Спецификация тактов

Спецификация для всех тактов и ассоциированных с тактами характеристик в вашем проекте просто необходима для точных результатов статического временного анализа. Временной Анализатор Quartus II TimeQuest поддерживает большинство команд SDC, которые регулируют различные схемы тактирования и прочие тактовые характеристики.
Эта глава описывает API .sdc файла, доступное для создания определённых тактовых характеристик.

Такты

Используйте команду create_clock для создания тактов на любых регистрах, портах или выводах. Вы сможете создать каждый такт с уникальными характеристиками. В примере 7-2 показаны команда create_clock и опции.

Example 7–2. create_clock Command

```
create_clock
-period <period value>
[-name <clock name>]
[-waveform <edge list>]
[-add]
<targets>
```

В таблице 7-6 описаны опции для команды create_clock.

Таблица 7-6. Опции команды create_clock

Опции	Описание
-period <period value>	Определяет период тактов. Вы сможете также определять период в единицах измерения частоты, например -period <num>MHz. (1)
-name <clock name>	Имя определенного такта; например, sysclock. Если вы не определили имя такта, его имя будет иметь названия узла, назначенного для этого такта.
-waveform <edge list>	Определяет нарастающие и спадающие фронты тактовых импульсов. Список фронтов чередует нарастающие и спадающие фронты. Например, период 10 нс сначала имеет нарастающий фронт с 0 нс, а затем спадающий фронт на 5 нс; он записывается как: -waveform {0 5}. Все различия должны быть описаны внутри одного периода, а нарастающий фронт должен идти прежде спадающего. По умолчанию, список фронтов должен быть {0 <period>/2}, или 50% рабочий цикл.
-add	Позволяет вам определить более одного такта для одного порта или вывода.
<targets>	Определяет порты или выводы, которым добавляются назначения. Если исходный объект не определен, такт становится виртуальным тактом. Обратитесь к "Виртуальные такты" на странице 7-28 за дополнительной информацией.

Замечания к таблице 7-6: (1) Единица измерения времени по умолчанию во временном анализаторе Quartus II TimeQuest – наносекунды (нс).

В примере 7-3 показано, как создать такт 10 нс с 50% рабочим циклом, который начинается с 0 нс, и добавляется к порту clk.

Example 7–3. 100MHz Clock Creation

```
create_clock -period 10 -waveform { 0 5 } clk
```

В примере 7-4 показано, как создать такт 10 нс с 50% рабочим циклом, который сдвинут по фазе на 90 градусов, и добавляется к порту clk_sys.

Example 7–4. 100MHz Shifted by 90 Degrees Clock Creation

```
create_clock -period 10 -waveform { 2.5 7.5 } clk_sys
```

Такты, определенные командой create_clock имеют нулевую исходную задержку. Временной анализатор Quartus II TimeQuest автоматически вычисляет сетевую задержку тактов для не виртуальных тактов.

Сгенерированные такты

Временной анализатор Quartus II TimeQuest рассматривает делители тактов, пульсирующие такты, или цепи, которые модифицировали или изменили характеристики поступающих или основных тактов, в качестве сгенерированных тактов. Вам нужно определить выход таких цепей - как сгенерированные такты.
Это определение позволяет временному анализатору Quartus II TimeQuest анализировать эти такты и вычислить соответствующую задержку для каждых тактовых сетей.

Используйте команду create_generated_clock для создания сгенерированных тактов. В примере 7-5 показаны команда create_generated_clock и доступные опции.

Example 7–5. create_generated_clock Command

```
create_generated_clock
[-name <clock name>]
-source <master pin>
[-edges <edge list>]
[-edge_shift <shift list>]
[-divide_by <factor>]
[-multiply_by <factor>]
[-duty_cycle <percent>]
[-add]
[-invert]
[-master_clock <clock>]
[-phase <phase>]
[-offset <offset>]
<targets>
```

В таблице 7-7 описаны опции для команды create_generated_clock.

Таблица 7-7. Опции команды create_generated_clock (часть 1 из 2)

Опции	**Описание**
-name <clock name>	Имя сгенерированного такта; например, clk_x2. Если вы не определили имя такта, его имя будет иметь названия узла, назначенного для этого такта.
-source <master pin>	<master pin> определяет узел проекта, от которого наследуются настройки тактов.
-edges <edge list> | -edge_shift <shift list>	Опция -edges определяет новые нарастающие и спадающие фронты по отношению к фронтам основного такта. Нарастающие и спадающие фронты основного такта нумеруются 1..<n>, и начинаются с первого нарастающего фронта; например, фронт 1. Первый фронт спада после этого будет иметь номер 2, следующий нарастающий – 3, и т.д. Список фронтов <edge list> должен быть в порядке возрастания. Один и тот же фронт может быть использован в двух блоках для индикации тактовых импульсов, определенных в основном рабочем цикле. Опция -edge_shift определяет величину сдвига для каждого фронта в <edge list>. Опция -invert используется для инвертирования тактов после применения –edges и -edge_shift (1).
-divide_by <factor> | -multiply_by <factor>	Коэффициенты -divide_by или -multiply_by основываются на первом возрастающем фронте тактов, и определяют увеличение или уменьшение временной диаграммы на определенный коэффициент. Например, -divide_by 2 эквивалентно -edges {1 3 5}. Для различных тактов, определяются также рабочие циклы. Временной анализатор Quartus II TimeQuest поддерживает одновременное назначение коэффициентов умножения и деления.

