Проектирование в условиях временных ограничений: компиляция проектов

Ростислав ГРУШВИЦКИЙ
RIGrushvitsky@mail.eltech.ru
Петр ШАШКИН
PPPetruha@yandex.ru

Со времени публикации в журнале статей авторов по проблемам компиляции ПЛИС [3, 4] прошло не так уж и много времени (чуть более трех лет), но существенные изменения, происходящие в современных САПР, заставляют вернуться к этим проблемам снова. Методики проектирования, применяемые при разработке устройств на базе ПЛИС ведущих фирм Altera и Xilinx, модифицируются при выпуске каждой новой версии САПР. Возможности таких модификаций резко возрастают. Материалы последующих статей ориентируются на возможности САПР фирмы Altera версии Quartus 10.0.

Введение

В статьях рассматриваются два важнейших этапа проектирования — этап компиляции и этап отладки проекта. Хотя материалы работы (как и следовало ожидать) в своей основе опираются на документацию фирмы-разработчика САПР и лишь в некоторой части являются результатом собственных работ авторов над реальными проектами, все же они могут представлять интерес для читателей журнала.

Рассматриваются основные аспекты концепции инкрементальной компиляции и ее составляющих: разбиение проекта на логические разделы, задание топологических назначений и сохранение результатов предыдущих итераций этой процедуры. Рассмотрено влияние процедуры инкрементальной компиляции на этап отладки проектов при использовании внутрикристального отладочного оборудования.

Этот цикл статей предназначен для разработчиков, проектирующих устройства на базе ПЛИС фирмы Altera и уже имеющих опыт работы в САПР Quartus II. Он призван сократить разрыв между возможностями, предоставляемыми данной САПР, и средствами, применяемыми разработчиками, тем самым способствуя повышению эффективности их работы.

Рынок диктует свои условия, и для того, чтобы оставаться на нем конкурентоспособным, компаниям — производителям ПЛИС приходится не просто создавать новые образцы микросхем с улучшенными техническими характеристиками, но и предоставлять разработчикам такие САПР для этих микросхем, которые бы учитывали современные требования, предъявляемые к процессу проектирования устройств на ПЛИС.

При этом из-за постоянного усложнения проектируемых систем по всем параметрам и характеристикам на первый план выходит проблема проектирования в условиях дефицита времени. Несмотря на увеличение сложности, в последнее время временные рамки разработки практически любого проекта жестко ограничены, и на весь цикл проектирования и изготовления нового прибора отводится значительно меньше времени, чем раньше при более простых проектах. В связи с этим САПР постоянно совершенствуются. Совершенствование идет во многих направлениях: благодаря росту производительности вычислительных средств, используемых при проектировании, появилась поддержка многоядерных процессоров (что позволяет распараллеливать работу САПР, а следовательно, более эффективно выполнять те этапы проектирования, которые зависят от производительности вычислительных средств, например, компиляцию), в САПР появляются новые инструменты проектирования, улучшаются алгоритмы работы уже существующих инструментов.

В результате компании публикуют огромное количество материалов к выпускаемым ими ПЛИС и САПР (различные обзоры, справочники, руководства пользователя и т. д.). Их объем постоянно растет. Так, например, руководство пользователя для ПЛИС Altera Cyclone имеет объем 503 страницы, а руководство пользователя для ПЛИС Altera Stratix IV — 1115 страниц. Руководство пользователя по САПР Altera Quartus II v6.0 — 4150 страниц (в руководствах для более поздних версий этой САПР объем текста серьезно ужат, но все равно колеблется в районе 2500 страниц).

Разработчику трудно ориентироваться в этом океане информации. В то же время он обязан иметь представление о современных методиках проектирования и знать, как использовать предоставляемые ему средства разработки в рамках этих методик с наибольшей эффективностью. Ведь сегодня именно это во многом и определяет успех реализации проекта.

Авторы поставили цель дать читателям журнала наглядное представление о современных методиках проектирования ПЛИС, рассмотреть проблемы, связанные с ними, и дать рекомендации по практической работе в САПР Quartus II фирмы Altera.

Материалы работы (как уже отмечалось выше) в своей основе опираются на документацию фирмы-разработчика САПР и лишь в некоторой части являются результатом собственных работ авторов над реальными проектами. В ряде приводимых примеров для возможности параллельной работы с оригиналом [1] обеспечено совпадение в рисунках и тексте используемых имен. В специально оговоренных случаях используются результаты работы САПР с примером, приведенным в статье [4].

