Conformal® Constraint Designer Database Access Object and Attribute Reference

Product Version 23.2
October 2023

Cadence Design Systems, Inc., 2655 Seely Avenue, San Jose, CA 95134, USA

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. The information contained herein is the proprietary and confidential information of Cadence or its licensors, and is supplied subject to, and may be used only by Cadence's customer in accordance with, a written agreement between Cadence and its customer. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Cadence is committed to using respectful language in our code and communications. We are also active in the removal and/or replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Contents

1

Related Documents

The following lists the documents related to Conformal Constraint Designer:

Document	Description
Conformal Constraint Designer User Guide	
	Describes how to use theConformal Constraint Designer solution.
Conformal Constraint Designer Command Reference	
	Describes the commands for Conformal Constraint Designer.
Conformal Constraint Designer Database Access Object and Attribute Reference	
	Describes the database access objects and attributes.
Conformal Constraint Designer Rule Check Reference	
	Describes the modeling messages, policy rule checks lint rule checks, the CDC rule checks, and the atomic checks.
Conformal HDL Rule Check Reference	
	Describes the HDL rule checks that apply to all Conformal tools.

2

Using Database Access Objects

This document describes the Conformal Constraint Designer attributes and objects.

An *attribute* is a setting that controls how Conformal Constraint Designer operates on objects.

An *object* is anything Conformal Constraint Designer can manipulate, such as designs, ports, constraints, rules, and so on.

Design data is originally stored in the design hierarchy on the corresponding objects when reading in the libraries, the HDL files, and the constraints. During the synthesis session, the design information hierarchy (including the objects and attributes) is continuously updated.

In this book, attributes are organized according to functional categories. In each functional category, attributes are listed with the object types on which they can be set. Each attribute also specifies whether it is settable through the `set_attribute` command.

Some attributes can apply to several objects, so they will be listed under a section called “Common Attributes”.

2.1 Database Objects in Conformal Constraint Designer

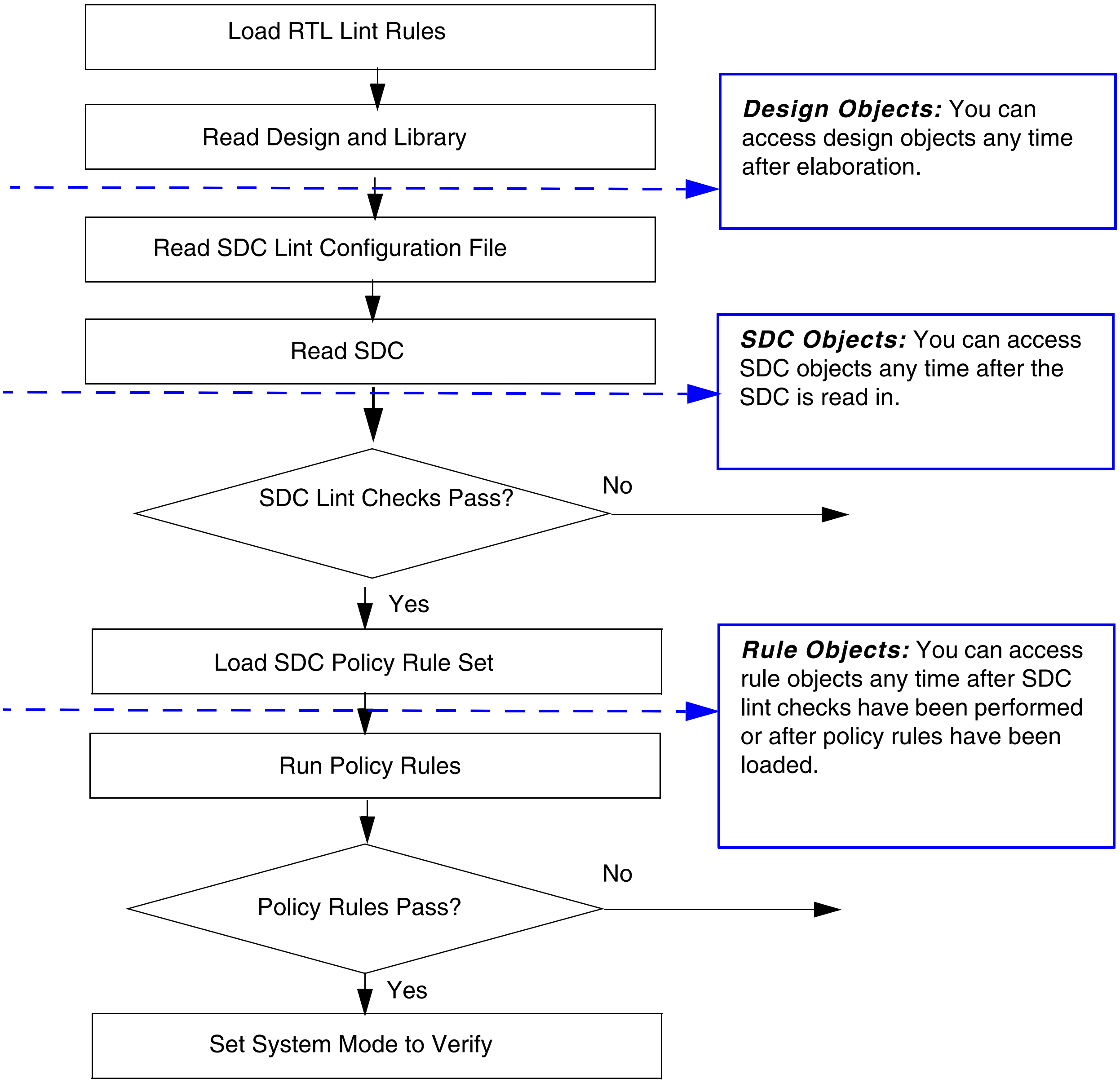
An *object* is anything Conformal Constraint Designer can manipulate, such as designs, ports, constraints, rules, and so on. Each object has a set of attributes. An *attribute* is a setting that controls how Conformal Constraint Designer operates on objects.

