Одновременная разработка программ и аппаратуры для встраиваемых систем

при помощи симулятора аппаратуры Vista Virtual Prototyping

Анна СЕРГЕЕВА
annserge@rambler.ru

В статье подробно рассмотрены основные функциональные возможности программного пакета Vista Virtual Prototyping компании Mentor Graphics. Он служит для создания и работы с готовыми и настраиваемыми пользователями виртуальными прототипами аппаратных платформ, в том числе и для распространенных в настоящее время платформ на базе процессоров ARM.

Введение

Каждый раз, когда в руки системному интегратору попадает новый комплект встраиваемой системы от зарекомендовавшей себя компании, он может уверенно приступить к его внедрению и рассчитывать на гарантию высокой надежности, хорошей производительности и оптимального режима работы этого устройства. Почему же так происходит? Как именно производитель обеспечивает высокое качество функционирования, столь критичное для подобных устройств?

Следует принять во внимание тот факт, что все наиболее распространенные итерационные методики разработки программного обеспечения, такие как Agile, scrum, extreme programming и другие [1], значительно сокращают цикл «проектирование → разработка → тестирование → внедрение» за счет того, что более эффективно реагируют на изменение требований, а также позволяют разработчикам и тестировщикам более оперативно получать обратную связь по вопросам качества.

А в реальной жизни разработчики мечтают о том, чтобы по такой методике они могли бы начинать отладку своей части программ еще на той стадии работы над проектом, когда физически аппаратуры как таковой еще не существует и аппаратное устройство само еще находится на этапе разработки. К тому же одновременная отладка дизайна такого устройства и работающих на нем программ в значительной мере позволяет добиться требуемой оптимизации и надежности конечного продукта.

Таким образом, в отрасли назрела необходимость в средствах создания быстрых, точных и дешевых моделей, симулирующих работу аппаратуры. И, как было сказано выше, необходимо предоставить эти модели команде разработчиков встроенного программного обеспечения на самых ранних стадиях процесса разработки.

Все разработчики встраиваемых устройств, в частности, такие производители, как IAR и Keil, имеют собственные симуляторы для разрабатываемых ими микроконтроллеров. Однако они симулируют, по сути, «голое железо», предоставляют ограниченный набор функций и ориентированы только на процессорное ядро. В отличие от них рассматриваемый в этой статье симулятор Vista Virtual Prototyping от Mentor Graphics предоставляет возможности симуляции аппаратуры, шины, производительности, работы операционной системы и программ. За счет этого Vista Virtual Prototyping на порядок более удобный и информативный, чем большинство симуляторов, представленных на рынке.

Симуляторы аппаратного обеспечения Vista Virtual Prototyping

Что же представляет собой симулятор аппаратного обеспечения? Это программный продукт, симулирующий работу реальной аппаратуры для таких уровней программного обеспечения, как операционная система, которая будет работать на этом оборудовании, и приложения, запускаемые под этой операционной системой. При этом операционная система и соответствующие приложения работают поверх симулятора, воспринимая его точно так же, как если бы это была настоящая аппаратура. Принцип подхода показан на рис. 1.

Осенью 2013 года компания Mentor Graphics, работающая в области автоматизации проектирования электроники (EDA) для электроники и электротехники, выпустила Vista Virtual Prototype Simulator — симу-

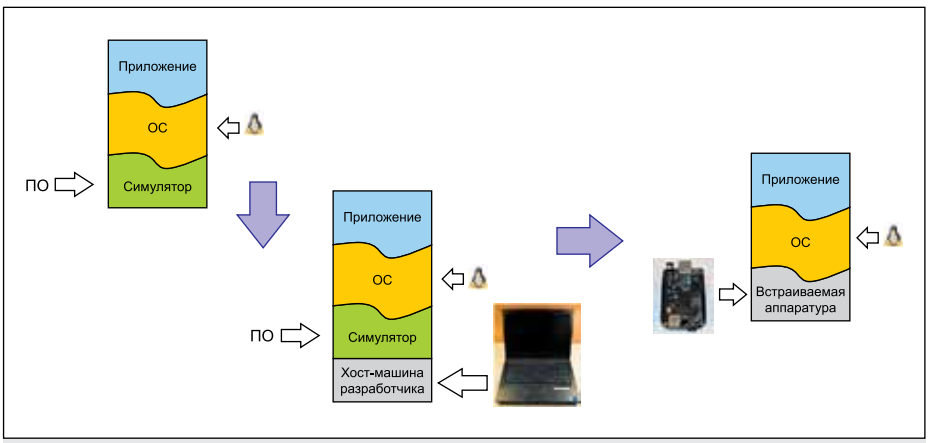

Рис. 1. Подход к применению симуляторов при отладке ПО на виртуальной и реальной аппаратуре

лятор работы с двухъядерным процессором ARM Cortex-A9. Его применение дает возможность отладки программ без привязки к физическому оборудованию.

Vista Virtual Prototyping использует первичную, абстрактную функциональную модель физического оборудования и предназначен для работы программистов до этапа реализации оборудования в «железе», а значит, предоставляет разработчикам встраиваемых программных продуктов возможность работы без затрат на настоящее аппаратное обеспечение.

Vista Virtual Prototyping более удобен для разработчика, чем стандартные пакеты поддержки плат (Board Support Packages, BSPs) 3]. Так, в нем реализованы возможности визуального представления виртуального оборудования и управления симуляцией, что дает возможность всесторонней взаимосвязанной отладки аппаратного и программного обеспечения, а также оптимизации разрабатываемых программ для достижения необходимых характеристик потребления мощности и производительности встраиваемого устройства.

