Установка типа списка соединений для раздела проекта

Тип списка соединений – свойство каждого раздела проекта, которое позволяет вам выбрать тип списка соединений или исходный файл, который будет использовать компилятор при входе в каждый раздел. Тип списка – свойство контроля за процессом инкрементной компиляции, как это описано в главе "Компиляция проекта с использованием инкрементной компиляции" на странице 2-9. Это свойство определяет, какой список соединений будет использован на стадии Сшивателя Разделов во время следующей компиляции.

Для просмотра и модификации типа списка соединений, в меню **Назначения**, кликните **Окно разделов проекта**. Двойной клик на **Тип списка** соединений модуля. Другой способ, правыми кликом на модуль, кликните **Свойства раздела проекта**, затем изменяйте **Тип списка** соединений на вкладке **Компиляция**.

В таблицах 2-2 и 2-3 описаны стандартные и расширенные настройки типа списка соединений, объяснено поведение программы Quartus II после каждой настройки, и определены последствия использования каждой настройки. За примерами, в которых описывается, как использовать эти настройки для достижения различных целей, обратитесь к главе "Рекомендованные процессы проектирования и примеры приложений компиляции" на странице 2-49.

Таблица 2-2. Стандартные настройки типов списка соединений

Тип списка соединений раздела	**Поведение программы Quartus II во время компиляции раздела**
Исходный файл	Всегда компилирует раздел, используя ассоциированный исходный файл(ы). Используйте этот Тип списка соединений для перекомпиляции разделов из исходного кода, используя новые настройки синтеза и компоновщика.
Пост-синтез	Сохранение результатов пост-синтеза для раздела и повторное их использование в списке соединений пост-синтез, пока сохраняются следующие условия: ■ Пост-синтез список соединений доступен после предыдущего синтеза ■ Не было изменений, способных перезапустить автоматически синтез раздела, с момента предыдущего синтеза. Подробнее в главе "Какие изменения запускают автоматический синтез раздела?" на странице 2-23. Компиляция разделов происходит из исходных файлов, если запущен автоматический синтез, или если список соединений пост-синтез не доступен. Использование этого списка соединений сохраняет результаты синтеза, пока вы не сделаете изменения в проекте, при этом вы можете перекомпоновывать разделы, используя новые настройки Компоновщика.
Пост-компоновка	Сохранение результатов пост-компоновка для раздела и повторное их использование в списке соединений пост-компоновка, пока сохраняются следующие условия: ■ Список соединений пост-компоновка доступен после последней компиляции ■ Не было изменений, способных перезапустить автоматически синтез раздела, с момента предыдущего синтеза. Подробнее в главе "Какие изменения запускают автоматический синтез раздела?" на странице 2-23. Когда список соединений пост-компоновка не доступен, программа использует повторно список соединений пост-синтез (если он доступен), или компилирует из исходного файла. Раздел компилируется из исходного файла, если запускается синтез. Уровень сохранения компоновки определяет, какой уровень информации сохраняется в списке соединений пост-компоновка. Подробнее в главе "Уровень сохранения компоновки" на странице 2-21. Используйте этот тип списка соединений для сохранения результатов компоновки до тех пор, пока вы не сделаете изменения в проекте. Вы можете использовать этот список соединений для применения глобальной оптимизации, например, Оптимизация физического синтеза, которая происходит в Компоновщике, для определения разделов, которые сохраняют результаты компоновки для других разделов.

Таблица 2-3. Расширенные настройки типов списка соединений

Тип списка соединений раздела	**Поведение программы Quartus II во время компиляции раздела**
Пост-компоновка (Строго)	Всегда сохраняет результаты пост-компоновка для раздела, несмотря на то, что были сделаны изменения для ассоциированного с ним исходного файла с момента последней компоновки. Неправильное использование типа списка соединений пост-компоновка (строго) может привести к функционально некорректным спискам соединений, когда изменяется исходных файл проекта. Используйте список соединений пост-разводки каждый раз, когда была сделана ассоциация исходных файлов после последней разводки. Осторожно применяйте эти назначения. Дополнительно в "Форсированное использование списка соединений пост-компоновка после изменения раздела" на странице 2-25. Когда список соединений пост-компоновка не доступен, программа использует повторно список соединений пост-синтез (если он доступен), или компилирует из исходного файла. Уровень сохранения компоновки определяет, какой уровень информации сохраняется в списке соединений пост-компоновка. Подробнее в главе "Уровень сохранения компоновки" на странице 2-21.
Пусто	Использует список соединений Пусто для резервирования площади под раздел и автоматически добавляет виртуальные выводы на границах раздела. Вы можете использовать этот список соединений для пропуска компиляции раздела. Дополнительно о настройках Пусто в главе "Пустые разделы" на странице 2-22.