In Tcl mode, you have access to the following objects.

Design Related Objects	SDC Related Objects	Rule Related Objects
`design`	`sdcmode`	`ruleset`
`instance`	`sdcdsgn`	`rulegrp`
`port`	`sdcstmt`	`ruleinst`
`pin`	`sdcobj`	`rulesrc`
`net`		`occr`
`lib`		`rulefilter`
`libcell`		
`libpin`		

The following figure illustrates (using the Rule Check Flow) when each type of design object can be accessed.

Figure 2-1 Object Usage Model


Load RTL Lint Rules
Read Design and Library
Design Objects: You can access design objects any time after elaboration.
Read SDC Lint Configuration File
Read SDC
SDC Objects: You can access SDC objects any time after the SDC is read in.
SDC Lint Checks Pass?
No
Yes
Load SDC Policy Rule Set
Rule Objects: You can access rule objects any time after SDC lint checks have been performed or after policy rules have been loaded.
Run Policy Rules
Policy Rules Pass?
No
Yes
Set System Mode to Verify

2.2 Accessing Database Objects

This section describes how you can use various Tcl commands to access database objects and their attributes.

2.2.1 Using the Tcl find Command

You can use the Tcl find command to find a specific object type and to view the value of its attributes:

```
find -<object_type>
     [<patterns> | <object_list> | -of_objects <object_list>]
     [ -sensitive | -nosensitive ]
     [-hierarchical] [-filter <condition>]
```

Where `object_type` is one of the following database object types (such as `sdcdsgn`, `ruleset`, and so on).

For example:

```
    set myinst [ find -ruleinst myruleset_1/grp_a/ri_1 ]
    set all_rule_srcs [ find -rulesrc ]
    set all_rule_insts [ find -ruleinst ]
```

2.2.2 Using the REPORT RULE SOURCE Command

Criteria for the CDC rules is specified through attributes. To view the CDC rule attributes from the tool, use the following command:

```
report rule source "cdc_*" -verbose
```

2.2.3 Using the Tcl get_attribute and set_attribute Commands

Use the `get_attribute` Tcl command to retrieve the value of an attribute, and the `set_attribute` command to set the value of a specific attribute.

```
get_attribute <rule_obj_handle><attr_name>
set_attribute <rule_objc_handle><attr_name><value>
```

For example, the following changes attribute `source_clock` to value `clka` for rule instance s1, and then re-runs the rule instance:

```
#changes value of attribute abc
set R1 [find -ruleinst s1/g1/r1]
set_attribute $R1 source_clock [lindex [find -sdcobj clka ] 0]
run_rule_check s1
```

For example:

```
get_attribute $R1 source_clock //Retrieves attribute source_clock of rule instance
s1/g1/r1
```

3

Conformal Objects

Describes the attributes that belong to Conformal objects. If an attribute is read-write, its value can be set through the set_attribute command. The value of read-only attributes is set by the tool.

3.1 cdc_config_objects

Specifies configuration objects (or configuration registers) that can be ignored by CDC checks.

