КОЛЛЕКТИВНОЕ ПРОЕКТИРОВАНИЕ FPGA

ДЖЕФФ ГАРРИСОН (JEFF GARRISON), директор отдела маркетинга, Synopsys

Возможности FPGA невероятно выросли. 28-нм FPGA является устройством, эквивалентным ASIC с 20—30 млн вентилей. При таких размерах FPGA ее средства проектирования, которые традиционно применялись одним-двумя инженерами проекта, перестают эффективно действовать. Теперь даже небольшая группа разработчиков не в состоянии выполнить проектирование и верификацию этих устройств за разумное время.

В проектах с использованием FPGA все чаще принимают участие достаточно большие коллективы инженеров, работающих в географически разных точках мира. Данная тенденция оказывает большое влияние на средства проектирования, верификации и управлении этими сложными электронными устройствами.

Рассмотрим по порядку три главных аспекта коллективной работы над проектом:

– распределенная и параллельная разработка;

– технологический процесс разработки;

– отслеживание состояния проекта и отчеты.

РАСПРЕДЕЛЕННОЕ И ПАРАЛЛЕЛЬНОЕ ПРОЕКТИРОВАНИЕ

Рабочая сила компаний распределяется по всему миру. Однако большинство требований к совместному пользованию ресурсами и сотрудничеству в рамках проекта, по сути, остается прежним. Части проекта, разработкой которых занимается тот или иной инженер или группа сотрудников, должны быть автономными и легко интегрироваться в проект высшего уровня.

Эти подпроекты часто находятся на разных этапах разработки, поэтому требуется система управления ими. Кроме того, повторное использование IP-ядер из предыдущего проекта или полученных от независимой фирмы является довольно-таки распространенной практикой проектирования с применением FPGA. В настоящее время нецелесообразно создавать новый код RTL для всей функциональности, реализуемой в современных больших FPGA. При распределенном и параллельном проектировании необходимо учитывать следующие возможности:

– управления и интеграции подпроектов в систему высшего уровня, в т.ч. управление версиями исходного кода;

– повторного использования проекта или IP-ядер;

– контекстного синтеза.

В большинстве случаев коллектив разработчиков состоит из нескольких групп, занимающихся субблоками, и интегратора проекта на высшем уровне. Возможность индивидуального проектирования каждого субблока и их регулярная (например, в ночные часы) автоматическая интеграция в структуру высшего уровня чрезвычайно полезна. Такие системы управления версиями как CVS и Perforce становятся привычными средствами управления большими проектами с FPGA, RTL-код которых постоянно меняется из-за множества источников (см. рис. 1).

Полезным побочным эффектом независимой разработки и управления субблоками является намного более простая возможность использовать их повторно в будущих проектах. Исходный RTL-код и проектные ограничения для FPGA можно заархивировать как единый верифицированный функциональный блок для быстрой интеграции в проекты следующего поколения.

Одним из главных вызовов для команды разработчиков является время, отведенное на создание системы, а также бюджет на ресурсы для каждого субблока. Руководителю группы требуются средства, позволяющие назначать каждому субблоку определенные ресурсы, чтобы не превысить возможности FPGA в отношении RAM, DSP и справочных таблиц (LUT). Эти средства позволяют избежать ситуации, в которой две независимые группы, работающие над своими частями проекта, не знают о ресурсах, которые отведены под использование другими субблоками.

В тех случаях, когда определенная информация о нескольких субблоках системы известна заранее, во время синтеза осуществляются дополнительные оптимизации, что позволяет в целом улучшить временные характеристики. Например, рассмотрим случай с константой, передаваемой по всем субблокам системы. В этой ситуации существует возможность оптимизировать схему, устранив ненужную логику. В отсутствие контекстной информации, полученной с помощью пакета синтеза схем, такая оптимизация невозможна.

Кроме того, эти средства должны обладать определенной гибкостью, чтобы обеспечить селективную граничную оптимизацию при появлении критического пути между субблоками одной схемы.

