Отладка многоядерной потоковой рекуррентной вычислительной системы

А.А. Прокофьев, ИПИ РАН

Аннотация: В статье приведено описание основных подходов к отладке аппаратуры. Перечислены требования, которые предъявляются к средствам отладки аппаратуры: обеспечение наблюдаемости, управляемости и контроля выполнения проекта; описаны стандартные методологии отладки. Описаны особенности отладки МП РВС.

Введение

Так как аппаратура лежит в основе вычислительных систем (ВС) к ней предъявляются высокие требования. Поэтому неотъемлемой частью процесса разработки ВС является проведение тестирования и отладки, целью которых является проверка соответствия поведения разрабатываемой системы спецификациям. Основным методом, используемым для проверки реализации ЭС является моделирование. При разработке под ПЛИС моделирование шляется незаменимым инструментом на первых этапах разработки. С его помощью возможна отладка как отдельных блоков ВС, так и всего устройства в целом. Однако для воспроизведения всех возможных ситуаций объем моделирования оказывается столь велик, что реализация даже малой части этого объема физически неосуществима. К тому же, нередко ошибка связана с такой комбинацией или временной последовательностью сигналов на входах ПЛИС, которая, по мысли разработчика, не возникает при работе и поэтому не проверяется и при моделировании.

1. Требования к отладке

Ни одна система не может быть реализована без дефектов - ошибки возникают на различных этапах проектирования: начиная с разработки спецификаций и заканчивая процессом тестирования (рис. 1) [1, 2].

Процесс отладки можно разделить на три этапа: выявление ошибки, диагностика (локализация) ошибки и исправление ошибки. Первый этап покрывается процессом верификации, чья цель определить действительно ли система работает в соответствии со спецификацией. Второй этап включает в себя выявление причин и места возникновения ошибки. Локализация часто включает процесс определения когда и при каких условиях появляется ошибка. Последний шаг - изменение неисправной части системы с целью устранить ошибку в проекте.

Чтобы стало возможным локализовать ошибку нужно, как минимум, организовать достаточную *наблюдаемость* внутреннего состояния системы - т.е. доступ к фиксации и отображения состояния схемы (значения внутренних

регистров и памяти). Увеличить эффективность, упросить и ускорить процесс отладки поможет обеспечение *управляемости* и *контроля выполнения.* К управляемости можно отнести возможность изменения внутреннего состояния системы, а к контролю - выполнение по шагам и останов в контрольных точках.

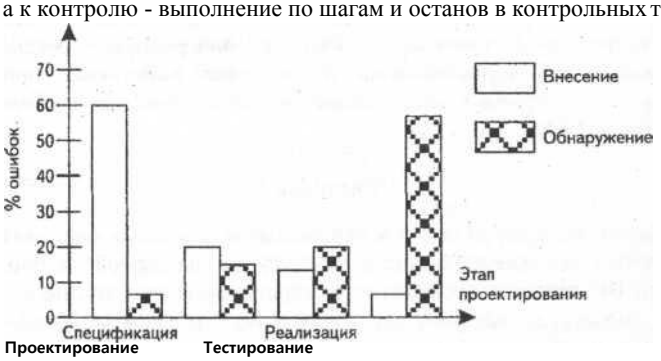

Рисунок 1. Ошибки проектирования и их устранение

2. Методологии отладки

Существуют следующие методологии отладки:

- *Выполнение проекта в аппаратуре для функциональной и временной верификации:* Так как реализация в ПЛИС является мало затратной (главным образом по времени), проект может быть реализован и выполнен в аппаратуре для функциональной временной верификации. Моделирование по-прежнему остается необходимым, так как дает возможность избавить систему от очевидных недостатков, но запуск в реальных временных условиях может увеличить производительность, сократив время выполнения до секунд или минут, вместо часов или дней в среде моделирования.
- *Простые изменения проекта для обеспечения видимости внутренних сигналов:* Опять же в связи с тем, что ПЛИС может быть перепрограммирована множество раз, для отладки оборудования в него могут быть временно включены конструкции обеспечивающие дополнительную наблюдаемость, управляемость и контроль исполнения для проверки проектов. Примеры схем которые могут быть добавлены: оцепи сканирования (для наблюдаемости и управляемости) о механизм аппаратных точек останова о встроенные логические анализаторы о аппаратура сбора статистики о аппаратура самотестирования

Естественно, что внесение этих изменений имеет свои недостатки с точки зрения верификации, главным образом в том, что изменяются временные характеристики схемы и увеличивается занимаемая на кристалле площадь.

