Экспорт и импорт разделов

Процесс восходящего проектирования – это способ разработки, при котором проект сначала делится на малые подпроекты, которые разрабатываются в виде отдельных проектов, зачастую разными разработчиками. Результаты компиляции этих низкоуровневых проектов экспортируются и передаются разработчику (или руководителю проекта), который импортирует их в головной проект для получения функционально целого проекта. Этот вид процесса проектирования требуется только тогда, когда разработчики проектов нижнего уровня собираются независимо оптимизировать их размещение и разводку, чтобы потом передать свои проекты руководителю, который будет использовать их результаты по размещении и разводке. С другой стороны, руководитель проекта может объединить исходные коды HDL нескольких разработчиков в один проект Quartus II, и использовать стандартный процесс инкрементной компиляции, как это было описано ранее.

В процессе восходящего проектирования, руководитель головного проекта должен спланировать устройство проекта, чтобы снабдить ограничениями разработчиков низкоуровневых блоков. Скрипты восходящего проектирования, генерируемые программой Quartus II, упрощают планирование в восходящем проектировании и ограничивают трудности, возникающие при объединении отдельных проектов. Обратитесь к главе "Генерация скриптов раздела восходящего проектирования для менеджера проекта" на странице 2-40.

Примеры использования командных проектов и восходящего проектирования для достижения целей проекта, находятся в главе "Рекомендованные процессы проектирования и примеры приложений компиляции" на странице 2-49. Большинство ограничений, относящихся к процессу восходящего проектирования в программе Quartus II, описаны в главе "Ограничения инкрементной компиляции" на странице 2-60.

В этой главе описаны средства экспорта и импорта, поддерживаемые процессом восходящего проектирования.

Эта глава содержит следующие разделы:

■ "Файлы экспорта разделов Quartus II (.qxp)" на странице 2–30

■ "Общие сведения об инкрементной компиляции в восходящем проектировании"

■ "Подготовка проекта для инкрементной компиляции в восходящем проектировании" на странице 2-31

■ "Экспорт низкоуровневого раздела для использования его в головном проекте" на странице 2-34

■ "Экспорт низкоуровневого блока внутри проекта" на странице 2-35
■ "Использование .qxp файла в качестве исходного файла в головном проекте" на странице 2-36
■ "Импорт низкоуровневого раздела в головной проект" на странице 2-36
■ "Импорт назначений и расширенные настройки импорта" на странице 2-38
■ "Генерация скриптов раздела восходящего проектирования для менеджера проекта" на странице 2-40
■ "Импорт SDC ограничений из низкоуровневых проектов" на странице 2-45

Файлы экспорта разделов Quartus II (.qxp)

Процесс восходящей инкрементной компиляции использует файл **.qxp** для описания низкоуровневых разделов проекта. Файл **.qxp** – это бинарный файл, содержащий результаты компиляции, описывающие экспортируемый раздел, которые содержат список соединений пост-синтез или пост-компоновка, и устанавливают назначения, обычно содержащие ограничения размещения регионов LogicLock. Файл **.qxp** не содержит оригинальный исходный файл низкоуровневого проекта.