Этот программный пакет включает два отдельных компонента, каждый из которых решает свой набор задач и предназначен для определенного круга инженеров.

Во-первых, это средства для создания платформы для моделирования на уровне транзакций (Transaction-Level Modeling, TLM). Эти средства предназначены для создателей платформ TLM (как правило, это системные архитекторы и разработчики SoC или проектировщики физического оборудования).

Во-вторых, это инструментарий для работы с готовым виртуальным прототипом. Его используют конечные пользователи виртуальных прототипов (как правило, это программисты и разработчики встроенных приложений).

Программа Mentor Graphics Sourcery CodeBench предоставляет возможности для управления моделированием, визуализации аппаратного обеспечения, а также совместной отладки аппаратного и программного обеспечения. А программа Mentor Graphics Sourcery Analyzer позволяет просмотреть данные обработки симулятором программного обеспечения и соответствующие аналитические данные работы аппаратного обеспечения.

Перейдем к более подробному рассмотрению этих компонентов.

Построение виртуальных прототипов

Для создания прототипов предназначена программа Mentor Graphics Sourcery CodeBench.

Она применяется для разработки встраиваемого программного обеспечения на языке C/C++ на базе таких архитектур, как ARM, PowerPC, ColdFire и др. Системные архитекторы используют ее для исследования и проверки архитектуры, результатом чего является создание платформ TLM и их дальнейшее использование.

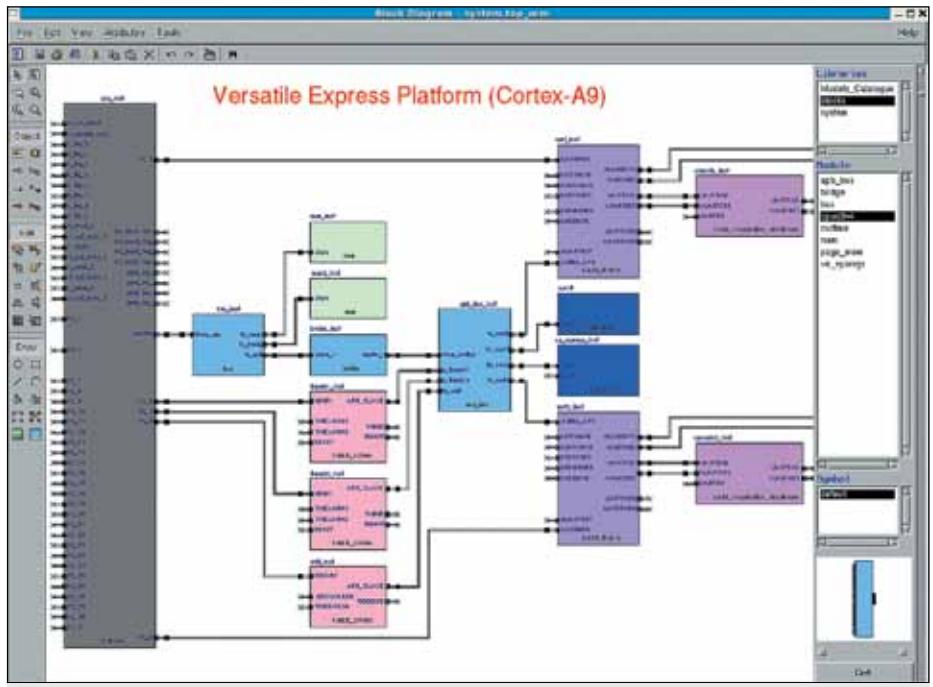

Рис. 2. Редактор блок-схем Vista Block Diagram editor

Как говорилось выше, виртуальный прототип строится на базе платформы TLM. TLM представляет первичную модель аппаратного обеспечения платформы и является абстрактным приближением к моделируемым функциональным узлам цифровых систем. Платформа TLM состоит из взаимосвязанных моделей уровня транзакций, представленных в формате структурированного кода на языке SystemC.

Vista поставляется вместе с набором библиотек, среди которых есть и библиотека предустановленных модулей уровня транзакций. Эти модули можно модифицировать или создавать новые.

С помощью построителя моделей (Vista Model Building) можно проводить подстройку атрибутов этих моделей (например, характеристики синхронизации и мощности), создавать новые модели TLM или же импортировать внешние модели в библиотеку моделей Vista.

С помощью редактора блок-схем (Vista Block Diagram editor) легко и удобно создавать платформу TLM. Интерфейс редактора приведен на рис. 2.

Каждой конкретной модели уровня транзакций соответствует свой графический символ, который система генерирует автоматически. Для определения топологии полного проекта платформы TLM достаточно соединить эти символы между собой в нужной последовательности. Так формируется схемное представление и описание всех объектов разрабатываемого проекта.

Каждый раз, когда выполняется операция сохранения проекта, редактор блок-схем Vista Block Diagram editor автоматически генерирует структурированный код на языке SystemC, соответствующий схемному представлению модели. Этот код можно просматривать и модифицировать в интерфейсе редактора блок-схем для любого экземпляра модели TLM. Также можно видеть на схеме отображение, соответствующее исходному коду любого иерархического блока модели.

