел в факториальные является промежуточным шагом при генерации перестановок на основе факториальных чисел [1]. Перестановки широко используются на практике для решения задач комбинаторной оптимизации, например, задачи поиска оптимального решения [2], а также при помехоустойчивой передаче данных и защите их от несанкционированного доступа. Задача обратного преобразования актуальна при необходимости нумерации перестановок и восстановления исходных данных после передачи [1, 3].

Перестановки на основе факториальных чисел получаются в соответствии со следующим алгоритмом, рассмотренным в [1].

Цифра старшего разряда факториального числа остаётся без изменений и считается первым элементом строящейся перестановки. Следующую цифру сравнивают с первым элементом перестановки; если она больше его, то необходимо увеличить данную цифру на 1, в противном случае она без изменений записывается как второй элемент перестановки. Цифры следующих разрядов сравнивают сначала с наименьшим элементом строящейся перестановки. Если значение цифры при этом больше значения данного элемента, то необходимо увеличить её на 1 и сравнивать с наименьшим из оставшихся элементов перестановки, и если значение цифры больше его, то она увеличивается на 1. Сравнение производится до тех пор, пока значение цифры не станет меньше того значения элемента строящейся перестановки, с которым данная цифра сравнивается, или же пока не будет произведено сравнение со всеми элементами. Таким образом, получается очередной элемент перестановки.

Для обратного перехода от перестановки к факториальному числу необходимо каждый элемент перестановки уменьшить на число единиц, равное количеству предыдущих элементов перестановки, меньших данного элемента. В результате каждый элемент перестановки преобразуется в цифру факториального числа.

Как следует из приведенного выше алгоритма, для получения перестановок требуются факториальные числа, которые, в свою очередь, получают путём преобразования степенных чисел.

Под факториальной системой счисления понимается выражение вида:

$$F_{<ф>} = X_n \cdot n! + X_{n-1} \cdot (n-1)! + \ldots X_i \cdot i! + \ldots + X_1 \cdot 1! + X_0 \cdot 0!, \qquad (1)$$

где $i = 0, 1, \ldots, 0 \le X_i \le i$.

Факториальная система счисления относится к системам счисления со смешанным основанием. Максимальное число Fmax в факториальной системе имеет вид: n(n-1)…i…210. Тогда

$$F_{<ф>max} = (n+1)! - 1. \qquad (2)$$

Минимальное число 00…0…00 в факториальной системе счисления Fmin = 0.

Диапазон факториальных чисел учитывает нуль, поэтому определяется как

$$P = Fmax + 1. \qquad (3)$$

Для решения задачи преобразования двоичных чисел в факториальные используется алгоритм, разработанный в [1]. Однако этот алгоритм имеет недостаток – сложность его технической реализации. Поэтому в данной работе ставятся следующие задачи:

1) найти простой алгоритм получения факториальных чисел;

2) разработать структуру устройства, реализующего предложенный алгоритм.

2. Решение задачи

Предлагаемый в работе метод преобразования двоичных и факториальных чисел заключается в организации одновременного счёта в направлении убывания чисел, начиная с исходного, и возрастания чисел до числа, которое необходимо получить. Наиболее простым и эффективным способом реализации этого алгоритма является использование суммирующего и вычитающего счётчиков.

При преобразовании двоичного числа в факториальное на вычитающий двоичный счётчик подаётся исходная двоичная комбинация. Особенностью суммирующего счётчика является то, что пересчёт ведётся в факториальной системе счисления. Таким образом, одновременно подавая счётный сигнал на оба счётчика, мы увеличиваем на единицу значение числа в факториальном счётчике и уменьшаем в двоичном. Для полного преобразования числа потребуется количество счётных импульсов, равное величине преобразуемого числа. Преимуществом данного метода является простота его реализации, что компенсирует снижение его быстродействия.

Схема, реализующая данный алгоритм, изображена на рис. 1. В её состав входят: блок управления, двоичный вычитающий счётчик, факториальный счётчик. Входные данные представляют собой исходное двоичное число, а также различные управляющие сигналы. С помощью блока управления осуществляется контроль за процессом преобразования. В начале цикла преобразования исходное двоичное число подаётся с блока управления на двоичный вычитающий счётчик. Факториальный счётчик при этом устанавливается в нуль. Тактирующие импульсы, подаваемые одновременно на входы двоичного и факториального счётчиков, уменьшают двоичное число, записанное в двоичном вычитающем счётчике, и увеличивают число в факториальном счётчике. При достижении нулевой комбинации в двоичном вычитающем счётчике формируется сигнал окончания счёта, запрещающий через блок управления подачу тактирующих импульсов до начала следующего цикла. Далее факториальное число, записанное в факториальном счётчике, передаётся на выход устройства.

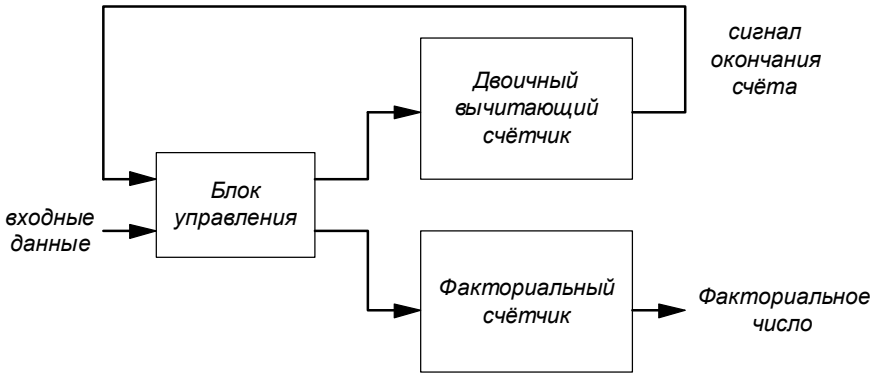


Рис. 1. Система преобразования двоичных чисел в факториальные

Обратное преобразование осуществляется по схожей схеме, но в этом случае используется двоичный суммирующий счётчик и факториальный вычитающий счётчик (рис. 2). Исходные данные, представляющие собой факториальное число, записываются в факториальный вычитающий счётчик. Разряды двоичного счётчика в исходном состоянии содержат нули. Счётными импульсами уменьшается число в факториальном счётчике и увеличивается в двоичном. Нулевая комбинация в факториальном счётчике свидетельствует об окончании счёта, далее происходит считывание двоичной комбинации с выхода двоичного счётчика.

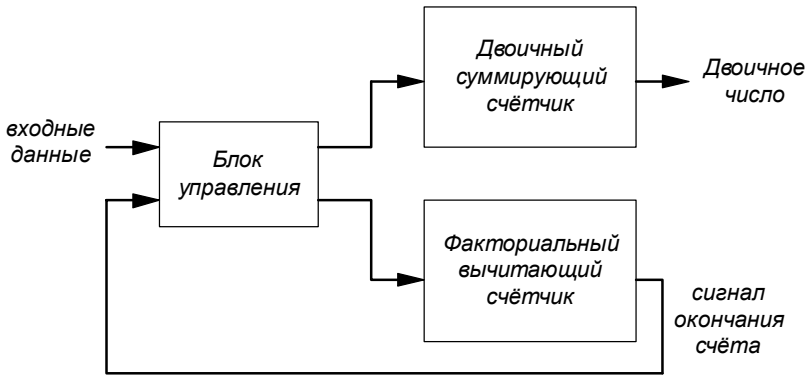


Рис. 2. Система преобразования факториальных чисел в двоичные

При необходимости осуществлять как прямое преобразование двоичных чисел в факториальные, так и обратное – факториальных чисел в двоичные может использоваться схема с двумя реверсивными счётчиками (рис. 3). Двоичный и факториальный счётчики в этом случае могут вести пересчёт как в суммирующем режиме, так и в вычитающем. За выбор режима работы схемы и корректную её работу в выбранном режиме отвечает блок управления на основании получаемых входных данных.

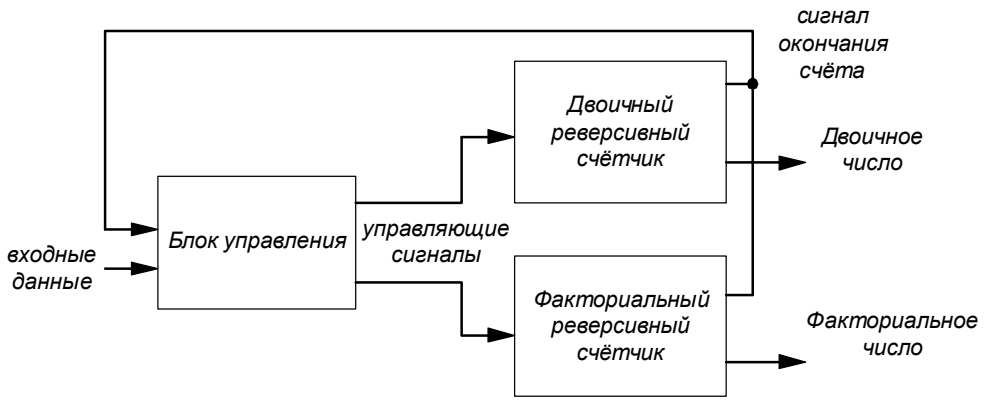


Рис. 3. Система, осуществляющая переход между двоичными и факториальными числами

Структурная схема блока управления рассматриваемой системы представлена на рис. 4. В неё входят: блок считывания входных данных, блок, отвечающий за хранение факториальных и двоичных чисел, блок управления направлением счёта, блок вывода данных. Блок считывания входных данных предназначен для приёма и последующей передачи в другие блоки устройства данных о направлении преобразования чисел (двоичные числа преобразуются в факториальные или наоборот), о преобразуемых числах (исходная комбинация двоичного или факториального числа в зависимости от направления счёта), о сигналах синхронизации с передающим и принимающим устройствами. Блок хранения осуществляет приём исходной комбинации от блока считывания и хранение её до начала цикла преобразования. После считывания всех входных данных исходная комбинация передаётся из блока хранения в счётчик, работающий в вычитающем режиме. Блок управления направлением счёта на основе информации, получаемой от блока считывания, устанавливает двоичный и факториальный счётчики в требуемый режим счёта. Блок вывода данных получает сигнал об окончании счёта от счётчика, работающего в вычитающем режиме, после чего обеспечивает вывод выходной комбинации из счётчика, работающего в суммирующем режиме.

Данная схема может использоваться и в системах, осуществляющих только один вид преобразования двоичных и факториальных чисел. Однако в этом случае отпадает необходимость в блоке управления направлением счёта.

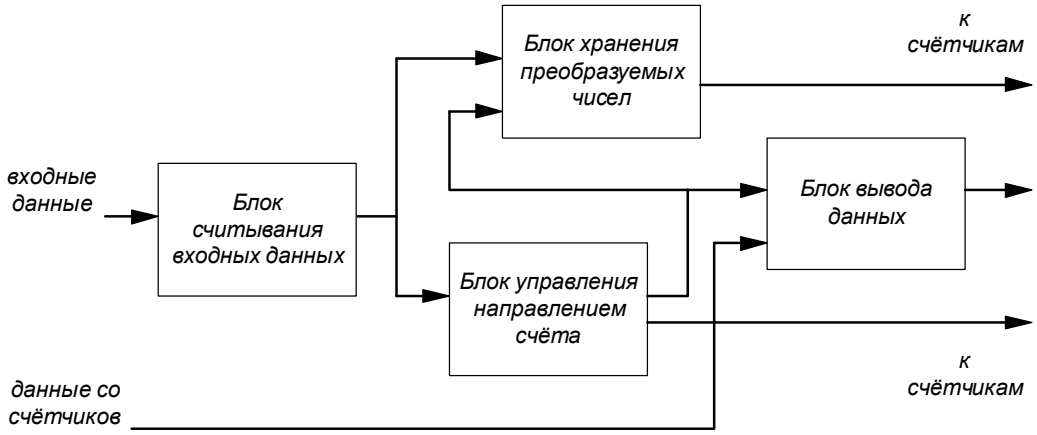


Рис. 4. Структура блока управления системы перехода между двоичными и факториальными числами

3. Структура и работа счётчиков

Согласно [4], факториальный счётчик представляет собой последовательность соединённых между собой двоичных счётчиков, коэффициент пересчёта которых возрастает на единицу для каждого последующего счётчика от 1 до N, где N – величина максимального разряда факториального числа. Для построения суммирующего факториального счётчика могут использоваться все методы, описанные в [4]. Отличие вычитающего факториального счётчика от суммирующего состоит в инвертировании сигналов между элементами счётчиков разрядов. В данном случае для построения счётчиков разрядов факториального счётчика предпочтительно использовать метод предустановки, так как счётчики разряда при переходе из нулевого состояния в первое необходимо устанавливать в максимальное значение данного разряда факториального числа. При использовании реверсивных счётчиков разрядов, позволяющих вести как прямой, так и обратный счёт, необходимо комбинировать методы управления сбросом счётчика, модификации межразрядных связей счётчика и предустановки.

Пример построения факториального счётчика показан на рис. 5. Сигналы f1, f2, … fN обозначают значение цифр разрядов факториального числа, p – сигнал переноса. Нулевой разряд факториального числа всегда принимает значение «0», поэтому не участвует в пересчёте.

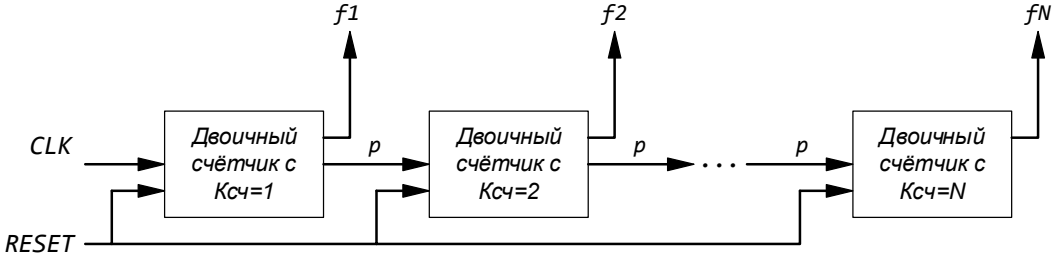


Рис. 5. Структурная схема N-разрядного факториального счётчика

4. Оценка параметров преобразователя

Рассмотрим требования, предъявляемые к счётчикам в зависимости от максимальной разрядности преобразуемого числа.

Для представления k-го разряда факториального числа требуется $\lceil \log_2(k+1) \rceil$ двоичных разрядов, следовательно, для k-разрядного факториального числа требуется следующее количество разрядов факториального счётчика:

$$K_\Phi = \sum_{i=1}^{k} \lceil \log_2(k+1) \rceil. \qquad (4)$$

Величина десятичного числа, соответствующего k-разрядному факториальному числу, будет определяться соответственно формуле (1):

$$D = \sum_{i=1}^{k} (X_k \cdot k!), \tag{5}$$

где X_k – значение k-го разряда факториального числа.

Отсюда число разрядов двоичного числа (и, соответственно, требуемая разрядность двоичного счётчика) определяется следующим образом:

$$K_Д = \lceil \log_2 D \rceil = \left\lceil \log_2 \left(\sum_{i=1}^{k} (X_k \cdot k!) \right) \right\rceil. \tag{6}$$

Пример. Определить разрядность двоичного и факториального счётчиков, требуемую для получения пятиразрядных факториальных чисел.

Решение. Максимальное пятиразрядное факториальное число в десятичной форме записи имеет следующий вид: F=43210. Для записи данного числа в двоичном виде потребуется следующее количество разрядов (формула (4)):

$$K_Ф = |\log_2(4+1)| + |\log_2(3+1)| + |\log_2(2+1)| + |\log_2(1+1)| + |\log_2(0+1)| = 9.$$

Величина десятичного числа D, соответствующего числу F (формула (5)):

$$D = 4 \cdot 4! + 3 \cdot 3! + 2 \cdot 2! + 1 \cdot 1! + 0 \cdot 0! = 119.$$

Число разрядов двоичного счётчика (формула (6)) $K_Д = \lceil \log_2 119 \rceil = 5$.

Время преобразования числа не зависит от способа преобразования (двоичное в факториальное или наоборот), а определяется исключительно количеством счётных импульсов, т.е. величиной числа D, а также частотой работы системы f:

$$T_P = D / f = \sum_{i=1}^{k} (X_k \cdot k!) / f. \tag{7}$$

Рассмотрим данную зависимость при фиксированной частоте работы системы, исследуя только число циклов её работы. Определим максимальное время преобразования в зависимости от разрядности факториального числа. В табл. 1 показаны максимальные значения параметра D для разрядности факториального числа Чр.

Таблица 1

Чр	2	3	4	5	6	7	8
D	5	23	119	719	5039	40319	362879

Графически рассматриваемая зависимость показана на рис. 6. Анализируя график зависимости, можно сделать вывод, что при увеличении разрядности факториального числа зависимость приближается к экспоненциальной.

Рис. 6. Зависимость максимального времени работы системы от разрядности преобразуемого числа при фиксированной частоте работы системы

Зависимость среднего времени преобразования числа от разрядности факториального числа представлена в табл. 2 и на рис. 7. Данную величину можно оценить, проведя исследование на некотором выбранном интервале, либо воспользовавшись следующей формулой:

$$D_{k\,cp} = (D_k + 1)/2\,, \tag{8}$$

где k – разрядность факториального числа; D_k – максимальное время работы системы для разрядности числа k.

Таблица 2

Чр	2	3	4	5	6	7	8
Dср п	3	12	60	360	2520	20160	181440
Dср т	3	14	57,3	349,5	2551,6	20119,7	181279,7

В табл. 2 Dср т – среднее время преобразования, полученное по формуле (8), Dср п – среднее время преобразования, полученное на основе практического исследования. Совпадение результатов практического исследования и расчётных данных свидетельствует об отсутствии ошибок при определении величины.

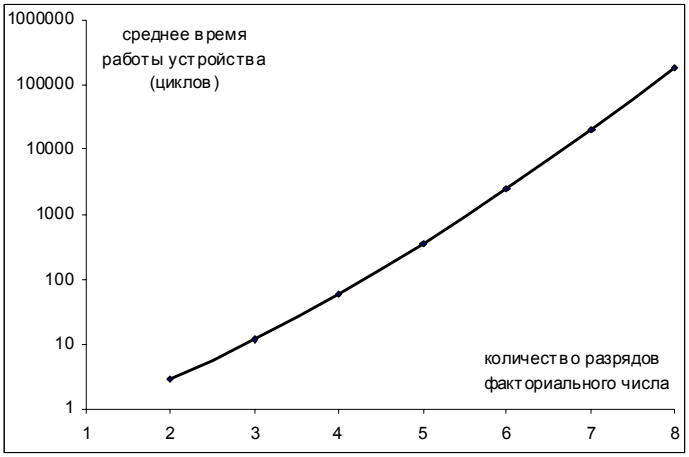


Рис. 7. Зависимость среднего времени работы системы от разрядности преобразуемого числа при фиксированной частоте работы системы

5. Выводы

Для решения поставленной задачи использовался метод, предполагающий применение двух счётчиков: суммирующего факториального и вычитающего двоичного. Основным достоинством этого метода является простота его реализации, а также возможность увеличения диапазона преобразуемых чисел путём добавления дополнительных структурных элементов факториального счётчика и увеличения коэффициента пересчёта двоичного счётчика. Данный метод может эффективно использоваться в устройствах, требующих повышенную надёжность и не предъявляющих высоких требований к быстродействию.

Список литературы: 1. *Борисенко А.А., Кулик И.А., Горячев А.Е.* Электронная система генерации перестановок на базе факториальных чисел // Вісник СумДУ. Технічні науки. 2007. №1. С.183-188. **2.** *Рейнгольд Э., Нивергельт Ю., Део Н.* Комбинаторные алгоритмы: теория и практика. М.: Мир, 1980. 477 с. **3.** *Borisenko A.A., Kalashnikov V.V., Kulik I.A., Goryachev A.E.* Generation of Permutations Based Upon Factorial Numbers. Eighth International Conference on Intelligent Systems Design and Applications. Kaohiung, Taiwan, 2008. P. 57-61. **4.** *Горячев А.Е.* Построение факториальных чисел на основе двоичных счётчиков // Вісник СумДУ. Технічні науки. 2008. №4.

Поступила в редколлегию 28.08.2009

Горячев Алексей Евгениевич, аспирант кафедры электроники и компьютерной техники Сумского государственного университета. Научные интересы: теория информации и кодирования. Адрес: Украина, 40007, Сумы, ул. Римского-Корсакова, 2, тел. 392-322, e-mail: electron@sumdu.edu.ua

УДК 681.518:004.93.1

А.С. ДОВБИШ, К.В. АЛТИННІКОВА

ІЄРАРХІЧНИЙ АЛГОРИТМ РОЗПІЗНАВАННЯ ЕЛЕКТРОНОГРАМ

Розглядається інформаційно-екстремальний ієрархічний алгоритм розпізнавання електронограм, одержаних в електронній мікроскопії в режимі мікродифракції, який дозволяє підвищити функціональну ефективність навчання системи при збільшенні потужності алфавіту класів. Оброблення електронограм в полярних координатах дає можливість зробити алгоритм інваріантним до зсуву та повороту.

Вступ

Розпізнавання одержаних в електронній мікроскопії в режимі мікродифракції електронограм є важливою науково-практичною задачею [1], актуальною в металургійній та хімічній промисловості, геології, кристалографії та інших галузях. Існуючі аналітико-геометричні методи розпізнавання електронограм [2] вимагають значних часових витрат і потребують високого рівня кваліфікації особи, що приймає рішення. Більшість відомих машинних алгоритмів розпізнавання зображень [2-4] орієнтовано на розв'язання модельних задач, які виключають перетин класів, характеризуються невисокою достовірністю розпізнавання і потребують на підготовчому етапі навчання нормалізації апріорно деформованих образів, що на практиці, як правило, є ускладненим.

Одним із шляхів вирішення цієї проблеми є використання ідей і методів інформаційно-екстремальної інтелектуальної технології (ІЕІ-технології), що ґрунтується на максимізації інформаційної спроможності системи розпізнавання шляхом введення в процесі навчання додаткових інформаційних обмежень [5]. У працях [6,7] досліджувалися у рамках ІЕІ-технології питання аналізу і синтезу систем розпізнавання електронограм за неієрархічним алгоритмом, який є чутливим до збільшення потужності алфавіту класів розпізнавання. У статті розглядається питання стиснення та оброблення відеоінформації в інформаційно-екстремальних алгоритмах навчання системи розпізнавання електронограм, які мають ієрархічну структуру.

1. Постановка задачі

Розглянемо задачу загального синтезу системи розпізнавання зображень. Нехай ефективність навчання розпізнаванню реалізацій класу X_m^o, $m = \overline{1,M}$, характеризується значенням E_m критерію функціональної ефективності (КФЕ). Відома навчальна матриця $\| y_{m,i}^{(j)} \|, i = \overline{1,N}, j = \overline{1,n}$, де N,n – кількість ознак розпізнавання і випробувань відповідно. Рядок матриці $\{y_{m,i}^{(j)} \mid i = \overline{1,N}\}$ утворює j-ту реалізацію образу, а стовпець $\{y_{m,i}^{(j)} \mid j = \overline{1,n}\}$ – навчальну вибірку з генеральної сукупності значень і-ї ознаки розпізнавання. Треба для структурованого вектора параметрів функціонування системи розпізнавання $g_m =< g_{m,1},...,g_{m,q},...,g_{m,Q} >$, які будемо називати параметрами навчання і для яких відомі обмеження $R_q(g_1,...,g_Q) \le 0$, шляхом організації послідовних ітераційних процедур знайти екстремальні значення координат вектора g_m, що забезпечують максимум КФЕ навчання системи розпізнавання:

$$E_{max}^{*} = \max_{G} E_m, \qquad (1)$$

де G – область допустимих значень параметрів навчання.

На етапі екзамену треба за побудованими в процесі навчання системи розпізнавання вирішальними правилами визначити належність реалізації образу, що розпізнається, до відповідного класу розпізнавання із заданого алфавіту $\{ X_m^o \}$.

Метою роботи є підвищення достовірності та оперативності розпізнавання електронограм за ІЕІ- технологією при збільшенні потужності алфавіту класів розпізнавання.

2. Алгоритм навчання системи розпізнавання

Математичну (категорійну) модель процесу навчання системи розпізнавання за IEI-технологією подамо у вигляді діаграми відображень множин. При обґрунтуванні гіпотези нечіткої компактності має місце нечітке розбиття $\widetilde{\Re}^{|M|} \subset \Omega$, де Ω – простір ознак розпізнавання. Введемо оператор θ нечіткої факторизації простору ознак: $\theta : Y \to \widetilde{\Re}^{|M|}$ і оператор класифікації y: $\widetilde{\Re}^{|M|} \to I^{|l|}$, який перевіряє основну статистичну гіпотезу про належність реалізацій $\{ x_m^{(j)} | j=\overline{1,n} \}$ нечіткому класу X_m^o. Тут l- кількість статистичних гіпотез. Оператор g: $I^{|1|} \to \Im^{|q|}$ шляхом оцінки статистичних гіпотез формує множину точнісних характеристик $\Im^{|q|}$, де $q=l^2$ – кількість точнісних характеристик. Оператор $\varphi : \Im^{|q|} \to E$ обчислює множину значень інформаційного КФЕ, який є функціоналом точнісних характеристик. Контур оптимізації геометричних параметрів нечіткого розбиття $\widetilde{\Re}^{|M|}$ шляхом пошуку максимуму КФЕ навчання розпізнаванню реалізацій класу X_m^o замикається оператором r: $E \to \widetilde{\Re}^{|M|}$.

Структурну діаграму відображень множин у процесі навчання за базовим інформаційно-екстремальним алгоритмом показано на рис. 1.

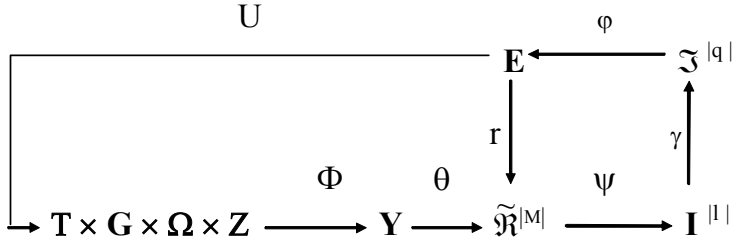


Рис. 1. Діаграма відображень множин у процесі навчання системи розпізнавання

Оператор U: $E \to G \times T \times \Omega \times Z$ регламентує процес навчання і дозволяє оптимізувати параметри його плану, які визначають, наприклад, обсяг і структуру випробувань, черговість розгляду класів розпізнавання та інше.

Вхідною інформацією для навчання за базовим алгоритмом є багатовимірна навчальна матриця $\| y_{m,i}^{(j)} | m=\overline{1,M}; i=1,N; j=\overline{1,n} \|$, де M, N, n – кількість класів, ознак розпізнавання та векторів-реалізацій класів відповідно; система полів контрольних допусків $\{\delta_{K,i}\}$ на ознаки розпізнавання і рівні селекції $\{\rho_m\}$ координат еталонних векторів-реалізацій, які за замовчуванням дорівнюють 0,5 для всіх класів розпізнавання.

Основні етапи реалізації алгоритму:

1. Побудова ієрархічної структури алфавіту класів розпізнавання. При цьому перший ярус структури складається з типових представників якісно відмінних класів електронограм, які визначають алфавіт класів розпізнавання $\{X_m^{(1)}\}$, а наступні яруси – з представників їх класів та підкласів. Кожна гілка вищого ярусу утворює страту, яка визначає свій алфавіт класів розпізнавання $\{X_{k,m}^{(r)}\}$, де k, r – номери страт і ярусів відповідно.

2. Для кожного класу будується спектрограма яскравості шляхом оброблення електронограми у полярних координатах за умови, що центр електронного пучка приймається за центр електронограми.

3. Для кожного алфавіту формується вхідна навчальна матриця $\| y_{k,m,i}^{(j)} \|^{(r)}$, вектори-реалізації якої утворюються шляхом квантування у часі відповідної спектрограми яскравості.

4. Для кожного алфавіту формується бінарна навчальна матриця $\| x_{k,m,i}^{(j)} \|^{(r)}$, елементи якої дорівнюють

$$x_{k,m,i}^{(j)} = \begin{cases} 1, & \text{if } \ y_{k,m,i}^{(j)} \in \delta_{K,i}; \\ 0, & \text{if } \ y_{k,m,i}^{(j)} \notin \delta_{K,i}. \end{cases} \tag{2}$$

5.Формування масиву еталонних двійкових векторів $\{x_{k,m,i} \mid m = \overline{1,M},\ i = \overline{1,N}\}$, елементи якого визначаються за правилом:

$$x_{k,m,i} = \begin{cases} 1, & \text{if } \ \frac{1}{n}\sum_{j=1}^{n} x_{k,m,i}^{(j)} > \rho_{k,m}; \\ 0, & \text{if } \ \text{else}, \end{cases} \tag{3}$$

де $\rho_{k,m}^{(r)}$ – рівень селекції координат вектора $x_{k,m}^{(r)} \in X_{k,m}^{(r)}$.

6. Розбиття множини еталонних векторів на пари найближчих «сусідів»: $\Re_{k,m}^{|2|} = <x_m, x_l>$, де x_l – еталонний вектор сусіднього класу $X_{k,m}^{(r)}$, за таким алгоритмом:

а) структурується множина еталонних векторів, починаючи з вектора x_1 базового класу X_1^o, який характеризує найбільшу функціональну ефективність системи розпізнавання;

б) будується матриця кодових відстаней між еталонними векторами розмірності M гM;

в) для кожного рядка матриці кодових відстаней знаходиться мінімальний елемент, який належить стовпцю вектора, найближчого до вектора, що визначає рядок. За наявності декількох однакових мінімальних елементів вибирається з них будь-який, оскільки вони є рівноправними;

г) формується структурована множина елементів попарного розбиття $\{\Re_{k,m}^{|2|} \mid m = \overline{1,M}\}$, яка задає план навчання.

7. Оптимізація кодової відстані $d_{k,m}^{(r)}$ відбувається за рекурентною процедурою. При цьому приймається $E_{k,m}^{(r)}(0) = 0$.

8.Процедура закінчується при знаходженні максимуму КФЕ в робочій області визначення його функції: $E_{k,m}^{*(r)} = \max_{\{d\}} E_m$, де $\{d\} = \{d_1, ..., d_k, ..., d_{max}\} \in [0; d(x_m \oplus x_1) - 1]$ – множина радіусів концентрованих гіперсфер, центр яких визначається вершиною еталонного вектора $x_{k,m}^{(r)} \in X_{k,m}^{(r)}$. При цьому множина $\{d\}$ є так само множиною кроків навчання системи розпізнавання.

Таким чином, базовий алгоритм навчання є ітераційною процедурою пошуку глобального максимуму інформаційного КФЕ в робочій області визначення його функції:

$$d_{k,m}^{*(r)} = \arg\max_{\{d\}} E_{k,m}^{*(r)}. \tag{4}$$

Параметри навчання системи розпізнавання за базовим алгоритмом – оптимальні кодові відстані $\{d_{k,m}^{*(r)}\}$ і оптимальні еталонні вектори-реалізації $\{x_{k,m}^{*(r)}\}$ для заданого алфавіту $\{X_{k,m}^{(r)}\}$ є обов'язковими вхідними даними для функціонування системи розпізнавання в режимі екзамену, тобто безпосереднього прийняття рішень.

Отже, основною функцією базового алгоритму навчання у рамках ІЕІ-технології є обчислення на кожному кроці навчання інформаційного КФЕ і організація пошуку його глобального максимуму в робочій області визначення функції критерію з метою визначення оптимальних геометричних параметрів розбиття простору ознак на класи розпізнавання.

Як критерій оптимізації параметрів навчання у рамках ІЕІ-технології може розглядатися будь-яка статистична інформаційна міра, яка є функціоналом від точнісних характеристик. Широкого використання в алгоритмах навчання набула модифікація інформаційної міри Кульбака [7], в якій розглядається відношення правдоподібності у вигляді логарифмічного відношення повної ймовірності правильного прийняття рішень P_t до повної ймовірності помилкового прийняття рішень P_f. Для рівноймовірних гіпотез, що характеризує найбільш важкий у статистичному розумінні випадок прийняття рішень, міру Кульбака подамо у вигляді

$$E_{k,m}^{(r)} = \log_2 \frac{P_{t,k,m}}{P_{f,k,m}} * [P_{t,k,m} - P_{f,k,m}] = \left| \begin{array}{l} P_{t,k,m} = 0{,}5D_{1,k,m} + 0{,}5D_{2,k,m}; \\ P_{f,k,m} = 0{,}5\alpha_{k,m} + 0{,}5\beta_{k,m}. \end{array} \right| =$$

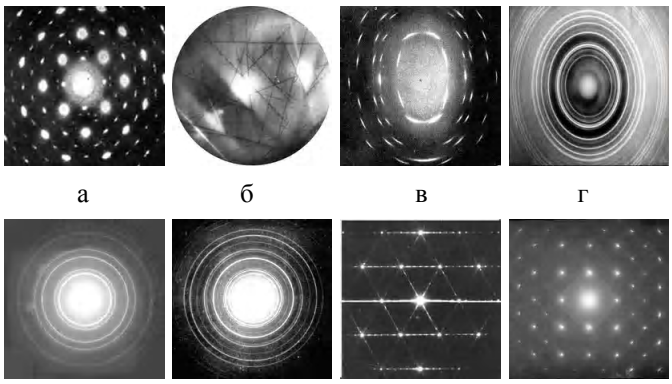
$$= \frac{1}{2}\log_2 \left(\frac{D_{1,k,m} + D_{2,k,m}}{\alpha_{k,m} + \beta_{k,m}} \right) \left[(D_{1,k,m} + D_{2,k,m}) - (\alpha_{k,m} + \beta_{k,m}) \right] =$$

$$= \log_2 \left(\frac{2 - (\alpha_{k,m} + \beta_{k,m})}{\alpha_{k,m} + \beta_{k,m}} \right) \left[2 - (\alpha_{k,m} + \beta_{k,m}) \right], \qquad (5)$$

де $D_{1,k,m}, D_{2,k,m}, \alpha_{k,m}, \beta_{k,m}$ – точнісні характеристики розпізнавання реалізацій класу $\{X_{k,m}^{(r)}\}$: перша і друга достовірності, помилки першого та другого роду відповідно.

3. Реалізація ієрархічного алгоритму розпізнавання електронограм

Для реалізації прикладу роботи системи розпізнавання електронограм використовувались отримані на просвічуючому електронному мікроскопі електронограми, зображені на рис. 2.



а б в г

д е є ж

Рис. 2. Електронограми: а – мозаїчного монокристалу; б – з Кікучі-лініями; в – текстури; г – полікристалу; д – алюмінію; е – NaCl; є – тодорокіту; ж – золота

Оброблення електронограм, зображених на рис. 2, здійснювалось у полярних координатах, які дозволяють зробити електронограми, що досліджуються, інваріантними до зсуву і повороту. При обробленні зображень в полярних координатах рядок навчальної матриці – вектор-реалізація образу формувався з ознак розпізнавання, які обчислювалися за формулою

$$\Theta_j = \frac{\sum_{i=1}^{N} \theta_i}{N}, \qquad (6)$$

де Θ_j – числове значення спектра в j-му радіусі кола зчитування, $j = \overline{1,R}$; θ_i – значення яскравості в i-му пікселі, $i = 1, N$; N – загальна кількість пікселів у колі зчитування.

Було побудовано ієрархічну структуру, яка зображена на рис. 3.

На першому ярусі ієрархічної структури (див. рис. 3) розташовано чотири класи основних типів електронограм: монокристалу – клас $X_1^{(1)}$ (електронограми з рефлексами у вигляді плям), з Кікучі-лініями – клас $X_2^{(1)}$, текстури – клас $X_3^{(1)}$ (з рефлексами у вигляді дуг) та полікристалу – клас $X_4^{(1)}$ (концентричні кільця). На другому ярусі розташовані: тодорокіт – клас $X_{1,1}^{(2)}$, золото – клас $X_{1,2}^{(2)}$ та алюміній – клас $X_{4,1}^{(2)}$ і NaCl – клас $X_{4,2}^{(2)}$.

Базовий алгоритм навчання проводився при значенні параметра поля контрольних допусків δ = 15. Графік залежності усередненого КФЕ від радіуса контейнера для класу $X_4^{(1)}$ зображений на рис. 4.

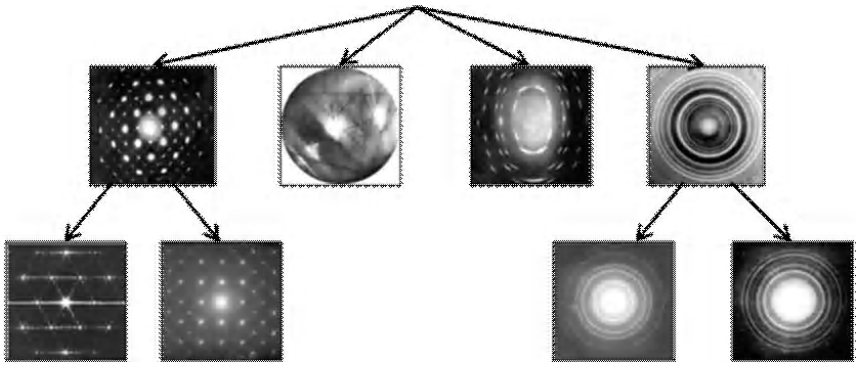


Рис. 3. Ієрархічна структура алфавіту класів розпізнавання

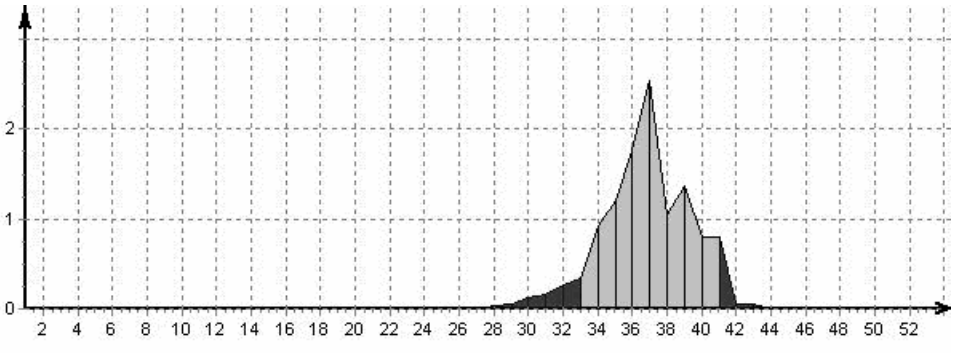


Рис. 4. Графік залежності КФЕ від радіуса контейнера для класу $X_4^{(1)}$

Світла ділянка на графіку рис. 4 визначає робочу область, в якій проводиться пошук глобального максимуму КФЕ (5). Аналіз графіка показує, що максимальне значення КФЕ для класу досягається при значенні радіуса контейнера d = 37 і становить 2.53. При цьому мають місце такі точнісні характеристики: перша достовірність D1=0,9, друга достовірність D2=0,9, помилка першого роду $\alpha = 0{,}1$, помилка другого роду $\beta = 0{,}1$.

Середнє значення КФЕ після проведення навчання для першого ярусу ієрархічної структури дорівнює $\overline{E} = 2{,}01$.

З метою перевірки працездатності розробленого ієрархічного алгоритму в режимі екзамену на монітор комп'ютера електронного растрового мікроскопа РЕМ-103М виробництва ВАТ «Selmi» (Суми, Україна) транслювалася електронограма алюмінію, яка в ієрархічній структурі (див. рис. 3) знаходиться у другому ярусі (клас $X_{4,1}^{(2)}$). Результати екзамену наведено на рис. 5.

Рис. 5. Інтерфейс програми в режимі проведення екзамену

Аналіз рис. 5 показує, що у режимі екзамену електронограма, що розпізнавалася, була правильно віднесена до відповідного класу за максимальним значенням геометричної

(дистанційної) функції належності реалізації образу відповідному гіперсферичному контейнеру і обчислювалася за формулою

$$\mu_{k,m}^{(r)} = 1 - \frac{d(x_{k,m}^{*} \oplus x^{(j)})}{d_{k,m}^{*}}, \qquad (7)$$

де $x_{k,m}^{*(r)}, x^{(j)}$ – еталонний вектор-реалізація класу $X_{k,m}^{(r)}$ і реалізація класу, що розпізнається, відповідно; $d_{k,m}^{*(r)}$ – оптимальний радіус контейнера класу $X_{k,m}^{(r)}$, побудований на етапі навчання.

Таким чином, ієрархічний алгоритм екзамену складається з послідовних процедур визначення максимальної функції належності (7) реалізації, що розпізнається, класу першого ярусу ієрархічної структури, переходу на відповідну страту другого ярусу, визначення максимальної функції належності для алфавіту класів цієї страти і так до тих пір, поки не буде знайдено фінальну вершину, яка не утворює свою страту.

Висновки

1. Запропоновано ієрархічний інформаційно-екстремальний алгоритм розпізнавання електронограм, одержаних в електронній мікроскопії у режимі мікродифракції, який шляхом оптимізації у процесі навчання параметрів функціонування за інформаційним критерієм дозволяє підвищити достовірність розпізнавання та зменшити чутливість системи до збільшення потужності інформаційного фонду електронограм. При цьому оброблення електронограм у полярних координатах дозволяє забезпечити інваріантність алгоритму розпізнавання до їх зсуву та повороту.

2. У перспективі при розширенні інформаційного фонду електронограм для побудови безпомилкових за навчальною вибіркою вирішальних правил необхідно здійснювати оптимізацію додаткових просторово-часових параметрів функціонування системи розпізнавання.

Список літератури: 1. *Томас Г., Гориндж М.Дж.* Просвечивающая электронная микроскопия материалов: Пер. с англ. / Под ред. Б.К. Вайнштейна. М.:Наука. 1983. 320 с. **2.** *Васильев В.И.* Распознающие системы: Справочник. 2-е изд., перераб. и доп. Киев: Наук. думка, 1983. 422 с. **3.** *Duda R. O., Hart P. E., Stork D. G.* Pattern Classification, second ed. John Wiley & Sons, New York, 2001. 738 p. **4.** *Shalkoff R.J.* Digital image processing and computer vision. New York-Chichester-Brisbane-Toronto-Singapore: John Wiley & Sons, Inc., 1989. 489 p. **5.** *Краснопоясовський А.С.* Інформаційний синтез інтелектуальних систем керування, що навчаються: Підхід, що грунтується на методі функціонально-статистичних випробувань. Суми: Видавництво СумДУ, 2003. 257 с. **6.** *Краснопоясовський А.С., Козинець М.В., Шелехов І.В.* Розпізнавання електронограм в електронній мікроскопії // Открытые информационные и компьютерные интегрированные технологии. Харьков: Нац. аерокосмический ун–т «ХАИ», 2002. Вып. 12. С. 140–146. **7.** *Довбиш А.С., Мартиненко С.С.* Інформаційно-екстремальний метод розпізнавання електронограм // Вісник СумДУ. Технічні науки. 2009. №2. С. 85-91.

Надійшла до редколегії 13.07.2009

Довбиш Анатолій Степанович, д-р техн. наук, професор, завідувач кафедри інформатики Сумського державного університету. Наукові інтереси: інформаційний аналіз і синтез інтелектуальних систем, що навчаються (самонавчаються). Адреса: Україна, 40035, Суми, вул. Заливна, б. 7, кв. 40, тел. (0542) 77-08-27, e-mail: kras@id.sumdu.edu.ua.

Алтиннікова Катерина Василівна, аспірантка кафедри інформатики Сумського державного університету. Наукові інтереси: інформаційний аналіз і синтез інтелектуальних систем, що навчаються (самонавчаються). Захоплення та хобі : туризм та плавання. Тел. (0542) 77-08-27, e-mail: meja13@mail.ru.