Уровень сохранения компоновки

Уровень сохранения компоновки – это свойство, определяющее, какая информация используется компилятором из списка соединений пост-компоновка.

В меню **Назначения**, кликните на **Окно разделов проекта**. Для того, чтобы посмотреть и модифицировать уровень сохранений компоновки, дважды кликните на модуль. Другой способ, правым кликом выберите **Свойство**, затем отредактируйте **Уровень сохранения компоновки** во вкладке **Компиляция**.

В таблице 2-4 описаны настройки уровня сохранения компоновки.

Таблица 2-4. Настройки уровня сохранения компоновки (часть 1 из 2)

Уровень сохранения компоновки	**Поведение программы Quartus II во время компиляции раздела**
Размещение	Сохраняет атомы списка соединений и их размещение в разделе проекта. Переразводит раздел проекта. Эта настройка сберегает значительное время компиляции, поскольку Компоновщику не требуется перекомпоновывать узлы в разделе.
Размещение и разводка	Сохраняет атомы списка соединений, их размещение и разводку. Минимальный уровень сохранения требуется для сохранения изменений ECO, сделанных в списке соединений пост-компоновка, и добавленных к проекту выводов SignalProbe. Эта настройка ещё больше, по сравнению с только Размещением, уменьшает время компиляции, но при этом снижается

	гибкость для разводчика, для внесения изменений, если изменены другие части проекта.

Таблица 2-4. Настройки уровня сохранения компоновки (часть 2 из 2)

Уровень сохранения компоновки	**Поведение программы Quartus II во время компиляции раздела**
Размещение, разводка и высокоскоростные элементы	Сохраняет атомы списка соединений, их размещение и разводку в разделе проекта, а также настройки высокоскоростных энергетических ячеек. Эта настройка увеличивает сохранение характеристик для критичных временных путей, поскольку позволяет переключать низкоскоростные ячейки на высокоскоростные, если требуется в оставшемся изменённом проекте. Эта настройка доступна только в чипах с конфигурируемыми энергетическими ячейками.
Только список соединений	Сохраняет атомы списка соединений раздела проекта, но перемещает и переразводит сам раздел проекта. Список соединений пост-компоновка с сохранёнными атомами может отличаться от списка соединений пост-синтез, поскольку в нём содержится оптимизация Компоновщика; например, изменения физического синтеза, сделанные во время последней компоновки. Вы можете использовать эту настройку для: ■ Сохранения оптимизации Компоновщика, но позволяя программе снова выполнить размещение и разводку ■ Переопределения точнее оптимизации Компоновщика (таких как Компоновщик, физический синтез), которая может быть недоступна, когда размещение закреплено ■ Разрешения конфликтов ресурсов между двумя импортированными разделами в процессе восходящего проектирования.

Пустые разделы

Вы можете использовать настройку **Пусто** для пропуска компиляции разделов, которые еще не закончены или отсутствуют в головном проекте. Вы можете воспользоваться ей, если хотите компилировать только определённые разделы в проекте, например, во время оптимизации или, если время компиляции одного раздела слишком большое, то вы можете исключить его из проекта. Эта опция идеально подходит, когда вы хотите оптимизировать размещение критичного ко времени блока, допустим IP ядра, с тем, чтобы зафиксировать его размещение, прежде чем добавлять остальную логику в процессе восходящего проектирования.

Для того, чтобы установить тип списка соединений **Пусто**, в меню **Назначения** выберите **Окно разделов проекта** и дважды кликните на блоке, или правым кликом на блоке, клик на **Свойства раздела проекта**, затем на **Пустой**. Эта настройка определяет компилятору Quartus II использовать пустой, резервирующий место, список соединений для раздела.

Если список соединений определен как **Пусто**, виртуальные выводы автоматически создаются на границах раздела. Этими средствами программа создает промежуточные карты I/O выводов в низкоуровневых блоках для внутренних ячеек взамен выводов в процессе компиляции.

Каждый дочерний раздел пустого раздела в иерархии проекта автоматически устанавливается как пустой, независимо от его настроек.

Если вы планируете полностью использовать преимущества настройки **Пусто**, то очень важно сохранить логику проекта в ветвях иерархического древа проекта, чтобы добиться большей гибкости. Если же вы имеете логику в первом иерархическом уровне и дополнительную логику в

дочерней иерархии, вы не сможете изолировать логику верхнего уровня в пустом разделе, без того, чтобы не обозначить как **Пусто** раздел нижнего уровня.

Вы можете использовать вариант процесса нисходящего проектирования, при котором некоторые разделы установлены как **Пусто**, пока вы независимо друг от друга разрабатываете компоненты проекта, чтобы в дальнейшем скомбинировать их в головном проекте.

