SDC and TimeQuest API Reference Manual

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com

MNL-SDCTMQ-1.1

I.S. EN ISO 9001

Contents

About this Reference Manual

This manual provides comprehensive information about SDC and TimeQuest API operations.

Document Revision History

The following table shows the revision history for this reference manual.

Date and Document Version	Changes Made	Summary of Changes
March 2007 v1.1	● Updates to sdc, sdc_ext, and sta API operations. ● Added Revision History to this reference manual.	Updated the sdc and TimeQuest API operations based on the latest Quartus II software release.
May 2006 v1.0	Initial release.	

How to Contact Altera

For the most up-to-date information about Altera products, go to the Altera world-wide web site at www.altera.com. For technical support on this product, go to www.altera.com/mysupport. For additional information about Altera products, consult the sources shown below.

Information Type	USA and Canada	All Other Locations
Technical support	www.altera.com/mysupport/	altera.com/mysupport/
	(800) 800-EPLD (3753) (7:00 a.m. to 5:00 p.m. Pacific Time)	(408) 544-7000 *(1)* (7:00 a.m. to 5:00 p.m. Pacific Time)
Product literature	www.altera.com	www.altera.com
Altera literature services	literature@altera.com *(1)*	lit_req@altera.com *(1)*
Non-technical customer service	(800) 767-3753	(408) 544-7000 (7:30 a.m. to 5:30 p.m. Pacific Time)
FTP site	ftp.altera.com	ftp.altera.com

Note to table:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

This document uses the typographic conventions shown below.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Command names, dialog box titles, checkbox options, and dialog box options are shown in bold, initial capital letters. Example: **Save As** dialog box.
bold type	External timing parameters, directory names, project names, disk drive names, filenames, filename extensions, and software utility names are shown in bold type. Examples: **f_{MAX}, \qdesigns** directory, **d:** drive, **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Document titles are shown in italic type with initial capital letters. Example: *AN 75: High-Speed Board Design.*
Italic type	Internal timing parameters and variables are shown in italic type. Examples: t_{PIA}, $n + 1$. Variable names are enclosed in angle brackets (< >) and shown in italic type. Example: *<file name>*, *<project name>***.pof** file.
Initial Capital Letters	Keyboard keys and menu names are shown with initial capital letters. Examples: Delete key, the Options menu.
"Subheading Title"	References to sections within a document and titles of on-line help topics are shown in quotation marks. Example: "Typographic Conventions."
`Courier type`	Signal and port names are shown in lowercase Courier type. Examples: `data1`, `tdi`, `input`. Active-low signals are denoted by suffix `n`, e.g., `resetn`. Anything that must be typed exactly as it appears is shown in Courier type. For example: `c:\qdesigns\tutorial\chiptrip.gdf`. Also, sections of an actual file, such as a Report File, references to parts of files (e.g., the AHDL keyword `SUBDESIGN`), as well as logic function names (e.g., `TRI`) are shown in Courier.
1., 2., 3., and a., b., c., etc.	Numbered steps are used in a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ● •	Bullets are used in a list of items when the sequence of the items is not important.
✓	The checkmark indicates a procedure that consists of one step only.
☞	The hand points to information that requires special attention.
⚠ CAUTION	A caution calls attention to a condition or possible situation that can damage or destroy the product or the user's work.
⚠	A warning calls attention to a condition or possible situation that can cause injury to the user.
↵	The angled arrow indicates you should press the Enter key.
👣	The feet direct you to more information on a particular topic.

1. Introduction to the SDC and TimeQuest API Reference Manual

Introduction

The TimeQuest Timing Analyzer is a powerful ASIC-style timing analysis tool that validates the timing performance of all logic in a design using industry standard constraint, analysis, and reporting methodology.

The TimeQuest Timing Analyzer includes support for Synopsis Design Constraints (SDC). Additionally, the TimeQuest Timing Analyzer includes an extensive tool command language (Tcl) scripting API. This reference manual includes the supported SDC constraints and commands, and the TimeQuest Tcl API commands.

This overview covers the TimeQuest Timing Analyzer support for scripted operation. At the end of this overview is a table of additional documentation and resources.

What's Inside the SDC and TimeQuest API Reference Manual?

The *Quartus II SDC and TimeQuest API Reference Manual* is your reference guide to TimeQuest Timing Analyzer constraints and commands, including command details, usage, and examples.

All the information included in the *Quartus II SDC and TimeQuest API Reference Manual*, as well as the most up-to-date list of commands, can also be found in the Quartus II software Tcl API and command-line executable online help reference, Qhelp. To access this information within Quartus II design software, type the following command at the command prompt:

```
quartus_sh --qhelp
```

TimeQuest Timing Analyzer Scripting Support

The **sdc** and **sta** packages are supported in the **quartus_sta** command-line executable.

The **sdc** package contains the Synopsys Design Constraints (SDC) functions used to specify constraints and exceptions to the TimeQuest Timing Analyzer.

The **sta** package contains the set of Tcl functions for obtaining advanced information from the TimeQuest Timing Analyzer.

The **quartus_sta** command-line executable validates the timing performance of all logic in a design using industry standard constraint, analysis, and reporting methodology. You can use the TimeQuest analyzer's graphical user interface (GUI) or command-line interface to constrain, run, and view results for all timing paths in your design.

For information about other command-line executables and Tcl packages, refer to the *Quartus II Scripting Reference Manual*.

Related Documentation

Table 1–1 presents additional resources and documentation for the Quartus II development software.

Table 1–1. Quartus II Software Resources and Documentation (Part 1 of 2)

Resource Name	Resource Type	Access	Description	User Level
Quartus II Online Demonstrations	Videos	www.altera.com/quartusdemos	Demonstration of the Quartus II software command-line operation and scripting features.	Beginning to Advanced
Introduction to Quartus II Manual	PDF	www.altera.com/literature/manual/intro_to_quartus2.pdf	An overview of the capabilities of Quartus II software in programmable logic design.	Beginning to Intermediate
Scripting and Constraint Entry section of the Quartus II Handbook	PDF	www.altera.com/literature/lit-qts.jsp	Detailed instruction for command-line operation and Tcl scripting in Quartus II software.	Beginning to Advanced
Qhelp	Quartus II software online help	Run `quartus_sh --qhelp` from the command line	Detailed listing of all command-line executables and Tcl commands including usage examples.	Beginning to Advanced
Enhance Your FPGA Design Flow With Command-Line and Tcl Scripting	Net Seminar (one hour)	www.altera.com/education/net_seminars/past/ns-tcl.html	Overview of Quartus II command-line operation and scripting support.	Beginning to Intermediate

Table 1–1. Quartus II Software Resources and Documentation (Part 2 of 2)

Resource Name	Resource Type	Access	Description	User Level
Design Examples	html	www.altera.com/support/examples/quartus/quartus.html	Instructions for implementing various functions using Quartus II design software.	Beginning to Advanced
Online Training	PowerPoint (PPT) and audio	www.altera.com/training	Detailed instruction examples.	Beginning to Advanced

2. SDC and TimeQuest API Package and Commands

Package Introduction

Table 2–1 lists all the SDC and TimeQuest API commands in alphabetical order. The table lists the package name and the references the page number that documents each command.

Table 2–1. SDC and TimeQuest API Commands (Part 1 of 3)

Command Name	Package	Page
`all_clocks`	sdc	2–5
`all_inputs`	sdc	2–6
`all_outputs`	sdc	2–7
`all_registers`	sdc	2–8
`check_timing`	sta	2–60
`create_clock`	sdc	2–9
`create_generated_clock`	sdc	2–11
`create_slack_histogram`	sta	2–62
`create_timing_netlist`	sta	2–64
`create_timing_summary`	sta	2–66
`delete_timing_netlist`	sta	2–68
`derive_clocks`	sdc	2–13
`derive_pll_clocks`	sdc_ext	2–46
`enable_sdc_extension_collections`	sta	2–69
`get_assignment_groups`	sdc_ext	2–47
`get_cell_info`	sta	2–70
`get_cells`	sdc	2–14
`get_clock_domain_info`	sta	2–71
`get_clock_fmax_info`	sta	2–72
`get_clock_info`	sta	2–73
`get_datasheet`	sta	2–75
`get_clocks`	sdc	2–16
`get_default_sdc_file_names`	sta	2–77
`get_delay_info`	sta	2–78
`get_edge_info`	sta	2–80

Table 2–1. SDC and TimeQuest API Commands (Part 2 of 3)

Command Name	Package	Page
get_keepers	sdc_ext	2–51
get_fanins	sdc_ext	2–48
get_fanouts	sdc_ext	2–49
get_net_info	sta	2–82
get_nets	sdc	2–17
get_node_info	sta	2–83
get_nodes	sdc_ext	2–52
get_object_info	sta	2–84
get_partition_info	sta	2–85
get_partitions	sdc_ext	2–53
get_path_count	sta	2–86
get_path_info	sta	2–88
get_pin_info	sta	2–91
get_pins	sdc	2–18
get_point_info	sta	2–92
get_port_info	sta	2–96
get_ports	sdc	2–20
get_register_info	sta	2–97
get_registers	sdc_ext	2–54
get_timing_paths	sta	2–98
read_sdc	sta	2–101
remove_clock	sdc_ext	2–55
remove_clock_groups	sdc	2–21
remove_clock_latency	sdc	2–22
remove_clock_uncertainty	sdc	2–24
remove_input_delay	sdc	2–25
remove_output_delay	sdc	2–26
report_clock_transfers	sta	2–104
report_clock_fmax_summary	sta	2–102
report_clocks	sta	2–106
report_datasheet	sta	2–108
report_min_pulse_width	sta	2–110
report_net_timing	sta	2–112

Table 2–1. SDC and TimeQuest API Commands (Part 3 of 3)

Command Name	Package	Page
`report_path`	sta	2–114
`report_sdc`	sta	2–117
`report_timing`	sta	2–119
`report_ucp`	sta	2–122
`reset_design`	sdc	2–27
`set_clock_groups`	sdc	2–28
`set_clock_latency`	sdc	2–29
`set_clock_uncertainty`	sdc	2–31
`set_false_path`	sdc	2–32
`set_input_delay`	sdc	2–34
`set_max_delay`	sdc	2–36
`set_min_delay`	sdc	2–38
`set_multicycle_path`	sdc	2–40
`set_output_delay`	sdc	2–43
`set_scc_mode`	sdc_ext	2–56
`set_time_format`	sdc_ext	2–57
`timing_netlist_exist`	sta	2–124
`update_timing_netlist`	sta	2–125
`use_timequest_style_escaping`	sta	2–126
`write_sdc`	sta	2–128

sdc

Synopsys Design Constraint (SDC) is a format used to specify the design intent, including the timing and area constraints of the design. The TimeQuest Timing Analyzer only implements the set of SDC commands required to specify the timing constraints of the design. For area constraints, the QSF file should be used.

This package implements the SDC Spec Version 1.5 (June 2005).

Any command in this package can be specified in a TimeQuest SDC file.

This package is loaded by default in the following executable:

- quartus_sta

This package includes the following commands:

all_clocks

Usage

```
all_clocks
```

Options

None

Description

Returns a collection of all clocks in the design.

Example

```
project_open chiptrip
create_timing_netlist
foreach_in_collection clk [all_clocks] {
   puts [get_clock_info -name $clk]
}
delete_timing_netlist
project_close
```

all_inputs

Usage

```
all_inputs
```

Options

None

Description

Returns a collection of all input ports in the design.

Example

```
project_open chiptrip
create_timing_netlist
set_input_delay -clock clock1 2.0 [all_inputs]
delete_timing_netlist
project_close
```

all_outputs

Usage

```
all_outputs
```

Options

None

Description

Returns a collection of all output ports in the design.

Example

```
project_open chiptrip
create_timing_netlist
set_output_delay -clock clock1 2.0 [all_outputs]
delete_timing_netlist
project_close
```

all_registers

Usage

```
all_registers
```

Options

None

Description

Returns a collection of all regisers in the design.

Example

```
project_open chiptrip
create_timing_netlist
report_timig -from [all_regisers] -to [all_registers]
delete_timing_netlist
project_close
```

create_clock

Usage

```
create_clock [-add] [-name <clock_name>] -period <value> [-waveform
<edge_list>] <targets>
```

Options

```
-add: Adds clock to a node with an existing clock
-name <clock_name>: Clock name of the created clock
-period <value>: Speed of the clock in terms of clock period
-waveform <edge_list>: List of edge values
<targets>: List or collection of targets
```

Description

Defines a clock. If -name is not specified, the clock name is the same as the first target in the list or collection. The clock name is used to refer to the clock in other commands.

The -period option specifies the clock period. It is also possible to specify a frequency in this field to define the clock period. This can be done by using -period option followed by either <frequency>MHz or "<frequency> MHz". However, this is a TimeQuest only extension and makes the SDC non-standard.

The -waveform specifies the rising and falling edges (or duty cycle) of the clock, and is specified as a list of two time values: the first rising edge and the next falling edge. The rising edge must be within the range [0, period). The falling edge must be within one clock period of the rising edge. The waveform defaults to {0 period/2}.

If a clock is already assigned to a given target, the create_clock command overwrites the clock for that target unless the -add option is used. The -add option can be used to assign multiple clocks to a pin or port.

If the target of the clock is internal (i.e. not an input port), the source latency defaults to zero.

If a clock is on a path after another clock, then it blocks or overwrites the previous clock from that point forward.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_styl_style_escaping for details.

Example

```
# Create a simple 10ns with clock with a 60% duty cycle
create_clock -period 10 -waveform {0 6} -name clk [get_ports clk]

# Create a clock with a falling edge at 2ns, rising edge at 8ns,
# falling at 12ns, etc.
create_clock -period 10 -waveform {8, 12} -name clk [get_ports clk]

# Assign two clocks to an input port that are switched externally
create_clock -period 10 -name clk100Mhz [get_ports clk]
create_clock -period 6.667 -name clk150Mhz -add [get_ports clk]

# Two ways to use MHz to define clock period (TimeQuest only)
create_clock -period 250MHz -name clk250MHz [get_ports clk]
create_clock -period "250 MHz" -name clk250MHz [get_ports clk]
```

create_generated_clock

Usage

```
create_generated_clock [-add] [-divide_by <factor>] [-duty_cycle
<percent>] [-edge_shift <shift_list>] [-edges <edge_list>] [-invert] [-
master_clock <clock>] [-multiply_by <factor>] [-name <clock_name>] [-
offset <time>] [-phase <degrees>] -source <clock_source> <targets>
```

Options

```
-add: Add clock to existing clock node
-divide_by <factor>: Division factor
-duty_cycle <percent>: Specifies the duty cycle as a percentage of the clock
period - accepts floating point values
-edge_shift <shift_list>: List of edge shifts
-edges <edge_list>: List of edge values
-invert: Invert the clock waveform
-master_clock <clock>: Specifies clock of the source node
-multiply_by <factor>: Multiplication factor
-name <clock_name>: Clock name of generated clock
-offset <time>: Specifies the offset as an absolute time shift
-phase <degrees>: Specifies the phase shift in degrees
-source <clock_source>: Source node for the generated clock
<targets>: List or collection of targets
```

Description

Defines an internally generated clock. If the "-name" option is not specified, the clock name is the same as the first target in the list or collection. The clock name is used to refer to the clock in other commands.

