HLCCAD – среда редактирования, симуляции и отладки аппаратного обеспечения

Михаил Долинский,
Вячеслав Литвинов,
Андрей Галатин,
Игорь Ермолаев

dolinsky@gsu.unibel.by

Введение

Стремительное развитие цифровых электронных технологий вызывает потребность в адекватном развитии средств автоматизации. В данной работе описывается программно-аппаратная система HLCCAD (High Level Chip Computer-Aided Design), которая предназначена для эффективной разработки аппаратного обеспечения функционально-сложных цифровых систем.

Система HLCCAD (http://NewIT.gsu.unibel.by/hlccad) в течение ряда лет разрабатывается в Гомельском государственном университете им. Ф. Скорины (Республика Беларусь), внедрена в учебный процесс, многократно использовалась на практике, демонстрировалась на белорусских республиканских и международных выставках, в том числе на CeBIT (Ганновер, Германия) в марте 2002 года. В ноябре 2002 года отмечена специальным дипломом на международной выставке «Перспективные технологии и системы» (г. Минск).

Таблица 1

	PeakVHDL	Active–CAD	Max+PLUSII	Renoir
ModelSim				
Редактирование	1*	2	1	2
Многоуровневый откат	–	+	–	+
Разработка «сверху вниз» с возможностью симуляции	–	–	–	–
Разработка «снизу вверх»	+	+	+	+
Командная разработка проекта	–	–	–	–
Способы создания	1	3	2	5
HDL (Hardware Description Language)	+	+	+	+
HLL (High Level programming Language)	–	–	–	–
Схема	–	+	+	+
Машина состояний	–	+	–	+
Таблица истинности	–	–	–	+
Блок-схема	–	–	–	+
Моделирование	0	0	1	0
Создание интерактивной среды отладки	–	–	–	–
Создание пакетной среды отладки	–	–	–	–
Язык тестов	–	–	+	–
Моделирование МП/МК	–	–	–	–
Моделирование мультипроцессорных систем	–	–	–	–
Анализ	2	2	1	5
Отладчик схемы/HDL	+	+	–	+
Диаграммы	+	+	+	+
История	–	–	–	+
Дамп памяти	–	–	–	–
Регистры и флаги	–	–	–	+
Биты	–	–	–	+

* — количество символов «+» в соответствующем столбце для нижеследующей группы признаков

1. Сравнительный анализ распространенных средств автоматизации разработки аппаратного обеспечения

По мнению авторов, наиболее распространенными среди отечественных разработчиков средствами автоматизации проектирования цифровых систем на сегодняшний день являются: Max+Plus II фирмы Altera, Renoir+ModelSim фирмы Mentor Graphics, PeakVHDL фирмы PeakVHDL, Active-CAD фирмы Aldec.

В таблице 1 приведен сравнительный анализ важнейших функциональных возможностей вышеперечисленных систем. Сравнение проводилось по четырем существенным для разработчиков группам харатеристик: «Редактирование», «Способы создания компонентов проектов», «Моделирование», «Анализ».

Легко заметить, что в каждой из групп имеются недостатки, присущие практически всем указанным продуктам, а именно:

В категории «Редактирование» во всех системах отсутствуют следующие возможности:

а) разработка «сверху вниз» с возможностью симуляции проектов на любом уровне детализации;
б) поддержка коллективной разработки проекта.

В категории «Способы создания компонентов проектов» во всех системах отсутствует возможность создавать компоненты с использованием языков программирования высокого уровня и динамически подгружаемых библиотек.

В категории «Моделирование» во всех анализируемых системах отсутствуют следующие возможности:

- создание интерактивной среды отладки;
- создание пакетной среды отладки;
- моделирование микропроцессоров/микроконтроллеров (МП/МК);
- моделирование мультипроцессорных систем.

В категории «Анализ» во всех системах отсутствуют средства анализа дампа памяти — как памяти данных, так и, тем более, памяти программ.

На момент проведения сравнительного анализа в нашем распоряжении отсутствовали специализированные средства разработки для микросхем семейств Xilinx, однако можно с уверенностью утверждать, что сделанные в результате сравнительного анализа выводы справедливы и для них.

Все это в совокупности с ростом в потребности разработки систем, использующих МП/МК, и привело к разработке и развитию HLCCAD.

2. Методология разработки аппаратного обеспечения с использованием HLCCAD

2.1. Основные этапы разработки

HLCCAD эффективно поддерживает следующие этапы разработки аппаратного обеспечения:

1. Создание интерактивной среды отладки для динамической проверки состояния и свойств реализованного проекта в текущий момент. Для реализации среды отладки используются различные виды устройств, симулирующие ввод-вывод (клавиатура, кнопочные панели, индикаторы, дисплеи), а также отладочные средства, позволяющие изменять значения переменных (прямо на схеме, в окне просмотра дампа памяти и др.).
2. Создание пакетной среды отладки для организации перманентного регрессионного (на всем продолжении разработки) тестирования проекта. Для реализации данного механизма разработан язык тестов для проверки алгоритма работы отдельной схемы устройства, а также язык сценариев для обеспечения автономного пакетного режима тестирования списка устройств.
3. Разработка высокоуровневой модели для окончательного уточнения технического задания на разработку, а также автоматизации создания среды тестирования и непосредственно тестов. Предусматривается создание поведенческих моделей цифровых устройств на произвольном языке программирования высокого уровня с использованием всех доступных средств операционной системы, на которой происходит исполнение модели.
4. Подготовка к распределенной коллективной разработке для повышения производительности команды разработчиков в результате распараллеливания создания независимых блоков устройства. Проект, представленный в начальный момент высокоуровневой моделью, разбивается на несколько «подпроектов», для каждого из которых необходимо произвести дальнейшую декомпозицию.
5. Интерактивная асинхронная декомпозиция создаваемого аппаратного обеспечения до синтезируемых блоков (устройств).
6. Регрессионное тестирование, предусматривающее процесс тестирования разрабатываемого проекта на всех этапах разработки как «сверху вниз», так и «снизу вверх», с помощью разработанных интерактивных и пакетных сред.
7. Автоматическая генерация синтезируемого описания на языке VHDL по разработанным и отлаженным схемам цифровых устройств.

На рис. 1 отображена схема последовательности выполнения основных этапов разработки синтезируемых описаний аппаратного обеспечения вычислительных систем.

