# OpenSTA

# Table of Contents

## Command Line Arguments

The command line arguments for `sta` are shown below.

```
sta
  -help             show help and exit
  -version          show version and exit
  -no_init          do not read ~/.sta
  -no_splash        do not print the splash message
  -threads count|max use count threads
  -exit             exit after reading cmd_file
  cmd_file          source cmd_file
```

When OpenSTA starts up, commands are first read from the user initialization file `~/.sta` if it exists. If a TCL command file *cmd_file* is specified on the command line, commands are read from the file and executed before entering an interactive TCL command interpreter. If -exit is specified the application exits after reading *cmd_file*. Use the TCL `exit` command to exit the application. The `–threads` option specifies how many parallel threads to use. Use `–threads max` to use one thread per processor.

## Example Command Scripts

To read a design into OpenSTA use the `read_liberty` command to read Liberty library files. Next, read hierarchical structural Verilog files with the `read_verilog` command. The `link_design` command links the Verilog to the Liberty timing cells. Any number of Liberty and Verilog files can be read before linking the design.

Delays used for timing analysis are calculated using the Liberty timing models. If no parasitics are read only the pin capacitances of the timing models are used in delay calculation. Use the `read_spef` command to read parasitics from an extractor, or `read_sdf` to use delays calculated by an external delay calculator.

Timing constraints can be entered as TCL commands or read using the `read_sdc` command.

The units used by OpenSTA for all command arguments and reports are taken from the first Liberty file that is read. Use the `set_cmd_units` command to override the default units.

## Timing Analysis using SDF

A sample command file that reads a library and a Verilog netlist and reports timing checks is shown below.

```
read_liberty example1_slow.lib
read_verilog example1.v
link_design top
read_sdf example1.sdf
create_clock -name clk -period 10 {clk1 clk2 clk3}
set_input_delay -clock clk 0 {in1 in2}
report_checks
```

This example can be found in `examples/sdf_delays.tcl`.

## Timing Analysis with Multiple Process Corners

An example command script using three process corners and +/-10% min/max derating is shown below.

```
define_corners wc typ bc
read_liberty -corner wc  example1_slow.lib
read_liberty -corner typ example1_typ.lib
read_liberty -corner bc  example1_fast.lib
read_verilog example1.v
link_design top
set_timing_derate -early 0.9
set_timing_derate -late 1.1
create_clock -name clk -period 10 {clk1 clk2 clk3}
set_input_delay -clock clk 0 {in1 in2}
report_checks -path_delay min_max
report_checks -corner typ
```

This example can be found in `examples/spef_parasitics.tcl`. Other examples can be found in the `examples` directory.

## Power Analysis

OpenSTA also supports static power analysis with the `report_power` command. Probabalistic switching activities are propagated from the input ports to determine switching activities for internal pins.

```
read_liberty sky130hd_tt.lib
read_verilog gcd_sky130hd.v
link_design gcd
read_sdc gcd_sky130hd.sdc
read_spef gcd_sky130hd.spef
set_power_activity -input -activity 0.1
set_power_activity -input_port reset -activity 0
report_power
```

In this example the activity for all inputs is set to `0.1`, and then the activity for the `reset` signal is set to zero because it does not switch during steady state operation.

```
Group               Internal  Switching   Leakage      Total
                       Power      Power     Power      Power (Watts)
             -------------------------------------------------------
Sequential          3.27e-04   7.87e-05   2.96e-10   4.06e-04   36.4%
Combinational       2.34e-04   3.10e-04   6.95e-10   5.43e-04   48.7%
Clock               4.68e-05   1.20e-04   2.30e-11   1.67e-04   15.0%
Macro               0.00e+00   0.00e+00   0.00e+00   0.00e+00    0.0%
Pad                 0.00e+00   0.00e+00   0.00e+00   0.00e+00    0.0%
             -------------------------------------------------------
Total               6.07e-04   5.09e-04   1.01e-09   1.12e-03  100.0%

                      54.4%      45.6%      0.0%
```

This example can be found in `examples/power.tcl`.

Gate level simulation results can be used to get a more accurate power estimate. For example, the Icarus verilog simulator can be used to run the the test bench `examples/gcd_tb.v` for the gcd design in the previous example.

```
iverilog -o gcd_tb gcd_tb.v
vvp gcd_tb
```

The test bench writes the VCD (Value Change Data) file `gcd_sky130hd.vcd` which can then be read with the `read_vcd` command.

```
read_liberty sky130hd_tt.lib
read_verilog gcd_sky130hd.v
link_design gcd
read_sdc gcd_sky130hd.sdc
read_spef gcd_sky130hd.spef
read_vcd -scope gcd_tb/gcd1 gcd_sky130hd.vcd.gz
report_power
```

This example can be found in `examples/power_vcd.tcl`.

Note that in this simple example design simulation based activities does not significantly change the results.

## TCL Interpreter

Keyword arguments to commands may be abbreviated. For example,

```
report_checks -unique
```

is equivalent to the following command.

```
report_checks -unique_paths_to_endpoint
```

The help command lists matching commands and their arguments.

```
> help report*
report_annotated_check [-setup] [-hold] [-recovery] [-removal] [-nochange]
 [-width] [-period] [-max_skew] [-max_lines liness] [-
```

```
list_annotated]group_path_count
    [-list_not_annotated] [-constant_arcs]
report_annotated_delay [-cell] [-net] [-from_in_ports] [-to_out_ports]
    [-max_lines liness] [-list_annotated] [-list_not_annotated] [-constant_arcs]
report_arrival pin
report_check_types [-violators] [-verbose] [-corner corner]
    [-format slack_only|end] [-max_delay] [-min_delay] [-recovery] [-removal]
    [-clock_gating_setup] [-clock_gating_hold] [-max_slew] [-min_slew]
    [-max_fanout] [-min_fanout] [-max_capacitance] [-min_capacitance
    [-min_pulse_width] [-min_period] [-max_skew] [-net net] [-digits digits
    [-no_line_splits] [> filename] [>> filename]
report_checks [-from from_list|-rise_from from_list|-fall_from from_list]
    [-through through_list|-rise_through through_list|-fall_through
through_list]
    [-to to_list|-rise_to to_list|-fall_to to_list] [-unconstrained]
    [-path_delay min|min_rise|min_fall|max|max_rise|max_fall|min_max]
    [-corner corner] [-group_path_count path_count]
    [-endpoint_path_count path_count]
    [-unique_paths_to_endpoint] [-slack_max slack_max] [-slack_min slack_min]
    [-sort_by_slack] [-path_group group_name]
    [-format full|full_clock|full_clock_expanded|short|end|summary]
...
```

Many reporting commands support redirection of the output to a file much like a Unix shell.

```
report_checks -to out1 > path.log
report_checks -to out2 >> path.log
```

# Debugging Timing

Here are some guidelines for debugging your design if static timing does not report any paths, or does not report the expected paths.

Debugging timing problems generally involves using the following commands to follow the propagation of arrival times from a known arrival downstream to understand why the arrival times are not propagating:

```
report_edges
report_arrivals
report_net
```

report_edges -from can be used to walk forward and report_edges -to to walk backward in the netlist/timing graph. report_arrivals shows the min/max rise/fall arrival times with respect to each clock that has a path to the pin. report_net shows connections to a net across hierarchy levels.

### No paths found

The report_checks command only reports paths that are constrained by timing checks or SDC commands such as set_output_delay. If the design has only combinational logic (no registers or latches), there are no timing checks, so no paths are reported. Use the -unconstrained option to report_checks to see unconstrained paths.

```
% report_checks -unconstrained
```

If the design is sequential (has registers or latches) and no paths are reported, it is likely that there is a problem with the clock propagation. Check the timing at an register in the design with the `report_arrivals` command.

```
% report_arrivals r1/CP
 (clk ^) r 0.00:0.00 f INF:-INF
 (clk v) r INF:-INF f 5.00:5.00
```

In this example the rising edge of the clock "clk" causes the rising arrival min:max time at 0.00, and the falling edge arrives at 5.00. Since the rising edge of the clock causes the rising edge of the register clock pin, the clock path is positive unate.

The clock path should be positive or negative unate. Something is probably wrong with the clock network if it is non-unate. A non-unate clock path will report arrivals similar to the foillowing:

```
% report_arrivals r1/CP
 (clk ^) r 0.00:0.00 f 0.00:0.00
 (clk v) r 5.00:5.00 f 5.00:5.00
```

Notice that each clock edge causes both rise and fall arrivals at the register clock pin.

If there are no paths to the register clock pin, nothing is printed. Use the `report_edges -to` command to find the gate driving the clock pin.

```
% report_edges -to r1/CP
i1/ZN -> CP wire
  ^ -> ^ 0.00:0.00
 v -> v 0.00:0.00
```

This shows that the gate/pin i1/ZN is driving the clock pin. The `report_edges -to` commond can be used to walk backward or forward through the netlist one gate/net at a time. By checking the arrivals with the `report_arrival` command you can determine where the path is broken.

## No path reported an endpoint

In order for a timing check to be reported, there must be an arrival time at the data pin (the constrained pin) as well as the timing check clock pin. If `report_checks -to` a register input does not report any paths, check that the input is constrained by a timing check with `report_edges -to`.

```
% report_edges -to r1/D
CP -> D hold
  ^ -> ^ -0.04:-0.04
  ^ -> v -0.03:-0.03
CP -> D setup
  ^ -> ^ 0.09:0.0
  ^ -> v 0.08:0.08
in1 -> D wire
  ^ -> ^ 0.00:0.00
 v -> v 0.00:0.00
```

This reports the setup and hold checks for the D pin of r1.

Next, check the arrival times at the D and CP pins of the register with `report_arrivals`.

```
% report_arrivals r1/D
  (clk1 ^) r 1.00:1.00 f 1.00:1.00
% report_arrivals r1/CP
  (clk1 ^) r 0.00:0.00 f INF:-INF
  (clk1 v) r INF:-INF f 5.00:5.00
```

If there are no arrivals on an input port of the design, use the `set_input_delay` command to specify the arrival times on the port.

# Commands

---

**all_clocks**

---

The `all_clocks` command returns a list of all clocks that have been defined.

---

| **all_inputs** | [-no_clocks] |
|---|---|
| -no_clocks | Exclude inputs defined as clock sources. |

---

The `all_inputs` command returns a list of all input and bidirect ports of the current design.

---

**all_outputs**

---

The `all_outputs` command returns a list of all output and bidirect ports of the design.

---

| **all_registers** | [-clock *clock_names*]<br>[-cells \| -data_pins \| -clock_pins \| -async_pins<br> \| -output_pins]<br>[-level_sensitive]<br>[-edge_triggered] |
|---|---|
| -clock *clock_names* | A list of clock names. Only registers clocked by these clocks are returned. |
| -cells | Return a list of register instances. |
| -data_pins | Return the register data pins. |
| -clock_pins | Return the register clock pins. |
| -async_pins | Return the register set/clear pins. |
| -output_pins | Return the register output pins. |
| -level_sensitive | Return level-sensitive latches. |

---

```
-edge_triggered           Return edge-triggered registers.
```

The `all_registers` command returns a list of  register instances or register pins in the design. Options allow the list of registers to be restricted in various ways. The -clock keyword restrcts the registers to those that are clocked by a set of clocks. The `-cells` option returns the list of registers or latches (the default). The --data_pins, -clock_pins, -async_pins and -output_pins options cause `all_registers` to return a list of register pins rather than instances.

| | |
|---|---|
| **check_setup** | [-verbose]<br>[-unconstrained_endpoints]<br>[-multiple_clock]<br>[-no_clock]<br>[-no_input_delay]<br>[-loops]<br>[-generated_clocks]<br>[> *filename*]<br>[>> *filename*] |

```
-verbose                  Show offending objects rather than just error counts.

-unconstrained_endpoin    Check path endpoints for timing constraints (timing
ts                        check or set_output_delay).

-multiple_clock           Check register/latch clock pins for multiple clocks.

-no_clock                 Check register/latch clock pins for a clock.

-no_input_delay           Check for inputs that do not have a set_input_delay
                          command.

-loops                    Check for combinational logic loops.

-generated_clocks         Check that generated clock source pins have been
                          defined as clocks.
```

The `check_setup` command performs sanity checks on the design. Individual checks can be performed with the keywords. If no check keywords are specified all checks are performed. Checks that fail are reported as warnings. If no checks fail nothing is reported. The command returns 1 if there are no warnings for use in scripts.

| | |
|---|---|
| **connect_pin** | *net*<br>*port*\|*pin* |

```
net                       A net to add connections to.

port                      A port to connect to net.

Pin                       A pin to connect to net.
```

The `connect_pin` command connects a port or instance pin to a net.

| | |
|---|---|
| **create_clock** | `-period period`<br>`[-name clock_name]`<br>`[-waveform edge_list]`<br>`[-add]`<br>`[pin_list]` |

`-period period`      The clock period.

`-name clock_name`      The name of the clock.

`-waveform edge_list`      A list of edge rise and fall time.

`-add`                   Add this clock to the clocks on `pin_list`.

`pin_list`                A list of pins driven by the clock.

The `create_clock` command defines the waveform of a clock used by the design.

If no `pin_list` is specified the clock is *virtual*. A virtual clock can be refered to by name in input arrival and departure time commands but is not attached to any pins in the design.

If no clock name is specified the name of the first pin is used as the clock name.

If a wavform is not specified the clock rises at zero and falls at half the clock period. The waveform is a list with time the clock rises as the first element and the time it falls as the second element.

If a clock is already defined on a pin the clock is redefined using the new clock parameters. If multiple clocks drive the same pin, use the `-add` option to prevent the existing definition from being overwritten.