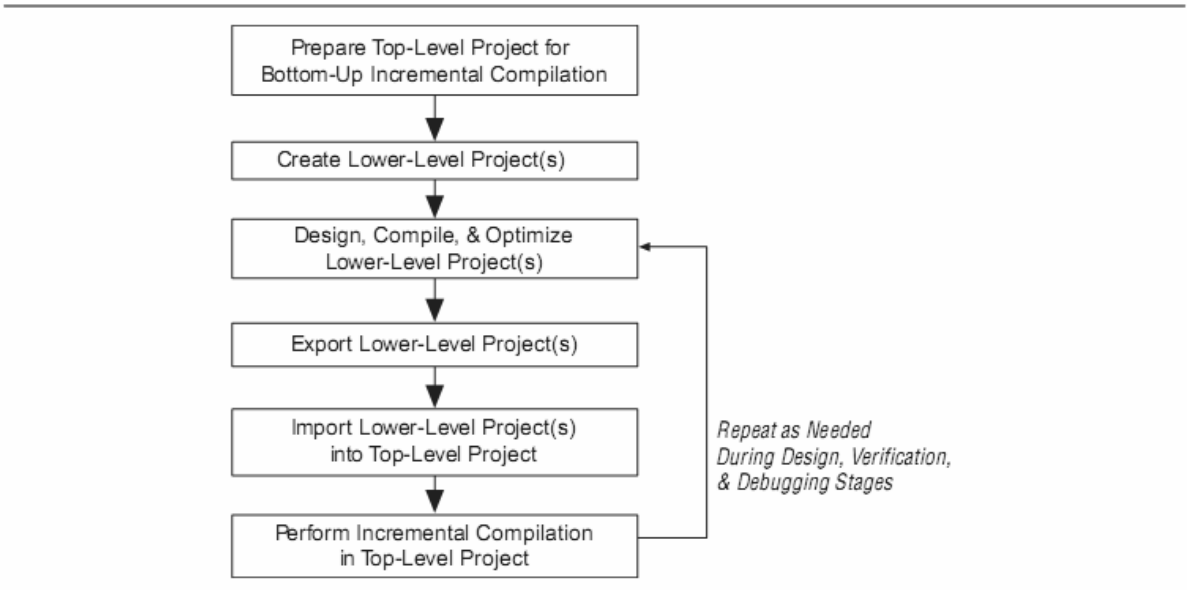
В следующих разделах представлено описание процесса инкрементной компиляции в восходящем проектировании, и описывается, как генерировать файл **.qxp** для низкоуровневого раздела и как импортировать его в головной проект.

Общие сведения об инкрементной компиляции в восходящем проектировании

Блок-схема на рисунке 2-12 показывает процесс инкрементной компиляции, использованный в методе восходящего проектирования, при котором низкоуровневые разделы компилируются отдельно, прежде чем будут импортированы в головной проект. Следующие подразделы описывают шаги, закреплённые в этом процессе.

Сначала головной проект готовится для инкрементной компиляции. Затем разрабатываются, оптимизируются, верифицируются и отлаживаются низкоуровневые проекты. Экспортируется иерархия каждого низкоуровневого проекта в виде файлов Quartus II **.qxp**, и импортируются файлы **.qxp** в головной проект. В заключении, компилируется весь головной проект.

Figure 2–12. Summary of Bottom-Up Incremental Compilation Flow



Подготовка проекта для инкрементной компиляции в восходящем проектировании

Для подготовки проекта для метода восходящего проектирования, руководитель проекта или разработчик головного модуля должен выполнить следующие шаги:

1. Создать головной проект Quartus II, применить расширенные настройки проекта и глобальные назначения.
   a. Создать исходный код для "скелета" этого проекта, который включает в себя иерархию и интерфейс портов для проектов нижнего уровня. Файл головного проекта состоит из блоков нижнего уровня, которые вы планируете компилировать в качестве отдельных проектов Quartus II. Если вы хотите компилировать проект в отсутствии блоков нижнего уровня, создайте пустые исполняемые файлы черных ящиков для каждого блока проекта, чтобы определить сами блоки и порты.
   b. Создайте все глобальные назначения, состоящие из назначений чипа, выводов и временных характеристик, так чтобы конечный проект имел собственные технические требования. Разработчики низкоуровневых проектов могут добавить свои собственные ограничения для своих разделов, а затем передать их разработчику головного проекта, но расширенные ограничения проекта, которые влияют более чем на одно ограничение проекта нижнего уровня, должны передаваться разработчиком головного проекта или руководителем проекта, чтобы избежать многих конфликтов и добиться того, чтобы в проектах нижнего уровня использовались корректные назначения.
2. Для каждого блока проекта нижнего уровня, импортируемого в головной проект, разработать модуль в виде раздела проекта с типом списка соединений **Пусто**. Обратитесь к главам "Создание назначений разделов проекта" на странице 2-16 и "Установка типа списка соединений для раздела проекта" на странице 2-19.
3. Если руководитель проекта планирует импортировать информацию о размещении из низкоуровневых проектов, создайте регионы LogicLock для каждого низкоуровневого раздела, чтобы создать архитектуру проекта. Обратитесь к главе "Создание архитектуры проекта с назначениями локализации LogicLock" на странице 2-25.
4. Опционально, выполните полную компиляцию для каркаса проекта. В меню **Проект**, кликните **Генерировать Скрипты раздела в восходящем проектировании**. Передайте каждому разработчику проекта нижнего уровня сгенерированный **Tcl** файл, чтобы он смог создать собственный проект с необходимыми ограничениями. Если вы используете **команды создания файла** в разработке вашего проекта, снабдите **созданием файла** каждый раздел. Обратитесь к главе "Генерация скриптов раздела восходящего проектирования для менеджера проекта" на странице 2-40.