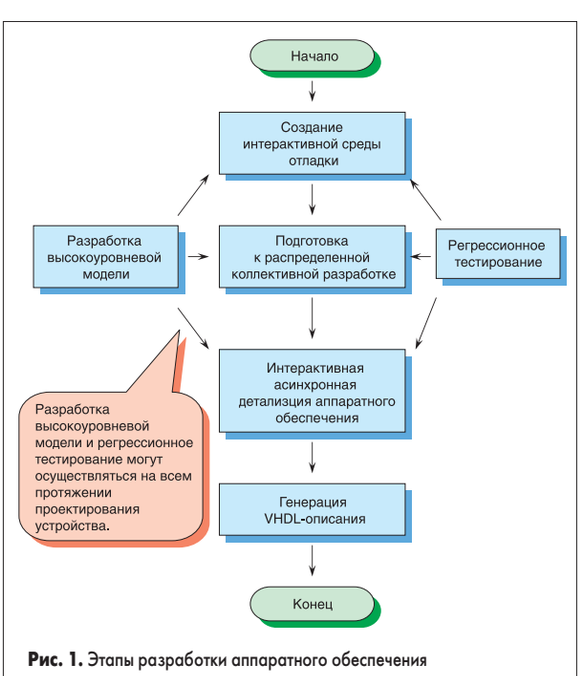

Рис. 1. Этапы разработки аппаратного обеспечения

2.2. Создание интерактивной среды отладки

Процесс создания интерактивной среды отладки подразумевает создание схемы устройства, состоящей из следующих элементов:

- функциональных блоков устройства, реализующих вычислительный алгоритм;
- устройств ввода, для подачи входных воздействий;
- устройств вывода, для обеспечения визуализации результатов симуляции функциональной части устройства.

На рис. 2 приведены примеры привычных для разработчиков устройств вывода: линейная шкала, семисегментные индикаторы, матричные дисплеи, включенных в комплект поставки HLCCAD.

Рис. 2. Визуализация при помощи индикаторов

Возможно использование уже имеющихся в HLCCAD моделей устройств ввода-вывода, либо создание своих моделей устройств на языках программирования высокого уровня, позволяющих построить 32-битную библиотеку (DLL).

2.3. Пакетная среда регрессионного тестирования

Для реализации возможности пакетного тестирования предусмотрены язык тестовых воздействий и язык сценария.

С помощью языка тестовых воздействий разработчик имеет возможность в нужный момент модельного времени указать тестовые воздействия, подаваемые на входы устройства, и эталонные значения для выходов. Файл теста представляется в виде текстовых команд.

В случае, когда воздействия на схему производятся интерактивно, можно автоматически получить файл тестов по результатам моделирования. В процессе симуляции производится сохранение всех значений устанавливаемых на контактах. Разработчик может выбрать необходимый набор контактов, значения которых требуется тестировать. Дополнительно можно определить временной диапазон для сохранения. В результате будет построен файл тестовых воздействий, который можно использовать для автоматического тестирования схемы устройства, содержимое которого может в дальнейшем модифицироваться.

После определения тестовых воздействий для схем устройства можно построить «сценарий» автоматического тестирования.

Полученный файл сценария можно использовать для пакетного тестирования проектов.

Во время исполнения пакетного тестирования системой ведется файл результатов исполнения, так называемый LOG-файл. Этот файл по умолчанию автоматически сохраняется в каталог с файлом сценария. По содержимому этого файла можно отследить весь процесс исполнения.

Таким образом, используя язык тестовых воздействий и язык сценария, можно обеспечить регрессивное тестирование всего проекта на всех этапах разработки «сверху вниз» и «снизу вверх».

Ввиду обширности темы предполагается написание отдельной статьи «Верификация проектов в HLCCAD: создание пакетной среды тестирования с помощью языка тестовых воздействий и языка сценария».

2.4. Разработка высокоуровневых моделей разрабатываемого аппаратного обеспечения

Для обеспечения разработки «сверху вниз», а также обеспечения отладки уже на первых этапах разработки и высокой скорости симуляции необходим высокоуровневый механизм создания моделей устройств. Наиболее удачным подходом является использование технологии COM, которая обеспечивает эффективную интеграцию приложений, написанных на разных языках программирования.

Для реализации ответной реакции на события системы в интерфейсе моделей предусмотрены процедуры и функции, которые являются обработчиками событий:

- AutoStart — процедура вызывается в нулевой момент времени и предназначена для инициализации внутренней памяти и переменных модели;
- OnChanged — процедура вызывается после изменения значения на входах;
- function ExecAfter : int64 — функция вызывается по истечении определенного промежутка времени. Эта функция вызывается в нулевой момент модельного времени. После обработки события функция возвращает величину времени в пикосекундах, через которую ее следует снова вызвать. Если значение меньше нуля, то данная функция больше вызываться не будет.

Если объект является моделью процессора, то для отладочных средств системы определяются функции для получения дополнительной информации:

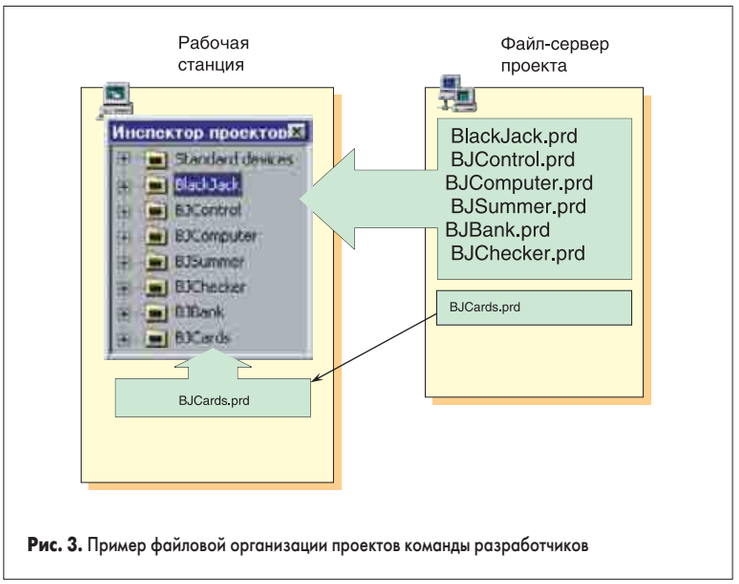

Рис. 3. Пример файловой организации проектов команды разработчиков

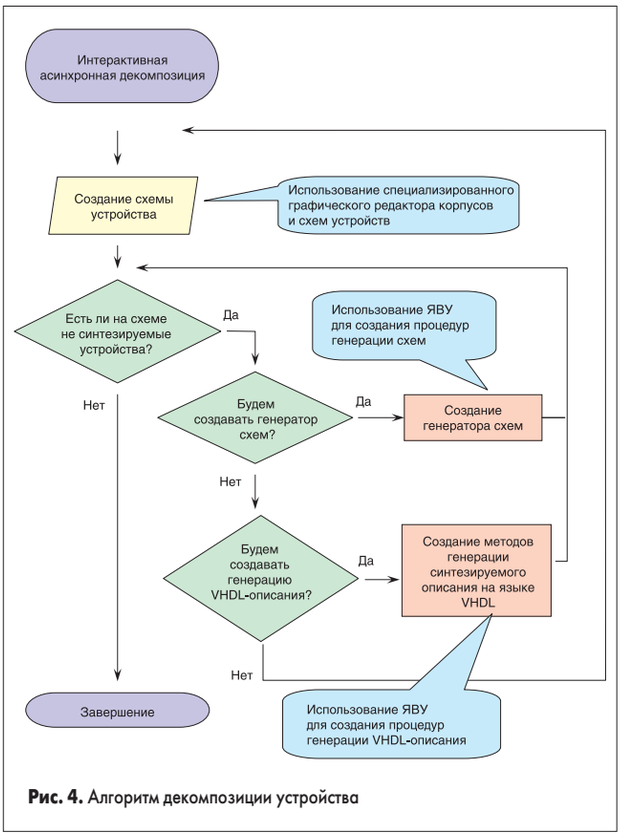

Рис. 4. Алгоритм декомпозиции устройства

- внутреннее устройство памяти модели, регистры, биты и флаги;
- мнемоника инструкции по коду;
- размер инструкции по коду;
- размер кодовой памяти;
- количество выполненных инструкций за текущий сеанс моделирования.

Для генерации синтезируемого VHDL-описания модель должна уметь передать системе соответствующее описание в текстовом виде. Для реализации данного механизма предусмотрены процедуры, позволяющие передать необходимую информацию системе в виде текста.

Ввиду обширности темы предполагается отдельная статья на тему «Технология разработки высокоуровневых моделей компонентов в системе HLCCAD».

2.5. Подготовка к распределенной коллективной разработке

При наличии нескольких групп разработчиков руководителю проекта предоставляется возможность распараллелить процесс разработки.

Таблица 2. Устройства библиотеки «Standard»

Логика	NOT	Логический инвертор
	OR	Логическое ИЛИ
	XOR	Логическое исключающее ИЛИ
	AND	Логическое И
Комбинационные схемы		
	CD	Приоритетный шифратор
	DC	Дешифратор
	MS	Мультиплексор
	DMS	Демультиплексор
Константы	0	Генератор логического «0»
	1	Генератор логического «1»
	BF	Тристабильный буфер
Память	T	Триггер
	RG	Регистр
	CT	Счетчик
	ROM	Постоянное запоминающее устройство
	RAM	Оперативное запоминающее устройство
Математические операции	CMP	Компаратор
	SUM	Сумматор
	MUL	Умножитель

Вначале создается первый уровень детализации разрабатываемого аппаратного обеспечения и модели всех компонентов этого уровня. Далее производится перенос компонентов, предполагающих дальнейшую детализацию, в отдельные файлы проекта. В сети выделяется файл-сервер, куда копируются все файлы проектов и файлы тестовых воздействий.

Каждая команда разработчиков получает задание разработать схему устройства для определенного проекта. Соответствующий файл проекта копируется с сервера на рабочую станцию (рис. 3). Каждая команда работает с локальной версией определенного проекта, а все остальные файлы проекта, которые содержат несинтезируемые модели остальных блоков, подключаются с файл-сервера. Это позволяет производить симуляцию «полной» схемы устройства.

Таким образом, несколько команд разработчиков могут параллельно заниматься разработкой различных блоков одного устройства. Преимущество данного подхода перед другими методами заключается в том, что разработчики могут проводить симуляцию всей схемы, не имея детального описания остальных блоков. Детализация разрабатываемого блока осуществляется с помощью синтезируемых устройств различных библиотек, либо с помощью аналогичных высокоуровневых блоков, представленных COM-моделями.

По завершении работ команда разработчиков может скопировать новый файл проекта в определенный каталог на файл-сервере. Тесты, полученные на ранних этапах разработки, могут быть использованы для проверки работоспособности устройства.

2.6. Интерактивная иерархическая асинхронная декомпозиция создаваемого аппаратного обеспечения и генерация синтезируемых VHDL-описаний

Декомпозиция разрабатываемого устройства или его блока для получения симулируемых компонент осуществляется одним из двух способов.

1) Создание схемы с использованием синтезируемых устройств. Таковыми являются устройства из базовой параметризированной библиотеки устройств «Standard.prd». В таблице 2 приведен список устройств, входящих в состав этой библиотеки. Разработчик может изменять добавлять или удалять контакты, изменять их параметры и разрядность в соответствии с функциональностью устройства. Высокоуровневая модель автоматически настраивается на алгоритм работы, соответствующий созданному УГО корпуса и его параметрам.
2) Создание схемы при помощи высокоуровневых (поведенческих несинтезируемых) блоков, каковыми являются разрабатываемые поведенческие модели на языках высокого уровня.

Конечная цель проектирования аппаратного обеспечения в HLCCAD — получение соответствующего синтезируемого VHDL-описания всего проекта. Поэтому в конце концов все высокоуровневые модели, не замененные композицией элементов из стандартной библиотеки, должны обеспечить синтезируемое описание своего компонента одним из следующих способов:

1) Генерация синтезируемого описания на языке VHDL.
2) Генерация схем.

