Поддержка скриптов

Вы сможете запускать процедуры и делать настройки, описанные в этой главе, в Tcl скриптах. Вы сможете также запускать некоторые процедуры из командной строки. Детальная информация об опциях команд скрипта, описана в Командной строке Quartus II и браузере Помощи Tcl API. Для запуска браузера Помощи, наберите следующую команду в командной строке:

```
quartus_sh --qhelp ↵
```

Ссылка на руководство скриптирования в Quartus II содержит эту же информацию в виде PDF.

За большей информацией о Tcl скриптировании обратитесь к разделу "Tcl Скриптирование" в томе 2 настольной книги Quartus II. Обратитесь к ссылке на руководство по файлу настроек Quartus II за информацией обо всех настройках и ограничениях в программе Quartus II. За большей информацией о скриптах командной строки, обратитесь к разделу "Скрипты командной строки" в томе 2 настольной книги Quartus II.

Подготовка проекта для инкрементной компиляции

Для того чтобы установить или модифицировать текущий режим инкрементной компиляции, используйте следующую команду:

```
set_global_assignment -name INCREMENTAL_COMPILATION <value> ↵
```

Настройка *<value>* инкрементной компиляции может иметь одно из следующих значений:

■ FULL_INCREMENTAL_COMPILATION – полная инкрементная компиляция (по умолчанию)

■ OFF – инкрементная компиляция не выполняется

Создание разделов проекта

Для создания раздела, используйте следующую команду:

```
set_instance_assignment -name PARTITION_HIERARCHY \
<file name> -to <destination> -section_id <partition name>
```

Под *<destination>* понимается краткий иерархический путь. Краткий иерархический путь – это полный иерархический путь без заголовка головного модуля (включая кавычки), например:

```
"ram:ram_unit|altsyncram:altsyncram_component"
```

Для головного проекта, вы используете символ черты (|), чтобы отметить блок верхнего уровня.

За большей информацией о конвертировании иерархических имен, обратитесь в "Конвертирование имен узлов в интегрированном синтезе Quartus II" в главе " Интегрированный синтез Quartus II" тома 1 настольной книги Quartus II.

<partition name> - это выбранное пользователем имя раздела, которое может быть уникальным и быть не более 1024 символов. Имя должно быть только цифро-символьным, с чертой (|), двоеточием (:), подчеркиванием (_). Altera рекомендует помещать имя в кавычки ("").

<file name> - имя генерированных файлов внутренних списков соединений, получаемых во время инкрементной компиляции. Списки соединений называются автоматически в программе Quartus II по имени блока, если вы пользуйтесь пользовательским интерфейсом. Если вы используете Tcl для создания своих разделов, вы можете назначить собственное имя файла, которое будет уникально для всех разделов. Для головного раздела, собственное имя файла будет проигнорировано; вы можете использовать пустое значение. Чтобы гарантировать, что сохраненное имя файла и его основа будут независимы, имя файла должно быть уникально, вне зависимости от вариантов. Например, если вы используете имя раздела my_file, то никакой другой раздел не должен использовать имя файла MY_FILE. Для простоты, Altera рекомендует вам связывать каждое имя файла с соответствующим именем блока раздела.

Программа помещает все списки соединений в папку базы данных компиляции **\incremental_db**.

Настройка свойств разделов проекта

После того, как раздел проекта был создан, задайте его типу списка соединений следующую команду:

```
set_global_assignment -name PARTITION_NETLIST_TYPE <value> \
-section_id <partition name>
```

Типу списка соединений *<value>* установите одно из следующих значений:

- SOURCE – исходный файл
- POST_SYNTH - пост-синез
- POST_FIT – пост-компоновка
- STRICT_POST_FIT - пост-компоновка (строго)
- IMPORTED – импортированный
- IMPORT_BASED_POST_FIT - пост-компоновка (основано на импорте)
- EMPTY - пусто

Установите уровень сохранения компоновки для списка соединений **пост-компоновка** или **импортированный**, используя следующую команду:

```
set_global_assignment -name PARTITION_FITTER_PRESERVATION_LEVEL \
<value> -section_id <partition name>
```

Уровень сохранения компоновки *<value>* может быть установлен в одно из следующих значений:

- NETLIST_ONLY – только список соединений
- PLACEMENT – размещение
- PLACEMENT_AND_ROUTING – размещение и разводка
- PLACEMENT_AND_ROUTING_AND_TILE – размещение и разводка, также элемент настройки энергии: высокоскоростной или низко потребляющий

Детально об этих свойствах, обратитесь к главе "Установка типа списка соединений для раздела проекта" на странице 2-19.