Создание и компиляция проектов нижнего уровня

Разработчик каждого проекта нижнего уровня должен создать отдельный проект Quartus II.

Если вы создаёте проект вручную, создайте новый Quartus II проект для подпроекта со всеми требуемыми настройками. Создайте его с назначениями для регионов LogicLock и глобальными назначениями (включая настройки тактов), как это определено руководителем проекта, также создайте назначения виртуальных выводов для портов, которые описывают соединения логики ядра, включая внешние выводы чипа, в головном модуле.

Если у вас есть скрипт раздела для восходящего проектирования от разработчика головного проекта, используйте Tcl скрипт для создания проекта Quartus II со всеми требуемыми настройками и назначениями из головного проекта.

Если вы используете команды создания файла, используйте эту команду, чтобы создать проект Quartus II со всеми требуемыми настройками и назначениями, а потом скомпилируйте проект. Определите зависимости в созданном файле, чтобы понять, какой исходный файл должен быть ассоциирован с каким разделом.

Компилируйте и оптимизируйте каждый проект нижнего уровня как отдельный проект Quartus II.

Экспорт низкоуровневых проектов

Когда вы достигли проектных требований, предъявляемых проектам нижнего уровня, экспортируйте каждый проект в виде раздела для проекта верхнего уровня.

Если вы не используете команды создания файла, в меню Проект, используйте диалоговое окно Экспорт разделов проекта для экспорта каждого проекта нижнего уровня. Обратитесь к главе "Экспорт низкоуровневого раздела для использования его в головном проекте" на странице 2-34. Если вы хотите экспортировать только часть в проекте нижнего уровня, обратитесь к главе "Экспорт низкоуровневого блока внутри проекта" на странице 2-35. Каждый разработчик проекта нижнего уровня должен передать файл **.qxp** руководителю проекта.

Если ваша команда разработчиков использует команду создания файла, руководитель проекта может использовать команду **создать** с **master_makefile** для экспорта низкоуровневых разделов и создать файлы .**qxp**, чтобы затем импортировать их в головной проект.

Включение или импорт низкоуровневых проектов в головной проект

После экспорта проектов нижнего уровня, руководитель проекта может объединить файлы **.qxp**, присланные разработчиками каждого подпроекта нижнего уровня.

Если вы хотите использовать экспортируемую информацию файла **.qxp** в качестве файлов проекта в головном проекте, просто добавьте **.qxp** файлы в качестве исходников в проект. В этом случае, модули в **.qxp** файле не будут разделами в головном проекте. Обратитесь к главе "Использование .qxp файла в качестве исходного файла в головном проекте" на странице 2-36.

Если вы хотите импортировать только информацию о размещении, в меню **Проект**, кликните **Импорт раздела проекта** и определите раздел в головном проекте, который будет описан файлом подпроекта **.qxp**. Обратитесь к главе "Импорт низкоуровневого раздела в головной проект" на странице 2-36. Повторите этот процесс для каждого раздела в проекте, который вы хотите импортировать.

Вы можете автоматизировать процесс импорта, используя команды создать файл: команда **master_makefile** импортирует каждый раздел в головной проект. Убедитесь в том, что вы определили, какие исходные файлы должны быть ассоциированы с каким разделом, так чтобы программа смогла заново собрать проект, если будут внесены изменения в исходные файлы.

Подробно о том, какие назначения импортируются и как избежать конфликтов, в главе "Импорт назначений и расширенные настройки импорта" на странице 2-38.

