находится в обратной зависимости с оценочной величиной порога p^*.

Экстремальный характер зависимости оценки p^* порога перколяции от состава частиц бидисперсной фазы (см. рис. 5 б) свидетельствует о фрактальном характере явления протекания, который отмечается и в регулярных структурах, но в случайной нерегулярной структуре является более выраженным – возможно, вплоть до мультифрактального [2].

В ходе дальнейших, более детальных исследований возможно установление строгих закономерностей, развивающих обнаруженные и представленные в данной работе особенности перколяции в случайных нерегулярных структурах. Результаты могут иметь разнообразное практическое применение в материаловедении пористых композитов, Так, важной прикладной задачей является исследование поглощения влаги цементным камнем [4]. Аналогично, распределение влаги в объеме силикагеля существенно влияет на эффективность специальных средств защиты от электромагнитных излучений [5].

СПИСОК ЦИТИРОВАННЫХ ИСТОЧНИКОВ

1. Тарасевич, Ю.Ю. Перколяция: теория, приложения, алгоритмы / Ю.Ю. Тарасевич. – М.: Наука, 2002. – 112 с.
2. Федер, Е. Фракталы / Е. Федер. – М.: Мир, 1991.- 262 с.
3. Дереченник, А.С. Стохастическая модель многочастичной неупорядоченной системы для исследования топологических свойств дисперсных структур / А.С. Дереченник // Вестник БрГТУ. Физика, математика, информатика, 2008.
4. Bentz, D.P. Protected paste volume in concrete: extension to internal curing using saturated lightweight fine aggregate / D.P. Bentz, K.A. Snyder // Cement and Concrete Research. – 1999. – Vol. 29, № 11. – P. 1863-1867.
5. Богуш, В.А. Электромагнитные излучения. Методы и средства защиты / В.А. Богуш [и др.]; под ред. Л.М. Лынькова. – Минск: Бестпринт, 2003. – 406 с.

Материал поступил в редакцию 20.11.2008

DERECHENNIK A.S., DERECHENNIK S.S. The distinctive features of percolation in the non-regular dispersed structures

New characterization technique for percolation clusters in the non-regular structures is proposed, based on the stochastic connectivity matrixes definition. Using numerical simulation the distinctive features of percolation in the non-regular mono- and bi-dispersed plain structures are investigated. The percolation transition type in such structures and its correlation with system coordination number distribution are analyzed. An estimate is made for the threshold pore density value of the percolation starting point.

УДК 681.3

Мухов С.В., Муравьев Г.Л.

ОРГАНИЗАЦИЯ БИБЛИОТЕКИ ФОРМИРОВАТЕЛЕЙ СИГНАЛОВ ДЛЯ МОДЕЛИРОВАНИЯ ПРОЕКТОВ ЦИФРОВЫХ УСТРОЙСТВ

Введение. При проектировании сложных технических систем с большим числом элементов, типов элементов, взаимным влиянием узлов и процессов, наличием управления, к числу которых относятся и цифровые устройства (ЦУ), обязательным этапом является моделирование, требующее эффективных средств. Моделирование проектов ЦУ с учетом временных задержек, разрабатываемых, например, с использованием специализированных языков спецификации верхнего уровня, таких как HDL, VERILOG-HDL, VHDL [1-5], сводится к имитации преобразования в соответствии с логикой функционирования проекта ЦУ входной информации, представленной сигналами, в выходные сигналы.

В работе рассматривается подход к организации информационного обеспечения моделирования применительно к языку VHDL (стандарты в области автоматизации проектирования VHDL'93 - ANSI/IEEE Std 1076-1993 и VHDL-AMS - Std 1076.1-1999). Исходными данными служат: входные воздействия, имитирующие реальное окружение ЦУ, выходные и эталонные реакции ЦУ, интерфейсы проектов и пакеты заказчика, описывающие пользовательские типы сигналов.