The following command creates a clock with a period of 10 time units that rises at time 0 and falls at 5 time units on the pin named clk1.

```
create_clock -period 10 clk1
```

The following command creates a clock with a period of 10 time units that is high at time zero, falls at time 2 and rises at time 8. The clock drives three pins named `clk1`, `clk2`, and `clk3`.

```
create_clock -period 10 -waveform {8 2} -name clk {clk1 clk2 clk3}
```

| | |
|---|---|
| **create_generated_clock** | [-name *clock_name*] |
| | -source *master_pin* |
| | [-master_clock *master_clock*] |
| | [-divide_by *divisor*] |
| | [-multiply_by *multiplier*] |
| | [-duty_cycle duty_cycle] |
| | [-invert] |
| | [-edges *edge_list*] |
| | [-edge_shift *shift_list*] |
| | [-add] |
| | *pin_list* |

| | |
|---|---|
| -name *clock_name* | The name of the generated clock. |
| -source *master_pin* | A pin or port in the fanout of the master clock that is the source of the generated clock. |
| -master_clock *master_clock* | Use -master_clock to specify which source clock to use when multiple clocks are present on *master_pin*. |
| -divide_by *divisor* | Divide the master clock period by *divisor*. |
| -multiply_by *multiplier* | Multiply the master clock period by *multiplier*. |
| -duty_cycle *duty_cycle* | The percent of the period that the generated clock is high (between 0 and 100). |
| -invert | Invert the master clock. |
| -edges *edge_list* | List of master clock edges to use in the generated clock. Edges are numbered from 1. *edge_list* must be 3 edges long. |
| -edge_shift *shift_list* | Not supported. |
| -add | Add this clock to the existing clocks on *pin_list*. |
| *pin_list* | A list of pins driven by the generated clock. |

The create_generated_clock command is used to generate a clock from an existing clock definition. It is used to model clock generation circuits such as clock dividers and phase locked loops.

The -divide_by, -multiply_by and -edges arguments are mutually exclusive.

The -multiply_by option is used to generate a higher frequency clock from the source clock. The period of the generated clock is divided by *multiplier*. The clock *multiplier* must be a positive integer. If a duty cycle is specified the generated clock rises at zero and falls at period * duty_cycle / 100. If no duty cycle is specified the source clock edge times are divided by *multiplier*.

The `-divide_by` option is used to generate a lower frequency clock from the source clock. The clock *divisor* must be a positive integer. If the clock divisor is a power of two the source clock period is multiplied by *divisor*, the clock rise time is the same as the source clock, and the clock fall edge is one half period later. If the clock divisor is not a power of two the source clock waveform edge times are multiplied by *divisor*.

The `-edges` option forms the generated clock waveform by selecting edges from the source clock waveform.

If the `-invert` option is specified the waveform derived above is inverted.

If a clock is already defined on a pin the clock is redefined using the new clock parameters. If multiple clocks drive the same pin, use the `-add` option to prevent the existing definition from being overwritten.

In the example show below generates a clock named `gclk1` on register output pin `r1/Q` by dividing it by four.

```
create_clock -period 10 -waveform {1 8} clk1
create_generated_clock -name gclk1 -source clk1 -divide_by 4 r1/Q
```

The generated clock has a period of 40, rises at time 1 and falls at time 21.

In the example shown below the duty cycle is used to define the derived clock waveform.

```
create_generated_clock -name gclk1 -source clk1 -duty_cycle 50 \
                       -multiply_by 2 r1/Q
```

The generated clock has a period of 5, rises at time .5 and falls at time 3.

In the example shown below the first, third and fifth source clock edges are used to define the derived clock waveform.

```
create_generated_clock -name gclk1 -source clk1 -edges {1 3 5} r1/Q
```

The generated clock has a period of 20, rises at time 1 and falls at time 11.

| | |
|---|---|
| **create_voltage_area** | [-name *name*] |
| | [-coordinate *coordinates*] |
| | [-guard_band_x *guard_x*] |
| | [-guard_band_y *guard_y*] |
| | *cells* |

This command is parsed and ignored by timing analysis.

| | |
|---|---|
| **current_design** | [*design*] |

| **current_instance** | [*instance*] |
|---|---|
| *instance* | Not supported. |

| **define_corners** | *corner1 [corner2]...* |
|---|---|
| *corner* | The name of a delay calculation corner. |

Use the define_corners command to define the names of multiple process/temperature/voltage corners. The define_corners command must follow set_operating_conditions -analysis_type and precede any reference to the corner names and can only appear once in a command file. There is no support for re-defining corners.

For analysis type single, each corner has one delay calculation result and early/late path arrivals. For analysis type best_case/worst_case and on_chip_variation, each corner has min/max delay calculation results and early/late path arrivals.

| **delete_clock** | [-all] *clocks* |
|---|---|
| *clocks* | A list of clocks to remove. |

| **delete_from_list** | *list objects* |
|---|---|
| *list* | *A list of objects.* |
| *objects* | *A list of objects to delete from list.* |

| **delete_generated_clock** | [-all] *clocks* |
|---|---|
| *clocks* | A list of generated clocks to remove. |

| **delete_instance** | *instance* |
|---|---|
| *instance* | Instance to delete. |

The network editing command `delete_instance` removes an instance from the design.

| | |
|---|---|
| **delete_net** | *net* |

| | |
|---|---|
| *net* | Net to delete. |

The network editing command `delete_net` removes a net from the design.

| | |
|---|---|
| **disconnect_pin** | *net*<br>*port* \| *pin* \| -all |

| | |
|---|---|
| *net* | The net to disconnect pins from. |
| *port* | A port to connect to *net*. |
| *pin* | A pin to connect to *net*. |
| -all | Disconnect all pins from the net. |

Disconnects a port or pin from a net. Parasitics connected to the pin are deleted.

| | |
|---|---|
| **elapsed_run_time** | |

Returns the total clock run time in seconds as a float.

| | |
|---|---|
| **find_timing_paths** | [-from *from_list*<br> \|-rise_from *from_list*<br> \|-fall_from *from_list*]<br>[-through *through_list*<br> \|-rise_through *through_list*<br> \|-fall_through *through_list*]<br>[-to *to_list*<br> \|-rise_to *to_list*<br> \|-fall_to *to_list*]<br>[-unconstrained]<br>[-path_delay min\|min_rise\|min_fall<br>          \|max\|max_rise\|max_fall<br>          \|min_max]<br>[-group_path_count *path_count*]<br>[-endpoint_path_count *endpoint_path_count*]<br>[-unique_paths_to_endpoint]<br>[-corner *corner*]<br>[-slack_max *max_slack*]<br>[-slack_min *min_slack*]<br>[-sort_by_slack]<br>[-path_group *groups*] |

```
-from from_list              Return paths from a list of clocks, instances,
                             ports, register clock pins, or latch data pins.

-rise_from from_list         Return paths from the rising edge of clocks,
                             instances, ports, register clock pins, or latch data
                             pins.

-fall_from from_list         Return paths from the falling edge of clocks,
                             instances, ports, register clock pins, or latch data
                             pins.

-through through_list        Return paths through a list of instances, pins or
                             nets.

-rise_through               Return rising paths through a list of instances,
through_list                 pins or nets.

-fall_through               Return falling paths through a list of instances,
through_list                 pins or nets.

-to to_list                  Return paths to a list of clocks, instances, ports
                             or pins.

-rise_to to_list             Return rising paths to a list of clocks, instances,
                             ports or pins.

-fall_to to_list             Return falling paths to a list of clocks, instances,
                             ports or pins.

-unconstrained               Report unconstrained paths also.

-path_delay min              Return min path (hold) checks.

-path_delay min_rise         Return min path (hold) checks for rising endpoints.

-path_delay min_fall         Return min path (hold) checks for falling endpoints.

-path_delay max              Return max path (setup) checks.

-path_delay max_rise         Return max path (setup) checks for rising endpoints.

-path_delay max_fall         Return max path (setup) checks for falling
                             endpoints.

-path_delay min_max          Return max and max path (setup and hold) checks.

-group_path_count            The number of paths to return in each path group.
path_count
```

```
-endpoint_path_count        The number of paths to return for each endpoint.
endpoint_path_count


-unique_paths_to_endpoint   Return multiple paths to an endpoint that traverse
                            different pins without showing multiple paths with
                            different rise/fall transitions.


-corner corner              Return paths for one process corner.


-slack_max max_slack        Return paths with slack less than max_slack.


-slack_min min_slack        Return paths with slack greater than min_slack.


-sort_by_slack              Sort paths by slack rather than slack within path
                            groups.


-path_group groups          Return paths in path groups. Paths in all groups are
                            returned if this option is not specified.
```

The `find_timing_paths` command returns a list of path objects for scripting. Use the `get_property` function to access properties of the paths.

```
get_cells              [-hierarchical]
                       [-hsc separator]
                       [-filter expr]
                       [-regexp]
                       [-nocase]
                       [-quiet]
                       [-of_objects objects]
                       [patterns]


-hierarchical          Searches hierarchy levels below the current instance for
                       matches.


-hsc separator         Character to use to separate hierarchical instance names
                       in patterns.


-filter expr           A filter expression of the form
                          "property==value"
                       where property is a property supported by the
                       get_property command.  See the section "Filter
                       Expressions" for additional forms.


-regexp                Use regular expression matching instead of  glob pattern
                       matching.


-nocase                Ignore case when matching.  Only valid with –regexp.


-quiet                 Do not warn if no matches are found.
```

| | |
|---|---|
| -of_objects *objects* | The name of a pin or net, a list of pins returned by get_pins, or a list of nets returned by get_nets. The – hierarchical option cannot be used with -of_objects. |
| *patterns* | A list of instance name patterns. |

The get_cells command returns a list of all cell instances that match *patterns*.

| | |
|---|---|
| **get_clocks** | [-regexp]<br>[-nocase]<br>[-filter *expr*]<br>[-quiet]<br>*patterns* |
| -regexp | Use regular expression matching instead of  glob pattern matching. |
| -nocase | Ignore case when matching.  Only valid with –regexp. |
| -filter *expr* | A filter expression of the form<br>   "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |
| -quiet | Do not warn if no matches are found. |
| *patterns* | A list of clock name patterns. |

The get_clocks command returns a list of all clocks that have been defined.

| | |
|---|---|
| **get_fanin** | -to *sink_list*<br>[-flat]<br>[-only_cells]<br>[-startpoints_only]<br>[-levels *level_count*]<br>[-pin_levels *pin_count*]<br>[-trace_arcs timing\|enabled\|all] |
| -to *sink_list* | List of pins, ports, or nets to find the fanin of. For nets, the fanin of driver pins on the nets are returned. |
| -flat | With –flat pins in the fanin at any hierarchy level are returned. Without -flat only pins at the same hierarchy level as the sinks are returned. |
| -only_cells | Return the instances connected to the pins in the fanin. |
| -startpoints_only | Only return pins that are startpoints. |

| | |
|---|---|
| -level *level_count* | Only return pins within *level_count* instance traversals. |
| -pin_levels *pin_count* | Only return pins within *pin_count* pin traversals. |
| -trace_arcs timing | Only trace through timing arcs that are not disabled. |
| -trace_arcs enabled | Only trace through timing arcs that are not disabled. |
| -trace_arcs all | Trace through all arcs, including disabled ones. |

The get_fanin command returns traverses the design from *sink_list* pins, ports or nets backwards and return the fanin pins or instances.

| | |
|---|---|
| **get_fanout** | -from *source_list*<br>[-flat]<br>[-only_cells]<br>[-endpoints_only]<br>[-levels *level_count*]<br>[-pin_levels *pin_count*]<br>[-trace_arcs timing\|enabled\|all] |
| -from *source_list* | List of pins, ports, or nets to find the fanout of. For nets, the fanout of load pins on the nets are returned. |
| -flat | With –flat pins in the fanin at any hierarchy level are returned. Without -flat only pins at the same hierarchy level as the sinks are returned. |
| -only_cells | Return the instances connected to the pins in the fanout. |
| -endpoints_only | Only return pins that are endpoints. |
| -level *level_count* | Only return pins within *level_count* instance traversals. |
| -pin_levels *pin_count* | Only return pins within *pin_count* pin traversals. |
| -trace_arcs timing | Only trace through timing arcs that are not disabled. |
| -trace_arcs enabled | Only trace through timing arcs that are not disabled. |
| -trace_arcs all | Trace through all arcs, including disabled ones. |

The get_fanout command returns traverses the design from *source_list* pins, ports or nets backwards and return the fanout pins or instances.

| | |
|---|---|
| **get_full_name** | *object* |

| | |
|---|---|
| *object* | A library, cell, port, instance, pin or timing arc object. |

Return the name of *object*. Equivalent to [get_property *object* full_name].

| | |
|---|---|
| **get_lib_cells** | [-of_objects *objects*]<br>[-hsc *separator*]<br>[-filter *expr*]<br>[-regexp]<br>[-nocase]<br>[-quiet]<br>*patterns* |
| -of_objects *objects* | A list of instance objects. |
| -hsc *separator* | Character that separates the library name and cell name in *patterns*. Defaults to '/'. |
| -filter *expr* | A filter expression of the form<br>    "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |
| -regexp | Use regular expression matching instead of  glob pattern matching. |
| -nocase | Ignore case when matching.  Only valid with –regexp. |
| -quiet | Do not warn if no matches are found. |
| *patterns* | A list of library cell name patterns of the form library_name/cell_name. |

The get_lib_cells command returns a list of library cells that match *pattern*. The library name can be prepended to the cell name pattern with the *separator* character, which defaults to hierarchy_separator.

| | |
|---|---|
| **get_lib_pins** | [-of_objects *objects*]<br>[-hsc *separator*]<br>[-filter *expr*]<br>[-regexp]<br>[-nocase]<br>[-quiet]<br>*patterns* |
| -of_objects *objects* | A list of library cell objects. |

| | |
|---|---|
| -hsc *separator* | Character that separates the library name, cell name and port name in *pattern*. Defaults to '/'. |
| -filter *expr* | A filter expression of the form<br>   "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |
| -regexp | Use regular expression matching instead of  glob pattern matching. |
| -nocase | Ignore case when matching.  Only valid with –regexp. |
| -quiet | Do not warn if no matches are found. |
| *patterns* | A list of library port name patterns of the form library_name/cell_name/port_name. |

The get_lib_pins command returns a list of library ports that match *pattern*.    Use *separator* to separate the library and cell name patterns from the port name in *pattern*.

| | |
|---|---|
| **get_libs** | [-filter *expr*]<br>[-regexp]<br>[-nocase]<br>[-quiet]<br>*patterns* |
| -filter *expr* | A filter expression of the form<br>   "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |
| -regexp | Use regular expression matching instead of  glob pattern matching. |
| -nocase | Ignore case when matching.  Only valid with –regexp. |
| -quiet | Do not warn if no matches are found. |
| *patterns* | A list of library name patterns. |

The get_libs command returns a list of clocks that match *patterns*.

| **get_nets** | [-hierarchical] |
| | [-hsc *separator*] |
| | [-filter *expr*] |
| | [-regexp] |
| | [-nocase] |
| | [-quiet] |
| | [-of_objects *objects*] |
| | [*patterns*] |

| | |
|---|---|
| -hierarchical | Searches hierarchy levels below the current instance for matches. |
| -hsc *separator* | Character that separates the library name, cell name and port name in *pattern*. Defaults to '/'. |
| -filter *expr* | A filter expression of the form "*property==value*" where *property* is a property supported by the get_property command. See the section "Filter Expressions" for additional forms. |
| -regexp | Use regular expression matching instead of glob pattern matching. |
| -nocase | Ignore case when matching. Only valid with –regexp. |
| -quiet | Do not warn if no matches are found. |
| -of_objects *objects* | The name of a pin or instance, a list of pins returned by get_pins, or a list of instances returned by get_cells. The –hierarchical option cannot be used with –of_objects. |
| *patterns* | A list of net name patterns. |

The get_nets command returns a list of all nets that match *patterns*.

| **get_name** | *object* |
|---|---|
| *object* | A library, cell, port, instance, pin or timing arc object. |

Return the name of *object*. Equivalent to [get_property *object* name].

| **get_pins** | [-hierarchical]<br>[-hsc *separator*]<br>[-filter *expr*]<br>[-regexp]<br>[-nocase]<br>[-quiet]<br>[-of_objects *objects*]<br>[*patterns*] |
|---|---|
| -hierarchical | Searches hierarchy levels below the current instance for matches. |
| -hsc *separator* | Character that separates the library name, cell name and port name in *pattern*. Defaults to '/'. |
| -filter *expr* | A filter expression of the form<br>    "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |
| -nocase | Ignore case when matching.  Only valid with –regexp. |
| -quiet | Do not warn if no matches are found. |
| -of_objects *objects* | The name of a net or instance, a list of nets returned by get_nets, or a list of instances returned by get_cells. The –hierarchical option cannot be used with –of_objects. |
| *patterns* | A list of pin name patterns. |

The get_pins command returns a list of all instance pins that match *patterns*.