Выполнение инкрементной компиляции в головном проекте

После того, как вы импортировали разделы проекта, чтобы создать головной проект, вы можете выполнить полную компиляцию. Программа компилирует импортированные разделы тем же способом, что и разделы, определённые в головном проекте. Программа перекомпилирует импортированные разделы только, если они были импортированы с момента последней компиляции.

Типы списков соединений для импортированных разделов

Разделы, которые импортируются из других проектов, используют два дополнительных типа списков соединений, а головной проект использует тип списка соединений **Пусто** для создания резерва площади для разделов, пока они не ассоциированы с .**qxp** файлами импорта других разработчиков. Эти типы списков соединений описаны в таблице 2-5.

Таблица 2-5. Типы списков соединений для импортированных разделов

Тип списка соединений раздела	**Поведение программы Quartus II во время компиляции раздела**
Импортированный	Компиляция раздела, используя тип списка соединений, импортируемый из .**qxp** файла. Программа не может модифицировать или перезаписывать во время компиляции оригинальный импортированный список соединений. Для сохранения изменений сделанных в импортированном списке соединений (например, при перемещении импортированных регионов LogicLock), используйте следующую настройку Пост-компоновка (основано на импорте) для окончательной компиляции с использованием импортированных списков соединений. Обратитесь к главе "Экспорт и импорт разделов проекта в восходящем проектировании" на странице 2-29. Уровень сохранения компоновки показывает, какой уровень информации сохраняется из списка соединений пост-компоновка. Обратитесь к главе "Уровень сохранения компоновки" на странице 2-21. Эта настройка не доступна, если вы не импортируете список соединений, используя команду **Импорт разделов проекта**.
Пост-компоновка (основано на импорте)	Сохранение результатов пост-компоновки для раздела и повторное использование списка соединений пост-компоновка так долго, пока сохраняются следующие условия: ■ список соединений пост-компоновки доступен после предыдущей компоновки; ■ не были сделаны изменения для ассоциации импортированного списка соединений после предыдущей компоновки. Компиляция раздела с импортированным списком соединений не возможна, если изменяется импортированный список соединений (когда он заново импортирован), или если список соединений не доступен. Изменение назначений не является основанием для перекомпиляции. Уровень сохранения компоновки показывает, какой уровень информации сохраняется из списка соединений пост-компоновка. Подробнее в главе "Уровень сохранения компоновки" на странице 2-21. Вы можете использовать этот тип списка соединений для сохранения изменений при размещении и разводки импортированного списка соединений. Эта настройка не доступна, если вы не импортируете список соединений, используя команду **Импорт разделов проекта**.
Пусто	Используется пустой резервирующий место список соединений для разделов и автоматически добавляются виртуальные выводы на границах раздела. Вы можете использовать этот тип списка соединений для пропуска компиляции раздела нижнего уровня, чтобы импортировать его позже. Обратитесь к главе "Пусты разделы на странице" 2-22.

Экспорт низкоуровневого раздела для использования его в головном проекте

Каждый низкоуровневый подпроект компилируется как отдельный Quartus II проект. В каждом проекте используйте эти инструкции для выполнения процессов экспорта и импорта:

■ Если у вас есть скрипт раздела восходящего проектирования для головного проекта, используйте Tcl скрипт для создания проекта и всех назначений из головного проекта. Действия по созданию многих назначений описаны ниже. Проследите, чтобы регион LogicLock использовал только ресурсы, отведённые ему руководителем головного проекта.

■ Оцените, какие тактовые сигналы должны быть использованы в ресурсах глобальной трассировки, - этим вы избежите конфликтов в головном проекте.

- ■ Установите назначение **Глобальный сигнал ВКЛ** для того, чтобы сильно ветвящиеся сигналы были распределены по глобальным сетям.
- ■ Для предотвращения размещения других сигналов в глобальных сетях, в меню **Назначения** выберите **Настройки** и выключите **Автоматический глобальный такт** и **Автоматический глобальный контроль регистров** под кнопкой **Больше настроек** на странице **Компоновщик** во вкладке **Настройки**.
- ■ Альтернативно, вы можете установить назначения **Глобальный сигнал ВЫКЛ** для сигналов, которые не должны быть размещены в глобальных сетях. Дальнейшее размещение LAB зависит от того, какие входы логики ячеек LAB будут использовать глобальные такты. Вы можете столкнуться с проблемой, когда сигналы в низкоуровневом блоке не используют глобальные сети, а используют их в головном проекте.