ТЕХНОЛОГИЧЕСКИЙ ПРОЦЕСС РАЗРАБОТКИ

Всегда имеются компромиссы при установлении приоритетных требований к схеме, например, по тактовой частоте, потребляемой мощности и

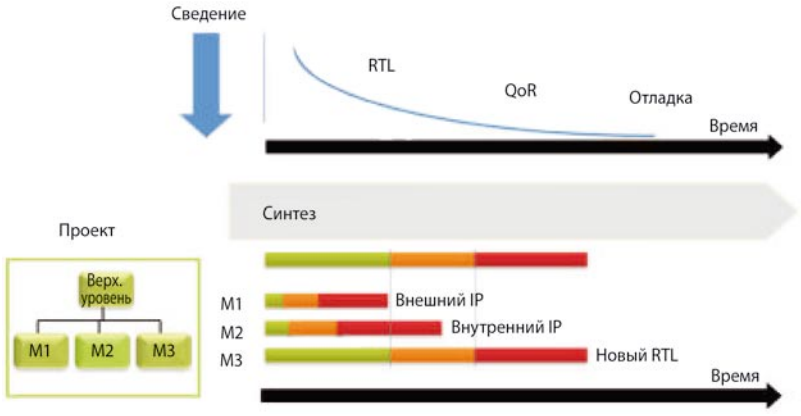

Рис. 1. **Коллективная разработка проекта независимо обеспечивает устойчивую работу каждого блока таким образом, чтобы их объединение на высшем уровне гарантированно оказалось работоспособным**

размеру. Синтез на высшем уровне, на котором осуществляется монтаж и трассировка, как правило, обеспечивает наилучшее качество результатов (QoR), но часто затрагивает те участки схемы, которые уже прошли проверку. В результате приходится затрачивать время на повторное исправление ошибок. Синтез снизу вверх позволяет исключить ненужные участки схемы при монтаже и трассировке, обеспечив предсказуемость и стабильность работы схемы, но могут ухудшить суммарный показатель QoR. Возможно, сочетание этих двух методов было бы наиболее целесообразным. В идеальном случае желательно достичь всех поставленных целей, не идя на большие компромиссы. Для этого разработчикам необходимо следующее:

– сочетание методологий «сверху вниз» и «снизу вверх»;

– быстрый цикл создания системы и максимальное качество QoR;

– интеграция с инструментами поставщиков FPGA.

Смешанная методология позволяет нескольким группам независимо работать над своей частью проекта, тогда как интегратор высшего уровня производит отладку всей схемы, обеспечивая наилучшее качество. Некоторые критичные к временным параметрам блоки, которые уже были разработаны и протестированы, могут быть исключены из оптимизации во избежание ненужных изменений, тогда как другие подвергнутся этому процессу. При желании осуществляется финальный синтез сверху вниз для достижения оптимального качества. Разработка общей топологии кристалла может быть затруднена и привести к непредсказуемым результатам, поэтому она не рассматривается как общее требование для всех проектов. В методологии коллективной разработки крайне необходимо иметь автономные функциональные блоки, включающие RTL и соответствующие схемные ограничения и поддерживающие повторное использование модулей или проекта.

Быстрый цикл создания устройства методом синтеза, а также монтаж и трассировка — самые главные задачи по мере перехода от 40-нм FPGA к 28-нм. Во многих случаях в связи с этим время прогона длительностью в 20 ч уже неприемлемо. Обязательным условием параллельной обработки данных стало применение многопроцессорных компьютеров, независимо от того, разделяется ли проект на субблоки инженером вручную или это осуществляется автоматически с помощью соответствующих инструментов.

В случае проектирования процессорных систем с FPGA средства синтеза должны совместно работать с такими инструментами поставщиков FPGA как, например, Xilinx EDK или Altera SoPC Builder. Встраиваемые процессорные подсистемы, созданные с помощью EDK или SoPC Builder, можно легко интегрировать в процесс синтеза.

Вопрос качества QoR всегда стоит на повестке дня, особенно если FPGA применяются в конечном продукте. Смешанная методология «сверху вниз» и «снизу вверх» позволяет добиться компромисса между устойчивостью схемы и ее характеристиками. Средства разработки опытных образцов позволяют исключить при оптимизации большинство блоков, чтобы обеспечить высокий уровень предсказуемости проекта, тогда как разработчики FPGA имеют возможность исключить только большинство критичных блоков и с помощью инструментальных средств выполнить граничную оптимизацию блоков, улучшив временные характеристики.