A useful idiom to find the driver pin for a net is the following.

    get_pins -of_objects [get_net *net_name*] -filter "direction==output"

| **get_ports** | [-filter *expr*]<br>[-regexp]<br>[-nocase]<br>[-quiet]<br>[-of_objects *objects*]<br>[*patterns*] |
|---|---|
| -filter *expr* | A filter expression of the form<br>    "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |

```
-regexp                    Use regular expression matching instead of  glob pattern
                           matching.

-nocase                    Ignore case when matching.  Only valid with –regexp.

-quiet                     Do not warn if no matches are found.

-of_objects objects        The name of  net or a list of nets returned by get_nets.

patterns                   A list of port name patterns.
```

The `get_ports` command returns a list of all top level ports that match *patterns*.

| **get_property** | [-object_type *object_type*]<br>*object*<br>*property* |
|---|---|

```
-object_type               The type of object when it is specified as a name.
object_type                cell|pin|net|port|clock|library|library_cell|
                           library_pin|timing_arc

object                     An object returned by get_cells, get_pins, get_nets,
                           get_ports, get_clocks, get_libs, get_lib_cells,
                           get_lib_pins, or get_timing_arcs, or object name.
                           -object_type is required if object is a name.

property                   A property name.
```

The properties for different objects types are shown below.

cell (SDC lib_cell)

```
    base_name
    filename
    full_name
    library
    name
```

clock

```
    full_name
    is_generated
    is_propagated
    is_virtual
    name
    period
    sources
```

edge

```
    delay_max_fall
    delay_min_fall
    delay_max_rise
    delay_min_rise
    full_name
    from_pin
    sense
    to_pin
```

instance (SDC cell)

```
    cell
    full_name
    is_buffer
    is_clock_gate
    is_hierarchical
    is_inverter
    is_macro
    is_memory
    liberty_cell
    name
    ref_name
```

liberty_cell (SDC lib_cell)

```
    area
    base_name
    dont_use
    filename
    full_name
    is_buffer
    is_inverter
    is_memory
    library
    name
```

liberty_port (SDC lib_pin)

```
    capacitance
    direction
    drive_resistance
    drive_resistance_max_fall
    drive_resistance_max_rise
    drive_resistance_min_fall
    drive_resistance_min_rise
    full_name
    intrinsic_delay
    intrinsic_delay_max_fall
    intrinsic_delay_max_rise
    intrinsic_delay_min_fall
    intrinsic_delay_min_rise
    is_register_clock
    lib_cell
    name
```

library

    filename (Liberty library only)
    name
    full_name

net

    full_name
    name

path (PathEnd)

    endpoint
    endpoint_clock
    endpoint_clock_pin
    slack
    startpoint
    startpoint_clock
    points

pin

    activity  (activity in transitions per second, duty cycle, origin)
    slew_max_fall
    slew_max_rise
    slew_min_fall
    slew_min_rise
    clocks
    clock_domains
    direction
    full_name
    is_hierarchical
    is_port
    is_register_clock
    lib_pin_name
    name
    slack_max
    slack_max_fall
    slack_max_rise
    slack_min
    slack_min_fall
    slack_min_rise

port

    activity
    slew_max_fall
    slew_max_rise
    slew_min_fall
    slew_min_rise
    direction
    full_name
    liberty_port

```
    name
    slack_max
    slack_max_fall
    slack_max_rise
    slack_min
    slack_min_fall
    slack_min_rise
```

point (PathRef)

```
    arrival
    pin
    required
    slack
```

---

| **get_timing_edges** | [-from *from_pins*]<br>[-to *to_pins*]<br>[-of_objects *objects*]<br>[-filter *expr*]<br>[*patterns*] |
|---|---|
| -f*rom from_pin* | A list of pins. |
| -to *to_pin* | A list of pins. |
| -of_objects *objects* | A list of instances or library cells. The –from and -to options cannot be used with –of_objects. |
| *-filte*r *expr* | A filter expression of the form<br>  "*property==value*"<br>where *property* is a property supported by the get_property command.  See the section "Filter Expressions" for additional forms. |

The get_timing_edges command returns a list of timing edges (arcs) to, from or between pins. The result can be passed to get_property or set_disable_timing.

---

| **group_path** | -name *group_name*<br>[-weight *weight*]<br>[-critical_range *range*]<br>[-from *from_list*<br> \|-rise_from *from_list*<br> \|-fall_from *from_list*]<br>[-through *through_list*]<br>[-rise_through *through_list*]<br>[-fall_through *through_list*]<br>[-to *to_list*<br> \|-rise_to *to_list*<br> \|-fall_to *to_list*] |
|---|---|
| -name *group_name* | The name of the path group. |

---

```
-weight weight            Not supported.

-critical_range range     Not supported.

-from from_list           Group paths from a list of clocks, instances, ports,
                          register clock pins, or latch data pins.

-rise_from from_list      Group  paths from the rising edge of clocks, instances,
                          ports, register clock pins, or latch data pins.

-fall_from from_list      Group paths from the falling edge of clocks, instances,
                          ports, register clock pins, or latch data pins.

-through through_list     Group paths through a list of instances, pins or nets.

-rise_through             Group rising paths through a list of instances, pins or
through_list              nets.

-fall_through             Group falling paths through a list of instances, pins or
through_list              nets.

-to to_list               Group paths to a list of clocks, instances, ports or
                          pins.

-rise_to to_list          Group rising paths to a list of clocks, instances, ports
                          or pins.

-fall_to to_list          Group falling paths to a list of clocks, instances,
                          ports or pins.
```

The group_path command is used to group paths reported by the report_checks command. See set_false_path for a description of allowed *from_list*, *through_list* and *to_list* objects.

| **include** | [-e\|-echo]<br>[-v\|-verbose]<br>*filename*<br>[> *log_filename*]<br>[>> log_*filename*] |
|---|---|

```
-e|-echo                  Print each command before evaluating it.

-v|-verbose               Print each command before evaluating it as well as the
                          result it returns.

filename                  The name of the file containing commands to read.

> log_filename            Redirect command output to log_filename.

>> log_filename           Redirect command output and append log_filename.
```

Read STA/SDC/Tcl commands from *filename*.

The `include` command stops and reports any errors encountered while reading a file unless `sta_continue_on_error` is 1.

| | |
|---|---|
| **link_design** | [-no_black_boxes]<br>[*cell_name*] |
| -no_black_boxes | Do not make empty "black box" cells for instances that reference undefined cells. |
| *cell_name* | The top level module/cell name of the design hierarchy to link. |

Link (elaborate, flatten) the the top level cell *cell_name*. The design must be linked after reading netlist and library files. The default value of *cell_name* is the current design.

The linker creates empty "block box" cells for instances the reference undefined cells when the variable `link_create_black_boxes` is `true`. When `link_create_black_boxes` is `false` an error is reported and the link fails.

The `link_design` command returns 1 if the link succeeds and 0 if it fails.

| | |
|---|---|
| **make_instance** | *inst_path*<br>*lib_cell* |
| *inst_path* | A hierarchical instance name. |
| *lib_cell* | The library cell of the new instance. |

The `make_instance` command makes an instance of library cell *lib_cell*.

| | |
|---|---|
| **make_net** | *net_name_list* |
| *net_name_list* | A list of net names. |

Creates a net for each hierarchical net name.

| | |
|---|---|
| **read_liberty** | [-corner *corner*]<br>[-min]<br>[-max]<br>[-infer_latches]<br>*filename* |
| -corner *corner* | Use the library for process corner *corner* delay calculation. |

| -min | Use library for min delay calculation. |
| -max | Use library for max delay calculation. |
| *filename* | The liberty file name to read. |

The `read_liberty` command reads a Liberty format library file. The first library that is read sets the units used by SDC/TCL commands and reporting. The `include_file` attribute is supported.

Some Liberty libraries do not include `latch` groups for cells that are describe transparent latches. In that situation the `-infer_latches` command flag can be used to infer the latches. The timing arcs required for a latch to be inferred should look like the following:

```
cell (infered_latch) {
  pin(D) {
    direction : input ;
    timing () {
      related_pin : "E" ;
      timing_type : setup_falling ;
    }
    timing () {
      related_pin : "E" ;
      timing_type : hold_falling ;
    }
  }
  pin(E) {
    direction : input;
  }
  pin(Q) {
    direction : output ;
    timing () {
      related_pin : "D" ;
    }
    timing () {
      related_pin : "E" ;
      timing_type : rising_edge ;
    }
  }
}
```

In this example a positive level-sensitive latch is inferred.

Files compressed with gzip are automatically uncompressed.

| **read_saif** | `[-scope `*scope*`]`<br>*filename* |
| --- | --- |
| *scope* | The SAIF scope of the current design to extract simulation data. Typically the test bench name and design under test instance name. Scope levels are separated with '/'. |

| *filename* | The name of the SAIF file to read. |

The `read_saif` command reads a SAIF (Switching Activity Interchange Format) file from a Verilog simulation and extracts pin activities and duty cycles for use in power estimation. Files compressed with gzip are supported. Annotated activities are propagated to the fanout of the annotated pins.

| **read_sdc** | `[-echo]`<br>*filename* |
| --- | --- |
| `-echo` | Print each command before evaluating it. |
| *filename* | SDC command file. |

Read SDC commands from *filename*.

The `read_sdc` command stops and reports any errors encountered while reading a file unless `sta_continue_on_error` is 1.

Files compressed with gzip are automatically uncompressed.

| **read_sdf** | `[-corner` *corner*`]`<br>`[-unescaped_dividers]`<br>*filename* |
| --- | --- |
| `-corner` *corner* | Process corner delays to annotate. |
| `-unescaped_dividers` | With this option path names in the SDF do not have to escape hierarchy dividers when the path name is escaped. For example, the escaped Verilog name "\inst1/inst2 " can be referenced as "inst1/inst2". The correct SDF name is "inst1\/inst2", since the divider does not represent a change in hierarchy in this case. |
| *filename* | The name of the SDF file to read. |

Read SDF delays from a file. The min and max values in the SDF tuples are used to annotate the delays for *corner*. The typical values in the SDF tuples are ignored. If multiple corners are defined `-corner` must be specified.

Files compressed with gzip are automatically uncompressed.

INCREMENT is supported as an alias for INCREMENTAL.

The following SDF statements are not supported.

```
PORT
INSTANCE wildcards
```

| **read_spef** | [-min]<br>[-max]<br>[-path path]<br>[-corner *corner*]<br>[-keep_capacitive_coupling]<br>[-coupling_reduction_factor *factor*]<br>[-reduce]<br>*filename* |
|---|---|
| -min | Annotate parasitics for min delays. |
| -max | Annotate parasitics for max delays. |
| *path* | Hierarchical block instance path to annotate with parasitics. |
| -corner *corner* | Annotate parasitics for one process corner. |
| -keep_capacitive_coupling | Keep coupling capacitors in parasitic networks rather than converting them to grounded capacitors. |
| *-coupling_reduction_facto r*<br>*factor* | Factor to multiply coupling capacitance by when reducing parasitic networks. The default value is 1.0. |
| -reduce | Reduce detailed parasitics and do not save the detailed parastic network. |
| *filename* | The name of the parasitics file to read. |

The `read_spef` command reads a file of net parasitics in SPEF format. Use the `-report_parasitic_annotation` command to check for nets that are not annotated.

Files compressed with gzip are automatically uncompressed.

Separate parasitics can be annotated for corners and min and max paths using the `-corner`, `–min` and `-max` arguments. To use the same parastiics for every corner and for min/max delay calculation read the SPEF without `-corner`, `-min`, and `-max` options.

```
read_spef spef1
```

To use separate parastics for min/max delay, use the `-min`, and `-max` options for each SPEF file.

```
read_spef -min spef1
read_spef -max spef2
```

To use separate parastics for each corner, use the `-corner` option for each SPEF file.

```
read_spef -corner ss spef1
read_spef -corner tt spef2
read_spef -corner ff spef3
```

To use separate parastics for each corner and separate min/max delay calculation, use the `-corner` option along with the `-min`, and `-max` options.

```
read_spef -corner ss -min spef1
read_spef -corner ss -max spef2
read_spef -corner ff -min spef3
read_spef -corner ff -max spef4
```

With the `-reduce` option, the current delay calculator reduces the parastic network to the appropriate type and deletes the parasitic network. This substantially reduces the memory required to store the parasitics.

Coupling capacitors are multiplied by the `–coupling_reduction_factor` when a parasitic network is reduced.

The following SPEF constructs are ignored.

```
*DESIGN_FLOW (all values are ignored)
*S slews
*D driving cell
*I pin capacitances (library cell capacitances are used instead)
*Q r_net load poles
*K r_net load residues
```

If the SPEF file contains triplet values the first value is used.