Read-write Conformal attribute.

Type

Object/list

Value

`<list_of_port_or_register_leaf_instance_objects>`

Example

The following examples use the global attribute `cdc_conf_objects` to specify configuration registers:

(using wild cards)

```
set_attribute [find -conformal ] cdc_config_objects \
  [find -instance {top/config_reg*}]
```

(specifies a list of registers)

```
set_attribute [find -conformal] cdc_config_objects \
  [find -instance [list config1_reg config2_reg \
  config3_reg]]
```

3.2 cdc_convergence_check_failed_sync_chain

This attribute controls the clock domain crossing paths for convergence check. If this attribute is set to false, only clock domain crossing paths with valid control synchronization is considered for the convergence synchronization check. If this attribute is set to true, all clock domain crossing paths considered as control synchronization is considered for convergence synchronization whether there is valid control synchronizer or not. By default this attribute is set to true.

Read-write Conformal attribute.

Type

Boolean

Value

```
< 1 | 0 > or < true | false > or < on | off >
```

3.3 cdc_convergence_depth_limit

Specifies how deep to search for convergence (also known as sequential depth). Depth count starts after the last flip flop in the synchronization chain. For example, a depth of 1 means the tool will search up to the first flip flop after the synchronization chain, a depth of 2 means the tool will search up to the second flop flop after the end of the synchronization chain, and so on.

The higher the limit, the longer it will take to perform convergence checks.

- 0—Specifies no depth limit; all sequential depths are explored.
- N—After the end of the synchronization chain, convergence is checked up to the specified depth limit.

Read-write Conformal attribute.

Type

Integer

Value

```
<N>
```

3.4 cdc_convergence_skip_unreached

Specifies whether convergence checks will consider convergence end points that are not reachable by any primary output or black boxes.

- 1—Skips convergence checks on convergence end points that can not be reached from any primary output or blackbox input pin.

- 0—Performs convergence check on convergence end points that can not be reached from any primary output or blackbox input pin. Default value is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.5 cdc_convergence_stop_at_datapath

Specifies whether to perform convergence checks across CDC datapath crossings in the control cone.

- 1—Perform convergence checks across CDC datapath crossings. With this option, convergence checking takes longer than when this attribute is set to zero.
- 0—Do not perform convergence checks across CDC datapath crossings. Convergence checking is faster when this attribute is set to zero.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.6 cdc_optimize_convergence_violation

Specifies whether to optimize the convergence candidates across sequential boundaries. This can help reduce convergence violations.

Default is 1.

Read-write Conformal attribute.

Type

Boolean

Value

`<1 | 0>` or `<true | false>` or `<on | off>`

3.7 cdc_optimize_convergent_point

Selects the convergence point reported by the convergence check.

- 1 —Finds the point closest to the source where the CDC paths start to converge. With this option, convergence checking takes longer than when this attribute is set to zero.
- 0—Finds the convergent point closest to the convergent instance. Convergence checking is faster when this attribute is set to zero.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.8 cdc_skip_instances

Specifies which hierarchical instances to skip for CDC checks

Read-write Conformal attribute.

Type

Object /list

Value

`<list_of_hierarchical_instance_objects>`

3.9 cdc_sync_chain_guidance_flow

Specifies prioritization during sync chain expansion. By default, sync chains are expanded using the specified min/max values (set through `mux_sync_scheme` and `dff_sync_scheme` attributes). When this attribute is set to1, the sync chain is instead expanded based on the sync chain's logic and fanouts (set through `sync_chain_logic` and `sync_chain_fanout` attributes). Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>`

3.10 cdc_target_instances

Specifies the hierarchical instances for which to perform CDC checks.

Note: If an instance is given both a `cdc_skip_instance` and a `cdc_target_instance` attribute, `cdc_target_instance` takes precedence.

Read-write Conformal attribute.

Type

Object/list

Value

`<list_of_hierarchical_instance_objects>`

3.11 cdc_user_sync_modules

Specifies user sync modules.

Read-write Conformal attribute.

Type

Object/list

Value

`<list_of_user_sync_modules>`

3.12 cfm_root_dir

Specifies the Conformal root directory.

Read-only Conformal attribute.

Type

string

Value

path to `$VERPLEX_HOME`

3.13 command_profile

When set to 1 (or true/on), the tool records the order in which commands are executed and the memory use. The profile includes commands used in the graphical user interface mode.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.14 constrain_clock_source_object

Read-write Conformal attribute.

Default is 1.

