УДК 681.518.54

ИЕРАРХИЧЕСКАЯ КОМПИЛЯТИВНАЯ СИСТЕМА МОДЕЛИРОВАНИЯ И ГЕНЕРАЦИИ ТЕСТОВ

А. И. АНДРЮХИН окончил Донецкий государственный университет. С 1991 г. кандидат технических наук. Научный сотрудник отдела управляющих систем. Круг интересов: моделирование и диагностика цифровой аппаратуры, проблемы программирования и искусственного интеллекта. Автор 30 печатных работ.

Д. В. СПЕРАНСКИЙ окончил Саратовский государственный университет. С 1975 г. кандидат физико-математических наук, с 1983 г. — доктор технических наук, в 1985 г. утвержден в ученом звании профессора. Заведующий отделом теории управляющих систем Института прикладной математики и механики НАН Украины. Круг интересов: проблемы моделирования и технической диагностики цифровой аппаратуры, включая исследования в области теории и разработки прикладных программных систем.

Система предназначена для автоматического моделирования и диагностирования современных дискретных устройств (ДУ). Основой системы являются семантико-синтаксический анализатор и препроцессор. Описаны алгоритмы функционирования этих модулей и процесс создания программных модулей стандартных элементов. Описаны особенности программной реализации системы и ее характеристики (время моделирования исправных и неисправных ДУ).

Система призначена для автоматичного моделювання і діагностування сучасних дискретних пристроїв (ДП). Основою системи є семантико-синтаксичний аналізатор і препроцесор. Описані алгоритми функціонування цих модулів і процес створення програмних моделей стандартних елементів. Описані особливості програмної реалізації системи і її характеристики (час моделювання справних та несправних ДП).

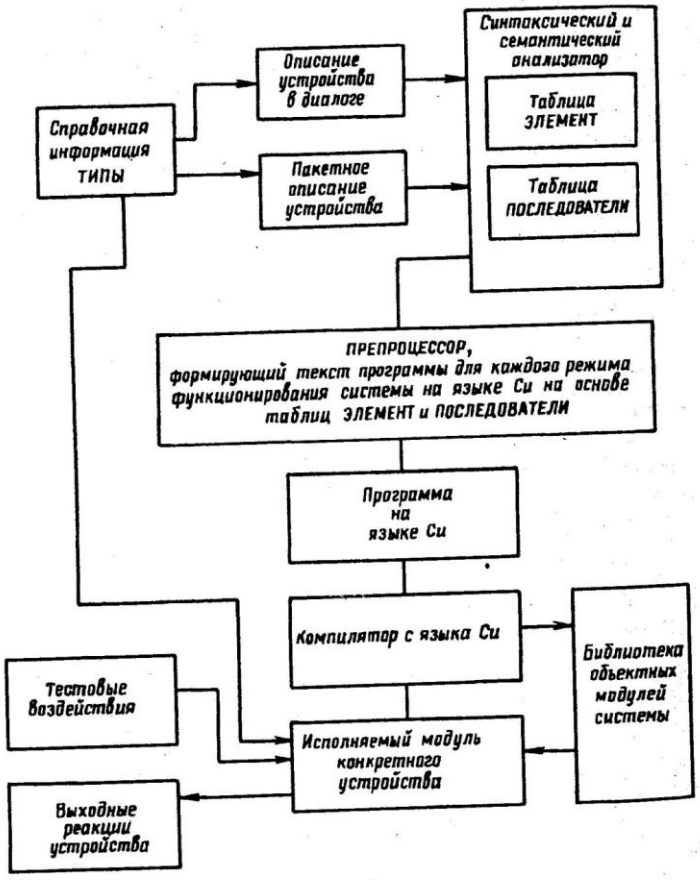
Иерархическая компилятивная система моделирования (ИКСМ) является инструментальной оболочкой для построения специализированных систем моделирования, в частности логического моделирования цифровых устройств и моделирования информационных систем. Структура подсистемы логического моделирования ИКСМ представлена на рисунке, аналогичную структуру имеют и все другие ее подсистемы. По принципу построения ИКСМ имеет много общего с системой, описанной в [1]. Основу

инструментальной оболочки составляют семантико-синтаксический анализатор и препроцессор.

С использованием семантико-синтаксического анализатора производится формирование стуктурного устройства (схемы, системы), состоящего из описания его компонентов и связей между ними. Такое описание фиксируется в таблицах с именами ЭЛЕМЕНТЫ и СВЯЗИ.