■ Используйте назначения для **виртуальных выводов**, для того, чтобы отобразить выводы подпроекта, которые не являются выводами в головном блоке. Это критично, когда подпроект имеет больше выводов, чем доступно ножек у чипа для проекта. Использование виртуальных выводов поможет вам также оптимизировать межблочные соединения в конечном проекте, снабжая дополнительной информацией все порты ввода-вывода подпроекта, такой как назначения локализации и временные назначения.

■ Поскольку разделы компилируются независимо, в отсутствие информации об остальных разделах, вам потребуется владеть большей информацией об временных путях, которые могут влиять на другие разделы в головном проекте. Вам потребуется использовать назначения локализации для каждого вывода, чтобы определить, где подключенный порт был локализован при его добавлении в головной проект. Вам также необходимо сделать временные назначения для портов I/O подпроекта, чтобы выполнить временное планирование.

Дополнительно о том, как выполнить балансировку ресурсов и временное планирование в главе "Лучшие примеры для инкрементной компиляции разделов и назначений архитектуры" в томе 1 настольной книги Quartus II.

Если ваш подпроект откомпилирован, используя эти инструкции, и подготовлен для объединения с головным проектом, экспортируйте подпроект как раздел, используя эти инструкции:

1. В подпроекте используйте один из следующих способов, чтобы открыть вкладку **Экспорт раздела проекта**:
   - ■ В **Планировщике Раздела Проекта** (доступен в меню **Инструменты**), правым кликом на подсвеченный блок раздела, и выберите **Экспорт раздела проекта**;
   - ■ В меню Проект, кликните **Экспорт раздела проекта**.

2. Во вкладке **Экспорт файла** выберите файл с расширением **.qxp**. По умолчанию, директория с файлом совпадает с директорией текущего проекта.
3. Вы можете выбрать также **Иерархия раздела на экспорт**. По умолчанию, главный раздел (собственно проект) будет экспортирован, но вы можете выбрать для экспорта результаты компиляции другой иерархии в проекте, как это описано в главе "Экспорт низкоуровневого блока внутри проекта" на странице 2-35. Выберите иерархию раздела для экспорта в раскрывшемся списке.
4. Под **Списком соединений на экспорт** выберите либо **список соединений пост-компоновка**, либо **список соединений пост-синтез**. По умолчанию это **список соединений пост-компоновка**. Для списка соединений пост-компоновка включите или выключите опцию **Экспорт разводки**, если требуется.
5. Кликните **ОК**. Программа Quartus II создаст файл **.qxp** в выбранной директории.

Альтернативно, вы можете настроить ваш проект так, что процесс экспорта будет выполняться каждый раз при компиляции:

1. В меню **Назначения**, кликните **Настройки**.
2. В диалоговом окне **Настройки**, под **Настройками процесса компиляции**, выберите вкладку **Инкрементная компиляция**.
3. Включите **Автоматический экспорт разделов проекта после компиляции**.
4. Если вы хотите посмотреть или изменить настройки экспорта по умолчанию, кликните на кнопку **Настройки экспорта раздела проекта**.
5. В диалоговом окне **Настройки экспорта раздела проекта**, измените настройки, если это требуется для процедуры экспорта в пунктах 2 – 4. Кликните **ОК**.
6. Кликните **ОК** для закрытия диалогового окна **Настройки**. Во время следующей полной компиляции, программа создаст файл **.qxp** в выбранной директории.

Экспорт низкоуровневого блока внутри проекта

Пункт 3 в главе "Экспорт низкоуровневого раздела для использования его в головном проекте" разрешает вам создать файл **.qxp** для низкоуровневого блока внутри проекта Quartus II. Когда вы проделываете это, команда экспортирует всю иерархию выбранного раздела в **.qxp** файл.