3. Стандартные инструменты отладки

Одни методологии аппаратной отладки включают оснащение

инструментальными средствами всего проекта *{глобальные подходы),* другие определенной локальной части *{локальные подходы).* Примерами глобальных подходов являются цепи сканирования и обратное считывание конфигурации (readback), где весь проект ПЛИС модифицируется с целью улучшения наблюдаемости схемы. Локальные подходы включают как отладочные порты, так и встроенные логические анализаторы (embedded logic analyzer, ELA). В этом случае изменения в проекте производятся с целью контроля определенных сигналов. Локальные подходы требуют меньшее количество аппаратных ресурсов, поэтому меньше влияют на скорость работы устройства, но обеспечивают меньшую наблюдаемость и управляемость.

Встроенный логический анализатор (embedded logic analyzer, ELA) использует незадействованные ресурсы ПЛИС и имеет доступ ко всем логическим сигналам. Он позволяет запоминать последовательность сигналов или их комбинаций в сдвиговом регистре, организованном в ПЛИС на одном или нескольких блоках памяти. Причем запись может вестись как в каждом такте, так и по условию, выработанному на основе логических сигналов ПЛИС. Запись останавливается (по достижении ошибки) либо по условию, выработанному внутри ПЛИС, либо по внешнему сигналу, поступающему, например, от тестового оборудования. После этого сдвиговый регистр хранит информацию и о сигналах на момент проявления ошибки, и о сигналах за предыдущее время (предыстория). Считав эти данные, можно получить информацию о непосредственном проявлении ошибки и о том, что ей предшествовало [3].

Использование внешнего контрольно-измерительного оборудования связано с обеспечением доступа к внутренним регистрам посредствам подключения их к неиспользуемым внешним выводам ПЛИС. Такой подход позволяет использовать для регистрации сигналов большой объем памяти внешнего оборудования, что очень полезно при отладке в случаях, когда сбой и вызвавшая его причина сильно разнесены во времени.

Данные подходы обеспечивают достаточно хорошую наблюдаемость проекта, но совершенно отсутствует возможность какого-либо контроля над работой ВС.

Интерфейс JTAG дает наиболее гибкие и широкие возможности. С его помощью в схему можно внедрить функциональность, позволяющую обеспечить достаточно высокий уровень как наблюдаемости, так и управляемости проекта.

4. Особенности реализации МП РВС

Архитектура МП РВС является двухуровневой, состоящей из управляющего и операционного уровней. На управляющий уровень возложены минимально необходимые функции управления операционным уровнем и связь с внешним окружением, и он реализован на традиционных ФН-принципах. Многоядерный операционный уровень реализован в виде рекуррентного обработчика сигналов (РОС), предназначенного для эффективного исполнения параллельных алгоритмов.

В РОС используется капсульный стиль программирования, основные понятия которого соответствуют принятым в телекоммуникационных приложениях: инкапсуляция, собственно капсула и декапсуляция. Суть капсульного программирования заключается в процессе формирования капсулы-шаблона (инкапсуляции), которая заносится в буферную память. По мере прихода в ВС обрабатываемых данных, управляющий уровень производит процесс заполнения шаблона, после завершения которого капсула инициируется и поступает на обработку. Процессу обработки капсулы на операционном уровне предшествует процедура декапсуляции [4].