Отметим, что такой подход является более наглядным, интуитивно понятным и простым в использовании по сравнению с кодированием в стандартном текстовом редакторе.

Конечный виртуальный прототип, созданный на основе платформы TLM либо с помощью диалога создания прототипа (Create Virtual Prototype dialog) в Vista GUI, как это показано на рис. 3, либо с помощью специальной команды vista_create_vp, является самостоятельным исполняемым файлом.

Vista Virtual Prototyping позволяет пользователю определять параметры прототипа в специальном файле — *Vista parameters*, а именно задавать значения по умолчанию для параметров разрабатываемого проекта. Позже, во время работы прототипа, эти значения могут быть модифицированы. (При этом повторное создание исполняемого файла не требуется.)

Виртуальные прототипы Vista можно запускать как на рабочих станциях с операционной системой Linux, так и из-под Windows. Отметим также, что создатель платформы может контролировать использование виртуального прототипа определенными поль-

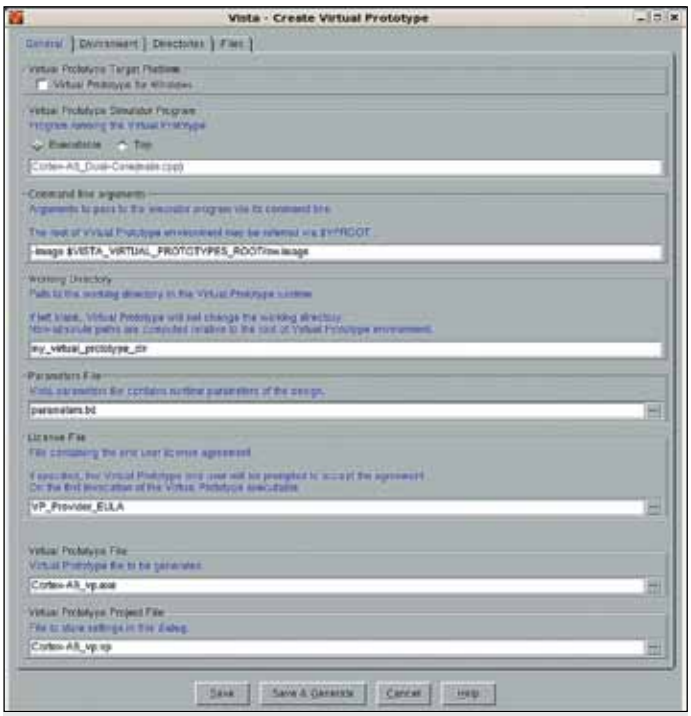

Рис. 3. Создание виртуального прототипа с помощью интерфейса Vista GUI

зователями, добавив к виртуальному прототипу дополнительное условие в лицензии.

Результатом работы является прототип, который состоит из собственно исполняемого файла симулятора и всех тех библиотек, какие необходимы для его запуска. Таким образом, он может распространяться и использоваться другими командами разработчиков, у которых нет необходимости в создании с нуля или изменении модели и топологии платформы TLM.

На следующих этапах конечные пользователи прототипов, а именно программисты и инженеры, разрабатывающие встроенное программное обеспечение, используют Vista Virtual Prototyping для интеграции и оптимизации своих программ, отрабатывая их на виртуальном прототипе.

Далее мы перечислим и рассмотрим подробнее основные возможности использования прототипов пользователями.

Использование виртуальных прототипов

После завершения этапа создания платформы TLM появляется соответствующий виртуальный прототип. Его могут использовать программисты и инженеры, разрабатывающие встроенное программное обеспечение. Они задействуют прототип для решения всех задач разработки и верификации, среди которых разработка программного обеспечения, взаимосвязанная симуляция аппаратного и программного обеспечения, их отладка и анализ.

С помощью Vista Virtual Prototyping полностью выполняются следующие задачи:

- управление симуляцией;
- разработка программного обеспечения;
- синхронизация аппаратного обеспечения в различных режимах;
- визуальное представление аппаратного обеспечения;
- взаимосвязанная отладка программ и аппаратуры;
- управление симуляцией аппаратуры.

Управление симуляцией

Запустить виртуальный прототип Vista можно или из командной строки, или же из интегрированной среды разработки Sourcery CodeBench IDE.

В режиме командной строки, в качестве аргументов строки есть возможность передачи переменных среды запуска прототипа.

При работе в интегрированной среде разработки Sourcery CodeBench IDE можно получать визуальное представление аппаратуры, осуществлять взаимосвязанную отладку программной и аппаратной частей, выполнять взаимодействие с файловой системой. Эти функциональные возможности доступны при подключении к среде Sourcery CodeBench, которая взаимодействует с API виртуального прототипа.

Разработка программного обеспечения

Пакет Vista Virtual Prototyping дает возможность разработки программного обеспечения, интеграции и проверки функционирования на целевой аппаратуре, поскольку виртуальный прототип обладает такими же функциональными возможностями, какими должны обладать целевые прототипы и пакеты поддержки плат (BSPs).

Необходимое отлаживаемое программное обеспечение запускается на виртуальном прототипе точно так же, как оно позже будет запущено на вновь разработанной аппаратуре. Есть поддержка пользовательского интерфейса, стеков приложений, промежуточной прошивки и драйверов, запускаемых поверх таких операционных систем, как Linux, Android и Nucleus, а также в режиме без операционной системы — на «голом железе» (Bare-Metal).