If a clock is already assigned to a given target, the create_generated_clock command overwrites the clock for that target unless the "-add" option is used. The "-add" option can be used to assign multiple clocks to a pin or port, and is recommended be used with "-master_clock" option.

The source of the generated clock, specified by "-source", is a pin or port in the design. All waveform modifications are relative to this point. If more than one clock feeds the source node, the "-master_clock" option must be used to specify which clock to modify.

The source latency of the generated clock is based on its own clock network, and not the clock network of the "-source" node. This latency is added to any source latency of the master clock.

The -divide_by, "-multiply_by", "-invert", "-duty_cycle", "-edges", and "-edge_shift" options modify the waveform relative to the waveform at the source node.

Clock division and multiplication using "-divide_by" and "-multiply_by" is performed relative to the first rising edge. Clock division is based on edges in the master clock waveform, and scaled if the division is an odd number. The "-duty_cycle" option can be used to specify the new duty cycle for clock multiplication. The "-phase" "option" can be used to specify any phase shift relative to the new clock period. The "-offset" option can be used to specify an arbitrary offset or time shift. The "-invert" option inverts the waveform.

Clock generation can also be specified with the "-edges" and "-edge_shift" options. The "-edges" option takes a list of three numbers specifying the master clock edges to use for the first rising edge, next falling edge, and next rising edge. Edges of the master clock are labeled according to the first rising edge (1), next falling edge (2), next rising edge (3), etc. For example, a basic clock divider can be specified equivalently with "-divide_by 2" or "-edges {1 3 5}". The "-edge_shift" option takes a list of three time values, the amount to shift each of the three edges.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Create a clock and a divide-by-2 generated clock
create_clock -period 10 [get_ports clk]
create_generated_clock -divide_by 2 -source [get_ports clk] -name clkdiv \
    [get_registers clkdiv]

# An equivalent generated clock
create_generated_clock -edges {1 3 5} -source [get_ports clk] -name \
    clkdiv [get_registers clkdiv]

# Specify a clock multipler with a 60% duty cycle
create_generated_clock -multiply_by 2 -duty_cycle 60 \
    [get_pins clkmult|combout]

# Specify an inverted divide-by-2 clock relative to the output of the
# source clock
create_generated_clock -divide_by 2 -invert -source [get_ports clk] -name \
    nclkdiv [get_registers clkdiv]

# Specify a divide-by-2 clock with a 90-degree phase shift
create_generated_clock -divide_by 2 -phase 90 -source [get_ports clk] \
    -name clkdiv [get_registers clkdiv]

# Assign two clocks to an input port that are switched externally,
# along with an internal clock divider.
create_clock -period 10 -name clk100Mhz [get_ports clk]
create_clock -period 6.667 -name clk150Mhz -add [get_ports clk]
create_generated_clock -divide_by 2 -name clk50Mhz -source [get_ports \
    clk] -master_clock clk100Mhz -add [get_registers clkdiv]
create_generated_clock -divide_by 2 -name clk75Mhz -source [get_ports \
    clk] -master_clock clk150Mhz -add [get_registers clkdiv]
```

derive_clocks

Usage

```
derive_clocks -period <period_value> [-waveform <edge_list>]
```

Options

```
-period <period_value>: Speed of the default clock in terms of clock period
-waveform <edge_list>: List of edge values
```

Description

Creates a clock on sources of clock pins in the design that do not already have at least one clock sourcing the clock pin. This command is equivalent to calling create_clock on each clock source in the design that does not already have a clock assigned to it.

See the help for create_clock for more information.

Altera does not recommend using this command during final sign-off analysis of a design. derive_clocks should only be used early in the design phase when the clocks are not completely known. When possible, create_clock and create_generated_clock should be used instead.

Example

```
# Automatically create a 10ns, 60% duty cycle clock on all
# unconstrained clock sources
derive_clocks -period 10 -waveform {0 6}
```

get_cells

Usage

```
get_cells [-compatibility_mode] [-hierarchical] [-nocase] <filter>
```

Options

```
-compatibility_mode: Use simple Tcl matching (TAN style)
-hierarchical: Specifies if hierarchical searching method should be used
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of cells in the design. All cell names in the collection match the specified pattern. Wildcards can be used to select multiple cells at once.

There are three Tcl string matching schemes available with this command: default, "-hierarchical", and "-compatibility".

When you use the default matching scheme, use pipe characters to separate one hierarchy level from the next. They are treated as special characters and are taken into account when string matching with wildcards is performed. When this matching scheme is enabled, the specified pattern is matched against absolute cell names - the names that include the entire hierarchical path. Note that a full cell name can contain multiple pipe characters in it to reflect the hierarchy. All hierarchy levels in the pattern are matched level by level. Any included wildcards refer to only one hierarchical level. For example, "*" and "*|*" will produce different collections since they refer to the highest hierarchical level and second highest hierarchical level respectively.

When using the "-hierarchical" matching scheme, pipe characters are treated as special characters and are taken into account when string matching with wildcards is performed. This matching scheme forces the search to proceed recursively down the hierarchy. The specified pattern is matched against the relative cell names: the immediate names that do not include any of the hierarchy information. Note that a short cell name cannot contain pipe characters in it. Any included wildcards are expanded to match the relative pin names.

The "-compatibility" matching scheme mimics the string matching behavior in the Classic timing analysis engine. The simple Tcl string matching on full, absolute cell names is used. Pipes are not treated as special characters when used with wildcards.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Find a cell called "reg" using case insensitive search
get_cells -nocase reg
# Create a collection of all cells whose names start with "reg"
get_cells reg*
# Create a collection of all cells on the highest hierarachical level
set mycollection [get_cells *]
# Create a collection of all cells in the design
set fullcollection [get_cells -hierarchical *]
```

get_clocks

Usage

```
get_clocks [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of clocks in the design. When used as an argument to another command, such as the -from or -to of set_multicycle_path, each node in the clock represents all nodes clocked by the clocks in the collection.

```
# The following multicycle applies to all paths ending at registers
# clocked by clk
set_multicycle_path -to [get_clocks clk] 2
```

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open chiptrip
create_timing_netlist
read_sdc
update_timing_netlist
set clocks [get_clocks c* -nocase]
foreach_in_collection clk $clocks {
    set name [get_clock_info -name $clk]
    set period [get_clock_info -period $clk]
    puts "$name: $period"
}
delete_timing_netlist
project_close
```

get_nets

Usage

```
get_nets [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of nets in the design. All net names in the collection match the specified pattern. Wildcards can be used to select multiple nets at once.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Find a net called "reg" using case insensitive search
get_nets -nocase reg
# Create a collection of all nets whose names start with "reg"
get_nets reg*
# Create a collection of all nets in the design
set mycollection [get_nets *]
```

get_pins

Usage

```
get_pins [-compatibility_mode] [-hierarchical] [-nocase] <filter>
```

Options

```
-compatibility_mode: Use simple Tcl matching (TAN style)
-hierarchical: Specifies if hierarchical searching method should be used
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of pins in the design. All pin names in the collection match the specified pattern. Wildcards can be used to select multiple pins at once.

There are three Tcl string matching schemes available with this command: default, "-hierarchical", and "-compatibility".

By default, pipe characters are used to separate one hierarchy level from the next. They are treated as special characters and are taken into account when string matching with wildcards is performed. When this matching scheme is enabled, the specified pattern is matched against absolute pin names - the names that include the entire hierarchical path. All hierarchy levels in the pattern are matched level by level. Pin names of the form <absolute full cell name>|<pin suffix> are used for matching. Note that a full cell name can contain multiple pipe characters in it to reflect the hierarchy. Any included wildcards refer to only one hierarchical level. For example, "*|*" and "*|*|*" will produce different collections since they refer to the highest hierarchical level and second highest hierarchical level respectively.

When uisng the "-hierarchical" matching scheme, Pipe characters are treated as special characters and are taken into account when string matching with wildcards is performed. This matching scheme forces the search to proceed recursively down the hierarchy. The specified pattern is matched against the relative pin names: the immediate names that do not include any of the hierarchy information. Pin names of the form <relative short cell name>|<pin suffix> are used for matching. Note that a short cell name cannot contain pipe characters. Any included wildcards are expanded to match the relative pin names. For example, "*" and "*|*" will matchexactly the same pins since the former is expanded into the latter.

The "-compatibility" matching scheme mimics the string matching behavior of the Classic timing analysis engine. The simple Tcl string matching on full, absolute pin names is used. Pipe characters are not treated as special characters when used with wildcards.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Get regout pin of "reg" cell
get_pins -nocase reg|regout
# Create a collection of all pins of "reg" cell
get_pins reg|*
# Create a collection of all pins on the highest hierarachical level
set mycollection [get_pins *]
# Create a collection of all pins in the design
set fullcollection [get_pins -hierarchical *]
```

get_ports

Usage

```
get_ports [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of ports (design inputs and outputs) in the design.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open chiptrip
create_timing_netlist

# Get all ports starting with "In".
set ports [get_ports In*]
foreach_in_collection port $ports {
    puts [get_port_info -name $port]
}

delete_timing_netlist
project_close
```

remove_clock_groups

Usage

```
remove_clock_groups -all
```

Options

```
-all: Specify remove all clock group settings
```

Description

Remove all clock group assignments. This command will remove any clock groups that have been previously set. There is no way to remove specific groups.

Example

```
project_open top
create_timing_netlist
create_clock -period 10.000 -name clkA [get_ports sysclk[0]]
create_clock -period 10.000 -name clkB [get_ports sysclk[1]]

# Set clkA and clkB to be mutually exclusive clocks.
set_clock_groups -exclusive -group {clkA} -group {clkB}
set_clock_groups -exclusive -group {clkC} -group {clkD}

# Remove clock groups A, B, C, and D. Result is that there
# will no longer be any mutually exclusive clocks.
remove_clock_groups -all
```

remove_clock_latency

Usage

```
remove_clock_latency -source <targets>
```

Options

```
-source: Specifies the source clock latency
<targets>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Removes clock latency for a given clock or clock target.

There are two types of latency: network and source. Network latency is the clock network delays between the clock and register clock pins. Source latency is the clock network delay between the clock and its source (e.g., system clock or base clock of a generated clock).

The TimeQuest Timing Analyzer automatically computes network latencies for all register and generated clocks. Overriding clock network latencies is not supported by the TimeQuest analyzer. Therefore, the "-source" option must always be specified. Remove_clock_latency requires this option as well.

You can apply clock latency to a clock, which affects all targets of the clock, or to a specific clock target. Therefore, you can remove clock latency from a collection of clocks, or from a collection of target nodes. The remove_clock_latency command removes all latencies from a clock or node, so removing a node's clock latency with respect to a particular clock, or removing only latencies with particular conditions is not supported.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
create_clock -name SYSCLK -period 10.000 [get_ports inclk]
create_generated_clock -name OUTCLK -divide_by 1 -source [get_ports \
    inclk] [get_ports outclk]
create_generated_clock -name FDBKCLK -divide_by 1 -source \
    [get_ports outclk] [get_ports fdbkclk]

# Apply a simple 2.000 ns source latency to the system clock.
set_clock_latency -source 2.000 [get_clocks SYSCLK]

# Specify feedback clock latencies between output port outclk
# and the output port fdbkclk.
set_clock_latency -source -late -rise 0.800 [get_clocks FDBKCLK]
set_clock_latency -source -late -fall 0.750 [get_clocks FDBKCLK]
set_clock_latency -source -early -rise 0.500 [get_clocks FDBKCLK]
set_clock_latency -source -early -fall 0.460 [get_clocks FDBKCLK]

# Remove all clock latency from FDBKCLK
remove_clock_latency -source [get_clocks FDBKCLK]
```

remove_clock_uncertainty

Usage

```
remove_clock_uncertainty -from <from_clock> -to <to_clock>
```

Options

```
-from <from_clock>: Valid destinations (string patterns are matched using
Tcl string matching)
-to <to_clock>: Valid destinations (string patterns are matched using Tcl
string matching)
```

Description

Removes clock uncertainty between (from-to) a collection of clocks and another collection of clocks. The source and destination clocks can be any arbitrary collection of clocks. This command removes all uncertainty between two clocks. If there is no uncertainty between two clocks specified in remove_clock_uncertainty, the command does nothing for those two clocks but continues to remove uncertainty between other clocks specified.

The values of the "-from" and "-to" options are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
set_clock_uncertainty -setup -rise_from {clk1 clk2} -fall_to {clk3 clk4} \
    200ps
set_clock_uncertainty -from {clk5 clk6} -to {clk7 clk8} 300ps
remove_clock_uncertainty -from {clk3 clk5} -to {clk4 clk7}
```

remove_input_delay

Usage

```
remove_input_delay <targets>
```

Options

```
<targets>: Collection or list of input ports
```

Description

Removes input delay from a port. For each input port specified, removes all input delays for that port. This means that rise, fall, maximum, and minimum delays for each clock and reference pin on the input port are all removed.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Simple input delay with the same value for min/max and rise/fall
set_input_delay -clock clk 1.5 [get_ports {in1 in2}]
set_input_delay -clock clk2 1.5 [get_ports {in1 in2}]
set_input_delay -clock clk 1.6 [get_ports {in3 in4}]

# Remove input delay on ports in1 and in4,
# for all flags and reference pins and flags
remove_input_delay [get_ports {in1 in4}]
```

remove_output_delay

Usage

```
remove_output_delay <targets>
```

Options

```
<targets>: Collection or list of input ports
```

Description

Removes output delay from a port. For each output port specified, removes all output delays for that port. This means that rise, fall, maximum, and minimum delays for each clock and reference pin on the output port are all removed.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Simple output delay with the same value for min/max and rise/fall
set_output_delay -clock clk 1.5 [get_ports {out1 out2}]
set_output_delay -clock clk2 1.5 [get_ports {out1 out2}]
set_output_delay -clock clk 1.6 [get_ports {out3 out4}]

# Remove input delay on ports out1 and out4,
# for all flags and reference pins and flags
remove_output_delay [get_ports {out1 out4}]
```

reset_design

Usage

```
reset_design
```

Options

None

Description

Removes all assignments from the design. This includes clocks, generated clocks, derived clocks, input delays, output delays, clock latency, clock uncertainty, clock groups, false paths, multicycle paths, min delays, and max delays. After reset_design is called, the design should be in the same state as it would be if create_timing_netlist was just called.