Parasitic networks (DSPEF) can be annotated on hierarchical blocks using the -path argument to specify the instance path to the block. Parasitic networks in the higher level netlist are stitched together at the hierarchcal pins of the blocks.

| **read_vcd** | `[-scope scope]`<br>`filename` |
|---|---|
| *scope* | The VCD scope of the current design to extract simulation data. Typically the test bench name and design under test instance name. Scope levels are separated with '/'. |
| *filename* | The name of the VCD file to read. |

The `read_vcd` command reads a VCD (Value Change Dump) file from a Verilog simulation and extracts pin activities and duty cycles for use in power estimation. Files compressed with gzip are supported. Annotated activities are propagated to the fanout of the annotated pins.

| | |
|---|---|
| **read_verilog** | *filename* |

| | |
|---|---|
| *filename* | The name of the verilog file to read. |

The `read_verilog` command reads a gate level verilog netlist. After all verilog netlist and Liberty libraries are read the design must be linked with the `link_design` command.

Verilog 2001 module port declaratations are supported. An example is shown below.

```
module top (input in1, in2, clk1, clk2, clk3,
            output out);
```

Files compressed with gzip are automatically uncompressed.

| | |
|---|---|
| **replace_cell** | *instance_list*<br>*replacement_cell* |

| | |
|---|---|
| *instance_list* | A list of instances to swap the cell. |

| | |
|---|---|
| *replacement_cell* | The replacement lib cell. |

The `replace_cell` command changes the cell of an instance. The replacement cell must have the same port list (number, name, and order) as the instance's existing cell for the replacement to be successful.

| | |
|---|---|
| **replace_activity_annotation** | [-report_unannotated]<br>[-report_annotated] |

| | |
|---|---|
| -report_unannotated | Report unannotated pins. |

| | |
|---|---|
| -report_unannotated | Report annotated pins. |

Report a summary of pins that are annotated by `read_vcd`, `read_saif` or `set_power_activity`. Sequential internal pins and hierarchical pins are ignored.

| | |
|---|---|
| **report_annotated_check** | [-setup]<br>[-hold]<br>[-recovery]<br>[-removal]<br>[-nochange]<br>[-width]<br>[-period]<br>[-max_skew]<br>[-max_line *lines*]<br>[-list_annotated]<br>[-list_not_annotated]<br>[-constant_arcs] |
| -setup | Report annotated setup checks. |

```
-hold                 Report annotated hold checks.

-recovery             Report annotated recovery checks.

-removal              Report annotated removal checks.

-nochange             Report annotated nochange checks.

-width                Report annotated width checks.

-period               Report annotated period checks.

-max_skew             Report annotated max skew checks.

-max_line lines       Maximum number of lines listed by the –list_annotated
                      and -list_not_annotated options.

-list_annotated       List annotated timing arcs.

-list_not_annotated   List unannotated timing arcs.

-constant_arcs        Report separate annotation counts for arcs disabled by
                      logic constants (set_logic_one, set_logic_zero).
```

The `report_annotated_check` command reports a summary of SDF timing check annotation. The `-list_annotated` and `–list_not_annotated` options can be used to list arcs that are annotated or not annotated.

```
report_annotated_delay  [-cell]
                        [-net]
                        [-from_in_ports]
                        [-to_out_ports]
                        [-max_lines lines]
                        [-list_annotated]
                        [-list_not_annotated]
                        [-constant_arcs]
-cell                   Report annotated cell delays.

-net                    Report annotated internal net delays.

-from_in_ports          Report annotated delays from input ports.

-to_out_ports           Report annotated delays to output ports.

-max_lines lines        Maximum number of lines listed by the –list_annotated
                        and -list_not_annotated options.

-list_annotated         List annotated timing arcs.
```

```
-list_not_annotated      List unannotated timing arcs.

-constant_arcs           Report separate annotation counts for arcs disabled by
                         logic constants (set_logic_one, set_logic_zero).
```

The `report_annotated_delay` command reports a summary of SDF delay annotation. Without the `-from_in_ports` and `–to_out_ports` options arcs to and from top level ports are not reported. The `-list_annotated` and `–list_not_annotated` options can be used to list arcs that are annotated or not annotated.

```
report_checks               [-from from_list
                             |-rise_from from_list
                             |-fall_from from_list]
                            [-through through_list
                             |-rise_through through_list
                             |-fall_through through_list]
                            [-to to_list
                             |-rise_to to_list
                             |-fall_to to_list]
                            [-unconstrained]
                            [-path_delay min|min_rise|min_fall
                                        |max|max_rise|max_fall
                                        |min_max]
                            [-group_path_count path_count]
                            [-endpoint_path_count endpoint_path_count]
                            [-unique_paths_to_endpoint]
                            [-corner corner]
                            [-slack_max max_slack]
                            [-slack_min min_slack]
                            [-sort_by_slack]
                            [-path_group groups]
                            [-format end|full|short|summary
                                        |full_clock|full_clock_expanded
                                        |json]
                            [-fields fields]
                            [-digits digits]
                            [-no_line_split]
                            [> filename]
                            [>> filename]

-from from_list             Report paths from a list of clocks, instances, ports,
                            register clock pins, or latch data pins.

-rise_from from_list        Report  paths from the rising edge of clocks,
                            instances, ports, register clock pins, or latch data
                            pins.

-fall_from from_list        Report paths from the falling edge of clocks,
                            instances, ports, register clock pins, or latch data
                            pins.

-through through_list       Report paths through a list of instances, pins or nets.
```

```
-rise_through           Report rising paths through a list of instances, pins
through_list            or nets.

-fall_through           Report falling paths through a list of instances, pins
through_list            or nets.

-to to_list             Report paths to a list of clocks, instances, ports or
                        pins.

-rise_to to_list        Report rising paths to a list of clocks, instances,
                        ports or pins.

-fall_to to_list        Report falling paths to a list of clocks, instances,
                        ports or pins.

-unconstrained          Report unconstrained paths also. The unconstrained path
                        group is not reported without this option.

-path_delay min         Report min path (hold) checks.

-path_delay min_rise    Report min path (hold) checks for rising endpoints.

-path_delay min_fall    Report min path (hold) checks for falling endpoints.

-path_delay max         Report max path (setup) checks.

-path_delay max_rise    Report max path (setup) checks for rising endpoints.

-path_delay max_fall    Report max path (setup) checks for falling endpoints.

-path_delay min_max     Report max and max path (setup and hold) checks.

-group_path_count       The number of paths to report in each path group. The
path_count              default is 1.

-endpoint_path_count    The number of paths to report for each endpoint. The
endpoint_path_count     default is 1.

-unique_paths_to_endpoi When multiple paths to an endpoint are specified with
nt                      -endpoint_path_count many of the paths may differ only
                        in the rise/fall edges of the pins in the paths. With
                        this option only the worst path through the set of pis
                        is reported.

-corner corner          Report paths for one process corner. The default is to
                        report paths for all process corners.

-slack_max max_slack    Only report paths with less slack than max_slack.

-slack_min min_slack    Only report paths with more slack than min_slack.
```

| | |
|---|---|
| -sort_by_slack | Sort paths by slack rather than slack grouped by path group. |
| -path_group *groups* | List of path groups to report. The default is to report all path groups. |
| -format end | Report path ends in one line with delay, required time and slack. |
| -format full | Report path start and end points and the path. This is the default path type. |
| -format full_clock | Report path start and end points, the path, and the source and and target clock paths. |
| -format full_clock_expanded | Report path start and end points, the path, and the source and and target clock paths. If the clock is generated and propagated, the path from the clock source pin is also reported. |
| -format short | Report only path start and end points. |
| -format summary | Report only path ends with delay. |
| -format json | Report in json format. -fields is ignored. |
| -fields *fields* | List of capacitance\|slew\|input_pins\|hierarcial_pins\|nets\|fanout\|src_attr |
| -digits *digits* | The number of digits after the decimal point to report. The default value is the variable sta_report_default_digits. |
| -no_line_splits | Do not split long lines into multiple lines. |

The report_checks command reports paths in the design. Paths are reported in groups by capture clock, unclocked path delays, gated clocks and unconstrained.

See set_false_path for a description of allowed *from_list*, *through_list* and *to_list* objects.

```
report_check_types    [-violators]
                      [-verbose]
                      [-format slack_only|end]
                      [-max_delay]
                      [-min_delay]
                      [-recovery]
                      [-removal]
                      [-clock_gating_setup]
                      [-clock_gating_hold]
                      [-max_slew]
                      [-min_slew]
                      [-min_pulse_width]
                      [-min_period]
                      [-digits digits]
                      [-no_split_lines]
                      [> filename]
                      [>> filename]
```

-violators            Report all violated timing and design rule constraints.

-verbose              Use a verbose output format.

-format slack_only    Report the minumum slack for each timing check.

-format end           Report the endpoint for each check.

-max_delay            Report setup and max delay path delay constraints.

-min_delay            Report hold and min delay path delay constraints.

-recovery             Report asynchronous recovery checks.

-removal              Report asynchronous removal checks.

-clock_gating_setup   Report gated clock enable setup checks.

-clock_gating_hold    Report gated clock hold setup checks.

-max_slew             Report max transition design rule checks.

-max_skew             Report max skew design rule checks.

-min_pulse_width      Report min pulse width design rule checks.

-min_period           Report min period design rule checks.

-min_slew             Report min slew design rule checks.

-digits digits        The number of digits after the decimal point to report.
                      The default value is the variable
                      sta_report_default_digits.

```
-no_split_lines          Do not split long lines into multiple lines.
```

The `report_check_types` command reports the slack for each type of timing and design rule constraint. The keyword options allow a subset of the constraint types to be reported.

| | |
|---|---|
| **report_clock_latency** | `[-clock clocks]`<br>`[-include_internal_latency]`<br>`[-digits digits]` |
| `-clock clocks` | The clocks to report. |
| `-include_internal_latency` | Include internal clock latency from liberty min/max_clock_tree_path timing groups. |
| `-digits digits` | The number of digits to report for delays. |

Report the clock network latency.

| | |
|---|---|
| **report_clock_min_period** | `[-clocks clocks]`<br>`[-include_port_paths]` |
| `-clocks clocks` | The clocks to report. |
| `-include_port_paths` | Include paths from input port and to output ports. |

Report the minimum period and maximum frequency for *clocks*. If the `-clocks` argument is not specified all clocks are reported. The minimum period is determined by examining the smallest slack paths between registers the rising edges of the clock or between falling edges of the clock. Paths between different clocks, different clock edges of the same clock, level sensitive latches, or paths constrained by set_multicycle_path, set_max_path are not considered.

| | |
|---|---|
| **report_clock_properties** | `[clock_names]` |
| `clock_names` | List of clock names to report. |

The `report_clock_properties` command reports the period and rise/fall edge times for each clock that has been defined.

| | |
|---|---|
| **report_clock_skew** | `[-setup|-hold]`<br>`[-clock clocks]`<br>`[-include_internal_latency]`<br>`[-digits digits]` |
| `-setup` | Report skew for setup checks. |

| | |
|---|---|
| -hold | Report skew for hold checks. |
| -clock *clocks* | The clocks to report. |
| -include_internal_latenc<br>y | Include internal clock latency from liberty min/max_clock_tree_path timing groups. |
| -digits *digits* | The number of digits to report for delays. |

Report the maximum difference in clock arrival between every source and target register that has a path between the source and target registers.

| **report_dcalc** | [-from *from_pin*]<br>[-to *to_pin*]<br>[-corner *corner*]<br>[-min]<br>[-max]<br>[-digits *digits*]<br>[> *filename*]<br>[>> *filename*] |
|---|---|
| -from *from_pin* | Report delay calculations for timing arcs from instance input pin *from_pin*. |
| -to *to_pin* | Report delay calculations for timing arcs to instance output pin to_*pin*. |
| -corner *corner* | Report paths for process *corner*. The -corner keyword is required if more than one process corner is defined. |
| -min | Report delay calculation for min delays. |
| -max | Report delay calculation for max delays. |
| -digits *digits* | The number of digits after the decimal point to report. The default is sta_report_default_digits. |

The report_dcalc command shows how the delays between instance pins are calculated. It is useful for debugging problems with delay calculation.

| **report_disabled_edges** |
|---|

The report_disabled_edges command reports disabled timing arcs along with the reason they are disabled. Each disabled timing arc is reported as the instance name along with the from and to ports of the arc. The disable reason is shown next. Arcs that are disabled with set_disable_timing are reported with constraint as the reason. Arcs that are disabled by constants are reported with constant as the reason along with the constant instance pin and value. Arcs that are disabled to break combinational feedback loops are reported with loop as the reason.

```
> report_disabled_edges
u1 A B constant B=0
```

| **report_edges** | `[-from `*`from_pin`*`]`<br>`[-to `*`to_pin`*`]` |
|---|---|
| -from *from_pin* | Report edges/timing arcs from pin *from_pin*. |
| -to *to_pin* | Report edges/timing arcs to pin to_*pin*. |

Report the edges/timing arcs and their delays in the timing graph from/to/between pins.

| **report_instance** | *instance_path*<br>`[> `*`filename`*`]`<br>`[>> `*`filename`*`]` |
|---|---|
| *instance_path* | Hierarchical path to a instance. |

| **report_lib_cell** | *cell_name*<br>`[> `*`filename`*`]`<br>`[>> `*`filename`*`]` |
|---|---|
| *cell_name* | The name of a library cell. |

Describe the liberty library cell *cell_name.*

| **report_net** | `[-digits `*`digits`*`]`<br>*net_path*<br>`[> `*`filename`*`]`<br>`[>> `*`filename`*`]` |
|---|---|
| -digits *digits* | The number of digits after the decimal point to report. The default value is the variable sta_report_default_digits. |
| *net_path* | Hierarchical path to a net. |

Report the connections and capacitance of a net.

| **report_parasitic_annotation** | `[-report_unannotated]`<br>`[> `*`filename`*`]`<br>`[>> `*`filename`*`]` |
|---|---|
| -report_unannotated | Report unannotated and partially annotated nets. |

Report SPEF parasitic annotation completeness.

| | |
|---|---|
| **report_power** | [-instances *instances*]<br>[-highest_power_instances *count*]<br>[-digits *digits*]<br>[> *filename*]<br>[>> *filename*] |
| -instances *instances* | Report the power for each instance of *instances*. If the instance is hierarchical the total power for the instances inside the hierarchical instance is reported. |
| -highest_power_instanc es count | Report the power for the *count* highest power instances. |
| -digits *digits* | The number of digits after the decimal point to report. The default value is the variable sta_report_default_digits. |

The report_power command uses static power analysis based on propagated or annotated pin activities in the circuit using Liberty power models. The internal, switching, leakage and total power are reported. Design power is reported separately for combinational, sequential, macro and pad groups. Power values are reported in watts.

