Импорт SDC ограничений из низкоуровневого проекта

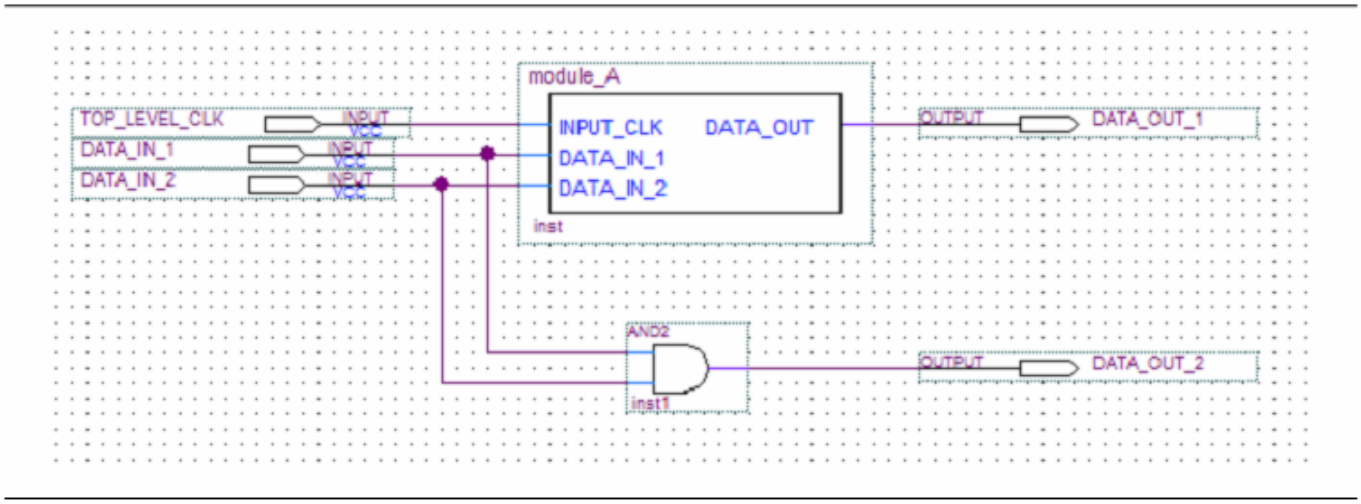
Скрипты раздела восходящего проектирования, описанные в предыдущих главах, автоматизируют процесс переноса информации и назначений из головного проекта в проекты нижнего уровня, так что низкоуровневые разработчики имеют согласованное представление об ограничениях, которые применяются к каждому проекту. Разработчики разделов нижнего уровня могут доносить новые ограничения до руководителя проекта, так что эти ограничения содержатся в заключительной редакции проекта. Команда **Импорт** может быть использована для импорта назначений из раздела проекта нижнего уровня в головной проект; однако, автоматический импорт не содержит формат SDC ограничений для временного анализатора TimeQuest.

Сохранение дополнительных временных ограничений в головном проекте должно управляться осторожно. В этой главе содержатся рекомендации для управления временными ограничениями в процессе восходящего проектирования.

Чтобы добиться отсутствия конфликтов между ограничениями руководителя проекта в головном модуле и ограничениями, добавленными разработчиком нижнего уровня, используйте два файла ограничения проекта Synopsys **(.sdc)** для каждого проекта нижнего уровня: один **.sdc** файл руководителя проекта, содержащий общие ограничения проекта, и один **.sdc** файл, созданный разработчиком низкоуровневого раздела, содержащий специальные ограничения для раздела.

В этой главе используется пример проекта, показанный на рисунке 2-13, чтобы проиллюстрировать эти рекомендации. Головной проект состоит из блока проекта нижнего уровня, называемого *module_A*, который является разделом проекта и разрабатывается другим разработчиком в отдельном проекте Quartus II.

Figure 2–13. Example Design to Illustrate SDC Constraints



В этом головном проекте, имеется одна настройка тактов, называемая *clk*, и ассоциированная с входом FPGA, называемым *top_level_clk*. Файл ограничений **.sdc** имеет следующее ограничение для этого такта:

```
create_clock -name {clk} -period 3.000 -waveform { 0.000 1.500 }
[get_ports {TOP_LEVEL_CLK}]
```

Создание .sdc файла с общими ограничениями для проекта

Файл .**sdc** с общими ограничениями для низкоуровневого проекта должен содержать все ограничения, которые не локализуются полностью в разделе нижнего уровня. Этот файл должен создаваться руководителем головного проекта. Руководитель проекта должен добиться, чтобы эти временные ограничения передавались индивидуальным владельцам разделов, и чтобы они были синтаксически корректны для всех проектов нижнего уровня. Это особенно перспективно, когда проект находится в текущих или иерархических изменениях. Руководитель проекта может использовать инструмент **Генерировать скрипты раздела в восходящем проектировании**, для автоматической генерации этих ограничений, как было описано в предыдущей главе.