С помощью COM-интерфейсов HLCCAD предоставляет возможность создания и модификации схемы устройства. С помощью функций интерфейсов генератор может создавать схему, добавлять контакты и устройства, изменять параметры устройств, создавать новые устройства на схеме и их схемы. Этот способ удобен в том случае, если необходимо многократно использовать на схеме устройства, алгоритм которых слегка изменя-

Таблица 3. Использование основных объектов в блоках

Типы объектов комплекса	Хранилище	Редактор	Моделирование	Визуализация	Генерация	Описание
TFileStream	X					Доступ к файлу проекта
TResourceFile	X					Доступ к ресурсам проекта
TProjectItem	X					Элемент структуры проекта
TProject	X					Доступ к элементам проекта
TProjectInspector	X	X				Окно Инспектора проекта
TCorpus	X	X	X	X	X	УГО корпуса устройства
TContact		X	X	X	X	Контакт корпуса
TCorpusText		X				Текст на корпусе
TCorpusParameter		X	X		X	Пользовательский параметр устройства
TCorpusEditor		X				Редактор корпуса
TScheme	X	X	X	X	X	Схема устройства
TIOContact		X	X	X	X	Контакт схемы
Tnet		X		X		Связи на схеме
TConnect		X		X		Элемент связывания
TSchemeEditor		X				Редактор корпуса
TPremodelingLine			X			Результат компиляции связей (непосредственное связывание линий контактов)
TContactModel			X			Модель контакта
TCorpusModel			X			Модель корпуса
TSchemeModel			X			Модель схемы
TVHDLBuilder					X	
TVHDLTest					X	Конвертор тестовых воздействий в VHDL-устройство
TVHDLTestVec					X	Конвертор тестовых воздействий в файл векторов

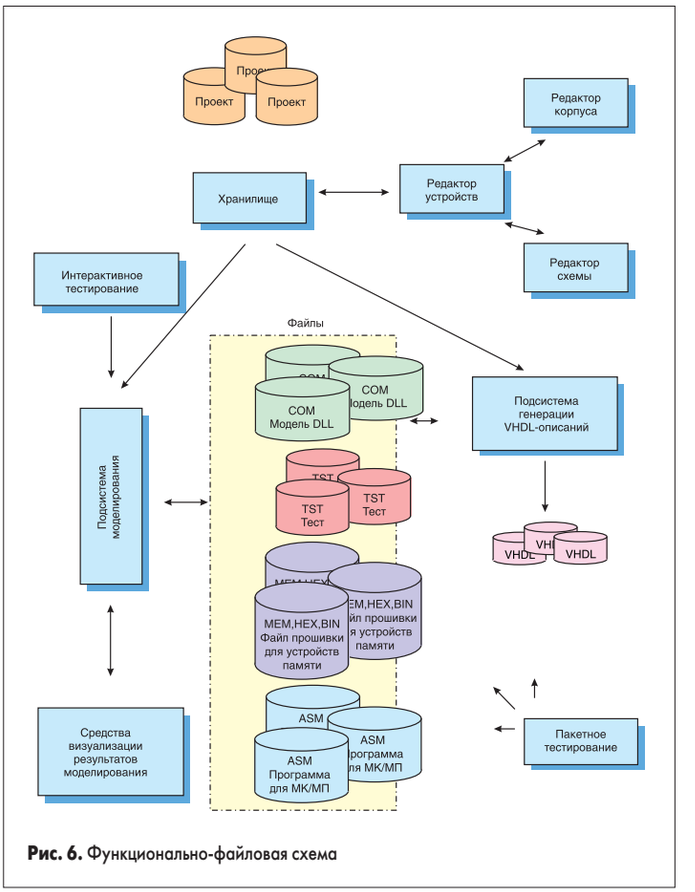

Рис. 6. Функционально-файловая схема

ется в зависимости от значений параметров (например, устройства, реализующие фильтры). Следовательно, можно создать настраиваемую высокоуровневую модель устройства и генератор схемы для него.

Для реализации генератора схем может использоваться произвольный язык программирования высокого уровня (ЯВУ), позволяющий построить 32-битную библиотеку (DLL).

На рис. 4 изображен алгоритм декомпозиции устройства. Процесс декомпозиции продолжается до тех пор, пока схема устройства не будет представлена в виде блоков, для которых автоматически можно построить синтезируемое VHDL-описание.

3. Организация системы HLCCAD

3.1. Программно-файловая структура комплекса HLCCAD

Реализованный набор программных средств состоит из средств ввода и редактирования моделей аппаратного обеспечения, системы моделирования, средств визуализации результатов моделирования, а также средств генерации VHDL-описания.

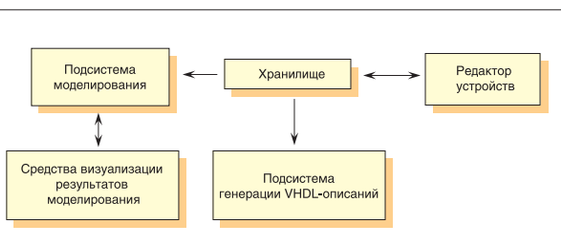

Рис. 5. Структурная схема

Основная часть комплекса реализована на языке Object Pascal в среде Delphi 5. Общее число строк исходного текста 115 000.

Все файлы с компонентами комплекса группируются по каталогам.

Каталог BIN содержит исполняемый файл, файлы скриптов (цветовые настройки для подсветки синтаксиса в текстовом редакторе), файлы фраз для различных языков, динамически подгружаемые библиотеки (текстовый редактор, синтаксический анализатор, окна визуализации от системы WInter).

В каталоге Help содержатся файлы помощи, пошаговых учебников и демонстрации.

И, наконец, в каталоге Projects содержатся файлы проектов.

Структурная схема разработанных средств отображена на рис. 5.

Базовым блоком комплекса является «Хранилище». Объекты этого блока предоставляют доступ к данным проектных файлов. Данные представляются в виде самостоятельных объектов TProject, TCorpus, TScheme и др.

Блок «Редактор устройств» содержит объекты, которые позволяют создавать и модифицировать УГО корпуса и схему устройств.

Блок «Моделирование» содержит библиотеку объектов, осуществляющих компиляцию устройств и их симуляцию.

Блок «Генерации» предоставляет средства для построения по схеме устройства описания на синтезируемом подмножестве языка VHDL.

Использование основных типов объектов в блоках отражено в таблице 3. На рис. 6 изображена функционально-файловая схема разработанных средств.

4. Технология эксплуатации HLCCAD

4.1. Интерактивная иерархическая декомпозиция

Система HLCCAD позволяет создавать устройства как «снизу вверх», так и «сверху вниз». При разработке «сверху вниз» устройство описывается как набор блоков и связей между ними. При этом для каждого блока создается поведенческая модель на ЯВУ. Такую схему можно смоделировать и, проанализировав результаты, либо дальше детализировать блоки (если устройство работает верно, и есть необходимость в дальнейшей детализации), либо исправить ошибки в самой схеме или высокоуровневых моделях (если таковые имеются). Таким образом, разбиение блоков ведется до тех пор, пока в конечном результате в нижних уровнях не окажутся устройства, модель которых может сгенерировать синтезируемое описание на VHDL.

Для создания нового проекта необходимо выбрать пункт меню главного окна «Файл | Новый проект». После указания имени файла новый проект будет добавлен в окно «Инспектора проектов». Для того чтобы создать новое устройство, нужно вызвать локальное меню над именем проекта и выбрать пункт «Новое устройство», указав в диалоговом окне его имя.