Требования к описанию сигналов. В VHDL сигналы наряду переменными относятся к категории объектов данных (object ::= constant | signal | … variable), используются в структурном, потоковом, поведенческом стилях описания моделей. Это входные, выходные, входные-выходные сигналы, представленные значениями любых типов и соответствующим образом декларированные в заголовке проекта (путем объявления портов проекта - физических входов-выходов проекта, обеспечивающих каналы динамической связи с внешней средой) как

interface_signal_declaration ::= [signal] signal_identifier: [mode] type [bus] [:= expression];

и внутренние сигналы, декларированные в теле архитектуры проекта как

signal_declaration :: = signal signal_identifier : type [signal_kind] [:= expression]; .

Здесь mode ::= IN | OUT | INOUT, signal_kind ::= register | bus.

С каждым сигналом связан формирователь (драйвер), отображающий характер изменения сигнала во времени и являющийся источником его значений. Это структура данных, хранящая упорядоченные во времени списки прошлых и будущих значений сигналов. Формирователи сигналов составляют информационную базу модели и являются основой организации и ведения списков событий при моделировании.

Это требует эффективного хранения и обработки значений формирователей до начала моделирования (для подготовки входных воздействий, включая тестовые входные и эталонные выходные), в его процессе (чтение, анализ, изменение, сохранение значений) и после завершения. При моделировании реальных проектов, отличающихся иерархичностью моделей, организация формирователей должна производиться с учетом их привязки к проектам, портам, версиям архитектурной реализации, требований атрибутивной арифметики и набором типов, допустимых синтаксисом и семантикой VHDL, а также с учетом поддержки статичного и динамического способов представления входных воздействий [6-9].

Логическая структура библиотеки. Структурно библиотека включает следующие разделы: **1.** Логическое оглавление верхнего уровня с ключевым полем - описание тестовых наборов данных проекта (ТНД). Представляет собой лексикографический перечень имен проектов, соответствующих им имен описаний структур тестовых наборов данных (СТНД) и списков имен самих тестовых наборов данных (в отношении "один к многим"). Для хранения списка наборов данных в библиотеке используются либо структуры операционной среды (каталоги) с их буферизацией в рабочей области, либо предварительно размеченные последовательные наборы данных, поддерживающие эту матричную структуру. Здесь идентификация наборов данных осуществляется с использованием префиксации и обеспечивается иерархическое упорядочивание объектов. **2.** Библиотеку описаний СТНД (как в логическом так и в машинном форматах) для статичного (СТНДС) и динамического способов представления зна-

чений сигналов, входных воздействий (СТНДД). **3.** Библиотеку ТНД для статичного (ТНДС) и динамического способов представления значений сигналов, входных воздействий (ТНДД). **4.** Библиотеку выходных реакций. Представляет из себя набор данных, который содержит выходные реакции системы. Строится по аналогии с библиотекой ТНДС. Связывание с заданным на входе тестовым вектором обеспечивается заданием идентификатора тестового вектора с указанием его номера в соответствующем префиксе тестового вектора. Сопровождение набора поддерживается средствами операционной системы.

Тестовые наборы могут описываться как наборы векторов входных и выходных значений сигналов, носящих характер статических либо динамических последовательностей. При этом используется разный синтаксис описания векторов и различная их интерпретация при моделировании проектов. Описание, в том числе, включает указание требований к точности результатов, задаваемых индивидуально по каждому сигналу или по всем сигналам набора, и способ задания точности. Более детально разделы библиотеки представлены ниже.

Библиотека описаний СТНД включает разделы:

- библиотеку описаний структур статических тестовых наборов данных СТНДС, используемых в качестве маски соответствующих наборов тестовых векторов (структура Маски_ТНДС). Представляет из себя последовательный набор данных, который содержит описание полей тестового вектора на языке высокого уровня, используемого при моделировании. Может включаться в программные тексты классическим образом (директивой include) и сопровождаться средствами операционной системы. Генерируется с помощью специальной процедуры на основании описания проекта на языке VHDL;
- библиотеку описаний СТНДС в машинном формате. Представляет из себя последовательные наборы данных, которые содержат описание полей тестового вектора в кодах (структура Описание_ТНДС). Используется в процедурах системы для редактирования тестовых векторов при работе с экраном или при обслуживании процедур моделирования. Сопровождается средствами операционной системы;
- библиотеку описаний структур динамических тестовых наборов данных СТНДД для использования в качестве маски соответствующих наборов тестовых векторов (структура Маски_ТНДД). По функциональному назначению и построению является аналогом библиотеки СТНДС, но учитывает специфику задания динамических тестовых векторов;
- библиотеку описаний СТНДД в машинном формате. Является аналогом библиотеки СТНДС (структура Описание_ТНДД), но учитывает специфику задания динамических тестовых векторов.

Библиотека ТНД включает разделы:

- ТНДС, который представляет из себя последовательные наборы данных, содержащие статические тестовые векторы фиксированной длины;
- ТНДД, который представляет из себя набор данных, который содержит динамические тестовые векторы. Строится либо аналогично ТНДС, либо с использованием специального формата, учитывающего специфику динамических тестовых векторов. Их сопровождение поддерживается средствами операционной системы.

Рекомендации по физической реализации библиотеки. Поскольку в исходном представлении значениям формирователей каждого из сигналов соответствует отдельный файл, а сами значения, хранящиеся в разных файлах, неупорядочены друг относительно друга и файлы имеют разную длину, то во внутреннем представлении в целях удобства хранения и обработки формат хранения тестов в динамическом варианте приводится к формату статичного варианта. Для этого исходные временные диаграммы сигналов заменяются множеством тестовых векторов, каждый из которых соответствует одному временному срезу. А в качестве временных срезов выбираются точки временной оси, где происходит изменение хотя бы одного сигнала. При этом в тестовый вектор вместо его номера вносится значение времени изменения сигнала, а поле значения содержит собственно новое значение сигнала.

Оглавление содержит описательные структуры папок следующего вида

```
struct     def_dir
{
     char dir_sw; /* флаги */
     char[dir_prjl] dir_prj;  /* идентификатор проекта */
     char[dir_textl] dir_text;  /* комментарий */
     char[dir_mtdsl] dir_mtds; /* маска */
     char[dir_dtdsl] dir_dtds;  /* идентификатор ТНД */
     char[dir_freel] dir_free;  /* резерв */
}  .
```

Здесь поле dir_sw - флаг записи (0x80 - последний элемент набора; 0x40 - элемент свободен; 0x20 - проект содержит ТНДС; 0x10 - проект содержит ТНДД), dir_prj - имя проекта, dir_text - поле комментария, dir_mtds - имя Маски_ТНДх, dtds - имя Описания_ТНДх, поле free - резервное. Длина имени проекта dir_prjl, маски dir_mtdsl, описания dir_dtdsl, dir_freel - предварительно определенные константы.

Описание_ТНД содержит структуры следующего вида

```
struct def_dtds
{
     char dtds_sw;     /* флаги */
     char dtds_swx;   /* флаги */
     char dtds_swz;   /* флаги */
     char dtds_swe;   /* флаги */
     int dtds_error;     /* погрешность */
     int dtds_error1;  /* погрешность */
     char[dtds_textl] dtds_text;  /* комментарий */
     char[dir_mtdsl] dtds_mtds; /* маска */
     int dtds_nmbx;    /* количество сигналов */
     int dtds_recl;  /* длина записи вектора */
     int dtds_prfl;  /* длина префикса вектора */
     struct def_sig  dtds_signal [dtds_nmb]; /* описания сигналов */
}   .
```