Система предоставляет все средства, необходимые для построения kernel-ядер Linux, а также для быстрой загрузки операционной системы.

Виртуальные прототипы Vista можно подключать к физическим устройствам, таким как терминалы и дисплеи рабочих хост-станций, что дает программистам возможность управления прототипами. С помощью соединений, устанавливаемых через Ethernet и USB, виртуальный прототип можно запустить на рабочих хост-станциях, используя их в качестве реальной среды функционирования. В режиме семихостинга пользователь может выводить на экран сообщения о прохождении операций, отслеживать текущее время симуляции и устанавливать детализацию сообщений об ошибках.

Синхронизация аппаратуры в различных режимах

Vista Virtual Prototyping может запускать симуляцию аппаратуры в двух режимах: в функциональном и в режиме представления.

В функциональном режиме осуществляется интеграция, верификация и отладка программного обеспечения. Режим представления позволяет проводить анализ и оптимизацию программного обеспечения для достижения лучшей производительности и снижения энергопотребления. Для работы в этих режимах модель виртуального прототипа строится на двух уровнях детализации синхронизации: синхронизация, не привязанная ко времени исполнения (Loosely Timed, LT), и синхронизация, учитывающая время исполнения (Approximately Timed, AT).

При работе в функциональном режиме виртуальные прототипы задействуют уровень синхронизации, не привязанной ко времени исполнения. Здесь каждая транзакция представляет собой завершенный перенос данных по аппаратной шине, вне зависимости от того, как именно возник перенос данных, и от того, сколько времени это заняло. В этом режиме скорость симуляции измеряется в сотнях MIPS, что равняется или очень близко к скорости обработки в реальном времени.

При работе в режиме представления виртуальные прототипы задействуют уровень синхронизации, учитывающий время исполнения транзакций. Здесь каждая транзакция представляет фазу переноса данных по конкретному протоколу шины, применяемой в модели (например, фазы адресации и данных для записи или чтения по шине AHB). Однако нужно учесть, что такая повышенная точность в режиме представления приведет к тому, что симуляция виртуального прототипа будет проводиться значительно медленнее, чем в функциональном режиме.

Vista Virtual Prototyping дает пользователю возможность выбора и переключения между двумя доступными режимами синхронизации непосредственно во время работы прототипа.

Функциональный режим является вполне достаточным для верификации и отладки большей части программного обеспечения верхнего уровня, такого как операционная система и уровни приложений.

А вот для анализа мощности и производительности аппаратного обеспечения под программным управлением и для низкоуровневого программного кода (такого как различные драйверы и программы реального времени) требуется работа виртуального прототипа в режиме представления. Отметим, что под программами реального времени понимаются те, для которых время исполнения программного кода зависит от задержек аппаратного обеспечения или от того, как данные разбиты по фазам синхронизации.

Визуальное представление аппаратуры

Интегрированная среда разработки Sourcery CodeBench IDE также обеспечивает непосредственное визуальное представление и управление аппаратными объектами разрабатываемой платформы. Эти объекты включают все периферийные регистры и локальные переменные, которые декларированы при создании компонента модели TLM.

Для распознавания объектов используются их иерархические пути, а связанные с ними величины представлены в отдельных ветках дерева, доступного в окне просмотра регистров Sourcery CodeBench Register. Все значения объектов можно редактировать, также в процессе отладки можно добавлять новые значения.

Взаимосвязанная отладка программ и аппаратуры

Пользователи могут проводить взаимосвязанную отладку аппаратной и программной частей, устанавливая точки прерывания в аппаратном обеспечении, что остановит его симуляцию при условии достижения точки останова. Позже симуляция может быть возобновлена, поскольку программный отладчик останавливается после завершения обработки инструкции, которая и привела к срабатыванию останова.

Это позволяет пользователю проверить состояние программы в данной точке и затем перейти к следующим программным инструкциям, в то же время просматривая состояние аппаратных объектов, которые меняются в зависимости от выполнения той или иной программной инструкции. Интерфейс отладчика приведен на рис. 4.

Управление симуляцией аппаратуры

Для управления симуляцией аппаратуры имеется набор команд, доступных из консоли Sourcery CodeBench. С помощью этих команд пользователь может:

- Выполнить перезапуск ядра CPU, обрабатывающего программный код, подключенный к отладчику, или осуществить перезапуск всех узлов платформы, включая ядра.