Если вы разрабатываете часть проекта в отсутствие информации об остальном проекте, то компилятору невозможно выполнить глобальную оптимизацию размещения. Для уменьшения такого эффекта, следуйте рекомендациям по хорошему созданию разделов, в которых порты входа и выхода раздела регистрируются, когда это возможно, и минимизируются соединения I/O между разделами.

Если вы установили раздел проекта как **Пусто**, файл проекта раздела требуется в Анализе и Синтезе для определения информации интерфейса портов, для того, чтобы корректно подключить к нему другую логику и другие разделы проекта. Если раздел импортирован из другого проекта, файл Экспортированного раздела Quartus II (.**qxp**) уже содержит эту информацию. За более подробной информацией об этих файлах, обратитесь к главе "Файлы экспортированного раздела Quartus II (.**qxp**)" на странице 2-30. Если это не файл .**qxp** или файл описания проекта этого модуля, вам нужно создать **файл-упаковщик** (называемый черным ящиком, болванкой или пустотелым файлом), чтобы определить блок проекта и его входы, выходы и двунаправленные порты. Например, на Verilog вы просто декларируете модуль, на VHDL – декларируете структуру и архитектуру.

База данных проекта содержит список соединений пост-синтез и пост-компоновка предыдущей генерации для неизмененных Пустых разделов, вы можете установить список соединений с **Пусто** на **Пост-Синтез** или **Пост-Компоновка**. Это значит, что программа будет повторно использовать информацию предыдущего списка соединений, а перекомпиляция из исходного кода не требуется.

Где хранятся базы данных списков соединений?

Директория базы данных инкрементной компиляции (\incremental_db) содержит всю информацию о списках соединений предыдущих компиляции. Чтобы избежать ненужной перекомпиляции, эти файлы базы данных не должны меняться или удаляться.

Если вы архивируете или восстанавливаете проект в другом месте, вам необходимо использовать **Файл архива Quartus II (.qar)**. Включите в него базу данных компиляции, чтобы сохранить результаты пост-синтеза или пост-компоновки. Подробнее в главе "Использование инкрементной компиляции с файлами архива Quartus II" на странице 2-61.

Чтобы вручную создать архив проекта, в котором сохраняются результаты компиляции без сохранения базы данных инкрементной компиляции, вам необходимо оставить все исходные файлы и файлы настроек, затем создать и сохранить файл .**qxp** для каждого раздела в проекте, который может быть импортирован в проект для импорта результатов компиляции. Обратитесь к главе "Экспорт низкоуровневых блоков внутри проекта" на странице 2-35 за подробной информацией о том, как создавать .**qxp** файл для раздела внутри вашего проекта.

Какие изменения перезапускают автоматический синтез раздела?

Раздел синтезируется из его исходных файлов, если нет доступного списка соединений пост-синтез после предыдущего синтеза, или если тип списка соединений установлен как **Исходный файл**. В дополнении к этому, некоторые изменения в разделе проекта автоматически перезапускают его синтез, даже когда тип списка соединений установлен как пост-синтез или пост-компоновка. Программа перезапускает синтез раздела в тех случаях, когда необходимо

привести в соответствие файлы описания проекта и файлы пост-размещения и разводки. Если вы хотите запретить автоматический перезапуск синтеза, установите тип списка соединений как **пост-компоновка** (**строгий**). Обратитесь к главе "Форсированное использование списка соединений пост-компоновка после внесения изменений в раздел" на странице 2-25.

В следующем списке приведены изменения, которые перезапускают автоматический синтез раздела, даже когда тип списка соединений установлен как пост-синтез или пост-компоновка:
■ Изменены настройки семейства чипа;
■ Изменены некоторые зависимые исходные файлы проекта. Обратитесь к главе "Определим, когда в разделах перезапускается синтез при изменении исходного кода" на странице 2-24;
■ Изменены границы раздела при добавлении, удалении или изменении портов на границах низкоуровневых разделов (которые определены как блоки нижнего уровня внутри этого раздела);
■ Зависимые исходные файлы компилируются в различных библиотеках (имеют разный аргумент –library);
■ Зависимые исходные файлы были добавлены или удалены, т.е. раздел зависит от набора исходных файлов;
■ Корневой блок проекта связан с разными элементами. На VHDL, каждый блок должен быть связан с определённым модулем и архитектурой. Если меняется назначение блока или архитектура, это перезапускает автоматический синтез;
■ Раздел имеет различные параметры в своей корневой или внешней иерархии AHDL (AHDL автоматически наследует параметры родительской иерархии). Это возникает, если вы напрямую меняете параметры иерархии, или если вы косвенно изменяете параметры родительской иерархии проекта.