Таблица 7-7. Опции команды create_generated_clock (часть 2 из 2)

-duty_cycle <percent>	Определяет рабочий цикл для генерированного такта. Рабочий цикл добавляется последним.
-add	Позволяет вам определить более одного такта для одного порта или вывода.
-invert	Инверсия добавляется к выходу такта после всех других модификаций, за исключением рабочего цикла.

-master_clock <clock>	Используется для определения такта, если на основном выводе могут быть несколько тактов.
-phase <phase>	Определяет фазу сгенерированного такта.
-offset <offset>	Определяет смещение сгенерированного такта.
<targets>	Определяет порты или выводы, которым добавляются назначения.

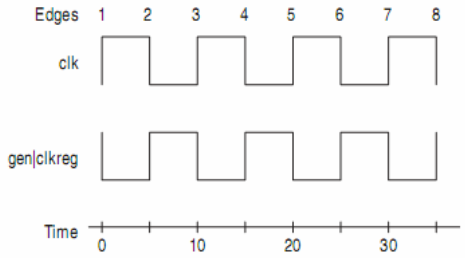
Замечания к таблице 7-7: (1) Временной анализатор Quartus II TimeQuest поддерживает максимум три значения в списке фронтов.

Исходные задержки основываются на задержках тактовых сетей основного такта (не обязательно основного вывода). Вы можете использовать команду set_clock_latency –source для отмены исходной задержки.

На рисунке 7-20 показано, как создать инвертированный генерированный такт, основанный на такте 10 нс.

Figure 7–20. Generating an Inverted Clock

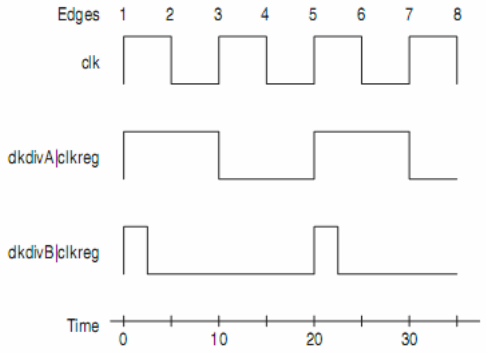
```
create_clock -period 10 [get_ports clk]
create_generated_clock -divide_by 1 -invert -source [get_registers clk] \
    [get_registers gen|clkreg]
```



На рисунке 7-21 показано, как модифицировать сгенерированный такт, используя опции -edges и -edge_shift.

Figure 7–21. Edges and Edge Shifting a Generated Clock

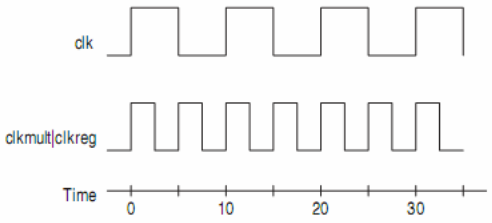
```
create_clock -period 10 -waveform { 0 5} [get_ports clk]
# Creates a divide-by-t clock
create_generated_clock -source [get_ports clk] -edges {1 3 5 } [get_registers \
clkdivA|clkreg]
# Creates a divide-by-2 clock independent of the master clocks' duty cycle (now 50%)
create_generated_clock -source [get_ports clk] -edges {1 1 5} -edge_shift { 0 2.5 0 } \
[get_registers clkdivB|clkreg]
```



На рисунке 7-22 показан эффект от использования опции -multiply_by для сгенерированного такта.