Здесь поле dtds_sw - флаг записи (0x80 - последний элемент набора; 0x40 - элемент свободен; 0x20 - расширение dtds; 0x10 - последний элемент расширения dtds; 0x08 – описание ТНДС; 0x04 - описание ТНДД; 0x02 – время, номер такта; 0x01 - время типа real). Поле dtds_swx - флаг записи (0x80 - нет выходных реакций), dtds_swe - флаг (0x80 - общая погрешность; 0x40 - число старших разрядов общей погрешности; 0x20 – абсолютное значение общей погрешности; 0x10 - относительное значение общей погрешности; 0x08 - формат int общей погрешности; 0x04 - формат float общей погрешности; 0x02 - погрешность по каждому сигналу). Поле dtds_error - значение общей погрешности, dtds_error1 - значение общей погрешности (поле расширения), dtds_text - поле комментария, dtds_mtds - имя Маски_ТНДх, dtds_nmbx - количество сигналов, dtds_recl - длина записи тестового вектора, dtds_prfl - длина префикса тестового вектора, dtds_signal - массив описаний сигналов. Здесь dtds_text, dtds_nmb - предварительно определенные константы. При большом количестве сигналов происходит выделение дополнительных элементов dtds с установкой соответствующих флагов.

Описание сигналов выполняется с помощью следующей структуры:

```
struct def_sig
{
     char sig_sw;   /* флаги */
     char sig_swf;   /* флаги */
     char sig_swtv;     /* флаги */
     char sig_swe;  /* флаги */
     char[sig_idtxtl] sig_idtxt; /* идентификатор сигнала */
     int sig_value;  /* значение сигнала */
     int sig_value1;     /* поле расширения */
     int sig_error;       /* значение погрешности */
     int sig_error1;     /* поле расширения */
}  .
```

Здесь поле sig_sw - флаг записи (0x80 - последний элемент набора; 0x40 - элемент свободен; 0x20 – режим сигнала IN; 0x10 – режим сигнала OUT; 0x08 – INOUT; 0x04 - снятие выходных реакций). Поле sig_swf - флаг формата внутреннего представления данных, sig_swtv - флаг формата экранного представления данных, sig_swe - флаг представления погрешности (см. dtds_swe), sig_idtxt - текстовый идентификатор сигнала, sig_value - значение сигнала, sig_value1 - значение сигнала (поле расширения), sig_error - значение погрешности (см. dtds_swe), sig_error1 - значение погрешности (поле расширения). Здесь sig_idtxt - предварительно определенная константа.

Таким образом, данные тестового вектора представляют из себя линейную область памяти, полностью определяемую структурой Описание_ТНД. Набор ТНД есть матричная структура из элементов типа {< Префикс_тестового_вектора >, < Тестовый_вектор >}. Длина элемента матричной структуры определена полем dtds_recl, длина префикса тестового вектора полем dtds_prfl. Соответственно начало тестового вектора с заданным номером задается смещением, которое определяется значениями соответствующих полей, например, как <Номер_вектора> * (<Длина_вектора> + <Длина_префикса_вектора>), что может быть легко реализовано с использованием дескрипторных файлов. <Префикс_вектора> включает поле <Номер_такта> или <Время>, что позволяет на одной структуре реализовать как ТНДД так и набор данных, описывающих выходные реакции.

При реализации библиотеки необходимо также учитывать, что внутреннее представление форматов тестовых данных определяется в описании ТНД (поле sig_swf). При поддержке экранного интерфейса выполняется преобразование данных согласно значению поля sig_swtv. При необходимости преобразования данных во время вызова моделирующей программы дополнительно может быть определена структура, аналогичная def_dtds, но с замененными полями, что позволит с минимальными затратами обеспечить их преобразование.

Заключение. Сформулированы требования к организации хранения и обработки значений формирователей сигналов с учетом типовых задач, решаемых в ходе постановки имитационных экспериментов с проектами цифровых устройств, особенностей использования формирователей при моделировании. Применительно к языку VHDL предложены форматы и структуры хранения информации в базе данных с учетом их привязки к проектам, портам, версиям архитектурной реализации. Это обеспечивает формирование структуры базы данных, ориентированной на поддержку и эффективное обеспечение задач моделирования.

Результаты создают основу для реализации соответствующих библиотек и для построения системы с автоматической настройкой модулей на интерфейс моделируемого проекта, что автоматизирует подготовку входных воздействий для проектируемого устройства, а в ходе моделирования - накопление и анализ реакций модели с выдачей результатов.