Type

Boolean

Value

`<1 | 0> <true | false> <on | off>`

3.15 current_sdc_design

Returns the name of the current SDC design.

In the hierarchical flow, use the `SET SDC DESIGN` command to change the name of the current SDC design. Note that the command `READ HIERARCHICAL SDC` can also change the current SDC design, but at the end of its execution, it leaves the current SDC design set to “`/`”.

Read-only Conformal attribute.

Type

string

3.16 current_sdc_mode

In multi-mode and SDC comparison flows, returns the name of the current SDC mode. Otherwise, the attribute’s value is an empty string.

Use the `SET SDC MODE` command to change the current SDC mode. In the SDC comparison flow, this attribute can be either “golden” or “revised”, based on the last option used to read in SDC files (`-golden` or `-revised`, respectively).

Read-only Conformal attribute.

Type

string

3.17 cut_iopad

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.18 design_search_path

Specifies the design search path.

Read-write Conformal attribute.

Type

string/list

3.19 design_state

Specifies the current design state of the tool.

Read-only Conformal attribute.

Type

string

Value

`<none | read_library | read_design | read_sdc | verify | propagate_clock | commit_clock>`

3.20 echo_result

Controls the printing of command results (1 turns it on; otherwise, 0).

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.21 echo_command

Controls the echoing of command printing (1 turns it on; otherwise, 0).

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.22 enable_recovery_removal_arcs

Read-write Conformal attribute.

Default is 0.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.23 fifo_check_async_mem

When set to 1, the tool checks whether the memory and output registers are asynchronous.

Default value is 1.

Read-only FIFO attribute.

Atomic Check

`fifo_chk_atomic_async_mem`

Type

boolean

Values

< 1 | 0 >

3.24 fifo_check_gray_comb_loop

Set to 1 to check if there is a combinational loop for a gray code register.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_gray_comb_loop`

Type

boolean

Value

< 1 | 0 >

3.25 fifo_check_gray_func

Set to 1 to run functional checks on FIFO read/write gray code registers.

Read-write FIFO attribute.

Default is 0.

Atomic Check

```
fifo_chk_atomic_gray_func
```

Type

boolean

Value

<0 | 1>

3.26 fifo_check_gray_size

When set to 1, the tool runs checks that the gray code size is equal to or greater than the minimum read graycode size.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_gray_size`

Type

`boolean`

Value

<1|0>

3.27 fifo_check_mem_out_exclusive

Set to 1 to check if the memory and output registers are exclusive.

Default is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_mem_out_exclusive`

Type

boolean

Value

< 1 | 0 >

3.28 fifo_check_mem_out_size

Set to 1 to check if the memory size is a multiple of the output size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_out_size
```

Type

boolean

Value

< 1 | 0 >

3.29 fifo_check_mem_size

Set to 1 to checks that the memory size is equal to or larger than the minimum memory size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_size
```

Type

boolean

Value

< 1 | 0 >

3.30 fifo_check_mem_supported_cell_type

Set to 1 to check if all the element types are supported cell types.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_mem_supported_cell_type`

Type

boolean

Value

< 1 | 0 >

3.31 fifo_check_out_size

Set to 1 to check that the output size is equal to or greater than the minimum output size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_out_size`

Type

boolean

Value

< 1 | 0 >

3.32 fifo_check_readptr_size

Set to 1 to checks that the read point size is equal to or larger than the minimum read pointer size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_readptr_size`

Type

boolean

Value

< 1 | 0 >

3.33 fifo_check_readptr_sync

When set to 1, the tool checks if the read pointer is synchronous to the output.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_readptr_sync`

Type

boolean

Value

<1|0>

3.34 fifo_check_single_rgray

When set to 1, the tool checks if there is only one read graycode register candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_rgray`

Type

boolean

Value

<1|0>

3.35 fifo_check_single_rptr

When set to 1, the tool checks if there is only one read pointer candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_readptr`

Type

boolean

Value

<1|0>

3.36 fifo_check_single_synch

When set to 1, the tool checks if there is only one sync candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_single_sync
```

Type

boolean

Value

<1|0>

3.37 fifo_check_single_wgray

When set to 1, the tool checks if there is only one write graycode register candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_single_wgray
```

Type

boolean

Value

<1|0>

3.38 fifo_check_single_wptr

When set to 1, the tool checks if there is only one write pointer candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_single_writeptr
```

Type

boolean

Value

<1|0>

3.39 fifo_check_sync_size

When set to 1, the tool checks if the sync size is equal to the gray code candidate size.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_sync_size
```