Figure 7–22. Multiplying a Generated Clock

```
create_clock -period 10 -waveform { 0 5 } [get_ports clk]
# Creates a multiply-by-2 clock
create_generated_clock -source [get_ports clk] -multiply_by 2 [get_registers \
clkmult|clkreg]
```



Виртуальные такты

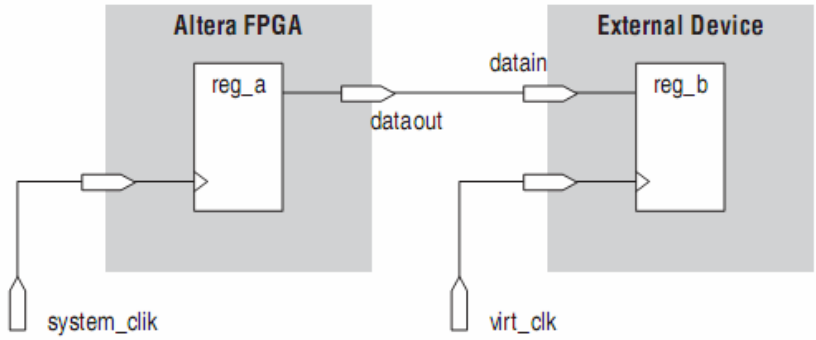
Виртуальные такты – это такты, которые не имеют реального источника в проекте или не имеют чёткого взаимодействия с проектом. Например, если такт идёт только на тактовый порт внешнего чипа и не тактовый порт проекта, а внешний чип затем ведёт на порт проекта, это рассматривается как виртуальный вывод.

Используйте команду create_clock для создания виртуальных тактов, которым не определено значение в исходной опции.

Вы можете использовать виртуальные такты для ограничений set_input_delay и set_output_delay.

На рисунке 7-23 показан пример, в котором виртуальный вывод используется во временном анализаторе Quartus II TimeQuest для должного анализа взаимосвязи между внешним регистром и тем, что в проекте. Поскольку генератор отметил virt_clk, который не взаимодействует с чипом Altera, но является исходным тактов для внешнего регистра, такт virt_clk должен быть декларирован. В примере 7-6 показана команда создания 10 нс виртуального такта, называемого virt_clk, с 50% рабочим циклом, первый нарастающий фронт которого начинается в 0 нс. Виртуальный такт используется в качестве исходного такта для ограничения выходной задержки.

Figure 7–23. Virtual Clock Board Topology



После того, как вы создадите виртуальный такт, вам необходимо будет выполнить анализ от регистра к регистру между регистром в чипе Altera и регистром внешнего устройства.

Example 7–6. Virtual Clock Example 1

```
#create base clock for the design
create_clock -period 5 [get_ports system_clk]
#create the virtual clock for the external register
create_clock -period 10 -name virt_clk -waveform { 0 5 }
#set the output delay referencing the virtual clock
set_output_delay -clock virt_clk -max 1.5 [get_ports dataout]
```

В примере 7-7 показана команда создания 10 нс виртуального вывода с 50% рабочим циклом, фаза которого сдвинута на 90°.

Example 7–7. Virtual Clock Example 2

```
create_clock -name virt_clk -period 10 -waveform { 2.5 7.5 }
```

Мульти-частотные такты

Естественно, что проекты имеют более одного источника тактов, идущего на один тактовый порт проекта. Дополнительные такты могут использоваться в качестве низкопотребляющих тактов, с меньшей частотой, в отличие от основного такта. Чтобы анализировать этот тип проекта, команда create_clock поддерживает опцию –add, которая позволяет вам добавлять более одного такта для тактового узла.

На примере 7-8 показана команда создания 10 нс такта, добавленного тактовому порту clk, и так же добавление дополнительного 15 нс такта тому же тактовому порту. Временной анализатор Quartus II TimeQuest использует оба этих такта, когда выполняет временной анализ.

Example 7–8. Multi-Frequency Example

```
create_clock -period 10 -name clock_primary -waveform { 0 5 } [get_ports
clk]
create_clock -period 15 -name clock_secondary -waveform { 0 7.5 }
[get_ports clk] -add
```

Автоматическое детектирование тактов

Для создания тактов для всех тактовых узлов в вашем проекте автоматически, используйте команду derive_clocks. Эта команда создает такты на портах или регистрах, чтобы обеспечить каждый регистр тактом в проекте.
Пример 7-9 показывает команду derive_clocks и её опции.

Example 7–9. derive_clocks Command

```
derive_clocks
[-period <period value>]
[-waveform <edge list>]
```

В таблице 7-8 перечислены опции команды derive_clocks

Таблица 7-8. Опции команды derive_clocks

Опция	Описание
-period <period value>	Создает тактовый период. Вы можете также определить частоту в качестве -period <num>MHz. (1)
-waveform <edge list>	Создает фронты нарастания и спада тактов. Список фронтов чередуется между фронтами нарастания и спада. Например, период 10 нс сначала имеет нарастающий фронт с 0 нс, а затем спадающий фронт на 5 нс; он записывается как: -waveform {0 5}. Все различия должны быть описаны внутри одного периода, а нарастающий фронт должен идти прежде спадающего. По умолчанию, список фронтов должен быть {0 <period>/2}, или 50% рабочий цикл.

Замечания к таблице 7-9: (1) Единица измерения времени по умолчанию во временном анализаторе Quartus II TimeQuest – наносекунды (нс).