УДК 681.326:519.613

В.И. ХАХАНОВ, Е.И. ЛИТВИНОВА, И.А. ПОБЕЖЕНКО, TIECOURA YVES (ТИЕКУРА ИВ), NGENE CHRISTOPHER UMERAH (НГЕНЕ КРИСТОФЕР УМЕРАХ)

ТЕСТИРОВАНИЕ И ВЕРИФИКАЦИЯ HDL-МОДЕЛЕЙ КОМПОНЕНТОВ SOC. II

Предлагается алгебрологическая модель для вычисления критериев тестопригодности системных HDL-моделей, ориентированная на существенное повышение качества проектируемых компонентов цифровых систем на кристаллах (yield) и уменьшение времени разработки (time-to-market).

1. Введение

Модели верификации программного HDL-кода получены с использованием среды моделирования, тестопригодного анализа логической структуры HDL-программы для квазиоптимального размещения механизма ассерций [1], применяемой в hardware design and test. Разработанные критерии управляемости и наблюдаемости [2] использованы для оценки качества графа управления в целях его улучшения и эффективного диагностирования семантических ошибок. Представлены примеры вычисления функций и критериев тестопригодности, а также поиска ошибок в программном HDL-коде реального цифрового изделия [3].

2. Анализ тестопригодности графа управления

Учитывая, что автоматная модель программного продукта представлена взаимодействием операционного и управляющего автомата [4] (рис. 1), то наряду с моделированием транзакционного графа необходимо иметь возможность анализировать тестопригодность граф-схемы алгоритма управления (ГСА).

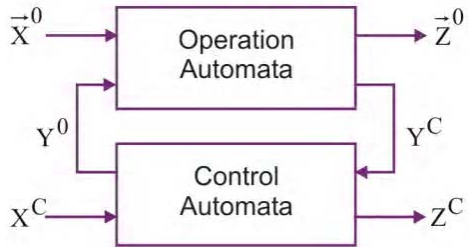


Рис. 1. Автоматная модель HDL-программы

Предлагается ГСА представить в виде содержательного графа управления (СГУ), который является подобным транзакционному графу. Здесь вершины есть операции программного кода, а дуги представляют условия перехода из одной вершины в другую для выполнения команды, обозначенной вершиной-стоком. Следовательно, для СГУ можно использовать процедуры, ранее разработанные для подсчета критериев тестопригодности транзакционного графа в части управляемости и наблюдаемости. Примером содержательного графа может служить рис. 2, имеющий 6 вершин и 9 дуг.

Подсчет управляемостей графа [3, с. 239, формулы (1), (4)], представленного на рис. 2, имеет следующий вид:

$$S_1 = T_3^3 T_4^1 \vee T_1^2;\ S_3 = T_3^3;$$

$$S_4 = T_3^3 T_4^1 T_6^1 \vee T_3^3 T_5^1 \vee T_3^3 T_8^2 T_9^1,$$

$$S_2 = T_3^3 T_4^1 T_6^1 T_7^2 \vee T_1^2 T_2^2 \vee T_3^3 T_5^1 T_7^2 \vee T_3^3 T_4^1 T_7^2 \vee T_3^3 T_8^2 T_9^1 T_7^2,$$

$$S_5 = T_3^3 T_8^2.$$

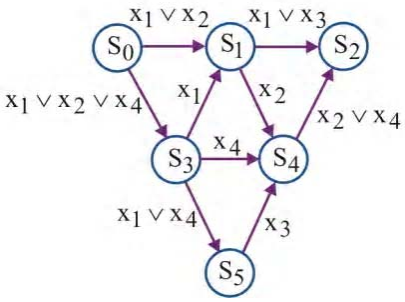


Рис. 2. Содержательный граф HDL-программы

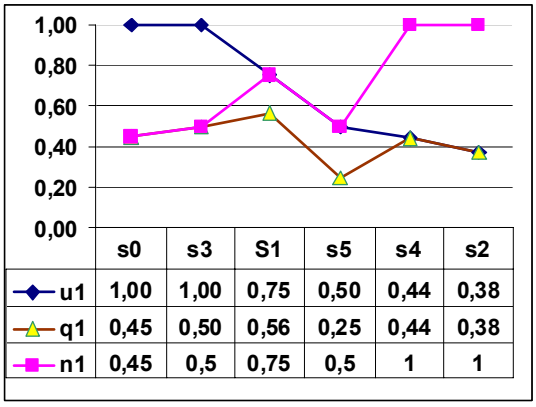
Подсчет наблюдаемостей графа [3, с. 239, формулы (1), (4)], представленного на рис. 2, содержит следующие выражения:

$$S_1 = T_2^2 \vee T_7^2 T_6^1,$$

$$S_3 = T_7^2 T_5^1 \vee T_7^2 T_9^1 T_8^2 \vee T_7^2 T_6^1 T_4^1 \vee T_2^2 T_4^1,$$

$$S_0 = T_2^2 T_1^2 \vee T_7^2 T_6^1 T_4^1 T_3^3 \vee T_7^2 T_5^1 T_3^3 \vee T_7^2 T_5^1 T_3^3 \vee T_7^2 T_9^1 T_8^2 T_3^3 \vee T_2^2 T_4^1 T_3^3.$$

$$S_4 = T_7^2;\ S_5 = T_7^2 T_9^1.$$



	s0	s3	S1	s5	s4	s2
u1	1,00	1,00	0,75	0,50	0,44	0,38
q1	0,45	0,50	0,56	0,25	0,44	0,38
n1	0,45	0,5	0,75	0,5	1	1

Рис. 3. Графики тестопригодности для графа управления

Для использования тестопригодности выполняется построение управляемости и наблюдаемости всех компонентов HDL-модели (рис. 3). Затем вычисляется обобщенная характеристика – тестопригодность каждого компонента как произведение управляемости и наблюдаемости:

$$Q = \frac{1}{n} \sum_{i=1}^{n} (U_i \times N_i). \qquad (1)$$

Далее интерес представляет создание таблицы тестопригодности, управляемости и наблюдаемости [2, 4], а также соответствующий им график для визуального контроля «плохих» компонентов. Фиксация определенной планки тестопригодности, ниже которой значения будут считаться неприемлемыми, позволит разработчику создавать ассерции и другие дополнительные средства повышения тестопригодности для проблемных функциональных блоков. Кроме того, средства повышения тестопригодности должны обеспечивать глубину диагностирования до функционального компонента и привязанных к нему операций в целях быстрого восстановления работоспособности программной HDL-модели. В целях построения алгоритмов поиска ошибок в программном коде можно использовать таблицу неисправностей, по аналогии с технологией тестирования hardware. Любопытное решение в процессе проверки функциональных блоков связано с сигнатурным анализом, где обобщенная сигнатура отождествляется с исправным поведением всего кода, а также с каждым компонентом. Любое несовпадение эталонной сигнатуры с фактической приводит к выполнению процедуры диагностирования и восстановления работоспособности HDL-модели путем исправления семантики кода.

Предложенная модель верификации HDL-проекта использует testbench, функциональное покрытие, механизм ассерций, описанную выше метрику оценки тестопригодности, таблицу неисправностей и вектор экспериментальной проверки (ВЭП), формируемый по заданным контрольным точкам путем сравнения сигнатур. Функциональное ограничение testbench связано с неразличимостью компонентов программного кода, в которых могут быть ошибки. Его основное назначение – проверка исправности HDL-модели. Поэтому в качестве дополнения к процедуре проверки придается механизм ассерций [4-6], основная цель которого с заданной глубиной – до программного компонента – определить место и вид ошибки на стадии выполнения диагностирования, после того, как testbench зафиксировал неправильное функционирование программного проекта. Векторная модель среды верификации [4, 6] имеет вид:

$$\begin{bmatrix} T_1 \\ T_2 \\ T_i \\ T_n \end{bmatrix} \oplus \begin{bmatrix} S_1 \\ S_2 \\ S_i \\ S_n \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ A_i \\ A_n \end{bmatrix} \xrightarrow{d} \begin{bmatrix} L_1 \\ L_2 \\ L_i \\ L_n \end{bmatrix} \tag{2}$$

В процессе моделирования выполняется сравнение реакций testbench и HDL-модели, что формирует состояния координат ассерционного вектора:

$$A_i = f(T_i, S_i) = T_i \oplus S_i,\ A_i = \{0,1,X\}.$$

Затем существенные $\{0,1\}$-координаты вектора ассерций маскируют матрицу достижимостей для получения списка программных блоков с ошибками путем выполнения одной из процедур, определенных выражениями:

$$\begin{cases} L_S(A) = \left(\bigcap_{\forall i(A_i=1)} A_i\right) \setminus \left(\bigcup_{\forall i(A_i=0)} A_i\right); \\ L_m(A) = \left(\bigcup_{\forall i(A_i=1)} A_i\right) \setminus \left(\bigcup_{\forall i(A_i=0)} A_i\right). \end{cases} \tag{3}$$

Практически, если выполнены условия тестопригодности и правильно расставлены ассерции в критических точках программного кода для диагностирования всех компонентов, то ВЭП может однозначно идентифицировать адрес (место) и тип ошибки на основе построенной ранее таблицы неисправностей – механизма ассерций.

3. Диагностирование по матрице достижимостей графа

Демонстрация технологии диагностирования неисправных блоков представлена следующим примером. Пусть имеется функционально-логический граф HDL-модели, изображенный на рис. 4.

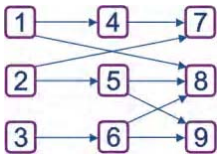


Рис. 4. Функционально-логический граф HDL-модели

Структура взаимосвязей графа представлена матрицей достижимостей:

M	1	2	3	4	5	6	7	8	9
1	1								
2		1							
3			1						
4	1			1					
5		1			1				
6			1			1			
7	1	1		1			1		
8	1	1	1		1	1		1	
9		1	1		1	1			1

(4)

Векторная модель диагностирования, заданная выражением (2), содержит списки блоков-предшественников для каждой вершины графа, которые получены из матрицы достижимостей. Каждой вершине графа ставится в соответствие ассерция, которая в процессе моделирования может быть доопределена значением {0,1}. В данном случае векторная модель поиска ошибочных программных блоков (2) для графа, представленного на рис. 4, трансформируется к виду:

T_1		S_1		$A_1 = X$		$L_1 = S_1$
T_2		S_2		$A_2 = X$		$L_2 = S_2$
T_3		S_3		$A_3 = X$		$L_3 = S_3$
T_4		S_4		$A_4 = 0$	d	$L_4 = S_1, S_4$
T_5	$\oplus$	S_5	=	$A_5 = X$	$\rightarrow$	$L_5 = S_2, S_5$
T_6		S_6		$A_6 = X$		$L_6 = S_3, S_6$
T_7		S_7		$A_7 = 1$		$L_7 = S_1, S_2, S_4, S_7$
T_8		S_8		$A_8 = 1$		$L_8 = S_1, S_2, S_3, S_5, S_6, S_8$
T_9		S_9		$A_9 = X$		$L_9 = S_2, S_3, S_5, S_6, S_9$

В результате выполнения диагностирования по системе уравнений (3), заключающейся в пересечении всех неисправных компонентов, которые соответствуют единичным координатам вектора ассерций, с последующим вычитанием объединения всех неисправных модулей, соответствующих нулевым координатам вектора A, получается список дефектных программных блоков (при условии существования только одного ошибочного модуля):

$$L_s(A_4 = 0, A_7 = 1, A_8 = 1) = L_7 \cap L_8 \setminus L_4 =$$

$$= S_1, S_2, S_4, S_7 \cap S_1, S_2, S_3, S_5, S_6, S_8 \setminus S_1, S_4 = S_2.$$

При использовании второго уравнения из выражения (3) можно получить список всех программных блоков, которые могут иметь ошибки, при условии существования нескольких дефектных компонентов:

$$L_m(A_4 = 0, A_7 = 1, A_8 = 1) = L_7 \cup L_8 \setminus L_4 =$$

$$= S_1, S_2, S_4, S_7 \cup S_1, S_2, S_3, S_5, S_6, S_8 \setminus S_1, S_4 =$$

$$= S_2, S_3, S_5, S_6, S_7, S_8.$$

Для иллюстрации следующего примера диагностического эксперимента можно убрать из рассмотрения первые два вектора, которые не являются существенными для процесса обработки списков неисправных блоков на основе анализа координат ассерционного вектора $L = d(A,S)$. Процедура вычисления списка одиночных дефектов имеет вид:

$A_1 = X$		$L_1 = S_1$		
$A_2 = X$		$L_2 = S_2$		
$A_3 = X$		$L_3 = S_3$		
$A_4 = X$	d	$L_4 = S_1, S_4$	$(L_8 \cap L_9) -$	
$A_5 = 0$	$\rightarrow$	$L_5 = S_2, S_5$	$-(L_5 \cup L_7)$	S_3, S_6
$A_6 = X$		$L_6 = S_3, S_6$		
$A_7 = 0$		$L_7 = S_1, S_2, S_4, S_7$		
$A_8 = 1$		$L_8 = S_1, S_2, S_3, S_5, S_6, S_8$		
$A_9 = 1$		$L_9 = S_2, S_3, S_5, S_6, S_9$		

Для множественных неисправных блоков на одном и том же векторе ассерций результат имеет большее число дефектных модулей:

$A_1 = X$		$L_1 = S_1$		
$A_2 = X$		$L_2 = S_2$		
$A_3 = X$		$L_3 = S_3$		
$A_4 = X$	d	$L_4 = S_1, S_4$	$(L_8 \cup L_9) -$	$S_3, S_6,$
$A_5 = 0$	$\rightarrow$	$L_5 = S_2, S_5$	$-(L_5 \cup L_7)$	S_8, S_9
$A_6 = X$		$L_6 = S_3, S_6$		
$A_7 = 0$		$L_7 = S_1, S_2, S_4, S_7$		
$A_8 = 1$		$L_8 = S_1, S_2, S_3, S_5, S_6, S_8$		
$A_9 = 1$		$L_9 = S_2, S_3, S_5, S_6, S_9$		

4. Диагностирование по векторно-логической форме графа

Интересна векторная форма модели проведения диагностического эксперимента. Здесь представлена матрица достижимостей, встроенная в модель диагностирования, где ассерции изображены в виде троичного вектора:

A	L	1	2	3	4	5	6	7	8	9
X	L_1	1								
X	L_2		1							
X	L_3			1						
X	L_4	1			1					
0	L_5		1			1				
X	L_6			1			1			
0	L_7	1	1		1			1		
1	L_8	1	1	1		1	1		1	
1	L_9		1	1		1	1			1
	L_s			1			1			

$$L_s = (L_8 \wedge L_9) \wedge$$
$$\wedge (\overline{L_5 \vee L_7}) =$$
$$= (001001000) =$$
$$= S_3, S_6$$

Результат поиска программных блоков выполнен при условии существования в проекте одного неисправного модуля, который записан в последней строке. Процедура диагностирования на основе ассерционного вектора A путем модификации (3) к выполнению векторных операций конъюнкции, дизъюнкции и отрицания

$$\begin{cases} L_s(A) = \left(\bigwedge_{\forall i(A_i = 1)} A_i\right) \wedge \left(\overline{\bigvee_{\forall i(A_i = 0)} A_i}\right); \\ L_m(A) = \left(\bigvee_{\forall i(A_i = 1)} A_i\right) \wedge \left(\overline{\bigvee_{\forall i(A_i = 0)} A_i}\right). \end{cases}$$

определена в правой части матричной модели диагностирования. Аналогичные вычисления для случая существования кратных дефектов в программных модулях проекта дают следующий результат:

A	L	1	2	3	4	5	6	7	8	9
X	L_1	1								
X	L_2		1							
X	L_3			1						
X	L_4	1			1					
0	L_5		1			1				
X	L_6			1			1			
0	L_7	1	1		1			1		
1	L_8	1	1	1		1	1		1	
1	L_9		1	1		1	1			1
	L_m			1			1		1	1

$$L_m = (L_8 \vee L_9) \wedge$$
$$\wedge (\overline{L_5 \vee L_7}) =$$
$$= (001001011) =$$
$$= S_3, S_6, S_8, S_9$$

В реальности результат диагностирования гарантирует наличие хотя бы одного дефектного блока из списка компонентов, подозреваемых в наличии неисправностей, определенных в [4].

5. Диагностирование по алгебрологической форме графа

Предлагается процедура диагностирования HDL-модели по структурно-логическому (транзакционному) графу, представленному алгебраической формой описания графовых структур [4]. Ее преимущества заключаются в компактности задания матрицы достижимостей, которая к тому же обладает свойством структуризации графа в форме задания всех путей, нагруженных на каждую вершину. Кроме того, сочетания дефектных компонентов, определяемые конъюнктивными термами, дают более точный результат диагностирования, по сравнению с заданием неисправных модулей в виде неупорядоченного множества элементов.

Для структуры, представленной на рис. 4, алгебраическая форма графа и вычисление списка одиночных неисправностей имеют следующий вид:

A	L	
X	$L_1 = S_1$	$L_s = (L_8 \wedge L_9) - (L_5 \vee L_7) =$
X	$L_2 = S_2$	$= (S_2S_5 \vee S_3S_6) - S_2S_5 =$
X	$L_3 = S_3$	$= S_3S_6$
X	$L_4 = S_1S_4$	
0	$L_5 = S_2S_5$	
X	$L_6 = S_3S_6$	
0	$L_7 = S_1S_4S_7 \vee S_2S_7$	
1	$L_8 = S_1S_8 \vee S_2S_5S_8 \vee S_3S_6S_8$	
1	$L_9 = S_2S_5S_9 \vee S_3S_6S_9$	
	$L_s = S_3, S_6$	

Процедура анализа логических функций графовой структуры содержит три пункта:

1. Все термы и переменные в ДНФ, соответствующей нулевому значению ассерционной координаты, равны нулю.

2. В других ДНФ, соответствующих единичному значению ассерционной координаты, левая часть конъюнктивного терма, включая переменную, ранее определенную нулем, удаляется.

3. Для единичных функций выполняется пересечение (объединение) оставшихся термов, если выполняется диагностирование одиночных (кратных) неисправных блоков.

Такие преобразования ДНФ путем специального моделирования нулевых сигналов переменных в единичных, относительно ассерций, логических функциях представления графа в целях получения списка неисправных программных модулей приведены в правой части алгебрологической модели диагностирования.

Следующий пример также подтверждает состоятельность анализа логических функций для вычисления множественных неисправных блоков:

A	L	
X	$L_1 = S_1$	$L_m = (L_8 \vee L_9) - (L_5 \vee L_7) =$
X	$L_2 = S_2$	$(S_1S_8 \vee S_2S_5S_8 \vee S_3S_6S_8$
X	$L_3 = S_3$	$S_2S_5S_9 \vee S_3S_6S_9) -$
X	$L_4 = S_1S_4$	$-(S_2S_5 \vee S_1S_4S_7 \vee S_2S_7) =$
0	$L_5 = S_2S_5$	$= S_3S_6S_8 \vee S_3S_6S_9$
X	$L_6 = S_3S_6$	
0	$L_7 = S_1S_4S_7 \vee S_2S_7$	
1	$L_8 = S_1S_8 \vee S_2S_5S_8 \vee S_3S_6S_8$	
1	$L_9 = S_2S_5S_9 \vee S_3S_6S_9$	
	$L_m = S_3, S_6, S_8, S_9$	

Для полученной функции L_m применение дополнительной ассерционной точки $A_6 = S_3S_6$ позволяет повысить глубину диагностирования до двух неисправных блоков:

$$L_m = S_3S_6S_8 \vee S_3S_6S_9 = \begin{cases} S_3S_6 \leftarrow A_6 = 1; \\ S_8 \vee S_9 \leftarrow A_6 = 0. \end{cases}$$

Проверка ассерционной точки $A_6 = 1$ не исключает блок S_8 из списка неисправных, но гарантирует факт наличия ошибки в компонентах S_3S_6. Поэтому после процедуры исправления некорректностей в блоках S_3S_6 необходимо повторять диагностический эксперимент. Что касается проверки ассерционной точки $A_6 = 0$, то она исключает наличие ошибок в блоках S_3S_6. Поэтому нулевая проверка всегда доставляет более ценную информацию для диагностического эксперимента. В целях уменьшения подозреваемого множества неисправных блоков следует еще одна проверка $(A_3 \vee A_9)$, которая устанавливает точный диагноз:

$$L_m = \begin{cases} = S_3 S_6 = \begin{cases} S_3 \leftarrow A_3 = 1; \\ S_6 \leftarrow A_3 = 0; \end{cases} \\ = S_8 \vee S_9 = \begin{cases} S_8 \leftarrow A_9 = 0\ ; \\ S_9 \leftarrow A_9 = 1. \end{cases} \end{cases}$$

После устранения ошибок диагностический эксперимент повторяется для поиска других составляющих кратных неисправностей HDL-блоков, которые могут присутствовать в программном коде [4].

Предложенная технология верификации является достаточно простой, ориентирована на обработку HDL-моделей большой размерности, включающей тысячи строк кода, который описывает системные структуры на языках VHDL, Verilog.

Технологический маршрут верификации программного кода представлен матрицей выполнения последовательно-параллельных процедур:

		P	
1	$F = f_2(P,S)$	$S = f_1(P)$	$T = f_3(P,S,F)$
2		$G = f_4(S,F)$	
3		$U = f_5(G,S,F)$	
4		$A = f_6(G,U,S,F,T)$	
5		$L_S = d(T,F,S,A)$	

,

которые имеют следующее содержание в виде 5 пунктов:

1) Создание HDL-модели, testbench, функционального покрытия по спецификации проекта в параллельном режиме.

2) Синтез транзакционного графа (вершины – элементы хранения информации и дуги – HDL-операторы, выполняющие транзакции между вершинами), представляющего структуру программного кода в компонентах и операторах HDL-языка. Операторы циклов размещаются в одном блоке. Граф не должен содержать глобальных обратных связей.

3) Определение тестопригодности графа путем подсчета наблюдаемостей всех вершин для планирования верификационного диагностического эксперимента.

4) Размещение ассерций в n% (25%) вершин, имеющих минимальные оценки наблюдаемостей. Их число должно обеспечивать заданную глубину диагностирования. Ассерции должны быть управляемыми, что обеспечивает их включение или отключение в процессе итеративной верификации в зависимости от результата предыдущей проверки. Данное свойство может существенно сократить время отладки программного кода. Такой подход условного (ассерционного) зондового диагностирования широко используется при тестировании сложных аппаратных цифровых изделий.

5) Диагностирование и исправление ошибок путем совместного моделирования HDL-кода, ассерций, на тестовых последовательностях testbench при условиях полноты, определенных функциональным покрытием. После устранения ошибок в дефектном блоке процедура моделирования и диагностирования повторяется.

6. Верификация DCT IP-core, Xilinx

Представленные модели верификации программного HDL-кода проверены на реальном проекте Xilinx IP-core в целях определения наличия в нем ошибок. При этом удалось получить положительный результат относительно неверной семантики работы программы для последующего исправления кода. Фрагмент модуля дискретного косинусного преобразования представлен листингом 1 [Xilinx.com]. Вся HDL-модель насчитывает 900 строк кода System Verilog.

Листинг 1

```
module Xilinx
‘timescale 1ns/10ps
module dct ( CLK, RST, xin,dct_2d,rdy_out);
output [11:0] dct_2d;
```

```
input CLK, RST;
input[7:0] xin; /* input */
output  rdy_out;
wire[11:0] dct_2d;
……..……..……..
/* The first 1D-DCT output becomes valid after 14 +64 clk cycles. For the first 2D-DCT output to be valid
it takes 78 + 1clk to write into the ram + 1clk to write out of the ram + 8 clks to shift in the 1D-DCT values + 1clk
to register the 1D-DCT values + 1clk to add/sub + 1clk to take compliment + 1 clk for multiplying + 2clks to add
product. So the 2D-DCT output will be valid at the 94th clk. rdy_out goes high at 93rd clk so that the first data
is valid for the next block*/
Endmodule
```

В соответствии с правилами тестопригодного анализа, приведенными выше, спроектирован транзакционный граф, представленный на рис. 5, который для DCT-module Xilinx имеет 28 вершин-компонентов (входная и выходная шины, логические и регистровые переменные, векторы и память).

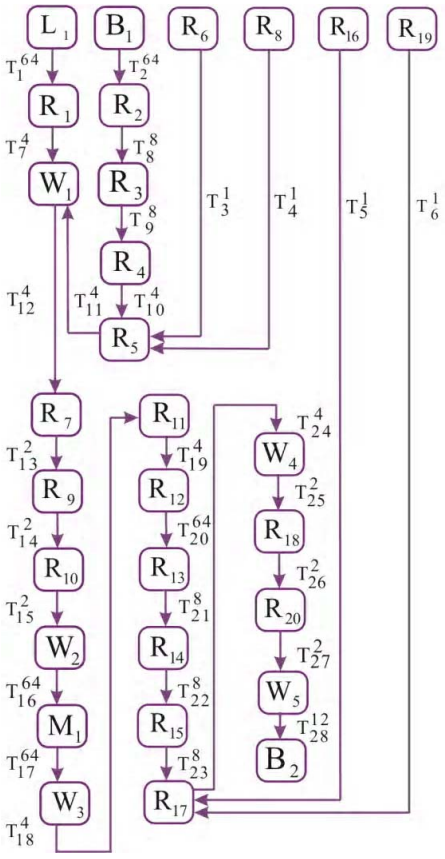


Рис. 5. Транзакционный граф Xilinx модели

Идентификатор дуги имеет верхний индекс, обозначающий число транзакций в программе между исходящей и входящей вершинами. Для каждой вершины строятся логические функции управляемости и наблюдаемости. Пример логической функции управляемости для вершины B_2 имеет следующий вид:

$$U^f(B_2) = T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}(T_5^1 \vee T_6^1 \vee T_{12}^4T_{13}^2T_{14}^2T_{15}^2T_{15}^{64}T_{17}^{64}T_{18}^4T_{19}^4T_{20}^{64}T_{21}^8T_{22}^8T_{23}^8(T_1^{64}T_7^4 \vee$$

$$\vee T_{11}^4T_2^{64}T_8^8T_9^8T_{10}^4 \vee T_{11}^4T_3^1 \vee T_{11}^4T_4^1)) =$$

$$= T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_5^1 \vee T_6^1T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4} \vee$$

$$\vee T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_{12}^4T_{13}^2T_{14}^2T_{15}^2T_{15}^{64}T_{17}^{64}T_{18}^4T_{19}^4T_{20}^{64}T_{21}^8T_{22}^8T_{23}^8T_1^{64}T_7^4 \vee$$

$$\vee T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_{12}^{4}T_{13}^{2}T_{14}^{2}T_{15}^{2}T_{15}^{64}T_{17}^{64}T_{18}^{4}T_{19}^{4}T_{20}^{64}T_{21}^{8}T_{22}^{8}T_{23}^{8}T_{11}^{4}T_{2}^{64}T_{8}^{8}T_{9}^{8}T_{10}^{4} \vee$$

$$\vee T_{11}^{4}T_{3}^{1}T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_{12}^{4}T_{13}^{2}T_{14}^{2}T_{15}^{2}T_{15}^{64}T_{17}^{64}T_{18}^{4}T_{19}^{4}T_{20}^{64}T_{21}^{8}T_{22}^{8}T_{23}^{8} \vee$$

$$\vee T_{11}^{4}T_{4}^{1}T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_{12}^{4}T_{13}^{2}T_{14}^{2}T_{15}^{2}T_{15}^{64}T_{17}^{64}T_{18}^{4}T_{19}^{4}T_{20}^{64}T_{21}^{8}T_{22}^{8}T_{23}^{8}.$$

Для остальных вершин аналогично выполняется вычисление ДНФ управляемостей.

Для вершины L_1 ДНФ наблюдаемости имеет вид:

$$N^{f}(L_1) = T_{28}^{12}T_{27}^{2}T_{22}^{2}T_{25}^{2}T_{24}^{4}T_{23}^{8}T_{22}^{8}T_{21}^{8}T_{20}^{64}T_{19}^{4}T_{18}^{4}T_{17}^{64}T_{16}^{64}T_{15}^{2}T_{14}^{2}T_{7}^{4}T_{1}^{64}.$$

Синтезированные логические функции задают все возможные пути управления, как во времени, так и в пространстве, что можно считать новой аналитической формой описания тестопригодности проекта. По ДНФ, следуя выражениям для подсчета тестопригодности [1], можно определить критерии управляемости (наблюдаемости) для всех компонентов HDL-модели. Здесь можно рассматривать два варианта (сценария) обсчета программной модели. 1) Учитывается только графовая структура, где вес каждой дуги равен 1, независимо от числа транзакций в программном коде. 2). Все дуги графа отмечаются реальным количеством транзакций, имеющих место между двумя вершинами транзакционного графа. Оценки тестопригодности описанных процедур могут существенно отличаться друг от друга. Пользователь должен определиться, что важнее только структура программного кода – применить первый сценарий или иметь более сложную и точную модель транзакций, распределенных во времени, на множестве графовых компонентов. В качестве примера ниже приводится процедура вычисления управляемости для B_2:

$$U(B_2) = \frac{1}{22 \times 6} \times (6+6+19+22+19+19) = 0{,}54 .$$

Применение аналогичных вычислений управляемостей (наблюдаемостей) для других вершин графа дает результат в виде графика, представленного на рис. 6, которые позволяют определить критические точки для установки необходимых ассерций.

Такой вершиной может быть компонент R_{15}, если транзакционный граф представлен одиночными дугами. Для случая, когда дуги отмечены реальным количеством транзакций, критические вершины принадлежат компонентам, находящимся ближе к выходной шине B_2. Здесь существенным представляется не структура графа, а вес входящей дуги, который в большей степени оказывает негативное влияние, если структурная глубина рассматриваемого компонента достаточно высока. Используется формула (1) вычисления тестопригодности с мультипликативными членами $U_i \times N_i$, что дает оценку ниже, чем любой из сомножителей (управляемость, наблюдаемость).

Если модифицировать формулу (1) исчисления тестопригодности для компонентов к следующему виду: $Q_i = U_i + N_i$, то кривая тестопригодности существенно поднимется вверх по оси ординат, чем обеспечивается меньший разброс параметров для каждой вершины. Данное обстоятельство фиксирует несколько отличные таблицы и графики, представленные ниже (рис. 7).

Интересным представляется поведение отдельных вершин. Например, управляемость вершины R_{17} в мультипликативном транзакционном графе HDL-кода неожиданно «упала» вниз по сравнению с графом единичных дуг. Это связано с высоким весом транзакций, поступающих на рассматриваемую вершину со стороны входных компонентов L_1, B_1, которые практически превращают в ноль значимость единичных транзакций от вершин R_{16}, R_{19}.После определения управляемостей и наблюдаемостей вершин транзакционного графа выполняется подсчет обобщенного критерия тестопригодности программного кода (формула (5), [3]). Для Xilinx DCT-модели такая оценка равна 0,382. Она характеризует качество проектного варианта, что представляется весьма существенным при сравнении нескольких альтернативных решений. В качестве примера позитивного использования разработанных моделей и методов был выполнен анализ тестопригодности программного кода дискретного косинусного преобразования (DCT) из Xilinx библиотеки. Построена транзакционная модель, вычислены характеристики тестопригодности и определены критические точки (R_1, R_5, R_9, R_{15}). В соответствии с числом и типами компонентов было разработано функциональное покрытие, фрагмент которого представлен листингом 2.

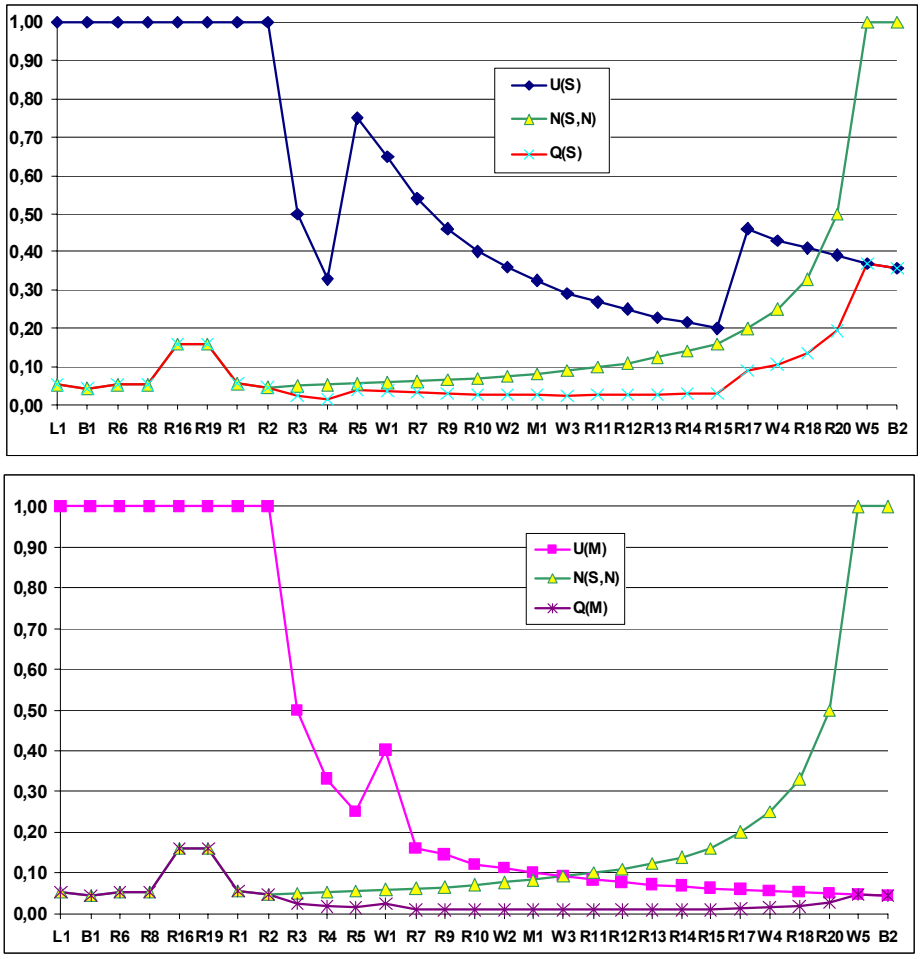


Рис. 6. Графики М-тестопригодности Xilinx модели

Листинг 2

```
c0: coverpoint xin
 {
 bins minus_big={[128:235]};
 bins minus_sm={[236:255]};
 bins plus_big={[21:127]};
 bins plus_sm={[1:20]};
 bins zero={0};
 }
 c1: coverpoint dct_2d
 {
 bins minus_big={[128:235]};
 bins minus_sm={[236:255]};
 bins plus_big={[21:127]};
 bins plus_sm={[1:20]};
 bins zero={0};
 bins zero2=(0=>0);
 }
endgroup
```

Для критических точек, определенных в результате анализа тестопригодности транзакционного графа, разработана ассерционная модель проверки основных характеристик дискретного косинусного преобразования. Существенный фрагмент кода механизма ассерций представлен листингом 3.

Листинг 3

```
sequence first( reg[7:0] a, reg[7:0]b);
 reg[7:0] d;
 (!RST,d=a)
 ##7 (b==d);
  endsequence
 property f(a,b);
 @(posedge CLK)
 // disable iff(RST||$isunknown(a)) first(a,b);
 !RST |=> first(a,b);
  endproperty
 odin:assert property (f(xin,xa7_in))
 // $display("Very good");
 else $error("The end, xin =%b,xa7_in=%b", $past(xin, 7),xa7_in);
```

В результате верификации программной HDL-модели дискретного косинусного преобразования в среде моделирования Active-HDL были найдены неточности в восьми строках исходного кода HDL-модели:

```
// add_sub1a <= xa7_reg + xa0_reg;//
```

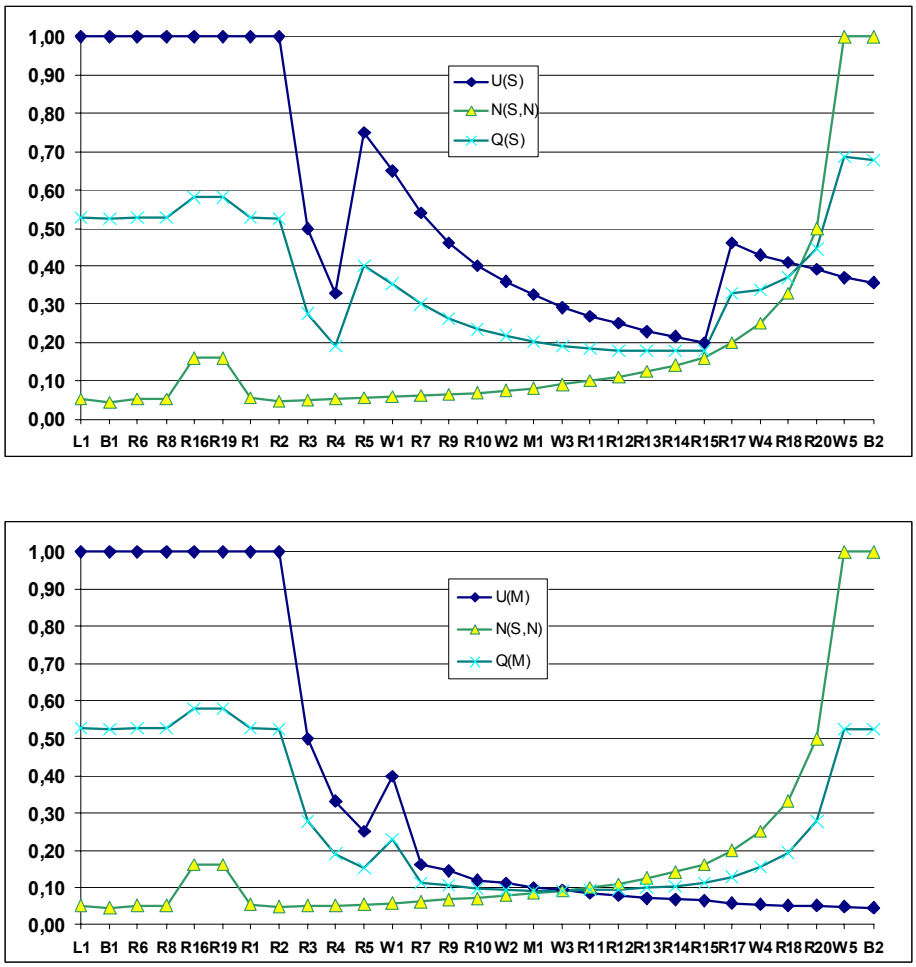


Рис. 7. Графики А-тестопригодности Xilinx модели

Последующее исправление ошибок привело к появлению исправленного фрагмента кода, который показан в листинге 4.

Листинг 4

```
add_sub1a <= ({xa7_reg[8],xa7_reg} + {xa0_reg[8],xa0_reg});
add_sub2a <= ({xa6_reg[8],xa6_reg} +{xa1_reg[8],xa1_reg});
add_sub3a <= ({xa5_reg[8],xa5_reg} +{xa2_reg[8],xa2_reg});
add_sub4a <= ({xa4_reg[8],xa4_reg} + {xa3_reg[8],xa3_reg});
```

```
end
else if (toggleA == 1'b0)
begin
add_sub1a <= ({xa7_reg[8],xa7_reg} - {xa0_reg[8],xa0_reg});
add_sub2a <= ({xa6_reg[8],xa6_reg} - {xa1_reg[8],xa1_reg});
add_sub3a <= ({xa5_reg[8],xa5_reg} - {xa2_reg[8],xa2_reg});
add_sub4a <= ({xa4_reg[8],xa4_reg} - {xa3_reg[8],xa3_reg});
```

7. Выводы

1. Предложен комплекс технологических мероприятий и рекомендаций, ориентированных на тестопригодный анализ и последующий синтез программных продуктов, пригодных для тестирования и верификации.

2. Показаны примеры анализа тестопригодности путем подсчета управляемости и наблюдаемости транзакционного и управляющего графов в целях определения критических точек с последующим решением практической проблемы поиска и устранения ошибок в реальном DSP-проекте от компании Xilinx.

3. Построены логические функции управляемости, наблюдаемости для двух реальных примеров и графики (таблицы), соответствующие им, которые дают возможность определить критические точки для последующего улучшения кода проекта путем установки ассерций.

4. Практическая значимость предложенных методик и моделей заключается в рыночной привлекательности и высокой заинтересованности технологических компаний в инновационных решениях проблемы эффективного тестирования и верификации программно-аппаратных изделий на системном уровне проектирования в целях уменьшения time-to market и повышения выхода годной продукции – yield.

5. Дальнейшие исследования будут направлены на разработку стандартных интерфейсов в целях последующей интеграции моделей, методов и программных средств в технологические маршруты проектирования цифровых систем на кристаллах.

Список литературы: 1. *Harry Foster, Adam Krolnik, David Lacey.* Assertion-based design.– Second edition. Kluwer Academic Publishers. Springer. 2005. 392 p. **2.** *Abramovici M., Breuer M.A. and Friedman A.D.* Digital System Testing and Testable Design. Computer Science Press. 1998. 652 p. **3.** *Хаханов В.І., Литвинова Є.І., Побеженко І.О., Ngene Christopher Umerah, Чумаченко С.В.* Тестирование и верификация HDL-моделей компонентов SOC. I // Радиоэлектроника и информатика. 2009. № 3. С. 38-45. **4.** *Хаханов В.И., Литвинова Е.И., Гузь О.А.* Проектирование и тестирование цифровых систем на кристаллах. Харьков: ХНУРЭ. 2009. 484 с. **5.** *Janick Bergeron, Eduard Cerny, Alan Hunter, Andrew Nightingale.* Verification Methodology. Manual for SystemVerilog. Springer. 2005. 528 p. **6.** *Bergeron, Janick.* Writing testbenches: functional verification of HDL models. Boston: Kluwer Academic Publishers. 2001. 354 p. **7.** *Шаршунов С.Г.* Построение тестов микропроцессоров. 1. Общая модель. Проверка обработки данных // Автоматика и телемеханика. 1985. №11. С. 145-155. **8.** *Jerraya A.A.* System Level Synthesis SLS. TIMA Laboratory. Annual Report. 2002. P. 65-75.

Поступила в редколлегию 11.07.2009

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, профессор кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Литвинова Евгения Ивановна, канд. техн. наук, доцент кафедры технологии и автоматизации производства РЭС и ЭВС ХНУРЭ. Научные интересы: автоматизация диагностирования и встроенный ремонт компонентов цифровых систем в пакете кристаллов. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-421. E-mail: kiu@kture.kharkov.ua.

Побеженко Ирина Александровна, аспирантка кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Tiecoura Yves, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Ngene Christopher Umerah, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

УДК 004.652

В. В. ЕВСЕЕВ, Ю. В. ШОВКОПЛЯС

СИНТЕЗ МОДЕЛИ УПРАВЛЕНИЯ ПРОЕКТАМИ РАЗРАБОТКИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ В УСЛОВИЯХ РИСКА И НЕОПРЕДЕЛЕННОСТИ

Рассматривается задача моделирования управления проектами разработки сложных технических систем в условиях неопределенности и риска. Формулируется постановка задачи моделирования. В результате проведенных исследований предлагается математическая модель планирования проектов разработки сложных технических систем в условиях неопределенности и риска. В качестве метода исследования был выбран метод Монте-Карло, который не используется в существующих программных средствах моделирования управления проектами. Разрабатывается программное средство планирования проектов в условиях неопределенности и риска. Описываются результаты, которые подтверждают правильность выбора метода исследования и эффективность предложенной математической модели.

Введение

В настоящее время большое количество организаций связывает свою работу с выполнением проектов или организовывает производство по принципу «управления от проектов» [1]. Учитывая высокую степень неопределенности и риска при планировании и управлении проектами сложных технических систем, актуальной является задача разработки модели и алгоритма управления проектами в таких условиях. Для решения данного вида задач используются методы вероятностного сетевого планирования. На данный момент к наиболее распространенным методам вероятностного сетевого планирования относятся метод оценки и анализа программ (PERT), метод статистических испытаний, или метод Монте-Карло, метод графической оценки и анализа программ (GERT). Наиболее распространен метод оценки и анализа программ [2].