Example

```
# Constrain design
create_clock -name clk -period 4.000 -waveform { 0.000 2.000 } \
    [get_ports clk]
set_input_delay -clock clk2 1.5 [get_ports in*]
set_output_delay -clock clk 1.6 [get_ports out*]
set_false_path -from [get_keepers in] -through [get_nets r1] -to \
    [get_keepers out]

# Reset the design to the state that it was in before any constraints
# were entered
reset_design
```

set_clock_groups

Usage

```
set_clock_groups [-asynchronous] [-exclusive] -group <names>
```

Options

```
-asynchronous: Specify mutually exclusive clocks (same as the -exclusive
option).  Exists for compatibility.
-exclusive: Specify mutually exclusive clocks
-group <names>: Valid destinations (string patterns are matched using Tcl
string matching)
```

Description

Clock groups provide a quick and convenient way to specify which clocks are not related. Asynchronous clocks are those that are completely unrelated (e.g., have different ideal clock sources). Exclusive clocks are those that are not active at the same time (e.g., multiplexed clocks). TimeQuest treats both options "-exclusive" and "-asynchronous" as if they were the same.

The result of set_clock_groups is that all clocks in any group are cut from all clocks in every other group. This command is equivalent to calling set_false_path from each clock in every group to each clock in every other group and vice versa, making set_clock_groups easier to specify for cutting clock domains. The use of a single -group option tells TimeQuest to cut this group of clocks from all other clocks in the design, including clocks that are created in the future.

Example

```
project_open top
create_timing_netlist
create_clock -period 10.000 -name clkA [get_ports sysclk[0]]
create_clock -period 10.000 -name clkB [get_ports sysclk[1]]

# Set clkA and clkB to be mutually exclusive clocks.
set_clock_groups -exclusive -group {clkA} -group {clkB}

# The previous line is equivalent to the following two commands.
set_false_path -from [get_clocks clkA] -to [get_clocks clkB]
set_false_path -from [get_clocks clkB] -to [get_clocks clkA]
```

set_clock_latency

Usage

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

Options

```
-clock <clock_list>: Valid clock destinations (string patterns are matched
using Tcl string matching)
-early: Specifies the early clock latency
-fall: Specifies the falling transition clock latency
-late: Specifies the late clock latency
-rise: Specifies the rising transition clock latency
-source: Specifies the source clock latency
<delay>: Latency delay value
<targets>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Specifies clock latency for a given clock or clock target.

There are two types of latency: network and source. Network latency is the clock network delays between the clock and register clock pins. Source latency is the clock network delay between the clock and its source (e.g., system clock or base clock of a generated clock).

The TimeQuest Timing Analyzer automatically computes network latencies for all register and generated clocks. Overriding clock network latencies is not supported by the TimeQuest analyzer. Therefore, the "-source" option must always be specified.

You can apply clock latency to a clock, which affects all targets of the clock, or to a specific clock target. If you specify a specific clock target that is clocked by more than one clock, the "-clock" option is used to specify which clock to use. Latencies assigned to a clock target override any latencies assigned to a clock.

Different clock latencies can be specified for early ("-early") and late ("-late") latencies, as well as for rising edges ("-rise") and falling edges ("-fall"). If only some combinations are specified, the other combinations are used by default. For example, if only a "-rise -early" latency and a "-fall -early" latency are specified, then "-rise -late" latency is assumed to be the same as the "-rise -early" latency and the "-fall -late" latency is assumed to be the same as the "-fall -early" latency. If neither "-rise" nor "-fall" are used or neither "-early" nor "-fall" are used, then the latency applies to both conditions.

Source latency can also be assigned to generated clocks. As an example, this may be useful for specifying board level delays from a clock output port to a clock input port when the clock input port is acting as a feedback clock.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
create_clock -name SYSCLK -period 10.000 [get_ports inclk]
create_generated_clock -name OUTCLK -divide_by 1 -source [get_ports \
    inclk] [get_ports outclk]
create_generated_clock -name FDBKCLK -divide_by 1 -source \
    [get_ports outclk] [get_ports fdbkclk]

# Apply a simple 2.000 ns source latency to the system clock.
set_clock_latency -source 2.000 [get_clocks SYSCLK]

# Specify feedback clock latencies between output port outclk
# and the output port fdbkclk.
set_clock_latency -source -late -rise 0.800 [get_clocks FDBKCLK]
set_clock_latency -source -late -fall 0.750 [get_clocks FDBKCLK]
set_clock_latency -source -early -rise 0.500 [get_clocks FDBKCLK]
set_clock_latency -source -early -fall 0.460 [get_clocks FDBKCLK]
```

set_clock_uncertainty

Usage

```
set_clock_uncertainty [-fall_from <fall_from_clock>] [-fall_to
<fall_to_clock>] [-from <from_clock>] [-hold] [-rise_from
<rise_from_clock>] [-rise_to <rise_to_clock>] [-setup] [-to <to_clock>]
<uncertainty>
```

Options

```
-fall_from <fall_from_clock>: Valid destinations (string patterns are
matched using Tcl string matching)
-fall_to <fall_to_clock>: Valid destinations (string patterns are matched
using Tcl string matching)
-from <from_clock>: Valid destinations (string patterns are matched using
Tcl string matching)
-hold: Specifies the uncertainty value (applies to clock hold or removal
checks)
-rise_from <rise_from_clock>: Valid destinations (string patterns are
matched using Tcl string matching)
-rise_to <rise_to_clock>: Valid destinations (string patterns are matched
using Tcl string matching)
-setup: Specifies the uncertainty value (applies to clock setup or recovery
checks) (default)
-to <to_clock>: Valid destinations (string patterns are matched using Tcl
string matching)
<uncertainty>: Uncertainty
```

Description

Specifies clock uncertainty or skew for clocks or clock-to-clock transfers. You can specify uncertainty separately for setup and hold, and can specify separate rising and falling clock transitions. The setup uncertainty is subtracted from the data required time for each applicable path, and the hold uncertainty is added to the data required time for each applicable path.

The values of the "-from", and "-to" options are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
set_clock_uncertainty -setup -rise_from clk1 -fall_to clk2 200ps
```

set_false_path

Usage

```
set_false_path [-fall_from <names>] [-fall_to <names>] [-from <names>] [-
hold] [-rise_from <names>] [-rise_to <names>] [-setup] [-through <names>]
[-to <names>]
```

Options

```
-fall_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-fall_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-hold: Specifies the false_path value (applies only to clock hold or removal
checks)
-rise_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-rise_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-setup: Specifies the false_path value (applies only to clock setup or
recovery checks)
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
```

Description

Specifies a false-path exception, removing (or cutting) paths from timing analysis.

The "-from" and "-to" values are collections of clocks, registers, ports, pins, or cells in the design. If the "-from" or "-to" values are not specified, the collection is converted automatically into [get_keepers *]. It is worth noting that if the counterpart of the unspecified collection is a clock collection, it is more efficient to explicitly specify this collection as a clock collection only if the clock collection also generates the desired assignment.

Applying exceptions between clocks applies the exception from all register or ports clocked by the "-from" clock to all registers or ports clocked by the "-to" clock. Also, applying exceptions between a pair of clocks is more efficient than for specific node to node or node to clock paths.

If pin names or collections are used, the "-from" value must be a clock pin and the "-to" value must be any non-clock input pin to a register. Assignments from clock pins or to and from cells applies to all registers in the cell or clocked by the clock pin.

The "-through" values are collections of pins or nets in the design. An exception applied through a node in the design applies only to paths through the specified node.

The "-rise_from" and "-fall_from" options can be used in place of the "-from" destination nodes. The rise or fall value of the option indicates that the "from" nodes are clocked by the rising or falling edge of the clock that feeds this node, taking into consideration any logical inversions along the clock path. The "-from" option is the combination of both rising and falling "from" nodes. If the "from" collection is a clock collection, the assignment applies to those nodes that are clocked by the respective rising or falling clock edge.

The "-rise_to" and "-fall_to" options behave similarly to the "from" options described previously. These assignments restrict the given assignment to only those nodes or clocks that correspond to the specified rise or fall value, taking into consideration any logical inversions that are along the clock path.

The "-setup" and "-hold" options allow the false path to only be applied to the corresponding setup/recovery or hold/removal checks. The default if neither value is specified is to apply the false path to both "-setup" and "-hold".

The values for "-from", "-to", and "-through" are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

See also set_clock_groups.

Example

```
# Set a false-path between two unrelated clocks
# See also set_clock_groups
set_false_path -from [get_clocks clkA] -to [get_clocks clkB]

# Set a false-path for a specific path
set_false_path -from [get_pins regA|clk] -to [get_pins regB|aclr]

# Set a false-path from a node to a falling clock
set_false_path -from [get_pins regA|clk] -fall_to [get_clocks clkB]
```

set_input_delay

Usage

```
set_input_delay [-add_delay] -clock <name> [-clock_fall] [-fall] [-max] [-
min] [-reference_pin <name>] [-rise] [-source_latency_included] <delay>
<targets>
```

Options

```
-add_delay: Add to existing delays instead of overriding them
-clock <name>: Clock name
-clock_fall: Specifies input delay relative to the falling edge of the clock
-fall: Specifies the falling input delay at the port
-max: Applies value as maximum data arrival time
-min: Applies value as minimum data arrival time
-reference_pin <name>: Specifies a pin or port in the design to which the
input delay is relative
-rise: Specifies the rising input delay at the port
-source_latency_included: Specifies input delay including added source
latency
<delay>: Time value
<targets>: Collection or list of input ports
```

Description

Specifies the data arrival times at the specified input ports relative the clock specified by the "-clock" option. The clock must refer to a clock name in the design.

Input delays can be specified relative to the rising edge (default) or falling edge (-clock_fall) of the clock.

If the input delay is specified relative to a simple generated clock (a generated clock with a single target), the clock arrival times to the generated clock will be added to the data arrival time.

Input delays can be specified relative to a pin or port ("-reference_pin") in the clock network. Clock arrival times to the reference pin or port are added to data arrival times.

Input delays can already include clock source latency. By default the clock source latency of the related clock will be added to the input delay value, but when the "-souce_latency_included" option is specified, the clock source latency is not added because it was factored into the input delay value.

The maximum input delay (-max) is used for clock setup checks or recovery checks and the minimum input delay (-min) is used for clock hold checks or removal checks. If only one of "-min" and "-max" are specified (or neither) for a given port, the same value is used for both.

Separate rising (-rise) and falling (-fall) arrival times at the port can be specified. If only "-rise" or "-fall" are specified for a given port, the same value is used for both.

By default, set_input_delay removes any other input delays to the port except for those with the same "-clock," "-clock_fall," and "-reference_pin" combination. Multiple input delays relative to different clocks, clock edges, or reference pins can be specified using the "-add_delay" option.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Simple input delay with the same value for min/max and rise/fall
set_input_delay -clock clk 1.5 [get_ports in*]

# Input delay with respect to the falling edge of clock
set_input_delay -clock clk -clock_fall 1.5 [get_ports in*]

# Input delays for different min/max and irse/fall combinations
set_input_delay -clock clk -max -rise 1.4 [get_ports in*]
set_input_delay -clock clk -max -fall 1.5 [get_ports in*]
set_input_delay -clock clk -min -rise 0.7 [get_ports in*]
set_input_delay -clock clk -min -fall 0.8 [get_ports in*]

# Adding muliple input delays with respect to more than one clock
set_input_delay -clock clkA -min 1.2 [get_ports in*]
set_input_delay -clock clkA -max 1.8 [get_ports in*]
set_input_delay -clock clkA -clock_fall 1.6 [get_ports in*] -add_delay
set_input_delay -clock clkB -min 2.1 [get_ports in*] -add_delay
set_input_delay -clock clkB -max 2.5 [get_ports in*] -add_delay

# Specifying an input delay relative to an external clock output port
set_input_delay -clock clk -reference_pin [get_ports clkout] 0.8 \
    [get_ports in*]
```

set_max_delay

Usage

```
set_max_delay [-fall_from <names>] [-fall_to <names>] [-from <names>] [-
rise_from <names>] [-rise_to <names>] [-through <names>] [-to <names>]
<value>
```

Options

```
-fall_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-fall_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-rise_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-rise_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
<value>: Time Value
```

Description

Specifies a maximum delay exception for a given path.

The maximum delay is similar to changing the setup relationship (latching clock edge - launching clock edge), except that it can be applied to input or output ports without input or input delays assigned to them. Maximum delays are always relative to any clock network delays (if the source or destination is register) or any input or output delays (if the source or destination is a port). Therefore, input delays and clock latencies are added to the data arrival times. Clock latencies also added to data required times and output delays are subtracted from data required times.

The "-from" and "-to" values are collections of clocks, registers, ports, pins, or cells in the design. If the "-from" or "-to" values are not specified, the collection is converted automatically into [get_keepers *]. It is worth noting that if the counterpart of the unspecified collection is a clock collection, it is more efficient to explicitly specify this collection as a clock collection, but only if the clock collection also generates the desired assignment.

Applying exceptions between clocks applies the exception from all register or ports clocked by the "-from" clock to all registers or ports clocked by the "-to" clock. Also, applying exceptions between a pair of clocks is more efficient than for specific node to node or node to clock paths.

If pin names or collections are used, the "-from" value must be a clock pin and the "-to" value must be any non-clock input pin to a register. Assignments from clock pins or to and from cells applies to all registers in the cell or clocked by the clock pin.

The "-through" values are collections of pins or nets in the design. An exception applied through a node in the design applies only to paths through the specified node.

The "-rise_from" and "-fall_from" options can be used in place of the "-from" destination nodes. The rise or fall value of the option indicates that the "from" nodes are clocked by the rising or falling edge of the clock that feeds this node taking into consideration any logical inversions along the clock path. The "-from" option is the combination of both rising and falling "from" nodes. If the "from" collection is a clock collection, the assignment applies to those nodes that are clocked by the respective rising or falling clock edge.

The "-rise_to" and "-fall_to" options behave similarly to the "from" options described previously. These assignments restrict the given assignment to only those nodes or clocks that correspond to the specified rise or fall value taking into consideration any logical inversions that are along the clock path.

