Тактовые ограничения

Такты играют важную роль во временных ограничениях. Временной анализатор Quartus II TimeQuest поддерживает три различных типа тактов:

- Базовые такты,
- Производные такты,
- Виртуальные такты.

Такты используются для определения ограничений от регистра к регистру для синхронных переходов и управляют алгоритмами оптимизации компоновщика для достижения возможного наилучшего размещения и разводки проекта.

Такты должны быть первыми ограничениями, определёнными в любом SDC файле. Это очень важно, поскольку временной анализатор Quartus II TimeQuest читает SDC ограничения и исключения в порядке от начала к концу файла. Так как большинство ограничений ссылаются на такты, то такты должны быть определены первыми.

Базовые такты

Базовые такты – это первейшие входные такты, генерируемые для FPGA. Эти такты обычно генерируются внешними генераторами или поступают от внешнего чипа. Базовые такты не могут быть тактами, произведёнными внутри FPGA, например, с помощью PLL. Базовые такты должны быть определены первыми, поскольку производные такты и прочие ограничения ссылаются на них.

Используйте SDC команду *create_clock* для ограничения всех первейших входных тактов. Для определения назначений команды *create_clock* нужно использовать коллекцию *get_ports*. Например, определение требования 100 МГц для входного тактового порта *clk_sys*, показанное в примере 8-1.

Example 8–1. create_clock Command

```
create_clock -period 10 [get_ports clk_sys]
```

С помощью опции *-add*, вы можете добавить различные такты к этому тактовому узлу. Например, если два генератора управляют одним тактовым портом в FPGA, должна быть использована следующая SDC команда:

```
create_clock -period 10 [get_ports clk_sys] ↵

create_clock -period 5 [get_ports clk_sys] -add ↵
```

Производные такты

Производные такты – это такты, сгенерированные внутри FPGA. Эти такты модифицируют или синтезируют исходный тактовый сигнал. Например, производные такты от PLL или регистровых делителей частоты. Производные такты должны ограничиваться после того, как ограничены все базовые такты. Этим вы гарантируете то, что все свойства базового такта будут точно сообщены связанному с ним производному такту.

Используйте SDC команду *create_generated_clock* для ограничения всех сгенерированных тактов в проекте. Исходным для *create_generated_clock* должен быть узел в проекте, а не предыдущий ограниченный такт.

Программа TimeQuest поддерживает команду *derive_pll_clocks* для автоматического ограничения всех выходов PLL, используемых в FPGA. С её помощью вы избавляетесь от ручного назначения ограничений для каждого выходного такта PLL.

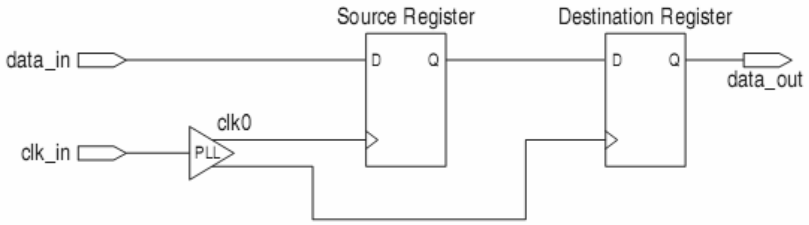
Виртуальные такты

Виртуальные такты – это такты, не имеющие реальных источников в проекте или не взаимодействующие непосредственно с проектом. Виртуальные такты создаются командой *create_clock*, но при этом не определяется цель. Например, для создания 10 нс виртуального такта используем следующую команду:

```
create_clock -period 10 -name my_virt_clk ↵
```

На рисунке 8-1 и примере 8-2 показано, как нужно использовать виртуальные такты. Если FPGA имеет интерфейс с внешним устройством или оба они имеют различные источники тактов, то источник тактов для внешнего устройства может быть смоделирован с помощью виртуального такта.

Figure 8–1. Inter-Clock Transfer



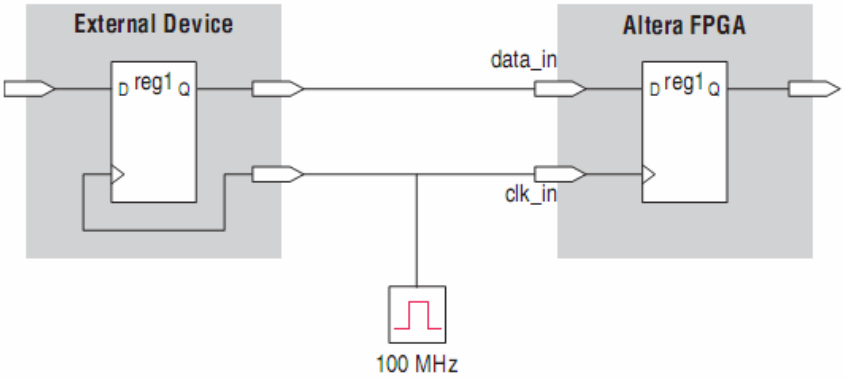
Example 8–2. Virtual Clock Example 1

```
#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]
```

Altera рекомендует вам использовать виртуальные такты, когда вы моделируете внешние задержки, используя ограничения *set_input_delay* и *set_output_delay*. Это очень важно, когда используется в проекте команда *derive_clock_uncertainty*. Виртуальный такт запрещает команде *derive_clock_uncertainty* применять тактовые неопределённости для любых внешних или внутренних переходов в тактовых переходах I/O интерфейса.

Для каждого такта в проекте, ведущего к входному или выходному порту, должен создаваться виртуальный такт. Рисунок 8-2 и пример 8-3 показывают это.

Figure 8–2. I/O Interface Specifications



Example 8–3. 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]
```