Создание назначений локализации архитектуры – исключение или фильтрация определенные элементов чипа (таких как блоки памяти и DSP)

Фильтрация ресурсов использует дополнительный Tcl аргумент *-exclude_resources* в функции *set_logiclock_contents* набора *incremental_compilation* Tcl. Если это было не оговорено, то фильтры ресурсов не будут созданы.

Аргумент устанавливается на входе списка ресурсов, которые будут исключены. Список – это строка, разделенная двоеточиями, с ключевыми словами из таблицы 2-6:

Таблица 2-6. Ключевые слова для исключаемых ресурсов

Ключевое слово	**Ресурс**
REGISTER	Любые регистры в логических ячейках
COMBINATIONAL	Любые комбинационные элементы в логических ячейках
SMALL_MEM	Небольшие блоки памяти TriMatrix (M512 или MLAB)
MEDIUM_MEM	Средние блоки памяти TriMatrix (M4K or M9K)
LARGE_MEM	Большие блоки памяти TriMatrix (M-RAM or M144K)
DSP	Любые блоки DSP
VIRTUAL_PIN	Любые виртуальные выводы

Например, следующая команда назначит всему *alu:alu_unit* региона ALU исключить все DSP и M512 блоки:

```
set_logiclock_contents -region ALU -to alu:alu_unit -exceptions \
"DSP:SMALL_MEM"
```

В файле **.qsf**, фильтрация ресурсов использует дополнительный элемент назначений LogicLock, называемый LL_MEMBER_RESOURCE_EXCLUDE. Например, следующая строка в файле **.qsf** использует определенный фильтр ресурсов для блока *alu:alu_unit*, назначенного региону ALU. Значение назначений использует тот же формат, как и строка перечисления ресурсов, оставшаяся после предыдущей Tcl команды.

```
set_instance_assignment -name LL_MEMBER_RESOURCE_EXCLUDE \
"DSP:SMALL_MEM" -to "alu:alu_unit" -section_id ALU
```

Генерация скриптов раздела восходящего проектирования

Для генерации скриптов, наберите следующую Tcl команду в Tcl строке:

```
generate_bottom_up_scripts <options> ↵
```

Команда является частью пакета *database_manager*, который должен быть загружен, используя следующую команду, прежде чем команда будет использоваться:

```
load_package database_manager
```

Вы должны сначала открыть проект, прежде чем генерировать скрипты.
Настройки Tcl также доступны в GUI. Точный формат каждой опции определен в таблице 2-7.

Таблица 2-7. Настройки для генерации скриптов раздела в восходящем проектировании с помощью Tcl команд (часть 1 из 2)

Настройка	**По умолчанию**
-include_makefiles <on|off>	Вкл.
-include_project_creation <on|off>	Вкл.

Таблица 2-7. Настройки для генерации скриптов раздела в восходящем проектировании с помощью Tcl команд (часть 2 из 2)	
Настройка	**По умолчанию**
-include_virtual_pins <on|off>	Вкл.
-include_virtual_pin_timing <on|off>	Вкл.
-include_virtual_pin_locations <on|off>	Вкл.
-include_logiclock_regions <on|off>	Вкл.
-include_global_signal_promotion <on|off>	Выкл.
-include_pin_locations <on|off>	Вкл.
-include_timing_assignments <on|off>	Вкл.
-include_design_partitions <on|off>	Вкл.
-remove_existing_regions <on|off>	Вкл.
-disable_auto_global_promotion <on|off>	Выкл.
-bottom_up_scripts_output_directory <output directory>	Текущая директория
-virtual_pin_delay <delay in ns>	Нет значения по умолчанию

Следующий пример показывает, как использовать Tcl команду:

```
load_package database_manager
set project test_proj
project_open $project
generate_bottom_up_scripts -bottom_up_scripts_output_directory test \
    -include_virtual_pin_timing on -virtual_pin_delay 1.2
project_close
```

Поддержка командной строки

Для генерации скриптов из командной строки, наберите следующую команду:

```
quartus_cdb <project name> --generate_bottom_up_scripts=on <options> ↵
```

Ещё раз, карта настроек такая же, как и в GUI. Для добавления опции, дополните вызов командной строки *"--<option_name>=<val>"*.

Опции командной строки такие же, какие доступы в GUI. Они перечислены в таблице 2-8.

Таблица 2-8. Настройки для генерации скриптов раздела в восходящем проектировании (часть 1 из 2)	
Настройка	**По умолчанию**
--include_makefiles_with_bottom_up_scripts=<on|off>	Вкл.
--include_project_creation_in_bottom_up_scripts=<on|off>	Вкл.
--include_virtual_pins_in_bottom_up_scripts=<on|off>	Вкл.
--include_virtual_pin_timing_in_bottom_up_scripts=<on|off>	Вкл.
--bottom_up_scripts_virtual_pin_delay=<delay in ns>	(1)
--include_virtual_pin_locations_in_bottom_up_scripts=<on|off>	Вкл.
--include_logiclock_regions_in_bottom_up_scripts=<on|off>	Вкл.
--include_all_logiclock_regions_in_bottom_up_scripts=<on|off>	Вкл.