The values of the -from, -to, -through, etc. are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Apply a 10ns max delay between two unrelated clocks
set_max_delay -from [get_clocks clkA] -to [get_clocks clkB] 10.000

# Apply a 2ns max delay for an input port (TSU)
set_max_delay -from [get_ports in[*]] -to [get_registers *] 2.000

# Apply a 2ns max delay for an output port (TCO)
set_max_delay -from [get_registers *] -to [get_ports out[*]] 2.000

# Apply a 2ns max delay for an input port to an output port (TPD)
set_max_delay -from [get_ports in[*]] -to [get_ports out[*]] 2.000

# Apply a 2ns max delay for an input port only to nodes clocked by
# the rising edge of clock CLK
set_max_delay -from [get_ports in[*]] -rise_to [get_clocks CLK] 2.000
```

set_min_delay

Usage

```
set_min_delay [-fall_from <names>] [-fall_to <names>] [-from <names>] [-
rise_from <names>] [-rise_to <names>] [-through <names>] [-to <names>]
<value>
```

Options

```
-fall_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-fall_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-rise_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-rise_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
<value>: Time Value
```

Description

Specifies a minimum delay exception for a given path.

The minimum delay is similar to changing the hold relationship (launching clock edge - latching clock edge), except that it can be applied to input or output ports without input or input delays assigned to them. Minimum delays are always relative to any clock network delays (if the source or destination is register) or any input or output delays (if the source or destination is a port). Therefore, input delays and clock latencies are added to the data arrival times. Clock latencies also added to data required times and output delays are subtracted from data required times.

The "-from" and "-to" values are collections of clocks, registers, ports, pins, or cells in the design. If the "-from" or "-to" values are not specified, the collection is converted automatically into [get_keepers *]. It is worth noting that if the counterpart of the unspecified collection is a clock collection, it is more efficient to explicitly specify this collection as a clock collection, but only if the clock collection also generates the desired assignment.

Applying exceptions between clocks applies the exception from all register or ports clocked by the "-from" clock to all registers or ports clocked by the "-to" clock. Also, applying exceptions between a pair of clocks is more efficient than for specific node to node or node to clock paths.

If pin names or collections are used, the "-from" value must be a clock pin and the "-to" value must be any non-clock input pin to a register. Assignments from clock pins or to and from cells applies to all registers in the cell or clocked by the clock pin.

The "-through" values are collections of pins or nets in the design. An exception applied through a node in the design applies only to paths through the specified node.

The "-rise_from" and "-fall_from" options can be used in place of the "-from" destination nodes. The rise or fall value of the option indicates that the "from" nodes are clocked by the rising or falling edge of the clock that feeds this node taking into consideration any logical inversions along the clock path. The "-from" option is the combination of both rising and falling "from" nodes. If the "from" collection is a clock collection, the assignment applies to those nodes that are clocked by the respective rising or falling clock edge.

The "-rise_to" and "-fall_to" options behave similarly to the "from" options described previously. These assignments restrict the given assignment to only those nodes or clocks that correspond to the specified rise or fall value taking into consideration any logical inversions that are along the clock path.

The values for "-from", "-to", and "-through" are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Apply a 0ns min delay between two unrelated clocks
set_min_delay -from [get_clocks clkA] -to [get_clocks clkB] 0.000

# Apply a 0ns min delay for an input port (TH)
set_min_delay -from [get_ports in[*]] -to [get_registers *] -.000

# Apply a 0.5ns min delay for an output port (MIN_TCO)
set_min_delay -from [get_registers *] -to [get_ports out[*]] 0.500

# Apply a 0.5ns min delay for an input port to an output port (MIN_TPD)
set_min_delay -from [get_ports in[*]] -to [get_ports out[*]] 0.500

# Apply a 0.5ns min delay for an input port only to nodes clocked by
# the falling edge of clock CLK
set_max_delay -from [get_ports in[*]] -fall_to [get_clocks CLK] 0.500
```

set_multicycle_path

Usage

```
set_multicycle_path [-end] [-fall_from <names>] [-fall_to <names>] [-from
<names>] [-hold] [-rise_from <names>] [-rise_to <names>] [-setup] [-start]
[-through <names>] [-to <names>] <value>
```

Options

```
-end: Specifies the multicycle relative to the destination clock waveform
(default)
-fall_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-fall_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-hold: Specifies the multicycle value that applies to clock hold or removal
checks
-rise_from <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-rise_to <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
-setup: Specifies the multicycle value that applies to clock setup or
recovery checks (default)
-start: Specifies the multicycle relative to the source clock waveform
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
<value>: Number of clock cycles
```

Description

Specifies a multicycle exception for a given set of paths.

Multicycles can be specified relative to the source clock ("-start") or destination clock ("-end"). This is useful when the source clock and destination clock are operating at different frequencies. For example, if the source clock is twice as fast (half period) as the destination clock, a "-start" multicycle of 2 is usually required.

Hold multicycles ("-hold") are computed relative to setup multicycles ("-setup"). The value of the hold multicycle represents the number clock edges away from the default hold multicycle. The default hold multicycle value is 0.

The "-from" and "-to" values are collections of clocks, registers, ports, pins, or cells in the design. If the "-from" or "-to" values are not specified, the collection is converted automatically into [get_keepers *]. It is worth noting that if the counterpart of the unspecified collection is a clock collection, it is more efficient to explicitly specify this collection as a clock collection but only if the clock collection also generates the desired assignment.

Applying exceptions between clocks applies the exception from all register or ports clocked by the "-from" clock to all registers or ports clocked by the "-to" clock. Also, applying exceptions between a pair of clocks is more efficient than for specific node to node or node to clock paths.

If pin names or collections are used, the "-from" value must be a clock pin and the "-to" value must be any non-clock input pin to a register. Assignments from clock pins or to and from cells apply to all registers in the cell or clocked by the clock pin.

The "-through" values are collections of pins or nets in the design. An exception applied through a node in the design applies only to paths through the specified node.

The "-rise_from" and "-fall_from" options can be used in place of the "-from" destination nodes. The rise or fall value of the option indicates that the "from" nodes are clocked by the rising or falling edge of the clock that feeds this node taking into consideration any logical inversions along the clock path. The "-from" option is the combination of both rising and falling "from" nodes. If the "from" collection is a clock collection, the assignment applies to those nodes that are clocked by the respective rising or falling clock edge.

The "-rise_to" and "-fall_to" options behave similarly to the "from" options described previously. These assignments restrict the given assignment to only those nodes or clocks that correspond to the specified rise or fall value, taking into consideration any logical inversions that are along the clock path.

The values for "-from", "-to", and "-through" are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
create_clock -period 10.000 -name CLK [get_ports clk]
create_generated_clock -divide_by 2 -source [get_ports clk] -name CLKDIV2 \
    [get_registers clkdiv]

# Apply a source multicycle 2 with a hold multicycle of 1 for all
# paths from the CLK domain to the CLKDIV2 domain.
set_multicycle_path -start -setup -from [get_clocks CLK] -to \
    [get_clocks CLKDIV2] 2
set_multicycle_path -start -hold -from [get_clocks CLK] -to \
    [get_clocks CLKDIV2] 1

# Apply a multicycle 3 (with a default hold multicycle of 0) for a
# specific path in the design.
set_multicycle_path -end -setup -from [get_pins rega|clk] -to \
    [get_pins regb|*] 2
```

```
# Apply a multicycle 2 to a given cell, except for the reset pin.
set_multicycle_path -end -setup -to [get_cells regb] 2
set_multicycle_path -end -setup -to [get_pins regb|aclr] 1

#Apply a multicycle 3 rising from a clock and falling to a node
set_multicycle_path -end -setup -rise_from [get_clocks CLK] -fall_to \
    [get_pins regb|datab] 3
```

set_output_delay

Usage

```
set_output_delay [-add_delay] -clock <name> [-clock_fall] [-fall] [-max]
[-min] [-reference_pin <name>] [-rise] [-source_latency_included] <delay>
<targets>
```

Options

```
-add_delay: Add to existing delays instead of overriding them
-clock <name>: Clock name
-clock_fall: Specifies output delay relative to the falling edge of the
clock
-fall: Specifies the falling output delay at the port
-max: Applies value as maximum data required time
-min: Applies value as minimum data required time
-reference_pin <name>: Specifies a pin or port in the design to which the
output delay is relative
-rise: Specifies the rising output delay at the port
-source_latency_included: Specifies input delay including added source
latency
<delay>: Time value
<targets>: Collection or list of output ports
```

Description

Specifies the data required times at the specified output ports relative the clock specified by the "-clock" option. The clock must refer to a clock name in the design.

Output delays can be specified relative to the rising edge (default) or falling edge ("-clock_fall") of the clock.

If the output delay is specified relative to a simple generated clock (a generated clock with a single target), the clock arrival times to the generated clock are added to the data required time.

Output delays can be specified relative to a pin or port ("-reference_pin") in the clock network. Clock arrival times to the reference pin or port are added to the data required time.

Output delays can already include clock source latency. By default the clock source latency of the related clock is added to the output delay value, but when the "-souce_latency_included" option is specified, the clock source latency is not added because it is factored into the output delay value.

The maximum output delay ("-max") is used for clock setup checks or recovery checks and the minimum output delay ("-min") is used for clock hold checks or removal checks. If only "-min" or "-max" (or neither) are specified for a given port, the same value is used for both.

Separate rising ("-rise") and falling ("-fall") required times at the port can be specified. If only one of "-rise" and "-fall" are specified for a given port, the same value is used for both.

By default, set_output_delay will remove any other output delays to the port except for those with the same "-clock," "-clock_fall," and "-reference_pin" combination. Multiple output delays relative to different clocks, clock edges, or reference pins can be specified using the "-add_delay" option.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Simple output delay with the same value for min/max and rise/fall
set_output_delay -clock clk 0.5 [get_ports in*]

# Output delay with respect to the falling edge of clock
set_output_delay -clock clk -clock_fall 0.5 [get_ports in*]

# Output delays for different min/max and irse/fall combinations
set_output_delay -clock clk -max -rise 0.5 [get_ports in*]
set_output_delay -clock clk -max -fall 0.4 [get_ports in*]
set_output_delay -clock clk -min -rise 0.4 [get_ports in*]
set_output_delay -clock clk -min -fall 0.3 [get_ports in*]

# Adding muliple output delays with respect to more than one clock
set_output_delay -clock clkA -min 0.2 [get_ports in*]
set_output_delay -clock clkA -max 0.8 [get_ports in*]
set_output_delay -clock clkA -clock_fall 0.6 [get_ports in*] -add_delay
set_output_delay -clock clkB -min 1.1 [get_ports in*] -add_delay
set_output_delay -clock clkB -max 1.5 [get_ports in*] -add_delay

# Specifying an output delay relative to an external clock output port
set_output_delay -clock clk -reference_pin [get_ports clkout] 0.8 \
    [get_ports in*]
```

sdc_ext

Timing Constraints not defined in the SDC Spec Version 1.5 are implemented in this package. Any command in this package can be specified in a TimeQuest SDC file.

This package is loaded by default in the following executable:

- quartus_sta

This package includes the following commands:

derive_pll_clocks

Usage

```
derive_pll_clocks [-use_tan_name]
```

Options

```
-use_tan_name: Use net names as clock names
```

Description

Identifies PLLs or similar resources in the design and creates generated clocks for their output clock pins. Multiple generated clocks may be created for each output clock pin if the PLL is using clock switchover, one for the inclk[0] input clock pin and one for the inclk[1] input clock pin.

This command does not create base clocks on input clock ports of the design that are driving the PLL.

By default the clock name is the output clock pin name. To use the net name (the name the classic Timing Analyzer would use), use the "-use_tan_name" option.

Example

```
project_open top
create_timing_netlist

# Create the base clock for the input clock port driving the PLL
create_clock -period 10.0 [get_ports sysclk]

# Create the generated clocks for the PLL.
derive_pll_clocks

update_timing_netlist

# Other user actions
report_timing

delete_timing_netlist
project_close
```

get_assignment_groups

Usage

```
get_assignment_groups [-keepers] [-ports] [-registers] <name>
```

Options

```
-keepers: Returns a keeper collection from the assignment group matching
the <name>
-ports: Returns a port collection from the assignment group matching the
<name>
-registers: Returns a register collection from the assignment group
matching the <name>
<name>: Assignment group name
```

Description

Returns a collection of <keepers>|<registers>|<ports> for the assignment group that matches <name>. This command can be used to retrieve the assignment group created and saved in the Quartus II Settings File.

The options -keepers, -registers and -ports are mutually exclusive. If no option is specified, the keeper collection is returned by default.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See help for the use_timequest_style_escaping command for details.

Example

```
get_assignment_groups my_assignments -registers
```

get_fanins

Usage

```
get_fanins [-asynch] [-clock] [-inverting_paths] [-no_logic] [-
non_inverting_paths] [-synch] <filter>
```

Options

```
-asynch: Traverse through asynch edges
-clock: Traverse through clock edges
-inverting_paths: Only follow inverting combinational paths
-no_logic: Do not follow combinational paths
-non_inverting_paths: Only follow non-inverting combinational paths
-synch: Traverse through synch edges
<filter>: Valid starting nodes (string patterns are matched using Tcl
string matching)
```

Description

Returns a collection of fanin nodes starting from the <filter> in the design. When you supply the -no_logic option, get_fanins ignores the paths that pass through combinational logic elements other than buffers and inverters.

When you use the -synch, -asynch or -clock options, get_fanins traverses the netlist through corresponding edges. More than one of these options can be specified. If you do not specify any of these three options, the command does not ignore any paths.

When the -non_inverting_paths option is used, no_logic does not follow any paths that includes odd number of inverters. Similarly, when the -inverting_paths option is used, no_logic does not follow any paths that includes even number of inverters. Both the -non_inverting_paths and -inverting_paths options require the -no_logic option and are mutually exclusive.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for the use_timequest_style_escaping command for details.

Example

```
set fanins [get_fanins $item -synch -clock]
foreach_in_collection fanin_keeper $fanins {
   lappend fanin_keeper_list [get_node_info $fanin_keeper -name]
}

set fanins_no_logic [get_fanins $item -no_logic -asynch]
foreach_in_collection fanin_keeper $fanins_no_logic {
   lappend fanin_keeper_list_no_logic [get_node_info $fanin_keeper \
      -name]
}
```

get_fanouts

Usage

```
get_fanouts [-inverting_paths] [-no_logic] [-non_inverting_paths] [-
through <names>] <filter>
```

Options

```
-inverting_paths: Only follow inverting combinational paths
-no_logic: Do not follow combinational paths
-non_inverting_paths: Only follow non-inverting combinational paths
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
<filter>: Valid starting nodes (string patterns are matched using Tcl
string matching)
```

Description

Returns a collection of fanout nodes starting from the <filter> in the design. When the -no_logic option is used, get_fanouts ignores the paths that pass through combinational logic elements other than buffers and inverters.

When the -non_inverting_paths option is used, no_logic does not follow any paths that includes odd number of inverters. Similarly, when the -inverting_paths option is used, no_logic does not follow any paths that includes even number of inverters. Both -non_inverting_paths and -inverting_paths options require the -no_logic option and are mutually exclusive.

When the -through option is used, only the fanouts that can be reached by going through those nodes are returned.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
set fanouts [get_fanouts $item]
foreach_in_collection fanout_keeper $fanouts {
   lappend fanout_keeper_list [get_node_info $fanout_keeper -name]
}

set fanouts_no_logic [get_fanouts $item -no_logic]
foreach_in_collection fanout_keeper $fanouts_no_logic {
   lappend fanout_keeper_list_no_logic \
       [get_node_info $fanout_keeper -name]
}
```

```
# Using through option to find the fanout registers whose enable input is
# connected to the signal while ignoring the inverting paths.
get_fanouts inst1 -no_logic -non_inverting_paths -through \
    [get_pins -hierarchical *|ena]
```

get_keepers

Usage

```
get_keepers [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of keeper nodes (non-combinational nodes) in the design.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See help for the use_timequest_style_escaping command for details.