Файл .**sdc** с общими ограничениями используется в проекте нижнего уровня, но не экспортируется обратно руководителю головного проекта. Разработчик проекта нижнего уровня не должен модифицировать этот файл. Если изменения требуются, то нужно связаться с руководителем проекта, чтобы он смог обновить SDC ограничения и снабдить новыми файлами всех разработчиков проектов нижнего уровня, если потребуется.

Файл .**sdc** должен содержать создание тактов и ограничения тактов для всех тактов, используемых в более чем одном проекте нижнего уровня. Это особенно важно, когда имеешь дело с комплексными тактовыми структурами, такими как:

■ Каскадные тактовые мультиплексоры;
■ Каскадные PLL;
■ Множественные зависимые такты от одного тактового вывода;
■ Избыточные тактовые структуры, требуемые для безопасных приложений;
■ Виртуальные такты и генерируемые такты, которые используются для согласования с источником синхронного интерфейса;
■ Неопределённость тактов.

Дополнительно, файл **.sdc** с общими ограничениями должен содержать все общие для проекта назначения временных исключений, таких как:

■ Назначения мультицикла, set_multicycle_path;
■ Назначения ложного пути, set_false_path;
■ Назначения максимума задержки, set_max_delay;
■ Назначения минимума задержки, set_min_delay.

Файл .**sdc** с общими ограничениями проекта должен содержать ограничения set_input_delay или set_output_delay для портов проекта нижнего уровня, поскольку это описывает внешние задержки к выбранному разделу. Если разработчик проекта нижнего уровня хочет установить эти ограничения внутри проекта нижнего уровня, то команда должна проконтролировать, чтобы имена I/O портов были идентичными в обоих проектах, так чтобы успешно импортировать назначения без каких-либо изменений.

Аналогично ограничение для пути, расположенного между границ разделов, должно содержаться в общем файле .**sdc** проекта, поскольку оно не локализуются полностью в одном проекте нижнего уровня.

Пример первого шага: Руководитель проекта создает файл .sdc с общими ограничениями проекта для проекта нижнего уровня

Вход чипа *top_level_clk* на рисунке 2-13 является входным портом *input_clk* для *module_A*. Чтобы сделать необходимые тактовые ограничения и корректно их сохранить для проекта нижнего уровня, руководитель проекта создаёт **.sdc** файл с общими ограничениями проекта для *module_A*, который содержит следующую команду:

```
create_clock -name {clk} -period 3.000 -waveform { 0.000 1.500 }
[get_ports {INPUT_CLK}]
```

Разработчик *module_A* включает этот **.sdc** в свой проект нижнего уровня.

Создание .sdc файла со специальными ограничениями для проекта

Файл .**sdc** со специальными ограничениями проекта должен содержать все ограничения, влияющие только на проект нижнего уровня. Например, ограничения set_false_path или set_multicycle_path для пути, полностью находящегося внутри низкоуровневого раздела, должны быть в специальном .**sdc** файле для раздела. Эти ограничения требуются для корректной компиляции раздела, но не требуются для описания других низкоуровневых проектов.

Файл .**sdc** со специальными ограничениями раздела должен разрабатываться индивидуально разработчиком раздела; это зона его ответственности, добавлять ограничения, требуемые для корректной компиляции и анализа его раздела.

Файл .**sdc** со специальными ограничениями раздела используется для низкоуровневого проекта и должен быть экспортирован обратно руководителю головного проекта. Руководитель проекта должен использовать специальные ограничения раздела для корректного ограничения размещения, разводки или обоих их, если логика раздела компонуется в головном модуле, и тем самым добиться аккуратного временного закрытия. Используйте следующие инструкции в файле .**sdc** со специальными ограничениями раздела для упрощения этих шагов экспорта/импорта:

■ Создайте иерархическую переменную для этого раздела (например, *module_A_hierarchy*) и задайте ей пустую строку, поскольку раздел является головным модулем в отдельном проекте. Руководитель проекта модифицирует эту переменную для иерархии головного проекта, уменьшает влияние от передачи ограничений от иерархии проекта нижнего уровня ограничениям, применяемым в иерархии верхнего уровня. Используйте следующую команду Tcl сначала для проверки, существует ли уже переменная, определённая в проекте, так чтобы головной проект не использовал пустой иерархический путь: *if {![info exists module_A_hierarchy]}*.