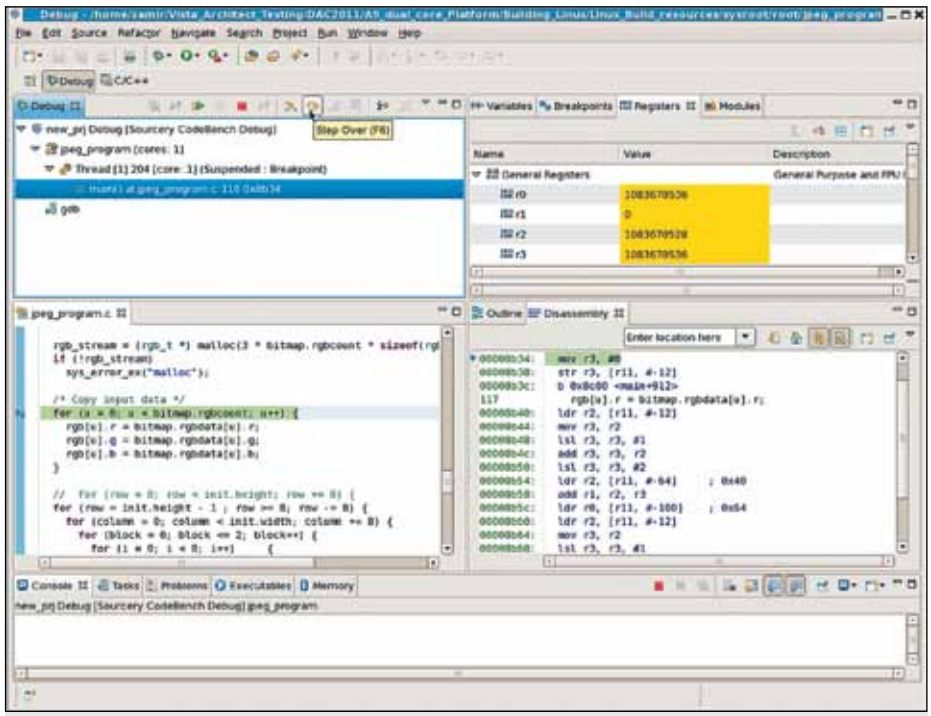

Рис. 4. Интерфейс пользователя Sourcery CodeBench для отладки аппаратного и программного обеспечения

- Установить точку прерывания, которая остановит симуляцию по завершении обработки инструкции, получившей доступ к аппаратному объекту (регистру или переменной). Точка прерывания подобна триггеру, срабатывающему на чтение, запись или на оба типа доступа.
- Запросить место нахождения точки прерывания и получить иерархический путь к нему.
- Удалить точку прерывания.
- Загрузить готовый файл в формате ELF в память текущего ядра CPU.
- Управлять режимом запуска платформы (функциональный или режим представления).
- Отображать режим функционирования, который установлен для платформы в данный момент.
- Отключить обработку DMI (Direct Memory Interface), что в функциональном режиме позволит быстрее отслеживать транзакции через шинную архитектуру платформы и опционально отображать их на встроенном осциллографе Vista Waveform.
- Отображать, установлена ли платформа в режим DMI.
- Отображать текущее время симуляции кода SystemC.
- Отображать информацию о процессе симуляции.
- Прерывать процесс симуляции с опционально заданным кодом выхода.

Дополнительно система Vista Virtual Prototyping позволяет проводить управление трассировкой ядер CPU с помощью предустановленных вызовов (callbacks) по значащим событиям. (Например, когда ядро перешло в режим сна.) Это можно использовать для дополнительной отладки и анализа, задаваемого пользователем. Также для управления симуляцией обеспечивается доступ по API с поддержкой семихостинга и других полезных задач. (Например, выдача сообщений каждый раз при выполнении инструкции.)

В системе Vista Virtual Prototyping также можно манипулировать файлами посредством запросов встроенной операционной системы, которая загружена на виртуальный прототип.

Это позволяет пользователю разрабатывать, выполнять сборку и отлаживать программные пакеты на хост-машине, затем получать к ним доступ с консоли целевой ОС без необходимости повторной симуляции и перезапуска ОС на виртуальном прототипе. Например, файлы могут копироваться с файловой системы хост-машины в целевую локальную директорию на прототипе (или наоборот) с использованием Linux-команды cp, вызываемой из встроенной операционной системы Linux.

Просмотр и управление аналитическими данными симуляции

Программный пакет Vista Virtual Prototyping предоставляет богатый набор функциональных возможностей для анализа встроенного программного обеспечения и аппаратуры с программным управлением.

Результаты анализа могут быть представлены для просмотра в виде графиков и таблиц, а также в формате сводных отчетов.

Анализ производительности и мощности

Во время функционирования виртуального прототипа пользователь может отслеживать или же сам создавать данные для анализа. Так можно увидеть, каким образом аппаратные атрибуты платформы и программ, запущенных на этой платформе, повлияли на изменение производительности и мощности в виртуальном прототипе.

Анализируя данные о пиках нагрузки, средних задержках, пропускной способности и загруженности тех или иных портов, шин или подсистем, пользователь может оптимизировать программное обеспечение для достижения необходимой мощности и производительности оборудования.

Виртуальный прототип не только в состоянии отследить и сгенерировать итоговые аналитические данные, но также может запускать средство просмотра таких результатов анализа. Анализатор дает пользователю возможность просмотреть распределение транзакций и переменных во времени, а также их статистическое распределение.

Данные могут быть представлены с различной степенью детализации, например, по выбранным транзакциям или для данного экземпляра модели. В соответствии с настройками пользователя результаты анализа могут отображаться в виде графиков, отчетов, а также в формате сводных таблиц данных.

Пользователь может просмотреть дерево полного проекта, выбрать любой объект и указать, какие измерения отображать. Форма просмотра аналитических данных представлена на рис. 5.

Отображаемые объекты выделяются цветом. Соответствующие связанные данные для каждого объекта прорисовываются в таблице под графиком. В дополнение к средним значениям вычисляются максимум и минимум. Для удобства просмотра они отображаются разным шрифтом, но одним цветом. Максимальные значения вычисляются для большинства типов аналитических данных, представляемых Vista (мощность, пропускная способность, задержка, время арбитража, конкуренция запросов). Минимальные значения вычисляются для любых типов данных, определенных пользователем.