Example

```
project_open chiptrip
create_timimg_netlist

set kprs [get_keepers *reg*]
foreach_in_collection kpr $kprs {
    puts [get_object_info -name $kpr]
}

delete_timing_netlist
project_close
```

get_nodes

Usage

```
get_nodes [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of nodes in the design.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See help for the use_timequest_style_escaping command for details.

Example

```
project_open chiptrip
create_timimg_netlist

set nodes [get_nodes *name*]
foreach_in_collection node $nodes {
    puts [get_object_info -name $node]
}

delete_timing_netlist
project_close
```

get_partitions

Usage

```
get_partitions [-cell] [-hierarchical] [-nocase] <filter>
```

Options

```
-cell: Returns a cell collection inside the partitions matching the
<filter>
-hierarchical: Specifies if hierarchical searching method should be used
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid partitions (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of partitions matching the filter by default. All partition names in the collection match the specified pattern. Wildcards can be used to select multiple partitions at once.

The -cell option creates and returns the collection of cells found inside the partitions matching the <filter> instead of returning a partition collection.

There are three Tcl string matching schemes available with this command: default, -hierarchical, -no_case.

When using the default matching scheme, pipe characters separate one hierarchy level from the next. They are treated as special characters and are taken into account when string matching with wildcards is performed. The default matching scheme does not force the search to proceed recursively down the hierarchy.

When using the hierarchical matching scheme, pipe characters are treated as special characters and are taken into account when string matching with wildcards is performed. This matching scheme forces the search to proceed recursively down the hierarchy.

The -nocase matching scheme uses case-insensitive matching behavior.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
#Get the partitions matching the filter
get_partitions *

#Get the collection of cells inside partitions matching the filter
get_partitions * -cell
```

get_registers

Usage

```
get_registers [-nocase] <filter>
```

Options

```
-nocase: Specifies the matching of node names to be case-insensitive
<filter>: Valid destinations (string patterns are matched using Tcl string
matching)
```

Description

Returns a collection of registers in the design.

The filter for the collection is a Tcl list of wildcards, and must follow standard Tcl or TimeQuest-extension substitution rules. See help for the use_timequest_style_escaping command for details.

Example

```
project_open chiptrip
create_timimg_netlist

set regs [get_registers *reg*]
foreach_in_collection reg $regs {
    puts [get_object_info -name $reg]
}

delete_timing_netlist
project_close
```

remove_clock

Usage

```
remove_clock [-all] <clock_list>
```

Options

```
-all: Removes all clocks from the design
<clock_list>: Clock(s) to be removed
```

Description

Removes the specified clock(s) from the design.

Example

```
# Create a clock and then remove it.
create_clock -period 10 -name CLK [get_ports clk]
remove_clock CLK
```

set_scc_mode

Usage

```
set_scc_mode [-size <size>] [-use_heuristic]
```

Options

```
-size <size>: Maximum SCC loop size
-use_heuristic: Always use heuristic for SCC processing
```

Description

Allows you to set maximum Strongly Connected Components (SCC) loop size or force TimeQuest Timing Analyzer to always estimate delays through SCCs.

When TimeQuest Timing Analyzer encounters a loop of size greater than the specified maximum SCC loop size, it uses a heuristic which only estimates delays through the loop.

If the loop is smaller than the maximum SCC loop size, a full processing of loops is performed unless the -use_heuristic option is used.

Example

```
# Make TimeQuest Timing Analyzer use normal processing for all loops
# the size of which is less than or equal to 100. For loops of size
# greater than 100, a runtime-saving heuristic will be used
set_scc_mode -size 100

# Force TimeQuest Timing Analyzer to use heuristic for all SCCs
# disregarding their size
set_scc_mode -use_heuristic
```

set_time_format

Usage

```
set_time_format [-decimal_places <decimal_places>] [-unit <unit>]
```

Options

```
-decimal_places <decimal_places>: Number of decimal places to use
-unit <unit>: Default time unit to use
```

Description

Sets time format, including time unit and decimal places.

Time units are assumed to be nanoseconds (ns) by default. The "-unit" option overrides the default time units. Legal time unit values are: ps, ns, us, ms.

Time units are displayed with three decimal places by default. The "-decimal_places" option overrides the default number of decimal places to show.

The smallest resolution of all times units is one picosecond (ps). Any additional specified precision will be truncated.

Example

```
# Create two clocks with a clock period of 8 nanoseconds.
create_clock -period 8.000 clk1

set_time_format -unit ps -decimal_places 0
create_clock -period 8000 clk2
```

sta

This package contains the set of Tcl functions for obtaining advanced information from the TimeQuest Timing Analyzer.

This package is loaded by default in the following executable:

- quartus_sta

This package includes the following commands:

check_timing

Usage

```
check_timing [-append] [-fall_from_clock <names>] [-fall_to_clock <names>]
[-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-include
<check_list>] [-less_than_slack <slack limit>] [-npaths <number>] [-off]
[-on] [-panel_name <name>] [-recovery] [-removal] [-rise_from_clock
<names>] [-rise_to_clock <names>] [-setup] [-stdout] [-through <names>] [-
to <names>] [-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-include <check_list>: Checks to perform
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Checks for problems with design or constraints on the design. (Must run update_timing_netlist before running check_timing). Based on user-specified variables and options, a series of different checks are executed. There is no default list of checks. Simply use the -include option to specify which checks to perform.

- no_clock - Checks that registers have at least one clock at their clock pin, and that ports determined to be clocks have a clock assigned to them. Also checks that PLLs have a clock assignment.
- multiple_clock Checks that registers have at most one clock at their clock pin. When multiple clocks reach a register's clock pin, it is undefined which clock will be used for analysis.
- generated_clock - Checks that generated clocks are valid. Generated clocks must have a source that is clocked by a valid clock.
- no_input_delay - Checks that every input port that is not determined to be a clock has an input delay set on it.
- no_output_delay - Checks that every output port output delay set on it.
- partial_input_delay - Checks that input delays are complete. Makes sure that input delays have a rise-min, fall-min, rise-max, and fall-max portion set.
- partial_output_delay - Checks that output delays are complete. Makes sure that output delays have a rise-min, fall-min, rise-max, and fall-max portion set.
- reference_pin - Checks if reference pins specified in set_input_delay and set_output_delay using the -reference_pin option are valid. A reference_pin is valid if the -clock option specified in the same set_input_delay/set_output_delay command matches the clock that is in the direct fanin of the reference_pin. Being in the direct fanin of the reference_pin means that there must be no keepers between the clock and the reference_pin.
- latency_override - Checks if clock latency set on a port or pin overrides the more generic clock latency set on a clock. Clock latency can be set on a clock, where the latency applies to all keepers clocked by the clock, whereas clock latency can also be set on a port or pin, where the latency applies to registers in the fanout of the port or pin.
- loops - Checks that there are no strongly connected components in the netlist. These loops prevent a design from being properly analyzed. Lets user know that the loops exist but were marked so that they would not be traversed.
- latches - Checks if there are latches in the design. TimeQuest cannot properly analyze latches. Warns user that the latches exist and cannot be properly analyzed.

Example

```
# Constrain design
create_clock -name clk -period 4.000 -waveform { 0.000 2.000 } \
    [get_ports clk]
set_input_delay -clock clk2 1.5 [get_ports in*]
set_output_delay -clock clk 1.6 [get_ports out*]
set_false_path -from [get_keepers in] -through [get_nets r1] -to \
    [get_keepers out]

# Check if there were any problems
check_timing -include {loops latches no_input_delay partial_input_delay}
```

create_slack_histogram

Usage

```
create_slack_histogram [-append] -clock_name <name> [-fall_from_clock
<names>] [-fall_to_clock <names>] [-file <name>] [-from <names>] [-
from_clock <names>] [-hold] [-less_than_slack <slack limit>] [-max_slack
<max_slack>] [-min_slack <min_slack>] [-npaths <number>] [-num_bins
<num_bins>] [-off] [-on] [-panel_name <name>] [-recovery] [-removal] [-
rise_from_clock <names>] [-rise_to_clock <names>] [-setup] [-stdout] [-
through <names>] [-to <names>] [-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-clock_name <name>: Name of the Clock Domain
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Hold Analysis
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-max_slack <max_slack>: Maximum slack value of the created histogram
-min_slack <min_slack>: Minimum slack value of the created histogram
-npaths <number>: Specifies the number of paths to report (default=1)
-num_bins <num_bins>: Number of bins
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Recovery Analysis
-removal: Removal Analysis
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Setup Analysis
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
```

```
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Creates a slack histogram in the timing report for the specified clock domain "-clock_name," showing the number of timing edges within various ranges of slacks for a clock setup analysis. The histogram can be named using the "-panel_name" option.

Use the "-setup", "-hold", "-recovery", or "-removal" options to specify which kind of analysis should be performed. If none is specified, setup analysis is used by default.

Reports can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical interface ("-panel_name"), or any combination of the three.

The range of reported slack values can be controlled by specifying the "-min_slack" and "-max_slack" options. The number of bins (histogram bars) can also be specified using the "-num_bins" option.

Example

```
project_open top
create_timing_netlist
read_sdc
update_timing_netlist

# Create a slack histogram for clk1, defaulting to
# the name "Slack Histogram (clk1)"
create_slack_histogram -clock_name clk1

# Create a slack histogram for clk2 named "MyHistogram"
create_slack_histogram -clock_name clk2 -panel_name MyHistogram

delete_timing_netlist
project_close
```

create_timing_netlist

Usage

```
create_timing_netlist [-cmp_report] [-fast_model] [-force_dat] [-no_latch]
[-post_map] [-speed <speed>] [-zero_ic_delays]
```

Options

```
-cmp_report: Option to add link from compiler report
-fast_model: Option to use fast timing model
-force_dat: Option to force delay annotation
-no_latch: Option to disable the analysis of latches as synchronous
elements
-post_map: Option to perform timing analysis on post-synthesis netlist
-speed <speed>: Speed grade
-zero_ic_delays: Option to set all IC delays to zero
```

Description

Creates the timing netlist by annotating the atom netlist with delay information using post-fitting results. Use the "-post_map" option to obtain post-synthesis results.

By default, delay annotation is skipped. Use "-force_dat" to rerun delay annotation. This is required if any delay annotation setting is changed in the Quartus II project revision (e.g. OUTPUT_PIN_LOAD).

Use the "-fast_model" option to run the analysis using the fast corner delay models.

Use "-no_latch" to analyze latches as combinational loops instead of synchronous elements.

Use "-zero_ic_delays" to set all IC delays in the netlist to zero.

Example

```
project_open my_top

# Create timing netlist before calling
# any report functions
create_timing_netlist

# Read SDC and update timing
read_sdc
update_timing_netlist

# Ready to call report functions
report_timing -npaths 1 -clock_setup

# The following command is optional
delete_timing_netlist
```

```
project_close

project_open my_top

# Report worst case period for -9 speed grade
create_timing_netlist -speed 9

# Read SDC and update timing
read_sdc
update_timing_netlist

report_timing -clock_setup -clock_filter clk
delete_timing_netlist

# Report hold violation for fastest corner
create_timing_netlist -fast_model

# Read SDC and update timing
read_sdc
update_timing_netlist

report_timing -clock_hold -clock_filter clk
delete_timing_netlist

# If Delay Annotation has been run for the fast corner
# Force Delay Annotation
create_timing_netlist -fast_model -force_dat

# Read SDC and update timing
read_sdc
update_timing_netlist

report_timing -clock_hold -clock_filter clk
delete_timing_netlist

# Report worst case period for post-technology mapping netlist
create_timing_netlist -post_map

# Read SDC and update timing
read_sdc
update_timing_netlist

report_timing -clock_setup -clock_filter clk
delete_timing_netlist

project_close
```

create_timing_summary

Usage

```
create_timing_summary [-append] [-fall_from_clock <names>] [-fall_to_clock
<names>] [-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-through <names>] [-to <names>] [-to_clock
<names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Hold Analysis
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Recovery Analysis
-removal: Removal Analysis
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Setup Analysis
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports the worst-case Clock Setup and Clock Hold slacks and endpoint TNS (total negative slack) per clock domain. Total negative slack is the sum of all slacks less than zero for each destination register or port in the clock domain.

This command generates the most important report as it shows the worst-case slack for each clock domain. You right click in these reports to run more detailed reports like Histograms and Report Timing

By default, this command creates a Setup Summary. This command can also generate a Hold Summary ("-hold"), Recovery Summary ("-recovery"), or Removal Summary ("-removal").

Report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical interface ("-panel_name"), or any combination of the three.

Example

```
project_open my_project

# Always create the netlist first and process constraints
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Create Clock Domain Summary
create_timing_summary -panel_name "Setup Summary"
create_timing_summary -hold -panel_name "Hold Summary"

# The following command is optional
delete_timing_netlist

project_close
```

delete_timing_netlist

Usage

```
delete_timing_netlist
```

Options

None

Description

Deletes the timing netlist.

Use this command to delete a timing netlist previously created using create_timing_netlist. This should be done at the end of a script or before calling create_timing_netlist again using different options or after recompiling the design.

enable_sdc_extension_collections

Usage

```
enable_sdc_extension_collections [-off] [-on]
```

Options

```
-off: Disable this setting.
-on: Enable this setting.
```

Description

Enable the support of SDC extension collections, such as keeper, register and node collections. When enable_sdc_extension_collections is not used, using these collections causes an error. Default to -on option.