После этого разработчик может модифицировать корпус и схему устройства. Для открытия редактора необходимо вызвать локальное меню над именем устройства и выбрать пункт «Редактор».

Рис. 7. Окно редактора УГО корпуса

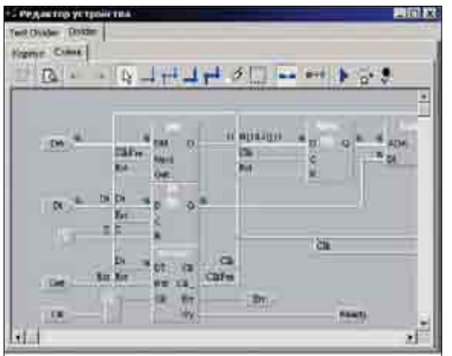

Рис. 8. Окно редактора схемы

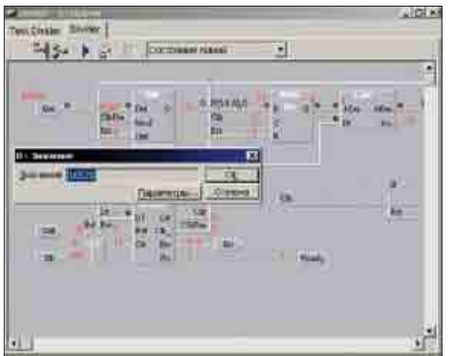

Рис. 9. Пример интерактивного ввода

Окно редактора устройства состоит из двух подредакторов: редактор УГО (условное графическое обозначение) корпуса (рис. 7) и редактор схемы (рис. 8).

Процесс создания устройства обычно начинается с корпуса. Редактор корпуса устройства позволяет разработчику изменять размер УГО, добавлять, удалять и модифицировать контакты или текст на корпус.

Для установки режима моделирования устройства необходимо нажать на кнопку «Тип модели». В диалоговом окне можно изменить параметры моделирования:

- Режим моделирования:
  - схема — модель устройства представлена в виде схемы;
  - DLL — модель устройства представлена моделью на ЯВУ;
  - BlackBox — модель устройства представлена DLL, однако описание модели в виде схемы может быть сгенерировано автоматически по установленным параметрам устройства.
- Параметры моделирования:
  - скрыть значения корпуса — отключение сохранения трассы значений контактов корпуса;
  - скрыть значения схем — отключение сохранения трассы для корпуса и всех элементов на схемах этого устройства;
  - не моделировать — отключение устройства из процесса моделирования. Позволяет отладить часть схемы без устройства, не изменяя саму схему;
  - параметры памяти — позволяет установить параметры инициализации всей внутренней памяти устройства.

После создания УГО корпуса и установки параметров моделирования, при условии, что модель будет представлена в виде схемы, разработчик может скопировать контакты корпуса на схему.

Схема представляется в виде совокупности устройств и связей между ними. Добавление устройств из других проектов производится при помощи технологии Drag & Drop. Для этого нужно выбрать устройство в окне «Инспектора проектов», нажать левую кнопку мыши, переместить указатель мыши на схему и отпустить кнопку. Для того чтобы создать новое устройство, необходимо вызвать локальное меню и выбрать соответствующий пункт.

Изменение УГО корпусов устройств после добавления на схему также осуществляется в окне редактора корпуса. Для этого в окне редактора предусмотрена возможность переключения редактируемого корпуса. После сохранения содержимого редактора новое УГО корпуса автоматически обновляется на схеме.

Редактирование связей осуществляется тремя способами:

- рисование отдельных линий;
- рисование ломаных;
- использование невидимой шины.

При рисовании линий разработчик определяет тип линии (одинарная или шина) путем нажатия соответствующей кнопки на панели. Рисование отдельных линий происходит по следующему алгоритму:

- нажимаем левую кнопку мыши на схеме;
- перемещаем указатель мыши;
- отпускаем кнопку мыши.

Рисование ломаных происходит так:

- щелкаем левой кнопкой мыши на схеме;
- перемещаем указатель мыши;
- для рисования отрезка ломанной щелкаем левой кнопкой мыши и повторяем с предыдущего этапа;
- для отмены нажимаем правую кнопку мыши.

При рисовании линий и шин допустимо использование именованных шин. Для ввода имени шины необходимо дважды щелкнуть левой кнопкой мыши над линией и в диалоговом окне ввести соответствующие названия.

Использование невидимой шины позволяет уменьшить число линий на схеме. Для ввода линий контакта в шину необходимо ввести имя, под которым эти линии будут доступны. Вызов окна ввода имени производится после двойного щелчка левой кнопкой мыши над контактом корпуса или схемы.

При разработке «снизу вверх» предоставляется возможность объединять уже готовые устройства, устанавливая связи между ними.

4.2. Интерактивное, пакетное и регрессионное тестирование

Наличие развитых средств отладки позволяет разработчику аппаратного обеспечения существенно сократить сроки разработки.

Тестовые воздействия на моделируемое устройство можно подавать несколькими способами.

Во-первых, разработчик может интерактивно изменять значения на контактах. Для этого необходимо открыть соответствующее окно отладчика схем. Затем двойным щелчком мыши вызвать окно изменения значения. После ввода нового значения достаточно нажать кнопку «Ok» (рис. 9).

Во-вторых, реализован пакетный режим тестирования. Разработчик может указать набор тестовых воздействий в виде специализированного текстового файла (рис. 10). Для этого в окне редактора схемы нужно вызвать локальное меню и выбрать пункт «Параметры». В окне «Параметры схемы» необходимо указать соответствующее имя файла. Перед запуском на моделирование необходимо установить в параметрах моделирования флаг «Использовать тесты».

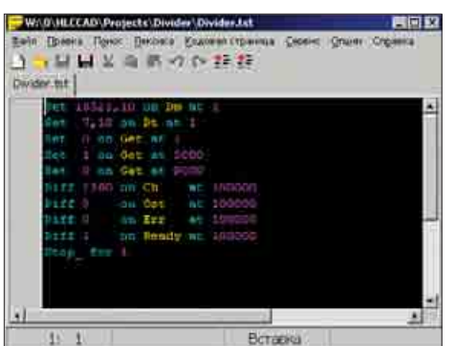

Рис. 10. Пример файла тестовых воздействий

В-третьих, разработчик может создавать интерактивную среду отладки. В качестве генератора входных воздействий будет выступать устройство. В процессе моделирования устройство может создавать дополнительное окно, в котором при помощи клавиатуры или мыши может изменять значения выходов устройства на схеме. Аналогичным образом на схеме могут находиться устройства, выполняющие функции тестирования или визуализации.

