method for time domain analysis of large-scale integrated circuits // IEEE Trans. on CAD of IC and Syst.— 1982.— 1, N 3.— P. 131.
5. **Mokari-Bolhassan M. E., Smart D., Trick T. N.** A new robust relaxation technique for VLSI circuit simulation // Digest of Technical Papers of IEEE Inter. Conf. on CAD, ICCAD-85, Nov. 18—21, Santa Clara, Colif.— Washington: IEEE Computer Society Press, 1985.— P. 26.
6. **Marong G., Sangiovanni-Vincentelli A.** Waveform relaxation and dynamic partitioning for the transient simulation of large-scale bipolar circuits // Ibid.— P. 32.
7. **Безносов Г. П., Ефименко В. В., Загоруйко А. С., Стукалин Ю. А.** Пакет программ

УДК 53.072 : 681.3

В. С. ЛОПАТИН, И. Е. МЕДВЕДКОВА, В. Е. МЕЖОВ

(Воронеж)

ОСНОВНЫЕ ПРИНЦИПЫ ПОСТРОЕНИЯ ПАКЕТА ИЕРАРХИЧЕСКОГО МОДЕЛИРОВАНИЯ ПРИАМ

Унифицированные интерактивные комплексы автоматизации проектирования изделий электронной и вычислительной техники типа «Кулон-4» позволяют осуществлять разработку схем различной сложности. Такие системы должны включать пакеты прикладных программ автоматизации всех этапов проектирования.

Программы поддержки функционально-логического, схемотехнического и топологического проектирования для комплексов этого класса уже разработаны [1, 2]. В настоящее время созданы лингвистические средства (входной язык) и пакет программ иерархического алгоритмического моделирования (ПРИАМ), который осуществляет поведенческое моделирование вычислительных систем.

Характерной чертой разработанного входного языка является возможность **иерархического** описания проекта, т. е. для моделирования проекта вычислительной системы на любом уровне абстракции (поведенческом, регистровом, логическом) используется единый метод ее описания.

Определим основные понятия, используемые при описании проекта. Проект вычислительной системы представляется как совокупность «черных ящиков», называемых **компонентами,** соединенных между собой линиями связи (в дальнейшем они будут называться **линиями**). Описание поведения «черного ящика» называется **моделью**.

Для моделирования проекта системы требуется язык, который давал бы возможность задавать: структуру проекта в целом; модели, т. е. поведение компонентов проекта; управление процессом моделирования.

В пакете ПРИАМ описание структуры проекта и алгоритма функционирования компонентов, из которых он состоит, разделено. При этом преследовались следующие цели:

— достижение большей наглядности, что сокращает затраты на подготовку исходной информации и внесение изменений;

— обеспечение возможности раздельной трансляции описания структуры и поведения, что дает значительный выигрыш как на этапе трансляции в случае внесения изменений в исходную информацию, так и при последующей программной обработке данных;

— раздельное описание структуры, позволяющее проводить автома-

тический переход с одного уровня детализации на другой как для всего проекта, так и для отдельных его фрагментов.

Далее более подробно будут рассмотрены вопросы задания поведения компонентов.

В пакете ПРИАМ описание моделей проводится на паскалеподобном языке АЛОС. Язык АЛОС представляет пользователю возможности: определения способа функционирования (поведения); обеспечения многоуровневого представления, т. е. описания различных компонентов с разной степенью детализации; спецификации информационных данных (описания потоков данных); обеспечения независимости описания моделей от конкретной реализации.

Язык АЛОС, наряду с выразительными возможностями, имеющимися в Паскале, включает также проблемно ориентированные операторы, облегчающие моделирование вычислительных систем. Эти операторы позволяют, в частности, организовывать параллельные процессы, осуществлять управление процессами посредством других процессов, вводить общие для всего проекта многофазные синхронизаторы, синхронизировать активизацию процессов и т. д.

В пакете ПРИАМ применяется компилятивный метод обработки входных конструкций. Общая структура пакета приведена на рис. 1. Исходное описание проекта обрабатывается независимо двумя трансляторами. Транслятор АЛГКОНТ проводит синтаксический и семантический контроль алгоритмического описания моделей, а транслятор СТРКОНТР контролирует описание структуры проекта. Результаты работы обоих трансляторов помещаются в базу данных. Затем из базы данных программой СБОРКА выбираются те модели и структуры, которые нужны для моделирования текущего проекта, и из них компилируется **имитатор,** т. е. файл на языке Паскаль, содержащий сами модели и вспомогательные программы, позволяющие проводить моделирование.