При анализе причин, приводящих к постоянным изменениям современных САПР для ПЛИС, легко увидеть, что за исключением задач, связанных с периодическим появлением новых семейств ИС, основные направления модификаций посвящены изменениям процедур синтеза. Прежде всего, это переход от традиционной процедуры проектирования «снизу вверх» к другим. Основой такого перехода является возросшая сложность современных проектов и иерархический принцип их построения. Иерархия дает возможность для сложных систем на различных этапах разработки проекта разбивать его на отдельные составные части и получать предварительные результаты (иногда оценки), полнота которых зависит от детальности задания отдельных частей.

Подобный подход уже давно успешно применяется в традиционном программном проектировании, где особенно эффектив-

ны решения, опирающиеся на концепции объектно-ориентированного проектирования. Ряд специфических особенностей, связанных с синтезом аппаратуры из ее описания, тормозит перенос прогрессивных решений из компиляторов программного кода в компиляторы аппаратуры.

Основной особенностью является, как правило, присущая этим описаниям необходимость создавать синтезированные конструкции, обеспечивающие параллельность исполнения. Для программных систем наличие алгоритма (не важно, семантически правильного или нет) позволяет создать исполняемый программный код. Разработка технологического описания из функционального (алгоритмического) задания на проект — не во всех случаях посильная задача для компиляторов аппаратуры. Другое дело создание технологического описания из RTL задания проекта, для современных аппаратных компиляторов эта задача всегда принципиально разрешимая.

Создание программной модели аппаратуры (как правило, первого этапа проектирования) выполняется большинством моделирующих программ. Поскольку перед моделирующей программой ставится задача реализации модели системы без привязки к определенному аппаратному базису, то создать такую модель, как правило, существенно проще, чем создать модель, из которой далее можно перейти к синтезу аппаратной системы в заданном технологическом базисе. Именно поэтому моделирующие программы (такие как ModelSim фирмы ModelTechnology) имеют значительно меньшее число ограничений при описании проектов. Моделирующие программы позволяют определить большинство конечных и промежуточных результатов работы проекта, заданного функциональным описанием.

Объединение в одной программе компилирующей части, доводящей ряд фрагментов проекта до RTL-модели или даже до технологического уровня, и моделирующей части, позволяющей довести остальные фрагменты только до уровня функциональной модели, дало бы возможность получать и решение, и отдельные характеристики проекта. Очевидно, что создание САПР с подобной организацией — дело не столь уж отдаленного будущего.

В настоящий момент САПР Quartus II поддерживаются следующие варианты компиляции:

- сплошная компиляция (Flat Flows);
- быстрая компиляция (Smart Flows);
- компиляция сверху вниз (Team-Based Design Flows);
- инкрементальная компиляция (Incremental compilation).

Первые три типа компиляции достаточно подробно рассмотрены в предыдущих работах [3, 4], и поэтому здесь мы остановимся на особенностях инкрементальной компиляции.