Вы можете воспользоваться этим свойством, чтобы добавить тестовую логику вокруг низкоуровневого блока для экспорта в виде раздела для головного проекта. Вы можете также установить дополнительные компоненты проекта в низкоуровневый проект так, чтобы они воссоздали окружение модуля в головном проекте. Например, вы включили ПЛЛ головного проекта в ваш проект нижнего уровня, так вы сможете оптимизировать проект, обладая информацией о частоте умножителей, фазовых сдвигов, компенсационной задержке и прочих параметрах ПЛЛ. Программа фиксирует временные и ресурсные ограничения более точно, потому что в низкоуровневом проекте обеспечивается более полный и точный временной анализ. Вы можете экспортировать низкоуровневый раздел в головной проект без вспомогательных компонентов.

Дополнительно, вы можете использовать это свойство в процессе нисходящего проектирования для создания файлов **.qxp** определённых завершенных разделов. Вы можете делать обратный импорт файлов **.qxp** и использовать **импортированный список соединений**, как будет описано в следующей главе. В этом случае, файл **.qxp** используется как архив раздела, хранящий данные о списке соединений, разводке и размещении в одном файле. Если вы меняете исходный код для раздела, вам необходимо изменить тип списка соединений обратно на **Исходный Файл**, чтобы использовать исходный элемент импортированной информации.

Использование .qxp файла в качестве исходного файла в головном проекте

При включении в проект списка соединений из **.qxp** файла, вы можете запросто использовать **.qxp** файл в качестве исходного файла в вашем проекте (также как исходные файлы на языках Verilog и VHDL).

Файл **.qxp** содержит блок проекта, экспортированного из подпроекта, и имеет то же имя, что и раздел. Когда вы устанавливаете блок проекта в головном проекте и включаете в него **.qxp** файл в качестве исходного файла проекта, программа добавляет экспортированный список соединений в базу данных головного проекта. Имена портов **.qxp** чувствительны к регистру, если оригинальный HDL в разделе нижнего уровня чувствителен к регистру.

Программа также фильтрует назначения из подпроекта, чтобы установить соответствующие назначения в головном проекте. Обратитесь к главе "Импорт назначений и расширенные настройки импорта" на странице 2-38, в которой описывается, какие назначения должны быть в головном проекте. Назначения в **.qxp** файле обрабатываются аналогично назначениям, сделанным в исходном HDL файле, и могут быть перезаписаны назначениями из головного проекта.

Когда вы используете **.qxp** файл в качестве исходного файла, в этом случае, вы не можете свободно импортировать любую базу данных пост-компоновки в ваш проект. Если вы хотите импортировать информацию пост-компоновки из экспортированного списка соединений, обратитесь к главе "Импорт проекта нижнего уровня в головной проект".

Когда вы используете **.qxp** файл в качестве исходного файла, вы можете выбрать, захотите ли вы иметь этот файл в качестве раздела головного проекта? Если вы не разрабатывали модуль **.qxp** в качестве раздела, то программа удалит неподключенные порты и неиспользуемую логику точно так же, как и в исходном файле. Если вы назначили модуль в качестве раздела проекта, границы раздела всегда сохраняются, как это описано в главе "Влияние разделов проекта на оптимизацию проекта" на странице 2-11.

Если вы используете команду **Локализовать** в отношении файла **.qxp** или пытаетесь открыть **.qxp** файл в программе Quartus II, то программа откроет файл информации об экспортированном разделе, в котором разъясняется содержание файла и предоставляется список портов. Поскольку **.qxp** файл – это бинарный файл, вы самостоятельно не сможете увидеть список соединений проекта.

Импорт низкоуровневого раздела в головной проект

Процесс импорта подразумевает импорт списка соединений проекта из файла **.qxp** в отдельный раздел и добавление этого списка соединений в базу данных головного проекта. Импортирование позволяет вам повторно использовать результаты пост-компоновки экспортированного раздела. Импортирование фильтрует назначения из подпроекта, чтобы создать соответствующие назначения в головном проекте. Прежде чем вы будете импортировать раздел, вы должны выполнить **разработку** иерархии проекта и назначить разделы проекта. Если вы разработали проект с **пустыми** разделами и не создали упаковочные файлы чёрных ящиков для

определения соединений портов, **Анализ и Разработка** сгенерирует сообщение об ошибке неопределённых портов, но вы всё равно можете продолжать процесс импорта.