Example

```
project_open top
enable_sdc_extension_collections -on
create_timing_netlist
read_sdc

report_timing -to_clock clk1

delete_timing_netlist
project_close
```

get_cell_info

Usage

```
get_cell_info [-buried_nodes] [-buried_regs] [-in_pin_names] [-in_pins] [-
location] [-name] [-out_pin_names] [-out_pins] [-pin_names] [-pins] [-
type] [-wysiwyg_type] <cell_id>
```

Options

```
-buried_nodes: Return a collection of buried node IDs
-buried_regs: Return a collection of buried register IDs
-in_pin_names: Return a list of input pin names
-in_pins: Return a collection of input pin IDs
-location: Return the atom location in device
-name: Return the cell name
-out_pin_names: Return a list of output pin names
-out_pins: Return a collection of output pin IDs
-pin_names: Return a list of input and output pin names
-pins: Return a collection of input and output pin IDs
-type: Return the cell type
-wysiwyg_type: Return the WYSIWYG type of the cell
<cell_id>: Cell ID
```

Description

Gets information about the specified cell (referenced by cell ID). Cell ID's can be obtained by Tcl commands such as get_cells. The -type option returns "cell". Options -name, -type, -pin_name, -in_pin_names, -out_pin_names, -pins, -clock_pins, -in_pins, -out_pins, -buried_nodes, -buried_regs, -location and -wysiwyg_type are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set cells [get_cells]
foreach_in_collection cell $cells {
    puts "[get_cell_info $cell -name]: [get_cell_info $cell -type]"
}
delete_timing_netlist
project_close
```

get_clock_domain_info

Usage

```
get_clock_domain_info [-hold] [-recovery] [-removal] [-setup]
```

Options

```
-hold: Hold Analysis
-recovery: Recovery Analysis
-removal: Removal Analysis
-setup: Setup Analysis (Default)
```

Description

Similar to create_timing_summary, get_clock_domain_info returns a Tcl list of information about each clock domain. Each entry in the list is a list of four elements: the clock name, worst-case slack, endpoint TNS, and edge TNS. TNS stands for total negative slack, and it is the sum of all slacks less than zero for either destination registers or ports in the clock domain (endpoint TNS) or for all edges affecting the clock domain (edge TNS).

This command can generate a Setup Summary ("-setup"), Hold Summary ("-hold"), Recovery Summary ("-recovery"), or Removal Summary ("-removal").

Example

```
project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Get domain summary object
set domain_list [get_clock_domain_info]
foreach domain $domain_list {
    set name [lindex $domain 0]
    set slack [lindex $domain 1]
    set keeper_tns [lindex $domain 2]
    set edge_tns [lindex $domain 3]

    puts "Clock $name : Slack = $slack , TNS = ( $keeper_tns , $edge_tns \
        )"
}

# The following command is optional
delete_timing_netlist

project_close
```

get_clock_fmax_info

Usage

```
get_clock_fmax_info
```

Options

None

Description

Computes potential FMAX for every clock in the design, regardless of the user-specified clock periods. FMAX is only computed for paths where the source and destination registers or ports are driven by the same clock. Paths of different clocks, including generated clocks, are ignored. For paths between a clock and its inversion, FMAX is computed as if the rising and falling edges are scaled along with FMAX, such that the duty cycle (in terms of a percentage) is maintained.

Example

```
project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Get domain summary object
set domain_list [get_clock_fmax_info]
foreach domain $domain_list {
    set name [lindex $domain 0]
    set fmax [lindex $domain 1]

    puts "Clock $name : FMAX = $fmax"
}

# The following command is optional
delete_timing_netlist

project_close
```

get_clock_info

Usage

```
get_clock_info [-divide_by] [-duty_cycle] [-edge_shifts] [-edges] [-fall]
[-is_inverted] [-latency] [-master_clock] [-master_clock_pin] [-max] [-
min] [-multiply_by] [-name] [-offset] [-period] [-phase] [-rise] [-targets]
[-type] [-waveform] <clk_id>
```

Options

```
-divide_by: Return the frequency divider (to the base clock)
-duty_cycle: Return the duty cycle
-edge_shifts: Return a list of edge shifts that the specified edges are to
undergo to yield the final generated clock waveform
-edges: Return a list of integer representing edges from the source clock
that are to form edges of the generated clock
-fall: Return clock fall latency
-is_inverted: Return a boolean value to indicate if the generated clock is
inverted
-latency: Return clock latency
-master_clock: Return the master clock name
-master_clock_pin: Return the master clock source pin
-max: Return max clock latency
-min: Return min clock latency
-multiply_by: Return the frequency multiplier (to the base clock)
-name: Return the clock name
-offset: Return the clock offset
-period: Return the clock period
-phase: Return the clock phase
-rise: Return clock rise latency
-targets: Return the clock targets collection
-type: Return the clock type
-waveform: Return the waveform (rise time and fall time)
<clk_id>: Clock ID
```

Description

Gets information about the specified clock (referenced by clock ID). Clock IDs can be obtained by Tcl commands such as get_clocks. The -type option returns "clk". Options -name, -type, -period, -duty_cycle, -waveform, -edges, -edge_shifts, -multiply_by, -divide_by, -is_inverted, -latency, -master_clock, and -targets are mutually exclusive. The -latency option requires a specified -max or -min option as well as a -rise or -fall option.

Example

```
project_open chiptrip
create_timing_netlist
set clocks [get_clocks]
foreach_in_collection clk $clocks {
    puts "[get_clock_info $clk -name]: [get_clock_info $clk -period]"
}
delete_timing_netlist
project_close
```

get_datasheet

Usage

```
get_datasheet
```

Options

None

Description

This function returns a tcl collection which contains the datasheet report. Its format is as follows:

```
{
  { tsu,
    { <tsu rise time>,
      <tsu rise time>,
      <input port>,
      <clock>
    }
  }

  { th,
    { <th rise time>,
      <th rise time>,
      <input port>,
      <clock>
    }
  }

  { tco,
    { <tco rise time>,
      <tco rise time>,
      <output port>,
      <clock>
    }
  }

  { mintco,
    { <mintco rise time>,
      <mintco rise time>,
      <output port>,
      <clock>
    }
  }

  { tpd,
    { <tpd rise time>,
      <tpd rise time>,
      <input port>,
      <output port>
    }
  }

  { mintpd,
    { <mintpd rise time>,
      <mintpd rise time>,
      <input port>,
      <output port>
    }
  }
}
```

There are no options for this command, and the data outputed is the same as in the report_datasheet command.

Example

```
project_open proj1
create_timing_netlist
read_sdc
update_timing_netlist

# get the datasheet collection
set datasheet [get_datasheet]

# loop through contents of datasheet collection
foreach i $datasheet {
    foreach j $i {
        foreach k $j {
            #
            # extract individual items or
            # manipulate as necessary
            #
        }
    }
}
```

get_default_sdc_file_names

Usage

```
get_default_sdc_file_names
```

Options

None

Description

Returns the default SDC file name(s) used by the Quartus II Compiler when doing timing-driven optimizations.

Returns the value for the QSF variable SDC_FILE. If multiple assignments are found, return them as a list If not specified, return <revision_name>.sdc.

Example

```
project_new test
create_timing_netlist
foreach file [get_default_sdc_file_names] {
   read_sdc $file
}
update_timing_netlist

report_timing

delete_timing_netlist
project_close
```

get_delay_info

Usage

```
get_delay_info [-cell] [-edge <Edge ID>] [-ff] [-fr] [-from <names>] [-ic]
[-max] [-min] [-rf] [-rr] [-to <names>] [-unateness] [-value]
```

Options

```
-cell: Specifies that cell delay must be set
-edge <Edge ID>: Valid edge ID
-ff: FF transition
-fr: FR transition
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-ic: Specifies that ic delay must be set
-max: Maximum value
-min: Minimum value
-rf: RF transition
-rr: RR transition
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-unateness: Type of information
-value: Type of information
```

Description

Returns the requested type of information (value or unateness) for the specified edge.

If -unateness option is specified, then only the component (-ic/-cell) is required. If -value option is specified, the component option is required, and transition can be specified to get a specific delay. If transition is not given, worst case value is returned.

-value and -unateness are mutually exclusive, and one of them must be specified. -max and -min are mutually exclusive.

If no -from/-to value is specified, the missing value is substituted by a "*".

The values of the -from and -to are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open <design>
create_timing_netlist

# Print the maximum RR delay of IC component for every edge
foreach_in_collection edge [get_timing_edges] {
    set max_ic_rr [get_delay_info -edge $edge -value -ic -rr]
    set unateness [get_delay_info -edge $edge -unateness -ic]
    puts "Max IC RR delay is $max_ic_rr"
    puts "Unateness is $unateness"
}

project_close
```

get_edge_info

Usage

```
get_edge_info [-delay] [-delay_type] [-dst] [-ff] [-fr] [-max] [-min] [-
name] [-rf] [-rr] [-src] [-type] [-unateness] <edge_id>
```

Options

```
-delay: Return the delay.
-delay_type: Return the type of the delay (ic/cell).
-dst: Return the destination node ID.
-ff: Return the fall-to-fall delay
-fr: Return the fall-to-rise delay
-max: Max delay
-min: Min delay
-name: Return the edge name
-rf: Return the rise-to-fall delay
-rr: Return the rise-to-rise delay
-src: Return the source node ID
-type: Return the edge type.
-unateness: Return the unateness.
<edge_id>: Edge ID
```

Description

Gets information about the specified edge (referenced by edge ID). Edge ID's can be obtained by Tcl commands such as get_node <node_id> -synch_edges.

- -type returns <edge>.
- -delay option returns the delay associated to the edge. Use -max/min and -rr/rf/fr/ff options to specify the type of returned delay. One of the -max/min options must be specified. One of the -rr/rf/fr/ffoptions must be specified.
- -unateness option returns the unateness associated to the edge.

Example

```
project_open chiptrip
create_timing_netlist
set nodes [get_nodes Reg*]
foreach_in_collection node $nodes {
   set edges [get_node_info $node -fanout_edges]
   foreach_in_collection edge $edges {
   set rr_delay [get_edge_info $edge -delay -rr]
   set rf_delay [get_edge_info $edge -delay -rf]
   set fr_delay [get_edge_info $edge -delay -fr]
   set ff_delay [get_edge_info $edge -delay -ff]
   puts "Total cell delay of edge $edge: $rr_delay $rf_delay \
        $fr_delay $ff_delay"
    }
}
delete_timing_netlist
project_close
```

get_net_info

Usage

```
get_net_info [-name] [-pin] [-type] <net_id>
```

Options

```
-name: Return the net name
-pin: Return the pin ID of this net
-type: Return the net type.
<net_id>: Net ID
```

Description

Gets information about the specified net (referenced by net ID). Net ID's can be obtained by Tcl commands such as get_nets. The -type option returns "net". Options -name, -type and -pin are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist

set nets [get_nets]
foreach_in_collection net $nets {
   puts [get_net_info $net -name]
}

delete_timing_netlist
project_close
```

get_node_info

Usage

```
get_node_info [-asynch_edges] [-cell] [-clock_edges] [-fanout_edges] [-
location] [-name] [-synch_edges] [-type] <node_id>
```

Options

```
-asynch_edges: Return a list of asynchronous edge IDs
-cell: Return the host cell
-clock_edges: Return a list of clock edge IDs
-fanout_edges: Return a list of fanout edge IDs
-location: Return the atom location in device
-name: Return the node name
-synch_edges: Return a list of synchronous edge IDs
-type: Return the node type
<node_id>: Node ID
```

Description

Gets information about the specified node (referenced by node ID). Use Tcl commands such as get_nodes to obtain node IDs. The -type option returns "reg", "port", "pin", or "comb". Options -name, -type, -clock_edges, -synch_edges, -asynch_edges, -fanout_edges, -cell and -location are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set registers [get_registers]
foreach_in_collection reg $registers {
    puts "[get_node_info $reg -name]: [get_node_info $reg -type]"
}
delete_timing_netlist
project_close
```

get_object_info

Usage

```
get_object_info [-name] [-type] <obj_id>
```

Options

```
-name: Return the object name
-type: Return the object type
<obj_id>: Object ID
```

Description

Gets information about the specified object (referenced by object ID). Object IDs can be obtained by Tcl commands such as get_clocks, get_ports, get_cells, etc. The -type option returns "clk", "reg", "port", "cell", "pin", "comb", "net", or "edge". Options -name and -type are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set ports [get_ports]
foreach_in_collection port $ports {
    puts [get_object_info $port -name]
}
delete_timing_netlist
project_close
```

get_partition_info

Usage

```
get_partition_info [-child] [-name] [-parent] [-type] <partition_id>
```

Options

```
-child: Return child partition name(s)
-name: Return the partition name
-parent: Return parent partition name
-type: Return the partition type
<partition_id>: Partition ID
```

Description

Gets information about the specified partition (referenced by partition ID). Partition ID's can be obtained by Tcl commands such as get_partitions.

Options -name, -type, -parent, -child are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set partitions [get_partitions *]
foreach_in_collection partition $partitions {
    puts "[get_partition_info $partition -name]"
}
delete_timing_netlist
project_close
```

get_path_count

Usage

```
get_path_count [-append] [-fall_from_clock <names>] [-fall_to_clock
<names>] [-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-through <names>] [-to <names>] [-to_clock
<names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

If -from and -to options are used, this command calculates the total number of paths that a signal traverses in order to reach destination (-to) from the source (-from). The "-through" option can be used to restrict the path count analysis only to paths which go through specified through nodes.

Example

```
get_path_count -from inst3 -to inst7 -through inst21
get_path_count -edge inst2|combout
```

get_path_info

Usage

```
get_path_info [-arrival_points] [-from] [-from_clock] [-
from_clock_is_inverted] [-hold_end_multicycle] [-hold_start_multicycle] [-
latch_time] [-launch_time] [-required_points] [-setup_end_multicycle] [-
setup_start_multicycle] [-slack] [-to] [-to_clock] [-to_clock_is_inverted]
<path_ref>
```

Options

```
-arrival_points: Return a collection of point objects for the arrival path
-from: Return the source node ID
-from_clock: Return the source clock ID
-from_clock_is_inverted: Return 1 if the source clock is inverted, 0
otherwise
-hold_end_multicycle: Return the hold end multicycle for the path
-hold_start_multicycle: Return the hold start multicycle for the path
-latch_time: Return the latch time for the path
-launch_time: Return the launch time for the path
-required_points: Return a collection of point objects for the required
path
-setup_end_multicycle: Return the setup end multicycle for the path
-setup_start_multicycle: Return the setup start multicycle for the path
-slack: Return the slack for the path
-to: Return the destination node ID
-to_clock: Return the destination clock ID
-to_clock_is_inverted: Return 1 if the destination clock is inverted, 0
otherwise
<path_ref>: Path object
```

Description

Get information about the referenced timing path object. References to path objects can be generated using the get_timing_paths function.

The -from and -to options return the ID of the nodes at the start and end, respecitvely, of the arrival path. If there is no node, an empty string is returned. The "to" node remains the same, regardless of the level of clock detail provided (that is, it is always the first node clocked by the "from" clock in the data arrival path). The node ID may be used with the get_node_info function to obtain additional informaion about the node.

The -from_clock and -to_clock options return the ID of the launching and latching clocks, respectively. If there is no clock, an empty string is returned. Additional information on the clocks can be obtained using the get_clock_info function.

The -arrival_points and -required_points options return a collection of point objects for the arrival and required paths, respecitvely. By iterating over the collection, and using the get_point_info function, the specific details of each portion of the path can be obtained.

If a path was created with additional clock detail, theelements of the clock path will be included in each collection of points.

The values of the -from, -to, etc. are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
# Define a few helper procedures to print out points
# on a path, and the path itself
proc get_clock_string { path clk } {
   set clk_str ""
   set clk_id [ get_path_info $path -${clk}_clock ]

   if { $clk_id ne "" } {
   set clk_str [ get_clock_info $clk_id -name ]

 if { [ get_path_info $path -${clk}_clock_is_inverted ] } {
   append clk_str " (INVERTED)"
   }
   }

   return $clk_str
}

proc print_point { point } {
   set total     [ get_point_info $point -total     ]
   set incr      [ get_point_info $point -incr      ]
   set node_id   [ get_point_info $point -node      ]
   set type      [ get_point_info $point -type      ]
   set rf        [ get_point_info $point -rise_fall]
   set node_name ""

   if { $node_id ne "" } {
   set node_name [ get_node_info $node_id -name ]
   }

   puts \
       [format "%10s %8s %2s %-6s %s" $total $incr $rf $type $node_name ]
}

proc print_path { path } {
   puts "Slack      : [ get_path_info $path -slack]"
   puts "To Clock   : [ get_clock_string $path to ]"
   puts "From Clock : [ get_clock_string $path from]"
   puts ""
   puts \
       [format "%10s %8s %-2s %-6s %s" "Total" "Incr" "RF" "Type" "Name"]
   puts \
"=================================================================="

   foreach_in_collection pt [ get_path_info $path -arrival_points ] {
   print_point $pt
   }
}

project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# And now simply iterate over the 10 worst setup paths, printing each
# path
foreach_in_collection path [ get_timing_paths -npaths 10 -setup ] {
   print_path $path
   puts ""
}

delete_timing_netlist
project_close
```

get_pin_info

Usage

```
get_pin_info [-is_clock_pin] [-is_in_pin] [-is_out_pin] [-name] [-net] [-
parent_cell] [-suffix] [-type] <pin_id>
```