Программа повторно использует результаты пост-синтеза, но перекомпоновывает проект, если вы изменяете настройки чипа внутри одного семейства чипов. Программа повторно использует список соединений пост-компоновка только, если вы изменяете быстродействие чипа.

Назначения для Синтеза и Компоновки, называемые также настройками оптимизации, временными настройками или настройками локализации Компоновщика, включающими в себя назначения выводов, не перезапускают автоматически компиляцию в процессе инкрементной компиляции. Подробнее о том, как вы можете влиять на размещение с помощью регионов LogicLock, в главе "Как изменения в LogicLock перезапускают компоновку?" на странице 2-28. Для перекомпиляции разделов с новыми назначениями, измените назначения типа списка соединений на одно из следующих:
■ Исходный файл – для рекомпиляции со всеми новыми настройками;
■ Пост-синтез – для рекомпиляции, используя существующие результаты синтеза, но новые настройки Компоновщика;
■ Пост-компоновка с уровнем сохранения разводки установленным на Размещение, для перезапуска трассировки, используя существующие результаты размещения, за исключением новых настроек трассировки (таких как, настройки цепей задержки).

Определим, когда в разделах перезапускается синтез при изменении исходного кода

Программа Quartus II использует алгоритм внешней контрольной суммы для определения, изменилось ли содержимое исходного файла. Исходные файлы – это все файлы, использованные для создания проекта: VHDL, Verilog, AHDL, .bdf, EDIF, VQM, файлы инициализации памяти, а также .qxp файлы экспортированных разделов. Изменения в других файлах, таких как файлы векторных диаграмм симуляции, не перезапускают компиляцию. Если файлы проекта в разделе зависят от других файлов, то изменение одного файла перезапускает автоматическую компиляцию других файлов. **Таблица Зависимые Файлы Раздела** в отчете **Анализа и Синтеза** показывает список файлов проекта, имеющих вклад в каждом разделе проекта. Вы можете использовать эту

таблицу, чтобы определить, какие разделы будут перекомпилированы при изменении определенных файлов.

Например, если в проекте есть файлы: A.v, являющийся содержимым блока A, B.v, являющийся содержимым блока B, и C.v, являющийся содержимым блока C - то в таблице **Зависимые Файлы Раздела** будет выведено: разделу A – принадлежит файл A.v, разделу B – принадлежит файл B.v, и разделу C – принадлежит файл C.v. Все зависимости могут быть переходными: если A.v зависит от B.v, и B.v зависит от C.v, то блок из файла A.v зависит и от B.v, и от C.v. В этом случае, файлы B.v, C.v будут в отчетной таблице, как файлы зависимости для раздела, содержащего блок A.

Если вы определяете параметры для модуля верхнего уровня, программа Quartus II проверяет значения этих параметров, когда определяет, каким разделам требуется повторный синтез. Если вы меняете параметр в модуле верхнего уровня, который влияет на модуль нижнего уровня, то модуль нижнего уровня будет повторно синтезирован. Параметрическая зависимость получается отдельно из зависимостей исходных файлов, поэтому определения параметров не показаны в списке Зависимые Файлы Раздела.

Если проект содержит общие файлы, например, includes.v, которые ссылаются на каждый блок по команде 'include includes.v', то все разделы зависят от этого файла. Изменение в файле includes.v перезапускает компиляцию проекта. Операторы VHDL используют work.all, типичным результатом этого является ненужная перекомпиляция, потому что этим делаются видимыми все элементы рабочей библиотеки для текущего элемента, в результате текущий элемент зависит от всех остальных элементов в проекте.

Чтобы избежать проблем такого типа, используйте только те общие файлы для всех элементов, называемые общими файлами включения, которые несут информацию, являющуюся действительно общей для всех элементов. Уберите оператор work.all из вашего VHDL файла и замените его включением только необходимых элементов для каждого модуля.

Форсированное использование списка соединений пост-компоновка после внесения изменений в раздел

Форсированное использование списка соединений пост-компоновка, когда изменено содержимое исходного файла, рекомендовано только для продвинутых пользователей, которые понимают, когда раздел должен перекомпилироваться. Вы можете использовать это назначение, например, если вы делаете изменения в исходном коде, но не хотите перекомпилировать раздел, пока не завершите отладку других разделов. Для форсирования Компоновщику использовать последний сгенерированный список соединений, пока вы изменяете исходный файл, необходимо выполнить установку типа списка соединений **Пост-Компоновка (строгий)**.

Неправильное использование типа списка соединений **Пост-Компоновка (строгий)** дает, в результате, функционально некорректный список соединений при генерации из измененного исходного файла. Следите за предупреждениями, когда добавляете эту установку.