Type

boolean

Value

<1|0>

3.40 fifo_check_wdata_size

Set to 1 to checks if the minimum wdata size has been exceeded.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

`fifo_chk_atomic_wdata_size`

Type

boolean

Value

< 1 | 0 >

3.41 fifo_check_writeptr_size

Set to 1 to checks if the write point size is equal to or greater than the minimum write pointer size.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

`fifo_chk_atomic_writeptr_size`

Type

boolean

Value

< 1 | 0 >

3.42 fifo_check_writeptr_sync

Set to 1 to checks if the write pointer is synchronuos to the FIFO memory.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

```
fifo_chk_atomic_writeptr_sync
```

Type

boolean

Value

< 1 | 0 >

3.43 fifo_skip_two_dimensional_check

Set to 1 to skip checking whether the memory is a two-dimensional register array.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_two_dimension
```

Type

boolean

Value

< 1 | 0 >

3.44 gen_critical_threshold

Instructs the tool to consider paths with logic length count greater than the `<percentage>` of the maximum logic length of all paths in the design. The percentage is a real number between 0 and 1. For example, a threshold of .8 corresponds to considering only paths that are longer than 80 percent of the longest path.

Read-write Conformal attribute.

Type

float

Value

`<0-1>`

3.45 gen_print_percentage

Read-write Conformal attribute.

Default is 0.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.46 gen_rename_clock_pin

Read-write Conformal attribute.

Default is "`<instance>/CK`".

Type

string

Value

`<name>`

3.47 gen_rename_data_pin

Specifies the pin name for the data pin of the RTL flip-flop primitive.

Read-write Conformal attribute.

Default is "`<instance>/D`".

Type

string

Value

`<name>`

3.48 gen_rename_reset_pin

Specifies the pin name for the reset port of the flip-flop.

Read-write Conformal attribute.

Default is "`<instance>/R`".

Type

string

Value

`<name>`

3.49 gen_rename_set_pin

Specifies the pin name for the set port of the flip-flop.

Read-write Conformal attribute.

Default is "`<instance>`/S".

Type

string

Value

`<name>`

3.50 gen_sfp_path_limit

Specifies the limit on how much effort to consider for generation. The default limit is 100,000 candidates.

Read-write Conformal attribute.

Default is 100000.

Type

integer

Value

`<number>`

3.51 gen_sfp_warn_on_set_reset

Specifies whether to generate asynchronous set and reset false-paths in a more compact form.

Read-write Conformal attribute.

Default is 1.

Type

Boolean

Value

`<1 | 0>` or `<on | off>` or `<true | false>`

3.52 gui_mode

Controls whether to change to the GUI mode.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.53 include_sdcstmt_ids_in_rule_report

Specifies whether to print the SDC statement ID in the verbose rule report.

Read-write Conformal attribute.

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

Example

For example, the following is a sample verbose rule report when this attribute is set to 0 (default):

```
SDC_LINT_CMD2: CCD unsupported command used
Severity: Error      Occurrence: 7
1: Fail: In line 15, file sdc/test.sdc (set_level_shifter_strategy)
```

With the attribute set to 1:

```
VERIFY> report rule check SDC_LINT_CMD2 -ver
SDC_LINT_CMD2: CCD unsupported command used
Severity: Error      Occurrence: 7
1: Fail: In line 15 (SdcStmt Id: 9), file sdc/test.sdc (set_level_shifter_strategy)
```

3.54 library_search_path

Specifies the library search path.

Read-write Conformal attribute.

Type

string/list

Value

<ccd_l | ccd_xl | ccd_mcc>

3.55 log_file

Specifies the default log file (can be NULL, if there is no log file).

Read-write Conformal attribute.

Type

string

Value

<file_name>

3.56 parallel_keepdir

When set to 1, the results files of the parallel session are saved (in case the run is interrupted and you want to continue from where it stopped previously).

Read-write Conformal attribute.

Type

Boolean

Value

<0 | 1> or <false | true> or <off | on>

3.57 parallel_lsf_kill_command

Specifies the command used to kill jobs in parallel validation runs that use the Load Sharing Facility.

Default is "bkill".

Read-write Conformal attribute.

Type

string

Value

<command_name>

3.58 parallel_lsf_log

This attribute is for parallel validation runs that use LSF.

Specifies either a file name or an email address. When a file name is specified, each client writes a log file in the current directory with name LSF_<job_name>.log, where <job_name> is assigned internally by the software. Or, an email address is specified and the log of each LSF job is sent by email.

Read-write Conformal attribute.

Type

string

Value

`<file_name | email_address>`

Default

" "

3.59 parallel_lsf_max_remote

This attribute is for parallel validation runs that use LSF.

Specifies the maximum number of clients for running parallel jobs in LSF. The default is 8.

Read-write Conformal attribute.

Type

integer

Value

`<number>`

Default

2

3.60 parallel_lsf_submit_command

Specifies the command used to submit jobs to LSF. The default is `bsub`.

Read-write Conformal attribute.

Type

string

Value

`<command_name>`

3.61 parallel_lsf_submit_options

Specifies any other options in the command line for LSF (for example, the queue and priorities). The default is `"  "`.

Read-write Conformal attribute.

Type

string

Value

`<submit_command_options>`

3.62 parallel_lsf_test

When set to 1, the tool checks if the environment is properly set up for LSF. The software will print a message and exit, and will not perform validation.

Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.63 parallel_run_recovery

When set to 1, parallel validation will collect any results from a previously-interrupted session and exit.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.64 rule_source_search_path

Specifies the rule source search path. **Read-write** Conformal attribute.

Type

string/list

3.65 screen_display

Specifies whether the transcript output is displayed on the terminal screen. **Read-write** Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.66 sdc_attr_map_file

Read-write Conformal attribute.

Type

string

Value

<file_name>

3.67 sdc_auto_check_severity

Specifies the lowest current severity of the SDC rules that will be checked when going from Setup to Verify mode. If a rule's severity is lower, it will not be checked until you execute the `run rule check` command.

Read-write Conformal attribute.

Type

string

Value

`<ignore | note | warning | error>`

3.68 sdc_genclk_in_all_clocks

Controls whether the SDC `get_clocks` and `all_clocks` commands report generated clocks.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

Default

1

3.69 sdc_library_genclk_use_group_name

Controls whether a Liberty `generated_clock` group generates a single clock named after the group (when set to 1, true, or on), or one clock for each of its clock pins, named after the respective pins.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

Default

1

3.70 sdc_max_errors

Specifies the maximum number of rule violations with Error severity before parser stops. Where 0 = no limit.

Read-write Conformal attribute

Type

Integer

3.71 sdc_report_statistics

If this attribute is set to 1, then after reading SDC files, Conformal Constraint Designer reports statistics about how many commands passed or failed, and how many were unsupported.

Read-write Conformal attribute.

Type

integer

Default

1

Value

<0 | 1>

3.72 sr_always_consider_cascaded_sync_chain

When this option is set (default), tool finds out the real sync chain used for set/reset synchronization. To find the real sync chain, tool starts from the sync chain registers present in the fanin cone of the driver. At every subsequent step, tool traces from current sync chain register's set/reset port to the new sync chain. The sync chain found at the end is the real sync chain. Sync chain found previously while tracing back are termed as cascade sync chains.

If this option is unset, tool treats sync chain found in the fanin of the set/reset driver as real sync chain.

Read-write Conformal attribute.

Type

Boolean

Default

True

Value

<1| 0> or <false | true> or <off | on>

3.73 sr_display_all_end_points

Specifies whether to display one end point or all end points for the set/reset driver.

Read-write Conformal attribute.

Default is false.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.74 sr_merge_same_driver_occurrence

Specifies whether to merge all occurrences related to same driver. If this attribute is set, tool will report only one occurrence for a set/reset driver.

Read-write Conformal attribute.

Default is false.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.75 sr_print_setreset_source

Specifies whether to print set/reset source names in set/reset check verbose report.

Read-write Conformal attribute.

Default is false.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.76 timing_traverse_logic_without_timing_arc

Specifies when timing traversal will be done over logic gates inside a library cell, depending on whether the library cells has timing arcs.

Read-write Conformal attribute.

Default is 1.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.77 undock_rulemgr_notebook

Undocks the Rule Manager notebook.