Синтезируемая модель РОС разрабатывается на языке VHDL, и используется системой капсульного программирования и отладки (СКАТ) в качестве виртуального процессора для отладки капсул [5]. СКАТ позволяет выполнять и отлаживать капсулы в пошаговом режиме, при этом инженер имеет возможность не только просматривать, но и изменять значения внутренних регистров и памяти. В связи с этим в модель РОС должны быть включены средства обеспечивающие управляемость проекта в достаточно высокой степени. А так как модель является еще и синтезируемой, то логично реализовать отладочную обвязку таким образом, чтобы ее можно было использовать и в процессе отладки в ПЛИС. Поэтому в МП РВС реализован отладочный модуль, который способен работать как в режиме моделирования, так и в реальной аппаратуре.

Также при реализации отладочной обвязки, с целью обеспечения высокого уровня контроля выполнения, нужно учитывать, что РОС состоит из четырех параллельно работающих процессорных ядер, в которых реализована конвейерная обработка данных [4].

5. Модуль отладки РОС

Базируясь на особенностях РОС было принято решение реализовывать модуль отладки в виде блока со стандартным интерфейсом памяти:

- 32-раз рядная шина адреса
- 32-разрядная шина входных данных
- 32-разрядная шина выходных данных
- 1-разрядный сигнал разрешения записи

Такая реализация позволяет взаимодействовать с модулем как управляющему процессору (аппаратный запуск МП РВС), так и СКАТу (режим работы РОС в качестве виртуального процессора). Кроме того к модулю отладки

организован доступ через интерфейс JTAG, что позволяет проводить отладку с host-машины [6]

Принцип работы отладочного модуля следующий:

1) После аппаратного сброса РОС работает в нормальном режиме;
2) Инициализация режима отладки осуществляется путем подачи на шину адреса значения 00000008 и сигнала разрешения записи;
3) В режиме отладки работа системы приостанавливается, и открывается доступ чтения/записи к внутренним регистрам:
    a. каждый регистр имеет свой адрес;
    b. в случае если длина регистра превышает 32 разряда он считывается/устанавливается по частям;
4) Выход из режима отладки происходит подачей на шину адреса значения 00000000 и сигнала разрешения записи.

Литература

1) *Грушвицкий Р., Михайлов М.* Проектирование в условиях временных ограничений: отладка проектов. - Компоненты и технологии, №6 2007 - с. 131-136
2) *Грушвицкий Р., Михайлов М.* Проектирование в условиях временных ограничений: отладка проектов. - Компоненты и технологии, №8 2007 - с. 88-94
3) *Погорилый А., Соколов А*. Отладка устройств, реализованных на ПЛИС. Логический анализатор размещенный внутри ПЛИС. - Электроника: Наука, Технология, Бизнес, 2004 - с.54-55
4) *Ю.А. Степченков, В.Н. Волчек, В.С. Петрухин, А.А. Прокофьев, Р.А. Зеленов.* Цифровой сигнальный процессор с нетрадиционной рекуррентной потоковой архитектурой // Всероссийская научно- техническая конференция "проблемы разработки перспективных микро- и наноэлектронных систем (мэс)\ Сборник трудов. - М.: Институт проблем проектирования в микроэлектронике РАН, 2010. - С. 412-417
5) *Зеленое Р.А., Степченков Ю.А., Волчек В.Н., Хилько Д.В., Шнейдер А.Ю., Прокофьев А.А.,* Система капсульного программирования и отладки // Системы и средства информатики. - М.: ИПИ РАН, 2010. - С. 24-30.
6) Altera. Virtual JTAG (sld_virtualjtag). Megafunction User Guide. URL: http://www.altera.com/literature/ug/ug_virtualitag.pdf (дата обращения: 14.10.2011).

Сведения об авторе:

Прокофьев Александр Александрович, год рождения: 1986. Место обучения: аспирантура ИПИ РАН. Место работы: ИПИ РАН, 22 отд., инженер- исследователь. Направления научных интересов: нетрадиционные архитектуры микропроцессоров; верификация и отладка аппаратно-программных средств.