На основании упомянутых таблиц и справочника системы с именем ТИПЫ препроцессор на языке Си формирует программы, которые соответствуют конкретным режимам моделирования и генерации тестов. Препроцессор включает в них неизменные программные тексты определенных алгоритмов моделирования и выполняет следующие действия:

определяет размер памяти и идентификаторы для выводов компонентов устройства и его переменных состояния;



Структура подсистемы логического моделирования ИКСМ.

ISSN 0235-3474. ТД и НК, 1994, № 2

определяет имена используемых подпрограмм, описывающих функционирование элемента конкретного типа, соответствующего компоненту устройства;

генерирует требуемые для данного режима моделирования структуры данных, в частности списки предшественников и последователей компонента устройства.

Отметим, что препроцессор представляет собой комплекс программ, каждая из которых предназначена для реализации определенного режима функционирования, хотя они меют единую идеологию построения, а также единые входные данные.

Опишем более подробно данные справочника ТИПЫ и формируемых таблиц ЭЛЕМЕНТЫ и СВЯЗИ. Для каждого типа элемента справочник ТИПЫ содержит: R — признак элемента, P — имя программы элемента, T — тип элемента, I (Q) — число входов (выходов) элемента, S — число внутренних переменных элемента, а также данные о номерах и типах входов и выходов элемента, которые могут быть двунаправленными.

Таблица ЭЛЕМЕНТЫ для каждого из множества компонентов устройства содержит запись вида (T, N), где T — тип элемента, N — имя компонента в устройстве. Таблица СВЯЗИ содержит связи компонентов в устройстве, каждая из которых описывается данными (d, k, d_i, k_i), где $k(k_i)$ — выход (вход) компонентов $d(d_i)$ соответственно, посредством которых элементы d и d_i связаны.

Остановимся кратко на алгоритме работы анализатора. Основной задачей анализатора является формирование структурного описания устройства с параллельной проверкой данных, которые вводит пользователь в диалоге с ним. Анализатор требует от пользователя ответа в виде символьной строки на следующие основные последовательные запросы:

1) введите имя схемы:

2) введите тип элемента;

3) введите имя элемента (после фиксации типа элемента в запросе 2);

4) введите начало цепи (имя элемента и номер его выхода);

5) введите последователей цепи в виде имени элемента и номера его входа (после фиксации входных данных запроса 4).

Проверка введенных данных заключается в анализе ответа на следующие вопросы:

1) является ли введенное имя (схемы или элемента) идентификатором языка Си?

2) не является ли введенный тип элемента новым, т. е. отсутствующим в справочнике?

3) не является ли введенный тип элемента новым, т. е. отсутствующим в справочнике?

4) не является ли имя элемента в запросах 4 и 5 новым, т. е. ранее не вводимом в запросе 3?

5) не являются ли номер входа и номер выхода новыми, т. е. остутствующими в справочнике системы?

6) не описан ли уже номер входа в запросе?

7) все ли входы всех компонентов устройства описаны?

При отрицательном результате проверки выдается сообщение об ошибочном ответе и запрос повторяется снова. При вводе специальных символов пользователь имеет возможность выхода из запроса или диалога с запоминанием введенной информации и получения подсказки о неописанных цепях, возможность корректировки и просмотра введенных данных об элементах и связях, а также просмотра данных справочника.

Опишем теперь алгоритм работы препроцессора. Рассмотрим в общих чертах функционирование препроцессора в режиме событийного моделирования синхронного устройства. Событийный алгоритм моделирования

представляет собой итерационный процесс вычисления значений выходов элементов, критерием окончания которого служит совпадение их значений на двух соседних итерациях. Он использует в своей работе очередь активных элементов, программу работы элемента и список его последователей и предшественников, значения его входов и выходов [2, 3]. Используя только адреса этих переменных, создан неизменный программный текст этого алгоритма. Препроцессор включает этот и другие тексты в программу путем выдачи строк вида *include* (имя текста). Получение адресов переменных осуществляется в два шага: конкретное выделение памяти под переменные, описывающее устройство, и организация массива адресов этих переменных. Обозначим через a число битов, необходимых для кодировки значений выбранного алфавита моделирования (в ИКСМ заложена возможность моделирования от 2-значного до 16-значного алфавита).

Выделение памяти под выходы и внутренние переменные компонентов устройства осуществляется путем выдачи в формируемую программу в цикле по всем элементам J ($<J>$ — имя элемента J) таблицы ЭЛЕМЕНТЫ строки вида int$D_<J>[Q][a]$, $M\_<J>[S][a]$.