By default, the various rule diagnosis windows (attribute table, schematics browser, and source code browser) are contained in a single window called a *notebook*. This notebook is embedded in the Rule Manager by default, unless undocked using the `undoc_rulemgr_notebook` attribute.

Read-write Conformal attribute.

Default is 0.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.78 undock_rulemgr_notepage

Undocks the rule diagnosis windows.

By default, the various rule diagnosis windows (attribute table, schematics browser, and source code browser) are contained in a single window called a *`notebook`*. This notebook is embedded in the Rule Manager by default, unless undocked using the `undoc_rulemgr_notebook` attribute.

When `undock_rulemgr_notepage` is set to 1, all diagnosis windows (attribute table, schematics browser, and source code browser) will be displayed in standalone windows.

Read-write Conformal attribute.

Default is 0.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.79 usage

Returns usage in list form.

Read-only Conformal attribute.

Type

string/list

3.80 val_exc_path_expansion

Read-write Conformal attribute.

Type

string

Value

`<dynamic | static>`

3.81 val_physical_path_limit_sfp

Specifies the maximum number of paths to be expanded and validated per SDC false path (FP) exception statement. If an exception statement reaches that limit, the status for the SDC validation for that statement is “path limit reached.”

Read-write Conformal attribute.

Type

integer

Value

`<0 | num>`

3.82 val_physical_path_limit_smcp

Specifies the maximum number of paths to be expanded and validated per SDC multi-cycle-path (MCP) exception statement. If an exception statement reaches that limit, the status for the SDC validation for that statement is “path limit reached.”

Read-write Conformal attribute.

Type

integer

Value

`<0 | num>`

3.83 val_physical_path_limit_trv

Specifies the maximum number of paths to be expanded and validated per SDC timing report validation (TRV) exception statement. If an exception statement reaches that limit, the status for the SDC validation for that statement is “path limit reached.”

Read-write Conformal attribute.

Type

integer

Value

`<0 | num>`

3.84 val_sfp_expand_cdc

Disables statement-level optimization through CDC path expansion.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.85 val_sfp_expand_disabled_clock

Disables statement-level disabled clock optimization by performing path expansion.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.86 val_sfp_expand_set_reset

Disables statement-level set/reset path optimization by performing path expansion.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.87 val_sfp_expand_tied

Disables statement-level tied path optimization by performing path expansion. Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.88 val_sfp_prove_cdc

Specifies whether to perform functional verification on CDC false paths. Default is 1.

Read-write Conformal attribute.

Type

Boolean

Value

`<1 | 0>` or `<true | false>` or `<on | off>`

3.89 val_sfp_prove_method

Specifies whether to verify SDC false-path exceptions by performing combinational or sequential checks. Default is combinational.

Read-write Conformal attribute.

Type

string

Value

`<combinational | sequential>`

3.90 val_sfp_prove_set_rest

Specifies whether to prove functional verification on false paths that end at set/reset ports of flip-flops. Default is 1.

Read-write Conformal attribute.

Type

Boolean

Value

`<1 | 0>` or `<true | false>` or `<on | off>`

3.91 val_sfp_skip_cdc

Specifies whether to skip functional verification for CDC false paths. Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.92 val_sfp_skip_reset

Specifies whether to skip functional verification for reset false paths. Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.93 val_sfp_skip_set

Specifies whether to skip functional verification for set false paths. Default is 0.

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.94 val_smcp_checks

Specifies the set of multi-cycle path (MCP) atomic properties (or sub checks) to validate for each MCP check. Default is `src_stb`.

Read-write Conformal attribute.

Type

string

Value

`<src_stb | src_avl | dest_stb | src_hold | dest_hold | all>`

3.95 val_smcp_class

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.96 val_smcp_skip_cdc

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.97 val_smcp_skip_reset

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.98 val_smcp_skip_set

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.99 val_store_exc_path

Specifies whether to store and validate all exception paths, or to store the first failed exception path found during validation and stop validating the remaining paths for the exception statement. Storing the first fail exception will require less memory. Storing all exception paths will require more memory if there is a large number of exception paths. Default is `first_fail`.

Read-write Conformal attribute.

Type

string

Value

`<first_fail | all>`

3.100 val_smcp_class_bbox_reg

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.101 val_smcp_class_pi_reg

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.102 val_smcp_class_reg_bbox

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.103 val_smcp_class_reg_po

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.104 val_smcp_class_reg_reg

Read-write Conformal attribute.

Type

Boolean

Value

`<0 | 1>` or `<false | true>` or `<off | on>`

3.105 val_write_sdc_monitor_sfp

Specifies a limit in the number of simulation monitors for the FP exceptions.

Read-write Conformal attribute.

Type

integer

3.106 val_write_sdc_monitor_smcp

Specifies a limit in the number of simulation monitors for the MCP exceptions.

Read-write Conformal attribute.

Type

integer

3.107 val_write_sdc_monitor_trv

Specifies a limit in the number of simulation monitors for the TRV exceptions.

Read-write Conformal attribute.

Type

integer

3.108 version_info

Returns the current `get_version_info` in list form.

Read-only Conformal attribute.

Type

string/list

Value