Для импорта низкоуровневого подпроекта в головной проект, необходимо осуществить следующие шаги:

1. В головном проекте, используйте один из следующих методов для открытия вкладки **Импорт Раздела Проекта**:
   ■ В **Планировщике Разделов проекта**, правым кликом внутри выделенного раздела и кликните **Импорт Раздела проекта**;
   ■ В окне **Разделы Проекта**, правым кликом на раздел, который вы хотите импортировать, и кликните **Импорт Раздела Проекта**;
   ■ В меню **Проект**, кликните **Импорт Раздела Проекта**.

2. Во вкладке **Разделы** отыщите требуемый раздел. Для выбора раздела, подсветите имя раздела во вкладке **Выбор Разделов** и используйте соответствующую кнопку для выбора или отмены выбора требуемого раздела.

Вы можете выбрать несколько разделов, если ваш головной проект состоит из нескольких блоков, и вы хотите использовать их списки соединений.

3. Под **Импортом файла**, определите файл с расширением **.qxp** или найдите файл, который вы будете использовать для импорта выбранного раздела. Этот файл будет использоваться только на стадии импорта и не будет использован во время компиляции, если только вы не будете делать повторного импорта раздела.

Если вы уже неоднократно импортировали файл .**qxp** для этого раздела, вы можете использовать ту же самую локализацию, что и в предыдущем импорте, основываясь на определённом имени файла. Чтобы это сделать, включите опцию **Реимпорт, используя последний импортированный файл и предыдущую локализацию**. Эта опция специально предназначена для того, чтобы импортировать новые файлы .**qxp** для нескольких разделов, которые вы уже неоднократно импортировали. Вы можете выбрать все разделы для импортирования во вкладке **Разделы**, и затем, использовать опцию **Реимпорт, используя последний импортированный файл и предыдущую локализацию** для импорта всех разделов, используя предыдущую локализацию, без определения индивидуальных имён файлов.

4. Опционально: Чтобы посмотреть содержимое выбранного .**qxp** файла, кликните **Загрузить Свойства**. Свойства содержат информацию о Типе Списка Соединений, Названии элемента, чипе и статистической информацией о размере и портах.
5. Опционально: Кликните **Расширенные настройки импорта** и выберите, какие регионы и назначения будут импортированы из подпроекта в головной проект. Во время импорта, эти регионы могут быть слегка изменены в размерах и положении. Кликните **ОК** для применения установок. Детально об импортируемых установках и избежании конфликтов в главе "Импорт назначений и расширенные настройки импорта" на странице 2-38.
6. Для начала импорта в окне **Импорта Разделов Проекта** кликните **ОК**. Выбранный файл .**qxp** будет импортирован в базу данных текущего головного проекта.

Импорт назначений и расширенные настройки импорта

Когда вы импортируете подпроект в головной проект, программа делает определенные установки по умолчанию и, таким образом, импортирует важные назначения из подпроекта в головной проект.

Свойства разделов проекта после импортирования

Когда вы импортируете раздел подпроекта, то в процессе импорта разделу устанавливается тип списка соединений **Импортированный**.

Если вы компилируете проект и вносите изменения в результаты размещения и разводки, используйте тип списка соединений **Пост-компоновка (основан на импорте)** для дальнейшей компиляции. Для отмены импортированного списка соединений и перекомпиляции из исходного кода, компилируйте раздел с установкой типа списка соединений **Исходный файл**, тем самым вы включаете в головной проект важный исходный код.

В процессе импорта разделам устанавливается **Уровень Сохранения Компоновки** наивысшего уровня, поддерживаемого импортируемым списком соединений. Например, если список соединений пост-компоновка импортирован с информацией о размещении, уровень будет установлен как **Размещение**, но вы можете изменить его на тип **Только Список Соединений**.

Обратитесь к главе "Установка типа списка соединений для разделов проекта" на странице 2-19 для подробного описания установок типов списка соединений и уровней сохранения компоновки.

Импортирование назначений раздела из подпроекта

Назначения раздела проекта, определенные внутри подпроекта, не импортируются в головной раздел. Вся логика подпроекта импортируется как один раздел в **.qxp** файле.