Определение имен программы используемых типов элементов в устройстве осуществляется записью в программу в цикле по всем компонентам устройства, если этот тип еще не встречался, строки вида int P().

Препроцессор далее организует массивы адресов определенных выше переменных. Например, адреса подпрограмм работы элементов, адреса полей значений выходов элементов и значений переменных состояния строятся путем выдачи в программу соответственно следующих операторов

$$\text{int}^{*} A\_P()[] = \{P_1, P_2, \ldots\};$$

$$\text{int}^{*} A\_D[] = \{D\_<1>, \ldots, D\_<J>, \ldots\};$$

$$\text{int}^{*} A\_M[] = \{M\_<1>, \ldots, M\_<J>, \ldots\}.$$

После построения программы препроцессор вызывает компилятор, который ее обрабатывает. Редактор связи, используя библиотеку объектных модулей элементов системы, получает выполнимый модуль конкретного устройства. Описанный подход позволяет получить программу, в которой наглядно представлены структура связей между компонентами, память под значения на этих связях и т. п. Однако для больших устройств размер программы будет слишком велик для обработки ее компиляторами для языка Си. Поэтому выделение памяти происходит динамически в два этапа: выделяется общее необходимое поле памяти для всех элементов схемы и далее выделяется память для массива указателей элементов, которые автоматически инициализируются соответствующими адресами в общем поле. Объем текста программы при таком подходе практически одинаков для различных устройств.

Опишем теперь процесс построения программных модулей элементов. Алгоритм функционирования исследуемого устройства задается либо структурной моделью (схемой из элементов уже включенных в систему), либо программной моделью, которая может описывать поведение устройства на различных уровнях.

Следуя терминологии [4], это соответствует структурной и алгоритмической моделям БИС. Предполагается, что программы, описывающие функционирование элементов, составляются пользователем (или генерируются автоматически, о чем речь пойдет далее) исходя из его представления в виде некоторой автоматной модели. Это означает возможность определения его выходов и новых значений переменных состояния на основе знания входных сигналов и предыдущих состояний

ISSN 0235-3474. ТД и НК, 1994, № 2

устройства. Такая модель является достаточно общей, так как в нее укладываются все концепции систем классической физики [5]. Описанное устройство может быть использовано в других схемах в качестве автономного компонента. Эта возможность обеспечивается путем применения специального режима работы системы, в котором происходит автоматическое занесение информации об устройстве в справочник системы, как об обычном элементе. Далее создается программа определенного вида для этой схемы, которая компилируется, и ее объектный модуль записывается в библиотеку элементов. Разработка программ функциональных моделей БИС является трудоемкой процедурой, требующей высокой квалификации пользователя. Поэтому более предпочтительным является путь разработки структурных моделей. Элементы ПЗУ и ПЛМ, функционирование которых зависит от информации, занесенной в каждый конкретный экземпляр микросхемы, представлены единой программной моделью, настройку параметров которой осуществляет сам пользователь. При разработке программной модели элемента могут быть использованы специальные функции, позволяющие выполнять многозначные логические операции, определять наличие фронтов и срезов сигналов, имитировать многозначный условный оператор. Элементы с двунаправленными контактами описываются специальным образом.

Подпрограммы элементов имеют единый список параметров: поле, в начале которого записаны значения входов элементов и за ними после отработки программы располагаются новые значения выходов; поле выходов; поле переменных состояния элемента; алфавит моделирования и текущие параметры функционирования системы.

Остановимся на некоторых особенностях реализации ИКСМ. Особое положение в системе занимают программные модули универсального элемента задержки, двунаправленного ключа и элемента одноразрядной шины, которые позволяют обеспечить автоматическую обработку устройств, имеющих магистральные структуры и элементы с двунаправленными выводами.

Логическое моделирование с задержками наиболее точно отражает поведение логических схем цифровых вычислительных и управляющих систем, но предъявляет высокие требования к скорости моделирования. Компилятивный метод моделирования является более быстродействующим, чем интерпретативный, однако вследствие трудностей учета задержек область его применения ограничивается в основном комбинационными и синхронными последовательностными схемами [6].

Известно, что алгоритм асинхронного событийного моделирования осуществляет учет связей между сигналами на линиях устройства с помощью двух списков данных — очереди активных в данный момент времени элементов и очереди будущих событий [2, 3]. Очередь будущих событий необходима для учета сигналов, появляющихся с задержкой по отношению к текущему такту времени. Она является по своей сущности динамической структурой данных и существует множество способов ее организации и обработки [3, 6]. Одним из неприятных моментов ее ведения является обработка так называемого колеса переполнения, связанная с ограниченностью оперативной памяти ЭВМ и возможными большими временами задержек распространения.