Команда derive_clocks не создает такты для выхода PLL.

Команда derive_clocks эквивалентна использованию команды create_clock для каждого регистра или порта на тактовый вывод регистра.

Использование команды derive_clocks для окончательного временного анализа не рекомендуется. Вам необходимо создать такты для всех источников тактов, используя команды create_clock и create_generated_clock.

Получение тактов PLL

PLL используются для управления и синтеза тактов в чипах Altera. Вы можете изменить такты, сгенерированные на выводах PLL, основываясь на ограничениях проекта. Поскольку такты будут созданы для всех тактовых узлов, все выводы PLL должны иметь ассоциированные такты.

Вы можете вручную создавать такты для каждого выхода PLL с помощью команды create_generated_clock, или вы можете использовать команду derive_pll_clocks, с автоматическим поиском временного списка соединений и созданием сгенерированных тактов для всех выходов PLL согласно настройкам, определенным для каждого выхода PLL. Используйте команду derive_pll_clocks для автоматического создания тактов для каждого выхода PLL. В примере 7-10 показана команда derive_pll_clocks и опции.

Example 7–10. derive_pll_clocks Command

```
derive_pll_clocks
[-create_base_clocks]
[-use_tan_name]
```

В таблице 7-9 перечислены опции команды derive_pll_clocks

Таблица 7-9. Опции команды derive_pll_clocks

Опция	Описание
-use_tan_name	По умолчанию, имя такта – это имя выходного такта. Эта опция использует имя цепи похожее с именами, используемыми Классическим временным анализатором Quartus II.
-create_base_clocks	Создает базовые такты на входном тактовом порте проекта, ведущего к PLL.

Команда derive_pll_clocks вызывает команду create_generated_clock на выходах PLL. Входной тактовый вывод PLL является исходным для команды create_generated_clock. Перед или после того, как команда derive_pll_clocks дала результат, вы можете вручную создать базовый такт для входного тактового порта PLL. Если такты не определены для тактового входа узла PLL, в отчёте о выходах PLL не будет тактов. Взамен того, временной анализатор Quartus II TimeQuest выдаст предупреждение, похожее с примером 7-11, когда обновится временной список соединений.

Example 7–11. Warning Message

```
Warning: The master clock for this clock assignment could not be
derived.
Clock: <name of PLL output clock pin name> was not created.
```

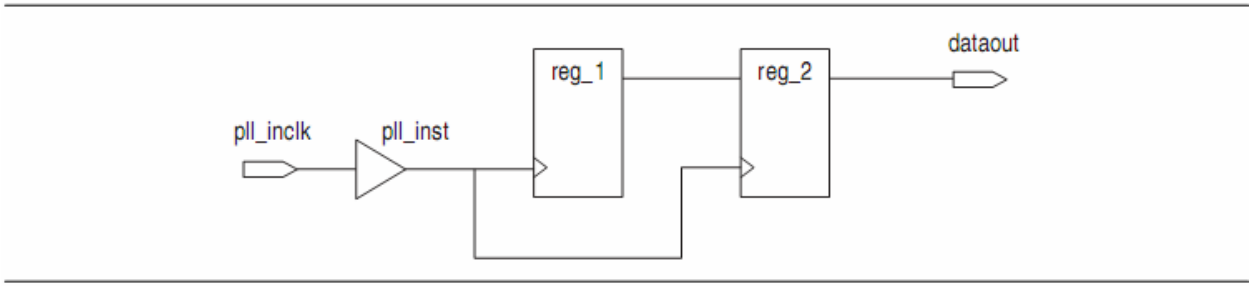
Вы можете использовать опцию -create_base_clocks для автоматического создания входных тактов на входах PLL.

Вы можете включить команду derive_pll_clocks в ваш .sdc файл, с помощью команды derive_pll_clocks возможно детектировать любые изменения в PLL. С помощью команды derive_pll_clocks в вашем .sdc файле, в каждый момент времени файл читается, соответственно генерится команда create_generated_clocks для выходные тактов PLL. Если вы используете расширения команды write_sdc после команды derive_pll_clocks, новый .sdc файл содержит индивидуальные команды create_generated_clocks для выходных тактов PLL и не содержит команду derive_pll_clocks. Некоторые изменения свойств PLL автоматически не отображаются в новом .sdc файле. Вам необходимо вручную обновить команду create_generated_clocks в новом .sdc файле записью команды derive_pll_clocks для отображения изменений в PLL.

Ограничения derive_pll_clocks могут также ограничивать некоторые LVDS передатчики или LVDS приёмники в проекте путём добавления соответствующих мультицикловых ограничений для подсчёта некоторых факторов преобразования последовательного кода в параллельный.