Целью данной работы является получение новых практических результатов с использованием программного средства планирования проектов в условиях риска и неопределенности при использовании метода Монте-Карло.

1. Постановка задачи

Пусть исходными данными задачи планирования проектов в условиях неопределенности являются:

1) некоторый перечень (номенклатура) работ проекта;

2) последовательность выполнения работ, характеризуемая установленными связями между работами и заданная на графе;

3) функции плотности распределения вероятности продолжительности каждой работы.

Пусть результатами решения задачи планирования проектов в условиях неопределенности являются:

1) функция плотности распределения вероятности общей продолжительности проекта;

2) функции плотности распределения вероятности ранних и поздних сроков начала и окончания отдельных работ [3].

Тогда предлагается следующая формулировка постановки задачи исследования: необходимо построить математическую модель планирования проектов на основе предложенных исходных данных и результатов решения задачи планирования; разработать алгоритм реализации предложенной модели; разработать программное средство реализации предложенного алгоритма.

2. Обзор существующих методов решения задачи

Сетевые модели, состоящие из работ, взаимная последовательность и продолжительности которых заданы однозначно, называются детерминированными сетевыми моделями. Для их расчета применяется, как правило, метод Гантта, направленный на расчет критического пути, определяющего длительность всего проекта.

Так как предложенная постановка задачи моделирования управления и планирования проектов в условиях неопределенности предполагает неопределенность длительностей работ, то для разработки математической модели планирования и управления проектами в условиях неопределенности предлагается использовать вероятностные (или стохастические) сетевые методы. В настоящее время известно множество методов вероятностного сетевого планирования. Наиболее распространенными из них являются [1]:

1) метод оценки и анализа программ (Program Evaluation and Review Technique, PERT);

2) метод статистических испытаний, или метод Монте-Карло;

3) метод графической оценки и анализа программ (Graphic Evaluation and Review Technique, GERT).

Особенность метода PERT заключается в возможности учета вероятностного характера продолжительностей всех или некоторых работ при расчете параметров времени на сетевой модели. Он позволяет определять вероятности окончания проекта в заданные периоды времени и к заданным срокам.

Вместо одной детерминированной величины продолжительности для работ проекта задаются (как правило, экспертным путем) три оценки длительности: оптимистическая (работа не может быть выполнена быстрее определенного заданного времени); пессимистическая (работа не может быть выполнена медленнее, чем за определенное заданное время) и наиболее вероятная. Затем вероятностная сетевая модель превращается в детерминированную путем замены трех оценок продолжительностей каждой из работ одной величиной, называемой ожидаемой продолжительностью и рассчитываемой как средневзвешенное арифметическое трех экспертных оценок длительностей данной работы. Основной недостаток этого метода – недостаточная точность [3].

Наиболее известным из числа альтернативных вероятностных методов сетевого планирования является разработанный в США в 1966 году метод графической оценки и анализа (метод GERT) [1]. Он применяется в тех случаях организации работ, когда последующие задачи могут начинаться после завершения только некоторого числа из предшествующих задач, причем не все задачи, представленные на сетевой модели, должны быть выполнены для завершения проекта. Основу применения метода GERT составляет использование альтернативных сетей, называемых в терминах данного метода GERT-сетями. По существу GERT-сети позволяют более адекватно задавать сложные процессы в тех случаях, когда затруднительно или невозможно (по объективным причинам) однозначно определить, какие именно работы и в какой последовательности должны быть выполнены для достижения намеченного результата (т.е. существует многовариантность реализации проекта). Основной недостаток данного метода – чрезвычайная сложность расчетов и составления моделей [4].

Метод статистических испытаний (иначе называемый методом Монте-Карло) заключается в рассмотрении сети в качестве вероятностной модели, на которой оценки продолжительностей отдельных работ могут принимать любые значения, лежащие в крайних (минимум и максимум) указанных экспертами пределах, и даже выходить за эти пределы в той степени, в которой это допускают законы теории вероятностей. Сущность метода статистических испытаний состоит в получении на ЭВМ очень большого количества (порядка десятков тысяч) отдельных реализаций рассматриваемой сетевой модели, отличающихся друг от друга тем, что продолжительности работ во всех вариантах модели случайно выбираются по законам, характеризующим распределение каждой из отдельных оценок продолжительностей [5]. Данный метод является наиболее точным методом реализации вероятностных сетевых моделей и наиболее простым с точки зрения реализации на ЭВМ. Принимая во внимание перечисленные выше достоинства метода статистических испытаний, в данном исследовании предлагается разработать модель планирования и управления проектов в условиях риска и неопределенности с использованием метода Монте-Карло [4].

3. Математическая модель

Опишем математическую модель планирования проектов в условиях риска и неопределенности с помощью метода Монте-Карло.

Пусть задана последовательность выполнения работ, характеризуемая установленными связями между работами и заданная на графе. Имитационный эксперимент представляет из себя следующую последовательность действий:

1) экспертными методами задаются функции распределения длительностей всех работ:

$$f_{12}(t), f_{13}(t),..., f_{ij}(t), i = \overline{1,K}, j = \overline{1,K}, i \neq j, \quad (1)$$

где К – количество всех работ; $f_{ij}(t)$ – функция распределения вероятностей длительностей ij-й работы;

2) генерируется случайное множество длительностей всех работ $\{t(i,j)\}_1$, с учетом заданных законов распределения;

3) полученная сеть рассчитывается как детерминированная методом Гантта;

4) описанные выше шаги повторяются некоторое количество раз. Поскольку точность метода Монте-Карло пропорциональна $\sqrt{N}$, где N – число испытаний, процесс целесообразно повторять $N \approx 10^4 \div 10^6$ раз;

5) получается ряд $(T_{kpn})_{n=1..N} = (T_{kp1}, T_{kp2},..., T_{kpN})$,

где T_{kpN} – длительность критического пути, полученная на N-й итерации расчета модели;

6) средняя продолжительность критического пути определяется по формуле

$$\overline{T}_{kp} \approx M[T_{kp}] = \frac{\sum_{n=1}^{N} T_{kpn}}{N}, \quad (2)$$

здесь $\overline{T}_{kp}$ – средняя продолжительность критического пути;

7) дисперсия длительности критического пути L_{kp} определяется по формуле

$$\sigma^2[T_{kp}] = \sigma^2_{kp} = (t(i,j)), \quad (3)$$

где $\sigma^2[T_{kp}]$ – дисперсия длительности критического пути; $t(i,j)$ – длительность работы, начинающейся в событии i и заканчивающейся в событии j;

8) поскольку длительности $t(i,j)$ – независимые случайные величины, T_{kp} может трактоваться как случайная величина, распределенная по нормальному закону $N(T_{kp}, \sigma_{kp})$ с математическим ожиданием T_{kp} и дисперсией σ^2_{kp}, для которого функция плотности вероятности имеет вид

$$F(T_{kp}) = \frac{1}{\sigma^2_{kp}\sqrt{2\pi}} e^{\frac{(T_{kp}-\overline{T_{kp}})^2}{2\sigma^2_{kp}}}, \quad (4)$$

где $F(T_{kp})$ – функция плотности вероятности распределения длительностей критического пути.

4. Разработка алгоритма реализации модели

Алгоритм реализации предложенной модели планирования проектов в условиях риска и неопределенности выглядит следующим образом:

1) начало;

2) ввод исходных данных: количества экспериментов, графа последовательности выполнения работ и функции плотности распределения вероятности ранних и поздних сроков начала и окончания отдельных работ;

3) генерирование случайным образом совокупности длительностей работ;

4) определение длительности критического пути методом Гантта и, как следствие, определение длительности всего проекта;

5) сохранение рассчитанных величин;

6) увеличение количества итераций на единицу;

7) если количество итераций меньше введенного количества экспериментов – перейти к шагу 2, если равно – перейти к шагу 7;

8) расчет функции плотности распределения вероятности общей продолжительности проекта (критического пути) с использованием предложенной математической модели;

9) расчет функции плотности распределения вероятности ранних и поздних сроков начала и окончания отдельных работ;

10) вывод результатов;

11) конец.

Для реализации данного алгоритма было разработано специальное программное средство моделирования управления проектами в условиях неопределенности с использованием языка программирования высокого уровня C#.

5. Экспериментальные исследования

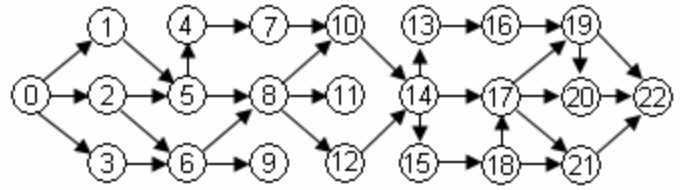
В качестве объекта исследования был выбран комплекс работ создания проекта технических средств информационно-образовательной среды, представленный на рис. 1.

Эксперимент 1.

Цель эксперимента – сравнение результатов расчета вероятностной сети методом Монте-Карло и методом PERT.

Исходные данные к эксперименту:

– топология сети представлена на рисунке;



– длительности работ – равномерно распределенные случайные величины в промежутке [0;2] дней.

По результатам расчета методом PERT наиболее вероятная длительность критического пути равна 5,1 дня.

Результаты проведения 1-го эксперимента представлены в табл. 1.

Таблица 1

Количество испытаний	10	100	1000	10000	100000	1000000
Средняя длительность критического пути	4,6	3,8	4,1	3,97	4,022	4,0012

По результатам проведения данного эксперимента видно, что при очень большом количестве экспериментов (и, соответственно, очень высокой точности) и при равномерном распределении длительностей работ, результаты, полученные методом PERT, существенно отличаются от результатов, полученных методом Монте-Карло, что доказывает более высокую точность метода статистических испытаний и пригодность его к расчету сетей с равномерным законом распределения длительностей работ.

Эксперимент 2.

Цель эксперимента – определение зависимости машинного времени, необходимого на расчет вероятностной сети, от количества проводимых экспериментов.

Исходные данные к эксперименту:

– топология сети представлена на рисунке;

– длительности работ – равномерно распределенные случайные величины в промежутке [0;2] дней.

Результаты проведения 2-го эксперимента представлены в табл. 2.

Таблица 2

Количество испытаний	10	100	1000	10000	100000	1000000
Машинное время расчета средней длительности критического пути	0,02	0,31	0,97	9,33	22,1	91,5

Результаты данного эксперимента показывают, что характер зависимости машинного времени, необходимого на расчет вероятностной сети, от количества проводимых экспериментов близок к линейному и что для достижения высокой точности результатов требуются достаточно высокие затраты машинного времени.

Выводы

Предложена математическая модель планирования проектов в условиях риска и неопределенности. С помощью имитационного моделирования был проведен ряд экспериментов, результаты которых (высокая точность полученных результатов) подтверждают правильность выбора метода статистических исследований в качестве метода исследования.

Научная новизна исследования состоит в том, что расчет временных характеристик вероятностной сети проводился с помощью метода Монте-Карло. При большом количестве статистических испытаний данный метод требует большого объема вычислительных ресурсов и времени, но методы, требующие меньшего объема времени и вычислительных ресурсов, дают лишь приближенные решения, использование которых при данной постановке задачи нецелесообразно.

Практическая значимость состоит в том, что экспериментальным путем выяснено, что метод Монте-Карло лучше остальных подходит для расчета сетей с равномерным законом распределения длительностей работ. Характер зависимости машинного времени, необходимого на расчет вероятностной сети, от количества проводимых экспериментов, близок к линейному.

Полученная имитационная модель планирования проектов была реализована на ЭВМ. Многократное использование разработанного программного средства показало, что оно позволяет проводить процесс моделирования планирования проектов с высокой степенью неопределнности и риска. Исходя из этого, можно сделать вывод об эффективности реализации полученной математической модели на ЭВМ.

Список литературы: 1. *Мазур И.И., Шапиро В.Д., Ольдерогге Н.Г.* Управление проектами: Учеб. пособие / Под общ. ред. И. И. Мазура. М.: Омега-Л, 2004. 664 с. **2.** *Федосеев В.В., Гармаш А.Н., Дайитбегов Д.М.* Экономико-математические методы и прикладные модели: Учеб. пособие для вузов. М.: ЮНИТИ, 2002. 391 с. **3.** *Новицкий Н.И.* Сетевое планирование и управление производством: Учеб.-практ. пособие. М.: Новое знание, 2004. 159 с. **4.** *Тулупьев А.Л., Николенко С.И., Сироткин А.В.* Байесовские сети. Логико-вероятностный подход. СПб.: Наука, 2006. 607 с. **5.** *Грачева М. В.* Анализ проектных рисков: Учеб. пособие для вузов. М.: ЗАО “Финстатинформ”, 1999.

Поступила в редколлегию 02.09.2009

Евсеев Виктор Владимирович, канд. техн. наук, профессор кафедры системотехники ХНУРЭ. Научные интересы: автоматизация проектирования сложных систем. Адрес: Украина, 61000, Харьков, пер. Хорошевский, 13, кв. 1, тел.: 372-56-16.

Шовкопляс Юрий Витальевич, аспирант кафедры системотехники ХНУРЭ. Научные интересы: математическое моделирование. Адрес: Украина, 61000, Харьков, ул. Конева, 13, кв. 70, тел. 712-47-78.

УДК 004.9

Ю.К. АПРАКСИН, И.О. ТУРЕГА

СИСТЕМА МОДЕЛИРОВАНИЯ ГЕТЕРОГЕННЫХ МИКРОКОНТРОЛЛЕРНЫХ СЕТЕЙ

Предлагается подход к построению многокомпонентной системы моделирования, ориентированной на исследование и проектирование микроконтроллерных сетей с разнородными узлами. Рассматривается модульная структура системы. Для описания моделей сетевых сущностей предлагается объектно-ориентированный подход. Анализ современных программных средств моделирования сетей подтверждает актуальность предложенной системы.

1. Введение

Сложные распределенные технические системы, к числу которых в первую очередь относятся сети, характеризуются разнотипным аппаратным и информационным обеспечением. Такие объекты относятся к классу гетерогенных систем. Они, с одной стороны, довольно широко востребованы в различных направлениях человеческой деятельности, но, с другой стороны, их конструирование и исследование существенно усложняется вследствие разнородности протекающих процессов и способов представления и передачи информации. Качественный анализ таких систем, позволяющий оценить результаты проектирования, требует использования универсальных средств моделирования. Существующие на сегодняшний день средства проектирования и моделирования в большинстве случаев ориентированы на множество протоколов и оборудование, используемых при построении глобальных и локальных компьютерных сетей. Применение данных инструментов для построения моделей сетей других типов сопряжено с необходимостью добавления новых протоколов, интерфейсов и классов устройств, что не всегда возможно ввиду узкой специализации системы, либо не оправдано из-за высокой трудоемкости требуемых изменений. Кроме того, данные средства во многих случаях распространяются на коммерческой основе и стоимость таких систем слишком высока для повсеместного их применения.

Ставится задача разработки системы проектирования и моделирования, использующей универсальные сетевые модели. Данная система должна быть ориентирована на проектирование и оптимизацию ЛВС, а также распределенных систем управления на базе универсальных микроконтроллеров, систем сбора информации и контроля, систем реального времени. Кроме того, следует учесть опыт создания подобных систем, а также возможность использования системы совместно с существующими средствами проектирования и моделирования.

2. Структура системы

В общем случае структура системы проектирования включает следующие базовые модули: ввода и хранения информации; проектирования; моделирования и анализа. Модуль ввода и хранения системы предоставляет возможность использовать уже имеющиеся экземпляры моделей типовых элементов, а также создавать новые на их базе, содержит средства для ввода требуемых свойств элементов, выполняет функции хранения с быстрым доступом к требуемой информации. Модуль проектирования позволяет создавать сетевые структуры на базе имеющихся элементов и моделей как автоматически с использованием итеративных и прочих алгоритмов, так и в ручном режиме, выполнять оптимизацию по ряду критериев. Модуль моделирования и анализа обеспечивает как непосредственно процедуру симуляции, так и сбор статистических данных. Исходной информацией для его работы является созданная модулем проектирования обобщенная структура взаимодействия моделей компонент распределенной системы.

Предлагаемая система в этом случае обладает свойствами универсальности и расширяемости, что позволяет отражать динамику функционирования сетевых объектов.

3. Модели сетевых объектов

Основными понятиями при рассмотрении структуры сетей являются: интерфейс, узел, топология. Иерархическая трехуровневая структура объектов рассматривает сетевые объекты на уровнях узла, слоя (совокупности узлов, соединенных интерфейсом), сети [1]. Однако для построения системы проектирования гетерогенных сетей необходимо более обширное множество моделей сетевых объектов, включающее модели интерфейсов, подробного рассмотрения сетевых соединений, использования различных сред передачи данных при различных соединениях и так далее. Выделим множество базовых объектов, позволяющих описать сетевую структуру, определим их свойства и связи. Наиболее удобным механизмом представления и моделирования взаимоотношений между объектами является объектно-ориентированный подход. В качестве инструмента используем наиболее популярный на сегодняшний день язык моделирования – UML (Unified Modeling Language – унифицированный язык моделирования).

Базовым объектом при рассмотрении сетевых моделей является узел. Структура узла сети, а также его поведение при обработке заявок от сетевых интерфейсов подробно рассмотрены ранее [2]. Однако в предложенных моделях узел рассматривался лишь как обработчик заявок, а интерфейсы вынесены за рамки модели, т.е. известны лишь характеристики потока заявок от каждого интерфейса. Но устройство может считаться сетевым только при наличии соответствующих интерфейсов связи и должно рассматриваться в контексте их работы, причем как по приему, так и по передаче данных. Таким образом, мы подходим к необходимости рассмотрения модели сетевого интерфейса.

Сетевой интерфейс состоит из приемника и передатчика. Приемник и передатчик имеют конвейерную структуру с точки зрения модели OSI (Open Systems Interconnection — взаимосвязь открытых систем). Каждая ступень такого конвейера будет состоять из накапливающего буфера и обработчика соответствующего протокольного уровня (рис. 1), причем структура передатчика соответствует развернутой в обратную сторону структуре конвейера приемника. В некоторых случаях буфер может быть общим, однако тогда система не сможет использовать преимущества конвейерной обработки. Особо следует отметить необходимость синхронизации производительности, а также соответствия размеров накопителей требованиям протоколов для различных уровней. Зачастую физический и канальный уровень имеют аппаратную реализацию, остальные уровни – программную. Существует ряд характеристик сетевых интерфейсов, таких как разрядность, способ синхронизации и другие. Данные характеристики влияют на количество физических сред, используемых для соединения с другими интерфейсами. В свою очередь при рассмотрении модели передающей среды необходимо учитывать время распространения сигнала, а также максимальную дистанцию его распространения.

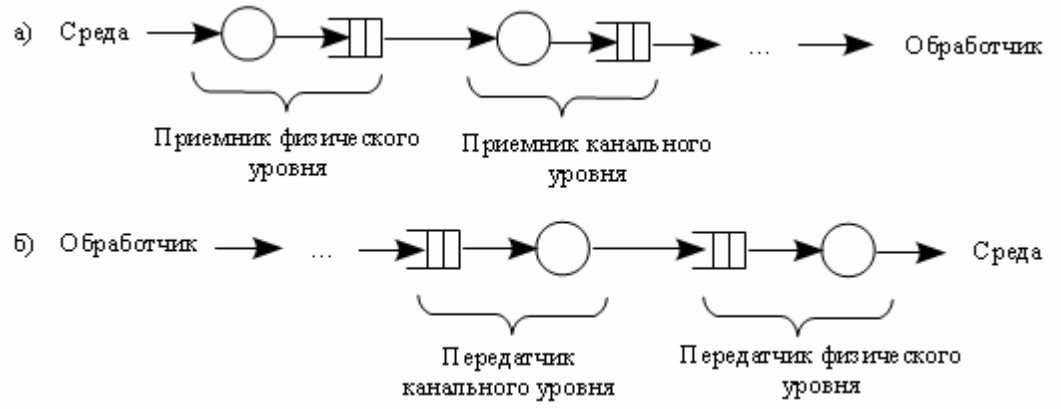


Рис. 1. Структура приемника (а) и передатчика (б)

Совокупность сетевых интерфейсов и обрабатывающего устройства образуют сетевой узел. Расширенная модель сетевого узла с учетом коммуникационных интерфейсов представлена на рис. 2.

Совокупность интерфейсов (приемников и передатчиков) узлов, соединенных посредством передающих сред, образует сетевые соединения (или каналы связи в широком смысле). Сетевые соединения могут быть однонаправленными (симплексными) и двунаправленными. Двунаправленные соединения могут быть полудуплексными и дуплексными. Однонаправленные соединения используют приемник либо передатчик каждого из интерфейсов, а также среды для передачи в одном направлении. Один интерфейс может использоваться двумя различными однонаправленными соединениями, а содержащий его узел является обработчиком заявок одного соединения и генератором для другого.

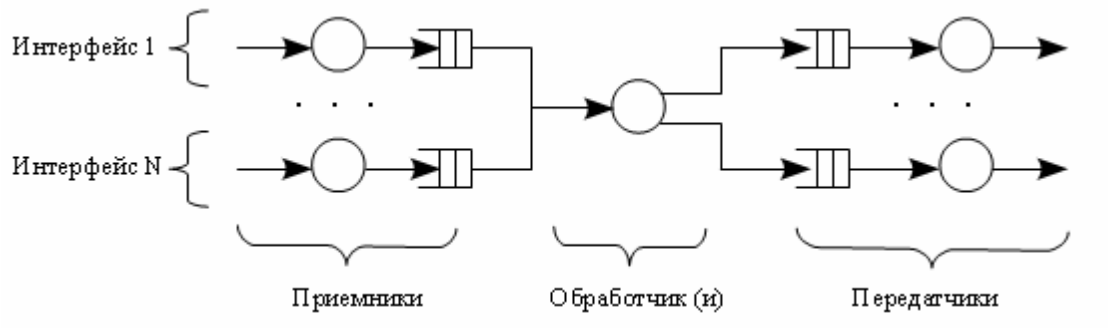


Рис. 2. Структура сетевого узла

Полудуплексные соединения используют приемники и передатчики узлов, а также одни и те же среды для передачи от любого из узлов, что требует использования средств арбитража (рис. 3).

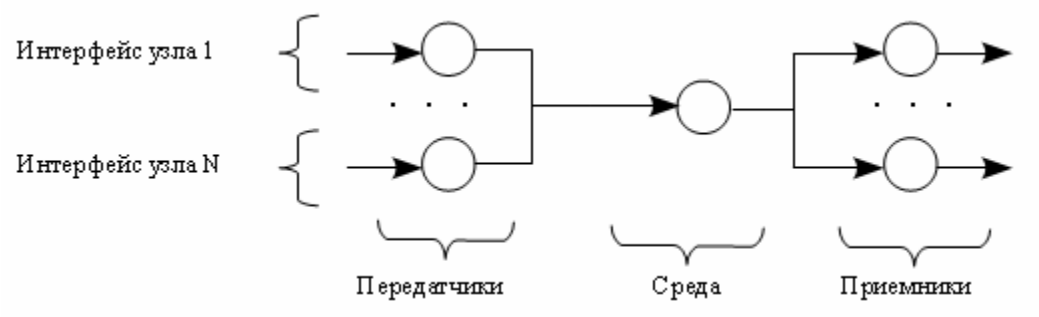


Рис. 3. Полудуплексное соединение

Дуплексные соединения позволяют передавать информацию одновременно в двух направлениях, что достигается использованием приемников и передатчиков, а также раздельных сред для каждого из направлений передачи (рис. 4). Таким образом, дуплексное соединение может быть рассмотрено как пара симплексных соединений между двумя узлами.

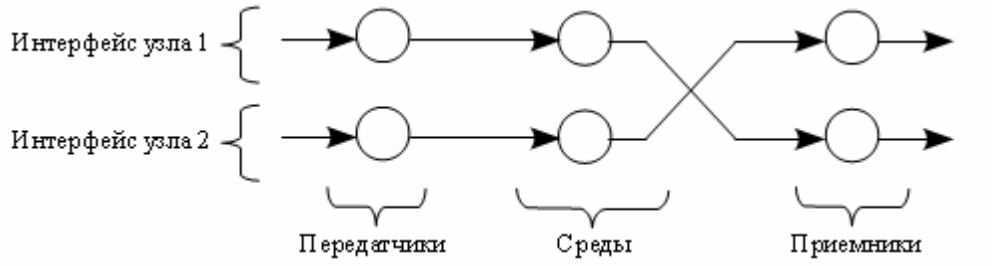


Рис. 4. Дуплексное соединение

Совокупность узлов, их коммуникационных интерфейсов, а также передающих сред, соединяющих интерфейсы, образует сети с различными топологическими структурами.

Для рассмотренных сетевых объектов разработана диаграмма классов с использованием элементов языка UML (рис. 5).

На диаграмме приведены классы, соответствующие моделям сетевых объектов, их свойства (атрибуты), а также отношения между ними. Так, объекты «Интерфейс» и

«Среда» связаны отношением зависимости. Различные типы соединений связаны с классом «Соединение» отношением обобщения. Большинство объектов на диаграмме связаны с другими объектами отношениями ассоциации и агрегирования (композиции), отражающими структурные связи между объектами. Именно поэтому данная модель наиболее важна при рассмотрении структуры сетей и их элементов.

4. Существующие системы

Использование рассмотренных объектов не ново. Существующие системы проектирования и моделирования, используя подмножества данных объектов, позволяют получать результаты в самых различных формах. Так, свободно распространяемая система моделирования NS2 использует в качестве примитивов узлы, интерфейсы и соединения. Также для моделирования поведения используются агенты, специальные объекты, описывающие протокольные действия для различных объектов. Система содержит средства анимации эксперимента, а дополнительные модули позволяют анализировать полученные в процессе моделирования статистические данные. К недостаткам данной системы можно отнести отсутствие встроенных графических средств построения сетевых структур. Коммерческие системы моделирования компьютерных сетей COMNET III и OPNET Modeler используют узлы и соединения различных типов в качестве компонентов модели. Данные системы включают библиотеки моделей сетевых объектов, соответствующих реальному оборудованию, позволяют имитировать наиболее распространенные протоколы и располагают встроенными средствами анализа и визуализации результатов моделирования.

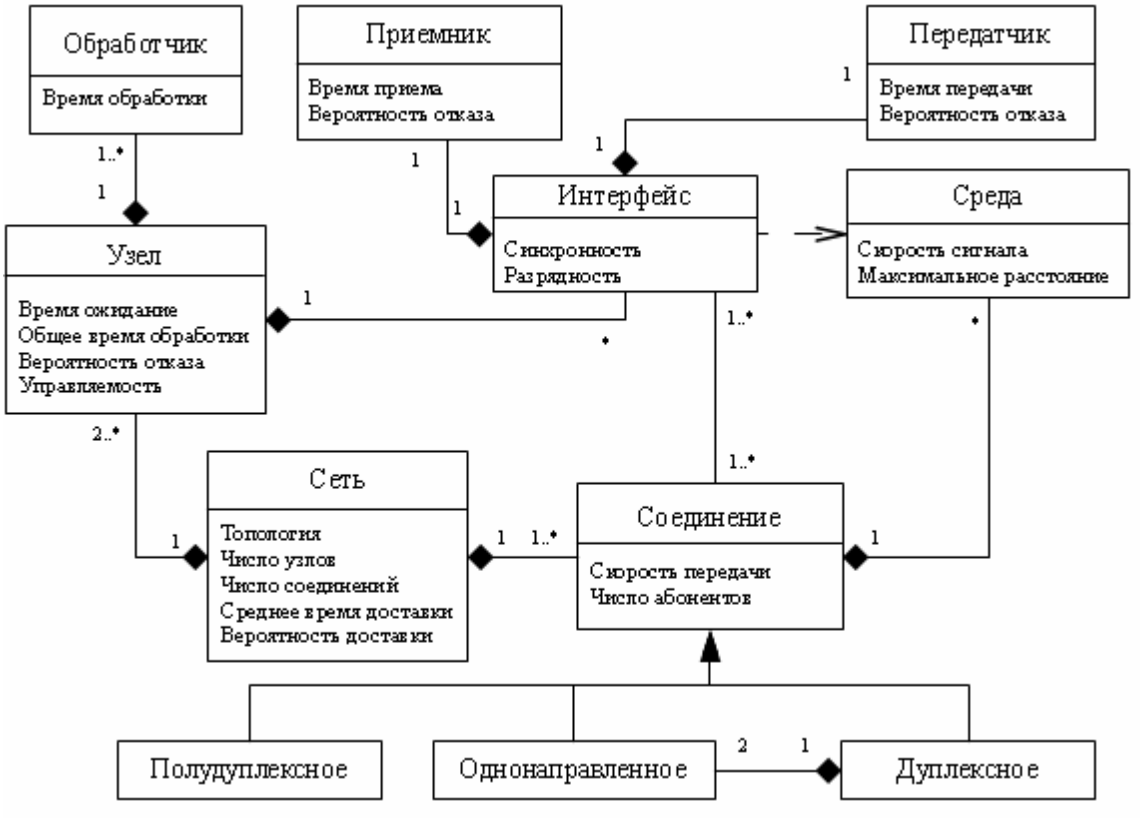


Рис. 5. Диаграмма классов для моделей сетевых объектов

Заключение

Предложенные в данной работе модели сетевых объектов могут быть использованы в системе проектирования и моделирования с наиболее общими задачами и входными параметрами. Благодаря этому появляется возможность выйти за рамки используемых систем моделирования компьютерных сетей и оптимизировать процессы разработки в таких перспективных направлениях как распределенные системы управления, встраиваемые системы, системы контроля доступа и автоматизированного сбора данных.

Список литературы: 1. *Бобылев С.Н., Шалимова Е. М.* Трехуровневая модель сетей микроконтроллеров // Международная научно-практическая конференция «Информационные технологии и информационная безопасность в науке, технике и образовании ИНФОТЕХ-2009». Материалы. Севастополь: Изд-во СевНТУ, 2009. С. 283–284. **2.** *Апраксин Ю.К., Турега И. О.* Аналитическая модель оценки потерь в микроконтроллере управляющей сети // Труды Одесского политехнического университета: Научный и производственно-практический сборник по техническим и естественным наукам. Одесса, 2009. № 1(31). С. 92–96.

Поступила в редколлегию 14.07.2009

Апраксин Юрий Константинович, д-р техн. наук, профессор Севастопольского национального технического университета. Научные интересы: программное и аппаратное обеспечение распределенных технических систем. Тел. (096) 750-07-77, e-mail: kvt@sevgtu.sebastopol.ua.

Турега Игорь Орестович, магистр, аспирант Севастопольского национального технического университета. Научные интересы: распределенные микропроцессорные системы. Тел. (067) 871-16-61, e-mail: tishka.ua@gmail.com.

УДК 004

З.В. ДУДАРЬ, М.В. ЗБИТНЕВА

МАРКОВСКИЕ МОДЕЛИ ДЛЯ ОЦЕНИВАНИЯ РЕЙТИНГА ВЕБ-САЙТА

Предлагается структура веб-агента, основанного на модели и цели. В качестве модели выбрана марковская модель, отражающая частоту и длительность посещаемости веб-сайта. Модели строятся для обобщенной и наиболее типичной структуры веб-сайта. В качестве цели выбрано повышение рейтинга веб-сайта. Веб-агент решает задачи анализа качества страниц веб-сайта, а именно выделяет наиболее приемлемые страницы для размещения рекламы, а также страницы, обладающие малой степенью полезности.

1. Введение

Качество контента определяется его актуальностью, отсутствием большого количества рекламы, стилем. Если в текстах присутствуют устаревшие сведения или информация о ваших товарах или услугах не соответствует действительности - это также говорит о том, что качество контента оставляет желать лучшего. О его качестве свидетельствует и то, насколько учтены особенности написания веб-текста.

Рейтинг сайта - комплексное понятие, привязанное к определённому отрезку времени. Под рейтингом сайта в поисковой системе понимается позиция, занимаемая сайтом по результатам запросов поисковой системы по конкретному ключевому слову или ключевой фразе. Под рейтингом сайта в Интернет-ресурсе, главная или одна из основных задач которого – ранжирование сайтов по определённым критериям, понимается рейтинг сайта по числу его посетителей в день.

Критерии, по которым производится ранжирование сайтов (расстановка, упорядочение, оценка), могут быть следующие:

– по числу посетителей сайта на конкретный час дня;

– по числу просмотренных веб-страниц сайта на конкретный час дня;

– по числу ссылок на сайт с других сайтов (“индекс цитирования”, “индекс популярности”);

– расстановка сайтов по алфавиту;

– расстановка сайтов по дате добавления в каталог;

– по оценке модераторов (экспертов) того каталога, в котором данный сайт зарегистрирован.

Целью данной работы является исследование первого критерия - числа посетителей сайта на конкретный час дня. Поэтому к задачам исследования относятся:

– построение полумарковской модели обобщенной структуры сайта;

– построение полумарковской модели наиболее типичной структуры сайта.

Так как закон перехода на страницы не является одним и тем же, следовательно, все модели считаются полумарковскими.

2. Суть исследования

В качестве автоматического средства выберем веб-агента. Наиболее употребимы несколько типов интеллектуальных агентов [4]: простые рефлексные агенты; рефлексные агенты, основанные на модели; агенты, основанные на цели; агенты, основанные на полезности.

Простейшим видом агента является простой рефлексный агент. Подобные агенты выбирают действия на основе текущего акта восприятия, игнорируя всю остальную историю актов восприятия. Данный тип работает с полностью наблюдаемой средой.

Наиболее эффективный способ организации работы в условиях частичной наблюдаемости - это применение агентов, основанных на модели. Такой агент поддерживает внутреннее состояние, отражающее акты восприятия. Для обновления информации необходимы сведения о том, как мир изменяется независимо от агента и как влияют на мир собственные действия агента. В случае необходимости при выборе решения учета цели добавляется учет цели в виде следующего типа агента, который определяет, достигнуто решение или нет. Это добавляет агенту гибкости, так как знания, на которые он опирается, представлены явно и могут быть модифицированы. Добавляя полезность, которая обозначает соответствующую степень m-ное значение «удовлетворенности», получают агентов, основанных на модели и на полезности. Полная спецификация функций полезности дает возможность выбирать решение при наличии конфликтующих целей или нескольких целей, каждая из которых не может быть достигнута со всей определенностью. В этом случае осуществляется способ взвешенной оценки вероятности успеха.

В качестве структуры агента для данной работы выбран агент, основанный на модели и цели (рис.1). В качестве модели выступают марковские модели. В качестве цели – реструктуризация структуры веб-сайта.

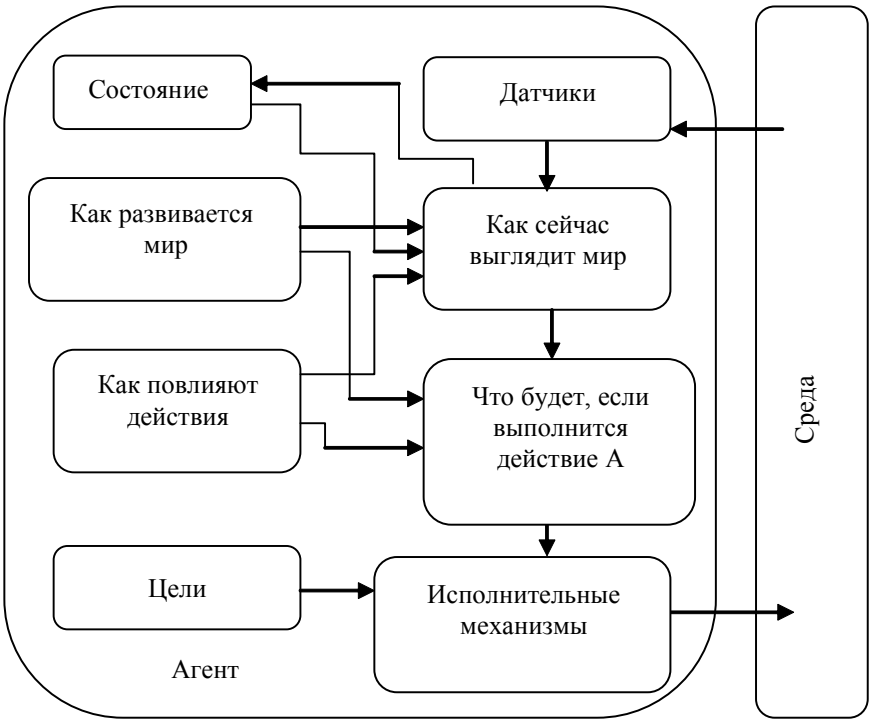


Рис. 1. Структуры агентов, основанные на модели и цели

Имеется класс математических моделей, аппроксимирующих широкий спектр случайных процессов. Такой класс составляют марковские процессы. Основные виды классифицируются в соответствии с задаваемыми значениями на временных и числовых множествах. Марковские цепи – это вид марковских процессов, которые описывают дискретный процесс с дискретным временем [5].

Рассмотрим посещение пользователей сайта как марковский процесс. Опишем его в виде обобщенной полумарковской модели.

Для структуры сайта, в котором из каждой страницы можно попасть на каждую, строится обобщенная полумарковская модель, состояния которой являются сообщающимися, а граф переходов – компонентой сильной связности. Граф состояний представлен на рис. 2. S_0 – первая или главная страница сайта, которая имеет 0-й уровень. S_{11} … S_{1n} – страницы, доступные для перехода с главной, описывают 1 уровень. S_{111} … S_{11n} – страницы, доступные для перехода с S_{11}, описывают уровень 2 т.д. Количество уровней вложенности и количество страниц разного уровня равно n. Таким образом, множество состояний образуют $\{S_0, S_{11}, S_{111} \dots S_{11n}, \dots S_{1n}, \dots S_{nn}\}$. Вектор начальных вероятностей (1, 0, …, 0). Рейтинг сайта, как и модели для него, как правило, оценивается за период времени – за месяц.

Рассмотрим первый показатель, применяемый для оценки рейтинга – число просмотренных веб-страниц. В качестве его первой составляющей являются переходы между страницами. Пусть количество страниц на одном уровне равно m, количество уровней - n.

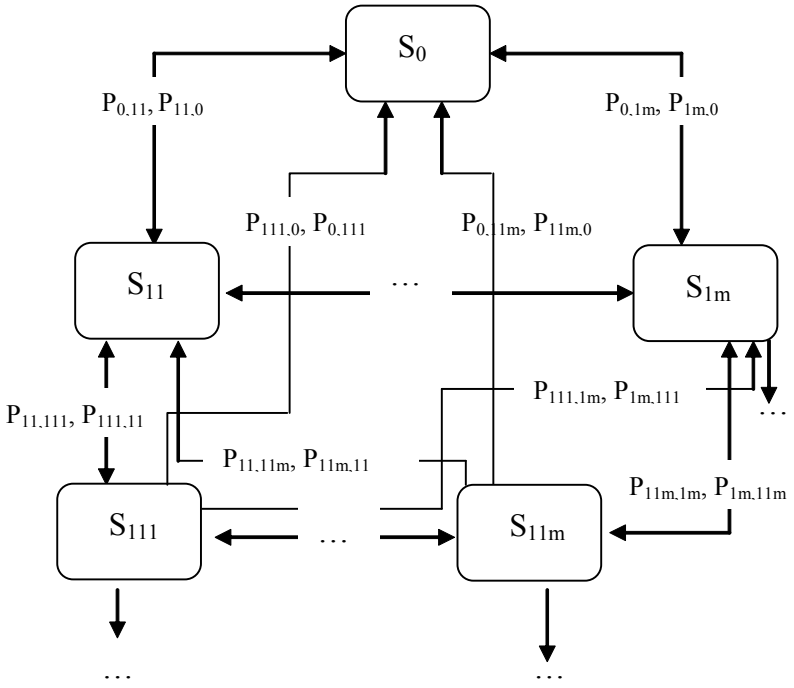


Рис. 2. Граф обобщенной полумарковской цепи веб-сайта

Матрица вероятностей переходов для данной модели имеет следующий вид:

$$P = [p_{ij}] = \begin{array}{c|cccccccc} & S_0 & S_1 & S_{11} & \cdots & S_{1m} & \cdots & S_n \\ S_0 & p_{0,0} & p_{0,1} & p_{0,11} & \cdots & p_{0,1m} & \cdots & p_{0,n} \\ S_1 & p_{1,0} & p_{1,1} & p_{1,11} & \cdots & p_{1,1m} & \cdots & p_{1,n} \\ S_{11} & p_{11,0} & p_{11,1} & p_{11,11} & \cdots & p_{11,1m} & \cdots & p_{11,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ S_{1m} & p_{1m,0} & p_{1m,1} & p_{1m,11} & \cdots & p_{1m,1m} & \cdots & p_{1m,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ S_n & p_{n,0} & p_{n,1} & p_{n,11} & \cdots & p_{n,1m} & \cdots & p_{n,n} \end{array},$$

где $P_{i,j}$ – вероятность перехода из состояния i в состояние j или вероятность перехода со страницы i на страницу j:

$$P_{ij} = \frac{k_j}{N},$$

здесь k_j – количество посещений j страницы; N – количество посещений всех страниц сайта за определенный промежуток времени.

Второй составляющей ячейки матрицы является длительность посещения страницы пользователем за определенный промежуток времени. Вероятность для нее рассчитывается по следующей формуле:

$$Q = \frac{d_j}{N_d},$$

где d_j – длительность посещения j страницы; N_d – суммарная длительность посещения всех страниц.

Сумма вероятностей дуг, исходящих из одной вершины, равняется единице, т.е. для каждой строки матрицы сумма вероятностей равна 1:

$$\sum_{S_{ij}} P_{ij} = 1.$$

Построим полумарковскую модель для сайта с наиболее типичной многоуровневой структурой (рис. 3). С каждой страницы есть возможность вернуться на главную. Движение между страницами заключается в движении внутри одного уровня и на уровень выше/ниже.

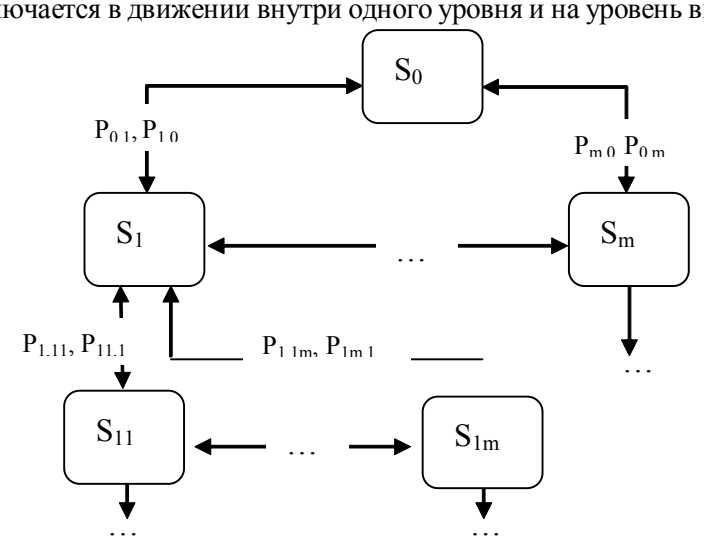


Рис. 3. Граф типичной полумарковской модели

Матрица вероятностей переходов для данной модели имеет следующий вид:

$$P = [p_{ij}] = \begin{array}{c|ccccccccc} & S_0 & S_1 & S_2 & S_{11} & \dots & S_{1n} & \dots & S_n \\ S_0 & p_{0,0} & p_{0,1} & p_{0,2} & p_{0,11} & \dots & p_{0,1n} & \dots & 0 \\ S_1 & p_{1,0} & p_{1,1} & p_{1,2} & p_{1,11} & \dots & p_{1,1n} & \dots & 0 \\ S_2 & p_{2,0} & p_{2,1} & p_{2,2} & p_{2,11} & \dots & p_{2,1n} & \dots & 0 \\ S_{11} & p_{11,0} & p_{11,1} & 0 & p_{11,11} & \dots & p_{11,1n} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ S_{1n} & p_{1n,0} & p_{1n,1} & 0 & p_{1n,11} & \dots & p_{1n,1n} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ S_n & p_{n,0} & 0 & 0 & 0 & \dots & 0 & \dots & p_{n,n} \end{array}$$

Выводы

К результатам исследования относятся построенные полумарковские модели для оценки рейтинга веб-сайта при использовании в качестве составляющих показателя числа просмотренных веб-страниц. Предложена программно-компонентная структура в виде веб-агента для решения поставленной цели. Научная новизна и практическая значимость выражается в разработанных моделях, а также во введенных составляющих критерия оценки рейтинга веб-сайта по числу посетителей.