Для реализации пакетной среды отладки разработчик может самостоятельно создать файл сценария либо воспользоваться специализированным конструктором. Конструктор с помощью диалогового окна позволяет определить параметры тестирования:

- Проекты для тестирования.
- Каталог тестирования.
- Виды тестирования:
  - тестирование схемы;
  - тестирование сгенерированного VHDL-описания.

Дополнительно можно указать режим полного тестирования. В результате исполнения, после моделирования схемы устройства по трассе значений на контактах будет автоматически построен файл тестов, по содержимому которого будет сгенерировано VHDL-описание для более полного тестирования.

Предусмотрена возможность вызова приложения для обработки статистики по результатам тестирования.

С помощью описанных выше способов тестирования обеспечивается регрессионное тестирование аппаратного обеспечения.

4.3. Визуализация результатов моделирования

Анализ результатов моделирования осуществляется при помощи широкого диапазона различных окон визуализации.

Структура моделируемого устройства отображается в окне «Дерево модели» (рис. 11). С помощью этого окна разработчик может открывать другие окна анализа для элементов дерева. Кроме этого, с помощью данного окна производится отображение результатов мониторинга моделирования.

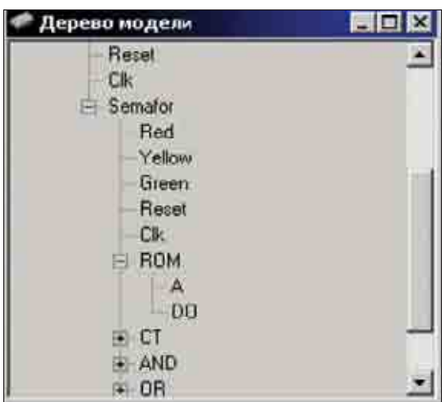

Рис. 11. Окно отображения дерева моделей устройства

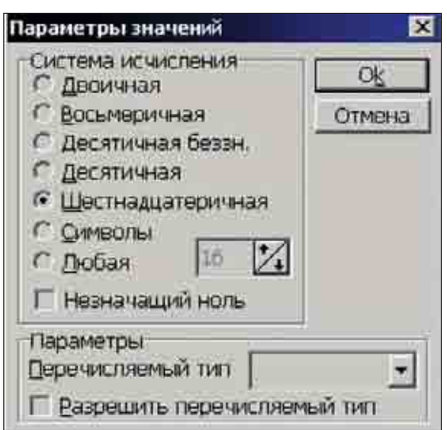

Рис. 12. Окно параметров значений

Для визуализации значений на схеме разработчик может воспользоваться окном отладчика схем (рис. 13).

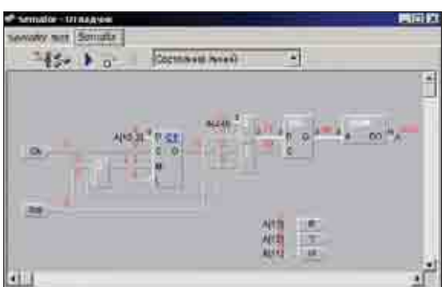

Рис. 13. Окно отладчика схем

Отладчик отображает схемы моделируемых устройств и значения, установленные на контакты. Значения на контактах могут отображаться в произвольной системе счисления от 2 до 256 (рис. 12). Для любого контакта можно установить перечисляемый тип, который позволяет задать список замен значений на текстовую строку. Разработчику доступны механизмы установки параметров значений для всех контактов определенного корпуса на схеме либо для всех контактов на схеме. Кроме этого, любое значение можно перенести в произвольное место на схеме.

Окно позволяет производить навигацию по дереву устройства. Разработчик может открыть схему для любого корпуса на данной схеме либо подняться на уровень выше от текущей.

Для изменения анализируемого модельного времени можно воспользоваться окном «Навигатор по времени» (рис. 14). Навигатор позволяет изменять время при помощи полосы прокрутки. Допускается числовой ввод времени.

Рис. 14. Окно навигатора времени

С помощью дополнительных кнопок можно изменять время на определенный интервал в любом направлении, а также производить переход к ближайшему модельному времени, в который произошло изменение значения на контактах схемы, открытой в окне отладчика схем. Кроме этого, имеется режим автоматического изменения анализируемого времени любым из выше описанных способов.

Для просмотра трассы значений для контактов предусмотрены два окна анализа: временных диаграмм и истории контактов.

Окно временных диаграмм (рис. 15) является традиционным для систем разработки аппаратного цифрового обеспечения. В левой части

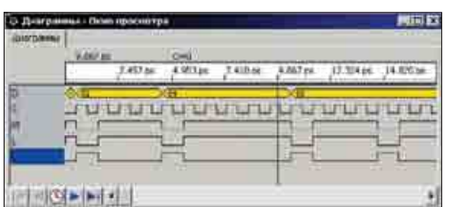

Рис. 15. Окно просмотра временных диаграмм

отображаются имена контактов, а в правой — трасса значений в виде диаграммы. Имеются возможности масштабирования, фильтрации и поиска значения. Средства навигации аналогичны окну «Навигатор времени».

Окно истории контактов (рис. 16) отображает трассу значений в виде таблицы. Каждый столбец таблицы содержит значения контакта в момент модельного времени, определенного строкой. Каждая строка таблицы всегда отличается от предыдущей, то есть информация сжата.

Для анализа значений внутренних переменных моделей предназначено окно просмотра

Рис. 16. Окно истории контактов

Рис. 17. Окно просмотра регистров и флагов

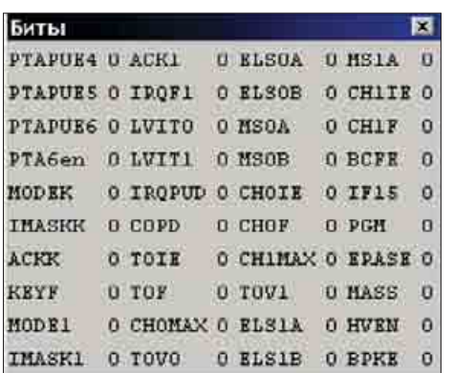

Рис. 18. Окно просмотра битов

регистров и флагов (рис. 17), а также окно просмотра битов (рис. 18). Значения соответствуют модельному времени установленному в «Навигаторе времени».