Options

```
-is_clock_pin: Return true if it is a clock pin, or false otherwise
-is_in_pin: Return true if it is an input pin, or false otherwise
-is_out_pin: Return true if it is an output pin, or false otherwise
-name: Return the pin name
-net: Return the net ID if this is an output pin
-parent_cell: Return the parent cell ID
-suffix: Return the suffix of the pin
-type: Return the pin type
<pin_id>: Pin ID
```

Description

Gets information about the specified pin (referenced by pin ID). Pin ID's can be obtained by Tcl commands such as get_pins. The -type option returns "pin". Options -name, -type, -parent_cell, -net, -suffix, -is_clk_pin, -is_in_pin and -is_out_pin are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set pins [get_pins]
foreach_in_collection pin $pins {
    set pin_name [get_pin_info $pin -name]
    set parent_cell [get_pin_info $pin -parent_cell]
     puts "Pin $pin_name belongs to cell [get_cell_info -name \
         $parent_cell]"
}
delete_timing_netlist
project_close
```

get_point_info

Usage

```
get_point_info [-incremental_delay] [-location] [-node] [-
number_of_fanout] [-rise_fall] [-total_delay] [-type] <point_ref>
```

Options

```
-incremental_delay: Return the incremental delay through this point
-location: Return a string indicating the location of the point's node, if
there is one, else an empty string
-node: Return the node ID for the node associated with this point.  If the
point has no node, this returns an empty string
-number_of_fanout: Return the number of fanout that this point has in the
netlist
-rise_fall: Return a string indicating the rise_fall type of this point.
Return values are r, f, rr, rf, fr, ff, or an empty string for undefined
-total_delay: Return the total delay of the path at this point.  This
includes the incremental delay for the point itself
-type: Return a string indicating the type of the point
<point_ref>: Point object
```

Description

Get information about the referenced timing point object. References to path objects can be generated using the get_path_info function.

A point object is the equivalent of a row in a path in the output from report_timing.

The -node option will return a node ID for the corrsponding node in the path. For points that do not have a corresponding node (such as points for the lumped clock network delay, launch time, latch time, etc.), the node ID will be an empty string. A non-empty node ID can be used in conjunction with the get_node_info function to obtain additional information about the node.

The -total_delay option returns the total delay along the path, up to and including the current point. The -incremental_delay option returns the delay incurred by going through this point in the path. Both delays are formated in terms of the current time units, excluding the unit string.

The -number_of_fanout option will return the number of fanouts that the corresponding node has in the timing netlist. If there is no node for this point, the return value will be 0.

The -location option returns a string indicating the location of the corresponding node in the part. If there is no corresponding node, this will return an empty string.

The -rise_fall option returns the transition type of this point.

Possible values for -rise_fall are:

Value	Description
(empty)	Unknown transition
r	Rising output
f	Falling output
rr	Rising input, rising output
rf	Rising input, falling output
fr	Falling input, rising output
ff	Falling input, falling output

The -type option returns a string indicating the type of delay that this point represents in the path.

Possible return values for -type are:

Value	Description
cell	Cell delay
clknet	Lumped clock network delay
clksrc	Clock source. Used to ensure that the end-point of a clock segment is marked in the path when source latency is specified, or when the actual path cannot be found.
comp	PLL clock network compensation delay
ic	Interconnect delay
iext	External input delay
latch	Clock latch time
launch	Clock launch time
oext	External output delay
srclat	Source latency for a clock segment. This will appear if latency was specified between two clocks, or if a path could not be found between them.
unc	Clock uncertainty
utco	Register micro-Tco time
utsu	Register micro-Tsu time
uth	Register micro-Th time

Example

```
# Define a few helper procedures to print out points
# on a path, and the path itself
proc get_clock_string { path clk } {
   set clk_str ""
   set clk_id [ get_path_info $path -${clk}_clock ]

   if { $clk_id ne "" } {
   set clk_str [ get_clock_info $clk_id -name ]

 if { [ get_path_info $path -${clk}_clock_is_inverted ] } {
   append clk_str " (INVERTED)"
   }
   }

   return $clk_str
}

proc print_point { point } {
   set total     [ get_point_info $point -total    ]
   set incr      [ get_point_info $point -incr     ]
   set node_id   [ get_point_info $point -node     ]
   set type      [ get_point_info $point -type     ]
   set rf        [ get_point_info $point -rise_fall]
   set node_name ""

   if { $node_id ne "" } {
   set node_name [ get_node_info $node_id -name ]
   }

   puts \
       [format "%10s %8s %2s %-6s %s" $total $incr $rf $type $node_name ]
}

proc print_path { path } {
   puts "Slack      : [ get_path_info $path -slack]"
   puts "To Clock   : [ get_clock_string $path to ]"
   puts "From Clock : [ get_clock_string $path from]"
   puts ""
   puts \
       [format "%10s %8s %-2s %-6s %s" "Total" "Incr" "RF" "Type" "Name"]
   puts \

"===================================================================="

   foreach_in_collection pt [ get_path_info $path -arrival_points ] {
   print_point $pt
   }
}

project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist
```

```
# And now simply iterate over the 10 worst setup paths, printing each
# path
foreach_in_collection path [ get_timing_paths -npaths 10 -setup ] {
    print_path $path
    puts ""
}

delete_timing_netlist
project_close
```

get_port_info

Usage

```
get_port_info [-is_inout_port] [-is_input_port] [-is_output_port] [-name]
[-type] <port_id>
```

Options

```
-is_inout_port: Return true if it is an inout port, or false otherwise
-is_input_port: Return true if it is an input port, or false otherwise
-is_output_port: Return true if it is an output port, or false otherwise
-name: Return the port name
-type: Return the port type
<port_id>: Port ID
```

Description

Gets information about the specified port (referenced by port ID). Port ID's can be obtained by Tcl commands such as get_ports. The -type option returns "port". Options -name, -type, -is_input_port, -is_output_port and is_inout_port are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set ports [get_ports]
foreach_in_collection port $ports {
    set port_type ""
    if [get_port_info $port -is_inout_port] {
        set port_type "bidir"
    } elseif [get_port_info $port -is_input_port {
        set port_type "in"
    } else {
        set port_type "out"
    }
    puts "[get_port_info $port -name]: $port_type"
}
delete_timing_netlist
project_close
```

get_register_info

Usage

```
get_register_info [-asynch_edges] [-clock_edges] [-fanout_edges] [-
is_latch] [-name] [-synch_edges] [-tch] [-tcl] [-tco] [-th] [-tsu] [-type]
<reg_id>
```

Options

```
-asynch_edges: Return a list of asynchronous edge IDs
-clock_edges: Return a list of clock edge IDs
-fanout_edges: Return a list of fanout edge IDs
-is_latch: Return "1" if this is a latch node, or "0" otherwise
-name: Return the object name
-synch_edges: Return a list of synchronous edge IDs
-tch: Return the Tch value
-tcl: Return the Tcl value
-tco: Return the Tco value
-th: Return the Th value
-tsu: Return the Tsu value
-type: Return the object type
<reg_id>: Register ID
```

Description

Gets information about the specified register (referenced by register ID). Register IDs can be obtained by Tcl commands such as get_registers. The -type option returns "reg". Options -name, -type, -tco, -tsu, -th, -tch, -tcl, -clock_edges, -synch_edges, -asynch_edges, -fanout_edges and -is_latch are mutually exclusive.

Example

```
project_open chiptrip
create_timing_netlist
set registers [get_registers]
foreach_in_collection reg $registers {
   set name [get_register_info $reg -name]
   set tco [get_register_info $reg -tco]
    puts "Tco of $name is $tco"
}
delete_timing_netlist
project_close
```

get_timing_paths

Usage

```
get_timing_paths [-detail <SUMMARY|PATH_ONLY|PATH_AND_CLOCK|FULL_PATH>] [-
fall_from_clock <names>] [-fall_to_clock <names>] [-from <names>] [-
from_clock <names>] [-hold] [-less_than_slack <slack limit>] [-npaths
<number>] [-off] [-on] [-recovery] [-removal] [-rise_from_clock <names>]
[-rise_to_clock <names>] [-setup] [-through <names>] [-to <names>] [-
to_clock <names>]
```

Options

```
-detail <SUMMARY|PATH_ONLY|PATH_AND_CLOCK|FULL_PATH>: Option to determine
how much detail should be shown in the path report
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Get a collection of path objects for the worst-case paths.

This command behaves the same as the report_timing command. However, instead of reporting the paths, it returns a Tcl collection of path objects. Properties of these objects can then be retrieved using the get_path_info and get_point_info commands.

For help on those options that are shared with report_timing, please see the report_timing help page.

Example

```
# Define a few helper procedures to print out points
# on a path, and the path itself
proc get_clock_string { path clk } {
    set clk_str ""
    set clk_id [ get_path_info $path -${clk}_clock ]

    if { $clk_id ne "" } {
    set clk_str [ get_clock_info $clk_id -name ]

 if { [ get_path_info $path -${clk}_clock_is_inverted ] } {
    append clk_str " (INVERTED)"
    }
    }

    return $clk_str
}

proc print_point { point } {
    set total     [ get_point_info $point -total    ]
    set incr      [ get_point_info $point -incr     ]
    set node_id   [ get_point_info $point -node     ]
    set type      [ get_point_info $point -type     ]
    set rf        [ get_point_info $point -rise_fall]
    set node_name ""

    if { $node_id ne "" } {
    set node_name [ get_node_info $node_id -name ]
    }

    puts \
        [format "%10s %8s %2s %-6s %s" $total $incr $rf $type $node_name ]
}

proc print_path { path } {
    puts "Slack      : [ get_path_info $path -slack]"
    puts "To Clock   : [ get_clock_string $path to ]"
    puts "From Clock : [ get_clock_string $path from]"
    puts ""
    puts \
        [format "%10s %8s %-2s %-6s %s" "Total" "Incr" "RF" "Type" "Name"]
    puts \
"==================================================================="

    foreach_in_collection pt [ get_path_info $path -arrival_points ] {
    print_point $pt
    }
}

project_open my_project

# Always create the netlist first
create_timing_netlist
```

```
read_sdc my_project.sdc
update_timing_netlist

# And now simply iterate over the 10 worst setup paths, printing each
# path
foreach_in_collection path [ get_timing_paths -npaths 10 -setup ] {
    print_path $path
    puts ""
}

delete_timing_netlist
project_close
```

read_sdc

Usage

```
read_sdc <file_name>
```

Options

```
<file_name>: Name of the SDC file
```

Description

Reads an SDC file with all current constraints and exceptions.

If an SDC file is not specified, read_sdc reads the default SDC file. The default SDC file is based on the following rules:

- If one or more SDC_FILE assignments exist in the QSF, read all of them in order.
- Otherwise, read the file <revision>.sdc if it exists.

Example

```
project_new test
create_timing_netlist
read_sdc test_constraits.sdc
update_timing_netlist

report_timing

delete_timing_netlist
project_close
```

report_clock_fmax_summary

Usage

```
report_clock_fmax_summary [-append] [-fall_from_clock <names>] [-
fall_to_clock <names>] [-file <name>] [-from <names>] [-from_clock <names>]
[-hold] [-less_than_slack <slack limit>] [-npaths <number>] [-off] [-on]
[-panel_name <name>] [-recovery] [-removal] [-rise_from_clock <names>] [-
rise_to_clock <names>] [-setup] [-stdout] [-through <names>] [-to <names>]
[-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports potential FMAX for every clock in the design, regardless of the user-specified clock periods. FMAX is only computed for paths where the source and destination registers or ports are driven by the same clock. Paths of different clocks, including generated clocks, are ignored. For paths between a clock and its inversion, FMAX is computed as if the rising and falling edges are scaled along with FMAX, such that the duty cycle (in terms of a percentage) is maintained.

Example

```
project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Output results in the form of messages
report_clock_fmax_summary
# Create "Fmax" report panel
report_clock_fmax_summary -panel_name Fmax
# Report both with report panel and messages
report_clock_fmax_summary -panel_name Fmax -stdout

# The following command is optional
delete_timing_netlist

project_close
```

report_clock_transfers

Usage

```
report_clock_transfers [-append] [-fall_from_clock <names>] [-
fall_to_clock <names>] [-file <name>] [-from <names>] [-from_clock <names>]
[-hold] [-less_than_slack <slack limit>] [-npaths <number>] [-off] [-on]
[-panel_name <name>] [-recovery] [-removal] [-rise_from_clock <names>] [-
rise_to_clock <names>] [-setup] [-stdout] [-through <names>] [-to <names>]
[-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Creates a clock transfer summary for hold analysis
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Creates a clock transfer summary for recovery analysis
-removal: Creates a clock transfer summary for removal analysis
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Creates a clock transfer summary for setup analysis
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Generates a timing report table showing all cross-clock transfers (i.e., data paths between one clock domain and a different clock domain). The from and to clocks are shown as well as the number of from and to registers involved in the cross-clock transfer.

Report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical user interface ("-panel_name"), or any combination of the three.

The -setup, -hold, -recovery, and -removal options determine the analysis type of the report, particularly the reporting of false_paths that apply to only one analysis type. If you do not specify any of these options, a report is generated for each analysis.

Example

```
project_open top
create_timing_netlist -skip_dat
report_clock_transfers -panel_name
delete_timing_netlist
project_close
```

report_clocks

Usage

```
report_clocks [-append] [-desc] [-fall_from_clock <names>] [-fall_to_clock
<names>] [-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-summary] [-through <names>] [-to <names>] [-
to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-desc: Sort the clocks by name in descending order (ascending order is
default)
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-summary: Create a single table with a summary of each clock
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical interface ("-panel_name"), or any combination of the three.

For the Tcl console, the clock details are reported in two sections. The first sections show all clocks, their period, and their waveform. This includes generated clocks after an update_timing_netlist. The second section shows details for all generated clocks. For the timing report, both sections are combined into a single timing report.