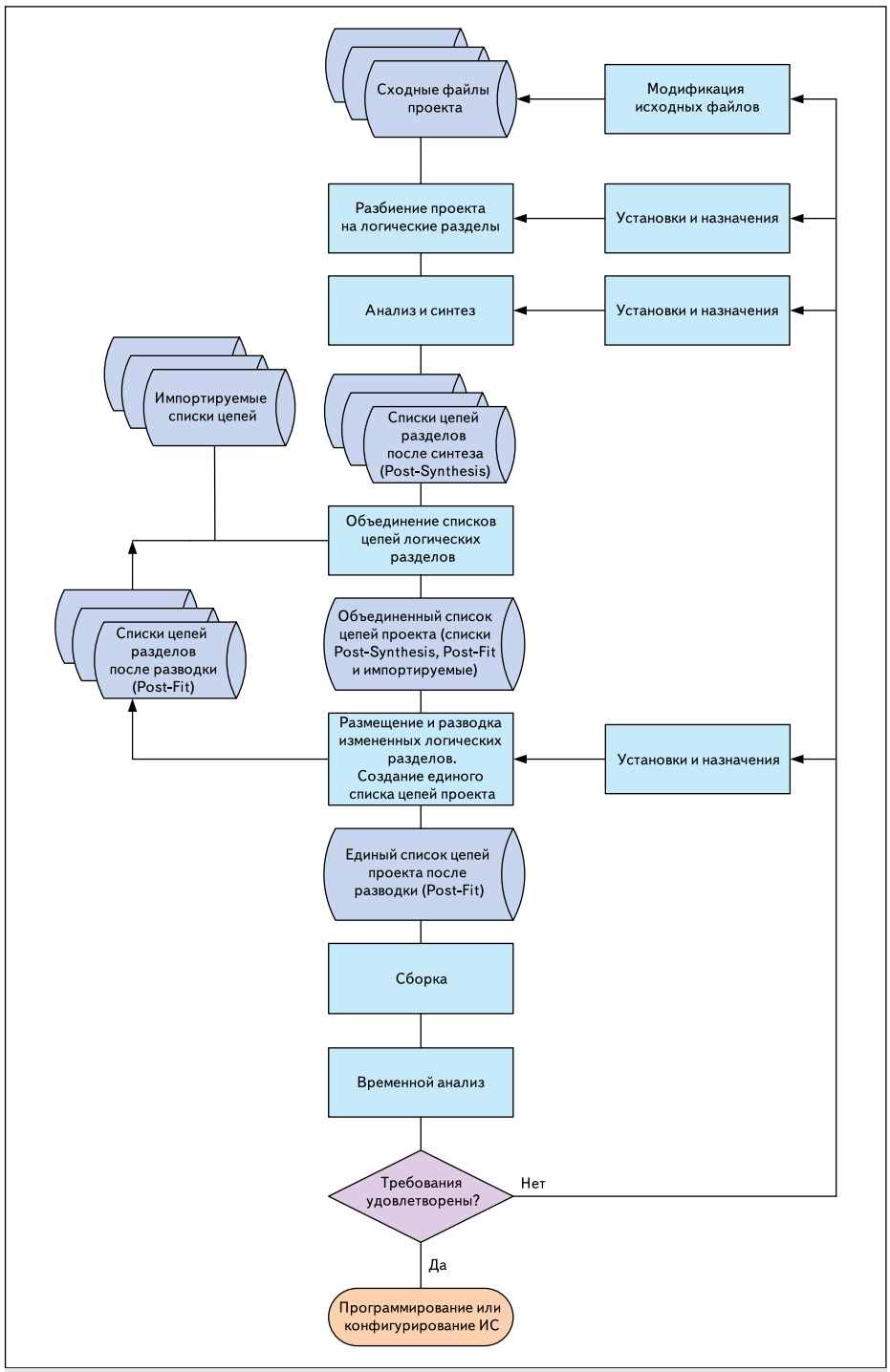

Рис. 1. Структурная схема инкрементального потока компиляции

Инкрементальный поток компиляции с разбиением проекта на разделы

В большинстве случаев инкрементальная компиляция оказывается более эффективной, чем стандартная процедура. Основное свойство инкрементальной компиляции — это разбиение проекта на блоки: логические или физические разделы (design partitions). При успешном размещении в компоновочном плане кристалла разделов проекта удается одновременно с уменьшением временных затрат на компиляцию сохранить или даже улучшить достигнутые результаты проектирования. Допустимость разбиения единого проекта на отдельные разделы позволяет применять модификации инкрементальной компиляции при различных стратегиях проектирования. Каждый вариант стратегии оказывается более предпочтительным при определенных разделах проектирования [1].

Структурная схема инкрементального потока компиляции изображена на рис. 1.

Проект разбивается на несколько разделов. Автоматически создается раздел верх-

него уровня иерархии, а также N разделов низкого уровня (задаются разработчиком). На этапе анализа и синтеза, в соответствии с заданными разработчиком установками и назначениями, компилятором производится уточнение границ разделов, а также логический синтез и технологическое отображение (Technology Mapping) каждого раздела в отдельности. В случае внесения разработчиком изменений в какой-либо раздел происходит его повторный анализ и синтез. Если же изменений не было, то список соединений для данного раздела остается прежним. В результате по завершении этапа анализа и синтеза создается индивидуальный список соединений (Post-Synthesis netlist) для каждого раздела.

На следующем этапе (Partition Merge) из отдельных списков соединений разделов создается единый полный список соединений проекта. В зависимости от используемого для каждого раздела типа списка соединений в слиянии могут быть задействованы результаты синтеза (Post-Synthesis netlist), монтирования (Post-Fit netlist) или импорта низкоуровневых проектов (Imported netlist).