В ИКСМ реализован способ, позволяющий в значительной мере обойти эти трудности. Основан он на известном представлении элемента в виде модели, состоящей из последовательно соединенных идеального логического элемента и инерционной задержки [6, 7]. Суть способа заключается, во-первых, в автоматическом построении промежуточного описания логической схемы из исходного путем внесения специальных элементов, функционирование которых позволяет имитировать задержки различных типов (инерционных, распространения), и во-вторых, в

генерации программы, являющейся компилятивной реализацией алгоритма событийного моделирования преобразованной схемы. Необходимо отметить, что только программная модель схемы является компилятивной. Программные модули элементов схемы разрабатываются автономно и подключаются на этапе редактирования.

Сущность функционирования универсального элемента задержки состоит в постоянной своей самоактивизации (путем занесения в очередь активных элементов) до тех пор, пока он хранит в своей динамической памяти изменяющиеся значения своего входа и моментов их передачи на свой выход. Заметим, что функции задержки, которые раньше имитировались с помощью очереди будущих событий, выполняются самим элементом. Это повлекло за собой исчезновение самой очереди и использование уже реализованного событийного алгоритма моделирования преобразованной схемы.

Автоматическое преобразование исходной схемы для инерционной задержки и задержек распространения происходит в соответствии с рис.4.6 из [6]. Соответствующие изменения исходных таблиц ЭЛЕМЕНТЫ и СВЯЗИ описываются следующим образом.

Для каждой записи таблицы СВЯЗИ (d, k, d_i, k_i): образуется уникальное имя элемента задержки для элемента d на выходном контакте k путем конкатенации идентификаторов d и k; в таблицу ЭЛЕМЕНТЫ добавляется элемент, имеющий имя d_k и тип универсального элемента задержки распространения;

каждая запись таблицы СВЯЗИ (d, k, d_i, k_i) заменяется на две записи $(d, k, d_k, 1)$ и (d_k, Σ, d_i, k_i).

Состояние элементов задержек описывается соответствующей для каждого типа задержек структурой внутренних параметров, приведенных ниже. Входными данными программных модулей элементов задержек являются их внутренние параметры, значение входа и номер такта модельного времени M.

Состояние элемента задержки распространения описывается следующими внутренними переменными: T — модельное время занесения значения; Y — значение выхода в момент T; E — указатели на первый (последний) элемент цепи событий для данного элемента задержек; F — флаг типа включения элемента задержки в очередь активных событий; D_T — значение задержки.

Цепь событий для элемента задержки описывает в момент модельного времени M изменения входа за время $(M-D_T, M-1)$ и представляет собой однонаправленный список, состоящий из структур данных $E = (T_E, X_E, P_E)$, где T_E — модельное время изменения; X_E — значение в момент T_E; P_E — указатель на следующее событие E.

Результат моделирования с инерционными задержками зависит от места проявления самой задержки, т. е. на входе или выходе элемента. В обоих случаях используется универсальный элемент инерционной задержки, функционирование которого заключается в игнорировании входного сигнала, если он подается менее, чем D_T тактов модельного времени.

Достоинства описанного подхода по сравнению с интерпретативной реализацией заключаются в уменьшении сложности организации временного колеса и колеса переполнения, поскольку построение и функционирование сводится к программной реализации и выполнению во время моделирования автономных модулей элементов распространения и в легкости внесения модификаций в систему моделирования.

Недостатком данного подхода можно считать наличие непроизводительных временных затрат на проверку срабатывания элементов задержек на каждом такте модельного времени, аналогом которой является программный опрос процессором состояния портов [8]. Это существенно, если очередь активных элементов содержит только элементы задержек с

Имя схемы	Время моделирования исправного устройства, с	Полный объем выходных реакций, Кб	Число вентилей/линий	Число входов/выходов	Объем таблиц, описывающих устройство	Число неисправностей	Время моделирования с неисправностями
c5315	46	317	2609/4508	178/123	161	5350	3 ч. 51
s8772	18	173	1419/1774	206/56	76	4198	43
c2670	22	198	1567/2215	233/140	88	2747	53
c432	6	34	204/342	36/7	12	524	1.5
c880	8	77	470/754	60/26	26	942	4.5
c1355	11	76	620/1095	41/32	36	1574	14.5
c1908	17	136	939/1522	33/25	54	1879	30

большими их значениями. Поскольку такая ситуация является типичной, то в программную реализацию внесена возможность ее обнаружения и автоматического увеличения модельного времени на количество тактов, в течение которых в модели не происходит изменений. Оно равно минимальному значению задержек среди элементов активной очереди.