Далее этот файл обрабатывается стандартными системными программами и вместе с постоянной частью образует **моделятор** (см. рис. 1), т. е. программу в машинных кодах, скомпилированную для данного проекта и осуществляющую его моделирование. Более детально моделятор будет рассмотрен ниже.

При разработке пакета ПРИАМ были поставлены следующие задачи обеспечения:

1) возможность интерактивного управления процессом моделирования — прерывания, отслеживания, печати информации о процессе моделирования, изменения значений на линиях проекта и т. д.;

2) моделирование проектов, содержащих широкий диапазон уровней абстракции, что, в частности, предполагает наличие в рамках одного проекта линий с произвольными типами данных (от битовых до массивов и записей);

3) предоставление средств для моделирования сложных алгоритмов поведения моделей, в том числе и моделирование процессов, находящихся в состоянии ожидания, прерывания, их активизация в зависимости от выполнения оговоренных в описании условий или изменения состояний линий проекта;

4) оптимальное использование памяти при моделировании.

Наиболее трудным моментом при организации моделирования в пакете ПРИАМ является вопрос о поддержке всего диапазона типов данных, который язык АЛОС допускает для линий. Эту проблему удалось решить путем разделения программы моделирования на две части, условно названные постоянной и переменной.

Переменная часть, называемая также **имитатором,** создается на базе моделей, описанных пользователем на языке АЛОС, т. е. в значительной степени определяется пользователем. Она является результатом обработки пакетом ПРИАМ файла исходных данных, созданного пользователем на языке АЛОС.