Таблица 2-8. Настройки для генерации скриптов раздела в восходящем проектировании (часть 2 из 2)	
Настройка	**По умолчанию**
--include_global_signal_promotion_in_bottom_up_scripts=<on|off>	Выкл.
--include_pin_locations_in_bottom_up_scripts=<on|off>	Вкл.
--include_timing_assignments_in_bottom_up_scripts=<on|off>	Вкл.
--include_design_partitions_in_bottom_up_scripts=<on|off>	Вкл.
--remove_existing_regions_in_bottom_up_scripts=<on|off>	Вкл.
--disable_auto_global_promotion_in_bottom_up_scripts=<on|off>	Выкл.
--bottom_up_scripts_output_directory=<output directory>	Текущая директория

Примечания к таблице 2-8. (1) Нет значений по умолчанию. Вы должны определить эту опцию, если используете временные характеристики для виртуальных выводов.

Экспорт раздела для использования в головном проекте

Используйте *quartus_cdb* для выполнения экспорта файла для процесса восходящей инкрементной компиляции с помощью следующей команды:

```
quartus_cdb --INCREMENTAL_COMPILATION_EXPORT=<file> \
[--incremental_compilation_export_netlist_type=<POST_SYNTH|POST_FIT>]
\
[--incremental_compilation_export_partition_name=<partition name>] \
[--incremental_compilation_export_routing=<on|off>]
```

Аргумент *<file>* - это путь до экспортируемого файла. *<partition name>* - имя раздела, но не иерархический путь. Если вы не определите настройки, будут использоваться к выполнению настройки из **.qsf** файла, или использоваться значения по умолчанию. Раздел по умолчанию является головной раздел проекта, список соединений по умолчанию – пост-компоновка, а разводка по умолчанию включена (для всех чипов, поддерживающих экспорт разводки).

Команда INCREMENTAL_COMPILATION_EXPORT_NETLIST_TYPE читает назначения для того, чтобы определить, какой тип списка соединений будет экспортирован; по умолчанию – это пост-компоновка.

Вы можете также использовать процесс INCREMENTAL_COMPILATION_EXPORT в Tcl команде *execute_flow*, содержащейся в пакете *процессы* Tcl.

Используйте следующие команды для экспорта **.qxp** файла исходного раздела, выберите тип списка соединений и определите, будете ли экспортировать разводку:

```
load_package flow
set_global_assignment -name INCREMENTAL_COMPILATION_EXPORT_FILE \
<filename>
set_global_assignment -name
INCREMENTAL_COMPILATION_EXPORT_NETLIST_TYPE \
<POST_FIT|POST_SYNTH>
set_global_assignment -name \
INCREMENTAL_COMPILATION_EXPORT_PARTITION_NAME <partition name>
set_global_assignment -name INCREMENTAL_COMPILATION_EXPORT_ROUTING \
<on|off>
execute_flow -INCREMENTAL_COMPILATION_EXPORT
```

Раздел по умолчанию является головной раздел проекта, список соединений по умолчанию – пост-компоновка, а разводка по умолчанию включена (для всех чипов, поддерживающих экспорт разводки).

Чтобы включить эту настройку навсегда при выполнении экспорта в следующей компиляции, используйте следующую Tcl команду:

```
set_global_assignment -name AUTO_EXPORT_INCREMENTAL_COMPILATION ON
```

Импорт раздела нижнего уровня в головной проект

Используйте команду *quartus_cdb* для импорта раздела нижнего уровня с помощью следующей команды:

```
quartus_cdb -- INCREMENTAL_COMPILATION_IMPORT ↵
```

Вы можете также использовать процесс, называемый INCREMENTAL_COMPILATION_IMPORT в Tcl команде из набора *процессы* Tcl команд. Следующий пример скрипта показывает, как импортировать раздел с помощью Tcl скрипта:

```
load_package flow
# commands to set the import-related assignments for each partition
execute_flow --INCREMENTAL_COMPILATION_IMPORT
```