■ Используйте иерархическую переменную в файле .**sdc** со специальными ограничениями раздела в качестве приставки для назначений в проекте. Например, взамен имени отдельного блока регистра *reg:inst*, используйте *${module_A_hierarchy}reg:inst*. Также используйте иерархическую переменную в качестве приставки для некоторых символов "дикой карты" (например "*").

■ Будьте внимательны с назначениями для I/O портов раздела. Главное, эти назначения должны быть определены в файле .**sdc** с общими ограничениями проекта, потому что интерфейс разделов относится к головному проекту. Если вы хотите установить I/O ограничения внутри проекта нижнего уровня, команда разработчиков должна будет следить за тем, что имя I/O порта одинаково в обоих проектах, так чтобы назначения успешно импортировались без каких-либо изменений.

■ Будьте внимательны с командами *derive_clocks* и *derive_pll_clocks*. В большинстве случаев, в файле .**sdc** с общими ограничениями проекта уже вызываются эти команды. Поскольку эти команды влияют на весь проект, неожиданный импорт их в головной проект ведёт к возникновению проблем.

Если команда разработчиков следует этим рекомендациям, то руководитель проекта должен искусно включать файлы .**sdc** со специальными ограничениями раздела в головной проект, чтобы добавлять .**sdc** ограничениями предоставляемые разработчиками проектов нижнего уровня.

Пример второго шага: Разработчик проекта нижнего уровня создает файл .sdc со специальными ограничениями раздела

Разработчик проекта нижнего уровня компилирует проект вместе с файлом .**sdc** с общими ограничениями проекта и желает добавить некоторые дополнительные ограничения. В этом примере, разработчику нужно определить ложный путь от регистра *reg_in_1* до всех конечных его назначений в этом блоке проекта, используя символ "*" дикой карты. Эти ограничения существуют целиком внутри раздела и должны быть экспортированы в головной проект, чтобы дополнить его файлом .**sdc** со специальными ограничениями раздела. Сначала разработчик определяет переменную *module_A_hierarchy* и использует её при записи ограничений:

```
if {![info exists module_A_hierarchy]} {
    set module_A_hierarchy ""
}
set_false_path -from [get_registers ${module_A_hierarchy}reg_in_1] -to
[get_registers ${module_A_hierarchy}*]
```

Объединение SDC файлов в головном проекте

Когда разработчики проектов нижнего уровня заканчивают свои проекты, они экспортируют результаты для руководителя головного проекта. Руководитель проекта принимает экспортируемые **.qxp** файлы и копирует.**sdc** файлы со специальными ограничениями раздела.

Чтобы установить головному файлу ограничений **.sdc** связь с .**sdc** файлами из проектов нижнего уровня, головной файл **.sdc** должен определить иерархическую переменную, определённую в **.sdc** файлах проектов нижнего уровня. Создать список переменных для каждого раздела нижнего уровня и установить каждому иерархический путь, вплоть до включения модулей разделов нижнего уровня в головной проект, до включения символа окончания иерархии "|".

Чтобы добиться того, что **.sdc** файлы используются в правильном порядке, руководитель проекта может использовать команду Tcl *Исходник* для загрузки каждого **.sdc** файла.

Пример третьего шага: Руководитель проекта выполняет окончательный временной анализ и заканчивает работу

С этими командами, файл головного проекта **.sdc** руководителя проекта выглядит следующим образом:

```
create_clock -name {clk} -period 3.000 -waveform { 0.000 1.500 }
[get_ports {TOP_LEVEL_CLK}]
# Include the lower-level SDC file
set module_A_hierarchy "module_A:inst|"  # Note the final '|' character
source <partition-specific constraint file such as
..\module_A\module_A_constraints>.sdc
```

Когда руководитель проекта выполняет временной анализ в головном проекте, назначение ложного пути из низкоуровневого проекта *module_A* удаляется следующим образом:

```
set_false_path -from module_A:inst|reg_in_1 -to module_A:inst|*
```

Добавление иерархического пути в виде приставки для SDC команды делает ограничение легальным в головном проекте, оно гарантирует, что дикие карты не будут влиять на узлы снаружи разделов, которым они были назначены внутри.

Следуя рекомендациям в этой главе, можно эффективно управлять распространением ограничений между разделами.