The read_vcd or read_saif commands can be used to read activities from a file based on simulation. If no simulation activities are available, the set_power_activity command should be used to set the activity of input ports or pins in the design. The default input activity and duty for inputs are 0.1 and 0.5 respectively. The activities are propagated from annotated input ports or pins through gates and used in the power calculations.

| Group | Internal Power | Switching Power | Leakage Power | Total Power | |
|---|---|---|---|---|---|
| Sequential | 3.29e-06 | 3.41e-08 | 2.37e-07 | 3.56e-06 | 92.4% |
| Combinational | 1.86e-07 | 3.31e-08 | 7.51e-08 | 2.94e-07 | 7.6% |
| Macro | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.0% |
| Pad | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.0% |
| Total | 3.48e-06 | 6.72e-08 | 3.12e-07 | 3.86e-06 | 100.0% |
| | 90.2% | 1.7% | 8.1% | | |

| | |
|---|---|
| **report_pulse_width_chec ks** | [-verbose]<br>[-digits *digits*]<br>[-no_line_splits]<br>[*pins*]<br>[> *filename*]<br>[>> *filename*] |
| -verbose | Use a verbose output format. |

```
-digits digits          The number of digits after the decimal point to report.
                        The default value is the variable
                        sta_report_default_digits.

-no_line_splits

pins                    List of pins or ports to report.
```

The `report_pulse_width_checks` command reports min pulse width checks for pins in the clock network.
If *pins* is not specified all clock network pins are reported.

| | |
|---|---|
| **report_slews** | `[-corner corner]`<br>*pin* |

```
-corner corner          Report paths for process corner. The -corner keyword is
                        required if more than one process corner is defined.

pin
```

Report the slews at *pin*

| | |
|---|---|
| **report_tns** | `[-min]`<br>`[-max]`<br>`[-digits digits]` |

```
-max                    Report the total max/setup slack.

-min                    Report the total min/hold slack.

-digits digits          The number of digits after the decimal point to report.
                        The default value is the variable
                        sta_report_default_digits.
```

Report the total negative slack.

| |
|---|
| **report_units** |

Report the units used for command arguments and reporting.

```
report_units
 time 1ns
 capacitance 1pF
 resistance 1kohm
 voltage 1v
 current 1A
 power 1pW
 distance 1um
```

| report_wns | [-min]<br>[-max]<br>[-digits *digits*] |
|---|---|
| -max | Report the worst max/setup slack. |
| -min | Report the worst min/hold slack. |
| -digits *digits* | The number of digits after the decimal point to report. The default value is the variable sta_report_default_digits. |

Report the worst negative slack. If the worst slack is posiive, zero is reported.

| report_worst_slack | [-min]<br>[-max]<br>[-digits *digits*] |
|---|---|
| -max | Report the worst max/setup slack. |
| -min | Report the worst min/hold slack. |
| -digits *digits* | The number of digits after the decimal point to report. The default value is the variable sta_report_default_digits. |

| set_assigned_check | -setup\|-hold\|-recovery\|-removal<br>[-rise]<br>[-fall]<br>[-corner *corner*]<br>[-min]<br>[-max]<br>[-from *from_pins*]<br>[-to *to_pins*]<br>[-clock rise\|fall]<br>[-cond sdf_cond]<br>[-worst]<br>*margin* |
|---|---|
| -setup | Annotate setup timing checks. |
| -hold | Annotate hold timing checks. |
| -recovery | Annotate recovery timing checks. |
| -removal | Annotate removal timing checks. |

| | |
|---|---|
| -rise | Annotate rising delays. |
| -fall | Annotate falling delays. |
| -corner *corner* | The name of a process corner. The -corner keyword is required if more than one process corner is defined. |
| -min | Annotate the minimum value of the process corner. |
| -max | Annotate the maximum value of the process corner. |
| -from *from_pins* | A list of pins for the clock. |
| -to *to_pins* | A list of pins for the data. |
| -clock rise\|fall | The timing check clock pin transition. |
| *margin* | The timing check margin. |

The set_assigned_check command is used to annotate the timing checks between two pins on an instance. The annotated delay overrides the calculated delay. This command is a interactive way to back-annotate delays like an SDF file.

| | |
|---|---|
| **set_assigned_delay** | -cell\|-net<br>[-rise]<br>[-fall]<br>[-corner *corner*]<br>[-min]<br>[-max]<br>[-from *from_pins*]<br>[-to *to_pins*]<br>*delay* |
| -cell | Annotate the delays between two pins on an instance. |
| -net | Annotate the delays between two pins on a net. |
| -rise | Annotate the rising delays. |
| -fall | Annotate the falling delays. |
| -corner *corner* | The name of a process corner. The -corner keyword is required if more than one process corner is defined. |
| -min | Annotate the minimum delays. |
| -max | Annotate the maximum delays. |
| -from *from_pins* | A list of pins. |

| | |
|---|---|
| -to *to_pins* | A list of pins. |
| *delay* | The delay between *from_pins* and *to_pins*. |

The `set_assigned_delay` command is used to annotate the delays between two pins on an instance or net. The annotated delay overrides the calculated delay. This command is a interactive way to back-annotate delays like an SDF file.

Use the `-corner` keyword to specify a process corner. The `-corner` keyword is required if more than one process corner is defined.

| | |
|---|---|
| **set_assigned_transition** | `[-rise]`<br>`[-fall]`<br>`[-corner` *corner*`]`<br>`[-min]`<br>`[-max]`<br>*slew*<br>*pin_list* |
| -rise | Annotate the rising transition. |
| -fall | Annotate the falling transition. |
| -corner *corner* | Annotate delays for process *corner*. |
| -min | Annotate the minimum transition time. |
| -max | Annotate the maximum transition time. |
| *slew* | The pin transition time. |
| *pin_list* | A list of pins. |

The `set_assigned_transition` command is used to annotate the transition time (slew) of a pin. The annotated transition time overrides the calculated transition time.

| | |
|---|---|
| **set_case_analysis** | `0\|1\|zero\|one\|rise\|rising\|fall\|falling`<br>*port_or_pin_list* |
| *port_or_pin_list* | A list of ports or pins. |

The `set_case_analysis` command sets the signal on a port or pin to a constant logic value. No paths are propagated from constant pins. Constant values set with the `set_case_analysis` command are propagated through downstream gates.

Conditional timing arcs with `mode` groups are controlled by logic values on the instance pins.

| | |
|---|---|
| **set_clock_gating_check** | [-setup *setup_time*] |
| | [-hold *hold_time*] |
| | [-rise] |
| | [-fall] |
| | [-high] |
| | [-low] |
| | [*objects*] |

| | |
|---|---|
| -setup *setup_time* | Clock enable setup margin. |
| -hold *hold_time* | Clock enable hold margin. |
| -rise | The setup/hold margin is for the rising edge of the clock enable. |
| -fall | The setup/hold margin is for the falling edge of the clock enable. |
| -high | The gating clock is active high (pin and instance objects only). |
| -low | The gating clock is active low (pin and instance objects only). |
| *objects* | A list of clocks, instances, pins or ports. |

The `set_clock_gating_check` command is used to add setup or hold timing checks for data signals used to gate clocks.

If no objects are specified the setup/hold margin is global and applies to all clock gating circuits in the design. If neither of the -rise and -fall options are used the setup/hold margin applies to the rising and falling edges of the clock gating signal.

Normally the library cell function is used to determine the active state of the clock. The clock is active high for AND/NAND functions and active low for OR/NOR functions. The -high and -low options are used to specify the active state of the clock for other cells, such as a MUX.

If multiple `set_clock_gating_check` commands apply to a clock gating instance he priority of the commands is shown below (highest to lowest priority).

```
clock enable pin
instance
clock pin
clock
global
```

| set_clock_groups | [-name *name*] |
| --- | --- |
| | [-logically_exclusive] |
| | [-physically_exclusive] |
| | [-asynchronous] |
| | [-allow_paths] |
| | -group *clocks* |

| -name *name* | The clock group name. |
| --- | --- |
| -logically_exclusive | The clocks in different groups do not interact logically but can be physically present on the same chip. Paths between clock groups are considered for noise analysis. |
| -physically_exclusive | The clocks in different groups cannot be present at the same time on a chip. Paths between clock groups are not considered for noise analysis. |
| -asynchronous | The clock groups are asynchronous. Paths between clock groups are considered for noise analysis. |
| -allow_paths | |
| *clocks* | A list of clocks in the group. |

The set_clock_groups command is used to deifine groups of clocks that interact with each other. Clocks in different groups do not interact and paths between them are not reported. Use a –group argument for each clock group.

| set_clock_latency | [-source] |
| --- | --- |
| | [-clock *clock*] |
| | [-rise] |
| | [-fall] |
| | [-min] |
| | [-max] |
| | *delay* |
| | *objects* |

| -source | The latency is at the clock source. |
| --- | --- |
| -clock *clock* | If multiple clocks are defined at a pin this use this option to specify the latency for a specific clock. |
| -rise | The latency is for the rising edge of the clock. |
| -fall | The latency is for the falling edge of the clock. |
| -min | *delay* is the minimum latency. |
| -max | *delay* is the maximum latency. |

| *delay* | Clock source or insertion delay. |
| *objects* | A list of clocks, pins or ports. |

The set_clock_latency command describes expected delays of the clock tree when analyzing a design using ideal clocks. Use the -source option to specify latency at the clock source, also known as insertion delay. Source latency is delay in the clock tree that is external to the design or a clock tree internal to an instance that implements a complex logic function.

| **set_clock_transition** | [-rise]<br>[-fall]<br>[-min]<br>[-max]<br>*transition*<br>*clocks* |
| --- | --- |
| -rise | Set the transition time for the rising edge of the clock. |
| -fall | Set the transition time for the falling edge of the clock. |
| -min | Set the min transition time. |
| -max | Set the min transition time. |
| *transition* | Clock transition time (slew). |
| *clocks* | A list of clocks. |

The set_clock_transition command describes expected transition times of the clock tree when analzying a design using ideal clocks.

| **set_clock_uncertainty** | [-from\|-rise_from\|-fall_from *from_clock*]<br>[-to\|-rise_to\|-fall_to *to_clock*]<br>[-rise]<br>[-fall]<br>[-setup]<br>[-hold]<br>*uncertainty*<br>[*objects*] |
| --- | --- |
| -from *from_clock* | Inter-clock uncertainty source clock. |
| -to *to_clock* | Inter-clock uncertainty target clock. |

| | |
|---|---|
| -rise | Inter-clock target clock rise edge, alternative to -rise_to.Inter-clock target clock rise edge, alternative to -rise_to. |
| -fall | Inter-clock target clock rise edge, alternative to -fall_to. |
| -setup | *uncertainty* is for setup checks. |
| -hold | *uncertainty* is for hold checks. |
| *uncertainty* | Clock uncertainty. |
| *objects* | A list of clocks, ports or pins. |

The `set_clock_uncertainty` command specifies the uncertainty or jitter in a clock. The uncertainty for a clock can be specified on its source pin or port, or the clock itself.

```
set_clock_uncertainty .1 [get_clock clk1]
```

Inter-clock uncertainty between the source and target clocks of timing checks is specified with the `-from|-rise_from|-fall_from` and `-to|-rise_to|-fall_to` arguments .

```
set_clock_uncertainty -from [get_clock clk1] -to [get_clocks clk2] .1
```

The following commands are equivalent.

```
set_clock_uncertainty -from [get_clock clk1] -rise_to [get_clocks clk2] .1
set_clock_uncertainty -from [get_clock clk1] -to [get_clocks clk2] -rise .1
```

| | |
|---|---|
| **set_cmd_units** | [-capacitance *cap_unit*]<br>[-resistance *res_unit*]<br>[-time *time_unit*]<br>[-voltage *voltage_unit*]<br>[-current *current_unit*]<br>[-power *power_unit*]<br>[-distance *distance_unit*] |
| -capacitance *cap_unit* | The capacitance scale factor followed by 'f'. |
| -resistance *res_unit* | The resistance scale factor followed by 'ohm'. |
| -time *time_unit* | The time scale factor followed by 's'. |
| -voltage *voltage_unit* | The voltage scale factor followed by 'v'. |
| -current *current_unit* | The current scale factor followed by 'A'. |
| -power *power_unit* | The power scale factor followed by 'w'. |

```
-distance              The distance scale factor followed by 'm'.
distance_unit
```

The `set_cmd_units` command is used to change the units used by the STA command interpreter when parsing commands and reporting results. The default units are the units specified in the first Liberty library file that is read.

Units are specified as a scale factor followed by a unit name. The scale factors are as follows.

```
M 1E+6
k 1E+3
m 1E-3
u 1E-6
n 1E-9
p 1E-12
f 1E-15
```

An example of the `set_units` command is shown below.

```
set_cmd_units -time ns -capacitance pF -current mA -voltage V
              -resistance kOhm -distance um
```

| **set_data_check** | [-from\|-rise_from\|-fall_from *from_pin*]<br>[-to\|-rise_to\|-fall_to *to_pin*]<br>[-setup]<br>[-hold]<br>[-clock *clock*]<br>*margin* |
|---|---|
| -from *from_pin* | A pin used as the timing check reference. |
| -to *to_pin* | A pin that the setup/hold check is applied to. |
| -setup | Add a setup timing check. |
| -hold | Add a hold timing check. |
| -clock *clock* | The setup/hold check clock. |
| *margin* | The setup or hold time margin. |

The `set_data_check` command is used to add a setup or hold timing check between two pins.

| **set_disable_inferred_clock_gati ng** | *objects* |
|---|---|
| *objects* | A list of clock gating instances, clock gating pins, or clock enable pins. |

The `set_disable_inferred_clock_gating` command disables clock gating checks on a clock gating instance, clock gating pin, or clock gating enable pin.

| `set_disable_timing` | `[-from `*`from_port`*`]`<br>`[-to `*`to_port`*`]`<br>*`objects`* |
|---|---|
| `-from `*`from_port`* | |
| `-to `*`to_port`* | |
| *`objects`* | A list of instances, ports, pins, cells, cell/port, or library/cell/port. |

The `set_disable_timing` command is used to disable paths though pins in the design. There are many different forms of the command depending on the objects specified in *`objects`*.