Можно просчитывать горизонтальный и вертикальный показатель вероятности. Горизонтальный – по страницам в рамках одного уровня. Вертикальный – по заданным индексам страниц разных уровней, например, для просмотра всех страниц по определенной теме.

Для оценки рейтинга сайта предлагается вычислить экспериментальным путем значения следующих показателей: $P_{\delta min}$ – значение вероятности, полученное экспериментальным путем, ниже значения которого страница считается малопосещаемой, обладающей малой степенью полезности, качества. Действия, которые необходимо предпринять – это реформировать контент или объединить с другой страницей, или удалить, или поднять/опустить страницу на уровень выше/ниже; $P_{\delta max}$ – показатель вероятности, выше значения которого страница считается достаточно посещаемой для размещения на ней рекламы, для поддержания рейтинга сайта.

На данный момент проведены исследования применения марковских процессов посещения сайтов и поведения посетителей в работах Дишпэйнда и Кариписа, Андерсона и Домингоса, Янсена и Сараевича. Дишпэйнд и Карипис исследуют проблему предсказания поведения посетителя веб-сайта с помощью марковских моделей [1]. Андерсон, Домингос и Велд [2] описывают посетительское поведение посредством «относительных» марковских моделей. Янсен, Спинк, Сараевич конструируют модель поведения системы пользователь – сайт. Такой подход не привязан к специфике сайта [3].

В качестве перспектив развития данного исследования можно отметить выявление законов формирования вероятностей для построенных моделей, уточнение типа моделей, а также моделирование сочетаний слов в тексте при учете качества веб-контента.

Литература: **1.** *Deshpande M., Karypis G.* Selective Markov models for predicting Web page accesses. ACM Transaction on Internet Technology (TOIT). May 2004, Volume 4, Issue 2, Pages 163-184. ACM Press, NY, USA. **2.** *Jansen B., Sprink A., Saraevic T.* Real Life, Real users and Real needs: a Study and Analysis of User Queries on the Web. Information Processing and Management. 36: 2000 Elsevier. P. 207-227. **3.** *Anderson C., Domingos P., Weld D.* Relational Markov models and their application to adaptive web navigation // Proc. of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining table of contents, 2002. P. 143-152. ACM Press, NY, USA. **4.** *Рассел Стюарт, Норвиг Питер.* Искусственный интеллект: современный подход, 2-е изд.: Пер.с англ. М.: Издательский дом «Вильямс», 2006. 1408 с. **5.** *Андронов А.М., Копытов Е.А., Гринглаз Л.Я.* Теория вероятностей и математическая статистика: Учебник для вузов. СПб.: Питер, 2004. 461 с.

Поступила в редколлегию 17.06.2009

Дударь Зоя Владимировна, канд. техн. наук, профессор кафедры ПОЭВМ ХНУРЭ. Научные интересы: математическое и программно-техническое обеспечение взаимодействия крупномасштабных систем баз данных в динамическом окружении. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, e-mail: software@kture.kharkov.ua.

Збитнева Майя Вячеславовна, канд. техн. наук, доцент кафедры ПОЭВМ ХНУРЭ. Научные интересы: интеллектуальные агенты. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, e-mail: mayazbt@yandex.ru.

УДК 004.7; 004.8; 007.85

Н.В. ГОЛОВИЙ

РАЗРАБОТКА И ИССЛЕДОВАНИЕ БАЗЫ ДАННЫХ И БАЗЫ ЗНАНИЙ ДЛЯ ПРИНЯТИЯ РЕШЕНИЙ В ИНФОРМАЦИОННОЙ СИСТЕМЕ ОБСЛУЖИВАНИЯ БАНКОМАТОВ

Предлагается способ извлечения скрытых знаний и построение базы знаний информационной системы в сфере сервисного обслуживания банкоматов на основе нейросетевого подхода. Описывается функциональная структура инструментального программного средства для извлечения знаний.

1. Введение

В настоящее время сети банкоматов и платежных киосков достигают огромных размеров. Тенденция развития таких сетей говорит о том, что в ближайшее время парк банкоматов и платежных киосков будет возрастать. С увеличением размеров сети самообслуживания остро возникает вопрос сопровождения. Уменьшение времени простоя, а значит обеспечение доступности банкомата – одна из главных задач сопровождения. Уменьшить время реакции можно снижением нагрузки на персонал, что возможно путем создания единого центра мониторинга, консолидирующего в своем составе данные о работе сети самообслуживания из различных автоматизированных систем.

Несмотря на то, что в последнее время появилось множество различных коммерческих систем, в той или иной мере использующих методы Data Mining, проблема применения баз данных (БД) как дополнительного источника знаний при построении баз знаний (БЗ) интегрированных экспертных систем (ИЭС) по-прежнему остается актуальной.

Это обусловлено сложностью стыковки двух различных технологий – извлечения знаний из баз данных и традиционной инженерии знаний, применяющей в качестве основного источника знаний эксперта (или экспертов). При использовании БД как дополнительного источника знаний при построении БЗ информационной системы возникает ряд проблем, начиная от формирования выборки для анализа и заканчивая верификацией полученных фрагментов БЗ.

2. Обнаружение знаний в данных

Процесс поиска полезных знаний в “сырых” данных включает в себя вопросы: подготовки данных, выбора информативных признаков, очистки данных, применения методов Data Mining (DM), постобработки данных и интерпретации полученных результатов. Безусловно, “сердцем” всего этого процесса являются методы DM, позволяющие обнаруживать знания.

Этими знаниями могут быть правила, описывающие связи между свойствами данных (деревья решений), часто встречающиеся шаблоны (ассоциативные правила), а также результаты классификации (нейронные сети) и кластеризации данных (карты Кохонена) и т.д.

Процесс извлечения знаний состоит из нескольких шагов:

Подготовка исходного набора данных. Этот этап заключается в создании набора данных, в том числе из различных источников, выбора обучающей выборки и т.д. Для этого должны существовать развитые инструменты доступа к различным источникам данных. Желательно иметь поддержку работы с хранилищами данных и наличие семантического слоя, позволяющего использовать для подготовки исходных данных не технические термины, а бизнес понятия.

Предобработка данных. Для того чтобы эффективно применять методы Data Mining, следует обратить внимание на вопросы предобработки данных. Данные могут содержать пропуски, шумы, аномальные значения и т.д. Кроме того, данные могут быть избыточны, недостаточны и т.д. В некоторых задачах требуется дополнить данные некоторой априорной информацией. Наивно предполагать, что если подать данные на вход системы в

существующем виде, то на выходе получим полезные знания. Данные должны быть качественны и корректны с точки зрения используемого метода DM. Более того, иногда размерность исходного пространства может быть очень большой, и тогда желательно применять специальные алгоритмы понижения размерности. Это как отбор значимых признаков, так и отображение данных в пространство меньшей размерности.

Трансформация, нормализация данных. Трансформация подразумевает оптимизацию данных для решения определенной задачи. Обычно на этом этапе выполняется исключение незначащих факторов, снижение размерности входных данных, нормализация, обогащение и другие преобразования, позволяющие лучше «приспособить» данные к решению аналитической задачи.

Data Mining. На этом шаге применяются различные алгоритмы для нахождения знаний. Это нейронные сети, деревья решений, алгоритмы кластеризации, установление ассоциаций и т.д.

Постобработка данных. Интерпретация результатов и применение полученных знаний в бизнес-приложениях.

3. Структура БД

Исследуемая задача характеризуется следующими факторами:

– неопределенность – при этом информация, необходимая для принятия решений, по большей части носит качественный характер;

– многокритериальность, причем исследуемые критерии представлены в различных шкалах измерений (интервальная, номинальная, порядковая);

– необходимость одновременного учета как количественных, так и качественных критериев оценки альтернатив;

– необходимость согласования групповых мнений экспертов;

– многоуровневость системы частных (локальных) критериев и их неравнозначность (критерии вносят различный вклад в интегральную оценку альтернативы);

– многократность процесса выбора;

– совместимость объективных и субъективных характеристик элементов задачи.

Разработанный комплекс моделей (функциональная, процессная, информационная) позволил получить полную и наглядную информацию об исследуемой предметной области, разработать структуру базы знаний и структуру разрабатываемой системы, определить требования к программному и аппаратному обеспечению систем. При этом информационная модель является логической структурой БД, а функциональная и процессная модель – основой для проектирования структуры БЗ.

В исследованиях использовалась база данных, структура которой разрабатывалась и оптимизировалась для компании, занимающейся сервисным обслуживанием банкоматов в процессе проведения описываемых исследований. Она охватывает основные, с точки зрения экспертной оценки, блоки принимающие участие в процессе:

– описание устройств и их атрибутов – это банкоматы и их основные характеристики, имеющие значение в процессе обслуживания;

– описание клиентской базы – это клиенты компании, соглашения о сервисе, данные о поставках и инсталляциях банкоматов;

– описание инженерного состава – это инженеры компании, которые реализуют обслуживание банкоматов, их квалификация, региональное распределение;

– описание процесса обслуживания – это заявки на выполнение работ, отчеты о выполненных работах, состояние устройств.

В результате исследования БД и проведенного статистического анализа [8] были выявлены основные показатели каждого из блоков, имеющие достаточное для предмета исследование значение. К ним относятся – долгосрочность банкоматов, их работоспособность, регламент проведения работ, среда функционирования устройств, загруженность специалистов, надежность сети.

4. Подсистема извлечения знаний

Автоматизация извлечения знаний из баз данных должна учитывать следующую специфику:

1. Данные имеют практически неограниченный объём.

2. В большинстве случаев данные являются числовыми, а если они качественные или текстовые, то для их обработки все равно используются количественные методы.

3. Извлеченные знания должны быть конкретны и понятны пользователю.

4. Инструменты обнаружения знаний должны быть просты в использовании и работать при наличии «сырых» данных.

В результате проведенных исследований для извлечения знаний предлагается двухэтапный метод с использованием нейронной сети и нечеткого интерпретатора. На первом этапе «сырые» данные обрабатываются нейронной сетью, которая в неявном виде формулирует скрытые закономерности. На втором этапе полученные знания извлекает, верифицирует и формулирует в лингвистической форме нечеткий интерпретатор.

а) Существующие модели и методы извлечения знаний

Первоначально основным инструментом анализа данных были классические методы математической статистики. Но современного пользователя все реже и реже удовлетворяют результаты статистического анализа, используемого для обобщения и интерпретации искомых закономерностей в явном виде $y = f(x)$, существующих между зависимой переменной y и m-мерным вектором $x = (x, ..., x_m)$ независимых переменных. В известной степени это обусловлено тем, что в рамках статистического подхода требуется, чтобы были приняты гипотезы об однородности имеющихся данных, взаимной независимости переменных и т.д. Пользователь должен знать основные понятия и положения математической статистики для того, чтобы с их помощью интерпретировать интересующие его закономерности. Методы математической статистики оказались полезными главным образом для проверки заранее сформулированных гипотез (verification-driven data mining) и для “грубого” предварительного анализа, составляющего основу оперативной аналитической обработки данных (online analytical processing, OLAP) [8].

В основу современных методов технологии Data Mining (discovery-driven data mining) [3] положена концепция шаблонов, отражающих фрагменты многоаспектных взаимоотношений в данных. Эти шаблоны представляют собой закономерности, найденные путем кластеризации. Для кластеризации необходима предварительная «очистка» данных, которая проводится известными в статистике методами сглаживания. Далее проводится анализ выявленных признаков на информативность. Выявленные закономерности в числовом формате подлежат верификации стандартными статистическими методами. На последнем этапе скрытые знания формируются в виде деревьев решений и сохраняются в базе знаний в форме продукционных правил.

Анализ задачи показывает, что практически идеальным средством для извлечения знаний из баз числовых данных являются нейронные сети. Исследования аппроксимирующих свойств нейронных сетей показывают, что они способны выявлять скрытые закономерности автоматически, без предварительной обработки данных и без выяснения степени взаимной корреляции входных переменных [4].

С формальной точки зрения нейронная сеть представляет собой универсальную аппроксимирующую модель в виде графа. Моделируя реальный объект, такой граф способен путем обучения повышать свою адекватность этому объекту за счет модификации весов межэлементных связей. Именно наличие формальных методик обучения (при условии существования обучающей выборки в виде экспериментальных пар «входы-выходы») является главным преимуществом нейронных сетей.

Важным преимуществом нейронных сетей является также то, что разработка экспертных систем, основанных на правилах, требует 12...18 месяцев, а нейросетевых - от нескольких недель до месяцев [5].Однако в качестве средства извлечения знаний нейронные сети имеют определенные недостатки:

– весовые коэффициенты межнейронных связей обученной нейронной сети не подлежат ясной и содержательной интерпретации;

– на сегодняшний день модели нейронных сетей, использующих качественные входы, развиты недостаточно, чтобы успешно моделировать сложные объекты;

– градиентный метод, который традиционно применяется для обучения нейронной сети, не всегда дает возможность достичь глобального минимума в различении модельных и экспериментальных значений выхода.

Последний недостаток устраняется использованием эволюционных методов настройки весовых коэффициентов сети.

Для извлечения знаний могут применяться также нечеткие системы логического вывода. В работе [5] предлагается метод построения нечетких БЗ путем их извлечения из данных эксперимента. В описываемом методе используется нейронечеткая сеть, которая адаптируется к исследуемому объекту путём оптимизации значений параметров функций принадлежности входных переменных и значений весовых коэффициентов, нечетких правил.

Для поиска оптимальных значений указанных параметров используется генетический алгоритм. Однако для построения нейронечеткой экспертной системы требуется предварительно заполнить базу знаний правилами, т.е. создать прототип системы, который дообучается путем параметрической оптимизации. Если априорные формулировки правил отсутствуют, то данный метод неприменим.

В книге [6] со ссылкой на работу [7] описывается простой метод извлечения лингвистических знаний из «сырых» числовых данных при помощи нечеткой интерпретации скрытых закономерностей. Применяя формализм теории нечетких множеств, метод можно описать следующим образом:

1. Для каждой входной и выходной переменной уточняется диапазон допустимых значений:

$$X_i = \left[\underline{x}_i, \overline{x}\right] , Y\left\lfloor \underline{y}_i, \overline{y}_j \right\rfloor, \quad (1)$$

В результате проведенного статистического анализа к входным переменным мы отнесли:

– долгосрочность: срок службы, наработка, новизна;

– работоспособность: время в ремонте, время неисправности;

– регламент проведения работ: время реакции, время выполнения;

– среду функционирования: показатели напряжения, заземления;

– загруженность специалистов: количество банкоматов, количество инженеров;

к выходным:

– надежность сети: количество отказов, количество просроченных работ и количество замен запчастей.

2. Каждая переменная получает лингвистическую интерпретацию, и диапазон её изменения разбивается на нечеткие интервалы, обозначенные термами:

– терм-множество переменной X_i:

$$A_i = \{a_i^1, a_i^2, ..., a_i^{li}\}, i = \overline{1,n}; \quad (2)$$

– терм-множество переменной Y:

$$D = \{d_1, d_2, ..., d_m\}. \quad (3)$$

Здесь a_i^p – p-й лингвистический терм переменной x_i; $p = \overline{1,l}, i = \overline{1,n}$, d_j – j-й лингвистический терм переменной y; m – число различных решений в рассматриваемой области.

Мощности терм-множеств Ai, $i = \overline{1,n}$ в общем случае могут быть различны, т.е. $l_1 \neq l_2 \neq ... \neq l_m$.

На примере переменной x_i – «срок службы» определяем терм-множество A_1:

$$A_1 = \{a_1^1, a_1^2, ..., a_1^{ll}\}, i = \overline{1,n}, \quad (4)$$

где a_1^1 – бессрочный – если изготовитель не установил срок службы; a_1^2 – минимальный – до 12 месяцев; a_1^3 – средний – от 12 до 36 месяцев; a_1^4 – максимальный – от 36 месяцев.

Получаем

$$A_1 = \{\text{бессрочный, минимальный, средний, максимальный}\}. \quad (5)$$

3. Каждый нечеткий интервал снабжается функцией принадлежности. Тогда нечеткие множества a_i^p и d_i определим так:

$$a_i^p = \sum_{k=1}^{q_i} \mu^{a_i^p}(x_i^k)/x_i^k, d_i = \sum_{r=1}^{q_m} \mu^{d_j}(y^r)/y^r \ , \qquad (6)$$

где $\mu^{a_i^p}(x_i^k)$ – степень принадлежности элемента $x_i^k \in x_i$ терму $a_i^p \in A_i, p = \overline{1,l}, i = \overline{1,n}, k = \overline{1,q_i}, ;$ $\mu^{d_j}(y^r)$ – степень принадлежности элемента $y^r \in Y$ терму-решению $d_j \in D, j = \overline{1,m}$.

4. Числовые данные группируются в виде пар «вход-выход»: $(X^l, y^l), l = 1,...M$, где $X^l = (x^l{}_1, x^l{}_2, ...x^l{}_n)$ - входной вектор и соответствующее значение выходной переменной y^l для l-й пары «вход-выход», $y^l \in \left[\underline{y_i}, \overline{y_j}\right]$.

5. Для каждой пары «вход-выход» вычисляются значения функций принадлежности $\mu^{a_i^p}(x_i^k)$ вектора X^l и $\mu^{d_j}(x_1^*, x_2^*, ..., x_n^*)$ для всех нечетких интервалов, на которые разбит диапазон допустимых значений $\left[\underline{y_i}, \overline{y_j}\right]$ выходной переменной y.

6. Для каждой входной переменной x_i^k отыскивается терм a_i^p, для которого функция принадлежности $\mu^{a_i^p}(x_i^l)$ имеет максимальное значение $\mu^{a_{i\max}^p}(x_i^l)$. Для выходной переменной также отыскивается терм d_j^l, для которого функция принадлежности $\mu^{d_j}(y^l)$ имеет максимальное значение $\mu^{d_{j\max}}(y^l)$.

7. Формируется правило типа «ЕСЛИ-ТО,», которое связывает лингвистические значения $a_{i\max}^1, a_{i\max}^2, ..., a_{i\max}^{l_i}$ входных переменных с лингвистическим значением выходной переменной $d_{j\max}^l$:

$$\left[\bigcap_{i=1}^n (x_i^l = a_{i\max}^{jp})\right] \to y = d_{j\max}^l, j = \overline{1,m} \ . \qquad (7)$$

8. Для полученного правила вычисляется степень истинности:

$$R^l = \mu^{d_{j\max}}(y^l) \prod_i \mu^{a_{i\max}^p}(x_i^l) \ . \qquad (8)$$

9. Отбор правил. Если в накопленной БЗ уже существует правило с номером t с такими же значениями $a_{i\max}^1, a_{i\max}^2, ..., a_{i\max}^{l_i}$ и $d_{j\max}^l$ и выполняется условие $R^l > R^t$, то правило l фиксируется в базе знаний вместо правила t.

Таким образом, накапливается БЗ в лингвистической форме на основе нечеткого анализа числовых данных. Несомненным достоинством метода является простота реализации. Однако у данного метода есть резервы повышения эффективности и точности, которые можно реализовать, если в качестве генератора правил использовать обученную нейронную сеть и применить более совершенный алгоритм отбора и верификации правил.

б) Нейронная сеть и формирование базы знаний

В изложенном выше методе числовые данные подвергаются нечеткому анализу без предварительной подготовки, а значит, содержат шумы и потенциальные противоречия. Естественно, данные можно предварительно обрабатывать статистическими методами, но гораздо удобнее использовать их для обучения нейронной сети. В таком случае сеть автоматически накапливает знания о скрытых многомерных зависимостях в числовых данных. После обучения сети ей можно предъявлять любые входные векторы из допусти-

мого диапазона и получать адекватные выводы при помощи описанного выше метода интерпретации, вообще не накапливая лингвистических знаний. В этом случае из перечисленных выше пунктов будут использоваться пункты 1...3, а степень истинности $\mu^{d_{jmax}}(y^l)$ интерпретируется как степень истинности ответа.

На данном этапе использовалась НС следующей конфигурации:

количество узлов входного слоя – 11;

количество узлов скрытого слоя – 5;

количество узлов в выходном слое – 3;

алгоритм обучения – обратное распространение;

функция активации – сигмоидальная.

Если ставится задача составить базу знаний лингвистического характера (например, для системы нечеткого логического вывода), то встает вопрос о процедуре отбора правил. Представляется разумным не отбрасывать правила с различными, пусть и небольшими значениями степени истинности, а накопить достаточно полную совокупность экземпляров правил по всем или по значительной части пар «вход-выход» обучающей выборки нейронной сети.

Обсудим вопросы верификации и отбора правил.

Если разработчик СППР обнаруживает, что система по результатам испытаний допускает значительное количество ошибок, у него есть несколько вариантов коррекции правил.

Во-первых, можно изменить структуру нейронной сети, используя такие подходы, как МГУА [5].

Во-вторых, можно настраивать параметры функций принадлежности, используя методы, описанные в работе [5]. В-третьих, можно верифицировать и отбирать правила, полученные на выходе интерпретатора, как это предлагается в изложенном выше методе. Но замена предыдущего правила новым, с увеличенным значением степени истинности, не представляется единственным и наиболее верным подходом. На наш взгляд, следует накопить значения истинности всех найденных одинаковых правил и вычислить усредненную степень истинности по всей выборке.

Если количество найденных одинаковых правил достаточно велико, то состоятельность результирующего правила измеряется средним значением степени истинности R. Если же количество найденных одинаковых правил не превышает определённого порогового значения A, которое устанавливается по результатам частотного анализа накопленной совокупности правил, то среднюю степень истинности R для одинаковых правил вычислим как R = k/A, где k – количество одинаковых правил в выборке. Для отбора надежных правил установим пороговое значение степени истинности R_n . В базу знаний попадут правила, у которых $R > R_n$.

Если в накопленной совокупности правил найдены выборки противоречивых правил, имеющих одинаковые левые части, но различные выводы d_j, следует провести дополнительный анализ для выяснения вопроса: отдать ли предпочтение одному из вариантов или удалить оба варианта правил из базы знаний. Для решения предлагается следующая методика.

Для каждой выборки правил вычисляется вектор средних значений всех входных переменных $(x_1^{-1}, x_2^{-1}, ...x_n^{-1})$ и $(x_1^{-2}, x_2^{-2}, ...x_n^{-2})$.

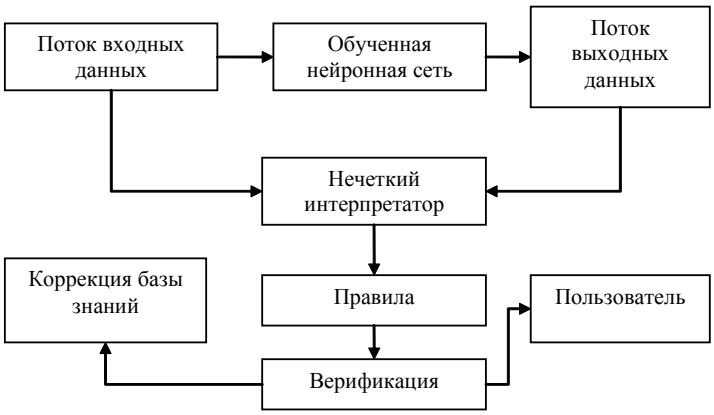
Вычисляется вектор средних значений по всем входным переменным по обеим выборкам $(\overline{x_1}, \overline{x_2}, ..., \overline{x_n})$.

Полученный в п. 2 вектор подается на вход нейронной сети и вычисляется числовое начение выхода сети.

Вход и выход сети интерпретируются в лингвистическую форму.

Если полученное правило совпадает с одним из конкурирующих, то предпочтение отдается выигравшему правилу. Если же новое правило не совпадает ни с одним из конкурирующих, то оба конкурирующих правила удаляются из БЗ. На рисунке показана функциональ-

ная структура разработанного инструментального программного средства, реализующего предлагаемый метод извлечения и интерпретации знаний, накопленных нейронной сетью.



Функциональная структура инструментального программного средства

5. Заключение

Предложен модифицированный метод извлечения скрытых знаний из базы данных с использованием обученной нейронной сети и нечеткого интерпретатора входных и выходных данных. Метод предназначен для быстрой разработки интеллектуальных СППР на основе нейросетевого подхода и предусматривает применение специальной процедуры отбора и верификации правил. Предложена функциональная структура инструментального программного средства для извлечения знаний, позволяющая проводить эксперименты с изложенной методикой извлечения знаний.

Список литературы: 1. *Рыбина Г.В.* Автоматизированное построение баз знаний для интегрированных экспертных систем // Изв. РАН. Теория и системы управления. 1998. №5. С.152-166. **2.** *Калинина Е.А., Рыбина Г.В.* Применение технологии Data Mining для автоматизированного построения баз знаний интегрированных экспертных систем //В кн.:КИИ-2000. Седьмая нац. конференция с межд. участием. Труды конференции. Том 2. М.:Физматлит, 2000. С. 119-127. **3.** *Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И.* Методы и модели анализа данных: OLAP и Data Mining / СПб.: БХВ-Петербург, 2004. 326 с. **4.** *Сетлак Г.* Интеллектуальные системы поддержки принятия решений. К.: Логос, 2004. 251 с. **5.** *Митюшкин Ю.И., Мокин Б.И., Ротштейн А.П.* Soft Computing: Идентификация закономерностей нечеткими базами знаний . Винница: УНИВЕРСУМ-Винница, 2002. 145 с. **6.** *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы. М.: Горячая линия, 2006. 193 с. **7.** *Wang. L.X., Mendel J. M.*, Generating Fuzzy Rules by Learning from Examples, IEEE Transactions on Systems, Man, and Cybernetics, November/December 1992. Vol. 22, № 6. P. 1414-1427. **8.** *Куземин А.Я., Головий Н.В., Даюб Я.* Реализация модели системы поддержки принятия решений в области сервисного обслуживания банкоматов // АСУ и приборы автоматики. 2009. №143. С.57 – 64.

Поступила в редколлегию 15.08.2009

Головий Наталья Владимировна, аспирантка кафедры информатики ХНУРЭ. Научные интересы: системный анализ данных. Адрес: Украина, 61166, Харьков, пр.Ленина,14, тел. (057) 702 15 15, e-mail: rica1982@mail.ru.

УДК 004.421: 548.55

А.П. ОКСАНИЧ, В.Р. ПЕТРЕНКО, С.А. ВОЛОХОВ

СИНТЕЗ ПРОГНОЗНОГО РЕГУЛЯТОРА ДЛЯ ПРОЦЕССА ВЫРАЩИВАНИЯ ОБЪЕМНЫХ МОНОКРИСТАЛЛОВ КРЕМНИЯ ДЛЯ СОЛНЕЧНЫХ ФЭП

На основе использования подхода Бокса-Дженкинса к синтезу моделей стохастических линейных динамических процессов разрабатывается АРМАХ-модель процесса вытягивания монокристаллического слитка, связывающая вариации скорости вытягивания с вариациями диаметра слитка. Полученная модель используется для синтеза оптимального прогнозного регулятора процесса выращивания на этапе вытягивания цилиндрической части слитка. Результаты моделирования работы регулятора подтвердили его работоспособность.

1. Введение

Высокочистый монокристаллический кремний является основным полупроводниковым материалом, используемым в современной электронной промышленности.

Дополнительный интерес к монокристаллическому кремнию появился в связи с решением проблем альтернативных источников энергии. Научные разработки возобновляемых источников энергии в последние десятилетия достигли практической реализации и стали активно внедряться в мировую экономику. Среди таких разработок особое место принадлежит солнечной энергетике, базирующейся на использовании полупроводниковых фотоэлектрических преобразователей (ФЭП).

По прогнозам аналитиков предусматривается, что общая мощность кремниевых фотоэлектрических станций в странах ЕС, в США и в Японии достигнет в 2010 году 9-11 ГВт (из них 7,5-9,2 ГВт будет получено с использованием ФЕП на основе кремния, причем около 40% придется на слитки монокристаллов), а до 2020 года увеличится еще в 20-30 раз, достигнув 15-17% от общих объемов энергопотребления в этих странах [1,2].

Таким образом, сегодня актуальной проблемой является создание новой, стратегически важной для Украины, отрасли производства – фотоэнергетического приборостроения, способной обеспечить потребности страны в альтернативных воспроизводимых источниках энергии и реализовать полный технологический цикл от производства кремния через производство солнечных элементов к конечному продукту солнечной энергетики – фотоэлектрическим модулям, батареям и автономным солнечным электростанциям различного назначения.

Первым этапом при реализации упомянутого технологического цикла в Украине является производство «солнечного» кремния диаметром 200мм. В ООО «Силикон» для этой цели была разработана и внедрена модернизированная установка «РЕДМЕТ-60А», которая базируется на наиболее распространённой технологии получения монокристаллических слитков кремния больших диаметров, реализующей метод Чохральского [3] . Ее эксплуатация показала, что для достижения более высоких значений показателей эффективности процесса выращивания монокристаллических слитков необходимо усовершенствование системы управления.

В работах [4,5] отмечено, что для повышения эффективности контроля и управления процессами промышленного производства Cz-Si-монокристаллов целесообразно использовать стохастический подход к описанию объекта управления (ОУ), который позволяет учитывать естественную неопределенность исходных данных и повышать степень адекватности моделей, используемых при синтезе системы управления. В связи с этим появилась необходимость в разработке более эффективных регуляторов (по сравнению с используемыми ПИД-регуляторами), в частности регуляторов, базирующихся на концепции прогнозного управления. Синтез такого регулятора для процесса выращивания объемных Cz-Si монокристаллов на установке «РЕДМЕТ-60А» и является *целью данной работы*.

Метод прогнозного управления был разработан в конце семидесятых годов прошлого века [6]. На сегодня известно значительное количество модификаций этого метода: EPSAC, GPC, MUSMAR, MAC, PFC, QDMC, SOLO [7-13]. В этих алгоритмах используется одна и та же концепция прогнозного управления: наличие внутренней модели, метод отступающего горизонта и вычисление последовательности прогнозных оптимальных сигналов управления. Алгоритмы отличаются применением различных моделей реальной системы, возмущающих воздействий и критериев оптимальности управления.

2. Постановка задачи

В общем виде алгоритм прогнозного управления может быть представлен последовательностью следующих шагов :

1. Предсказание в момент времени t (на основе модели объекта управления) значений выходной переменной системы $\hat{y}_t(t+k)$, где k=1,...,N_1. (При этом выход зависит от будущих управляющих воздействий $\hat{x}_t(t+k), k = 0,1,..., N_2$).

2. Выбор целевой функции управления и оптимизация с ее помощью $\hat{x}_t(t+k), k = 0,1,..., N_2$.

3. Реализация управления $x_t = \hat{x}_t(t)$.

4. Переход в момент времени (t+1) к шагу 1 и повторение шагов 1–4 до достижения цели управления.

Критерий оптимизации, используемый на шаге 2, может быть выбран из достаточно широкого класса критериев. Мы будем применять критерий вида:

$$J(t) = \sum_{k=1}^{N_1} [\hat{y}_t(t+k) - r_t(t+k)]^2 q_1(k) + \sum_{k=0}^{N_2-1} [\hat{x}_t(t+k)]^2 q_2(k).$$

Значения штрафов на управление и его ошибку можно изменять с помощью весовых коэффициентов $q_2(k)$ и $q_1(k)$. Предполагается, что всегда $N_2 \le N_1$ и что $\Delta\hat{x}_t(t+k) = 0$ для $k \ge N_2$.

Для обеспечения возможности реализации принципа прогнозного управления возникает необходимость в разработке адекватной модели объекта управления.

3. Разработка математической модели

Для построения математической модели предлагается использовать класс стохастических разностных моделей с дискретным временем [14]. Предположим, что модель передаточной функции

$$Y_t = \nu(B)X_t + N_t \qquad (1)$$

может быть экономично параметризована в виде

$$Y_t = \delta^{-1}(B)\omega(B)X_{t-b} + N_t, \qquad (2)$$

где b – чистое запаздывание; B – оператор сдвига назад, т.е. $BY_t = Y_{t-1}$;

$$\delta(B) = 1 - \delta_1 B - \delta_2 B^2 - ... - \delta_r B^r; \quad \omega(B) = \omega_0 - \omega_1 B - \omega_2 B^2 - ... - \omega_s B^s;$$

Y_i, $i = t, t-1, ..., t-r$ и X_j, $j = t-b, t-b-1, ..., t-b-s$ – соответственно отклонения выхода и входа от равновесных состояний; N_t – шум, генерируемый некоторым процессом авторегрессии-проинтегрированного скользящего среднего (АРПСС) и статистически независимый от X_t.

Таким образом, для построения модели необходимо по доступным наблюдениям $(X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N)$ определить оценки параметров r, s, b и начальные оценки параметров δ_i, $i = \overline{s,r}$ и ω_j, $j = \overline{0,s}$, а также идентифицировать и оценить модель шума.

Заметим, что путем взятия конечных разностей над процессами X_t, Y_t модель (1) можно привести к виду

$$y_t = v_0 x_t + v_1 x_{t-1} + v_2 x_{t-2} + ... + n_t, \qquad (3)$$

где $y_t = \nabla^d Y_t, x_t = \nabla^d X_t, n_t = \nabla^d N_t$ – стационарные процессы с нулевыми средними значениями, d-порядок разности.

Процедура построения модели передаточной функции в соответствии с подходом Бокса-Дженкинса сводится к выполнению следующих основных этапов:

– получение грубых оценок $\hat{\nu}_j$ импульсного отклика в (3) с помощью алгоритма, основанного на выравнивании спектра входа;

– определение оценок $\hat{r},\hat{s},\hat{b}$ параметров r, s, b на основе анализа поведения последовательности $\hat{\nu}_j$;

– вычисление начальных оценок $\hat{\delta}_i, i=\overline{1,r}$ и $\hat{\omega}_j, j=\overline{0,s}$ на основании оценок $\hat{\nu},\hat{r},\hat{s},\hat{b}$;

– определение структуры и начальных оценок параметров модели шума;

– уточнение оценок параметров комбинированной модели;

– диагностическая проверка разработанной модели.

Для определения оценок $\hat{\nu}_j$ параметров ν_j модели (3) использовалось соотношение

$$\hat{\nu}_j = \frac{r_{\alpha\beta}(j)s_\beta}{s_\alpha}, \quad j=0,1,2,\ldots, \tag{4}$$

где $r_{\alpha\beta}(j)$ – выборочная взаимная корреляционная функция процессов α_t и β_t, а s_β и s_α - выборочные стандартные отклонения для этих процессов.

Процесс α_t определяется путем подгонки АРСС-модели к процессу x_t, т е.

$$\varphi_x(B)\theta^{-1}{}_x(B)x_t = \alpha_t, \tag{5}$$

а процесс β_t – результат применения преобразования $\varphi_x(B)\theta^{-1}{}_x(B)$ к процессу y_t, т.е.

$$\beta_t = \varphi_x(B)\theta^{-1}{}_x(B)y_t. \tag{6}$$

Модель (3) при этом может быть представлена в виде

$$\beta_t = \nu(B)\alpha_t + \xi_t, \tag{7}$$

где $\xi_t = \varphi_x(B)\theta^{-1}{}_x(B)n_t$.

При известных $\hat{\nu}_j$ значения параметров r, s, b модели (2) можно оценить, используя следующие известные факты [1-8]: для модели вида (2) веса ν_j импульсного отклика состоят из b нулевых значений $\nu_0, \nu_1, \ldots, \nu_{b-1}$, последующих s-r+1 значений $\nu_b, \nu_{b+1}, \ldots, \nu_{b+s-r}$ с произвольным поведением (таких значений нет, если s<r) и значений ν_j при $j \geq b+s-r+1$, поведение которых определяется разностным уравнением r-го порядка с r начальными значениями $\nu_{b+s}, \nu_{b+s-1}, \ldots, \nu_{b+s-r+1}$.

Определение начальных оценок параметров $\delta_i, i=\overline{1,r}$ и $\omega_j, j=\overline{0,s}$ осуществляется путем использования следующей системы уравнений:

$$\begin{gathered}
\nu_j = 0, \quad j < b; \\
\nu_j = \delta_1\nu_{j-1} + \delta_2\nu_{j-2} + \ldots + \delta_r\nu_{j-r} + \omega_0, \quad j = b; \\
\nu_j = \delta_1\nu_{j-1} + \delta_2\nu_{j-2} + \ldots + \delta_r\nu_{j-r} + \omega_{j-b}, \quad j = b+1, b+2, \ldots, b+s; \\
\nu_j = \delta_1\nu_{j-1} + \delta_2\nu_{j-2} + \ldots + \delta_r\nu_{j-r}, \quad j > b+s.
\end{gathered} \tag{8}$$

При известных оценках параметров передаточной функции можно найти оценки n_t с помощью соотношения

$$\hat{n}_t = y_t - \hat{\delta}^{-1}(B)\hat{\omega}(B)x_{t-b}. \tag{9}$$

Далее с помощью известных методов идентификации АРСС-процессов определяется структура модели для n_t и начальные оценки ее параметров.

На этапе оценивания модели решается задача одновременного эффективного оценивания параметров $b, \overline{\delta}, \overline{\varpi}, \overline{\varphi}, \overline{\theta}$ ранее идентифицированной модели вида

$$y_t = \delta^{-1}(B)\omega(B)x_{t-b} + \varphi^{-1}(B)\theta(B)a_t. \tag{10}$$

Эта задача решается путем минимизации условной суммы квадратов

$$S_0(b,\bar{\delta},\bar{\omega},\bar{\varphi},\bar{\theta}) = \sum_{t=u+p+1}^{n} a_t^2(b,\bar{\delta},\bar{\omega},\bar{\varphi},\bar{\theta} \mid x_0, y_0, a_0), \qquad (11)$$

где u – большее из r и s+b. Для поиска оценок параметров, минимизирующих (11), использовался хорошо известный метод Марквардта [15]. При этом ковариационная матрица оценок V определяется формулой

$$V = (X'X)^{-1}\hat{\sigma}_a^2, \qquad (12)$$

здесь X – регрессионная матрица в линеаризованной модели, вычисленная на последней итерации в процедуре Марквардта, а $\hat{\sigma}_a^2$ – оценка остаточной дисперсии.

Диагностическая проверка соответствия комбинированной модели анализируемым данным основана на исследовании поведения остаточных ошибок $\hat{a}_t$, которые можно определить с помощью следующего соотношения:

$$\hat{a}_t = \hat{\theta}^{-1}(B)\hat{\varphi}(B)\left(y_t - \hat{\delta}^{-1}(B)\hat{\omega}(B)x_{t-b}\right), \qquad (13)$$

где $t \geq u+1$, $u = \max\{r, s+b\}$ и все $\hat{a}_j$, $j = \overline{1,u+p}$ равны нулю.

При этом вычисляется статистика

$$Q = m\sum_{k=1}^{K} r_{\hat{a}\hat{a}}^2(k), \qquad (14)$$

m – количество используемых в расчетах значений $\hat{a}_t$ (обычно $m = n-u-p$, n – объем выборки); K – задержка, для которой справедливо, что при $k > K$ автокорреляции пренебрежительно малы.

В [14] указано, что величина Q распределена примерно как χ^2 с $K-p-q$ степенями свободы. Если Q меньше табличного значения для заданного уровня значимости, то принимается гипотеза об адекватности разработанной комбинированной модели.

Ниже приведены результаты основных этапов синтеза модели рассматриваемого класса, связывающей отклонения диаметра растущего кристалла с отклонениями скорости вытягивания. В качестве исходных данных для решения задачи синтеза модели использовались два временных ряда, образованных наблюдаемыми значениями диаметра кристалла (Y_t) и скорости вытягивания (X_t). Данные снимались на этапе выращивания цилиндрической части слитка диаметром 200 мм с интервалом 1 мин. на установке „РЕДМЕТ-60А”. Фрагменты рядов приведены на рис.1.

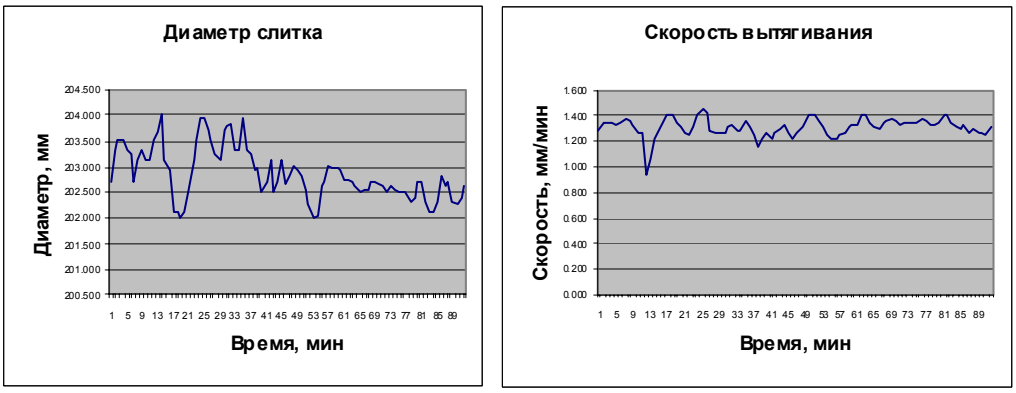


Рис. 1. Фрагменты исследуемых рядов

Выборочная автокорреляционная и частная автокорреляционная функции входа x_t представлены на рис. 2 и 3 соответственно.

Рис. 2. Автокорреляционная функция

Рис. 3. Частная автокорреляционная функция

Анализ автокорреляционной и частной автокорреляционной функций позволяет предположить, что вход системы может быть представлен моделью авторегрессии первого порядка. Для оценивания параметров модели входа x_t использовался метод, основанный на применении уравнений Юла – Уоккера для оценивания параметров оператора авторегрессии. Результаты приведены на рис. 4.

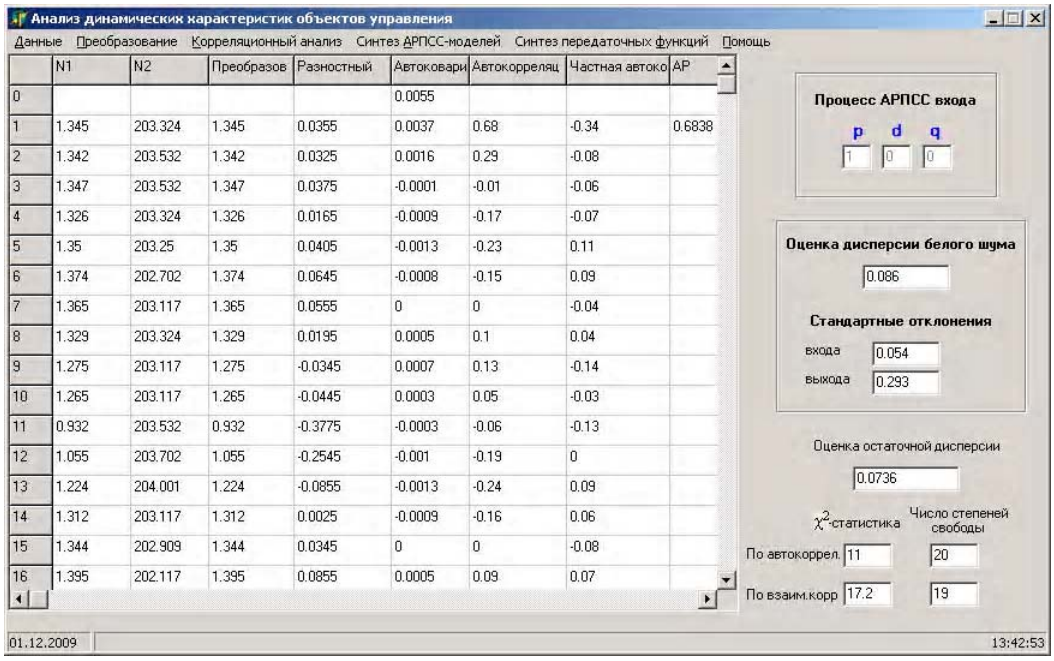


Рис.4. Исходные данные и результаты оценивания модели входа

Была получена следующая оценка параметра φ_1: 0,684. Таким образом, в нашем конкретном случае модель (5) приняла вид

$$(1-0.684B)x_t = \alpha_t, \tag{15}$$

с $s_\alpha = 0.29$, а модель (6) и ξ_t в модели (7) – соответственно

$$\beta_t = (1-0.63B)y_t, \tag{16}$$

$$\xi_t = (1-0.684B)n_t. \tag{17}$$

Полученные оценки отклика на единичный импульс приведены на рис.5. Анализируя поведение функции отклика на единичный импульс и следуя рекомендациям [14], получаем следующие оценки структурных параметров модели передаточной функции: $r = 2$, $s = 1$, $b = 1$.