Например, на рисунке 7-24 показан простой PLL проект с путём от регистра к регистру.

Figure 7–24. Simple PLL Design



Используйте команду derive_pll_clocks для автоматического ограничения PLL. Когда эта команда используется для проекта, показанного на рисунке 7-24, генерируются сообщения, показанные в примере 7-12.

Example 7–12. derive_pll_clocks Generated Messages

```
Info:
Info: Deriving PLL Clocks:
Info: create_generated_clock -source
pll_inst|altpll_component|pll|inclk[0] -divide_by 2 -name
pll_inst|altpll_component|pll|clk[0]
pll_inst|altpll_component|pll|clk[0]
Info:
```

Имя узла pll_inst|altpll_component|pll|inclk[0] используется для исходной опции ссылок на входной тактовый вывод PLL. В дополнение, имя выходного такта PLL – это имя выхода такта узла PLL, pll_inst|altpll_component|pll|clk[0].

Если PLL находится в режиме переключения тактов, различные такты будут созданы для выходного такта PLL; один для первого входного такта (например, inclk[0]), и один для второго входного такта (например, inclk[1]). В этом случае, вам необходимо вырезать первый и второй выходные такты, используя команду set_clock_groups с опцией -exclusive.

Прежде чем вы будете генерировать различные отчёты для этого проекта, вы можете создать базовый такт для входного тактового порта PLL. Используйте следующую команду или похожую:

create_clock -period 5 [get_ports pll_inclk]

Вы не можете генерировать базовый такт на входном тактовом выводе PLL: pll_inst|altpll_component|pll|inclk[0]. Такт, созданный на входном тактовом порте, распространяется во всех ветвлениях тактового порта, включая входной тактовый вывод PLL.

Ограничения тактов по умолчанию

Для того, чтобы провести полный анализ тактов, временной анализатор Quartus II TimeQuest, по умолчанию, автоматически создаёт такты для всех детектированных тактовых узлов в вашем проекте, которые не были ограничены, если те не являются базовыми тактовыми ограничениями в проекте. Временной анализатор Quartus II TimeQuest создаёт базовый такт 1 ГГц для неограниченных тактовых узлов, используя следующую команду:

derive_clocks -period 1

Индивидуальные тактовые ограничения (например, create_clock и create_generated_clock) должны быть сделаны для всех тактов в проекте. Этим добиваются досконального и реалистичного анализа временных ограничений проекта. Избегайте использования derive_clocks для финального временного анализа.
Тактовые ограничения по умолчанию добавляются только тогда, когда временной анализатор Quartus II TimeQuest детектирует такие синхронные элементы, с которыми не были ассоциированы такты. Например, если в проекте имеется два такта, а только один такт имеет ограничения, то ограничения по умолчанию не устанавливаются. Иначе, если оба такта не ограничены, то добавляются ограничения по умолчанию.

Тактовые группы

Многие такты могут быть в проекте; тем не менее, не все такты взаимодействуют друг с другом, и точное взаимодействие тактов не известно.
Используйте команду set_clock_groups для эксклюзивных или асинхронных тактов. В примере 7-13 показана команда set_clock_groups и её опции.

Example 7–13. set_clock_groups Command

```
set_clock_groups
[-asynchronous | -exclusive]
-group <clock name>
[-group <clock name>]
[-group <clock name>] ...
```

В таблице 7-10 перечислены опции команды set_clock_groups

Таблица 7-10. Опции команды set_clock_groups

Опция	Описание
-asynchronous	Асинхронные такты – это когда два такта не имеют фазовой зависимости и могут быть активными в любое время.
-exclusive	Эксклюзивные такты – это когда только один из двух тактов активен в данный момент. В качестве примера эксклюзивных тактовых групп – это когда два такта поступают на мультиплексор 2 в 1.
-group <clock name>	Определяет верное назначение тактовых имён, которые являются взаимно эксклюзивными. <clock name> используется для определения имён тактов.

Опция -exclusive используется для декларирования, когда два такта являются взаимно эксклюзивными друг другу, и не должны существовать в проекте в одно время. Это относится к различным тактам, которые создаются для одного узла или для тактового мультиплексора. Например, порт должен тактироваться поочередно тактами 25 и 50 МГц. Чтобы ограничить этот порт, два такта должны быть созданы командой create_clock для этого порта, затем, используя команду set_clock_groups –exclusive, декларировать который такт не будет существовать в проекте в это время.

Этим исключается всякое перемещение тактов, которое происходит между тактами 25 и 50 МГц. В примере 7-14 показаны соответствующие ограничения.

Example 7–14. Exclusive Option

```
create_clock -period 40 -name clk_A [get_ports {port_A}]
create_clock -add -period 20 -name clk_B [get_ports {port_A}]
set_clock_groups -exclusive -group {clk_A} -group {clk_B}
```

Группа определена с помощью опции -group. Временной анализатор TimeQuest вырезает временные пути между тактами каждой отдельной группы -group.