All timing paths though an instance are disabled when *`objects`* contains an instance. Timing checks in the instance are *not* disabled.

```
set_disable_timing u2
```

The `-from` and `-to` options can be used to restrict the disabled path to those from, to or between specific pins on the instance.

```
set_disable_timing -from A u2
set_disable_timing -to Z u2
set_disable_timing -from A -to Z u2
```

A list of top level ports or instance pins can also be disabled.

```
set_disable_timing u2/Z
set_disable_timing in1
```

Timing paths though all instances of a library cell in the design can be disabled by naming the cell using a hierarchy separator between the library and cell name. Paths from or to a cell port can be disabled with the `-from` and `-to` options or a port name after library and cell names.

```
set_disable_timing liberty1/snl_bufx2
set_disable_timing -from A liberty1/snl_bufx
set_disable_timing -to Z liberty1/snl_bufx
set_disable_timing liberty1/snl_bufx2/A
```

| `set_drive` | `[-rise]`<br>`[-fall]`<br>`[-max]`<br>`[-min]`<br>*`resistance`*<br>*`ports`* |
|---|---|

| | |
|---|---|
| -rise | Set the drive rise resistance. |
| -fall | Set the drive fall resistance. |
| -max | Set the maximum resistance. |
| -min | Set the minimum resistance. |
| *resistance* | The external drive resistance. |
| *ports* | A list of ports. |

The set_drive command describes the resistance of an input port external driver.

| | |
|---|---|
| **set_driving_cell** | [-lib_cell *cell_name*]<br>[-library *library*]<br>[-rise]<br>[-fall]<br>[-min]<br>[-max]<br>[-pin *pin*]<br>[-from_pin *from_pin*]<br>[-input_transition_rise *trans_rise*]<br>[-input_transition_fall *trans_fall*]<br>*ports* |
| -lib_cell *cell_name* | The driving cell. |
| -library *library* | The driving cell library. |
| -rise | Set the driving cell for a risingn edge. |
| -fall | Set the driving cell for a falling edge. |
| -max | Set the driving cell for max delays. |
| -min | Set the driving cell for min delays. |
| -pin *pin* | The output port of the driving cell. |
| -from_pin *from_pin* | Use timng arcs from *from_pin* to the output pin. |
| -input_transition_rise *trans_rise* | The transition time for a rising input at *from_pin*. |
| -input_transition_fall *trans_fall* | The transition time for a falling input at *from_pin*. |
| *ports* | A list of ports. |

The `set_driving_cell` command describes an input port external driver.

| set_false_path | [-setup]<br>[-hold]<br>[-rise]<br>[-fall]<br>[-from *from_list*]<br>[-rise_from *from_list*]<br>[-fall_from *from_list*]<br>[-through *through_list*]<br>[-rise_through *through_list*]<br>[-fall_through *through_list*]<br>[-to *to_list*]<br>[-rise_to *to_list*]<br>[-fall_to *to_list*]<br>[-reset_path] |
|---|---|
| -setup | Apply to setup checks. |
| -hold | Apply to hold checks. |
| -rise | Apply to rising path edges. |
| -fall | Apply to falling path edges. |
| -reset_path | Remove any matching set_false_path, set_multicycle_path, set_max_delay, set_min_delay exceptions first. |
| -from *from_list* | A list of clocks, instances, ports or pins. |
| -through *through_list* | A list of instances, pins or nets. |
| -to *to_list* | A list of clocks, instances, ports or pins. |

The `set_false_path` command disables timing along a path from, through and to a group of design objects.

Objects in *from_list* can be clocks, register/latch instances, or register/latch clock pins. The -rise_from and -fall_from keywords restrict the false paths to a specific clock edge.

Objects in *through_list* can be nets, instances, instance pins, or hierarchical pins,. The -rise_through and -fall_through keywords restrict the false paths to a specific path edge that traverses through the object.

Objects in *to_list* can be clocks, register/latch instances, or register/latch clock pins. The -rise_to and -fall_to keywords restrict the false paths to a specific transition at the path end.

| set_fanout_load | *fanout*<br>*port_list* |
|---|---|

This command is ignored.

| | |
|---|---|
| **set_hierarchy_separato r** | *separator* |
| *separator* | Character used to separate hierarchical names. |

Set the character used to separate names in a hierarchical instance, net or pin name. This separator is used by the command interpreter to read arguments and print results. The default separator is '/'.

| | |
|---|---|
| **set_ideal_latency** | [-rise] [-fall] [-min] [-max] *delay objects* |

The set_ideal_latency command is parsed but ignored.

| | |
|---|---|
| **set_ideal_network** | [-no_propagation] *objects* |

The set_ideal_network command is parsed but ignored.

| | |
|---|---|
| **set_ideal_transition** | [-rise] [-fall] [-min] [-max] *transition_time objects* |

The set_ideal_transition command is parsed but ignored.

| | |
|---|---|
| **set_input_delay** | [-rise]<br>[-fall]<br>[-max]<br>[-min]<br>[-clock *clock*]<br>[-clock_fall]<br>[-reference_pin *ref_pin*]<br>[-source_latency_included]<br>[-network_latency_included]<br>[-add_delay]<br>*delay*<br>*port_pin_list* |
| -rise | Set the arrival time for the rising edge of the input. |
| -fall | Set the arrival time for the falling edge of the input. |
| -max | Set the maximum arrival time. |
| -min | Set the minimum arrival time. |
| -clock *clock* | The arrival time is from *clock*. |

| | |
|---|---|
| -clock_fall | The arrival time is from the falling edge of *clock.* |
| -reference_pin *ref_pin* | The arrival time is with respect to the clock that arrives at *ref_pin*. |
| -source_latency_included | D no add the clock source latency (insertion delay) to the delay value. |
| -network_latency_included | Do not add the clock latency to the delay value when the clock is ideal. |
| -add_delay | Add this arrival to any existing arrivals. |
| *delay* | The arrival time after *clock*. |
| *pin_port_list* | A list of pins or ports. |

The set_input_delay command is used to specify the arrival time of an input signal.

The following command sets the min, max, rise and fall times on the in1 input port 1.0 time units after the rising edge of clk1.

```
set_input_delay -clock clk1 1.0 [get_ports in1]
```

Use multiple commands with the -add_delay option to specifiy separate arrival times for min, max, rise and fall times or multiple clocks. For example, the following specifies separate arrival times with respect to clocks clk1 and clk2.

```
set_input_delay -clock clk1 1.0 [get_ports in1]
set_input_delay -add_delay -clock clk2 2.0 [get_ports in1]
```

The –reference_pin option is used to specify an arrival time with respect to the arrival on a pin in the clock network. For propagated clocks, the input arrival time is relative to the clock arrival time at the reference pin (the clock source latency and network latency from the clock source to the reference pin). For ideal clocks, input arrival time is relative to the reference pin clock source latency. With the -clock_fall flag the arrival time is relative to the falling transition at the reference pin. If no clocks arrive at the reference pin the set_input_delay command is ignored. If no -clock is specified the arrival time is with respect to all clocks that arrive at the reference pin. The -source_latency_included and -network_latency_included options cannot be used with -reference_pin.

Paths from inputs that do not have an arrival time defined by set_input_delay are not reported. Set the sta_input_port_default_clock variable to 1 to report paths from inputs without a set_input_delay.

| | |
|---|---|
| **set_input_transition** | [-rise]<br>[-fall]<br>[-max]<br>[-min]<br>*transition*<br>*port_list* |

| | |
|---|---|
| -rise | Set the rising edge transition. |
| -fall | Set the falling edge transition. |
| -max | Set the minimum transition time. |
| -min | Set the maximum transition time. |
| *transition* | The transition time (slew). |
| *port_list* | A list of ports. |

The `set_input_transition` command is used to specify the transition time (slew) of an input signal.

| | |
|---|---|
| **set_level_shifter_strateg y** | [-rule *rule_type*] |

This command is parsed and ignored by timing analysis.

| | |
|---|---|
| **set_level_shifter_thresho ld** | [-voltage *voltage*] |

This command is parsed and ignored by timing analysis.

| | |
|---|---|
| **set_load** | [-rise]<br>[-fall]<br>[-max]<br>[-min]<br>[-subtract_pin_load]<br>[-pin_load]<br>[-wire_load]<br>*capacitance*<br>*objects* |
| -rise | Set the external port rising capacitance (ports only). |
| -fall | Set the external port falling capacitance (ports only). |
| -max | Set the max capacitance. |
| -min | Set the min capacitance. |
| -subtract_pin_load | Subtract the capacitance of all instance pins connected to the net from *capacitance* (nets only). If the resulting capacitance is negative, zero is used. Pin capacitances are ignored by delay calculatioin when this option is used. |

| | |
|---|---|
| -pin_load | *capcitance* is external instance pin capacitance (ports only). |
| -wire_load | *capcitance* is external wire capacitance (ports only). |
| *capacitance* | The capacitance, in library capacitance units. |
| *objects* | A list of nets or ports. |

The `set_load` command annotates wire capacitance on a net or external capacitance on a port. There are four different uses for the `set_load` commanc:

```
set_load -wire_load port  external port wire capacitance
set_load -pin_load port   external port pin capacitance
set_load port             same as -pin_load
set_load net              net wire capacitance
```

External port capacitance can be annotated separately with the `-pin_load` and `-wire_load` options. Without the `-pin_load` and `-wire_load` options pin capacitance is annotated.

When annnotating net wire capacitance with the `-subtract_pin_load` option the capacitance of all instance pins connected to the net is subtracted from *capacitance*. Setting the capacitance on a net overrides SPEF parasitics for delay calculation.

| | |
|---|---|
| **set_logic_dc** | *port_list* |
| *port_pin_list* | List of ports or pins. |

Set a port or pin to a constant unknown logic value. No paths are propagated from constant pins.

| | |
|---|---|
| **set_logic_one** | *port_list* |
| *port_pin_list* | List of ports or pins. |

Set a port or pin to a constant logic one value. No paths are propagated from constant pins. Constant values set with the `set_logic_one` command are **not** propagated through downstream gates.

| | |
|---|---|
| **set_logic_zero** | *port_list* |
| *port_pin_list* | List of ports or pins. |

Set a port or pin to a constant logic zero value. No paths are propagated from constant pins. Constant values set with the `set_logic_zero` command are **not** propagated through downstream gates.

| set_max_area | *area* |
|---|---|

*area*

The set_max_area command is ignored during timing but is included in SDC files that are written.

| set_max_capacitance | *capacitance*<br>*objects* |
|---|---|

*capacitance*

| *objects* | List of ports or cells. |
|---|---|

The set_max_capacitance command is ignored during timing but is included in SDC files that are written.

| set_max_delay | [-rise]<br>[-fall]<br>[-from *from_list*]<br>[-rise_from *from_list*]<br>[-fall_from *from_list*]<br>[-through *through_list*]<br>[-rise_through *through_list*]<br>[-fall_through *through_list*]<br>[-to *to_list*]<br>[-rise_to *to_list*]<br>[-fall_to *to_list*]<br>[-reset_path]<br>[-ignore_clock_latency]<br>*delay* |
|---|---|

| -rise | Set max delay for rising paths. |
|---|---|
| -fall | Set max delay for  falling paths. |
| -from *from_list* | A list of clocks, instances, ports or pins. |
| -through *through_list* | A list of instances, pins or nets. |
| -to *to_list* | A list of clocks, instances, ports or pins. |
| -ignore_clock_latency | Ignore clock latency at the source and target registers. |
| -reset_path | Remove any matching set_false_path, set_multicycle_path, set_max_delay, set_min_delay exceptions first. |
| *delay* | The maximum delay. |

The `set_max_delay` command constrains the maximum delay through combinational logic paths. See `set_false_path` for a description of allowed *from_list*, *through_list* and *to_list* objects. If the *to_list* ends at a timing check the setup/hold time is included in the path delay.

When the `-ignore_clock_latency` option is used clock latency at the source and destination of the path delay is ignored. The constraint is reported in the default path group (`**default**`) rather than the clock path group when the path ends at a timing check.

| | |
|---|---|
| **set_max_dynamic_power** | *power* [*unit*] |

The `set_max_dynamic_power` command is ignored.

| | |
|---|---|
| **set_max_fanout** | *fanout* |
| | *objects* |

*fanout*

*objects*                        List of ports or cells.

The `set_max_fanout` command is ignored during timing but is included in SDC files that are written.

| | |
|---|---|
| **set_max_leakage_power** | *power* [*unit*] |

The `set_max_leakage_power` command is ignored.

| | |
|---|---|
| **set_max_time_borrow** | *delay* |
| | *objects* |

*delay*                          The maximum time the latches can borrow.

*objects*                        List of clocks, instances or pins.

The `set_max_time_borrow` command specifies the maximum amount of time that latches can borrow. Time borrowing is the time that a data input to a transparent latch arrives after the latch opens.

| | |
|---|---|
| **set_max_transition** | [-data_path] |
| | [-clock_path] |
| | [-rise] |
| | [-fall] |
| | *transition* |
| | *objects* |

-data_path                       Set the  max slew for data paths.

-clock_path                      Set the  max slew for clock paths.

```
-rise                    Set the  max slew for rising paths.

-fall                    Set the  max slew for falling paths.

transition               The maximum slew/transition time.

objects                  List of clocks, ports or designs.
```

The set_max_transition command is specifies the maximum transition time (slew) design rule checked by the `report_check_types –max_transition` command.

If specified for a design, the default maximum transition is set for the design.

If specified for a clock, the maximum transition is applied to all pins in the clock domain. The `–clock_path` option restricts the maximum transition to clocks in clock paths. The `-data_path` option restricts the maximum transition to clocks data paths. The `–clock_path`, `-data_path`, `-rise` and `–fall` options only apply to clock objects.

| **set_min_capacitance** | *capacitance*<br>*objects* |
|---|---|
| *capacitance* | Minimum capacitance. |
| *objects* | List of ports or cells. |

The `set_min_capacitance` command is ignored during timing but is included in SDC files that are written.

| **set_min_delay** | `[-rise]`<br>`[-fall]`<br>`[-from `*from_list*`]`<br>`[-rise_from `*from_list*`]`<br>`[-fall_from `*from_list*`]`<br>`[-through `*through_list*`]`<br>`[-rise_through `*through_list*`]`<br>`[-fall_through `*through_list*`]`<br>`[-to `*to_list*`]`<br>`[-rise_to `*to_list*`]`<br>`[-fall_to `*to_list*`]`<br>`[-ignore_clock_latency]`<br>`[-reset_path]`<br>*delay* |
|---|---|
| -rise | Set min delay for rising paths. |
| -fall | Set min delay for  falling paths. |
| -from *from_list* | A list of clocks, instances, ports or pins. |

```
-through through_list    A list of instances, pins or nets.

-to to_list              A list of clocks, instances, ports or pins.

-ignore_clock_latency    Ignore clock latency at the source and target registers.

-reset_path              Remove any matching set_false_path, set_multicycle_path,
                         set_max_delay, set_min_delay exceptions first.

delay                    The minimum delay.
```

The `set_min_delay` command constrains the minimum delay through combinational logic. See `set_false_path` for a description of allowed *from_list*, *through_list* and *to_list* objects. If the *to_list* ends at a timing check the setup/hold time is included in the path delay.