Таким образом, модель передаточной функции принимает вид

$$(1-\delta_1 B-\delta_2 B^2)y_t = (\omega_0-\omega_1 B)x_{t-1}+n_t. \tag{18}$$

Начальные оценки «левосторонних» параметров ($\hat{\delta}_i$) можно получить путем решения системы уравнений

$$A\hat{\delta} = h, \tag{19}$$

где $a_{ij} = \begin{cases} \hat{\upsilon}_{b+s+i-j}, & s+i \ge j \\ \emptyset, & s+i < j \end{cases}$; $h_i = \hat{\upsilon}_{b+s+i}$; $i, j = 1,2,...r$.

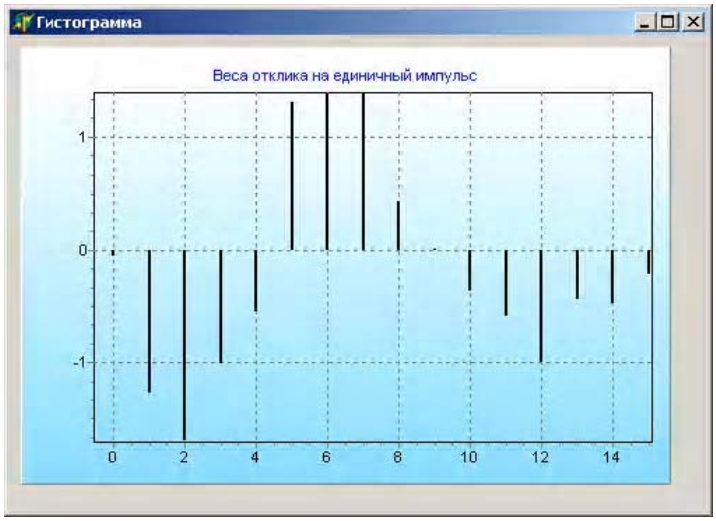


Рис. 5. Оценки отклика на единичный импульс

Для получения начальных оценок «правосторонних» параметров ($\hat{\omega}_j$) использовались следующие соотношения:

$$\hat{\omega}_0 = \hat{\upsilon}_b; \tag{20}$$

если $r \ge s$, то $\hat{\omega}_j = \sum_{i=1}^{j} \hat{\delta}_i \hat{\upsilon}_{b+j-i} - \hat{\upsilon}_{b+j}$;

если $r < s$, то $\hat{\omega}_j = \sum_{i=1}^{j} \hat{\delta}_i \hat{\upsilon}_{b+j-i} - \hat{\upsilon}_{b+j}$ для $j = 1,2,...r$;

$\hat{\omega}_j = \sum_{i=1}^{r} \hat{\delta}_i \hat{\upsilon}_{b+j-i} - \hat{\upsilon}_{b+j}$ для $j = r+1,...s$.

Используя (19), (20), а также полученные ранее оценки структурных параметров модели и оценки отклика на единичный импульс, получаем следующие начальные оценки параметров модели передаточной функции:

$$\hat{\delta}_1 = 0.63;\ \hat{\delta}_2 = -0.06;\ \hat{\omega}_0 = -1.28;\ \hat{\omega}_1 = 0.9\,.$$

Построение модели шума основывается на восстановлении последовательности n_t путем использования (3) и полученных оценок отклика на единичный импульс, т.е.

$$\hat{n}_t = y_t - \hat{\upsilon}_0 \cdot x_t - \hat{\upsilon}_1 \cdot x_{t-1} - \ldots - \hat{\upsilon}_k \cdot x_{t-k}\,,$$

где значение k должно удовлетворять условию $\upsilon_{k+i} = 0,\quad i = 1,2,\ldots$ В нашем случае, как видно из рис. 5, можно взять $k = 14$. К полученной последовательности $\hat{n}_t$ осуществляется подгонка АРСС-модели тем же самым способом, с помощью которого мы строили АРСС-модель для последовательности x_t.

Оценки автокорреляций и частных автокорреляций шума представлены соответственно на рис. 6 и 7. Их анализ позволил определить структуру модели шума в виде (2, 0, 0). После предварительного оценивания модели шума она приняла следующий вид:

$$(1 - 1.11B + 0.39B^2)n_t = a_t\,, \qquad (21)$$

$$s_a^2 = 0.078\,.$$

Уточнение оценок параметров комбинированной модели было выполнено по хорошо известному методу Марквардта [15], позволяющему получить эффективные оценки параметров b, $\overline{\delta}$, $\overline{\omega}$, $\overline{\varphi}$, $\overline{\theta}$ комбинированной модели путем минимизации условной суммы квадратов

$$S_0\left(b, \overline{\delta}, \overline{\omega}, \overline{\varphi}, \overline{\theta}\right) = \sum_{u+p+1}^{n} a^2{}_t\left(b, \overline{\delta}, \overline{\omega}, \overline{\varphi}, \overline{\theta} \mid x_0, y_0, a_0\right).$$

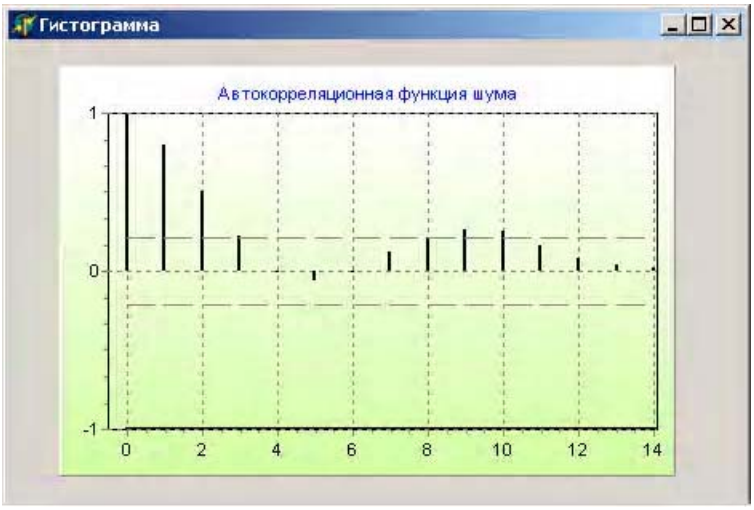


Рис. 6. Выборочная автокорреляционная функция шума

Рис. 7. Выборочная частная автокорреляционная функция шума

После этого комбинированная модель приняла следующий вид:.

$$(1-1.3B+0.76B^2)y_t=(-0.85+0.21B)x_{t-1}+\frac{1}{1-0.76B-0.08B^2}a_t$$

или после несложных преобразований

$$(1-2.06B+1.67B^2-0.48B^3-0.06B^4)y_t=-(0.85-0.86B+0.09B^2+0.02B^3)x_{t-1}+a_t; \quad (22)$$

$$s_a^2=0.073.$$

В целях диагностической проверки модели (22) с использованием (14) было определено значение статистики Q. Оно оказалось равным 11 при 20 степенях свободы. Критическое значение этой статистики при указанном числе степеней свободы и уровне значимости 0.05 равно 31.41. Таким образом, можно считать справедливой гипотезу об адекватности разработанной модели.

4. Синтез прогнозного регулятора

Сначала опишем процедуру синтеза регулятора в общем виде для класса моделей

$$Ay(t)=Bx(t-1)+a(t), \quad (23)$$

где $A=1+a_1S+a_2S^2+a_3S^3+\cdots+a_{n_a}S^{n_a}$; $B=b_0+b_1S+b_2S^2+b_3S^3+\cdots+b_{n_b}S^{n_b}$, $S-$оператор сдвига назад, т.е. $Sy(t)=y(t-1)$, а затем применим полученные результаты к модели (22)(легко заметить, что она принадлежит классу моделей (23)). Будем также считать, что горизонты прогнозирования и управления равны между собой ($N_1=N_2$) .

Представим (23) в виде

$$y(t)=\frac{B}{A}x(t-1)+\frac{1}{A}a(t). \quad (24)$$

Тогда уравнение прогноза с упреждением k будет иметь вид

$$y(t+k)=\frac{B}{A}x(t+k-1)+\frac{1}{A}a(t+k). \quad (25)$$

Используя диофантово тождество [12], имеющее для модели (24) вид

$$E_kA=1-S^kF_k, \quad (26)$$

где $E_k=e_0+e_1S+e_2S^2+\cdots+e_{n_e}S^{n_e}$; $F_k=f_0^k+f_1^kS+f_2^kS^2+\cdots+f_{n_f}^kS^{n_f}$;

$k\geq 1; n_e=k-1; n_f=n_a-1$; n_a- порядок полинома A, перепишем уравнение (25) в виде

$$y(t+k)=BE_kx(t+k-1)+F_ky(t)+E_ka(t+k). \quad (27)$$

При этом прогнозное значение выхода с упреждением k можно получить с помощью соотношения

$$\hat{y}(t+k/t)=BE_kx(t+k-1)+F_ky(t). \quad (28)$$

Реальный выход системы может быть записан в виде

$$y(t+k)=\hat{y}(t+k/t)+E_ka(t+k). \quad (29)$$

Полагая, что нужно получить прогнозы для некоторого диапазона значений k (от k=1 до k=N), запишем уравнение (27) в векторной форме, используя подход, предложенный в [16]:

$$Y=GX+f+a, \quad (30)$$

где

$$Y^T=[y(t+1),y(t+2),\cdots,y(t+N)];$$

$$X^T=[x(t),x(t+1),\cdots,x(t+N-1)];$$

$$f^T=[F_1y(t)+d_1,F_2y(t)+d_2,\cdots,F_Ny(t)+d_N];$$

$$a^T=[E_1a(t+1),E_2a(t+2),\cdots,E_Na(t+N)];$$

$$G = \begin{vmatrix} g_0 & 0 & 0 & \cdots & 0 \\ g_1 & g_0 & 0 & \cdots & 0 \\ g_2 & g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ g_{N-1} & g_{N-2} & g_{N-3} & \cdots & g_0 \end{vmatrix};$$

$g_i = h_i, i = \overline{0, N-1}; H = BE_N = h_0 + h_1 S^1 + h_2 S^2 + \cdots + h_{N-1} S^{(N-1)} + \cdots.$

$d_k = (BE_k - (h_0 + h_1 S^1 + h_2 S^2 + \cdots + h_{k-1} S^{(k-1)}))x(t+k-1), k = \overline{1, N}.$

Используя функцию цели и векторную форму модели системы (30), запишем критерий оптимальности управления в виде

$$E\{J(t)\} = E\{(GX + f + a - R)^T Q_1 (GX + f + a - R) + (X^T Q_2 X)\} > \min, \qquad (31)$$

где $R^T = [r(t+1), r(t+2), \cdots, r(t+N)]$ – заданное движение системы; $E\{x\}$ – математическое ожидание x; Q_1 и Q_2 – диагональные матрицы размерности NxN с элементами на главных диагоналях соответственно $Q_1(1), Q_1(2), \cdots, Q_1(N)$ и $Q_2(1), Q_2(2), \cdots, Q_2(N)$.

Дифференцируя (31) по X и приравнивая производную нулю, находим оптимальный вектор X:

$$X = (G^T Q_1 G + Q_2)^{-1} G^T Q_1 (R - f). \qquad (32)$$

Выражение (32) определяет оптимальный прогнозный регулятор.

Приведем некоторые ранее полученные нами [17] полезные формулы, упрощающие расчеты, связанные с синтезом оптимального регулятора.

Коэффициенты полинома E_k (его порядок равен k-1)можно получить с помощью следующего соотношения:

$$e_k = -\sum_{i=1}^{n_a} e_{k-i} a_i, \; k = \overline{1, N-1}, e_0 = 1, e_j = 0 \; \forall j < 0, \qquad (33)$$

где n_a – порядок полинома A; а a_i – его коэффициенты.

Коэффициенты полинома F_k, порядок которого $n_f = n_a - 1$, можно вычислить по формуле

$$f_j^k = -\sum_{i=1}^{n_a} e_{k-i} a_{j+i}, \; j = \overline{0, n_a - 1}, k = \overline{1, N}, a_n = 0 \forall n > n_a. \qquad (34)$$

Параметры матрицы G определяются с помощью следующего соотношения:

$$g_{k-1} = h_{k-1} = b_0 e_{k-1} + b_1 e_{k-2} + \cdots + b_{n_b} e_{k-n_b-1}, k = \overline{1, N}, e_j = 0 \forall j < 0. \qquad (35)$$

Для вычисления значений величин d_k целесообразно использовать следующее соотношение:

$$\begin{aligned} d_k = {} & (b_1 e_{k-1} + b_2 e_{k-2} + \cdots + b_{n_b} e_{k-n_b}) x(t-1) + \\ & + (b_2 e_{k-1} + b_3 e_{k-2} + \cdots + b_{n_b} e_{k-n_b+1}) x(t-2) + \\ & \cdots \\ & + b_{n_b} e_{k-1} x(t-n_b), \; k = \overline{1, N}, \; e_j = 0 \forall j < 0. \end{aligned} \qquad (36)$$

Рассмотрим синтез прогнозного регулятора для объекта, описываемого моделью (22), при N=3. В этом случае имеем:

$$n_a = 4; a_0 = 1; a_1 = -2.06; a_2 = 1.67; a_3 = -0.48; a_4 = -0.06; \quad n_b = 3;$$

$$b_0 = -0.85; b_1 = 0.86; b_2 = -0.09; b_3 = -0.02; n_f = 3; n_e = 2.$$

С помощью (33) находим $e_0 = 1; e_1 = 2.06; e_2 = 2.574$.

Используя (34), находим f_j^k:

$$f_0^1 = 2.06;\ f_1^1 = -1.67;\ f_2^1 = 0.48;\ f_3^1 = 0.06;$$
$$f_0^2 = 2.574;\ f_1^2 = -2.96;\ f_2^2 = 1.049;\ f_3^2 = 0.124;$$
$$f_0^3 = 2.341;\ f_1^3 = -3.249;\ f_2^3 = 1.359;\ f_3^3 = 0.154.$$

Далее находим элементы матрицы G, используя (35):

$$g_0 = -0.85;\ g_1 = -0.891;\ g_2 = -0.506.$$

С помощью (36) вычисляем величины d_k:

$$d_1 = 0.86x(t-1) - 0.09x(t-2) - 0.02x(t-3);$$
$$d_2 = 1.682x(t-1) - 0.205x(t-2) - 0.041x(t-3);$$
$$d_3 = 2.008x(t-1) - 0.273x(t-2) - 0.052x(t-3).$$

Формируем компоненты вектора f:

$$f(t+1) = f_0^1 y(t) + f_1^1 y(t-1) + f_2^1 y(t-2) + f_3^1 y(t-3) + d_1 =$$
$$2.060y(t) - 1.67y(t-1) + 0.48y(t-2) + 0.06y(t-3) + 0.86x(t-1) - 0.09x(t-2) - 0.02x(t-3);$$
$$f(t+2) = f_0^2 y(t) + f_1^2 y(t-1) + f_2^2 y(t-2) + f_3^2 y(t-3) + d_2 =$$
$$2.574y(t) - 2.96y(t-1) + 1.049y(t-2) + 0.124y(t-3) + 1.682x(t-1) - 0.205x(t-2) - 0.041x(t-3);$$
$$f(t+3) = f_0^3 y(t) + f_1^3 y(t-1) + f_2^3 y(t-2) + f_3^3 y(t-3) + d_3 =$$
$$2.341y(t) - 3.249y(t-1) + 1.359y(t-2) + 0.154y(t-3) + 2.008x(t-1) - 0.273x(t-2) - 0.052x(t-3).$$

В качестве матриц Q_1 и Q_2 возмем диагональные матрицы, все диагональные элементы которых равны q1=0.5 и q2=0.1 соответственно .

Подставив все полученные величины в (32) и выполнив необходимые матричные операции, получим следующее представление оптимального регулятора в векторной форме:

$$\begin{bmatrix} x(t) \\ x(t+1) \\ x(t+2) \end{bmatrix} = \begin{bmatrix} -0.77 & -0.206 & 0.037 \\ 0.602 & -0.616 & -0.206 \\ -0.135 & 0.602 & -0.77 \end{bmatrix} \begin{bmatrix} r(t+1) - f(t+1) \\ r(t+2) - f(t+2) \\ r(t+3) - f(t+3) \end{bmatrix}.$$

Поскольку на практике, как правило, реализуется принцип отступающего горизонта, то особое значение имеет определение управления x(t). После его реализации и получения отклика системы горизонт сдвигается на один такт и управление пересчитывается.

Для x(t) мы имеем

$$\begin{aligned} x(t) &= -0.77(r(t+1) - f(t+1)) - 0.206(r(t+2) - f(t+2)) + 0.037(r(t+3) - f(t+3)) = \\ &= -0.77(r(t+1) - (2.060y(t) - 1.67y(t-1) + 0.48y(t-2) + 0.06y(t-3) + \\ &+ 0.86x(t-1) - 0.09x(t-2) - 0.02x(t-3))) - \\ &- 0.026(r(t+2) - (2.574y(t) - 2.96y(t-1) + 1.049y(t-2) + 0.124y(t-3) + \\ &+ 1.682x(t-1) - 0.205x(t-2) - 0.041x(t-3))) + \\ &+ 0.037(r(t+3) - (2.341y(t) - 3.249y(t-1) + 1.359y(t-2) + 0.154y(t-3) + \\ &+ 2.008x(t-1) - 0.273x(t-2) - 0.052x(t-3))). \end{aligned}$$

Результаты экспериментального исследования работоспособности разработанного прогнозного регулятора представлены на рис.8 и 9.

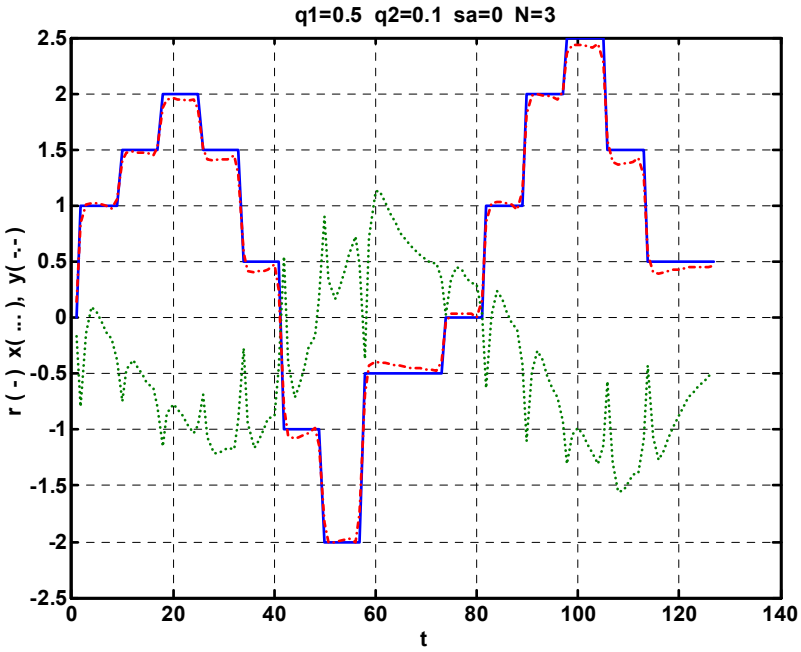


Рис.8. Работа регулятора при отсутствии шума на выходе

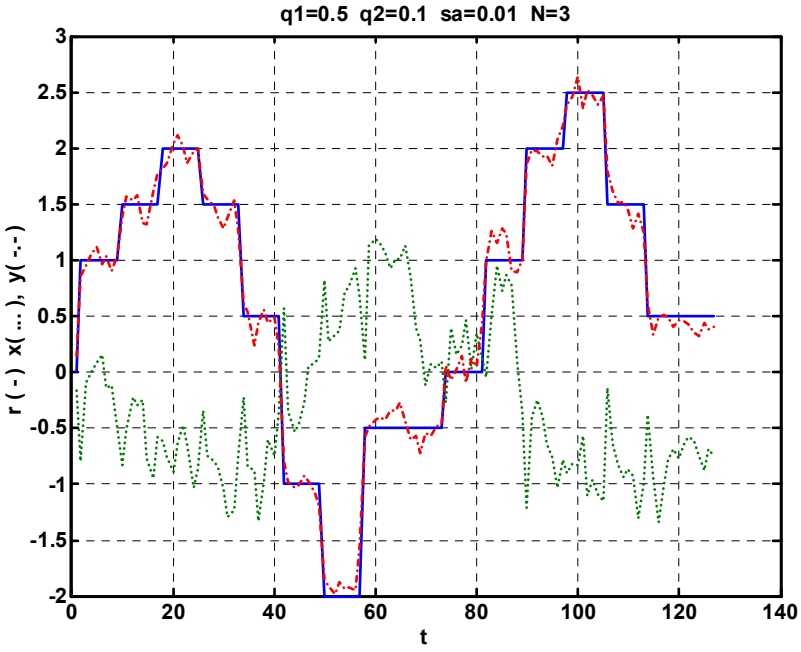


Рис. 9. Работа регулятора при наличии «белого» шума (s_a^2 =0.01) на выходе

Выводы

1. Синтезирована адекватная АРМАХ-модель процесса вытягивания монокристаллических слитков в условиях промышленной установки «РЕДМЕТ-60А», которая может быть использована для регулирования диаметра слитка по каналу скорости вытягивания.

2. Усовершенствована методика синтеза оптимального прогнозного регулятора на основе применения соотношений, позволяющих легко алгоритмизировать и автоматизировать процедуру синтеза оптимальных прогнозных регуляторов для установок выращивания слитков «солнечного» кремния.

3. На основе использования разработанной ARMAX-модели процесса выращивания выполнен синтез оптимального прогнозного регулятора для установки «РЕДМЕТ-60А», позволяющего учитывать прогнозируемые состояния регулируемого процесса с заданным упреждением, и получены результаты, подтверждающие его необходимое влияние на диаметр выращиваемого слитка кремния.

4. Дальнейшие работы следует посвятить экспериментальному исследованию синтезированного регулятора.

Список литературы: 1. *Hirshman W.P., Schmela M.* Silicon Shortage – so what! Photon International, 3, 2006. P. 100-125. **2.** Proc. of the 3rd Solar Silicon Conference, April 3, 2006; Munich, Germany. **3.** *Шашков Ю.М.* Выращивание монокристаллов методом вытягивания. М.: Металлургия, 1982. 312 с. **4.** *Разработка* стохастических моделей передаточных функций для системы управления процессом выращивания монокристаллов кремния большого диаметра / А.П. Оксанич, В.Р. Петренко // Вестник Херсонского государственного технического университета. 2002. № 2(15). С. 360-363. **5.** *Оценивание* адекватности стохастических моделей передаточных функций системы управления процессом выращивания монокристаллов кремния / А.П. Оксанич, В.Р. Петренко // Нові технології. Науковий вісник Інституту економіки та нових технологій. 2004. № 3(6). С. 12-14. **6.** *Model* predictive heuristic control / Richalet J., Rault A., Testud L.J., Papon J. // Automatica. 1978. Vol. 14. P. 413-428. **7.** Generalized predictive control. Part 1 and 2. / Clarke D.W., Mohtadi C., Tuffs P.S. // Automatica. 1987. Vol. 23. P. 137-160. **8.** *Peterka V*. Predictor-Based Self-Tuning Control / V. Peterka // Automatica. 1984. Vol. 20, № 1. P. 39-50. **9.** *Properties* of Generalized Predictive Control / D.W. Clarke, C. Mohtadi // Automatica. 1989. № 25. P. 859-875. **10.** *Analysis* and Tuning of Adaptive Generalized Predictive Control / McIntosh A.R., S.L. Shah, D.G. Fisher // The Canadian Journal of Chemical Engineering. 1991. Vol. 69. P. 97-110. **11.** *Красовский А.А.* Универсальные алгоритмы оптимального управления непрерывными процессами / А.А. Красовский, В.Н. Буко, В.С. Шендрик. М.: Наука, 1977. 272 с. **12.** *Generalized* predictive control / D. Clarke, C. Mohtadi, P. Tu's // Automatica. 1987. Vol. 23. P. 137-160. **13.** *Allg'ower F., Badgwell T.A., Qin J.S., Rawlings J.B., Wright S.J.* Advances in Control/ Highlights of ECC'99 // Chapt. 12. Nonlinear Predictive Controls and Moving Horizon Estimation. Springer, London. 1999. P. 391-449. **14.** *Анализ* временных рядов. Прогноз и управление / Бокс Дж., Дженкинс Г. / Под ред. В.Ф. Писаренко. М.: Мир, 1974. 197 с. **15.** *An Algorithm* for least squares estimation of non-linear parameters / D.W. Marqvardt // J. Int. Appl. Math. 1963. № 11. P. 431-440. **16.** *Adaptive* general predictive controller for nonlinear systems / O.M. Zhu, K. Warwick, J.L. Douce // IEEE Proceedings-D. 1991. Vol. 138, № 1. P. 33-40. **17.** *Петренко В.Р.* Синтез оптимального регулятора с предсказанием для процесса выращивания объемных Cz-Si монокристаллов // Складні системи і процеси. 2008. №2(14). С.64-76.

Поступила в редколлегию 28.08.2009

Оксанич Анатолий Петрович, д-р техн. наук, профессор, ректор, заведующий кафедрой компьютеризированных систем автоматики Кременчугского университета экономики, информационных технологий и управления. Научные интересы: методы и аппаратура контроля структурно-совершенных полупроводниковых монокристаллов. Адрес: Украина, 39600, Кременчуг, ул. Пролетарская, 24/37, тел. (05366) 3-11-44.

Петренко Василий Радиславович, д-р техн. наук, доцент, проректор по научной работе, заведующий кафедрой информатики Кременчугского университета экономики, информационных технологий и управления. Научные интересы: автоматизация процессов управления производством полупроводниковых материалов, информационные технологии. Адрес: Украина, 39600, Кременчуг, ул. Пролетарская, 24/37, тел. (05366) 3-11-44, E-mail: pvr@ient.net.

Волохов Сергей Александрович, генеральный директор ООО «Силикон». Научные интересы: технологии производства полупроводниковых материалов. Адрес: Украина, 36000, Светловодск, ул. Заводская, 3.

УДК 004.732

А.В. БАБИЧ, МУРАД АЛИ А.

МОДЕЛИ ОБРАТНОЙ СВЯЗИ ДЛЯ ПРОТОКОЛА RTCP

Рассматриваются модели обратной связи для протокола RTCP, использование которых позволяет решить проблему снижения нагрузки на сеть и сбалансированности широковещательного трафика RTP и RTCP. Исследуются их особенности, преимущества и недостатки. Предлагается расширение одной из моделей обратной связи (резюмирующая модель), позволяющее включить в отчеты получателя информацию о факторах, оказывающих наибольшее влияние на сеанс RTP и, таким образом, дающее возможность оперативно выявлять и устранять узкие места данного сеанса. Формулируются задачи, результатом решения которых будет реализация предложенного ранее расширения применительно к протоколу RTCP.

Введение

Благодаря многоадресной природе протоколов RTP/RTCP, все участники сеанса передачи мультимедийных данных получают отчеты обратной связи остальных участников и, таким образом, каждый из них может оценить общее и индивидуальное качество приема и передачи во время сеанса связи, а именно: скорость данных, уровень утерянных пакетов и уровень неравномерности передачи. Хотя трафик RTCP передается таким образом, что его доля в RTP-сеансе не превышает 5%, тем не менее, это чревато двумя проблемами. Во-первых, согласно стандарту [1], возрастание плотности мультимедийного трафика приводит к уменьшению RTCP-трафика и, как результат, снижает вероятность своевременной реакции участников сеанса на изменившиеся условия передачи. Во-вторых, 5% широковещательного трафика является достаточно большим значением. Доля широковещательного трафика в 8% от общей доли трафика уже рассматривается как широковещательный шторм, который может привести к значительному уменьшению пропускной способности сети. Таким образом, снижение уровня многоадресного RTCP-трафика, которое позволит избежать перегрузок в сети и сохранить исходную функциональность RTCP, является достаточно актуальной задачей.

Модели обратной связи RTCP

В качестве решения упомянутой выше задачи в [1] предлагается простая модель обратной связи. Такая модель (рис.1) - базовый механизм "отражения" RTCP-трафика, где каждый участник сеанса передачи мультимедийных данных отсылает нешироковещательным образом специальный пакет обратной связи, так называемый отчет получателя (Receiver Report, RR), к целевому узлу обратной связи (Feedback Target), который пересылает данные отчеты в исходном виде к источнику рассылки (Distribution Source). Далее источник рассылки "отражает" отчеты получателя широковещательным образом всем участникам сеанса передачи мультимедийных данных. Преимущество данного метода заключается в том, что для его использования существующая реализация модуля получателя требует лишь незначительной модификации. Вместо рассылки отчетов по групповому адресу получатель использует одноадресную передачу, в то же время получая "отраженный" RTCP-трафик широковещательным способом.

Таким образом, механизм "отражения" является неплохой альтернативой коммуникационному каналу "многие ко многим", но в то же время использование однонаправленного канала приводит к другой проблеме - ограничению по числу соединений и значительному сокращению масштабируемости для больших мультимедийных сеансов (например, IPTV). Более того, пересылка всех отчетов получателя от каждого участника сеанса мультимедийной передачи данных по однонаправленному каналу неэффективна. Например, в случае вычисления временных меток RTT, которые могут быть полезны только источнику мультимедийных данных, нет никакой необходимости пересылать их в группу участников мультимедиа-сеанса.

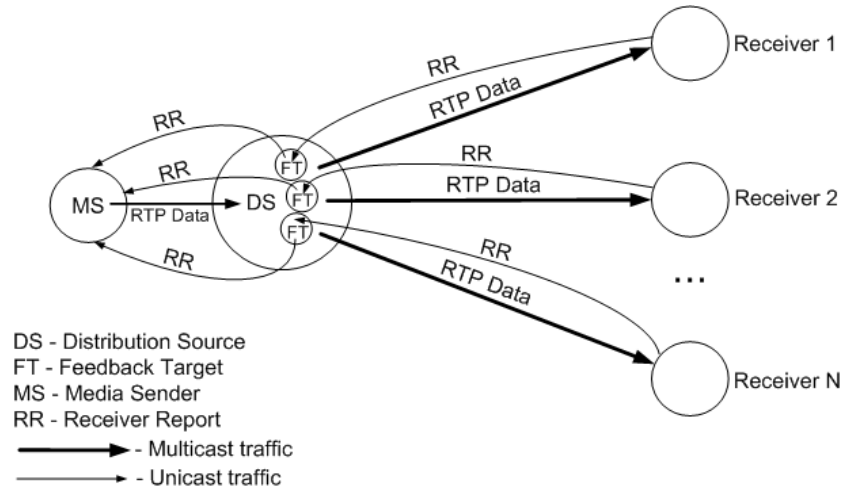


Рис.1. Простая модель обратной связи

Помимо простой модели обратной связи, в области оптимизации трафика обратной связи протокола RTP имеются и другие разработки. Так, в [1-3] представлены следующие модели и методы обратной связи:

– метод резюмирования,
– фильтрование обратной связи,
– алгоритм смещения,
– иерархическое агрегирование обратной связи.

Резюмирующая модель обратной связи источника рассылки - схема сводной отчетности, обеспечивающая экономичное использование пропускной способности путем слияния отчетов получателя на источнике рассылки, необязательно, но возможно при помощи целевого узла обратной связи, в сводные (резюмирующие) пакеты, которые затем рассылаются всем получателям (рис. 2).

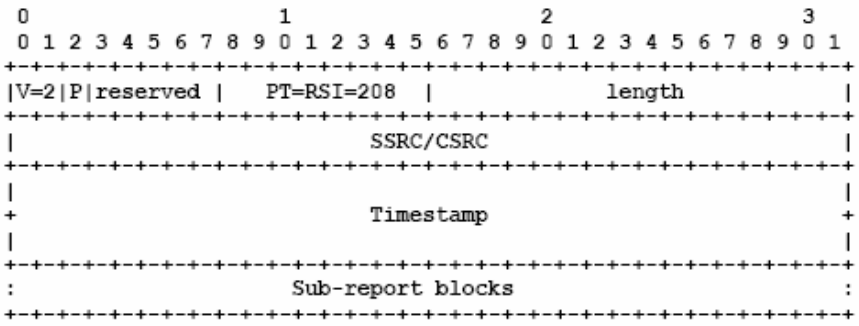


Рис. 2. Структура пакета RTCP-RSI (Report Summary Information) [1]

Преимущество применения последней схемы лучше всего проявляется в сеансах передачи мультимедийного трафика для больших групп, при использовании в которых механизма "отражения" RTCP-трафика, описанного ранее, имеет место генерация значительного числа пересылаемых пакетов во время репликации всей информации на всех получателей. Ясно, что метод резюмирования требует, чтобы все участники сеанса понимали новый формат сводного пакета (рис.3). Вдобавок, резюмирующая схема предоставляет опциональный механизм рассылки информации о данных обратной связи, сообщенных всей группой, в виде значений распределения или гистограммы.

Для однозначного распознавания каждого из рассмотренных методов рассылки отчетов вводится новый идентификатор SDP. При этом метод рассылки отчетов должен быть выбран перед началом сеанса передачи мультимедийных данных и должен оставаться неизменным в течение всего сеанса.

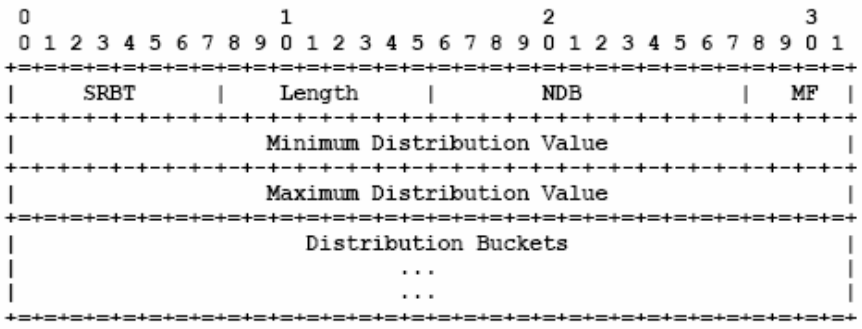


Рис. 3. Общая форма блока отчета [1]

К недостатку резюмирования можно отнести то, что некоторая информация обратной связи, ориентированная на получателя, такая как отображение значений обратной связи в сетевые адреса, больше получателям не доступна. Но для больших групп (каковые предполагаются для IPTV-сервиса, например) резюмирующие отчеты как индикаторы групповых явлений более полезны, чем индивидуальные отчеты получателя. Таким образом, резюмирование еще и обеспечивает возможность реализации функций мониторинга и отладки мультисервисной сети, которые в свою очередь могут быть дополнены персонализированными отчетами, если таковые требуются в заданных условиях функционирования сети.

Модель обратной связи с фильтрованием (рис. 4) базируется на концепции, согласно которой в организации обратной связи медиа-сервера с получателями будут задействованы только некоторые, так называемые выделенные, участники сеанса передачи мультимедийных данных. Здесь основной задачей, требующей решения, является корректный с точки зрения значимости для качества сеанса связи и полноты покрытия выбор выделенных участников мультимедиа-сеанса.

Метод смещения (рис.5) довольно похож на реализацию обратной связи с фильтрованием и также базируется на выборе ряда участников сеанса передачи мультимедийных данных в качестве выделенных. Однако, в отличие от модели обратной связи со смещением, здесь отчеты получателей отсылаются источнику RTP-трафика от всех участников сеанса, но трафик обратной связи от выделенных участников является более приоритетным и интенсивность его передачи не зависит от ширины полосы пропускания, занимаемой трафиком данных мультимедиа. Таким образом, результаты данного метода более объективны. Более того, подгруппа выделенных участников может быть реорганизована в соответствии со значимостью поведения остальной части группы участников сеанса.

Однако, указанный алгоритм чувствителен к вариабельности размеров смещенных групп вследствие мобильности участников сеанса передачи мультимедийных данных (прихода новых и ухода старых членов группы), а также к высокой динамичности значений обратной связи членов смещенной группы. Оба явления влияют не только на точность алгоритма смещения, но и на стандартный RTCP.

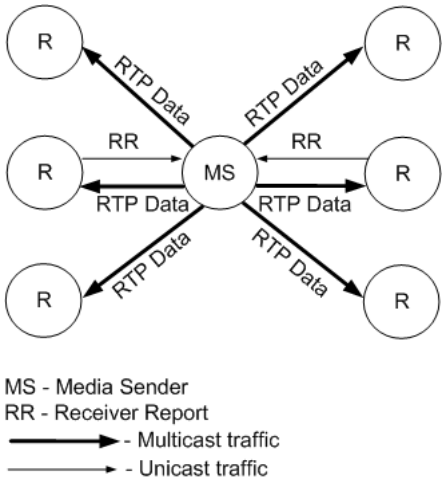


Рис. 4. Модель обратной связи с фильтрованием

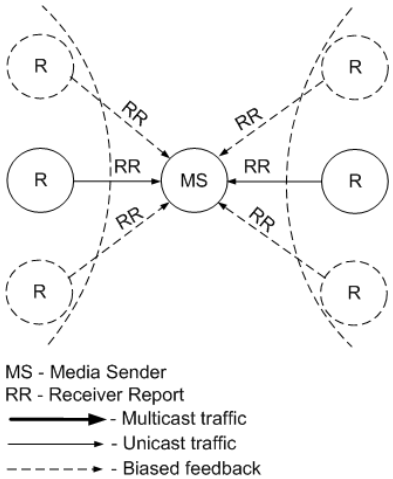


Рис. 5. Модель обратной связи со смещением

Для масштабных сеансов передачи мультимедийных данных предпочтителен метод иерархического агрегирования (рис.6). Он базируется на концепции, в которой данные обратной связи не отсылаются напрямую источнику мультимедиа данных от каждого участника сеанса, а выполняется разбиение всех участников сеанса на подгруппы, предпочтительно равные, в каждой подгруппе выбирается один участник, так называемый агрегатор (см.рис.6), который и является ответственным за сбор отчетов от каждого члена подотчетной ему подгруппы и передачу данных обратной связи источнику трафика RTP.

Реализация иерархического агрегирования также может быть представлена в многоуровневом виде и расширяться практически до любых размеров. Единственная задача, которую необходимо решить, заключается в выборе подходящих агрегаторов.

Главный недостаток иерархического агрегирования состоит в дополнительных временных затратах на передачу данных обратной связи. Другой недостаток заключается в том, что данный механизм зависим от эффективного размещения агрегаторов и гарантии отсутствия недостатков топологии, закладываемой еще на этапе проектирования компьютерной сети. Например, к серьезным последствиям может привести зацикливание пути прохождения через коммутаторы.

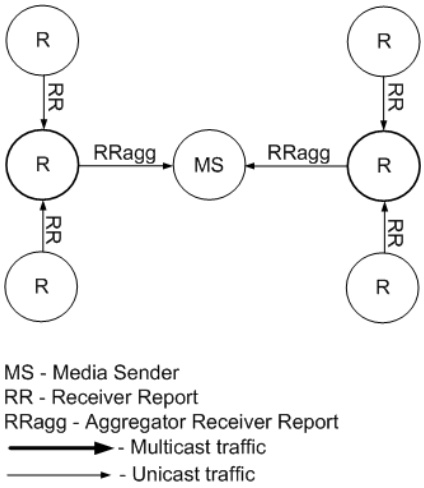


Рис.6. Модель обратной связи с иерархическим агрегированием

Постановка задачи и дальнейшие исследования

Согласно [3], метод резюмирования позволяет экономично использовать пропускную способность транспортной платформы и, как следствие, делает возможным увеличение объема и классов данных, включаемых в резюмирующие пакеты, таких как тип (например, загрузка процессора) или степень детализации и прочее, оставаясь, вместе с тем, в рамках стандартных ограничений полосы пропускания для RTCP. Таким образом, представляется возможным включение в резюмирующий отчет (Report Summary Information, RSI), в поле блоков подотчетов, информации о факторах, оказывающих наибольшее влияние на качество сеанса RTP. Это позволит своевременно выявить узкие места и избежать перегрузок в сети с трафиком реального времени.

Проблема предлагаемого подхода заключается в вычислении и оценивании уровня влияния наблюдаемых факторов. Для ее решения может быть использована теория планирования экспериментов (ТПЭ). ТПЭ является наиболее эффективным способом отбора различных репрезентативных наборов экспериментов, когда процесс или система включают в себя две или более переменных. Здесь ТПЭ формирует такой набор экспериментов, в котором все факторы независимы друг от друга и вместе с тем должны варьироваться одновременно. Результатом будет прогнозирующая модель, показывающая степень значимости всех факторов и их взаимодействий для выходного параметра. Уровень влияния наблюдаемых факторов и их взаимодействий вычисляется с использованием множественной регрессионной модели (1), (2):

$$Y = b_0 + b_i x_i, \quad (1)$$

где b_0 – свободный коэффициент; b_i – коэффициенты регрессии; x_i – факторы, включенные в эксперимент.

Для полнофакторного эксперимента 2^n коэффициенты регрессии b_i для факторов x_i вычисляются по следующей формуле:

$$b_i = \frac{\sum_{U=1}^{N} x_{iU} Y_U}{N}, \quad (2)$$

здесь N – число экспериментов, включенных в план; Y_U – значение параметра Y в каждом эксперименте.

Для реализации данной идеи необходимо решить следующие задачи:

– разработать факторную модель сеанса передачи данных RTP/RTCP,

– разработать план эксперимента для заданной факторной модели,

– выполнить практическую реализацию метода выявления факторов, оказывающих наибольшее влияние на качество сеанса RTP и механизма его внедрения в резюмирующую модель обратной связи RTCP.

Выводы

Рассмотрены модели обратной связи для протокола RTCP, использование которых позволяет решить проблему снижения нагрузки на сети и сбалансированности широковещательного трафика RTP и RTCP. Исследованы их особенности, преимущества и недостатки. Предложено расширение одной из моделей обратной связи (резюмирующая модель), позволяющее включить в отчеты получателя информацию о факторах, оказывающих наибольшее влияние на сеанс RTP и, таким образом, дающее возможность оперативно выявлять и устранять узкие места данного сеанса. Сформулированы задачи, результатом решения которых будет реализация предложенного ранее решения применительно к протоколу RTCP.