Example

```
project_open top
create_timing_netlist
read_sdc
update_timing_netlist

report_clocks

delete_timing_netlist
project_close
```

report_datasheet

Usage

```
report_datasheet [-append] [-fall_from_clock <names>] [-fall_to_clock
<names>] [-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-through <names>] [-to <names>] [-to_clock
<names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

This function creates a datasheet report which summarizes the timing characteristics of the design as a whole. It reports setup (tsu), hold (th), clock-to-output (tco), minimum clock-to-output (mintco), propagation delay (tpd), and minimum propagation delay (mintpd) times. These delays are reported for each clock or port for which they are relevant. If there is a case where there are multiple paths for a clock (for example if there is are multiplexed clocks), then the maximum delay is taken for the tsu, th, tco and tpd, and the minimum delay is taken for mintco and mintpd.

The datasheet can be outputed to the Tcl console ("-stdout", which is default), a file ("-file"), or a report panel ("-panel_name"). Additionally if the "-file" option is used then the "-append" option can be used to specify that new data should be written to the end of the specified file.

Example

```
project_open proj1
create_timing_netlist
read_sdc
update_timing_netlist

# Report the datasheet to a report panel
report_datasheet -panel_name Datasheet

# Report the datasheet to a file
report_datasheet -file file1.txt
```

report_min_pulse_width

Usage

```
report_min_pulse_width [-append] [-fall_from_clock <names>] [-
fall_to_clock <names>] [-file <name>] [-from <names>] [-from_clock <names>]
[-hold] [-less_than_slack <slack limit>] [-npaths <number>] [-nworst
<number>] [-off] [-on] [-panel_name <name>] [-recovery] [-removal] [-
rise_from_clock <names>] [-rise_to_clock <names>] [-setup] [-stdout] [-
through <names>] [-to <names>] [-to_clock <names>] <targets>
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-nworst <number>: Specifies the number of pulse width checks to report
(default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
<targets>: Registers or clock pins
```

Description

Reports the results of minimum pulse width checks. A minimum pulse width check verifies that a clock high or low pulse sustains long enough to qualify as a recognizable change in the clock signal. A failed minimum pulse width check indicates that the register may not recognize the clock transition.

Each register in the design is reported twice per clock that clocks the register: once for the high pulse and once for the low pulse. Specific registers can be checked by specifying a register or collection of registers. The number of checks reported can be limited using the "-nworst" option.

The results of the minimum pulse width checks can be output to the Tcl console ("-stdout," the default), a report panel ("-panel"), a file ("-file"), or a combination of the three.

Results are sorted from worst-case slack to best-case slack.

The value of the targets is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open top
create_timing_netlist
read_sdc
update_timing_netlist

# Report the worst 100 minimum pulse width checks
report_min_pulse_width -nworst 100

# Report minimum pulse width checks for the register test_reg[*]
report_min_pulse_width test_reg[*]

# Output the previous results to a report panel and a file.
report_min_pulse_width -panel_name "Min Pulse (test_reg)" test_reg[*]

# Output the previous results to a file.
report_min_pulse_width -file min_pulse_test_reg.txt test_reg[*]
```

report_net_timing

Usage

```
report_net_timing [-append] [-fall_from_clock <names>] [-fall_to_clock
<names>] [-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-nworst_delay <number>]
[-nworst_fanout <number>] [-off] [-on] [-panel_name <name>] [-recovery] [-
removal] [-rise_from_clock <names>] [-rise_to_clock <names>] [-setup] [-
stdout] [-through <names>] [-to <names>] [-to_clock <names>] <name>
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-nworst_delay <number>: Report worst n net delays
-nworst_fanout <number>: Report worst n fanout nets
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
<name>: Signal or Collection Name
```

Description

Reports delay and fanout information about a net in the design. A net corresponds to a cell output pin.

Report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical interface ("-panel_name"), or any combination of the three.

The value of the name is either a collection or a Tcl list of wildcards used to create a collection of the appropriate type. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open <design>
create_timing_netlist

# Show delay and fanout information for all nets
# that match "abc*"
report_net_timing [get_nets abc*]

# Report delay and fanout information for the 10
# nets showing higher delays
report_net_timing -nworst_delay 10

# Report delay and fanout information for the 10
# nets showing higher fanout
report_net_timing -nworst_fanout 10

project_close
```

report_path

Usage

```
report_path [-append] [-fall_from_clock <names>] [-fall_to_clock <names>]
[-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-summary] [-through <names>] [-to <names>] [-
to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-summary: Create a single table with a summary of each path found
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports the longest delay paths and the corresponding delay value.

The report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical user interface ("-panel_name"), or any combination of the three.

You can limit the analysis performed by this command to specific start and end points, using the "-from" and "-to" options. Any node or cell in the design is considered a valid endpoint. Additionally, the "-through" option can be used to restrict analysis to paths which go through specified pins or nets.

Use "-npaths" to limit the number of paths to report. If this option is not specified, only the single longest delay path is provided.

Use the "-pairs_only" option to filter the output further, restricting the results to only unique combinations of start and end points. This filtering is performed after the number of paths has been generated in accordance to the "-npaths" option. As a result, there may be fewer paths displayed than specified by "-npaths", if a particular set of start and end points appeared multiple times.

Use the "-summary" option to generate a single table listing only the highlights of each path.

The "-show_routing" option will display detailed routing information in the path.

The return value of this command is a two-element list. The first number is the number of paths found in the analysis. The second is the longest delay, in terms of the current default time unit.

The values of the -from, -to, -through, etc. are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Report path delay between nodes "foo" and "bar",
# reporting the longest delay if a path is found.

set my_list [report_path -from foo -to bar]
set num_paths [lindex $my_list 0]
set longest_delay [lindex $my_list 1]
if { $num_paths > 0 } {
    puts "Longest delay -from foo -to bar is $longest_delay"
}
```

```
# The following command is optional
delete_timing_netlist

project_close
```

report_sdc

Usage

```
report_sdc [-append] [-fall_from_clock <names>] [-fall_to_clock <names>]
[-file <name>] [-from <names>] [-from_clock <names>] [-hold] [-ignored] [-
less_than_slack <slack limit>] [-npaths <number>] [-off] [-on] [-panel_name
<name>] [-recovery] [-removal] [-rise_from_clock <names>] [-rise_to_clock
<names>] [-setup] [-stdout] [-through <names>] [-to <names>] [-to_clock
<names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-ignored: Reports full history of assignments to locate ignored ones
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports all the sdc constraints.

Example

```
project_new test
create_timing_netlist
create_clock -period 10 -name clk10 clk
set_multicycle_path -from [get_cells a] -to [get_cells b]
update_timing_netlist

report_sdc -panel_name sdc_report_panel

report_timing

delete_timing_netlist
project_close
```

report_timing

Usage

```
report_timing [-append] [-detail
<SUMMARY|PATH_ONLY|PATH_AND_CLOCK|FULL_PATH>] [-fall_from_clock <names>]
[-fall_to_clock <names>] [-file <name>] [-from <names>] [-from_clock
<names>] [-hold] [-less_than_slack <slack limit>] [-npaths <number>] [-off]
[-on] [-panel_name <name>] [-recovery] [-removal] [-rise_from_clock
<names>] [-rise_to_clock <names>] [-setup] [-stdout] [-through <names>] [-
to <names>] [-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-detail <SUMMARY|PATH_ONLY|PATH_AND_CLOCK|FULL_PATH>: Option to determine
how much detail should be shown in the path report
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from <names>: Valid sources (string patterns are matched using Tcl string
matching)
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-hold: Option to report clock hold paths
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-setup: Option to report clock setup paths
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to <names>: Valid destinations (string patterns are matched using Tcl
string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports the worst-case paths and associated slack.

Use the "-setup", "-hold", "-recovery", or "-removal" options to specify which kind of analysis should be performed.

The report can be directed to the Tcl console ("-stdout", default), a file ("-file"), the TimeQuest graphical user interface ("-panel_name"), or any combination of the three.

You can limit the analysis performed by this command to specific start and end points, using the "-from" and "-to" options. The anlaysis can be further limited to clocks using the "-from_clock" and "-to_clock" options, or to specific edges of the clock using the "-rise_from_clock", "-fall_from_clock", "-rise_to_clock", and "-fall_to_clock" options. Additionally, the "-through" option can be used to restrict analysis to paths which go through specified pins or nets.

Use "-npaths" to limit the number of paths to report. If this option is not specified, only the single worst-case path is provided. Use the "-less_than_slack" option to limit output to all paths with slack less than the specified value, up to the number specified by "-npaths".

Use the "-detail" option to specify the desired level of report detail. "summary" generates a single table listing only the highlights of each path (and is the same as "-summary" option, which this replaces). "path_only" reports the path from the source to the destination without any detail about the clock path. Instead, the clock network delay is shown as a single number. This is the default behavior. "path_and_clock" extends the arrival and required paths back to the launch and latch clocks. "full_path" will continue tracing back through generated clocks to the underlying base clock.

Use the "-pairs_only" option to filter the output further, restricting the results to only unique combinations of start and end points. This filtering is performed after the number of paths has been generated in accordance to the "-npaths" option. As a result, there may be fewer paths displayed than specified by "-npaths", if a particular set of start and end points appeared multiple times.

The return value of this command is a two-element list. The first number is the number of paths found in the analysis. The second is the worst-case slack, in terms of the current default time unit.

The "RF" column in the report output uses a two-letter symbol to indicate the rise/fall transition that occurs at that point in the path.

Possible "RF" values are:

Value	Description
(empty)	Unknown transition
R	Rising output
F	Falling output
RR	Rising input, rising output

RF	Rising input, falling output
FR	Falling input, rising output
FF	Falling input, falling output

The "Type" column in the report uses a symbol to indicate what type of delay occurs at that point in the path.

Possible "Type" values are:

Value	Description
CELL	Cell delay
COMP	PLL clock network compensation delay
IC	Interconnect delay
iExt	External input delay
oExt	External output delay
uTco	Register micro-Tco time
uTsu	Register micro-Tsu time
uTh	Register micro-Th time

The values of the -from, -to, -through, etc. are either collections or a Tcl list of wildcards used to create collections of appropriate types. The values used must follow standard Tcl or TimeQuest-extension substitution rules. See the help for use_timequest_style_escaping for details.

Example

```
project_open my_project

# Always create the netlist first
create_timing_netlist
read_sdc my_project.sdc
update_timing_netlist

# Run a setup analysis between nodes "foo" and "bar",
# reporting the worst-case slack if a path is found.

set my_list [report_timing -from foo -to bar]
set num_paths [lindex $my_list 0]
set wc_slack [lindex $my_list 1]
if { $num_paths > 0 } {
    puts "Worst case slack -from foo -to bar is $wc_slack"
}

# The following command is optional
delete_timing_netlist

project_close
```

report_ucp

Usage

```
report_ucp [-append] [-fall_from_clock <names>] [-fall_to_clock <names>]
[-file <name>] [-from_clock <names>] [-less_than_slack <slack limit>] [-
npaths <number>] [-off] [-on] [-panel_name <name>] [-recovery] [-removal]
[-rise_from_clock <names>] [-rise_to_clock <names>] [-stdout] [-summary]
[-through <names>] [-to_clock <names>]
```

Options

```
-append: If output is sent to a file, this option appends the result to
that file.  Otherwise, the file will be overwritten
-fall_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-fall_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-file <name>: Sends the results to a file
-from_clock <names>: Valid source clocks (string patterns are matched using
Tcl string matching)
-less_than_slack <slack limit>: Limit the paths reported to those with
slack values less than the specified limit.
-npaths <number>: Specifies the number of paths to report (default=1)
-off: Disable this setting.
-on: Enable this setting.
-panel_name <name>: Sends the results to the panel and specifies the name
of the new panel
-recovery: Option to report recovery paths
-removal: Option to report removal paths
-rise_from_clock <names>: Valid source clocks (string patterns are matched
using Tcl string matching)
-rise_to_clock <names>: Valid destination clocks (string patterns are
matched using Tcl string matching)
-stdout: Send output to stdout, via messages.  You only need to use this
option if you have selected another output format, such as a file, and would
also like to receive messages.
-summary: Generate only the summary panel.
-through <names>: Valid through nodes (string patterns are matched using
Tcl string matching)
-to_clock <names>: Valid destination clocks (string patterns are matched
using Tcl string matching)
```

Description

Reports unconstrained paths.

Example

```
project_open chiptrip
create_timing_netlist
read_sdc
update_timing_netlist
report_ucp
delete_timing_netlist
project_close
```

timing_netlist_exist

Usage

```
timing_netlist_exist
```

Options

None

Description

Checks if the timing netlist exists.

Returns 1, if the timing netlist exists. Returns 0, otherwise.

Example

```
if {![timing_netlist_exist]} {
    create_timing_netlist
}
```

update_timing_netlist

Usage

```
update_timing_netlist
```

Options

None

Description

Updates and applies SDC commands to the timing netlist. update_timing_netlist expands and validates generated clocks, warns about sources in the design that require clock settings, identifies and removes combinational loops, and warns about undefined input/output delays.

Example

```
project_open top
create_timing_netlist
read_sdc
update_timing_netlist

report_timing -to_clock clk1
report_timing -to_clock clk2

delete_timing_netlist
project_close
```

use_timequest_style_escaping

Usage

```
use_timequest_style_escaping [-off] [-on]
```

Options

```
-off: Disable this setting.
-on: Enable this setting.
```

Description

Use the TimeQuest style escaping. Default to -on option.

The values used to create a collection, whether explicitly using a collection command or implicitly as a value specified as a "-from", "-to", or similar option to various SDC and report commands, are a Tcl list of wildcards. This includes a single name with an exact match. The value must follow standard Tcl substitution rules for Tcl lists and "string match" as described below, unless using TimeQuest-style escaping (default).

For special characters such as '$', the character must be escaped using a single '\' character to prevent Tcl from interpreting the word after '$' as a Tcl variable, such as: Clk\$Signal.

A '\' character itself must be escaped with another '\' as in the '$' case, must be escaped again for the Tcl list, and must be escaped yet again for Tcl "string match." The final result is eight '\' characters, such as: Clk\\\\\\\\Signal.

Using Tcl "list" eliminates one level of escaping, since it will escape any '\' characters automatically for the Tcl list, such as:

```
    [list Clk\\\\Signal]
```

Using '{' and '}' characters also eliminates the need for one or two levels of escaping, since '{' and '}' prevent string substitution in the contents, such as:

```
    [List {Clk\\Signal}]
    {{Clk\\Signal}}
```

The use_timequest_style_escaping option, which is on by default, allows the user to specify a name containing '\' characters with only two '\' characters in all cases, such as: Clk\\Signal. The extra '\' characters required for Tcl list string substitution and "string match" are added automatically by TimeQuest.

To disable TimeQuest style string escaping, call "use_timequest_style_escaping -off" before adding any timing constraints or exceptions.

Example

```
project_open top
use_timequest_style_escaping -on
create_timing_netlist
set res [get_cells my_test|special_\\reg]
query_collection $res -all

delete_timing_netlist
project_close
```

write_sdc

Usage

```
write_sdc <file_name>
```

Options

```
<file_name>: Name of output file
```

Description

Generates an SDC file with all current constraints and exceptions.

Example

```
project_new test
create_timing_netlist
create_clock -period 10 -name clk10 clk
set_multicycle_path -from [get_cells a] -to [get_cells b]
update_timing_netlist

report_timing

write_sdc my_sdc_file.sdc

delete_timing_netlist
project_close
```