СПИСОК ЦИТИРОВАННЫХ ИСТОЧНИКОВ

1. VHDL для моделирования, синтеза и формальной верификации аппаратуры/Пер. с англ. - М.: Радио и связь, 1995. – 360 с.
2. Бибило П.Н. Основы языка VHDL. - М.: “СОЛОН-Р”, 2000. – 210 с.
3. Сергиенко А.М. VHDL для проектирования вычислительных устройств. - К.: “Корнейчук”, 2003. – 208 с.
4. Армстронг Дж. Р. Моделирование цифровых систем на языке VHDL. - М.: Мир, 1992. – 175 с.
5. Поляков А.К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры. - М.: “СОЛОН-Пресс”, 2003. – 320 с.
6. Прихожий А.А., Муравьев Г.Л. Система проектирования цифровых СБИС с языком VHDL на ПП ЭВМ. Разработка и использование ПЭВМ// Труды международного симпозиума INFO' 89. - Мн., 1989. - т.2, ч.1. – 6 с.
7. Муравьев Г.Л. Автоматизированная система подготовки тестов // Сборник трудов научно-технической конференции “Новые технологии в машиностроении и вычислительной технике”. – Брест: Брест. политехн. ин-т, 1996.
8. Дудкин А.А., Головко В.А. , Муравьев Г.Л. и др. Алгоритмы и подсистемы автоматизированного логического проектирования цифровых СБИС. Материалы по математическому обеспечению ЭВМ. - Мн., ИТК АН РБ, 1994. – 120 с.
9. Муравьев Г.Л., Шуть В.Н., Мухов С.В. Автоматизация построения имитационных моделей по их процессным испытаниям// Вестник БГТУ.– 2007.– № 5(47).– С. 39-42.

Материал поступил в редакцию 07.07.2008

MUHOV S.V., MURAVJOV G.L. Signal drivers library organization for digital device projects simulation

The approach to signal drivers library organization is considered. The library structure and data structures focused on realization by means of the programming languages of a high level are resulted.

УДК 004.514.62

Тавониус К.А., Костюк Д.А.

ПРИМЕНЕНИЕ МАСШТАБНЫХ ПРЕОБРАЗОВАНИЙ ПИКТОГРАММ ДЛЯ УПЛОТНЕННОГО ОТОБРАЖЕНИЯ ФАЙЛОВОЙ СИСТЕМЫ

Ведение. В последнее время все большее внимание уделяется попыткам использовать уменьшенный масштаб изображений, не находящихся в фокусе работы пользователя, для увеличения наглядности и интуитивности интерфейса [1]. Не в последнюю очередь оживление в данной области связано с ростом разрешающей способности дисплеев, делающей более информативным применение уменьшенных изображений объектов для предварительного просмотра (previews или thumbnails) - тумбнейлинга. Также, рост производительности систем позволяет добиться оптимального расчёта положения и размера объектов. Ниже описана предпринятая нами попытка применения переменного масштабирования в контейнерах пиктограмм на примере файлового менеджера - типичного приложения, интенсивно использующего данный вид элементов управления.

В последнее время масштабные преобразования начинают применяться в средствах навигации по файловой системе. Однако их использование ограничено одноуровневым тумбнейлингом – использованием сильно уменьшенных изображений или первых страниц документов вместо пиктограмм [2, 3].

Существует несколько аналогий (метафор), позволяющих эффективно представить пиктограммы переменного масштаба. В данной работе нами выбраны аналогии периферического зрения (ПЗ) и гравитационной аномалии (ГА).

Костюк Дмитрий Александрович, *к.т.н., доцент кафедры «ЭВМ и системы» Брестского государственного технического университета.*

Тавониус Кирилл Андреевич*, студент факультета электронно-информационных систем Брестского государственного технического университета.*

Беларусь, БрГТУ, 224017, г. Брест, ул. Московская, 267.