Список литературы: 1. *Schulzrinne H., Casner S., Frederick R., Jacobson V.* RTP: A Transport Protocol for RealTime Applications. RFC 3550. Internet Engeneering Task Force, July 2003. **2.** *Ott J., Chesterfield J., Schooler E.* RTCP Extensions for Single-Source Multicast Sessions with Unicast Feedback. IETF draft, AVT-RTCP-SSM, March 2007. **3.** *Chesterfield J., Schooler E.* An Extensible RTCP Control Framework for Large Multimedia Distributions. Proceedings of the Second IEEE International Symposium on Network Computing and Applications, IEEE Computer Society, 2003. **4.** *Komosny D., Novotny V.* Tree Structure for Source-Specific Multicast with Feedback Aggregation. The Sixth International Conference on Networking. Martinique, 2007, ISBN 0-7695-2805-8. **5.** *Komosny D., Novotny V.* Large-Scale RTCP Feedback Optimization. Journal of Networks, Vol. 3, No. 3, March 2008. **6.** *Burget R., Komosny D., Novotny V.* Transmitting Hierarchical Aggregation Information Using RTCP Protocol. International Journal of Computer Science and Network Security, Vol. 7, No. 11, November 2007. **7.** *Burget R., Komosny D., Smilek M.* One Source Multicast Model Using RTP in NS2. International Journal of Computer Science and Network Security, Vol. 7, No. 11, November 2005.

Поступила в редколлегию 14.07.2009

Бабич Анна Витальевна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: сетевые технологии, технологии дистанционного образования. Увлечения: активный отдых, путешествия, иностранные языки. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: babich@kture.kharkov.ua

Мурад Али А., аспирант кафедры АПВТ ХНУРЭ. Научные интересы: компьютерные сети следующего поколения . Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326.

УДК 681.326:519.713

Е.И. ЛИТВИНОВА

ИНФРАСТРУКТУРА ВЕРИФИКАЦИИ И ТЕСТИРОВАНИЯ SOC

Предлагается инфраструктура тестирования и верификации цифровых систем на кристаллах, позволяющая оценивать тестопригодность программно-аппаратных модулей путем построения транзакционного графа для использования механизма ассерций и IEEE 1500 SECT стандарта, что дает возможность повысить эффективность сервисных средств моделирования, диагностирования и восстановления работоспособности, а также существенно уменьшить время верификации HDL-моделей и тестирования аппаратных компонентов SoC.

Задачи исследования: 1. Разработка инфраструктуры верификации, совместимой с современными стандартами и технологиями проектирования цифровых систем на кристаллах от ведущих компаний планеты. 2. Разработка аналитической модели оценивания технологических и структурных решений для анализа цифровых систем, совместимых по интерфейсу с продуктом моделирования QuestaSim для верификации и исправления ошибок в процессе создания HDL-модели. 3. Валидация программно-аппаратных компонентов инфраструктуры отладки HDL-кода, совместимой с существующими системами моделирования, в том числе использующих аппаратные акселераторы. 4. Оценка эффективности промышленного применения инфраструктуры тестирования и верификации на реальных примерах цифровых систем на кристаллах.

1. Инфраструктура проектирования и верификации SoC

Общая инфраструктура разработки и верификации цифровой системы на кристалле является инвариантной (универсальной) по отношению к прототипу и HDL-модели (рис. 1).

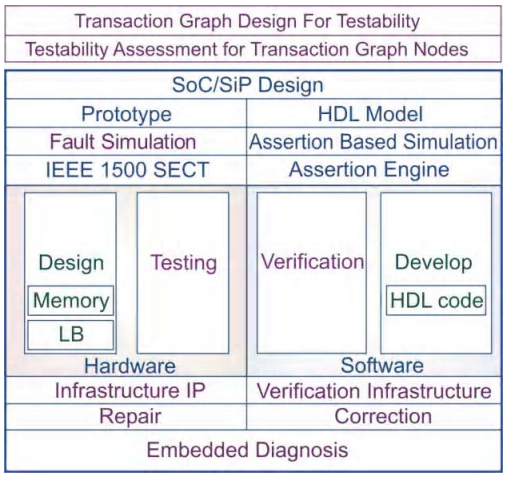


Рис. 1. Инфраструктура проектирования и верификации SoC

Она представляет собой концентрированную совокупность исследований и разработок [1-25], которая содержит следующие компоненты: 1. Transaction Graph Design for Testability [3], что представляет собой создание транзакционной модели программного кода, в которой вершинами являются логические и регистровые переменные, векторы, массивы и память, а дугами – операторы кода, выполняющие транзакции между вершинами. Транзакционный граф в части верификации прототипа строится и для аппаратных компонентов системы. 2. Testability Assessment for Transaction Graph Nodes [24-25] представляет собой вычислительные модели процессов оценки тестопригодности вершин транзакционного графа HDL-кода и прототипа. После этого выполняется процедура выбора достаточного количества вершин транзакционного графа с минимальной оценкой тестопригодности, в которые устанавливаются ассерции для HDL-модели или точки контроля для прототипа путем использования стандарта IEEE 1500 SECT. 3. SoC/SiP Design – выполняется синтез

основных функциональностей на основе применения существующих решений в виде IP-cores из библиотек ведущих компаний мира или путем разработки оригинального кода отдельных функциональностей (рис. 2).

Здесь исходной информацией является спецификация, конечным результатом – прототип, имплементированный в силиконовый кристалл. Кроме того, используется аппаратный акселератор компании Aldec [1-3], позволяющий в десятки раз повысить быстродействие моделирования. 4. Prototype – готовое аппаратное (программное) изделие, свободное от обнаруженных дефектов. 5. HDL-Model – результат автоматического генерирования HDL-кода из спецификации путем использования системных языков описания аппаратуры и промышленных симуляторов [4]. 6. IEEE 1500 SECT – стандарт, ориентированный на сервисное диагностическое обслуживание цифровых систем на кристаллах в процессе проектирования, производства и эксплуатации [5]. 7. Assertion Engine – технологический аппарат тестопригодного проектирования HDL-моделей цифровых систем, ориентированный на контроль исполнения программного кода в его критических точках [6]. 8. Design – проект цифровой системы на кристалле, ориентированный на выполнение актуальных для пользователя функциональностей, которые имплементированы в hardware. 9. Testing – процесс определения технического состояния изделия после его имплементации в силиконовый кристалл. 10. Verification – процесс определения технического состояния изделия на различных стадиях его проектирования путем сравнения со спецификацией на основе моделирования тестовых или функциональных воздействий. 11. Development – итеративный процесс создания изделия на основе спецификации с помощью промышленных систем моделирования и синтеза. 12. Hardware – аппаратная реализация функциональностей цифровой системы на кристалле, ориентированная на выполнение актуальных для EDA-рынка задач. 13. Software – программная реализация функциональностей цифр
ле. 14. Infrastructure IP – инфраструкт
вания функциональных компонентов
вания, производства и эксплуатаци
Infrastructure – совокупность про-
граммно-аппаратных средств, ори-
ентированная на проверку функци-
онирования и коррекции программ-
ного кода. 16. Repair – восстанов-
ление работоспособности аппарат-
ных компонентов системы на крис-
талле путем использования тесто-
вых или функциональных воздей-
ствий, процедур диагностирования
и средств наблюдения от стандар-
та IEEE 1500 SECT. 17. Correction
– процесс устранения семантичес-
ких ошибок HDL-кода, искажающих г
вание модели системы на кристалле
ции. 18. Embedded Diagnosis – встро
неисправностей HDL-модели или пр
внешних средств синтеза и анализа т

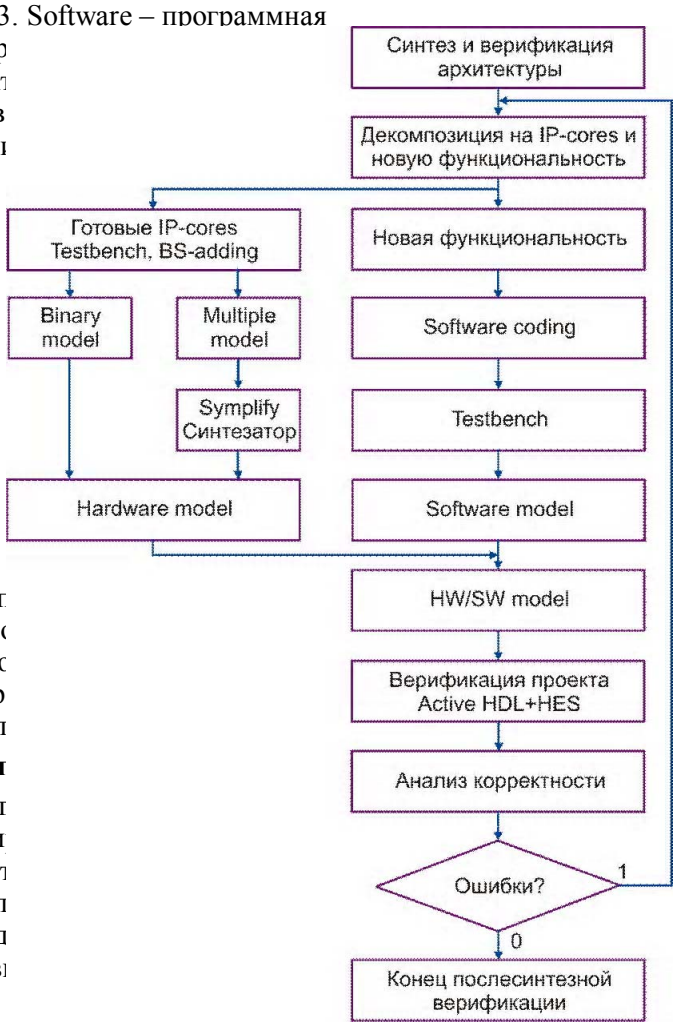


Рис. 2. Маршрут синтеза и верификации SoC

2. Верификация I-IP SoC комп

Процесс верификации компонент
висного обслуживания, как и проекти
этапов (рис. 3). Первый соответствует
нального поведения с помощью модел
скрипт программы Matlab для послед
го эталона, относительно которого в
последующих моделей. М-скрипт
позволяет считывать исходную ин-

формацию из файла, создавать тестовые и эталонные последовательности для модели устройства, выполнять анализ полученных результатов, формировать графический вывод исходных данных и результатов. Второй этап верификации соответствует модели системного уровня проектирования, с архитектурой исполнимой модели, разработанной в Simulink. При этом ввод тестов и анализ результатов осуществляется на основе скриптовых функций программной среды MATLAB, созданных на предыдущем этапе проектирования.

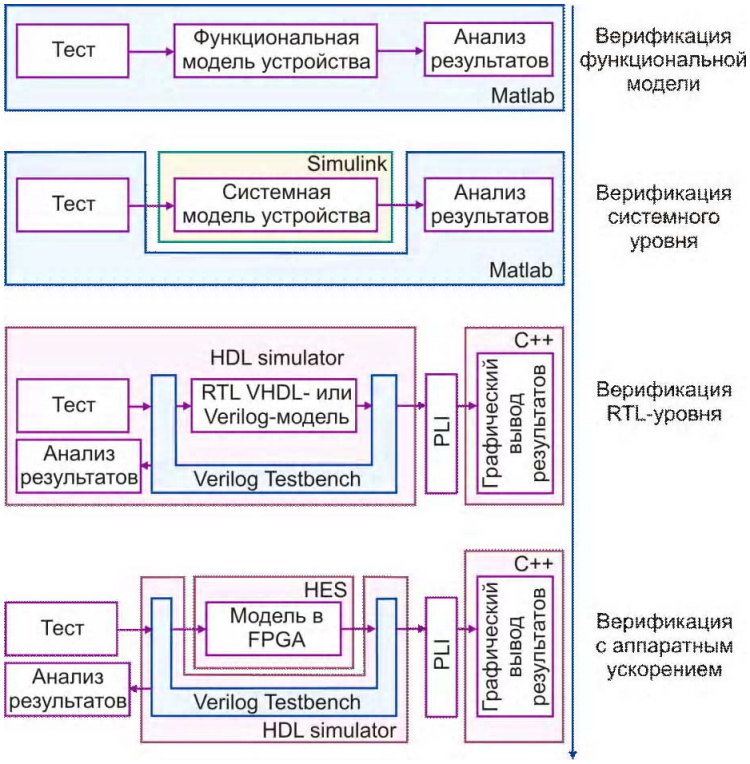


Рис. 3. Схема верификации компонентов I-IP SoC

Верификация RTL-модели, реализованной с помощью языков описания аппаратуры VHDL [10] или Verilog [11-14], выполняется с помощью HDL-симулятора, например Active-HDL или Modelsim. Testbench удобнее реализовать с использованием языка Verilog, поскольку он позволяет подключать через PLI-интерфейс программные модули, созданные с помощью языка C/C++ для визуального контроля процесса верификации. Для ускорения процессов тестирования и аппаратной верификации используется разработанная компанией Alatek плата аппаратного ускорения моделирования HES – Hardare Embedded Simulator. Компилирование проекта выполняется в Active-HDL симуляторе. Далее проект делится на 2 части: одна остается в симуляторе, а другая загружается в HES board. Моделирование проекта в аппаратуре выполняется на частоте 400 МГц. Testbench остается в программном симуляторе Active HDL.

Общая инфраструктура процесса тестирования на основе аппаратной верификации представлена на рис. 4. Здесь фигурируют компоненты: система моделирования Active HDL, Design Verification Manager, модуль создания аппаратной модели проекта и плата аппаратного ускорителя на основе кристалла Xilinx Virtex 4. В качестве примера следует привести основные параметры создания и отладки индустриально-ориентированного проекта Image Processing Design. Время подготовки модели – 40 часов. Время HW-моделирования – 25 секунд, SW-моделирования – 4 часа. Среднее ускорение моделирования для 5 промышленных проектов (900 – 5000 строк HDL-кода) равно двенадцать раз. Дополнительные средние затраты на создание аппаратной модели – 35 часов.

Эффективность разработанной инфраструктуры верификации и тестирования цифровых систем на кристаллах, а также быстродействие программных продуктов относительно базового варианта определяется по следующим формулам:

$$Q^t = \frac{1}{4}\left(\frac{Y^n}{Y^b}k_y + \frac{T^b}{T^n}k_t + \frac{T_v^b}{T_v^n}k_v + \frac{T_d^b}{T_d^n}k_d\right);$$

$$Q^s = \frac{1}{2}\left(\frac{T_s^b}{T_s^n}k_s + \frac{T_f^b}{T_f^n}k_f\right).$$

Здесь фигурируют отношения между новым и базовым вариантами разработок, где параметры: $Y^n, T^n, T_v^n, T_d^n, T_s^n, T_f^n$ – выход годной продукции, время создания изделия для его продажи на рынке, время верификации проекта, время диагностирования дефектных блоков и восстановления работоспособности, время исправного моделирования и моделирования дефектов; коэффициенты $k_y, k_t, k_v, k_d, k_s, k_f$ регламентируют вклад каждого компонента в общую эффективность предложенных разработок, где веса коэффициентов [0,1] определяются пользователем или экспертом в области проектирования.

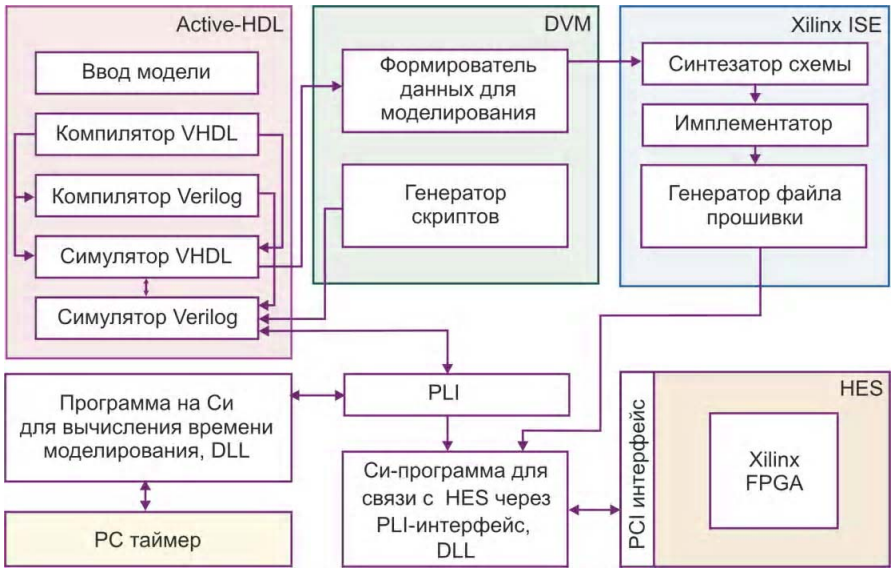


Рис. 4. Инфраструктура HW-ускоренного тестирования проекта

Оценка эффективности в соответствии с предложенными критериями, где все коэффициенты равны 1, проводилась на пяти реальных проектах, взятых из библиотек компаний Xilinx и Aldec. Информация, доступная для проведения статистических исследований, представлена в табл. 1.

Таблица 1. Эффективность инфраструктуры верификации и тестирования

Parameters / Projects	Code	Yield	TTM	VT	FFS	FS	DT	Q^t	Q^s
UART	1300	97/95	140/110	70/55	2500/35	5500/160	56/28	1,145	36
Wavelet–X	4300	94/91	220/160	120/100	4900/75	9900/230	76/33	1,115	37
JPEG 2000	5400	95/94	250/190	150/120	3300/45	6600/150	83/39	1,13	56,38
DSP-Aldec	11000	97/95	270/180	170/150	6500/95	12000/270	94/54	1,075	38,2
Golden Eye	13000	93/90	350/300	250/210	7500/140	15000/390	108/62	1,115	31,3

Здесь оцениваются параметры и характеристики (Yield – выход годной продукции (в %), Time-to-Market – время выхода изделия на рынок (в днях), Verification Time – длительность процесса верификации (в днях), Fault-Free Simulation – время исправного моделирования (в секундах), Fault Simulation – длительность процесса моделирования неисправностей (в секундах), Diagnosis Time – время диагностирования (в миллисекундах), Q^t – интегральная оценка эффективности, Q^s – интегральная оценка производительности инфраструктуры относительно длины HDL кода (Code), который непосредственно влияет на аппаратные затраты проектируемого цифрового изделия).

В соответствии со статистическими данными, представленными в табл. 1, ниже приведены графики: эффективности разработанной инфраструктуры (рис. 5), а также кривые наиболее существенных четырех параметров. Базовые варианты не имеют аппаратных ускорителей (Active HDL), разработанный прототип включает плату повышения производительности процесса моделирования компании Aldec.

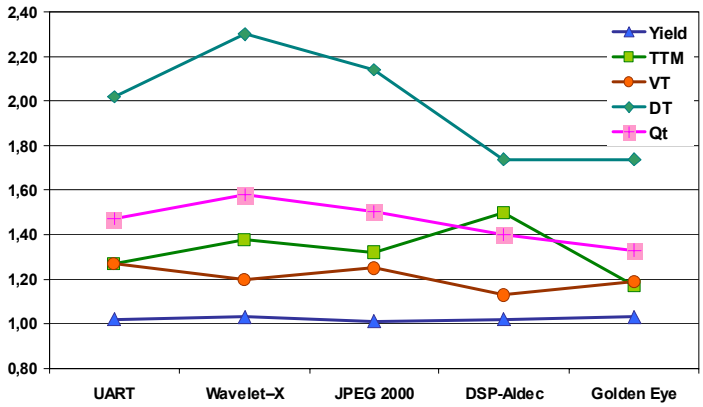


Рис. 5. Эффективность инфраструктуры верификации и тестирования

Повышение эффективности проектирования и эксплуатации цифрового изделия достигается за счет: 1) введения (5%) ассерционной избыточности в программный код функциональности; 2) дополнительных временных затрат – до 5%, взятых от времени создания системной HDL-модели, для построения транзакционного графа и выбора контрольных точек; 3) введения (5%) аппаратной избыточности в виде тестовой инфраструктуры на основе стандарта IEEE 1500 SECT для сервисного обслуживания функциональных модулей цифровой системы на кристалле в процессе ее эксплуатации.

Иерархию верификационной среды можно представить в виде организации компонентов (рис. 6).

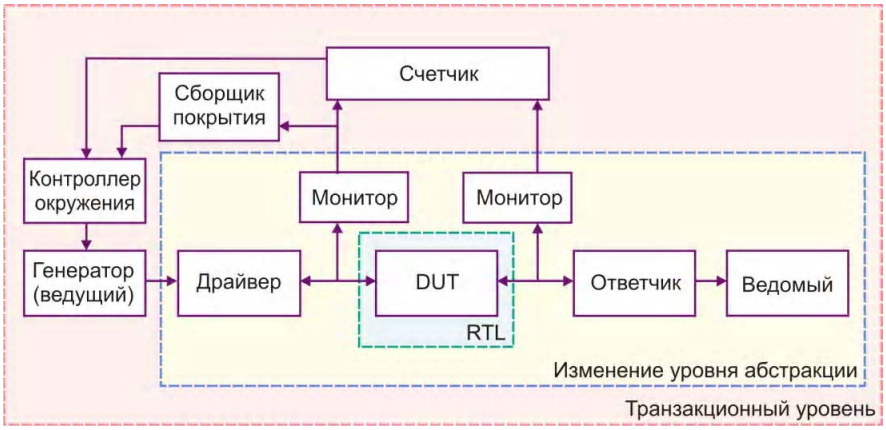


Рис. 6. Организация верификационного окружения

Внутренняя часть есть DUT компонент RTL-уровня абстракции, к которому подключается верификационное окружение TLM-уровня через верификационные компоненты-транзакторы. Роль каждого из них – конвертирование потока данных TLM-уровня для их восприятия на RTL-уровне, и наоборот. Окружение (см. рис. 6) – набор высокоуровневых

компонентов с транзакционными интерфейсами, которые предоставляют все необходимое для функционирования DUT. Компоненты окружения – генераторы тестов, ведущие (masters) и ведомые (slaves) блоки.

Генераторы тестовых воздействий создают поток транзакций для моделирования DUT. Они могут быть псевдослучайными, детерминированными или смешанными; самостоятельными или иметь управляющие сигналы; независимыми или синхронизированными. Простейший генератор выполняет псевдослучайную функцию синтеза тестов, которые передаются драйверу. Генератор сценариев создает детерминированные или смешанные последовательности, которые направлены на инициализацию специфических функций тестируемым устройством. Ведущий – двунаправленный компонент – отсылает запросы и принимает отклики, инициализирует активность, анализирует реакции для определения следующего сценария. Ведомый – двунаправленный компонент, обрабатывает запросы и возвращает отклики. Анализаторы (scoreboard – золотая модель, coverage collector – корзина тестового покрытия транзакций, адресного пространства, количества ошибок) получают информацию о верификационной среде и делают заключение о правильности и окончании процесса верификации. Контроллер формирует поток данных в верификационной среде и управляет ее активностью, пересылает данные от счетчика и коллектора покрытия компонентам окружения, запускает и останавливает генератор тестов.

3. Анализ тестопригодности для реальных HDL-проектов

Достаточно существенная избыточность HDL-модели предполагает ее эффективное использование в целях повышения тестопригодности структуры разработанного или написанного кода. Существующие стандарты тестопригодного проектирования аппаратуры (IEEE 1500, 11.49) можно адаптировать к верификации HDL-кода системных и регистровых программных моделей. Для этого используется модифицированный граф регистровых или транзакционных передач С.Г. Шаршунова [24-25], который предоставляет пользователю информацию о взаимосвязях булевых и регистровых переменных, памяти и интерфейсных шин, называемых транзакторами. Данные для синтеза графа получаются путем синтаксического анализа строк HDL-кода на предмет установления источников и приемников информации, которые являются вершинами (транзакторами) графа, соединенными ориентированными дугами. Каждая из дуг может быть отмечена количеством нагруженных на нее операторов. Построенный таким образом транзакционный граф (TG – Transaction Graph) [24-25] покрывает все функциональности (транзакции) программной модели и задает связи между вершинами, которые соответствуют приему, передаче и преобразованию информации.

Роль транзакционного графа заключается в создании модели передачи данных в целях определения тестопригодности всех вершин. Затем выделяется подмножество вершин, которое имеет минимальное значение тестопригодности, удовлетворяющее условию: $V = \forall i[Q(V_i) \le Q\min] \to Q^t \ge Q^t_{\min}$, где Q^t – качество покрытия тестом (testbench) неисправностей (корзины функциональных покрытий) при модификации структуры цифрового проекта или HDL-кода путем дополнительного мониторинга состояний критических вершин, для которых аппаратурная (программная) избыточность в реальных проектах составляет порядка 5% (Yervant Zorian).

Основными критериями эффективности проектирования изделия на рынке EDA являются выход годной продукции (Y – Yield) и относительное время создания продукта T^{Δ} – time-to-market. Совместно с относительными аппаратурными затратами проекта Z^{Δ} они формируют оценку E эффективности проектирования цифрового изделия, представленную в (1), как приведенное к общим затратам время, необходимое для создания программной и аппаратной функциональности $T^{\Delta}(S), T^{\Delta}(H)$, умноженное на аналогично приведенные программные и аппаратные затраты $Z^{\Delta}(S) \times Z^{\Delta}(H)$, а также на выход годной продукции Y. Последний параметр $Y = (1-P)^{n(1-Q)}$ зависит от тестопригодности (качества) проекта Q,

вероятности P существования в кристалле неисправных областей и числа необнаруженных дефектов n или D. Критерий временных затрат T^{Δ} также определяется тестопригодностью проекта Q и размерностью его программной Z(S), аппаратной модели Z(H), приведенной к дневной или часовой производительности $Z^1(S)\,[Z^1(H)]$ разработчика кода (аппаратуры). Коэффициенты k_q, k_w задают части временного интервала, необходимого для написания HDL-кода – $k_w = 0{,}3$ и верификации – $k_q = 0{,}7$ проекта, $k_q + k_w = 1$. Параметры $T^+(S), T^+(H)$ определяют время создания программной и аппаратной функциональностей, а $T^-(S), T^-(H)$ – затраты для их сервисного обслуживания, которые включают следующие компоненты:

$T(F^s), T(T^s), T(A), T(G^s)$ $[\,T(F^h), T(T^h), T(B), T(G^h)\,]$ – дополнительный период времени на создание функционального покрытия, тестовых последовательностей testbench, механизма ассерций (регистра граничного сканирования) и транзакционного графа программной (аппаратной) модели. Размерность HDL-кода и сложность аппаратуры для проекта определяется программными (аппаратными) компонентами:

$$Z^{\Sigma}(S) = Z(S), Z(F^s), Z(T^s), Z(A), Z(G^s)\ \ [\,Z^{\Sigma}(H) = Z(H), Z(F^h), Z(T^h), Z(B), Z(G^h)\,],$$

где в правой части равенства представлены компоненты: функциональность, функциональное покрытие, testbench (тест), механизм ассерций (регистр граничного сканирования), транзакционный граф программной (аппаратной) модели. Общая модель эффективности проектирования SoC имеет вид:

$$
\begin{aligned}
&E = \frac{1}{3}(Y + T^{\Delta} + Z^{\Delta}),\\
&Y = (1-P)^{n(1-Q)};\ D = 1 - Y^{(1-Q)};\\
&T^{\Delta} = T^{\Delta}(S) \times T^{\Delta}(H);\\
&Z^{\Delta} = Z^{\Delta}(S) \times Z^{\Delta}(H);\\
&T^{\Delta}(S) = \frac{T^+(S)}{T^+(S) + T^-(S)};\ T^{\Delta}(H) = \frac{T^+(H)}{T^+(H) + T^-(H)};\\
&T^+(S) = \left(k_w + k_q \frac{1-Q}{1+Q}\right)\frac{Z(S)}{Z^1(S)};\\
&T^-(S) = \frac{1-Q}{1+Q}\left[\frac{Z(F^s)}{Z^1(F^s)} + \frac{Z(T^s)}{Z^1(T^s)} + \frac{Z(A)}{Z^1(A)}\right] + \frac{Z(G^s)}{Z^1(G^s)};\\
&T^+(H) = \left(k_w + k_q \frac{1-Q}{1+Q}\right)\frac{Z(H)}{Z^1(H)};\\
&T^-(H) = \frac{1-Q}{1+Q}\left[\frac{Z(F^h)}{Z^1(F^h)} + \frac{Z(T^h)}{Z^1(T^h)} + \frac{Z(B)}{Z^1(B)}\right] + \frac{Z(G^h)}{Z^1(G^h)};\\
&T(S) = T^+(S) + T^-(S);\ T(H) = T^+(H) + T^-(H);\\
&Z^{\Delta}(S) = \frac{Z(S)}{Z(S) + Z(F^s) + Z(T^s) + Z(A) + Z(G^s)};\\
&Z^{\Delta}(H) = \frac{Z(H)}{Z(H) + Z(F^h) + Z(T^h) + Z(B) + Z(G^h)};
\end{aligned}
\qquad (1)
$$

$$Z^{\Sigma}(S) = Z(S) + Z(F^{s}) + Z(T^{s}) + Z(A) + Z(G^{s});$$

$$Z^{\Sigma}(H) = Z(H) + Z(F^{h}) + Z(T^{h}) + Z(B) + Z(G^{h});$$

$$Q = \{Q^{h}, Q^{s}\} = \frac{1}{n}\sum_{i=1}^{n}(U_i \times N_i);$$

$$U_i = \frac{1}{\tau}\sum_{j=1}^{x_i} T_j^i \times \frac{1}{d_i^x \vee t_i^x};\ N_i = \frac{1}{\tau}\sum_{j=1}^{y_i} T_j^i \times \frac{1}{d_i^y \vee t_i^y}.$$

В (1) интегральная оценка тестопригодности $Q = \{Q^{h}, Q^{s}\}$ транзакционных графов HDL-кода и аппаратной модели регистрового уровня функционально зависит от управляемости и наблюдаемости их вершин U_i, N_i, где n – количество вершин транзакционного графа. Управляемость и наблюдаемость есть метрика оценивания тестопригодности (культуры структуризации) не соединительных линий, а компонентов HDL-кода, таких как: регистр, счетчик, память или массивы, вход-выходные шины, векторы, логические или арифметические переменные цифрового проекта.

Управляемость вершины имеет функциональную зависимость от структурной глубины d_i^x нахождения транзактора относительно входов или длины конъюнктивного терма – t_i^x. Наблюдаемость имеет аналогичную зависимость $d_i^y \vee t_i^y$ относительно выходов. Для подсчета тестопригодности можно использовать один из параметров $d_i^x, t_i^x (d_i^y, t_i^y)$. Оценки управляемости и наблюдаемости зависят также от процентного отношения числа команд $\frac{1}{\tau}\sum_{j=1}^{x_i} T_j^i$, имеющих входной (выходной – $\frac{1}{\tau}\sum_{j=1}^{y_i} T_j^i$) доступ к вершине при анализе данной программы, к общему количеству команд τ, где x_i – число команд, формирующих доступ к входной вершине; (y_i) – количество команд, определяющих вершину как источник данных. Тестопригодность Q, представленная в (1), зависит от управляемости U, наблюдаемости (N), а также от стоимости реализации (Z) компонентов: метрики функционального покрытия (F), testbench (B), механизма ассерций (A), функциональности (S). Управляемость (наблюдаемость) есть функция от числа операторов, входящих в вершину (исходящих из вершины) транзакционного графа, а также от структурной глубины рассматриваемого элемента – расстояния от входов (выходов) или от количества временных тактов, необходимых для управления (наблюдения) компонента в заданном состоянии на временной оси. Влияние мощности L^m линий мониторинга проекта на изменение (увеличение, уменьшение) всех существенных параметров процесса проектирования цифровой системы на кристалле определяется следующим выражением:

$$[(L^m \uparrow \rightarrow (Z(B)\uparrow, Z(A)\uparrow, T(B)\uparrow, T(A)\uparrow)] \rightarrow [(Y\uparrow, Q\uparrow, D^f\uparrow)\,\&$$
$$\&\,(T^{\Delta}\downarrow, T^{sim}\downarrow, T^{t\_gen}\downarrow, T^{diag}\downarrow, T^{t\_hw}\downarrow, T^{v\_sw}\downarrow, L^{ud\_f}\downarrow)]. \qquad (2)$$

Вербальное и последовательное пояснение всех символов, входящих в выражение – увеличение числа точек наблюдения в программе или аппаратуре происходит за счет добавления аппаратной избыточности в виде регистра граничного сканирования, программной избыточности в виде механизма ассерций, а также за счет дополнительного времени создания упомянутых компонентов. Это дает возможность существенно увеличить выход годной продукции, тестопригодность проекта, глубину диагностирования дефектов и ошибок. Кроме того, существенно уменьшается время: выхода изделия на рынок, моделирования неисправностей, генерации тестов, диагностирования дефектов и ошибок, тестирования функциональности аппаратуры и верификации программного кода. Также существенно уменьшается число необнаруженных дорогостоящих дефектов и ошибок, которое влияет на выход годной продукции Yield.

Для численной оценки влияния параметров на качество и период проектирования, заданных в (2), ниже вводится интегральная нормированная средняя аддитивная оценка эффективности процесса проектирования. Она представлена суммой функционалов, формирую-

щих затратные (f_i^-) и выигрышные (f_i^+) функции, зависящие от числа линий мониторинга программного или аппаратного проектов:

$$E = \frac{1}{n}\sum_{i=1}^{n} k_i f_i,$$

$$f(L^m) = \begin{cases} \{f_1^- = Z(B);\ f_2^- = Z(A);\ f_3^- = T(B);\ f_4^- = T(A)\}; \\ \{f_5^+ = Y;\ f_6^+ = Q;\ f_7^+ = D^f\}; \\ \{f_8^+ = T^{\Delta};\ f_9^+ = T^{sim};\ f_{10}^+ = T^{t\_gen};\ f_{11}^+ = T^{diag}; \\ f_{12}^+ = T^{t\_hw};\ f_{13}^+ = T^{v\_sw};\ f_{14}^+ = L^{ud\_f}\}. \end{cases}$$

Здесь коэффициент k_i определяет весомозначность функционала в формировании интегральной оценки, которая используется для нахождения оптимизированной стратегии проектирования цифрового изделия.

Интегральной оценке E ставятся в соответствие графики зависимостей наиболее существенных функционалов от числа линий мониторинга, которые представлены на рис. 7.

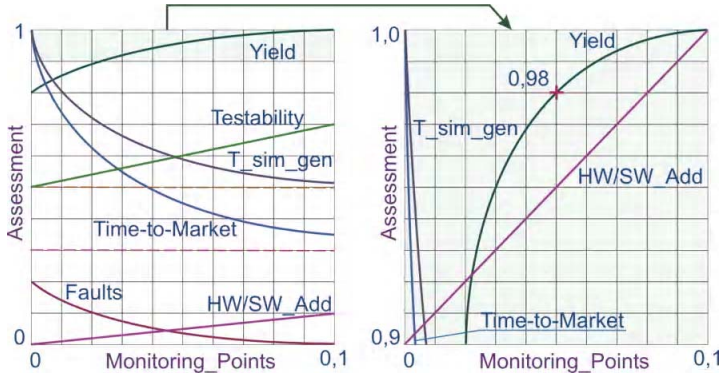


Рис. 7. Влияние точек контроля на эффективность проектирования

В левой части рис. 7 представлены зависимости эффективностей (Yield – выход годной продукции, Testability – тестопригодность проекта, T_sim_gen – время моделирования, генерации тестов и поиска дефектов, Time-to-Market – время выхода изделия на рынок, HW/SW_Add – дополнительные аппаратные и программные затраты для реализации IEEE 1500 регистра граничного сканирования и механизма ассерций, Faults – количество непроверенных дефектов) от числа линий наблюдения в интервале [0-0,1]. Правая часть рис. 7 иллюстрирует поведение отдельных функционалов в укрупненном масштабе (10:1) интервала [0,9-1] по оси ординат левого графика.

Для расчета суммарного эффекта от имплементации в проект дополнительных линий наблюдения и затрат (аппаратных и программных) необходимо модифицировать приведенный в (1) критерий эффективности к следующему виду:

$$E = \frac{1}{2 \times Z^{\Delta}}(Y^{\Delta} + T^{\Delta}).$$

Данный критерий определяет оценку практической значимости полезных приращений, приведенных к проценту аппаратурных и программных затрат, или эффект от разработанных моделей и методов тестопригодного проектирования, моделирования, тестирования и верификации на 5 реальных проектах. Все существенные параметры сведены в табл. 2.

Средняя оценка эффекта для пяти проектов равна 7,83. Это означает, каждый процент аппаратурной, программной избыточности приносит почти восемь процентов эффекта по времени моделирования, тестирования, верификации и качеству изделия. Графики приращения данных параметров проектирования в функциональной зависимости от избыточности представлены на рис. 8.

4. Выводы

Научная новизна и практическая значимость.

1. С учетом результатов, опубликованных ранее в [3, 16-25], достигнута основная цель исследования, которая заключается в существенном уменьшении времени верификации HDL-моделей и тестирования аппаратных компонентов SoC путем создания инфраструктуры анализа проекта, позволяющей: 1) оценивать тестопригодность программно-аппаратных модулей путем построения транзакционного графа для использования механизма ассерций и IEEE 1500 SECT стандарта; 2) модифицировать структуру проекта путем введения дополнительных точек мониторинга; повышать производительность сервисных средств моделирования, диагностирования и восстановления работоспособности программных и аппаратных компонентов цифровых систем на кристаллах.

Таблица 2. Приращения параметров проектирования

Design	Code	Ntk	%tk	ΔQH/QS	ΔAddH/S	ΔTsim	ΔTgen	ΔYield	ΔTTM	E
UART	1300	52	0,04	0,08	0,047	0,375	0,275	0,145	0,505	6,9
Wavelet–X	4300	167	0,04	0,085	0,045	0,365	0,265	0,14	0,5	7,1
JPEG 2000	5400	189	0,04	0,075	0,044	0,34	0,234	0,135	0,485	7,05
DSP-Aldec	11000	374	0,03	0,07	0,035	0,35	0,235	0,13	0,475	8,6
Golden Eye	13000	390	0,03	0,06	0,03	0,33	0,233	0,12	0,45	9,5

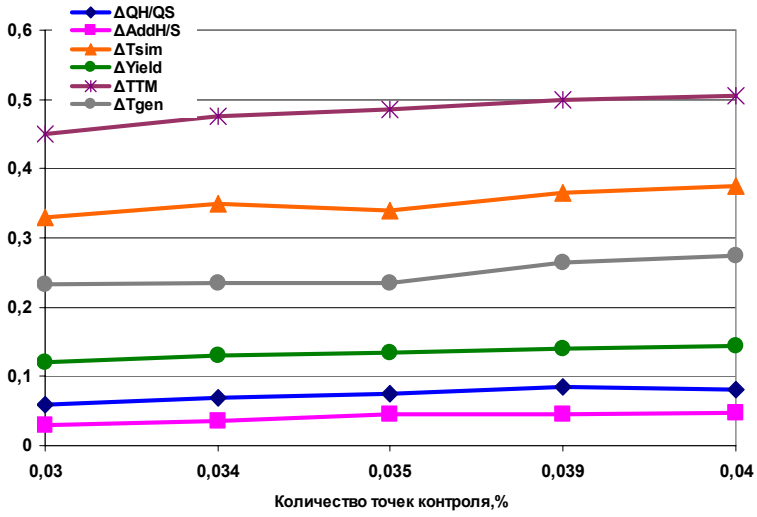


Рис. 8. Графики приращения параметров проектирования

2. Предложена инфраструктура верификации цифровых систем на кристаллах, которая является инвариантной (универсальной) по отношению к прототипу и HDL-модели и содержит следующие компоненты: 1) Transaction Graph Design for Testability – транзакционная модель программного кода. Здесь вершинами являются логические и регистровые переменные, векторы, массивы и память, а дугами – операторы кода, выполняющие транзакции между вершинами. Транзакционный граф в части верификации прототипа строится и для аппаратных компонентов системы. 2) Testability Assessment for Transaction Graph Nodes – вычислительные модели процессов оценки тестопригодности вершин транзакционного графа HDL-кода и прототипа. Они необходимы для выполнения процедуры выбора достаточного количества вершин транзакционного графа с минимальной оценкой тестопригодности, в которые устанавливаются ассерции (для HDL-модели) путем использования стандарта IEEE 1500 SECT для прототипа.

3. Инфраструктура верификации интегрирована с программой моделирования QuestaSim для исправления ошибок в процессе создания HDL-модели. Проведена валидация работоспособности инфраструктуры отладки программного кода совместно с системой моделирования Active HDL, Aldec, использующей аппаратный акселератор. Осуществлена оценка эффективности инфраструктуры тестирования и верификации путем выполнения экспериментов над реальными цифровыми системами.

4. Средняя оценка суммарного эффекта от внедрения инфраструктуры встроенного тестирования для пяти реальных проектов равна 7,83, что означает: каждый процент аппаратурной, программной избыточности приносит почти восемь процентов суммарного эффекта, включающего время моделирования, тестирования, верификации, выхода продукции на рынок. Дальнейшие исследования необходимо связывать с объектно-ориентированной верификацией (ООВ), которая представляет собой актуальнейшую и динамически развивающуюся область. Ведущие компании EDA-индустрии предлагают свои продукты, позволяющие внедрить ключевые идеи ООВ в верификационное окружение. Самые стабильные и конкурентоспособные методологии разрабатываются такими вендорами как Synopsys (VMM) и Mentor Graphics (AVM), которые являются инициаторами появления и внедрения SystemVerilog. Несмотря на достаточно существенные отличия в подходах, VMM и AVM технологии верификации становятся широко применимыми стандартами дефакто. Таким образом, основной тренд развития современных структур отладки связывается с созданием объектно-ориентированных библиотек тестовых решений и разработкой конкурентоспособных методов верификации на высоком уровне абстракции.

Список литературы: 1. *Aldec*, Inc. Official web-site: http://www.aldec.com. **2.** Zorian Yervant. What is Infrustructure IP? // IEEE Design & Test of Computers. 2002. P. 5-7. **3.** *Хаханов В.И.* Проектирование и тестирование цифровых систем на кристаллах / В.И. Хаханов, Е.И. Литвинова, О.А. Гузь. Харьков: ХНУРЭ, 2009. 484с. **4.** *Bergeron Janick.* Writing testbenches: functional verification of HDL models. Boston: Kluwer Academic Publishers. 2001. **5.** *IEEE* 1500 Web Site. http://grouper.ieee.org/groups/1500. **6.** *Harry Foster, Adam Krolnik, David Lacey.* Assertion-based design. Kluwer Academic Publishers. Springer. 2005. 392 p. **7.** *Tabatabaei Sassan, Ivanov Andre.* Embedded Timing Analysis: A SoC Infrastructure // IEEE Design and Test of Computers. 2002. P. 24-36. **8.** *Abramovici M., Breuer M.A. and Friedman A.D.* Digital System Testing and Testable Design.– Computer Science Press. 1998. 652 p. **9.** *Marinissen E. J., Zorian Y.* IEEE Std 1500 Enables Modular SoC Testing // Design & Test of Computers.Jan./Feb. 2009. P.8-16. **10.** *Shoukourian S., Vardanian V., Zorian Y.* SoC Yield Optimization via an Embedded-Memory Test and Repair Infrastructure // IEEE Design and Test of Computers. 2004. P. 200-207. **11.** *IEEE* standard 1076-2000, "IEEE Standard VHDL Language Reference Manual", January 2000. **12.** *IEEE* Std 1800-2005. IEEE Standard for System Verilog. **13.** *Synopsys*, Inc. Synopsys Delivers First Complete SystemVerilog Design and Verification Flow. Official web site, 2006. **14.** *Mentor* Graphics, Inc. Mentor Graphics Announces HDL Designer Series with SystemVerilog Support for Design-to-Verification Productivity. Official web site, 2007. **15.** *Janick Bergeron, Eduard Cerny, Alan Hunter, Andrew Nightingale.* Verification Methodology. Manual for SystemVerilog. Springer. 2005. **16.** *Шаршунов С.Г.* Построение тестов микропроцессоров. 1. Общая модель. Проверка обработки данных // Автоматика и телемеханика. 1985. №11. С. 145-155. **17.** *Hahanov V.* General testing models of SoC hardware-software components // Radioelectronics & Informatics. 2008. №1. P. 88-95. **18.** *Хаханов В.И.* Восстановление работоспособности встроенной памяти SoC // Науково-технічний журнал «Інформаційно-керуючі системи на залізничному транспорті». 2008. № 4. P. 120-127. **19.** *Hahanov V.* Embedded Method of SoC Diagnosis / V. Hahanov, E. Litvinova, V. Obrizan, W. Gharibi // Electronics and Electrical Engineering. 2008. № 8(88). P. 3-8. **20.** *Hahanov Vladimir.* Algebra-Logical Diagnosis Model for SoC F-IP / Vladimir Hahanov, Vladimir Obrizan, Eugenia Litvinova, Ka Lok Man // WSEAS Transactions on Circuits and Systems.– Issue 7, Vol. 7. July, 2008. P. 708-717. **21.** *Хаханов В.И.* Сервисное обслуживание современных цифровых систем на кристаллах / В.И. Хаханов, Е.И. Литвинова, Ngene Christopher Umerah // Радиоэлектронные и компьютерные системы. 2009. № 7 (41). С. 319-323. **22.** *Литвинова Е.И.* Технологии диагностирования и восстановления System-in-Package / Е. И. Литвинова // АСУ и приборы автоматики. 2009. № 146. С. 4–21. **23.** *Литвинова Е.И.* Метод покрытия неисправных логических блоков цифровых систем на кристаллах ремонтными клетками // Радиоэлектроника и информатика. 2009. №1. С. 46-49. **24.** *Хаханов В.И.* Технология покрытия дефектных блоков резервными компонентами / В.И. Хаханов, С.В. Чумаченко, Е.И. Литвинова, О.В. Захарченко// АСУ и приборы автоматики. 2009. Вып. 147. С. 52-64. **25.** *Хаханов В.И.* Тестирование и верификация HDL-моделей компонентов SOC. I. / В.И. Хаханов, Е.И. Литвинова, С.В. Чумаченко, И.А. Побеженко, Ngene Christopher Umerah // Радиоэлектроника и информатика. 2009. № 3. С. 45-52. **26.** *Хаханов В.И.* Тестирование и верификация HDL-моделей компонентов SOC. II. / В.И. Хаханов, Е.И. Литвинова, И.А. Побеженко, Tiecoura Yves // АСУ и приборы автоматики. 2009. Вып. 148. С. 26-37.