Файлы Ограничений Проекта Synopsys для временного анализатора Quartus II TimeQuest

Временные назначения, сделанные для временного анализатора Quartus II TimeQuest в файле ограничений проекта Synopsys (.**sdc**) не импортируются в головной проект. Убедитесь лично, что головной проект содержит все временные ограничения для этого проекта.

Обратитесь к главе "Импорт SDC ограничений из низкоуровневых проектов" на странице 2-45 за рекомендациями об управлении SDC ограничениями в головном проекте и проектах нижнего уровня.

Импортирование назначений LogicLock

Во время импорта все регионы LogicLock имеют фиксированный размер. Если вы используете несколько блоков из подпроекта в головном проекте, импортированным регионам необходимо установить **Произвольную локализацию**. Иначе, все они будут иметь **Фиксированную локализацию**. Вы можете изменить размещение регионов LogicLock после импорта, или меняете их тип на **Произвольную локализацию**, чтобы позволить программе размещать каждый регион, но сохранить при этом взаимную локализацию узлов внутри региона насколько это возможно. Для сохранения изменений, сделанных в разделах после компиляции, используйте список соединений **пост-компоновка (основан на импорте).**

Назначение **состояния элемента LogicLock** устанавливается **Закрытым** для обозначения фиксации региона.

Обратная аннотация LogicLock и данные о локализации узлов не импортируются, поскольку Quartus II Файл Экспорта Разделов (**.qxp**) уже содержит всю необходимую информацию о локализации. Altera настоятельно рекомендует не добавлять или не удалять элементы импортированного региона LogicLock.

Импорт других назначений для блоков

Импортируются все назначения для блоков, исключая назначения для разделов проекта, SDC констант, назначений LogicLock регионов, как было описано выше.

Импорт глобальных назначений

Глобальные назначения не импортируются. Руководитель проекта может делать глобальные назначения только в головном проекте. Обратите внимание на то, что настройки тактовых сигналов для Классического временного анализатора Quartus II являются глобальными назначениями, и также не импортируются. Когда это возможно, некоторые ограничения, глобальные назначения конвертируются в специальные назначения для модуля в отношении определённого раздела проекта.

Расширенные настройки импорта

Вкладка **Расширенные настройки импорта** позволяет вам определить специальные опции контроля над тем, какие назначения и регионы были интегрированы, и как разрешить конфликты, возникшие во время импорта раздела подпроекта в головной проект. Следующие подразделы описывают эти опции.

Позволить создавать новые назначения

Позволяет команде импорта добавлять новые назначения из импортируемого проекта в головной проект. Когда эта опция выключена, то импортированные назначения обновляются до существующих назначений, но не позволяется создавать новые.

Снабдить назначениями все модули в импортированном блоке

Конвертирует и снабжает назначениями на уровне объекта из подпроекта в назначения на уровне элемента в головном проекте.

Назначение разрешения конфликтов: регионы LogicLock

Выберите одну из следующих опций, чтобы определить, как справиться с конфликтными назначениями LogicLock (т.е., назначения подпроекта, которые не соответствуют назначениям головного проекта):

■ **Всегда замещать регионы в текущем проекте (по умолчанию)** – удаляет существующие регионы и замещает их новым регионом подпроекта. Некоторые изменения, сделанные в LogicLock регионе после импорта назначений, также удаляются.

■ **Всегда обновлять регионы в текущем проекте** – перезапись существующих назначений регионов, чтобы отразить только новые назначений подпроекта, за исключением **направляющих** регионов LogicLock, в этом случае руководитель проекта должен сделать назначения локализации в архитектуре головного проекта.

■ **Пропуск конфликтных регионов** – игнорирует и не импортирует назначения подпроекта, имеющие конфликты с другими назначениями в головном проекте.

Назначение разрешение конфликтов: прочие назначения

Выберите одну из следующих опций, чтобы определить, как справиться с конфликтами прочих назначений (т.е., назначения подпроекта, которые не соответствуют назначениям головного проекта):

■ **Всегда заменять назначения в текущем проекте (по умолчанию)** – переписывает и обновляет все назначения для существующих подпроектов новыми назначениями подпроектов.

■ **Пропускать конфликтные назначения** - игнорирует и не импортирует назначения подпроекта, имеющие конфликты с другими назначениями в головном проекте.