Опция -asynchronous используется для групп родственных и неродственных тактов. С помощью опции -asynchronous, такты, содержащиеся в группах, рассматриваются асинхронно друг другу. Всякие такты внутри каждой группы рассматриваются синхронно друг другу.

Например, предположим, вы имеете три такта: clk_A, clk_B, и clk_C. Такты clk_A и clk_B родственные друг другу, а такт clk_C работает полностью асинхронно с clk_A или clk_B. В примере 7-15 сделаны родственные clk_A и clk_B в одной группе, а неродственный clk_C во второй группе.

Example 7–15. Asynchronous Option Example 1

```
set_clock_groups -asynchronous -group {clk_A clk_B} -group {clk_C}
```

В примере 7-16 показан другой метод определения тех же самых ограничений, как и в примере 7-15.

Example 7–16. Asynchronous Option Example 2

```
set_clock_groups -asynchronous -group {clk_C}
```

Этим clk_C сделан неродственным любому другому такту в проекте, потому что clk_C находится только в группе ограничений.

Временной анализатор TimeQuest предполагает, что все такты по умолчанию родственные, если они не ограничены другим способом.

В примере 7-17 показана команда set_clock_groups и эквивалентная команда set_false_path.

Example 7–17. set_clock_groups

```
# Clocks A and C are never active when clocks B and D are active
set_clock_groups -exclusive -group {A C} -group {B D}

# Equivalent specification using false paths
set_false_path -from [get_clocks A] -to [get_clocks B]
set_false_path -from [get_clocks A] -to [get_clocks D]
set_false_path -from [get_clocks C] -to [get_clocks B]
set_false_path -from [get_clocks C] -to [get_clocks D]
set_false_path -from [get_clocks B] -to [get_clocks A]
set_false_path -from [get_clocks B] -to [get_clocks C]
set_false_path -from [get_clocks D] -to [get_clocks A]
set_false_path -from [get_clocks D] -to [get_clocks C]
```

Эффективные характеристики тактов

Команды create_clock и create_generated_clock создают идеальные такты, которые не рассчитывают прочие эффекты платы. В этом разделе описано, как рассчитать эффективные характеристики тактов, такие как задержка и неопределённость тактов.

Задержка тактов

Существует два вида задержки тактов: исходная и задержка сети. Исходная задержка – это задержка, распространяющаяся от источника тактов до определённой точки (например, тактового порта). Задержка сети – это задержка, распространяющаяся от определённой точки до тактового вывода регистра. Общая задержка (или задержка распространения тактов) на тактовом выводе регистра – это сумма исходной задержки и задержки сети по тактовому пути.

Команда set_clock_latency поддерживает только исходную задержку. Опция -source должна быть определена при использовании этой команды.

Используйте команду set_clock_latency для того, чтобы определить исходную задержку для некоторых портов проекта. В примере 7-18 показана команда set_clock_latency и опции.

Example 7–18. set_clock_latency Command

```
set_clock_latency
-source
[-clock <clock_list>]
[-rise | -fall]
[-late | -early]
<delay>
<targets>
```

В таблице 7-11 перечислены опции команды set_clock_latency

Таблица 7-11. Опции команды set_clock_latency (часть 1 из 2)

Опция	Описание
-source	Определяет исходную задержку.
-clock <clock list>	Определяет используемые такты, если цели назначены более одного такта.
-rise | -fall	Определяет нарастающую и спадающую задержки.
-late | -early	Определяет первичное или последнее время поступления тактов.

Таблица 7-11. Опции команды set_clock_latency (часть 2 из 2)

Опция	Описание
<delay>	Определяет значение задержки
<targets>	Определяет такты или источники тактов, если такт тактируется более чем одним тактом.

Временной анализатор Quartus II TimeQuest автоматически подсчитывает задержку сети; поэтому команда set_clock_latency определяет только исходную задержку.

Неопределённость тактов

Команда set_clock_uncertainty определяет неопределённость тактов, или расфазировку тактов, или перенос от такта к такту. Определяется неопределённость раздельно для установки и удержания. Определяется неопределённость раздельно для нарастающего и спадающего фронтов. Временной анализатор Quartus II TimeQuest выделяет неопределённость установки из требуемого времени данных для каждого применяемого пути и добавляет неопределённость удержания в требуемое время данных для каждого применяемого пути.
Используйте команду set_clock_uncertainty для определения некоторой неопределённости тактового порта. В примере 7-19 показана команда set_clock_uncertainty и опции.