Поступила в редколлегию 28.08.2009

Литвинова Евгения Ивановна, канд. техн. наук, доцент кафедры ТАПР ХНУРЭ. Научные интересы: алгоритмизация задач автоматизированного проектирования электронных вычислительных средств, автоматизация диагностирования и встроенный ремонт компонентов цифровых систем в пакете (SiP). Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-421, e-mail: kiu@kture.kharkov.ua.

УДК 681.326:519.713

В.И. ХАХАНОВ, И.А. ПОБЕЖЕНКО, В.А. ВАСИЛЕНКО, С.В. ЧУМАЧЕНКО

МЕТОД ВЕРИФИКАЦИИ HDL-КОДА НА ОСНОВЕ ТРАНЗАКЦИОННОГО ЛОГИЧЕСКОГО ГРАФА

Предлагается логический метод диагностирования ошибок программного HDL-кода, который использует транзакционный граф программы и ее компонентов, что позволяет определять критические точки для мониторинга выполнения программы в целях установки в них ассерционных операторов, ориентированных на существенное (40%) уменьшение времени верификации программной модели изделия. Приводятся примеры синтеза транзакционного графа и диагностирования ошибок программных модулей, подтверждающие эффективность практического использования метода.

Стоимость и временные затраты на создание проекта распределяются: на написание кода – 30%, на тестирование и верификацию – 70%. Поэтому тестовое обеспечение и сервисное обслуживание проекта и готового изделия есть важные составляющие обеспечения качества цифровой системы на кристалле, которые закладываются уже на стадии системного проектирования в виде инфраструктуры сервисного обслуживания, включающей тестопригодность, применение стандартов IEEE 1500, 11.49, 1450, механизмы ассерций, транзакционные графы управляющего и операционного автоматов, а также программных кодов. Обеспечение тестопригодности есть не что иное, как дополнительные аппаратурные и программные затраты, а также избыточное время для их создания. Тем не менее, издержки, составляющие порядка 5% от полезной функциональности, дают существенный эффект в части: повышения выхода годной продукции – 3-4%; уменьшения общего времени создания продукта – 35%; повышения производительности средств тестирования и верификации – x10 раз.

Предлагается метод верификации системных HDL-моделей, ориентированный на существенное повышение качества проектируемых компонентов цифровой системы, выполняющей функцию вейвлет-преобразования, что также позволяет уменьшить время разработки (time-to-market) путем использования программной избыточности в виде механизма ассерций. Методы ориентированы на поиск ошибок и дефектов с заданной глубиной в программном HDL-коде путем введения в критические точки транзакционной модели наблюдателя в виде ассерционной избыточности. Для определения критических точек используется известная в hardware design and test технология вычисления управляемости и наблюдаемости структурных компонентов программного кода в целях улучшения его тестопригодности для диагностирования семантических ошибок.

Цель – разработка логического метода диагностирования ошибок программного кода на основе синтеза транзакционного графа, позволяющего определять критические точки мониторинга HDL-кода для установки в них механизма ассерций, что позволяет на 40% уменьшить время верификации программной модели изделия.

Задачи исследования: 1. Описание современных технологий и маршрутов тестопригодного проектирования HDL-моделей на основе ассерционной избыточности, использования обязательных компонентов Testbench и Coverage. 2. Разработка аналитической модели верификации HDL-кода на основе механизма ассерций. 3. Создание метрики для оценивания тестопригодности HDL-кода на основе синтеза транзакционного графа. 4. Создание новой иерархической модели программного кода вейвлет-преобразования в целях подсчета тестопригодности для имплементации ассерций в критические точки мониторинга. 5. Синтез таблиц неисправностей для программных моделей вейвлет-преобразования в целях их анализа для поиска ошибок и дефектов. 6. Практическое использование логического метода для поиска семантических ошибок HDL-кода.

Источники исследования: 1. Модели, методы и средства создания тестов и testbench [10-12]. 2. Средства верификации системных моделей на основе механизма ассерций [13-16]. Стандарты тестопригодного проектирования программных продуктов от общества IEEE [17-19]. Современные разработки для верификации и анализа тестопригодности цифровых систем на кристаллах [7, 8, 14, 20-23].

1. Актуальные технологии тестопригодности HDL-кода

В плане предлагаемых исследований интересной представляется опубликованная в EE Times (12.2009) десятка перспективных технологий от Gartner Research Group (Gary Smith) для ближайших лет. В ней нашли отражение только технологии, связанные с разработкой специализированных цифровых изделий на кристаллах, хотя развитие программных технологий также будет оказывать сильное влияние на состояние рынка электроники. Глобально важными остаются технологии снижения потребляемой мощности и решения, направленные на уменьшение содержания ценных материалов в продукте. Данные технологии выступают двигателями многих направлений развития электроники, перечисленных ниже: 1. Биологическая обратная связь или электроника, управляемая мыслью. 2. Печатная электроника на основе использования органических материалов. 3. Пластиковая память на основе полимеров, проявляющих ферроэлектрические свойства. 4. Безмасочная литография на основе использования электронного луча для создания топологии схемы. 5. Параллельная обработка данных для многоядерных гетерогенных графических процессоров. 6. Сбор энергии от механических и электрических процессов в окружающей среде. 7. Биоэлектроника и wetware, сочетающие биологические объекты и электронику для медицины. 8. Резистивное ОЗУ или мемристор с эффектом памяти как четвертый пассивный элемент электронной схемы, дополняющий резистор, конденсатор и индуктивность. 9. Переходные отверстия в кремнии (Through-Silicon-Via – TSV) для создания реальных 3D-SiP и кристаллов. 10. Различные технологии батарей на основе сочетания никеля и лития.

За последние 30 лет происходит взаимовыгодный обмен технологиями между аппаратными и программными разработчиками, в качестве которых выступают ведущие компании планеты (Intel, Microsoft, Cadence, Mentor Graphics, Synopsys, Aldec). Следует привести некоторые отдельные факты плодотворного сотрудничества. 1. IEEE стандарты граничного сканирования [12] на уровне платы и кристалла породили механизм ассерций для верификации программных продуктов. 2. Тестопригодность [1,2], управляемость и наблюдаемость цифровых структур адаптирована для оценки качества программного кода и последующего его улучшения для быстрой отладки модели. 3. Графовые модели регистровых передач [18] адаптируются для оценки и улучшения тестопригодности программных продуктов путем установки ассерций. 4. Разделение автомата на управляющую [10, 16] и операционную части используется для упрощения процесса верификации программного кода, который фактически имеет аналогичные составляющие. 5. Компонент testbench [14, 9], используемый для тестирования аппаратных проектов, появляется и в программных продуктах, реализованных на уровне языков C++ и выше. Testbench – специализированная структура операционного устройства, реализованная в HDL-коде и предназначенная для тестирования и верификации цифрового проекта с помощью средств моделирования, ассерционной избыточности, а также компонентов, обеспечивающих управление, наблюдение и принятие решения о техническом состоянии проверяемого объекта. Он включает сгенерированные вручную или автоматически входные и выходные данные, описывающие идеальную модель устройства и составляющие основу testbench. 6. Инфраструктуры сервисного обслуживания F-IP в рамках I-IP [11] используются для встроенного тестирования компонентов программной системы на основе механизма ассерций. 7. Адресность компонентов SoC, реализованных в аппаратном исполнении [18-20], предоставляет также и программному продукту свойство самовосстановления работоспособности средствами I-IP на расстоянии, благодаря наличию связи кристалла с внешним миром посредством Internet или беспроводных технологий. Дистанционная коррекция программных ошибок возможна благодаря использованию ПЛИС, куда, в случае обнаружения неисправности, можно записать новый bit stream, не имеющий ошибок, что фактически создает новую аппаратуру путем повторного программирования кристалла.

Жизненный цикл программного или аппаратного изделия представлен на рис. 1. Он имеет стадии: проектирование, усовершенствование проекта и производства, сопровождение изделия. Здесь важно найти новые средства для поднятия кривой вверх и ее сжатия по оси времени за счет: быстрого устранения ошибок разработки; исправления кода, имплементированного в память системы на кристалле; выпуска service pack, корректирующего ошибки путем его распространения через Internet или спутники.

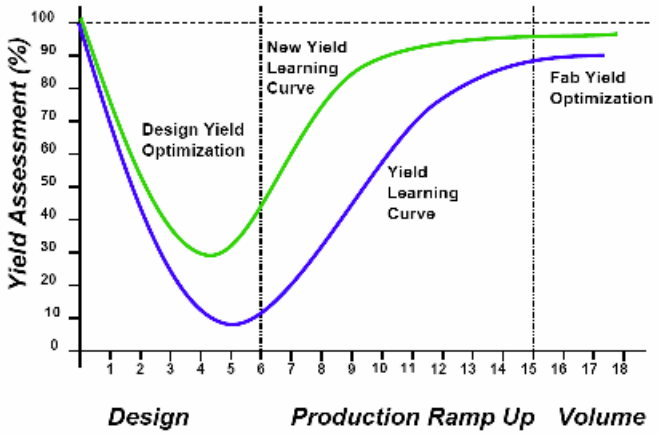


Рис. 1. Жизненный цикл программно-аппаратного продукта

Сущность данного исследования заключается в повышении уровня годной продукции и уменьшении времени создания проекта за счет имплементации в HDL-код программной избыточности в виде механизма ассерций [4, 5]. Для этого необходимо синтезировать транзакционные графы программных модулей, выполнить анализ их тестопригодности, найти критические точки мониторинга, установить в них ассерционные операторы. Все упомянутое выше дает возможность уменьшить время отладки программного проекта в среднем на 40% и повысить выход годной продукции на 3-4%.

2. Инфраструктура процесса верификации проекта

Используется уравнение обнаружения ошибок или дефектов на уровне системы или программных компонентов (T, A – тестовые и ассерционные воздействия с ожидаемыми реакциями; P, S, F – спецификация, HDL-модель функциональное покрытие оценки полноты теста для верификации проектируемого изделия): $T \oplus S = L, (T, A) \oplus (P, S, F) = L$. Для более точного понимания соотношений между ключевыми понятиями вводятся определения [5, 6]. Верификация – процесс анализа программных компонентов для определения правильности преобразований входного описания на очередной стадии проектирования. Валидация – определение работоспособности программных компонентов путем проверки соответствия требованиям спецификации на каждой стадии проектирования.

Ассерция – есть инверсное HDL-высказывание системного уровня, предназначенное для раннего определения ошибок проектирования относительно требований спецификации при моделировании проекта на тестовых воздействиях до и после выполнения синтеза. Предельное число ассерций есть диверсная модель проекта, на практике число ассерционных операторов составляет не более 5% HDL-кода проекта. Существует практически стандарт использования ассерций в среде верификации, представленной на рис. 2.

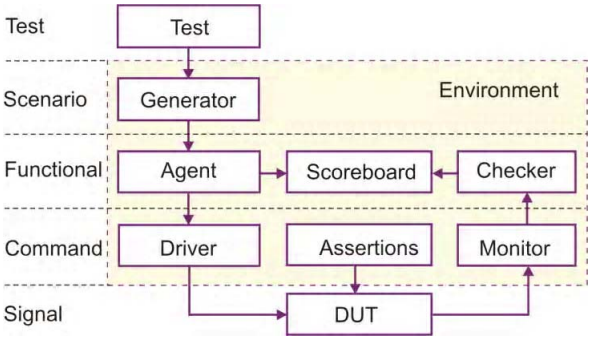


Рис. 2. Иерархическая среда верификации

Компоненты каждого уровня иерархии представляют собой транзакторы. Тестовый уровень иерархии состоит из слоев: 1) Test – компонент верхнего уровня, задает ограничения генератору последовательностей и конфигурирует режимы моделирования при каждом запуске. 2) Уровень сценариев содержит генератор (generator), который создает последовательности с ограничениями, полученными от test, а также все сценарии по псевдослучай-

ной генерации тестов. 3) Функциональный уровень формирует структуры данных для их обработки и передачи на командный (command) слой. Здесь проверяется правильность данных, выдаваемых DUT, с помощью компонентов: агент (agent), счетчик (scoreboard) и анализатор (checker).

Агент используется для обработки данных, передаваемых драйверу (driver). Счетчик верификационной среды содержит высокоуровневую «золотую» модель проекта. Он получает одинаковые с DUT тестовые последовательности и определяет корректную реакцию DUT. Анализатор сравнивает выходную информацию из DUT, доставленную посредством монитора (monitor) с ожидаемой эталонной реакцией, полученной от счетчика. 4) Командный уровень обеспечивает связь с DUT. Драйвер управляет значениями на входах DUT. Монитор следит за выходной информацией из DUT и отправляет ее анализатору. Внешние ассерции используются как составляющая часть проекта командного уровня. 5) Сигнальный (signal) уровень содержит верифицируемое устройство и интерфейс входных, выходных сигналов. Ассерции – блоки, добавляемые в исходный код проекта для наблюдения и управления поведением модели проекта. Они могут быть представлены операторами if для сообщений об ошибках, проявляющихся в процессе тестирования DUT. Ассерции создаются разработчиком или могут быть взяты из существующих библиотек для проверки типовых функций. Фирма Synopsys поставляет систему моделирования VCS с библиотекой SystemVerilog-ассерций. Одно из направлений – это обеспечение полной поддержки актуального стандарта SystemVerilog'2009. Инфраструктура интегрирует опыт и технологии всех ведущих компаний планеты в области тестирования и верификации цифровых проектов. Здесь основные компоненты: семейство языковых средств, ориентированных на ввод проекта; компиляторы с языков описания аппаратуры; средства отладки проекта; ориентированные на пользователя средства управления проектом и компоненты визуализации результатов моделирования.

Иначе, ассерция – предикат системного уровня, определяющий некорректности процесса проектирования относительно требований спецификации. Механизм ассерций совместно с HDL-моделью, тестом и функциональным покрытием представляет собой структуру, на которой можно формировать алгоритмы диагностирования дефектов или принятия решений о существовании ошибки в программном компоненте.

Аналитическая модель описания и решения логических уравнений для верификации HDL-кода представлена системой предикатов:

$$
\begin{aligned}
&P(m,A) = Q(m,A) = [0{,}0-1{,}0]; \qquad Q(m,A) = \frac{1}{3}[d(m,A) + \mu(m \in A) + \mu(A \in m)];\\
&P(m,A) = 1 \leftarrow m = A; \qquad Q(m,A) = \{Q_1, Q_2, ..., Q_i, ..., Q_n\};\\
&P(m,A) = \max_i Q_i(m,A); \qquad A = (A_1, A_2, ..., A_i, ..., A_m);\\
&P(m,A) = m \Delta (A_1, A_2, ..., A_i, ..., A_m); \qquad \Delta = \{\wedge, \vee, \neg\};\\
&A_i = (A_{i1}, A_{i2}, ..., A_{ij}, ..., A_{is}); \qquad A_{ij} = (A_{ij1}, A_{ij2}, ..., A_{ijr}, ..., A_{msq});\\
&m = (m_1, m_2, ..., m_r, ..., m_q); \qquad P(m,A) = m \wedge \{A_1, A_2, .., A_i, ..., A_m\} = p;\\
&m \wedge (A_1 \vee A_2 \vee ... \vee A_i \vee ... \vee A_m) = \max_i Q_i(m,A), \text{ (GRAF)};\\
&p = \{p_1, p_2, ..., p_r, ..., p_n\}; \qquad p = m \wedge (A_1 \vee A_2 \vee ... \vee A_i \vee ... \vee A_m);\\
&p_r(\in p) = m \underset{i=\overline{1,m}}{\overset{j=\overline{1,s}}{\wedge}} A_{ij} \leftarrow \begin{cases} \forall i \exists j \, (m \wedge A_{ij} \neq \varnothing); \\ \exists i \forall j (m \wedge A_{ij} = \varnothing) \wedge d(m, A_{ij}) = \max; \end{cases}\\
&m \wedge A_{ij} = m_r \overset{q}{\underset{r=1}{\wedge}} A_{ijr} = \begin{cases} m \leftarrow m_r \overset{q}{\underset{r=1}{\wedge}} A_{ijr} = m_r; \\ A_{ij} \leftarrow m_r \overset{q}{\underset{r=1}{\wedge}} A_{ijr} = A_{ijr}; \\ m \wedge A_{ij} \leftarrow m_r \overset{q}{\underset{r=1}{\wedge}} A_{ijr} = m_r \vee A_{ijr}; \\ \varnothing \leftarrow \exists (m_r \overset{q}{\underset{r=1}{\wedge}} A_{ijr} = \varnothing); \end{cases} \qquad (1)\\
&P(m,A) = p_r(\in p) \leftarrow Q_i(m, p_r) = \max, \, r = \overline{1,n}.
\end{aligned}
$$

Здесь определены предикаты: 1) $P(m,A)=Q(m,A)$ – задает аналитическую модель вычислительного процесса в виде предиката, определенного интегральным критерием принадлежности вектора экспериментальной проверки множеству технических состояний, одно из которых есть исправное, в интервале $Q(m,A)=[0,0-1,0]$, на совокупности введенных операций: объединение, пересечение и дополнение; 2) граф-упорядоченная совокупность $A=(A_1,A_2,...,A_i,...,A_m)$ взаимодействующих таблиц неисправностей программных компонентов, которые объединяются на верхнем уровне обобщенным графом целевой функциональности; 3) упорядоченная совокупность вектор-строк таблицы неисправностей каждого программного компонента $A_i=(A_{i1},A_{i2},...,A_{ij},...,A_{is})$, каждая из которых $A_{ij}=(A_{ij1},A_{ij2},...,A_{ijr},...,A_{msq})$ принимает значения из троичного алфавита $\{0,1,x\}$. Предикат $A_i=(A_{i1},A_{i2},...,A_{ij},...,A_{is})=1$ представлен совокупностью векторов, формирующих многозначную таблицу неисправностей. Вектор $A_{ij}=(A_{ij1},A_{ij2},...,A_{ijr},...,A_{msq})$ определяет собой решение – совокупность неисправных или ошибочных программных модулей, проверяемых тестом $T_{ij}\in T$, где каждая переменная задается в троичном алфавите $A_{ijr}\in\{0,1,x\}=\beta$. Взаимодействие $P(m,A)$ входного вектора запроса (экспериментальной проверки) $m=(m_1,m_2,...,m_r,...,m_q)$ с графом таблиц неисправностей $A=(A_1,A_2,...,A_i,...,A_m)$ формирует множество решений $p=\{p_1,p_2,...,p_r,...,p_n\}$, каждое из которых имеет интегральную оценку качества $Q(m,A)=\{Q_1,Q_2,...,Q_i,...,Q_n\}$. Выбор лучшего из них осуществляется на основе критерия:

$$P(m,A)=p_r(\in p)\leftarrow Q_i(m,p_r)=\max,\ r=\overline{1,n}.$$

Критерий качества выбранного решения определяется в виде следующих аддитивных оценок:

$$\begin{aligned}
&Q=\frac{1}{3}[d(m,A)+\mu(m\in A)+\mu(A\in m)],\\
&d(m,A)=\frac{1}{n}\left(n-\left|m_i\bigcap_{i=1}^{n}A_i=\varnothing\right|\right);\\
&\mu(m\in A)=2^{|m\cap A|-|A|}\leftarrow|m\cap A|=\left|m_i\bigcap_{i=1}^{n}A_i=x\right|\ \&\ |A|=\left|\bigcup_{i=1}^{n}A_i=x\right|;\\
&\mu(A\in m)=2^{|m\cap A|-|m|}\leftarrow|m\cap A|=\left|m_i\bigcap_{i=1}^{n}A_i=x\right|\ \&\ |m|=\left|\bigcup_{i=1}^{n}m_i=x\right|.
\end{aligned}\qquad(2)$$

Интегральная метрика для оценивания качества запроса есть функция качества (выбора) взаимодействия векторов $m\cap A$, которая определяется средней суммой трех нормированных параметров: кодовое расстояние $d(m,A)$, функция принадлежности $\mu(m\in A)$ и эффективность использования входного запроса – функция принадлежности $\mu(A\in m)$. Нормирование параметров, входящих в метрику, позволяет оценивать уровень взаимодействия вектора экспериментальной проверки и таблиц неисправностей в интервале [0,1]. Если зафиксировано предельное максимальное значение каждого параметра, равное 1, то векторы равны между собой, что соответствует точному диагнозу. Минимальная оценка, Q = 0, фиксируется в случае полного несовпадения векторов по всем n координатам.

Инфраструктура – совокупность моделей, методов и средств описания, анализа и синтеза структур данных для решения функциональных задач.

Функционирование инфраструктуры диагностирования имеет интерфейс, который представлен на рис. 3. При подаче на вход вектора экспериментальной проверки m, маскированного двоичным вектором X, инфраструктура должна сформировать на выходе вектор A, максимально непротиворечивый запросу m, а также оценку качества полученного решения в виде функции $Q=f(m,A);\ A=g(m,A)$.

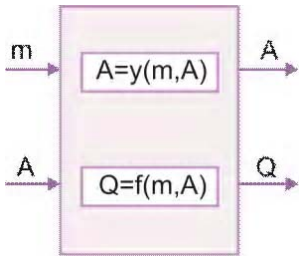


Рис. 3. Интерфейс системы верификации

Данная модель определяет индустриальный стандарт верификации программных продуктов и HDL-моделей, которого придерживаются все компании, которые входят в топ 200 мировых лидеров, формирующих индекс NASDAQ (2308,42 пункта – 04.01.2010) на Уолл-стрит. Поэтому очень важно все научные исследования ориентировать на имплементацию в существующие стандарты, маршруты и технологии мировых лидеров, что дает возможность выходить с практическими предложениями на рынок EDA. Что касается инфраструктуры сервисного обслуживания SoC вейвлет-преобразования, то она, несущественно отличаясь от приведенных моделей двухуровневой иерархией, имеет вид, представленный на рис. 4.

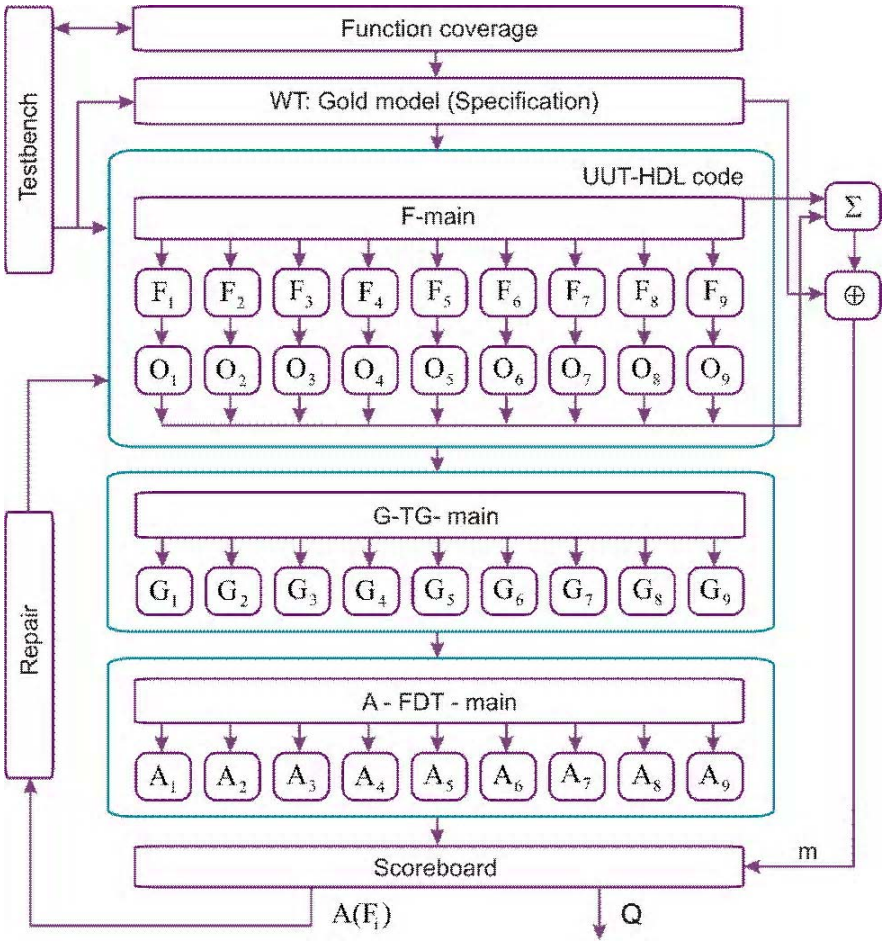


Рис. 4. Инфраструктура верификации WT SoC проекта

Здесь по спецификации строится модель проекта (9 программных модулей и одна основная программа), которая записывается в виде HDL-кода. Затем по модели строятся транзакционные графы, по которым находятся критические точки для мониторинга основного модуля и подпрограмм с помощью ассерционных операторов $O_i \in O$. Далее по тран-

закционным графам строятся таблицы (А) несправных модулей, которые используются для поиска и исправления ошибок. Качество теста гарантируется полнотой покрытия функциональностей, которая влияет на глубину диагностирования и качество проекта в целом. Система верификации и диагностирования содержит следующие компоненты: 1. Модуль ввода ассерций, представленный транслятором языка описания ассерций или проекта. 2. Внутренняя модель описания и обработки выражений в виде логико-временных отношений, алгоритмов диагностирования или директив верификации. 3. Модуль анализа ассерций для проверки заданных формальных логико-временных ограничений в процессе моделирования. 4. Блок синтеза квазиоптимальной модели анализа ассерций по имеющимся языковым конструкциям. 5. Структуры данных, обеспечивающие взаимодействие и совместимость между множеством ассерций и функциональностью в процессе компиляции и моделирования.

3. Анализ тестопригодности графов HDL-моделей

Граф HDL-модели создается путем семантического анализа кода для выявления: 1) всех вершин, определяемых как приемники и источники транзакций данных (регистр, счетчик, память или массивы, вход-выходные шины, векторы, логические или арифметические переменные), которые упоминаются в строках кода; 2) направления приема или передачи данных, формирующих множество дуг между вершинами. Дуги могут быть отмечены числом транзакций. Граф создает условия для оценки его тестопригодности, а значит и для определения качества структуризации кода программного продукта. Транзакционный граф записывается в виде алгебраической (логической) формы, что дает возможность не только получить все решения относительно достижимости вершины, но и оценивать тестопригодность каждой вершины и графа в целом.

Ассерционная избыточность HDL-модели должна быть эффективной в целях повышения тестопригодности структуры разработанного или написанного кода. Качество программного продукта на основе анализа графа транзакций (TG – Transaction Graph) определяется выражением:

$$Q = \frac{1}{Z}(U \times N) = \frac{Z(S)}{Z(S)+Z(F)+Z(T)+Z(A)} \times \left(\frac{1}{n}\sum_{i=1}^{n} U_i\right) \times \left(\frac{1}{n}\sum_{i=1}^{n} N_i\right);$$

$$U_i = \frac{1}{T}\sum_{j=1}^{x_i} T_j^i \times \frac{1}{d_i^x \vee t_i^x}; \; N_i = \frac{1}{T}\sum_{j=1}^{y_i} T_j^i \times \frac{1}{d_i^y \vee t_i^y}. \qquad (3)$$

Качество зависит от управляемости U, наблюдаемости N, а также дополнительных затрат (в знаменателе функции Q) для формирования функциональной корзины, теста, ассерций. Управляемость (наблюдаемость) есть функция от числа операторов, входящих в вершину (исходящих из вершины) транзакционного графа, а также от структурной глубины рассматриваемой вершины относительно входной (выходной) шины

Логические функции управляемости и наблюдаемости каждой вершины транзакционного графа записываются в форме конъюнкции дизъюнктивных термов:

$$U_i = \frac{1}{T}\left(\bigwedge_{j=1}^{x_1} T_{1j}^x\right)\ldots\left(\bigwedge_{j=1}^{x_{i-1}} T_{i-1j}^x\right)\left(\bigvee_{i=1}^{x_i} T_{ij}^x\right) \quad N_i = \frac{1}{T}\bigwedge_{j=1}^{x_i} T_j^i\left(\bigvee_{i=1}^{x_i} T_j^i d_i^x \vee t_i^x\right). \qquad (4)$$

Число дизъюнктивных термов соответствует количеству входящих в вершину дуг, конъюнкций – структурной глубине компонента. Если управляемость и наблюдаемость вершины графа вычисляется на основании алгебраической формы представления графа [3], то формулы подсчета критериев имеют следующий вид:

$$U_i = \frac{1}{t_{max}^x \times n_t^x} \times \sum_{i=1}^{n_t^x}\sum_{j=1}^{k_i^x}\left(t_{max}^x - \left|t_{ij}^x\right| + 1\right); \quad N_i = \frac{1}{t_{max}^y \times n_t^y} \times \sum_{i=1}^{n_t^y}\sum_{j=1}^{k_i^y}\left(t_{max}^y - \left|t_{ij}^y\right| + 1\right), \qquad (5)$$

где $t_{max}^x, n_t^x, k_i^x, \left|t_{ij}^x\right|$ – для критерия управляемости конъюнктивный терм максимальной длины; количество термов в логической функции управляемости; количество транзакций в

текущем терме функции; мощность рассматриваемой транзакции. Такие же обозначения используются и для наблюдаемости – $t^{y}_{max}, n^{y}_{t}, k^{y}_{i}, \left|t^{y}_{ij}\right|$.

Анализ тестопригодности графа управления. Учитывая, что автоматная модель программного продукта представлена взаимодействием операционного и управляющего автомата (рис. 5 слева), то наряду с моделированием транзакционного графа необходимо иметь возможность анализировать тестопригодность граф-схемы алгоритма управления (ГСА).

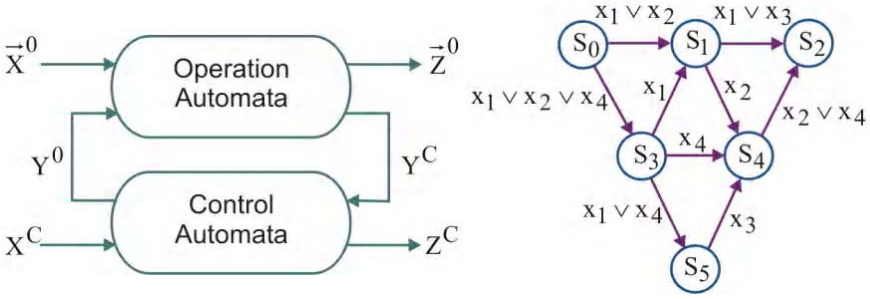


Рис. 5.Автоматная модель HDL-программы

Предлагается представить ГСА в виде содержательного графа управления (СГУ), который является подобным транзакционному графу. Здесь вершины есть операции программного кода, а дуги представляют условия перехода из одной вершины в другую для выполнения команды, обозначенной вершиной-стоком. Следовательно, для СГУ можно использовать процедуры, ранее разработанные для подсчета критериев тестопригодности транзакционного графа в части управляемости и наблюдаемости. Примером содержательного графа может служить рис. 5 (справа), имеющий 6 вершин и 9 дуг.

Подсчет управляемостей графа, реализующего алгоритм управления и представленного на рис. 5, имеет следующий вид:

$$S_3 = T_3^3;\ S_1 = T_3^3T_4^1 \vee T_1^2;\ \ S_2 = T_3^3T_4^1T_6^1T_7^2 \vee T_1^2T_2^2 \vee T_3^3T_5^1T_7^2 \vee T_3^3T_4^1T_7^2 \vee T_3^3T_8^2T_9^1T_7^2;$$

$$S_4 = T_3^3T_4^1T_6^1 \vee T_3^3T_5^1 \vee T_3^3T_8^2T_9^1;\ \ S_5 = T_3^3T_8^2.$$

Подсчет наблюдаемостей графа, представленного на рис. 5, содержит следующие выражения:

$$S_3 = T_7^2T_5^1 \vee T_7^2T_9^1T_8^2 \vee T_7^2T_6^1T_4^1 \vee T_2^2T_4^1;\ \ S_1 = T_2^2 \vee T_7^2T_6^1;$$

$$S_0 = T_2^2T_1^2 \vee T_7^2T_6^1T_4^1T_3^3 \vee T_7^2T_5^1T_3^3 \vee T_7^2T_5^1T_3^3 \vee T_7^2T_9^1T_8^2T_3^3 \vee T_2^2T_4^1T_3^3;\ S_4 = T_7^2;\ \ S_5 = T_7^2T_9^1.$$

Тестопригодность графа проекта и каждого компонента подсчитывается как произведение управляемости и наблюдаемости:

$$Q = \frac{1}{Z}(U \times N) = (\frac{1}{n}\sum_{i=1}^{n} U_i) \times (\frac{1}{n}\sum_{i=1}^{n} N_i). \qquad (6)$$

4. Диагностирование программных модулей WT SoC

Ниже представлены компоненты инфраструктуры для верификации и диагностического обслуживания двух модулей WT SoC: 1) транзакционные графы и построенные на их основе логические функции тестопригодности, управляемости и наблюдаемости HDL-моделей; 2) таблицы и графики оценивания тестопригодности всех вершин и всех графов программного HDL-кода; 3) таблицы неисправностей программных модулей WT SoC проекта для поиска дефектов на основе логических операций; 4) выполнены диагностические эксперименты, которые подтверждают эффективность и валидность предложенных моделей и метода поиска дефектов.

Системный граф совокупного программного продукта – HDL-кода, реализующего вейвлет-преобразование, представлен на рис. 6.

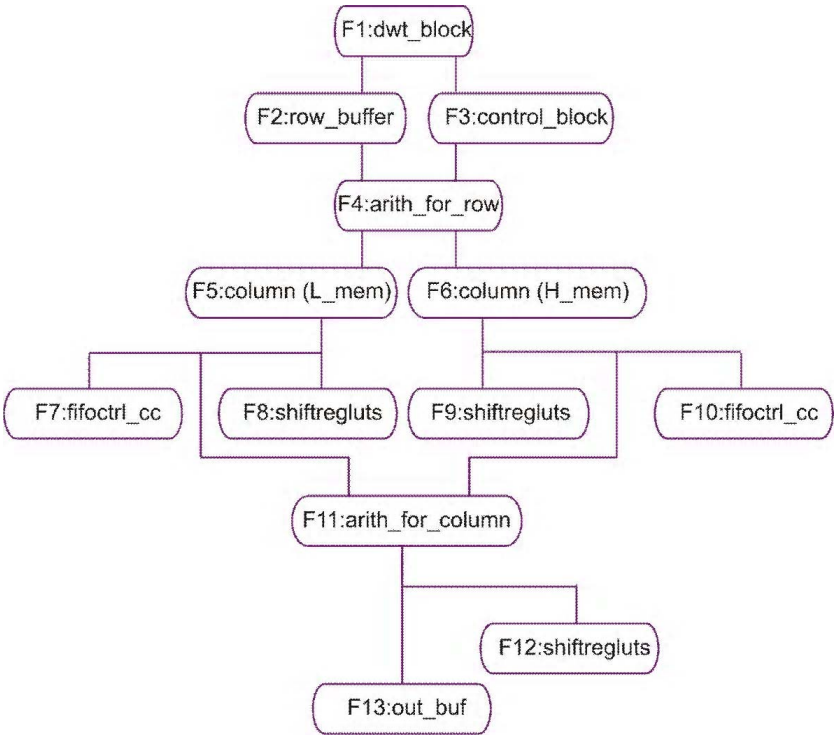


Рис. 6. Граф HDL-кода вейвлет-преобразования

Структурный анализа графа для определения тестопригодности дает возможность создать транзакционный граф, представленный на рис. 7.

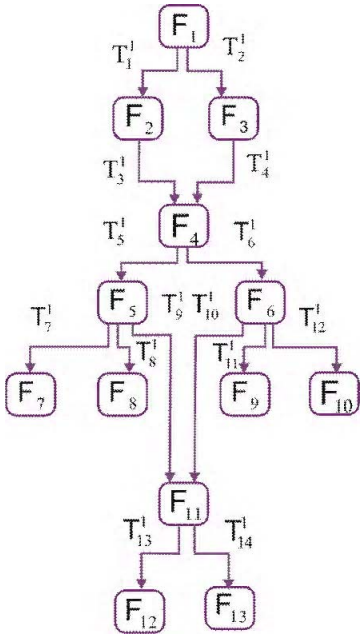


Рис. 7. Транзакционный граф main-HDL-кода

На основе транзакционного графа строятся логические функции управляемости и наблюдаемости, представленные ниже.

1) Управляемость входов:

$$F_2 = T_1^1;\ F_3 = T_2^1;\ F_4 = T_1T_3 \vee T_4T_2;\ F_5 = T_1^1T_5^1T_3^1 \vee T_5^1T_4^1T_2^1;\ F_6 = T_6^1T_1^1T_3^1 \vee T_6^1T_4^1T_5^1;$$

$F_7 = T_7^1T_1^1T_5^1T_3^1 \vee T_7^1T_5^1T_4^1T_2^1$; $F_8 = T_8^1T_1^1T_5^1T_3^1 \vee T_8^1T_5^1T_4^1T_2^1$; $F_9 = T_{11}^1T_6^1T_1^1T_3^1 \vee T_{11}^1T_6^1T_4^1T_5^1$;

$F_{10} = T_{12}^1T_6^1T_1^1T_3^1 \vee T_{12}^1T_6^1T_4^1T_5^1$; $F_{11} = T_{10}^1T_6^1T_1^1T_3^1 \vee T_{10}^1T_6^1T_4^1T_5^1 \vee T_9^1T_5^1T_1^1T_3^1 \vee T_9^1T_5^1T_4^1T_2^1$;

$F_{12} = T_{13}^1T_{10}^1T_6^1T_1^1T_3^1 \vee T_{13}^1T_{10}^1T_6^1T_4^1T_5^1T_{13}^1 \vee T_{13}^1T_9^1T_5^1T_1^1T_3^1 \vee T_{13}^1T_9^1T_5^1T_4^1T_2^1$;

$F_{13} = T_{14}^1T_{10}^1T_6^1T_1^1T_3^1 \vee T_{14}^1T_{10}^1T_6^1T_4^1T_5^1 \vee T_{14}^1T_9^1T_5^1T_1^1T_3^1 \vee T_{14}^1T_9^1T_5^1T_4^1T_2^1$.

2) Наблюдаемость выходов:

$F_{11} = T_{13}^1 \vee T_{14}^1$; $F_5 = T_7^1 \vee T_8^1 \vee T_9^1T_{13}^1 \vee T_9^1T_{14}^1$; $F_6 = T_{11}^1 \vee T_{12}^1 \vee T_{10}^1T_{13}^1 \vee T_{10}^1T_{14}^1$;

$F_4 = T_5^1T_7^1 \vee T_5^1T_8^1 \vee T_5^1T_9^1T_{13}^1 \vee T_5^1T_9^1T_{14}^1 \vee T_6^1T_{11}^1 \vee T_6^1T_{12}^1 \vee T_6^1T_{10}^1T_{13}^1 \vee T_6^1T_{10}^1T_{14}^1$;

$F_3 = T_4^1T_5^1T_7^1 \vee T_4^1T_5^1T_8^1 \vee T_4^1T_5^1T_9^1T_{13}^1 \vee T_4^1T_5^1T_9^1T_{14}^1 \vee T_4^1T_6^1T_{11}^1 \vee T_4^1T_6^1T_{12}^1 \vee T_4^1T_6^1T_{10}^1T_{13}^1 \vee T_4^1T_6^1T_{10}^1T_{14}^1$;

$F_2 = T_3^1T_5^1T_7^1 \vee T_3^1T_5^1T_8^1 \vee T_3^1T_5^1T_9^1T_{13}^1 \vee T_3^1T_5^1T_9^1T_{14}^1 \vee T_3^1T_6^1T_{11}^1 \vee T_3^1T_6^1T_{12}^1 \vee T_3^1T_6^1T_{10}^1T_{13}^1 \vee T_3^1T_6^1T_{10}^1T_{14}^1$;

$F_1 = T_2^1T_4^1T_5^1T_7^1 \vee T_2^1T_4^1T_5^1T_8^1 \vee T_2^1T_4^1T_5^1T_9^1T_{13}^1 \vee T_2^1T_4^1T_5^1T_9^1T_{14}^1 \vee T_2^1T_4^1T_6^1T_{11}^1 \vee T_2^1T_4^1T_6^1T_{12}^1 \vee T_2^1T_4^1T_6^1T_{10}^1T_{13}^1 \vee$
$\vee T_2^1T_4^1T_6^1T_{10}^1T_{14}^1 \vee T_1^1T_3^1T_5^1T_7^1 \vee T_1^1T_3^1T_5^1T_8^1 \vee T_1^1T_3^1T_5^1T_9^1T_{13}^1 \vee T_1^1T_3^1T_5^1T_9^1T_{14}^1 \vee T_1^1T_3^1T_6^1T_{11}^1 \vee T_1^1T_3^1T_6^1T_{12}^1 \vee$
$\vee T_1^1T_3^1T_6^1T_{10}^1T_{13}^1 \vee T_1^1T_3^1T_6^1T_{10}^1T_{14}^1$.

Логические функции управляемости и наблюдаемости обрабатываются на основе использования формулы (3), что дает возможность определить минимальные оценки вершин в части наблюдаемости, для которых строятся ассерции как дополнительные точки мониторинга программного кода. В табл. 1 такими критическими точками являются F_1, F_2, F_3.

В соответствии со значениями параметров тестопригодности на рис. 8 представлены графики управляемости, наблюдаемости всех вершин транзакционного графа главной программы, которая визуально иллюстрирует критические точки F_1, F_2, F_3 по наблюдаемости, интересные для установки в них ассерций.

Интегральные оценки качества тестопригодности, не учитывающие и учитывающие мультипликативность транзакционных дуг всего программного продукта, представлены в табл. 3. Общая оценка тестопригодностей двух видов графа (мульти- и одиночные дуги) имеет вид: $Q(F_0) = \{Q(F_0^S) = 0{,}33766; Q(F_0^M) = 0{,}61525\}$.