When the `-ignore_clock_latency` option is used clock latency at the source and destination of the path delay is ignored. The constraint is reported in the default path group (`**default**`) rather than the clock path group when the path ends at a timing check.

```
set_min_pulse_width     [-high]
                        [-low]
                        min_width
                        objects

-high                   Set the minimum high pulse width.

-low                    Set the minimum low pulse width.

min_width

objects                 List of pins, instances or clocks.
```

If `-low` and `-high` are not specified the minimum width applies to both high and low pulses.

```
set_multicycle_path      [-setup]
                         [-hold]
                         [-rise]
                         [-fall]
                         [-start]
                         [-end]
                         [-from from_list]
                         [-rise_from from_list]
                         [-fall_from from_list]
                         [-through through_list]
                         [-rise_through through_list]
                         [-fall_through through_list]
                         [-to to_list]
                         [-rise_to to_list]
                         [-fall_to to_list]
                         [-reset_path]
                         path_multiplier
```

-setup                  Set cycle count for setup checks.

-hold                   Set cycle count for hold checks.

-rise                   Set cycle count for rising path edges.

-fall                   Set cycle count for falling path edges.

-start                  Multiply the source clock period by *period_multiplier*.

-end                    Multiply the target clock period by *period_multiplier*.

-from *from_list*       A list of clocks, instances, ports or pins.

-through *through_list* A list of instances, pins or nets.

-to *to_list*           A list of clocks, instances, ports or pins.

-reset_path             Remove any matching set_false_path, set_multicycle_path,
                        set_max_delay, set_min_delay exceptions first.

*path_multiplier*       The number of clock periods to add to the path required
                        time.

Normally the path between two registers or latches is assumed to take one clock cycle. The
set_multicycle_path command overrides this assumption and allows multiple clock cycles for a timing
check. See set_false_path for a description of allowed *from_list*, *through_list* and *to_list*
objects.

| | |
|---|---|
| **set_operating_conditions** | [-analysis_type single\|bc_wc\|on_chip_variation]<br>[-library *lib*]<br>[*condition*]<br>[-min *min_condition*]<br>[-max *max_condition*]<br>[-min_library *min_lib*]<br>[-max_library *max_lib*] |
| -analysis_type single | Use one operating condition for min and max paths. |
| -analysis_type bc_wc | Best case, worst case analysis. Setup checks use *max_condition* for clock and data paths. Hold checks use the *min_condition* for clock and data paths. |
| -analysis_type on_chip_variation | The min and max operating conditions represent variations on the chip that can occur simultaineous. Setup checks use *max_condition* for data paths and *min_condition* for clock paths. Hold checks use *min_condition* for data paths and *max_condition* for clock paths. This is the default analysis type. |
| -library *lib* | The name of the library that contains *condition*. |
| *condition* | The operating condition for analysis type single. |
| -min *min_condition* | The operating condition to use for min paths and hold checks. |
| -max *max_condition* | The operating condition to use for max paths and setup checks. |
| -min_library *min_lib* | The name of the library that contains *min_condition*. |
| -max_library *max_lib* | The name of the library that contains *max_condition*. |

The set_operating_conditions command is used to specify the type of analysis performed and the operating conditions used to derate library data.

| | |
|---|---|
| **set_output_delay** | [-rise]<br>[-fall]<br>[-max]<br>[-min]<br>[-clock *clock*]<br>[-clock_fall]<br>[-reference_pin *ref_pin*]<br>[-source_latency_included]<br>[-network_latency_included]<br>[-add_delay]<br>*delay*<br>*port_pin_list* |
| -rise | Set the output delay for the rising edge of the input. |

| | |
|---|---|
| -fall | Set the output delay for the falling edge of the input. |
| -max | Set the maximum output delay. |
| -min | Set the minimum output delay. |
| -clock *clock* | The external check is to *clock*. The default clock edge is rising. |
| -clock_fall | The external check is to the falling edge of *clock*. |
| -reference_pin *ref_pin* | The external check is clocked by the clock that arrives at *ref_pin*. |
| -add_delay | Add this output delay to any existing output delays. |
| *delay* | The external delay to the check clocked by *clock*. |
| *pin_port_list* | A list of pins or ports. |

The `set_output_delay` command is used to specify the external delay to a setup/hold check on an output port or internal pin that is clocked by *clock*. Unless the -add_delay option is specified any existing output delays are replaced.

The –reference_pin option is used to specify a timing check with respect to the arrival on a pin in the clock network. For propagated clocks, the timing check is relative to the clock arrival time at the reference pin (the clock source latency and network latency from the clock source to the reference pin). For ideal clocks, the timing check is relative to the reference pin clock source latency. With the -clock_fall flag the timing check is relative to the falling edge of the reference pin. If no clocks arrive at the reference pin the set_output_delay command is ignored. If no -clock is specified the timing check is with respect to all clocks that arrive at the reference pin. The -source_latency_included and -network_latency_included options cannot be used with -reference_pin.

| | |
|---|---|
| **set_port_fanout_number** | [-min]<br>[-max]<br>*fanout*<br>*ports* |
| -min | Set the min fanout. |
| -max | Set the max fanout. |
| *fanout* | The external fanout of the ports. |
| *port_list* | A list of ports. |

Set the external fanout for *ports*.

| | |
|---|---|
| **set_power_activity** | [-global]<br>[-input]<br>[-input_ports *ports*]<br>[-pins *pins*]<br>[-activity *activity* \| -density density]<br>[-duty *duty*]<br>[-clock *clock*] |
| -global | Set the activity/duty for all non-clock pins. |
| -input | Set the default input port activity/duty. |
| -input_ports *input_ports* | Set the input port activity/duty. |
| -pins *pins* | Set the pin activity/duty. |
| -activity *activity* | The activity, or number of transitions per clock cycle. If *clock* is not specified the clock with the minimum period is used. If no clocks are defined an error is reported. |
| -density *density* | Transitioins per library time unit. |
| -d*uty* *duty* | The duty, or probability the signal is high (0 <= duty <= 1.0). Defaults to 0.5. |
| -clock *clock* | The clock to use for the period with -activity. This option is ignored if -density is used. |

The set_power_activity command is used to set the activity and duty used for power analysis globally or for input ports or pins in the design.

The default input activity for inputs is 0.1 transitions per minimum clock period if a clock is defined or 0.0 if there are no clocks defined. The default input duty is 0.5. This is equivalent to the following command:

    set_power_activity -input -activity 0.1 -duty 0.5

| | |
|---|---|
| **set_propagated_clock** | *objects* |
| *objects* | A list of clocks, ports or pins. |

The set_propagated_clock command changes a clock tree from an ideal network that has no delay one that uses calculated or back-annotated gate and interconnect delays. When *objects* is a port or pin, clock delays downstream of the object are used.

| set_pvt | [-min]<br>[-max]<br>[-process *process*]<br>[-voltage *voltage*]<br>[-temperature *temperature*]<br>*instances* |
|---------|-----------------------------------------------------------------------------------------------------------|
| -min | Set the PVT values for max delays. |
| -max | Set the PVT values for min delays. |
| -process *process* | A process value (float). |
| -voltage *voltage* | A voltage value (float). |
| -temperature *temperature* | A temperature value (float). |
| *instances* | A list instances. |

The set_pvt command sets the process, voltage and temperature values used during delay calculation for a specific instance in the design.

| set_sense | [-type clock\|data]<br>[-positive]<br>[-negative]<br>[-pulse *pulse_type*]<br>[-stop_propagation]<br>[-clock *clocks*]<br>*pins* |
|-----------|-------------------------------------------------------------------------------------------------------------------------------|
| -type clock | Set the sense for clock paths. |
| -type data | Set the sense for data paths (not supported). |
| -positive | The clock sense is positive unate. |
| -negative | The clock sense is negative unate. |
| -pulse *pulse_type* | rise_triggered_high_pulse<br>rise_triggered_low_pulse<br>fall_triggered_high_pulse<br>fall_triggered_low_pulse<br>Not supported. |

| | |
|---|---|
| -stop_propagation | Stop propagating *clocks* at *pins*. |

| | |
|---|---|
| *clocks* | A list of clocks to apply the sense. |
| *pins* | A list of pins. |

The `set_sense` command is used to modify the propagation of a clock signal. The clock sense is set with the -positive and –negative flags. Use the –stop_propagation flag to stop the clock from propagating beyond a pin. The –positive, -negative, -stop_propagation, and –pulse options are mutually exclusive. If the –clock option is not used the command applies to all clocks that traverse *pins*. The –pulse option is currently not supported.

| | |
|---|---|
| **set_timing_derate** | [-rise]<br>[-fall]<br>[-early]<br>[-late]<br>[-clock]<br>[-data]<br>[-net_delay]<br>[-cell_delay]<br>[-cell_check]<br>*derate*<br>[*objects*] |

| | |
|---|---|
| -rise | Set the derating for rising delays. |
| -fall | Set the derating for falling delays. |
| -early | Derate early (min) paths. |
| -late | Derate late (max) paths. |
| -clock | Derate paths in the clock network. |
| -data | Derate data paths. |
| -net_delay | Derate net (interconnect) delays. |
| -cell_delay | Derate cell delays. |
| -cell_check | Derate cell timing check margins. |
| *derate* | The derating factor to apply to delays. |
| *objects* | A list of instances, library cells, or nets. |

The `set_timing_derate` command is used to derate delay calculation results used by the STA. If the –early and –late flags are omitted the both min and max paths are derated. If the –clock and –data flags are not used the derating both clock and data paths are derated.

Use the `unset_timing_derate` command to remove all derating factors.

| | |
|---|---|
| **set_resistance** | `[-max]`<br>`[-min]`<br>*resistance*<br>*nets* |

| | |
|---|---|
| -min | The resistance for minimum path delay calculation. |
| -max | The resistance for maximum path delay calculation. |
| *resistance* | The net resistance. |
| *nets* | A list of nets. |

| | |
|---|---|
| **set_units** | `[-capacitance `*cap_unit*`]`<br>`[-resistance `*res_unit*`]`<br>`[-time `*time_unit*`]`<br>`[-voltage `*voltage_unit*`]`<br>`[-current `*current_unit*`]`<br>`[-power `*power_unit*`]`<br>`[-distance `*distance_unit*`]` |

| | |
|---|---|
| *-capacitance cap_unit* | The capacitance scale factor followed by 'f'. |
| *-resistance res_unit* | The resistance scale factor followed by 'ohm'. |
| *-time time_unit* | The time scale factor followed by 's'. |
| *-voltage voltage_unit* | The voltage scale factor followed by 'v'. |
| *-current current_unit* | The current scale factor followed by 'A'. |
| *-power power_unit* | The power scale factor followed by 'w'. |

The `set_units` command is used to **check** the units used by the STA command interpreter when parsing commands and reporting results. If the current units differ from the `set_unit` value a warning is printed. Use the `set_cmd_units` command to change the command units.

Units are specified as a scale factor followed by a unit name. The scale factors are as follows.

```
M 1E+6
k 1E+3
m 1E-3
u 1E-6
n 1E-9
```

```
p 1E-12
f 1E-15
```

An example of the `set_units` command is shown below.

```
set_units -time ns -capacitance pF -current mA -voltage V -resistance kOhm
```

| **set_wire_load_min_block_si ze** | *size* |
|---|---|

The `set_wire_load_min_block_size` command is not supported.

| **set_wire_load_mode** | top\|enclosed\|segmented |
|---|---|
| top | |
| enclosed | |
| segmented | |

The `set_wire_load_mode` command is ignored during timing but is included in SDC files that are written.

| **set_wire_load_model** | -name *model_name*<br>[-library *library*]<br>[-max]<br>[-min]<br>[*objects*] |
|---|---|
| -name *model_name* | The name of a wire load model. |
| -library *library* | Library to look for *model_name*. |
| -max | The wire load model is for maximum path delays. |
| -min | The wire load model is for minimum path delays. |
| *objects* | Not supported. |

| **set_wire_load_selection_gro up** | [-library *library*]<br>[-max]<br>[-min]<br>*group_name*<br>[*objects*] |
|---|---|

| | |
|---|---|
| *library* | Library to look for *group_name*. |
| -max | The wire load selection is for maximum path delays. |
| -min | The wire load selection is for minimum path delays. |
| *group_name* | A wire load selection group name. |
| *objects* | Not supported. |

The set_wire_load_selection_group command is parsed but not supported.

| **suppress_msg** | *msg_ids* |
|---|---|
| *msg_ids* | A list of error/warning message IDs to suppress. |

The suppress_msg command suppresses specified error/warning messages by ID. The list of message IDs can be found in *doc/messages.txt.*

| **unset_case_analysis** | *port_or_pin_list* |
|---|---|
| *port_or_pin_list* | A list of ports or pins. |

The unset_case_analysis command removes the constant values defined by the set_case_analysis command.

| **unset_clock_latency** | [-source]<br>*objects* |
|---|---|
| -source | Specifies source clock latency (clock insertion delay). |
| *objects* | A list of clocks, pins or ports. |

The unset_clock_latency command removes the clock latency set with the set_clock_latency command.

| **unset_clock_transition** | *clocks* |
|---|---|
| *clocks* | A list of clocks. |

The unset_clock_transition command removes the clock transition set with the set_clock_transition command.

| unset_clock_uncertaint y | [-from\|-rise_from\|-fall_from *from_clock*]<br>[-to\|-rise_to\|-fall_to *to_clock*]<br>[-rise]<br>[-fall]<br>[-setup]<br>[-hold]<br>[*objects*] |
|---|---|

-from *from_clock*

-to *to_clock*

-rise                    The uncertainty is for the rising edge of the clock.

-fall                    The uncertainty is for the falling edge of the clock.

-setup                   *uncertainty* is the setup check uncertainty.

-hold                    *uncertainty* is the hold uncertainty.

*uncertainty*            Clock uncertainty.

*objects*                A list of clocks, ports or pins.

The unset_clock_uncertainty command removes clock uncertainty defined with the set_clock_uncertainty command.

| unset_data_check | [-from\|-rise_from\|-fall_from *from_object*]<br>[-to\|-rise_to\|-fall_to *to_object*]<br>[-setup]<br>[-hold]<br>[-clock *clock*] |
|---|---|

-from *from_object*      A pin used as the timing check reference.