Отладка стала чрезвычайно трудной задачей, особенно с использованием таких традиционных подходов как наблюдение за сигналами на определенных выводах FPGA, например, с помощью встроенного в эту микросхему логического анализатора и средств ChipScope-II или SignalTap. Кроме того, очень трудно отслеживать имена сигналов, которые могли измениться при работе средств синтеза, проводивших отладку таблицы соединений схемы. Лучший и намного более продуктивный метод заключается в отладке непосредственно в исходном коде RTL, с которым разработчики хорошо знакомы. Чем сложнее функции FPGA, тем сложнее механизмы запуска, которые необходимы для отслеживания поведения схемы. Например, возможность запуска их на конечном автомате является чрезвычайно полезной. В этом случае количество состояний устанавливается при использовании измерительных приборов. Во время отладки пользователь определяет, что происходит в каждом состоянии и при переходе, сравнивая соответствие полученных значений с расчетными. Это позволяет эффективно отыскивать трудно обнаруживаемые ошибки в схеме.

ОТСЛЕЖИВАНИЕ СОСТОЯНИЯ ПРОЕКТА И ОТЧЕТЫ

Существует несколько возможностей отслеживания проекта и представления отчетов, которые упрощают задачу руководителя, позволяя объединить в одно целое несколько подпроектов и вовремя получить их от всех групп разработчиков. Наибольшую ценность представляет возможность управлять проектом таким образом, чтобы контролировать текущий статус каждого компонента всей схемы с указанием таких нарушений как, например, несоответствие временным ограничениям или несоблюдение размеров устройства. Ниже перечислено несколько ключевых пунктов, которые должны быть указаны в отчете.

– У инженера-разработчика или руководителя группы должна быть возможность получить отчет о состоянии субблока или всего проекта. Это позволяет оценить текущее состояние дел конкретной группы разработчиков, а также проекта в целом.

– В отчет должны быть включены такие ключевые данные как пути с отрицательным резервом, использование ресурсов FPGA (RAM, DSP и LUT) или информация о конкретном типе предупреждения компилятора.

– Возможно, в некоторых случаях при реализации стиля кодирования HDL может понадобиться предупреждающее сообщение об отклонении от политики.

– Поскольку в процессе реализации может появиться много (сотни или тысячи) предупреждающих сообщений, следует отфильтровать просмотренные сообщения, чтобы в дальнейшем работать только с новыми.

– Возможность отсортировать ошибочные сообщения по типу обеспечивает более эффективный способ справиться с большим количеством ошибок и предупреждающих сообщений.

– Член группы или ее руководитель должны иметь возможность легко сравнить и проанализировать различные реализации субблоков и проекта на высшем уровне. Возможность отслеживать состояние проекта и хранить результаты, полученные для разных реализаций блока, упрощают определение лучших из них и использование всех установочных параметров инструментальных средств и файлов constraint, необходимых для воспроизведения желаемых результатов.

ВЫВОДЫ

Для проектирования и верификации устройств на базе FPGA требуются большие коллективы инженеров. Переход от групп численностью в один-два инженера к группам из 10 и более разработчиков накладывает ограничения на средства проектирования, созданные для коллективов с меньшей численностью. В результате возникла потребность в расширении возможностей средств разработки, например использования смешанной методологии, распределенного проектирования, более быстрого выполнения итераций и ускоренного формирования отчетов на всех уровнях проектирования. Эти вопросы следует рассмотреть, прежде чем приступить к проектированию систем с FPGA.

Тайное становится явным

Приложение Agilent VSA 89600B для векторного анализа сигналов – это уникальная возможность наглядного анализа истории изменения кратковременных сигналов и переходных процессов одновременно в частотной, фазовой, временной областях и в области модуляции. Найдите и исправьте самые сложные ошибки в современных системах радиолокации и беспроводной связи. Это Agilent.

ПО VSA 89600B
Возможность одновременного отображения 20 трасс и 20 маркеров
Поддержка более 70 стандартов сигналов и видов модуляции
Расширенный анализ качества модуляции во временной и частотной областях
Поддержка более 30 измерительных платформ

Приложение для векторного анализа сигналов VSA: посмотрите видео-ролик и загрузите рекомендации по применению

www.agilent.com/find/newVSA

Тел. +7 (495) 7973963

Agilent Technologies