Пользователь может задавать следующие типы аналитических данных:

- Пропускная способность. Уровень активности выбранных объектов, измеряемый как число транзакций или число переданных байтов за указанный период времени.
- Задержка. Среднее время задержки за указанный период времени, необходимое для выполнения транзакции указанного типа (например, чтение или прерывание). Время задержки отобразится для каждой транзакции выбранного типа.
- Мощность. Динамическая мощность, статическая (утечка) и мощность, расходуемая для дерева синхронизации. Отобразится для полной схемы, а также для выбранных ее элементов.
- Распределение мощности. Вклад каждого элемента в суммарную мощность энергопотребления.
- Атрибуты (аналитические данные, настраиваемые пользователем). Значения атрибутов с изменением времени. В качестве таких атрибутов могут быть определяемые пользователем переменные, заданные в модели TLM.
- Пропускная способность шины/полоса пропускания. Для каждой шины, добавленной на разрабатываемую схему, можно открыть окно аналитических данных о пропускной способности шины (Bus Throughput analysis view). Оно показано на рис. 6.

Здесь пользователь может увидеть пропускную способность выбранного ведущего/ведомого сокета (на верхнем графике) и составляющие каждого из соответствующих ему ведомых/ведущих сокетов (на нижнем графике).

Для анализа уровня конкуренции запросов реализована универсальная модель шины Vista Generic Bus model. Данные вычисляются автоматически при трассировке сокетов. Предоставляется средневзвешенное по времени число запросов, ожидающих доступа к компоненту и проходящих через этот сокет. Поддерживается два типа анализа: конкуренция запросов фазы адресации (address phase contention) и конкуренция запросов фазы данных (data phase contention).

Конкуренция запросов фазы адресации — это атрибут ведущего сокета шины. Предоставляется средневзвешенное по времени число запросов, ожидающих доступа к шине, которая пытается отправлять транзакции через ведущий сокет шины к подключенным к нему периферийным устройствам.

Конкуренция запросов фазы данных — это атрибут, который предоставляет средневзвешенное по времени число ведущих или ведомых компонентов, ожидающих отправки данных. Он подразделяется на конкуренцию запросов фазы передачи данных от ведущих компонентов (forward), где сообщается о среднем числе ведущих компонентов, ожидающих прямой передачи данных к периферийным ведомым компонентам, и конкуренцию запросов фазы передачи данных от ведомых компонентов (backward), где сообщается о среднем числе периферийных ведомых компонентов, ожидающих отправки данных обратно к ведущему компоненту.

Универсальная модель шины Vista Generic Bus model также обеспечивает анализ уровня времени арбитража. Данные вычисляются автоматически при трассировке сокетов. Предоставляется среднее время, за которое транзакции получают доступ к шине с конкретного сокета шины.

Поддерживается два типа анализа времени арбитража: время арбитража адресации (address arbitration time) и время арбитража данных (data arbitration time).

Время арбитража адресации — это атрибут ведомого сокета шины. Предоставляется среднее время, проходящее с момента отправки сигнала Begin_Request от ведомого сокета к шине и до момента достижения этим сигналом ведущего сокета шины. Это то время, которое необходимо транзакции для получения фактического доступа к шине.

Время арбитража данных — это атрибут ведущего сокета шины. Предоставляется среднее время, проходящее с момента отправки сигнала Begin_Response от ведущего сокета к шине и до момента достижения этим сигналом ведомого сокета шины. Это время, которое занимает фаза отклика, с момента его начала и до достижения откликом ведомого сокета шины на противоположном конце шины.

Анализатор Vista Virtual Prototyping в состоянии автоматически вычислять отношение числа успешных и неуспешных обращений к кэш-памяти в рамках заданного интервала времени для тех ядер процессора, которые включают компоненты Icache и Dcache. Эти коэффициенты дополнительно могут быть разделены на значения, связанные с чтением кэш-памяти и записью в нее. Отметим тот факт, что эти аналитиче-

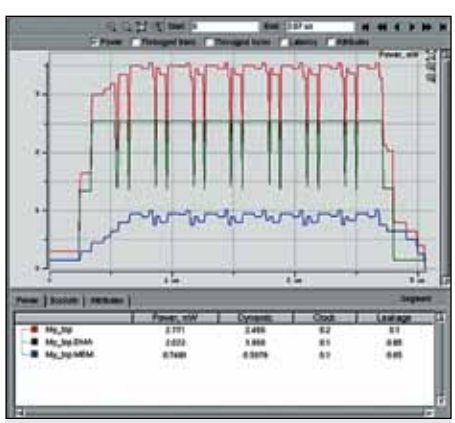

Рис. 5. Просмотр аналитических данных о мощности, включая мощность полного проекта, DMA и MEM

Рис. 6. Окно аналитических данных о пропускной способности шины

Рис. 7. Окно подробных аналитических данных о шине

ские данные могут быть как получены в режиме представления, так и использованы в функциональном режиме.

Форма просмотра этих аналитических данных приведена на рис. 7. На форме представлены следующие данные:

- Коэффициент успешных и неуспешных обращений к кэш-памяти для двухъядерного процессора Cortex-A9MP.
- Арбитраж передачи данных по шине от периферии к ведущему компоненту.
- Конкуренция запросов фазы адресации.

Анализ выполнен для оценки пропускной способности активных сокетов шины по отношению к памяти ведомого устройства.

Анализатор Vista Virtual Prototyping предоставляет возможность просмотра аналитических данных по распределению мощности как для всего дизайна, так и для отдельной выбранной ветви дерева. Окно просмотра распределения мощности показано на рис. 8.