Просмотр содержимого дампа памяти осуществляется двумя способами (рис. 19). Пер-

Рис. 19. Окна просмотра дампа памяти

вое окно позволяет изменять настройки окна произвольным образом: изменять систему счисления адреса и данных, устанавливать произвольный размер слова памяти. Второе окно специфично при отладке программного обеспечения для МП или МК.

Также для отладки программного обеспечения предназначены окно отладчика по исходным текстам (рис. 20) и окно дизассемблера (рис. 21).

Рис. 20. Окно отладчика по исходному тексту

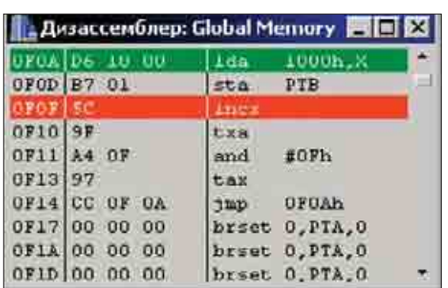

Рис. 21. Окно дизассемблера

Поддерживается несколько режимов отладки по исходному тексту:

- исполнение по инструкциям;
- исполнение по командам (без погружения в подпрограммы);
- исполнение до выхода из подпрограммы;
- исполнение до определенного адреса или строки исходного текста;
- исполнение с учетом точек останова.

Сохранение трассы значений позволяет отлаживать программу с любого модельного времени смоделированного диапазона по несколько раз за один сеанс. Кроме этого, поддерживается отладка (исполнение программы) в обратном направлении.

Для просмотра содержимого стека предусмотрено окно, изображенное на рис. 22.

Реализованы механизмы автоматического расположения окон визуализации в виде горизонтальной и вертикальной «черепицы».

Рис. 22. Окно просмотра стека

4.4. Моделирование микропроцессоров и отладка микропроцессорных систем

Добавление и настройка модели микропроцессоров осуществляется следующим образом. Каждый МП имеет свое УГО корпуса, которое добавляется на схему. Затем задаются связи между устройствами. Далее задается файл программы для МП. Таких способов несколько:

- программа в виде текста на соответствующем ассемблере;
- программа в кодах:
  - в Intel HEX формате;
  - бинарный файл;
  - внутренний формат памяти.

Программа МП может быть расположена как внутри устройства, так и во внешнем ПЗУ или ОЗУ, если микропроцессор поддерживает такой режим. Имя файла «прошивки» задается в дополнительных параметрах устройства. Имя параметра соответствует имени памяти устройства.

Во время отладки разработчик может анализировать содержимое объектов процессора, исполнять программу по исходному тексту или дизассемблеру.

Все допустимые средства отладки доступны из локального меню, которые вызываются над корпусом устройства:

- Регистры — окно отображает значения регистров и флагов МП, а также позволяет их модифицировать в процессе отладки.
- Биты — окно аналогично окну регистров.
- Память — окно отображает содержимое дампа произвольной памяти. Есть возможность изменения размера слова и ее содержимого.
- Дизассемблер — окно отображает дизассемблерный текст. Тип дизассемблера можно изменить командой локального меню на любой другой доступный в данный момент. Окно также позволяет модифицировать содержимое памяти. Текущий указатель (счетчик команд) может быть изменен на любой другой регистр. Для этого необходимо в окне регистров вызвать локальное меню для регистра и выбрать пункт «Как PC», а затем в локальном меню дизассемблера выбрать пункт «Новый PC».
- Окно исходного текста — доступно в случае наличия такового. Окно позволяет вести отладку программы по исходному тексту. Имеется возможность модификации программы и ее перекомпиляции и загрузки без выключения процесса моделирования.
- Окно стека — отображает содержимое стека. Указатель стека можно изменить тем же способом, что и указатель команд.
- Окно переменных — отображает значения всех переменных объявленных в исходном тексте программы. Окно доступно лишь при наличии окна исходного текста.

Все описанные выше команды также доступны из окон отладки по исходному тексту и дизассемблеру.

Исполнение программы доступно в нескольких режимах:

- исполнение по инструкциям;
- исполнение по командам (без погружения в подпрограммы);
- исполнение до выхода из подпрограммы;
- исполнение до указанного курсором адреса;
- исполнение с учетом точек останова.

Поддерживается возможность отладки по сохраненной трассе. Для этого необходимо изменить текущее модельное время в окне «Навигатора времени» и повторить исполнение программы без перекомпиляции и моделирования всей схемы.

Имеется возможность исполнения программы в обратном направлении по данным, сохраненным в трассе. Для этого необходимо «сбросить» флаг «Вперед».

5. Исследование производительности моделирования в HLCCAD мультипроцессорной гетерогенной системы

С целью исследования производительности моделирования и апробации средств отладки была разработана схема домашней сети, представляющая собой мультипроцессорную гетерогенную сеть микроконтроллеров для решения следующей прикладной задачи.

В некотором здании имеются датчики интенсивности расхода ресурсов: воды, газа, света и электричества. Требуется обеспечить сбор и визуализацию данных параметров. Для исследования каждый датчик соединен с микроконтроллером. Кроме того, имеется ведущий датчик, который будет обеспечивать сбор и вывод информации.

Все микроконтроллеры объединены в сеть, отвечающую спецификации I^2C (рис. 23). Сеть I^2C является стандартной для обмена данными между различными контроллерами.

Для имитации работы датчиков была разработана модель управления расхода ресурсом на языке программирования Object Pascal в среде Delphi. Во время моделирования каждая модель датчика создает окно управления расходом ресурса, где с помощью мышки можно изменять интенсивность расхода ресурсов.

Для имитации работы индикатора, на котором будет происходить визуализация, была разработана модель семисегментного индикатора.