В системе реализован параллельный метод моделирования константных неисправностей. Использование его позволяет определять полноту теста для этого класса неисправностей и синтезировать псевдослучайным методом тесты.

ИКСМ имеет возможность использовать описание схемы, созданное графическим редактором РСАД, а также ее программы визуализации результатов моделирования.

Остановимся еще на одной из возможностей системы. В ИКСМ предусмотрен специальный режим, в котором осуществляется подсчет показателей контролепригодности. Упомянутые показатели, в число которых входят управляемость, наблюдаемость и тестируемость, вычисляются по формулам, введенным в [9] на основе информационного подхода. Вычисление показателей осуществляется для каждой линии устройства в отдельности и для всего устройства в целом. Система выдает на печать список линий устройства с наихудшими показателями. По требованию пользователя система выдает рекомендации по оптимальному размещению в устройстве заданного числа контрольных точек, облегчающих локализацию неисправностей.

Приведем некоторые эксплуатационные характеристики системы. Данные о быстродействии системы представлены в таблице. Объектами моделирования являлись эталонные устройства конференций ISCAS-85, 89 [10]. Моделирование осуществлялось на псевдослучайном тесте длиной 125 наборов инструментальной машиной IBM PC AT/386 с тактовой частотой 40 МГц. Размер сгенерированного текста программы и соответствующего выполнимого модуля примерно одинаковы для всех устройств и равны соответственно 2,5 и 56 Кбайт.

Необходимо отметить, что данные колонок 7, 8 в таблице для моделирования с неисправностями характерны для последовательностных устройств (хотя указанные устройства в основном комбинационные, и для них быстродействие можно поднять на порядок, используя параллельную подачу входных воздействий в машинном слове [11]). Так, на тесте из 110 входных наборов моделировалось реальное устройство, содержащее 70 микросхем серий 555, 155 и 586. В качестве инструментальной использовались ЭВМ IBM PC-286. Моделирование исправного устройства потребовало 20 с, а моделирование всех константных неисправностей (около 1100) — 16 мин.

Библиотека содержит около 350 типов элементов различных серий (155, 555, 561, 500 и т. п.). Система требует для своей работы 3 Мбайт дисковой памяти и наличия системы программирования C/C++.

СПИСОК ЛИТЕРАТУРЫ

1. *Ганн Л.* Система моделирования на языке VHDL // Электроника. — 1990. — № 2. — С. 81—85.
2. *Савельев П. В., Коняхин В. В.* Функционально-логическое проектирование ВИС / Под ред. Г. Г. Казеннова. — М.: Высш. шк., 1990. — 156 с.
3. *Иоффе М. И.* Диагностирование логических схем. Алгоритмы моделирования и автоматического синтеза тестов. — М.: Наука, 1989. — 158 с.
4. *Иванников А. Д.* Моделирование микропроцессорных устройств. — М.: Энергоатомиздат, 1990. — 144 с.
5. *Пешель М.* Моделирование сигналов и систем. — М.: Мир, 1981. — 300 с.
6. *Киносита К., Асада К., Карацу С.* Логическое проектирование СВИС / Пер. с япон. — М.: Мир, 1988. — 309 с.
7. *Миллер Р.* Теория переключательных схем. Последовательностные схемы и машины. — М.: Наука, 1971. Т. 2 — 304 с.
8. *Проектирование* микропроцессорной электронно-вычислительной аппаратуры: Справочник / В. Г. Артюхов, А. А. Будняк, Ю. В. Лапий и др. — Киев: Техніка, 1988. — 263 с.
9. *Андрюхин А. И., Сперанский Д. В.* Энтропийная мера контролепригодности дискретных устройств // УСиМ. — 1991. — № 8. — С. 23—30.
10. *Brglez F., Fujiwara H.* «A Neutral Neetlist oa 10 Combinational Benchmark Circuits and a Target Translator in Fortran», Proc. IEEE Int. Symposium on Circuits and Systems; Special Session on ATRG and Fault Simulation, June 1985.
11. *Андрюхин А. И., Сперанский Д. В.* Об одном алгоритме псевдослучайного синтеза тестов для больших комбинационных схем // Автоматика и вычислительная техника. — 1989. — № 3. — С. 93—94.

Поступила в редакцию
14.06.93

Ин-т прикладной математики,
Донецк

УДК 621.019.3