Example 7–19. set_clock_uncertainty Command and Options

```
set_clock_uncertainty
[-rise_from <rise from clock> | -fall_from <fall from clock> |
   -from <from clock>]
[-rise_to <rise to clock> | -fall_to <fall to clock> | -to <to clock>]
[-setup | -hold]
<value>
-add
```

В таблице 7-12 перечислены опции команды set_clock_uncertainty

Таблица 7-12. Опции команды set_clock_uncertainty

Опция	Описание
-from <from clock>	Определяет такт from. (откуда)
-rise_from <rise from clock>	Определяет такт rise_from. (нарастает от)
-fall_from <fall from clock>	Определяет такт fall_from. (спадает от)
-to <to clock>	Определяет такт to. (куда)
-rise_to <rise to clock>	Определяет такт rise_to. (нарастает в)
-fall_to <fall to clock>	Определяет такт fall_to. (спадает в)
-setup | -hold	Определяет установку и удержание.
<value>	Значение неопределённости.
-add	Определяет, какое неопределённое <значение> должно быть добавлено к неопределённому значению, полученному командой derive_clock_uncertainty.

Вычисление тактовой неопределённости

Используйте команду derive_clock_uncertainty для автоматического добавления неопределённости: межтактовой, внутритактовой и I/O интерфейса. Обои неопределённости установки и удержания подсчитываются для каждого перехода от такта к такту. В примере 7-20 показана команда derive_clock_uncertainty и опции.

Example 7–20. derive_clock_uncertainty Command

```
derive_clock_uncertainty
[-overwrite]
[-add]
```

В таблице 7-13 перечислены опции команды derive_clock_uncertainty

Таблица 7-13. Опции команды derive_clock_uncertainty

Опция	Описание
-overwrite	Переписывает предыдущие назначения тактовой неопределённости.
-add	Добавляет вычисленные результаты неопределённости любому определённому пользователем назначению неопределённости для такта.

Временной анализатор Quartus II TimeQuest автоматически добавляет тактовую неопределённость переходам от такта к такту в проекте.

Некоторые ограничения тактовой неопределённости будут применяться к паре тактов исходного и назначения с помощью команды set_clock_uncertainty, которая имеет более высокий приоритет, по сравнению с вычислением тактовой неопределённости с помощью команды derive_clock_uncertainty для той же самой пары тактов исходного и назначения. Например, если значения set_clock_uncertainty применяется к clka и clkb, то значения derive_clock_uncertainty для перехода тактов будут проигнорированы по умолчанию. Ограничения set_clock_uncertainty имеют более высокий приоритет над derive_clock_uncertainty.

Значения тактовой неопределённости могут быть использованы; их отчёт представляется для информационных целей. Вы можете использовать команду -overwrite для перезаписи предыдущих назначений, или последующего удаления их вручную с помощью команды remove_clock_uncertainty. Вы можете также использовать опцию -add для добавления тактовой неопределённости, вычисленной командой derive_clock_uncertainty, некоторым определённым ранее значениям тактовой неопределённости.

Следующий список демонстрирует виды переходов от такта к такту, при которых может происходить уточнение тактов. Все они моделируются автоматически с помощью команды derive_clock_uncertainty.

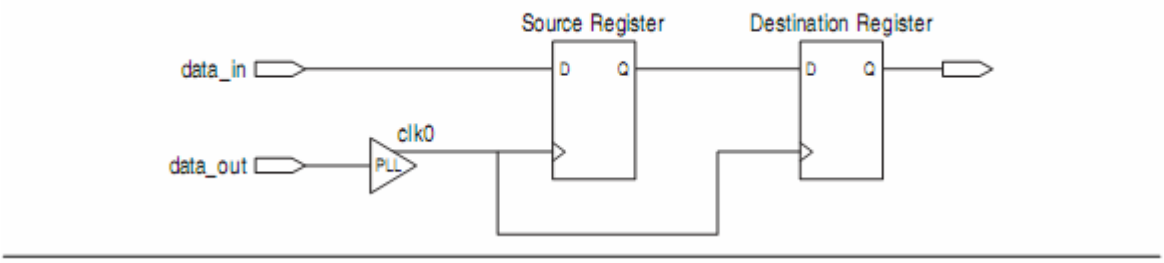
- Межтактовая
- Внутритактовая
- Интерфейса I/O

Altera рекомендует использовать команду derive_clock_uncertainty.

Межтактовые переходы

Межтактовые переходы получаются, когда переход от регистра к регистру происходит в ядре FPGA, а исходные такты и такты назначений поступают с выхода PLL или тактового порта. Пример межтактового перехода показан на рисунке 7-25.