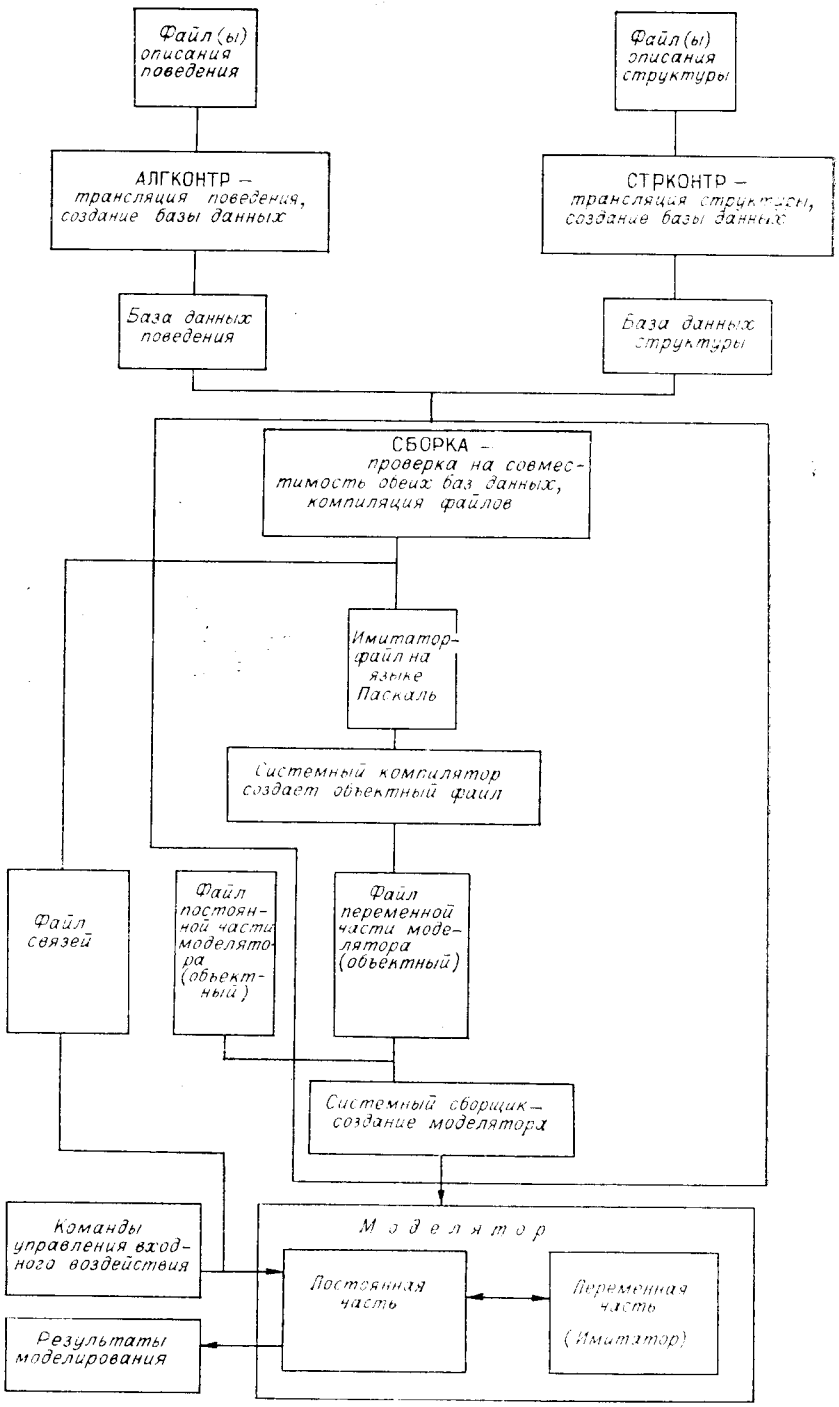

Рис. 1

Вся работа с типами данных проводится формализованными процедурами, входящими в состав имитатора. Эти процедуры компилируются для каждого типа линий данного проекта. Они осуществляют следующие функции: выделение для каждой линии проекта необходимого объема машинной памяти для хранения значений данных того типа, который был определен для нее пользователем; анализ и планирование событий на линиях схемы; вычисление и переприсвоение значений данных на линиях.

Постоянная часть является одной из программ пакета ПРИАМ и недоступна пользователю. Эта общая для всех проектов часть, она организует сам процесс моделирования, в том числе и обращение к моделям компонентов.

Из постоянной части и имитатора посредством системных команд стандартными системными программами компилируется **моделятор** (см. рис. 1).

При определении функционального назначения каждой части моделятора преследовалась еще одна цель. Дело в том, что процесс создания моделятора сложный и длительный. Поэтому желательно было организовать моделирование в пакете ПРИАМ таким образом, чтобы избежать перекомпиляции моделятора в том случае, если модели, входящие в состав проекта, не изменялись. Для этого данные о конфигурации проекта не «зашиты» в моделятор, как чаще всего бывает при компилятивном методе моделирования, а считываются моделятором каждый раз перед началом моделирования из файла связей, содержащего эту информацию.

Рассмотрим функции основных частей **моделятора.**

Постоянная часть организует процесс моделирования и выполняет функции контроллера, т. е. выполняет следующие действия: считывает из файла связей данные о структуре схемы и организует динамическое выделение памяти из всех компонентов и линий проекта; осуществляет продвижение модельного времени и организует планирование и активизацию событий; выполняет все функции по управлению процессом моделирования, а именно: а) запрашивает интерактивно или в пакетном режиме команды пользователя и обрабатывает их; б) осуществляет прерывание, отслеживание моделирования и просмотр результатов моделирования; в) управляет входными и выходными файлами моделятора.

В скомпилированном моделяторе эта часть является одинаковой для проектов, так как сама организация моделирования и внутренняя структура данных позволяют всю работу, связанную с переменной информацией, определяющей типы линий, глобальные и локальные переменные проекта, функционирование моделей и т. п., вынести в переменную часть, а информацию о структуре проекта, как уже отмечалось, моделятор получает из специального файла, называемого файлом связей.

Переменная часть выполняет все функции, связанные с конкретной реализацией моделей в проекте. Эти функции можно определить следующим образом: динамическое отведение памяти для компонента; выделение динамической памяти для текущего и планируемых значений линий; проверка, занесение, планирование и печать значений на линиях, т. е. осуществление всех действий по работе с типами данных; проверка выполнения условий, оговоренных в описании модели для активизации процессов или продолжения прерванных процессов; выполнение процессов, из которых состоит модель, а также организация печати их имен.

Рассмотрим процесс создания имитатора. Имитатор создается программой СБОРКА (см. рис. 1). Список моделей, включаемых в имитатор, либо непосредственно задается пользователем в интерактивном режиме, либо, по умолчанию, включает те и только те модели, которые необходимы для моделирования конкретного проекта.

Программа СБОРКА проводит проверку моделей, входящих в список, на совместимость, а также на соответствие их структурным описаниям.

Чтобы избежать создания моделятора, когда модели, включаемые в него, не модифицируются по сравнению с прошлым вариантом имитатора, при создании имитатора создается **протокол имитатора.** Он представляет собой файл, содержащий, в частности, список моделей, вошедших в имитатор, и время его создания. Прежде чем создать новый имитатор проекта, программа СБОРКА проверяет при помощи протокола имитатора наличие и состав уже существующего для данного проекта имитатора. Новый имитатор проекта будет создаваться в случае, если после создания существующего была изменена, по крайней мере, одна из моделей