Определите расположение импортируемого файла с помощью назначения PARTITION_IMPORT_FILE. Обратите внимание, что определенный в этом назначении файл будет читаться только во время импорта. Например, проект полностью зависит от нескольких файлов из низоуровневых проектов после импорта. В командной строке и процессе Tcl, те разделы, которые имеют не пустое назначение, будут импортированы. Следующие назначения определяют, каким образом будут импортированы разделы:

```
PARTITION_IMPORT_PROMOTE_ASSIGNMENTS = <on|off>
PARTITION_IMPORT_NEW_ASSIGNMENTS = <on|off>
PARTITION_IMPORT_EXISTING_ASSIGNMENTS = \
replace_conflicting | skip_conflicting
PARTITION_IMPORT_EXISTING_LOGICLOCK_REGIONS = \
replace_conflicting | update_conflicting | skip_conflicting
```

Создатель файлов

Для примера, каким образом использовать инкрементную компиляцию с помощью создателя файлов в качестве части процесса восходящей инкрементной компиляции, обратитесь к файлу **read_me.txt**, который сопровождает пример *incr_comp*, расположенный в директории **qdesigns/incr_comp_makefile**. Когда используется процесс восходящей инкрементной компиляции, средство **генерации скриптов раздела восходящего проектирования** может писать создатель файлов, которые автоматически экспортирует низкоуровневые разделы проекта и импортирует их в головной проект всякий раз, когда изменяются файлы проекта.

Рекомендуемый процесс разработки и примеры приложений компиляций – скриптирование и операции с командной строкой

В этой главе приведены примеры, которые охватывают большинство тем, описанных в основном разделе главы.

Скрипт, показанный в примере 2-1 открывает проект, названный *AB_project*, устанавливает два раздела, элементы *A* и *B*, и выполняет начальную полную компиляцию.

Example 2–1. AB_project

```
set project AB_project

package require ::quartus::flow
project_open $project

# Ensure that incremental compilation is turned on
set_global_assignment -name INCREMENTAL_COMPILATION \
FULL_INCREMENTAL_COMPILATION

# Set up the partitions
set_instance_assignment -name PARTITION_HIERARCHY \
 incremental_db/A_inst -to A -section_id "Partition_A"
set_instance_assignment -name PARTITION_HIERARCHY \
 incremental_db/B_inst -to B -section_id "Partition_B"

# Set the netlist types to post-fit for subsequent
# compilations (all partitions are compiled during the
# initial compilation since there are no post-fit
# netlists)
set_global_assignment -name PARTITION_NETLIST_TYPE \
 POST_FIT -section_id "Partition_A"
set_global_assignment -name PARTITION_NETLIST_TYPE \
 POST_FIT -section_id "Partition_B"

# Run initial compilation:
export_assignments
execute_flow -full_compile

project_close
```

Уменьшение времени компиляции, когда изменяется исходный файл для одного раздела – пример с командной строкой

Подоплёка примера. Вам нужно запустить процесс начальной компиляции, показанный в примере скрипта в предыдущем параграфе. Вы хотите модифицировать HDL исходный файл для раздела *A* и хотите перекомпилировать его.

Запустите команду стандартного процесса компиляции в вашем Tcl скрипте:

```
execute_flow -full_compile
```

Или наберите следующую команду в системной командной строке:

```
quartus_sh --flow compile AB_project↵
```

Если исходный файл для раздела *B* не зависит от *A*, перекомпилируется только *A*.

Размещение *B* и её временные характеристики сохраняются, это также сберегает значительное время компиляции.

Оптимизация размещения для критичных ко времени разделов

Подоплёка примера. Вам нужно запустить начальную компиляцию, показанную в примере скрипта в главе "Рекомендуемый процесс разработки и примеры приложений компиляций – скриптирование и операции с командной строкой" на странице 2-74. Вам хотелось бы применить оптимизацию Компоновщика, называемую физическим синтезом, только для раздела *A*. При этом не было сделано изменений в обоих HDL файлах.

Чтобы добиться сохранения результата предыдущей компиляции для раздела *B*, и чтобы оптимизация Компоновщика применялась только к списку соединений пост-синтеза раздела *A*, установите тип списка соединений для *B* как **пост-компоновка** (который уже готов после начальной компиляции, но будет безопасно повторен снова), а тип списка соединений *A* – **пост-синтез**, как показано в примере 2-2.

Example 2–2. AB_project (2)

```
set project AB_project

package require ::quartus::flow
project_open $project

# Turn on Physical Synthesis Optimization
set_global_assignment -name \
PHYSICAL_SYNTHESIS_REGISTER_RETIMING ON

# For A, set the netlist type to post-synthesis
set_global_assignment -name PARTITION_NETLIST_TYPE POST_SYNTH \
-section_id "Partition_A"

# For B, set the netlist type to post-fit
set_global_assignment -name PARTITION_NETLIST_TYPE POST_FIT \
-section_id "Partition_B"

# Run incremental compilation:
export_assignments
execute_flow -full_compile

project_close
```