Далее была разработана принципиальная схема для управляющего блока, состоящая из микроконтроллера Atmel 90S2313 и семисегментного индикатора. Для микроконтроллера была разработана управляющая про-

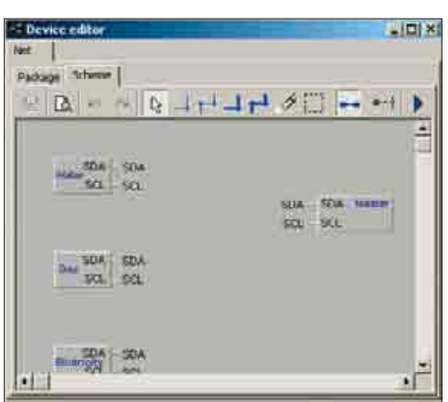

Рис. 23. Схема прикладной задачи HomeNet

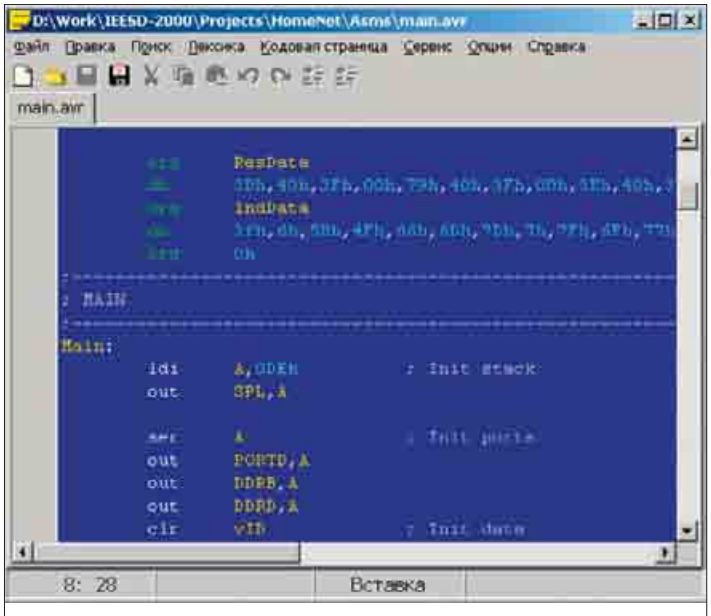

Рис. 24. Окно программы ведущего микроконтроллера Atmel 90S2313

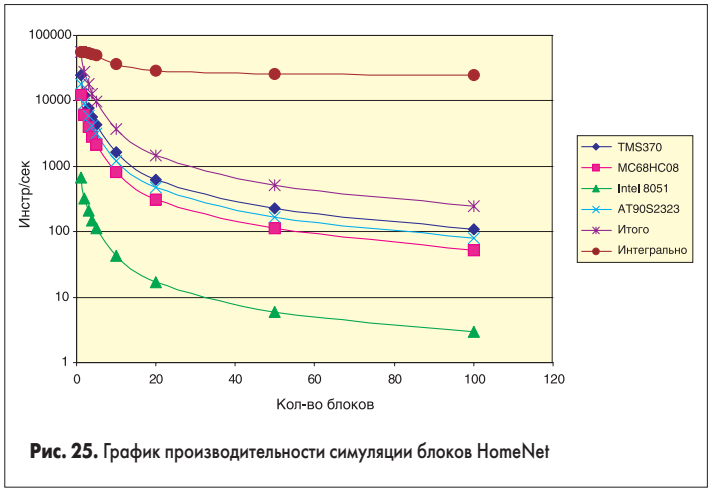

Рис. 25. График производительности симуляции блоков HomeNet

грамма, осуществляющая сбор информации с других блоков по протоколу I²C (рис. 24).

Были созданы принципиальные схемы для блоков, осуществляющих контроль ресурса. Затем разработаны программы для каждого микроконтроллера, реализующие сбор и передачу информации на ведущий блок.

Далее проведены серии тестов для определения производительности работы созданных средств и моделей процессоров. Были созданы схемы, включающие в себя блоки устройства HomeNet.

Результаты тестирования отображены в таблице 4. Каждый столбец отображает производительность каждого типа процессора (количество инструкций в секунду). Далее указана суммарная величина для отдельного блока HomeNet. И, наконец, общая интегральная производительность для всех блоков.

На рис. 25 изображены графики функций, отображающих зависимость производительности симуляции от количества блоков HomeNet (то есть фактически от числа моделируемых процессоров).

В результате исследования зависимости производительности симуляции от количества блоков была получена аппроксимированная функция: $IS=-((KB-6)^{1/5}-5)\times 9000$, где IS — количество инструкций в секунду, KB — количество блоков HomeNet.

Проведенный анализ интегральной производительности симуляции мультипроцессорной гетерогенной системы показывает, что при увеличении количества блоков от 1 до 20 производительность симуляции падает от 56442 до 28740 инструкций в секунду. Дальнейшее увеличение количества симулируемых процессорных блоков от 20 до 100 приводит лишь к незначительному снижению производительности симуляции, а именно от 28740 до 24400 инструкций в секунду, что составляет всего 15%. Это означает, что разработанные методы и средства обеспечивают эффективную симуляцию мультипроцессорных систем, включающих сотни процессоров.

Заключение

Данный материал посвящен системе (HLCCAD) отладки аппаратного обеспечения встроенных цифровых систем, которая создана в СНИЛ «Новые информационные технологии» Гомельского государственного университета им. Ф. Скорины. Разработка внедрена в учебный процесс университета, неоднократно использовалась при решении реальных проектных задач.

Система HLCCAD является одним из базовых компонентов комплекса IEESD-2000, предназначенного для сквозной совместной разработки программного и аппаратного обеспечения встроенных систем.

HLCCAD и IEESD-2000 многократно освещались в печати [1–3], экспонировались на белорусских и международных выставках, демонстрировались на семинарах в Москве, Санкт-Петербурге, Минске, Гомеле. В марте 2002 года в составе комплекса разработок система HLCCAD экспонировалась на CeBIT 2002 (Ганновер, Германия).

Литература

1. Долинский М. С., Литвинов В. А., Ермолаев И. Ю., Федорцов А. О. Пакет инструментальных комплексов сквозного совместного проектирования программного и аппаратного обеспечения встроенных мультипроцессорных систем // Chip News. 2002. № 5.
2. М. Долинский, В. Литвинов. Технология сквозного совместного проектирования программного и аппаратного обеспечения цифровых систем в IEESD-2000 // Chip News. 2002. № 3.
3. Долинский М., Литвинов В., Галатин А. Система высокоуровневого проектирования цифровых устройств HLCCAD // Электроника. 1998. № 10.

Таблица 4. Производительность симуляции

Кол-во блоков	1	2	3	4	5	10	20	50	100	200
TMS370	24860	12176	7867	5627	4324	1616	633	228	107	53
MC68HC08	12430	6088	3933	2814	2162	808	316	114	53	27
Intel 8051	679	324	210	150	115	43	17	6	3	2
AT90S2323	18473	9053	5849	4184	3214	1202	471	169	80	40
Итого	56442	27641	17859	12775	9815	3669	1437	517	243	122
Интегрально	56442	55282	53577	51100	49075	36690	28740	25850	24300	24400