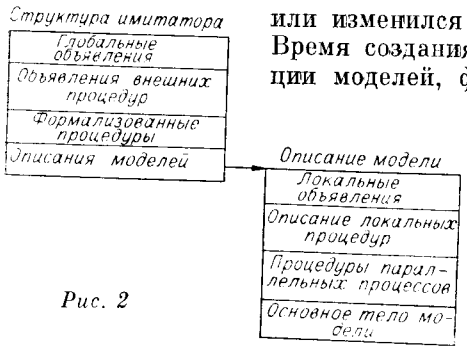

Рис. 2

или изменился состав моделей, включенных в него. Время создания имитатора, как и время модификации моделей, фиксируется соответственно в протоколе имитатора и в базе данных. Очевидно, что при такой организации пользователь имеет возможность работать с одним и тем же имитатором, если его задача состоит, например, в получении структуры схем определенного класса (ТТЛ, КМОП и т. д.) и он имеет в своем распоряжении готовый имитатор с набором удовлетворяющих его моделей.

Имитатор представляет собой программный модуль на языке Паскаль.

Как уже отмечалось, имитатор является основной составной частью моделятора. Он выполняет при моделировании всю работу, связанную с линиями проекта, функционированием моделей и динамическим распределением памяти. Упрощенно можно считать, что имитатор состоит (рис. 2) из раздела глобальных объявлений (типы данных, типы линий и т. д.), объявлений внешних процедур, которые используются моделями (они являются глобальными по отношению к моделям проекта), набора формализованных процедур для организации работы с различными типами данных (их функции уже были описаны) и описания моделей. Описание каждой модели (см. рис. 2), в свою очередь, состоит из локальных объявлений (выводов и переменных модели), описания локальных процедур, которые используются процессами или основным телом только данной модели, и процедур, осуществляющих выполнение параллельных процессов и основного тела модели, из которых состоит модель.

Рассмотрим теперь подробнее процесс создания имитатора.

Каждая модель, занесенная в базу данных поведения, состоит из двух частей: таблицы идентификаторов и кодов поведения.

Таблица идентификаторов содержит закодированную информацию о всех идентификаторах, используемых в описании модели. В начале работы программа СБОРКА последовательно анализирует таблицы идентификаторов всего списка моделей. Из таблиц выбирается информация о типах линий, синхронизаторах, глобальных константах, глобальных и локальных типах, формируются внутренние таблицы. При этом осуществляется проверка глобальных идентификаторов различных моделей на совместимость. Если используемые в проекте модели не совместимы (например, встречаются одинаковые идентификаторы с различными назначениями или значениями), то выдается соответствующее сообщение, а имитатор не создается.

На базе таблицы истинности создаются глобальные объявления для всего модуля и локальные объявления для каждой модели, а также формируются те из формализованных процедур, которые выполняют при моделировании всю работу с линиями (резервирование памяти для хранения новых и текущих значений на линиях, присваивание, изменение значений на линиях, распечатка значений на соответствующих типах линий).

Формирование описаний моделей осуществляется на основе **кодов поведения** модели, которые полностью характеризуют функционирование модели. В имитаторе каждое описание модели организовано как процедура, включающая в себя, в свою очередь, процедуры параллельных процессов и основного тела модели. Их назначение в том, что они организуют: а) работу параллельно действующих процессов, для чего синхронные и одновременные действия преобразуются в поток неодновременно выполняемых последовательных одиночных команд; б) замену

объявлений и операторов языка АЛОС рядом конструкций, операторов и подпрограмм на языке Паскаль; в) возможность запуска, прерывания и останова в любом месте процессов при моделировании (для этой цели некоторые операторы видоизменяются таким образом, что появляется возможность входа в тело циклов и условных операторов).

Формированием процедур описания моделей и заканчивается процесс создания имитатора.

Приведенная организация пакета ПРИАМ позволяет моделировать проекты сложных вычислительных систем. Средняя скорость моделирования на машинах типа «Электроника 82» составляет 1000 активностей/с. Под активностями здесь понимаются элементарные действия, например: планирование линии или процесса, присвоение значения переменной и т. п. Однако следует отметить, что назначение пакета состоит в отладке новых систем и алгоритмов. Следовательно, к его основным достоинствам следует отнести широкие возможности по описанию проекта и управлению процессом моделирования.

СПИСОК ЛИТЕРАТУРЫ

1. **Лобов И. Е., Межов В. Е., Чевычелов Ю. А.** Системы комплексной автоматизации проектирования изделий электронной и вычислительной техники на основе 32-разрядных супермини- и микроЭВМ // Школа-семинар молодых ученых и специалистов «Актуальные проблемы создания интеллектуальных САПР РЭА и СБИС».— Воронеж: ВПИ, 1989.
2. **Лобов И. Е., Межов В. Е., Чевычелов Ю. А.** Логическое моделирование и генерация тестов цифровых схем в системе «Кулон» // Там же.

Поступила в редакцию 12 октября 1989 г.