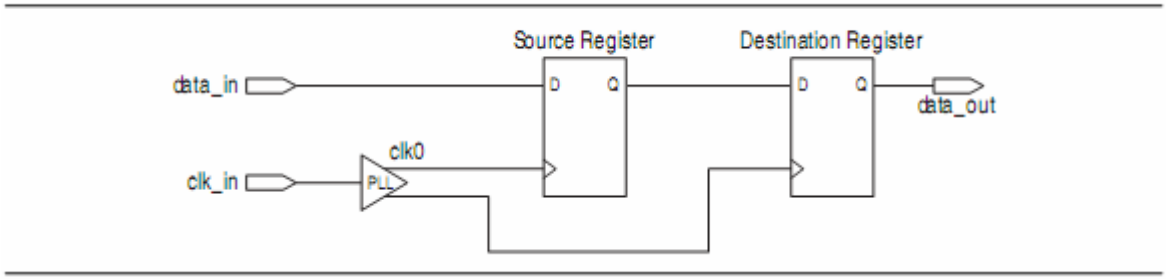
Figure 7–25. Intra-Clock Transfer



Внутритактовые переходы

Внутритактовые переходы получаются, когда переход от регистра к регистру происходит в ядре FPGA, а исходные такты и такты назначений поступают с различных выходов PLL или тактового порта. Пример внутритактового перехода показан на рисунке 7-26.

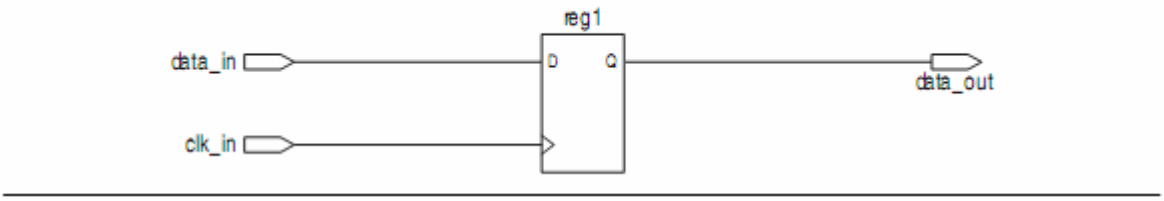
Figure 7–26. Inter-Clock Transfer



Тактовые переходы интерфейса I/O

Тактовые переходы интерфейса I/O получаются, когда данные перемещаются от I/O порта в ядро FPGA (на вход) или от ядра FPGA в I/O порт (на выход). Пример тактового перехода интерфейса I/O показан на рисунке 7-27.

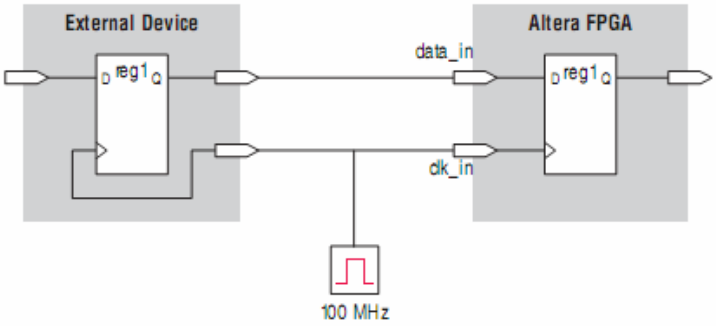
Figure 7–27. I/O Interface-Clock Transfer



Для неопределённости интерфейса I/O, вам необходимо создать виртуальный такт и ограничить порты входа и выхода с помощью команд set_input_delay и set_output_delay, которые ссылаются на виртуальный такт. Виртуальный такт требуется для того, чтобы предотвратить применения тактовой неопределённости некоторым межтактовым и внутритактовым переходам в момент тактового перехода интерфейса I/O, когда команды set_input_delay или derive_clock_uncertainty ссылаются на тактовый порт или выход PLL. Если на виртуальный такт не ссылаются команды set_input_delay или set_output_delay,

команда derive_clock_uncertainty подсчитывает значение неопределённости межтактовой или внутритактовой для интерфейса I/O.
Создавайте виртуальный вывод с теми же свойствами, что и оригинальный такт, управляющий I/O портом. Например, на рисунке 7-28 показан типичный вход I/O интерфейса со спецификацией тактов.

Figure 7–28. I/O Interface Specifications



В примере 7–21 показаны SDC команды ограничения I/O интерфейса, показанного на рисунке 7–28.

Example 7–21. SDC Commands to Constrain the I/O Interface

```
# Create the base clock for the clock port
create_clock -period 10 -name clk_in [get_ports clk_in]
# Create a virtual clock with the same properties of the base clock
# driving the source register
create_clock -period 10 -name virt_clk_in
# Create the input delay referencing the virtual clock and not the base
# clock
# DO NOT use set_input_delay -clock clk_in <delay_value>
# [get_ports data_in]
set_input_delay -clock virt_clk_in <delay value> [get_ports data_in]
```