Vista Virtual Prototyping имеет значительную гибкость в плане выбора отображаемых данных. Для графиков может варьироваться степень детализации, а также установка начальных и конечных значений.

При каждом новом просмотре все данные обновляются. И при движении мышью по графику можно увидеть значение данных в каждый момент времени. Пользователь может замерить изменения значений между двумя точками на графике как по вертикали,

Рис. 8. Просмотр распределения мощности

Рис. 9. Сравнение распределения мощности для двух архитектур

так и по горизонтали. При этом можно изменять масштаб графиков и по горизонтали, и по вертикали. Это позволяет задать требуемую точность для измерений.

Анализатор Vista Virtual Prototyping позволяет выполнять анализ и сравнение результатов нескольких сеансов симуляции, чтобы определить эффективность изменения системных конфигураций, выбора протокола, программных изменений. Пример результатов сравнения приведен на рис. 9.

Анализ программного обеспечения

Анализ программного обеспечения в Vista Virtual Prototyping осуществляется с помощью программы Mentor Embedded Sourcery Analyzer. Она работает с виртуальными прототипами Vista, интегрирует данные от одноядерных и многоядерных CPU и поддерживает операционные системы типа Linux и RTOS, а также работает в режиме без операционной системы — на «голом железе» (Bare-Metal).

Sourcery Analyzer включает библиотеку популярных и интуитивно понятных инструментов системного анализа и визуализации. Эти инструменты содержат большинство возможностей, широко востребованных программистами для оценки влияния функционирования CPU/ядра и программ на работоспособность, производительность и мощность конечного продукта.

Они позволяют пользователю отслеживать статистику и состояние CPU, активность файловой системы во времени, функциональные вызовы и статистику по ним, запасы задержки, таймеры ожидания блокировки и удержания, а также состояния потоков и процессов.

С помощью программы Sourcery Analyzer пользователь дополнительно может создать свои инструменты для воздействия на производительность, а также в отладочных целях. Такие инструменты, написанные на языке Java, помогают обеспечить анализ и визуализацию, характерную для конкретных приложений, а также достичь требуемой производительности всего проекта и оптимизировать ее.

С помощью программы Sourcery Analyzer можно получить доступ ко всему тому API, который использует все встроенные аналитические программы данного инструмента. Он также снабжен визардом для более легкого и удобного создания новых аналитических программ.

Наиболее важные аналитические характеристики программы Sourcery Analyzer:

- Состояние CPU (рис. 10). Показывает состояние CPU/ядра: работа, бездействие, прерывание процесса.
- Статистика по CPU. Какое количество времени каждое из ядер находилось в том или ином состоянии.

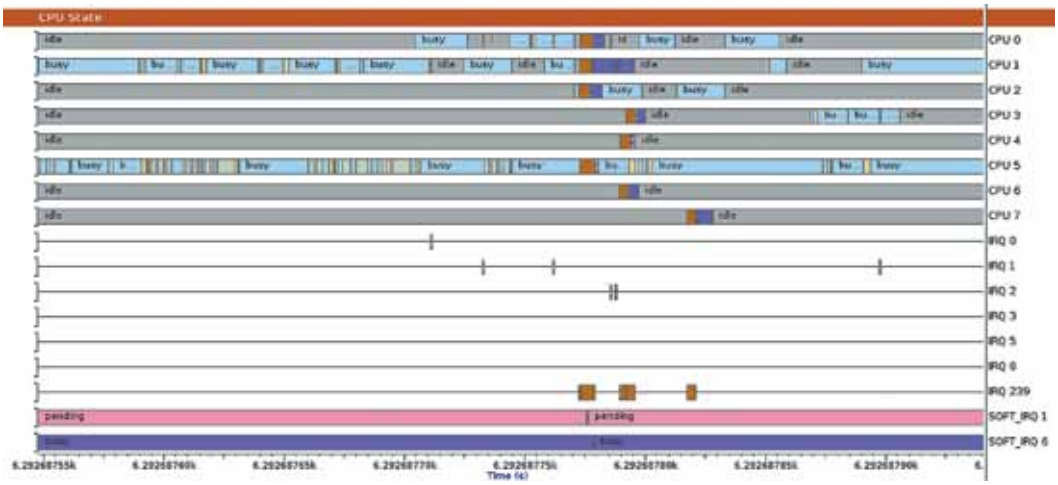

Рис. 10. Просмотр состояния CPU в окне Sourcery Analyzer

- Активность файловой системы. Активность файловой системы во времени.
- Функциональные вызовы. Когда и какие функции были запущены.
- Статистика по функциональным вызовам. Какие функциональные вызовы, кем и как часто осуществлялись.
- Запас задержки. На каких элементах израсходован запас задержки.
- Таймеры ожидания, блокировки и удержания. Сколько времени занимает выполнение операции блокировки и как долго сохранялось удержание.
- Состояние потоков и процессов (рис. 11). Когда программные потоки были запущены, когда они были в режиме сна и когда осуществляли системные вызовы.
- Размещение памяти приложения. Использование «кучи».
- Уровень сбоев страницы памяти. Частота сбоев страницы памяти.
- Мощность виртуального прототипа. Мощность, потребляемая виртуальным прототипом (общая и по элементам).
- Коэффициент успешных/неуспешных обращений к кэшу виртуального прототипа. Отношение числа успешных и неуспешных обращений к кэш-памяти процессора виртуального прототипа (также раздельно для режимов READ и WRITE).