Данные оценки являются существенными при сравнении нескольких вариантов проекта, сделанных различными разработчиками. Естественно, что проект, имеющий большую интегральную оценку тестопригодности, принимается к реализации.

Таблица 1. Тестопригодность графа

C	U(S)	U(M)	N(S,N)	Q(S)	Q(M)
F1	1,00	1,00	0,3	0,3	0,65
F2	1,00	1,00	0,375	0,375	0,6875
F3	1,00	1,00	0,375	0,375	0,6875
F4	0,5	0,5	0,5	0,25	0,5
F5	0,33	0,33	0,75	0,2475	0,54
F6	0,33	0,33	0,75	0,2475	0,54
F7	0,25	0,25	1,00	0,25	0,625
F8	0,25	0,25	1,00	0,25	0,625
F9	0,25	0,25	1,00	0,25	0,625
F10	0,25	0,25	1,00	0,25	0,625
F11	0,25	0,25	1,00	0,25	0,625
F12	0,25	0,25	1,00	0,25	0,625
F13	0,25	0,25	1,00	0,25	0,625
Q=				0,15731	0,32538

Как результат моделирования неисправностей на тесте, который проверяет все ошибочные программные модули, строится табл. 2. Используя данную таблицу и формулы, определяющие логический метод диагностирования, можно определить дефектные компоненты программного кода с наперед заданной глубиной диагностирования.

Здесь векторы m1 и m2 определяют результат диагностического эксперимента, выполненного по схе-

ме рис. 4. Аналитическая запись процесса диагностирования основана на формулах (1) и имеет следующий вид:

$$A = m \wedge (A_1 \vee A_2 \vee ... \vee A_i \vee ... \vee A_m) \approx$$

$$F(m_1) = m_1 \wedge (F_1 \vee F_2 \vee F_3 \vee F_4 \vee F_5 \vee F_6 \vee F_7 \vee F_8 \vee F_9 \vee F_{10} \vee F_{11} \vee F_{12} \vee F_{13}) = F_{12};$$

$$F(m_2) = m_2 \wedge (F_1 \vee F_2 \vee F_3 \vee F_4 \vee F_5 \vee F_6 \vee F_7 \vee F_8 \vee F_9 \vee F_{10} \vee F_{11} \vee F_{12} \vee F_{13}) = F_1 \vee F_3 \vee F_4;$$

$$Q_1(m_1, F_{12}) = 1;\ Q_2[m_2, (F_1 \vee F_3 \vee F_4)] = 0{,}75;$$

Таблица 2. Неисправности, обнаруживаемые тестом

Test	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	m1	m2
t1	1	1	0	1	1	1	1	0	0	0	0	0	0	0	X
t2	1	1	0	1	0	1	0	0	1	0	0	0	0	0	X
t3	1	0	1	1	1	0	0	0	0	0	1	1	0	1	1
t4	1	0	1	1	0	1	0	0	0	0	1	0	1	0	1
t5	1	0	1	1	0	1	0	0	0	1	0	0	0	0	1

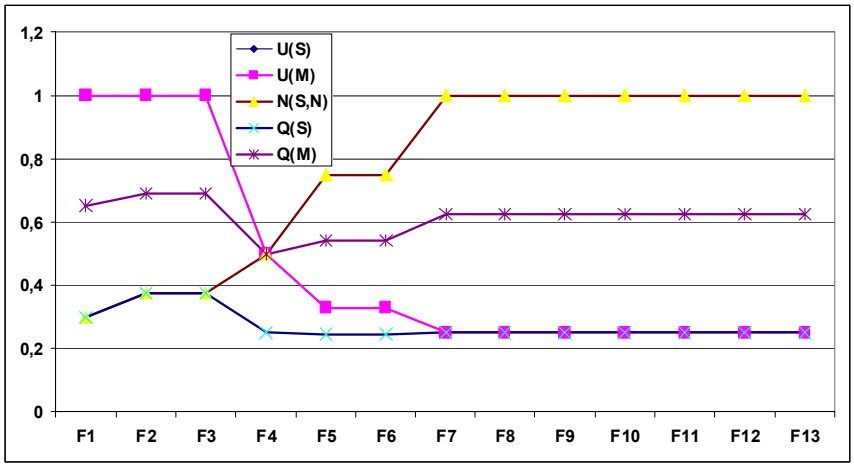


Рис. 8. Графики параметров тестопригодности

Здесь, в первом случае критерий качества, вычисляемый в соответствии с выражением (2), равен 1, что свидетельствует о присутствии данного дефекта в программном коде. Во втором случае, когда критерий не равен 1, можно говорить о возможном присутствии любого сочетания из трех неисправностей.

Далее представлено диагностическое обеспечение одного из программных модулей нижнего уровня, который является компонентом Row_buffer транзакционного графа main-программы (рис. 9). Для указанного компонента выполняются все вычислительные процедуры по аналогии с обработкой модуля main.

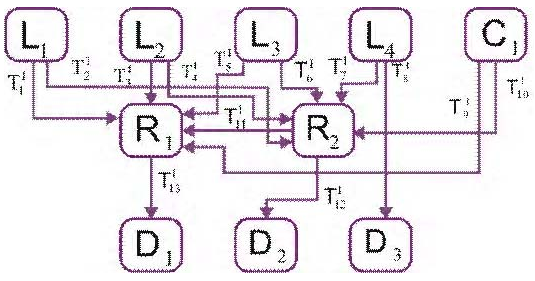


Рис. 9. Компонент Row_buffer транзакционного графа

На основе транзакционного графа строятся логические функции управляемости и наблюдаемости, представленные ниже.

Таблица 3. Тестопригодность структуры

C	Q(S)	Q(M)
F1	0,181	0,49009
F2	0,548	0,774
F3	0,20224	0,5294
F4	0,15663	0,44615
F5	0,25758	0,58531
F6	0,25758	0,58531
F7	0,2296	0,52823
F8	0,54	0,77
F9	0,54	0,77
F10	0,2296	0,52823
F11	0,15663	0,44615
F12	0,54	0,77
F13	0,55067	0,77533
Q(F0)=	0,33766	0,61525

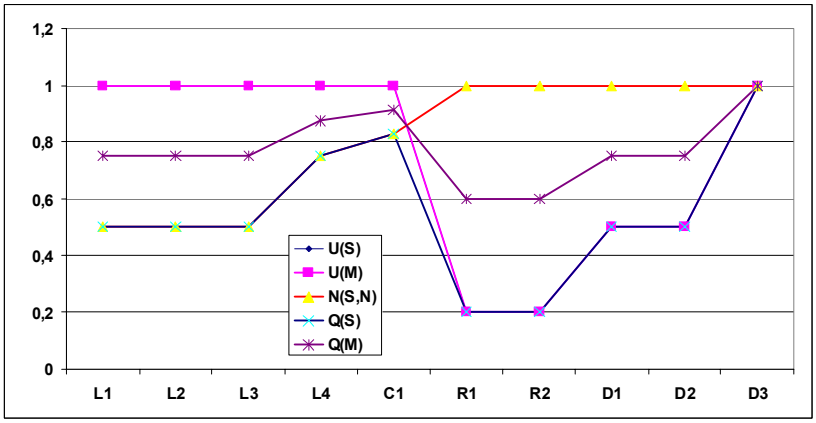


Рис. 10. Графики параметров тестопригодности Row_buffer

Как результат моделирования частей подпрограммы Row_buffer на тесте строится табл. 4 для неисправностей блоков.

Таблица 4. Неисправности компонента Row_buffer на тесте

Test	L1	L2	L3	L4	C1	R1	R2	D1	D2	m1	m2	m3
$A_1(t_1)$	1	0	0	0	0	1	0	1	0	1	X	1
$A_2(t_2)$	0	1	0	0	0	0	1	0	1	0	X	1
$A_3(t_3)$	0	0	1	0	0	1	1	1	0	1	0	0
$A_4(t_4)$	0	0	0	1	0	0	1	0	1	0	1	0
$A_5(t_5)$	0	0	0	0	1	1	1	1	0	1	0	0

Метод логического умножения вектор-столбцов. Он основан на применении операции логического умножения или конъюнкции вектора экспериментальной проверки на столбцы таблицы неисправностей $m_i \wedge (F_1 \vee F_2 \vee ... \vee F_j \vee ... \vee F_n)$ и подсчете качества взаимодействия векторов $Q_j(m_i \wedge F_j)$ в целях выбора лучшего из них. Предикатная запись процесса получения решения в виде совокупности ошибок, присутствующих в HDL-коде, представлена ниже:

$$P(m_i, F) = m_i \wedge (F_1 \vee F_2 \vee ... \vee F_j \vee ... \vee F_n) = \max Q_j(m_i \wedge F_j);$$

$$F^s = (F_j \in F) \leftarrow Q_j(m_i \wedge F_j) = \{1 \vee \max\},\ j = \overline{1,n};$$

$$F_i^m = (F_j \vee F_r) \leftarrow m_i \wedge (F_j \bigvee_{j=\overline{1,n-1}}^{r=\overline{j+1,n}} F_r) = \{1 \vee \max Q_{jr}[m_i \wedge (F_j \vee F_r)]\} \leftarrow$$

$$\leftarrow \forall j[Q_j(m_i \wedge F_j)] \neq 1;$$

$$F_i^m = \forall F_j \leftarrow (m_i \bigwedge_{j=1}^{n^*} F_j = F_j) \,\&\, (\bigvee_{j=1}^{n^*} F_j = m_i).$$

При этом столбец $F_j \in F$ фактически идентифицирует метрику поведения неисправности или дефектного блока на тестовых последовательностях. Достоинство метода – выбор всегда лучшего решения из всех возможных как для одиночных, так и для кратных дефектов. В последнем случае, если одиночный дефект не идентифицируется оценкой, равной 1, выполняется дизъюнкция таких вектор-строк (главное отличие метода от существующих технологий), которая сформирует оценку качества, равную 1 или максимально близкую к единице: $Q_{jr}(m_i \wedge (F_j \vee F_r))] = 1 \vee \max$. По существу, в список кратных дефектов включаются такие одиночные, которые при логическом умножении на вектор экспериментальной проверки дают результат в виде рассматриваемого вектор-столбца.

Используя таблицу и процедуры диагностирования (1), можно определить дефектные компоненты программного кода модуля Row_buffer методом логического умножения вектор-столбцов таблицы истинности на вектор экспериментальной проверки. Здесь векторы m1 и m2 формируют результат диагностического эксперимента, выполняемый по ранее определенной технологии (см. рис. 4). Результат диагностирования, с учетом (1), имеет следующий вид:

$$F(m_1) = m_1 \wedge (L_1 \vee L_2 \vee L_3 \vee L_4 \vee C_1 \vee R_1 \vee$$
$$\vee R_2 \vee D_1 \vee D_2) = R_1 \vee D_1;$$
$$F(m_2) = m_2 \wedge (L_1 \vee L_2 \vee L_3 \vee L_4 \vee C_1 \vee R_1 \vee$$
$$\vee R_2 \vee D_1 \vee D_2) = L_4 \vee D_2;$$
$$Q_1[m_1, (R_1 \vee D_1)] = 1;$$
$$Q_2[m_2, (L_4 \vee D_2)] = 0{,}75;$$

В случае неоднозначности (множественности) диагноза или определения места дефекта используется ассерция, которая осуществляет мониторинг блока, определенного в качестве дефектного. Для установления точного диагноза необходимо в худшем случае выполнить n-1 проверку, где n – мощность обнаруженных дефектных блоков, при условии, что в программном модуле существует один ошибочный блок.

Метод логического умножения вектор-строк. Другая стратегия определения ошибок программного кода по таблице неисправностей связана с анализом ее строк: 1) Вычисление произведения конъюнкции единичных строк ($A_i \leftarrow m_i = 1$) на отрицание дизъюнкции нулевых строк ($A_i \leftarrow m_i = 0$) для одиночных дефектных блоков. 2) Вычисление произведения дизъюнкции единичных строк ($A_i \leftarrow m_i = 1$) на отрицание дизъюнкции нулевых строк ($A_i \leftarrow m_i = 0$) для одиночных дефектных блоков:

$$\begin{cases} F^s = (\bigwedge\limits_{\forall m_i = 1} A_i) \wedge (\overline{\bigvee\limits_{\forall m_i = 0} A_i}); \\ F^m = (\bigvee\limits_{\forall m_i = 1} A_i) \wedge (\overline{\bigvee\limits_{\forall m_i = 0} A_i}); \end{cases} \qquad (7)$$
$$m = (m_1, m_2, ..., m_i, ..., m_n);$$
$$A = (A_1, A_2, ..., A_i, ..., A_n);$$

Выполнение диагностического эксперимента по (7) для вектора экспериментальной проверки $m_1 = (10101)$, заданного в последней таблице неисправностей, дает результат: $F^s(m_1, A) = R_1 \vee D_1$, который не хуже, чем ранее полученный методом логического умножения.

Диагностирование кратных дефектов методом логического умножения столбцов. Если, например, вектору $m_3 = (11000)$ экспериментальной проверки нет соответствующего столбца в табл. 4, то в данном случае вычисляются критерии качества (принадлежности) для всех столбцов таблицы неисправностей:

$$Q(m_3, A) = [Q(m_3, L_1) = 0{,}93;\ Q(m_3, L_2) = 0{,}93;\ Q(m_3, L_3) = 0{,}8;$$
$$Q(m_3, L_4) = 0{,}8;\ Q(m_3, C_1) = 0{,}8;\ Q(m_3, R_1) = 0{,}8;$$
$$Q(m_3, R_2) = 0{,}73;\ Q(m_3, D_1) = 0{,}8;\ Q(m_3, D_2) = 0{,}86].$$

Далее выбираются столбцы, имеющие максимальное значение критерия качества: $F^m = L_1 \vee L_2 \leftarrow [Q(m_3, L_1) = 0{,}93;\ Q(m_3, L_2) = 0{,}93]$. Дизъюнкция данных векторов дает вектор-столбец, равный вектору экспериментальной проверки: $L_1(10000) \vee L_2(01000) = m_3(11000)$. Вывод: в структуре присутствует кратный дефект, составленный из двух одиночных.

Диагностирование кратных дефектов методом логического умножения вектор-строк – сценарий F^m *(7)*. В соответствии с вектором $m_3 = (11000)$ экспериментальной проверки обрабатываются единичные и нулевые строки табл. 4, что приводит к следующему результату:

$$A(m_3^1, A) = A_1(100001010) \vee A_2(010000101) = (110001111);$$

$$A(m_3^0, A) = A_3(001001110) \vee A_4(000100101) \vee A_5(000011110) = (111111111);$$

$$F^m = A(m_3^1, A) \wedge \overline{A(m_3^0, A)} = (110001111) \wedge \overline{(111111111)} = (000000000).$$

В данном случае метод фиксирует отсутствие дефектов, когда на самом деле они однозначно имеются – вектор экспериментальной проверки имеет единичные координаты, что является недостатком метода логического умножения вектор-строк в части диагностирования кратных неисправностей. Однако метод логического умножения вектор-столбцов всегда дает результат, максимально приближенный к реальности.

Таким образом, предложенные аналитические модели и методы, проведенные эксперименты позволяют определять за два цикла логическую семантическую неисправность программного блока или группы операторов: в первом цикле вычисляется неисправный модуль (подпрограмма), во втором – ошибка в некоторой его части, мощность которой регулируется числом ассерционных операторов.

5. Выводы

1. Предложена инфраструктура верификации HDL-модели проекта, ориентированная на ускорение отладки программного кода на основе использования программной избыточности в виде механизма ассерций, который позволяет осуществлять поиск семантических ошибок с заданной глубиной диагностирования.

2. Предложен новый метод логического умножения строк и столбцов таблицы неисправностей для поиска одиночных и кратных дефектов на основе иерархической структуры таблиц дефектов, соответствующей программным модулям цифрового проекта путем нахождения оптимального решения, оцениваемого с помощью интегрального критерия качества, использующего функции принадлежности.

3. Получила дальнейшее развитие модель процесса определения тестопригодности проекта на основе использования графа транзакций, предназначенного для поиска критических точек программного кода для установки ассерционных операторов, повышающих управляемость и наблюдаемость структуры проекта.

4. Разработана инфраструктура верификации и диагностического обслуживания WT SoC: 1) Транзакционные графы и построенные на их основе логические функции тестопригодности, управляемости и наблюдаемости HDL-моделей. 2) Таблицы и графики оценивания тестопригодности всех вершин и всех графов программного HDL-кода. 3) Таблицы неисправностей всех программных модулей WT SoC проекта для поиска дефектов на основе логических операций. 4) Выполнены диагностические, в том числе и на реальных объектах, эксперименты, которые подтверждают эффективность и валидность предложенных моделей и метода поиска дефектов.

5. Предложены технологические мероприятия и рекомендации, ориентированные на улучшение процесса проектирования цифровых систем на кристаллах путем интеграции тестопригодного анализа и механизма ассерций в процедуры синтеза программных и аппаратных продуктов.

6. Показан пример практического использования анализа тестопригодности транзакционного и управляющего графов для реального DSP-проекта и последующего внедрения в критические точки программного кода механизма ассерций в целях поиска и устранения ошибок, что дало возможность найти и исправить некорректные фрагменты кода, необнаруженные авторами программы.

7. Индустриальная значимость предложенных моделей и метода заключается в высокой рыночной привлекательности и заинтересованности технологических компаний в новых решениях верификации программно-аппаратных изделий на системном уровне проектирования для уменьшения time-to-market и повышения выхода годной продукции – yield.

Список литературы: 1. *Daubechies I.* Factoring wavelet transforms into lifting steps / Daubechies I. and Sweldens W. Bell Laboratories. Lucent Technologies, 1996. 368p. **2.** *Daubechies I. and Sweldens W.* Factoring wavelet transforms into lifting schemes / I. Daubechies and W. Sweldens // The J. of Fourier Analysis and Applications. 1998. Vol. 4. P. 247-269. **3.** *Chui C.K.* An Introduction to Wavelets / C.K. Chui. New York – London: Academic Press, 1992. 266 p. **4.** *Taubman David S.* JPEG2000: image compression fundamentals, standards and practice / David S. Taubman, Michael W. Marcellin. Norwell: Kluwer Academic Pulishers, 2002. 774 p. **5.** *Петухов А.П.* Введение в теорию базисов всплесков / Петухов А.П. СПб.: Изд. СПбГТУ, 1999. 131с. **6.** *DeVore R.* Image Compression Trough Wavelet Transform Coding / R. DeVore, W. Jawerth, B. Lucier // IEEE Trans. on Information Theory. 1992. Vol. 39, No. 2. P. 719-746. **7.** *Bergeron J.* Writing Testbenches Using SystemVerilog / J. Bergeron // Springer Science and Business Media, Inc., 2006. 414 p. **8.** Bergeron J. Verification Methodology Manual for System Verilog /J. Bergeron, E. Cerny, A. Hunter, and A. Nightingale. Springer Science and Business Media, 100 Inc., 2006. 510 p. **9.** *Abramovici M.* Digital System Testing and Testable Design / M. Abramovici, M.A. Breuer and A.D. Friedman. Comp. Sc. Press. 1998. 652 p. **10.** *Bayraktaroglu* Ismet The Construction of Optimal Deterministic Partitionings in Scan–Based BIST Fault Diagnosis: Mathematical Foundations and Cost–Effective Implementations / Ismet Bayraktaroglu, Alex Orailoglu // IEEE Transactions on Computers. 2005. P.61–75. **11.** Densmore Douglas A Platform–Based taxonomy for ESL Design / Douglas Densmore, Roberto Passerone, Alberto Sangiovanni–Vincentelli // Design&Test of computers. 2006. P. 359–373. **12.** *DaSilva* Francisco Overview of the IEEE P1500 Standard / Francisco DaSilva, Yervant Zorian, Lee Whetsel, Karim Arabi, Rohit Kapur // ITC International Test Conference. 2003. P. 988–997. **13.** *Rashinkar P.* System–on–chip Verification: Methodology and Techniques / Rashinkar P., Paterson P., Singh L. Kluwer Academic Publishers, 2002. 324 p. **14.** *Zorian Yervant.* What is Infrustructure IP? / Yervant Zorian. // IEEE Design & Test of Computers. 2002. P. 5–7. **15.** *Zorian Yervant* Guest editors' introduction: Design for Yield and reliability / Yervant Zorian, Dmytris Gizopoulos // IEEE Design & Test of Computers. 2004. P. 177–182. **16.** *Zorian Yervant.* Guest Editor's Introduction: Advances in Infrastructure IP // IEEE Design and Test of Computers. 2003. Vol. 20, № 3. P. 49. **17.** *Thatte S.M.* Test generation for microprocessors / S.M. Thatte, J.A. Abraham // IEEE Trans. Comput. 1980. Vol. 29, No 6. P. 429–441. **18.** *Шаршунов С.Г.* Построение тестов микропроцессоров. 1. Общая модель. Проверка обработки данных // Автоматика и телемеханика. 1985. №11. С. 145–155. **19.** *Jerraya A.A.* System Level Synthesis SLS / A.A. Jerraya // TIMA Laboratory. Annual Report, 2002. P. 65–75. **20.** *Ghenassia Frank.* Transaction Level Modeling with SystemC / Frank Ghenassia. TLM Concepts and Applications for Embedded Systems. Published by Springer, 2005. 282 p. **21.** *Foster* Harry Assertion–based design / Harry Foster, Adam Krolnik, David Lacey. Kluwer Academic Publishers. Second edition. Springer, 2005. 392 p. **22.** *Meyer A.S.* Principles of Functional Verification / Meyer A.S. Elsevier Science, 2004. 206 p. **23.** *Хаханов В.И.* Проектирование и тестирование цифровых систем на кристаллах / В.И. Хаханов, Е.И. Литвинова, О.А. Гузь. Харьков: ХНУРЭ, 2009. 484с.

Поступила в редколлегию 02.09.2009

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, профессор кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Побеженко Ирина Александровна, аспирантка кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-421. E-mail: kiu@kture.kharkov.ua.

Василенко Василина Александровна, аспирантка кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-421. E-mail: kiu@kture.kharkov.ua.

Чумаченко Светлана Викторовна, д-р техн. наук, профессор кафедры АПВТ ХНУРЭ. Научные интересы: математическое моделирование и вычислительные методы, методы дискретной оптимизации. Увлечения: спорт, музыка, поэзия, путешествия. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326, e-mail: ri@kture.kharkov.ua.

РЕФЕРАТИ

УДК 621.396.2 : 625.316.2

Спектральний аналіз комбінованих моделей лінійного передбачення негаусових процесів / В.А. Тихонов, Н.В. Кудрявцева // АСУ та прилади автоматики. 2009. Вип. 148. С. 4-7.

Синтезовано комбіновані моделі лінійного передбачення, моделі узагальненої авторегресії-ковзного середнього. Знайдено вирази для параметричних спектральних оцінок комбінованих моделей негаусових випадкових процесів.

Іл.2. Бібліогр.: 5 назв.

UDC 621.396.2 : 625.316.2

Spectral analysis of combined linear prediction model for Non-Gaussian processes. / V.A. Tykhonov, N.V. Kudryavceva // Management Information System and Devices. 2009. N 148. P.4-7.

A combined linear prediction model of a generic autoregression-moving average are synthesized in this paper. The expressions for parametric spectral estimations of combined models of non-Gaussian stochastic processes are fond.

Fig. 2. Ref.: 5 items.

УДК 621.38

Синтез унітарних біноміальних лічильників/ О.А. Борисенко, В.В. Петров//АСУ та прилади автоматики. 2009. Вип.148. С. 8-13.

Отримані аналітичні вирази функцій збурення D-тригерів швидкодіючих n-розрядних унітарних біноміальних лічильників. Ці лічильники мають регулярну структуру та зручні при реалізації на ПЛІС.

Табл. 4. Іл. 8. Бібліогр.: 2 назви.

UDC 621.38

Synthesis of unitary binomial counters/ A.A. Borysenko, V.V. Petrov// Management Information System and Devices.2009. N 148. P.8-13.

In this paper analytical function of excitation of D-triggers of high speed n-digit unitary binomial counters is suggested. These counters have cellurar architecture which gives preference when using EPLD.

Tab.4. Fig. 8. Ref.: 2 items.

УДК 631.3.037.37

Перетворення двійкових і факторіальних чисел за допомогою лічильних пристроїв / О.Є. Горячев//АСУ та прилади автоматики. 2009. Вип. 148. С. 14-19.

Розглянуто один з методів отримання факторіальних чисел, що базується на використанні двох лічильників.Цей метод також дозволяє перетворювати факторіальні числа у двійкові. Розроблено структуру системи, що реалізує запропонований метод.

Табл. 2.Іл. 7. Бібліогр.: 4 назви.

UDC 631.3.037.37

Transformation of binary and factorial numbers based on counting devices/ A.E.Goryachev// Management Information System and Devices. 2009. N 148. P. 14-19.

One of the ways of getting factorial numbers based on the use of two counters is shown. This method also allows transforming factorial numbers to binary. The system realizing proposed algorithm is developed.

Tab.2. Fig. 7. Ref.: 4 items.

УДК 681.518:004.93.1

Ієрархічний алгоритм розпізнавання електронограм / А.С. Довбиш, К.В. Алтиннікова // АСУ та прилади автоматики. 2009. Вип. 148. С. 20-25.

Розглянуто метод розпізнавання електронограм, одержаних в режимі мікродифракції, в рамках інформаційно-екстремальної інтелектуальної технології, що ґрунтується на максимізації інформаційної спроможності системи розпізнавання в процесі її навчання. Розроблено алгоритм та програмне забезпечення системи розпізнавання електронограм. Побудовано ієрархічну структуру для етапів навчання та екзамену.

Іл. 5. Бібліогр.: 7 назв.

UDC 681.518:004.93.1'

Hierarchical algorithm of electronograms recognition / A.S. Dovbysh, K.V. Altynnikova // Management Information System and Devices. 2009. N 148. P. 20-25.

The article describes the method of recognition electronograms, gained in a microdiffraction mode, in context of information-extreme intellectual technology, which based on recognition system information capacity maximization in study and exam processes. Algorithm and software of recognition system were developed. The hierarchical structure for study and exam phases was build.

Fig. 5. Ref.: 7 items.

УДК 681.326:519.613

Тестування та верифікація HDL-моделей компонентів SOC. II / В.І. Хаханов, Є.І. Литвинова, І.О. Побіженко, Tiecoura Yves, Ngene Christopher Umerah // АСУ та прилади автоматики. 2009. Вип. 148. С. 26-37.

Запропоновано комплекс технологічних заходів і рекомендацій, орієнтованих на тестопридатний аналіз і послідовний синтез програмних продуктів і придатних для тестування й верифікації. Наведено приклади аналізу тестопридатності шляхом підрахунку керованості та спостережуваності транзакційного та керувального графів в цілях визначення критичних точок з наступним вирішенням практичної проблеми пошуку і усунення помилок в реальному DSP-проекті від компанії Xilinx.

Іл. 7. Бібліогр.: 8 назв.

UDC 681.326:519.613

Testing and verification of HDL-models for SOC components. II / V.I. Hahanov, E.I. Litvinova, I.O. Pobezhenko, Tiecoura Yves, Ngene Christopher Umerah // Management Information System and Devices. 2009. N 148. P. 26-37.

The technological tools, focused to testable analysis and subsequent synthesis of software is proposed. It is applicable for testing and verification. The examples of testability analysis by determination the controllability and observability of the transaction and control graph to detect the critical points with subsequent solving of the problem detection and removal of faults in a real DSP project of Xilinx.

Fig. 7. Ref.: 8 items.

УДК 004.652

Синтез моделі управління проектами розробки складних технічних систем в умовах ризику та невизначеності / В.В. Євсєєв, Ю.В. Шовкопляс // АСУ та прилади автоматики. 2009. Вип. 148. С. 38-42.

Досліджено управління проектами розробки складних технічних систем в умовах ризику та невизначеності. У роботі, у зв'язку з високою складністю об'єкта дослідження, формалізація процесу моделювання реалізується на основі імітаційного підходу. Як формальний апарат вибрано теорію імовірнісних мереж. Дані методи реалізовані на ЕОМ. Отримані експериментальні дані підтверджують доцільність та ефективність використаних методів.

Табл. 2. Іл. 1. Бібліогр.: 5 назв.

UDC 004.652

Project management model synthesis for developing complex technical systems in conditions of risk and uncertainty / V. V. Evssev, Y. V. Shovkoplyas // Avtomatizirovannie sistemi upravleniya i pribori avtomatiki. 2009. N 148. P. 38-42.

The object of study is project management of developing of complex technical systems in conditions of risk and uncertainty. In the work, in connection with the high complexity of the facility study, formalization of simulation-based simulation approach. As the formal apparatus of the theory of probability selected networks. These methods implemented on a computer. The obtained experimental data confirm the feasibility and effectiveness of methods.

Tab. 2. Fig. 1. Ref.: 5 items.

УДК 004.9

Система моделювання гетерогенних мікроконтроллерних мереж / Ю.К.Апраксін, І.О. Турега // АСУ та прилади автоматики. 2009. Вип. 148. С. 43-47.

Розглянута структура та моделі представлення мережевих об'єктів для побудови системи проектування та моделювання мікроконтроллерних мереж. Запропонована модульна структура системи. Виділено модуль введення та зберігання інформації, модуль проектування, модуль моделювання та аналізу. Розглянуто моделі основних мережевих об'єктів і їх властивості. Описані моделі приймачів та передавачів, представлених у вигляді конвеєрних структур відповідно зі стеком протоколів, що використовувався. Запропонована структура та моделі мережевих об'єктів дозволяють побудувати засіб для проектування та моделювання розподілених технічних систем.

Іл. 5. Бібліогр.: 2 назви.

UDC 004.9

The simulation system of heterogeneous microcontroller networks / Yu.K. Apraksin, I.O. Turega // Management information systems and devices. 2009. N. 148. P. 43-47.

The structure and presentation of the network object model for building system design and development of microcontroller networks are considered. The modular structure of the system is proposed. The input and storage module, design module, simulation and analysis module are described. The models of the basic network objects and their properties are considered. Models of receivers and transmitters, presented in the form of conveyor structures in accordance with stack of protocols that was used, are described. The proposed structure and model of network objects allow building a tool for design and simulation of distributed technical systems.

Fig. 5. Ref.: 2 items.

УДК 004

Марківські моделі для оцінювання рейтингу веб-сайту / З.В. Дудар, М.В. Збітнєва // АСУ та прилади автоматики.2009. Вип. 148. С. 47-51.

Побудовані марківські моделі, які відображають частоту та тривалість відвідування сайту, запропонована структура інтелектуального агенту як складова частина критерію оцінювання рейтингу веб-сайту. Розглянута загальна та типова структура сайту, наведені матриці вірогідностей переходу. До перспектив подальшого дослідження відноситься моделювання сполучення слів у тексті при урахуванні якості веб-контенту.

Іл. 3. Бібліогр.: 5 назв.

UDC 004

Markov Model for web-site rating estimation / Z.V. Dudar, M.V. Zbitneva // Management information systems and devices. 2009. N. 148. P.47-51.

There were proposed Markov models for displaying frequency and duration of visiting web-site. Intelligent Agents structure as part of criteria of estimation web-site rating was offered. It was considered general and typical web site structure, matrix of probability of transition. Modeling of combination of words in text for calculation of quality of web-content is related to perspective of future research.

Fig. 3. Ref.: 5 items.

УДК 004.7; 004.8; 007.85

Розробка та дослідження бази даних та бази знань для ухвалення рішень в інформаційній системі обслуговування банкоматів. / Н.В.Головій // АСУ та прилади автоматики.2009. Вип. 148. С. 52-58.

Особливістю розглянутого підходу є оригінальний метод щодо витягання прихованих знань і побудови бази знань інформаційної системи у сфері сервісного обслуговування банкоматів на основі нейромережевого підходу, який передбачає вживання спеціальної процедури відбору і верифікації правил.

Іл. 1. Бібліогр.: 8 назв.

UDC 004.7; 004.8; 007.85

Development and research of database and base of knoweledges for making decision in informative system of maintenance of ATMs./ N.V.Goloviy // Management information systems and devices. 2009. N. 148. P.52-58.

The feature of the considered approach is an original method, in relation to drawing out of the hidden knowledges and knowledge informative system acquisition in the field of service of ATMS on the basis of neuronic nets which foresees the use of the special procedure of selection verifications of rules.

Fig. 1. Ref.: 8 items.

УДК 004.421:548.55

Синтез прогнозного регулятора для процесу вирощування об'ємних монокристалів кремнію для сонячних ФЕП / А.П. Оксанич, В.Р. Петренко, С.О. Волохов // АСУ та прилади автоматики.2009. Вип. 148. С. 59-70.

На основі використання підходу Бокса-Дженкінса до синтезу моделей стохастичних лінійних динамічних процесів розроблена АРМАХ-модель процесу витягування монокристалічного злитка, що зв'язує варіації швидкості витягування з варіаціями діаметра злитка. Отримана модель використана для синтезу оптимального прогнозного регулятора процесу вирощування на етапі витягування циліндричної частини злитка. Наведені результати моделювання роботи регулятора, що підтвержують його працездатність.

Іл. 9. Бібліогр.: 17 назв.

UDC 004.421:548.55

Forecast Regulator Synthesis for the Growth Process of Cz-Si Single Crystals for the Solar PED / A.P. Oksanich, V.R. Petrenko, S.A. Volokhov // Management information systems and devices. 2009. N. 148. P. 59-70.

On the basis of Box-Jenkins's approach to synthesis of dynamic process stochastic linear models APMAX-model is developed of monocrystal ingot stretching process connecting variations of stretching speed with ingot diameter variations. The model produced is used for optimum predictive control regulator synthesis of the growth process on ingot cylindrical part stretching stage. The results of regulator operation modelling verified its serviceability.

Fig. 9. Ref.: 17 items.

УДК 004.732

Моделі зворотного зв'язку для протоколу RTCP / Г.В. Бабіч, Мурад Алі А. // АСУ та прилади автоматики.2009. Вип. 148. С. 71-75.

Розглянуто моделі зворотного зв'язку для протоколу RTCP, використання яких дозволяє вирішити проблему зниження навантаження на мережу та збалансованості широкомовного трафіка RTP/RTCP. Досліджено їх особливості, переваги та недоліки. Запропоновано розширення однієї з моделей зворотного зв'язку (модель, що резюмує), яке дозволило включити в звіти отримувача інформацію про фактори, які найбільш впливають на сеанс RTP. Сформульовано задачі, результатом вирішення яких є реалізація запропонованого раніш розширення стосовно протоколу RTCP.

Іл. 6. Бібліогр.: 7 назв.

UDC 004.732

RTCP Feedback models / G.V. Babich, Murad Ali A. // Management information systems and devices. 2009. N. 148. P. 71-75.

RTCP feedback models have been considered. The usage of RTCP feedback models allows to solve problem of networl load decreasing and braodcast traffic RTP/RTCP.balancing. Their features, advantages and shortcomings have been investigated. The enchancement for one of RTCP feedback model (summarising model), which allows to include into receiver reports the information about factors with the highest influence onto RTP session, has been proposed. The tasks for further realization of proposed RTCP enchancement have been formulated

Fig.6. Ref.: 7 items.

УДК 681.326:519.713

Інфраструктура верифікації та тестування SoC / Є.І. Литвинова // АСУ та прилади автоматики. 2009. Вип. 148. С. 76-86.

Запропоновано інфраструктуру сервісного обслуговування цифрових систем на кристалах, основану на сучасних стандартах і технологіях проектування SoC від провідних компаній світу. Використання механізму асерцій та IEEE 1500 SECT стандарту дає можливість підвищити ефективність сервісних засобів моделювання, діагностування та відновлення працездатності, а також суттєво зменшити час верифікації HDL-моделей і тестування апаратних компонентів SoC.

Табл. 2. Іл. 8. Бібліогр.: 26 назв.

UDC 681.326:519.613

Verification and Testing Infrastructure for SoC / E.I. Litvinova // Management information systems and devices. 2009. N. 148. P. 76-86.

Infrastructure IP for SoC, based on the SoC design standards and technologies from leading world companies, is proposed. The assertion engine and IEEE 1500 SECT allows increasing the productivity of simulation, diagnosis and repairing facilities, as well as decreasing the verification time for HDL-models and the testing time for hardware SoC components.

Tab. 2. Fig. 8. Ref.: 26 items.

УДК 681.326:519.713

Метод верифікації HDL-коду на основі транзакційного логічного графа / В.І. Хаханов, І.О. Побіженко, В.О. Василенко, С.В. Чумаченко // АСУ та прилади автоматики. 2009. Вип. 148. С. 87-101.

Запропоновано метод верифікації системних HDL-моделей, орієнтований на суттєве підвищення якості проектованих компонентів цифрової системи. Використання програмної надлишковості у вигляді механізму асерцій також дозволяє зменшити час розробки (time-to-market). Метод орієнтований на пошук помилок і дефектів із заданою глибиною в програмному HDL-коді шляхом введення у критичні точки транзакційної моделі спостерігача у вигляді асерційної надлишковості. Визначення критичних точок здійснюється за допомогою технології вирахування керованості та спостережуваності структур-

них компонентів програмного коду в цілях покращення його тестопридатності для діагностування семантичних помилок.

Табл. 4. Іл. 10. Бібліогр.: 23 назви.

UDC 681.326:519.613

Verification method for HDL-code based on transaction logic graph / V.I. Hahanov, I.A. Pobezhenko, V.A. Vasilenko, S.V. Chumachenko // Management information systems and devices. 2009. N. 148. C. 87-101.

Verification method for system HDL-models, focused on considerable increasing of the SoC components' quality is proposed. The software redundancy in the form of assertion engine allows decreasing the time-to-market. The method provides the detection of errors and faults with a given resolution in the software HDL-code by means of adding an observer in critical points of the transaction model as assertion redundancy. Determination of critical points is realized by means of calculation the controllability and observability for software components to improve the testability when detection of semantic errors.

Tab.4. Fig. 10. Ref.: 23 items.

ПРАВИЛА

оформления рукописей для авторов научно-технического сборника

"АСУ и приборы автоматики"

Формат страницы — А4 (210х297мм), поля: сверху, справа, слева, снизу – 30 мм. Редактор: Pagemaker 6.0, 6,5 (можно, но нежелательно Word), гарнитура Times New Roman Cyr, кегль – 11 пунктов, межстрочное расстояние — 110 %, табуляция — 5 мм.

Объем рукописи – до 10 с. (языки: русский, украинский, английский). Содержание должно отражать актуальность исследования, постановку задачи, цель, сущность, научные и практические результаты, сравнение с лучшими аналогами, выводы.

Структура рукописи: заголовок, аннотация, текст, литература, реферат на украинском и английском языках, сведения об авторах.

ОБРАЗЕЦ ОФОРМЛЕНИЯ

УДК 519.713

И.О. ФАМИЛИЯ

НАЗВАНИЕ РУКОПИСИ

Аннотация (абзац 5-10 строк, кегль 10) помещается в начале статьи и содержит информацию о результатах описанных исследований.

Основной текст можно разделять на 2 и более подразделов с заголовками, выделенными полужирным шрифтом, пронумерованными арабскими цифрами, как показано в следующей строке.

1. Название раздела

Рисунки и таблицы (черно-белые, контрастные) помещаются в текст после первой ссылки в виде *переносимых объектов* и раздельно нумеруются, при наличии более одного рисунка (таблицы), арабскими цифрами. Рисунок содержит подрисуночную центрированную подпись (текстовая строка, расположенная вне рисунка, кегль 10) под иллюстрацией, как показано на рис. 1.

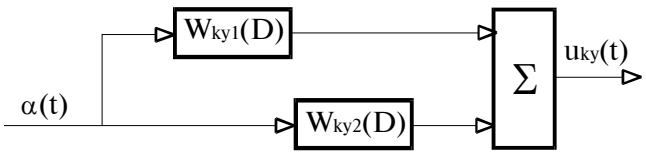


Рисунок 1 – Двухзвенная система

Табличный заголовок располагается справа над таблицей, что иллюстрируется табл.1. Редакторы: CorelDraw, Table Editor и др.

Таблица 1

Шаги	1	2	3	4	5	6
$\Phi_1(1,3)$	1	2	2	4	6	1
$\Phi_1(1,4)$	1	2	2	4	4	8

Формулы нумеруются при наличии ссылок на них в рукописи. Рекомендуемый кегль формульного набора: обычный (переменная) – 11 пунктов, крупный индекс – 8, мелкий индекс (над- и подиндекс) – 8, крупный символ (основной) – 12, мелкий (индексный) математический символ – 10:

$$F_{i+j} = \sum_{i=1}^{b^k} F_j^i - \prod_{j=1}^{1+h^2} P_{R\,j+i} + F^{j-1} + X^{\Sigma n^k}. \qquad (1)$$

Формат переменных (желательно не курсивом – без наклона) в тексте и формулах должен быть идентичным. В тексте над- и подиндексы составляют 70 % от кегля, которые рекомендуется опускать (поднимать) на 17 (33) % относительно основной строки.

Список литературы (включает опубликованные источники, на которые имеются ссылки в тексте, заключенные в квадратные скобки) печатается без отступа, кегль 9 пунктов.

Образец окончания текста рукописи (литература, сведения об авторах, реферат) представлен ниже.

Список литературы: 1. *Фамилия И.О.* Название книги. Город: Издательство, 1900. 000 с. 2. *Название* сборника / Под ред. И.О. Фамилия. Город: Издательство, 1900. 000 с. 3. *Фамилия И.О.* Название статьи / / Название журнала. Название серии. 1997. Т. 00, № 00. С. 00-00 .

Поступила в редколлегию 00.00.00

Фамилия, имя, отчество, ученая степень, звание, должность и место работы. Научные интересы. Адрес, контактный телефон.

Рефераты на украинском и английском языках. Текст аннотации не должен дублировать реферат.

УДК 000.000.00

Назва статті українською мовою / Ініціали. Прізвище // АСУ та прилади автоматики. 2000. Вип. 00. С. 000-000.

Текст реферату.

Табл. 00. Іл. 00. Бібліогр.: 00 назв.

UDC 000.000.00

Title of paper / Initials. Surname // Management Information System and Devices. All-Ukr. Sci. Interdep. Mag. 2000. N 00. P. 000-000.

Text.

Tab. 00. Fig. 00. Ref.: 00 items.

Представление материалов

Рукопись, реферат, сведения об авторах — в одном файле, *поименованном фамилией первого автора*, на дискете 3,5 дюйма. Твердая копия материалов – для граждан Украины — в одном экземпляре: рукопись, подписанная авторами, рефераты, акт экспертизы, внешняя рецензия, подписанная доктором наук, заявление на имя главного редактора со сведениями об авторах.

Адрес редакции: Украина, 61166, Харьков, просп. Ленина, 14, ХНУРЭ, комната 321, тел. 70-21-326.

E-mail: hahanov@kture.kharkov.ua.

Тематика статей, публикуемых в сборнике:

– Компьютерная инженерия
– Математическое моделирование
– Оптимизация и процессы управления
– Автоматизация проектирования и диагностика
– Информационные интеллектуальные системы
– Проектирование интегральных схем и микросистем
– Компьютерные технологии в образовании

Відповідальний випусковий В.І. Хаханов
Редактор О.П. Гужва
Комп'ютерна верстка Г.В. Хаханова, С.В. Чумаченко

Підп. до друку 27.09.2009. Формат 60x84 1/8. Умов. друк. арк. .
Обл.-вид. арк. . Тираж 300 прим.
Зам. № . Ціна договірна.

Харківський національний університет радіоелектроніки (ХНУРЕ).
Україна, 61166, Харків, просп. Леніна, 14.

Оригінал-макет підготовлено і збірник віддруковано
в навчально-науковому видавничо-поліграфічному центрі ХНУРЕ.
Україна, 61166, Харків, просп. Леніна, 14.