Далее совмещенный список соединений проходит обработку модулем монтирования (Fitter). При этом заново размещаются только те разделы, которые были изменены. В итоге формируется список всех соединений проекта для дальнейшего использования на этапах временного анализа и генерации. Кроме того, создаются отдельные списки соединений каждого раздела (Post-Fit netlist) для дальнейшего их использования на этапе слияния (Partition Merge).

Возможность задавать различные (индивидуальные) опции компиляции и трассировки для тех или иных разделов и проекта в целом позволяет обрабатывать отдельные блоки с использованием опций расширенной оптимизации, в то время как остальная часть проекта остается неизменной в ходе последующих компиляций. Таким образом, доводка фрагмента может осуществляться не только путем схемотехнических вариаций, но и путем изменений опций проектирования исключительно для отлаживаемого фрагмента [1].

Потоки проектирования «сверху вниз» и «снизу вверх»

При использовании потока проектирования «сверху вниз» компиляция и оптимизация проекта как единого целого на верхнем уровне иерархии производится уже на первом шаге проектирования. Как правило, за это действие отвечает руководитель проекта или ведущий разработчик. Поскольку разработка часто ведется бригадными методами, то нередко оказывается так, что отдельные блоки еще не доведены до уровня реализации в ПЛИС. Ведущий разработчик может создавать для них пустые разделы. Для этих логических разделов требуется интерфейсная оболочка. Кроме того, необходимо зарезервировать размеры соответствующих им топологических разделов.

Для проекта в целом необходимо задать требуемые установки и указать интерфейсные сигналы интегральной схемы. Далее можно откомпилировать готовые разделы и проект целиком, после этого сохранить результаты трансляции и продолжить работу над новыми блоками.

При потоке проектирования «снизу вверх» на первом этапе отдельные разработчики (группы разработчиков) занимаются проектированием и оптимизацией отдельных блоков в рамках своих собственных проектов. Затем происходит объединение всех проектов нижнего уровня в один общий проект. Для поддержки этого потока проектирования существует возможность импорта и экспорта проектов. Проектировщики блоков нижнего уровня могут экспортировать списки цепей своего проекта. Ведущий разработчик импортирует каждый такой проект нижнего уровня как раздел общего проекта. Для того чтобы включить такие проекты как разделы общего проекта с неизменными свойствами, необходимо создать соответствующий LogicLock регион.

До недавнего времени сохранности ранее полученных результатов компилирования можно было добиться только при потоке проектирования «снизу вверх». Но теперь благодаря инкрементальной компиляции для этого можно использовать и поток проектирования «сверху вниз». Возможность использования потока «сверху вниз» очень важна по нескольким причинам:

- Проектирование с использованием потока «сверху вниз» осуществляется легче, чем при потоке «снизу вверх». Например, не нужно производить операции импорта-экспорта.
- При потоке «сверху вниз» у САПР с самого начала имеется информация о всем проекте в целом, и, следовательно, она может проводить глобальные оптимизации над всем проектом по его размещению и разводке. Вследствие этого зачастую хороших результатов легче добиться при проектировании «сверху вниз», чем при проектировании «снизу вверх».

Достоинством инкрементальной компиляции является то, что эта процедура очень гибкая. Она поддерживает различные методологии проектирования, при этом в каждом конкретном проекте мы не обязаны жестко придерживаться только потока проектирования «сверху вниз» или «снизу вверх». Мы можем использовать смешанный поток, когда на разных этапах используется то один поток проектирования, то другой (при реальном проектировании сложных устройств обычно так и бывает).

Продолжение следует

Литература

1. Quartus II Version 10.0 Handbook Vol. 1: Section I. Design Flows.
2. www.altera.com
3. Грушвицкий Р., Михайлов М. Проектирование в условиях временных ограничений: компиляция проектов // Компоненты и технологии. 2007. № 12.
4. Грушвицкий Р., Михайлов М. Проектирование в условиях временных ограничений: компиляция проектов // Компоненты и технологии. 2008. № 1.
5. Компиляция в Quartus II — www.dsioffe.narod.ru
6. Грушвицкий Р., Михайлов М. Проектирование в условиях временных ограничений: отладка проектов // Компоненты и технологии. 2007. № 6.
7. Грушвицкий Р., Михайлов М. Проектирование в условиях временных ограничений: отладка проектов // Компоненты и технологии. 2007. № 8.
8. Грушвицкий Р., Михайлов М. Проектирование в условиях временных ограничений: отладка проектов // Компоненты и технологии. 2007. № 9.