Эти функциональные возможности в сочетании с атрибутами аппаратного обеспечения, такими как мощность и коэффициент успешных и неуспешных обращений к кэш-памяти, позволяют пользователю проводить анализ влияния работы его программного обеспечения на функциональность, производительность и потребление мощности у разрабатываемого изделия.

Рис. 11. Просмотр состояния потоков и процессов в окне Sourcery Analyzer

Сводные аналитические отчеты

Сводные отчеты системы Vista Virtual Prototyping предоставляют данные по важнейшим параметрам разрабатываемого проекта. Эти данные собираются во время симуляции. Внешний вид такого сводного отчета приведен на рис. 12.

Эти параметры либо связаны со значениями мощности (вычисляемыми как собственно параметры проекта), либо являются пользовательскими (то есть переменными, задекларированными пользователем в модели TLM). Они включают:

- Профили распределения мощности, а конкретно полную и динамическую мощность, мощность, потребляемую деревом синхронизации сигналов, а также утечку на элементах схемы.
- Профили синхронизации транзакций, а именно пропускную способность транзакций и потока байтов, а также задержки для каждого из сокетов проекта, включая разделение между режимами READ и WRITE. Транзакции к/от модели шины могут также включать время арбитража и уровень конкуренции запросов.
- Отчеты по атрибутам, таким как коэффициенты успешных и неуспешных обращений к кэшу инструкций и данных, плюс другие атрибуты, необходимые пользователю для анализа проекта.

Анализатор Vista Virtual Prototyping также может генерировать отчеты в формате ASCII.

- Файл *Transaction Report* (отчет о транзакциях) в формате ASCII содержит листинг всех транзакций, обращенных к базе данных модели TLM. Есть возможность указать конкретные элементы модели, в таком случае отчет будет составлен только для них.

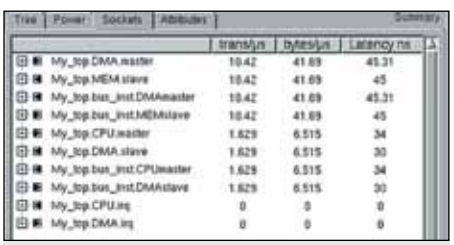

Рис. 12. Сводный отчет анализатора Vista Virtual Prototyping

- Файл *Summary Report* (сводный отчет) содержит полный набор данных, собранных за все время работы симулятора. Содержит информацию о полной мощности и типах мощности, израсходованных за время симуляции, а также о пропускной способности транзакций, данных транзакций и задержке для сокетов модели TLM.

Интегрирование виртуального прототипа в процесс разработки встроенных систем

Vista Virtual Prototyping интегрируется в поток разработки встроенного программного обеспечения, который совмещает этап, предшествующий работе с реальным оборудованием (совмещает переход от проверки и оптимизации программного обеспечения на абстрактной модели симулятора к эмулятору и далее к прототипу FPGA), и этап работы с реальным физическим оборудованием (то есть финальный продукт).

Так, с помощью инструментов, предоставляемых продуктом Mentor Embedded Sourcery CodeBench, одни инженеры разрабатывают виртуальные прототипы, другие используют готовые их экземпляры на виртуальном оборудовании, а третьи — уже на физическом. А это значит, что пользователи могут легко производить смену модели аппаратных средств с виртуальных прототипов Vista на физические аппаратные прототипы на платах и при этом по-прежнему использовать ту же самую интегрированную среду разработки Sourcery IDE. Наглядно это представлено на рис. 13.

Рис. 13. Унифицированная среда разработки ПО при переходе от виртуального к физическому оборудованию

Заключение

Итак, в статье показано, что симуляторы аппаратуры, с одной стороны, являются необходимыми инструментами для создания надежных встраиваемых систем. Если симуляторы внедрять на самых ранних этапах проектирования и разработки программ и аппаратуры, то они позволяют значительно сократить сроки работы над проектом и в то же время повысить качество производимого продукта.

С другой стороны, симуляторы весьма удобны в работе, поскольку не требуют привязки к конкретному оборудованию и позволяют программистам создавать свой код еще до того, как реальное оборудование будет выпущено с конвейера.

Мы подробно рассмотрели основные функциональные возможности программного пакета Vista Virtual Prototyping. С его помощью успешно и в полной мере решаются следующие задачи.

Во-первых, системные архитекторы и разработчики SoC или проектировщики физического оборудования создают платформы для моделирования на уровне транзакций. Результатом их работы являются готовые виртуальные прототипы, поставляемые конечным пользователям.

Во-вторых, конечные пользователи, а именно программисты и разработчики встроенных приложений, выполняют отладку своего кода на готовых виртуальных прототипах. Отметим важный момент, что выполняется также взаимосвязанная отладка программ и аппаратуры, анализируется их совместное влияние друг на друга. Таким образом удается достичь требуемого уровня оптимизации готового продукта.

Программный пакет Vista Virtual Prototyping удобен и тем, что инструменты, входящие в его состав, можно использовать как на этапах раннего проектирования, так и при работе с реальным оборудованием, то есть финальным продуктом.

Литература

1. Сергеева А. Тестирование работоспособности промышленного компьютера // Компоненты и технологии. 2014. № 2.
2. http://www.mentor.com/esl/
3. http://www.mentor.com/embedded-software/