*-to to_object*          A pin that the setup/hold check is applied to.

-setup                   Add a setup timing check.

-hold                    Add a hold timing check.

*clock*                  The setup/hold check clock.

The unset_clock_transition command removes a setup or hold check defined by the set_data_check command.

| unset_disable_inferred_clock_gatin g | *objects* |
|---|---|

|  | | |
|---|---|---|
| *objects* | | A list of clock gating instances, clock gating pins, or clock enable pins. |

The unset_disable_inferred_clock_gating command removes a previous set_disable_inferred_clock_gating command.

| **unset_disable_timing** | [-from *from_port*] |
|---|---|
| | [-to *to_port*] |
| | *objects* |

*from_port*

*to_port*

| *objects* | A list of instances, ports, pins, cells or [library/]cell/port. |
|---|---|

The unset_disable_timing command is used to remove the effect of previous set_disable_timing commands.

| **unset_input_delay** | [-rise] |
|---|---|
| | [-fall] |
| | [-max] |
| | [-min] |
| | [-clock *clock*] |
| | [-clock_fall] |
| | *port_pin_list* |

| -rise | Unset the arrival time for the rising edge of the input. |
|---|---|
| -fall | Unset the arrival time for the falling edge of the input. |
| -max | Unset the minimum arrival time. |
| -min | Unset the maximum arrival time. |
| *clock* | Unset the arrival time from *clock*. |
| -clock_fall | Unset the arrival time from the falling edge of *clock* |
| *pin_port_list* | A list of pins or ports. |

The unset_input_delay command removes a previously defined set_input_delay.

| **unset_output_delay** | [-rise]<br>[-fall]<br>[-max]<br>[-min]<br>[-clock *clock*]<br>[-clock_fall]<br>*port_pin_list* |
|---|---|
| -rise | This is the arrival time for the rising edge of the input. |
| -fall | This is the arrival time for the falling edge of the input. |
| -max | This is the minimum arrival time. |
| -min | This is the maximum arrival time. |
| *clock* | The arrival time is from this clock. |
| -clock_fall | The arrival time is from the falling edge of *clock* |
| *pin_port_list* | A list of pins or ports. |

The unset_output_delay command a previously defined set_output_delay.

| **unset_path_exceptions** | [-setup]<br>[-hold]<br>[-rise]<br>[-fall]<br>[-from\|-rise_from\|-fall_from *from*]<br>[-through\|-rise_through\|-fall_through *through*]<br>[-to\|-rise_to\|-fall_to *to*] |
|---|---|
| -setup | Unset path exceptions for setup checks. |
| -hold | Unset path exceptions for hold checks. |
| -rise | Unset path exceptions for rising path edges. |
| -fall | Unset path exceptions for falling path edges. |
| -from *from* | A list of clocks, instances, ports or pins. |
| -through *through* | A list of instances, pins or nets. |
| -to *to* | A list of clocks, instances, ports or pins. |

The unset_path_exceptions command removes any matching set_false_path, set_multicycle_path, set_max_delay, and set_min_delay exceptions.

| | |
|---|---|
| **unset_propagated_clock** | *objects* |
| *objects* | A list of clocks, ports or pins. |

Remove a previous set_propagated_clock command.

| |
|---|
| **unset_timing_derate** |

Remove all derating factors set with the set_timing_derate command.

| | |
|---|---|
| **unsuppress_msg** | *msg_ids* |
| *msg_ids* | A list of error/warning message IDs to unsuppress. |

The unsuppress_msg command removes suppressions for the specified error/warning messages by ID. The list of message IDs can be found in *doc/messages.txt.*

| |
|---|
| **user_run_time** |

Returns the total user cpu run time in seconds as a float.

| | |
|---|---|
| **with_output_to_variable** | *var { commands }* |
| *var* | The name of a variable to save the output of *commands* to. |
| *commands* | TCL commands that the output will be redirected from. |

The with_output_to_variable command redirects the output of TCL commands to a variable.

| | |
|---|---|
| **write_path_spice** | -path_args *path_args* <br> -spice_directory *spice_directory* <br> -lib_subckt_file *lib_subckts_file* <br> -model_file *model_file* <br> -power *power* <br> -ground *ground* <br> [-simulator hspice\|ngspice\|xyce] |
| *path_args* | -from\|-through\|-to arguments as in report_checks. |

| | |
|---|---|
| *spice_directory* | Directory for spice to write output files. |
| *lib_subckts_file* | Cell transistor level subckts. |
| *model_file* | Transistor model definitions .included by *spice_file*. |
| *power* | Voltage supply name in voltage_map of the default liberty library. |
| *ground* | Ground supply name in *voltage_map* of the default liberty library. |
| -simulator | Simulator that will read the spice netlist. |

The `write_path_spice` command writes a spice netlist for timing paths. Use *path_args* to specify `-from`/`-through`/`-to` as arguments to the `find_timing_paths` command. For each path, a spice netlist and the subckts referenced by the path are written in *spice_directory*. The spice netlist is written in path_<id>.sp and subckt file is `path_<id>.subckt`.

The spice netlists used by the path are written to *subckt_file*, which spice_file `.includes`. The device models used by the spice subckt netlists in *model_file* are also `.included` in spice_file. Power and ground names are specified with the `-power` and `-ground` arguments. The spice netlist includes a piecewise linear voltage source at the input and `.measure` statement for each gate delay and pin slew.

Example command:

```
write_path_spice -path_args {-from "in0" -to "out1" -unconstrained} \
  -spice_directory $result_dir \
  -lib_subckt_file "write_spice1.subckt" \
  -model_file "write_spice1.models" \
  -power VDD -ground VSS
```

When the simulator is hspice, `.measure` statements will be added to the spice netlist.

When the simulator is Xyce, the `.print` statement selects the CSV format and writes the waveform data to a file name `path_<id>.csv` so the results can be used by gnuplot.

| **write_sdc** | [-digits *digits*]<br>[-gzip]<br>[-no_timestamp]<br>*filename* |
|---|---|
| *digits* | The number of digits after the decimal point to report. The default is 4. |
| -gzip | Compress the SDC with gzip. |
| *-no_timestamp* | Do not include a time and date in the SDC file. |
| *filename* | The name of the file to write the constraints to. |

Write the constraints for the design in SDC format to *filename*.

| | |
|---|---|
| **write_sdf** | [-corner *corner*]<br>[-divider /\|.]<br>[-include_typ]<br>[-digits *digits*]<br>[-gzip]<br>[-no_timestamp]<br>[-no_version]<br>*filename* |
| *corner* | Write delays for *corner*. |
| -divider | Divider to use between hierarchy levels in pin and instance names. |
| -include_typ | Include a 'typ' value in the SDF triple that is the average of min and max delays to satisfy some Verilog simulators that require three values in the delay triples. |
| -digits *digits* | The number of digits after the decimal point to report. The default is 4. |
| -gzip | Compress the SDF using gzip. |
| -no_timestamp | Do not write a DATE statement. |
| -no_version | Do not write a VERSION statement. |
| *filename* | The SDF filename to write. |

Write the delay calculation delays for the design in SDF format to *filename*. If -corner is not specified the min/max delays are across all corners. With -corner the min/max delays for *corner* are written. The SDF TIMESCALE is same as the time_unit in the first liberty file read.

| | |
|---|---|
| **write_timing_model** | [-library_name *lib_name*]<br>[-cell_name *cell_name*]<br>[-corner *corner*]<br>*filename* |
| -library_name *lib_name* | The name to use for the liberty library. Defaults to *cell_name*. |
| -cell_name *cell_name* | The name to use for the liberty cell. Defaults to the top level module name. |
| -corner *corner* | The process corner to use for extracting the model. |
| *filename* | Filename for the liberty timing model. |

The `write_timing_model` command constructs a liberty timing model for the current design and writes it to *filename*. `cell_name` defaults to the cell name of the top level block in the design.

The SDC used to extract the block should include the clock definitions. If the block contains a clock network `set_propagated_clock` should be used so the clock delays are included in the timing model. The following SDC commands are ignored when bulding the timing model.

```
set_input_delay
set_output_delay
set_load
set_timing_derate
```

Using `set_input_transition` with the slew from the block context will be used will improve the match between the timing model and the block netlist.  Paths defined on clocks that are defined on internal pins are ignored because the model has no way to include the clock definition.

The resulting timing model can be used in a hierarchical timing flow as a replacement for the block to speed up timing analysis. This hierarchical timing methodology does not handle timing exceptions that originate or terminate inside the block. The timing model includes:

```
combinational paths between inputs and outputs
setup and hold timing constraints on inputs
clock to output timing paths
```

Resistance of long wires on inputs and outputs of the block cannot be modeled in Liberty. To reduce inaccuracies from wire resistance in technologies with resistive wires place buffers on inputs and ouputs.

The extracted timing model setup/hold checks are scalar (no input slew dependence). Delay timing arcs are load dependent but do not include input slew dependency.

| **write_verilog** | [-sort]<br>[-include_pwr_gnd]<br>[-remove_cells *lib_cells*]<br>*filename* |
|---|---|
| -sort | Sort the instances in the netlist. |
| -include_pwr_gnd | Inclue power and ground pins on instances. |
| -remove_cells *lib_cells* | Liberty cells to remove from the verilog netlist. Use get_lib_cells, a list of cells names, or a cell name with wildcards. |
| *filename* | Filename for the liberty library. |

The `write_verilog` command writes a verilog netlist to *filename*. Use -sort to sort the instances so the results are reproducible across operating systems. Use -remove_cells to remove instances of *lib_cells* from the netlist.

## Filter Expressions

The `get_cells`, `get_pins`, `get_ports` and `get_timing_edges` functions support filtering the returned objects by property values. Supported filter expressions are shown below.

| | |
|---|---|
| *property* | Return objects with *property* value equal to *1*. |
| *property==value* | Return objects with *property* value equal to *value*. |
| *property=~pattern* | Return objects with *property* value that matches *pattern*. |
| *property*!=value | Return objects with *property* value not equal to *value*. |
| *property*!~value | Return objects with *property* value that does not match *pattern*. |
| *expr1&&expr2* | Return objects with *expr1* and *expr2*. *expr1* and *expr2* are one of the first three property value forms shown above. |
| *expr1\|\|expr2* | Return objects with *expr1* or *expr2*. *expr1* and *expr2* are one of the first three property value forms shown above. |

where *property* is an property supported by the `get_property` command. Note that if there are spaces in the expression it must be enclosed in quotes so that it is a single argument.

## Variables

| | |
|---|---|
| **hierarchy_separator** | Any character. |

The `hierarchy_separator` separates instance names in a hierarchical instance, net, or pin name. The default value is `'/'`.

| | |
|---|---|
| **sta_bidirect_net_paths_enabled** | 0\|1 |

When set to `0`, paths from bidirectional (inout) ports back through nets are disabled. When set to `1`, paths from bidirectional paths from the net back into the instance are enabled. The default value is `0`.

| | |
|---|---|
| **sta_continue_on_error** | 0\|1 |

The `include` and `read_sdc` commands stop and report any errors encountered while reading a file unless `sta_continue_on_error` is `1`. The default value is `0`.

| | |
|---|---|
| **sta_crpr_mode** | same_pin\|same_transition |

When the data and clock paths of a timing check overlap (see `sta_crpr_enabled`), pessimism is removed independent of whether of the path rise/fall transitions. When `sta_crpr_mode` is `same_transition`, the pessimism is only removed if the path rise/fall transitions are the same. The default value is `same_pin`.

| | |
|---|---|
| **sta_cond_default_arcs_enabled** | 0\|1 |

When set to 0, default timing arcs with no condition (Liberty timing arcs with no "when" expression) are disabled if there are other conditional timing arcs between the same pins. The default value is 1.

| | |
|---|---|
| **sta_crpr_enabled** | 0\|1 |

During min/max timing analysis for on_chip_variation the data and clock paths may overlap. For a setup check the maximum path delays are used for the data and the minimum path delays are used for the clock. Because the gates cannot simultaneously have minimum and maximum delays the timing check slack is pessimistic. This pessimism is known as Common Reconvergent Pesssimism Removal, or "CRPR". Enabling CRPR slows down the analysis. The default value is 1.

| | |
|---|---|
| **sta_dynamic_loop_breaking** | 0\|1 |

When sta_dynamic_loop_breaking is 0, combinational logic loops are disabled by disabling a timing arc that closes the loop. When sta_dynamic_loop_breaking is 1, all paths around the loop are reported. The default value is 0.

| | |
|---|---|
| **sta_gated_clock_checks_enabled** | 0\|1 |

When sta_gated_clock_checks_enabled is 1, clock gating setup and hold timing checks are checked. The default value is 1.

| | |
|---|---|
| **sta_input_port_default_clock** | 0\|1 |

When sta_input_port_default_clock is 1 a default input arrival is added for input ports that do not have an arrival time specified with the set_input_delay command. The default value is 0.

| | |
|---|---|
| **sta_internal_bidirect_instance_paths_enabled** | 0\|1 |

When set to 0, paths from bidirectional (inout) ports back into the instance are disabled. When set to 1, paths from bidirectional ports back into the instance are enabled. The default value is 0.

| | |
|---|---|
| **sta_pocv_enabled** | 0\|1 |

Enable parametric on chip variation using statistical timing analysis. The default value is 0.

| | |
|---|---|
| **sta_propagate_all_clocks** | 0\|1 |

All clocks defined after `sta_propagate_all_clocks` is set to 1 are propagated. If it is set before any clocks are defined it has the same effect as

```
set_propagated_clock [all_clocks]
```

after all clocks have been defined. The default value is `0`.

| | |
|---|---|
| **sta_propagate_gated_clock_enable** | 0\|1 |

When set to 1, paths of gated clock enables are propagated through the clock gating instances. If the gated clock controls sequential elements setting `sta_propagate_gated_clock_enable` to `0` prevents spurious paths from the clock enable. The default value is `1`.

| | |
|---|---|
| **sta_recovery_removal_checks_enable** <br> **d** | 0\|1 |

When `sta_recovery_removal_checks_enabled` is `0`, recovery and removal timing checks are disabled. The default value is `1`.

| | |
|---|---|
| **sta_report_default_digits** | integer |

The number of digits to print after a decimal point. The default value is `2`.

| | |
|---|---|
| **sta_preset_clear_arcs_enabled** | 0\|1 |

When set to 1, paths through asynchronous preset and clear timing arcs are searched. The default value is `0`.

# Alphabetical Index

Version 2.6.0, Sep 23, 2024
Copyright (c) 2024, Parallax Software, Inc.