```
<build_no> <build_date> <32 | 64> <hostname> <platform>
```

3.109 usage_delta

Returns the results of `usage -delta` in list form.

Read-only Conformal attribute.

Type

string/list

3.110 usage_elapse

Returns the results of `usage -elapse` in list form.

Read-only Conformal attribute.

Type

string/list

3.111 license_mode

Returns the current license mode.

Read-only Conformal attribute.

Type

string

4

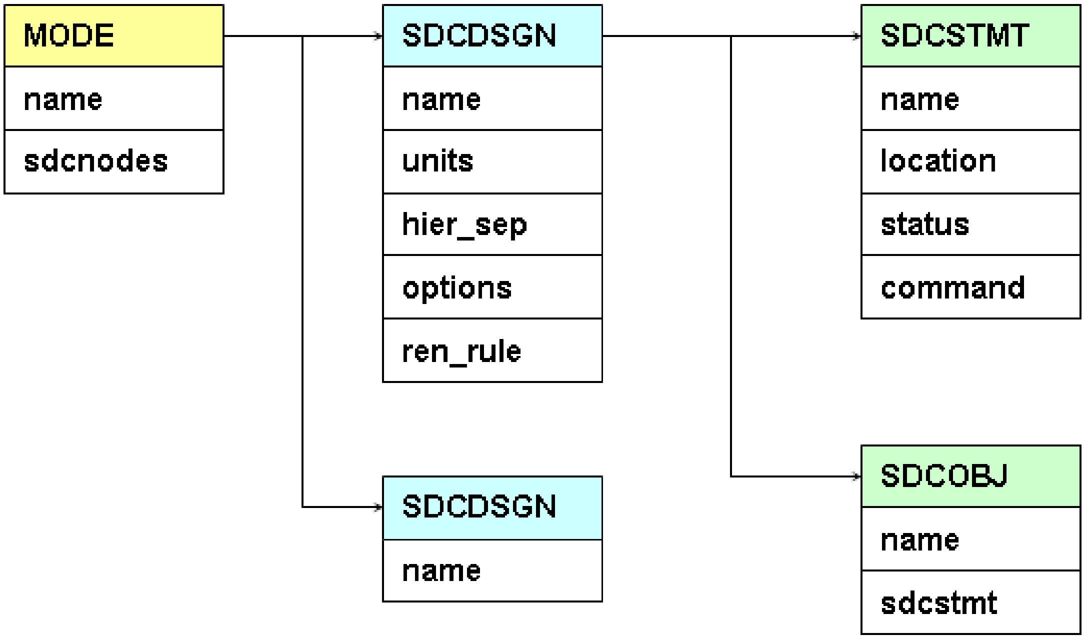
SDC Objects

SDC objects consist of following object types:

- SDC Object Overview on page 76
- Common Attributes for SDC Objects on page 76
- SDCOBJ on page 78
- SDCSTMT on page 107
- SDCMODE on page 111
- SDCDSGN on page 112

4.1 SDC Object Overview

The following figure illustrates how SDC objects relate to each other.



Note: When a rule of severity type "`error`" is reported for an `SDCSTMT`, the statement fails and no `SDCOBJ` is created for it.

4.2 Common Attributes for SDC Objects

The following lists the common attributes for SDCOBJ.

`command_name`

Name of the SDC command

Type: string

Value: `<command_name>`

`design_type`

Specifies the design type

Type: string

Value: `<golden | revised>`

`id`

Identification of SDCOBJ

Type: string

Value: `<sdcobj_id>`

`is_invalid`

SDC objects can be invalidated when they fail checks such as SDC_LINT_REF8, or when they refer to other invalid SDC objects.

Type: boolean

Value: <0 | 1>

`name`

Name of the SDCOBJ.

Type: string

Value: `<command_name>`

`object_type`

Specifies the object type

Type: string

Value: `sdcobj`

`sdcdsgn`

sdcdsgn

Type: object

Value: `<sdcdsgn_object>`

`sdcmode`

Null if in single-mode; otherwise, SDC mode name

Type: object

Value: `<sdcmode_object>`

`sdcstmts`

List of sdcstmts objects this sdcobj made out of

Type: list

Value: `{<list_of_sdcstmts_object> ..}`

4.3 SDCOBJ

`SDCOBJ` represents the final results of processing all `SDCSTMT` objects. All final constraints are accessible through `SDCOBJ`. Multiple `SDCSTMT` objects might result in one `SDCOBJ`, and conversely, one `SDCSTMT` could result in multiple `SDCOBJ`, depending on the type of constraints.

Examples

- The following command returns `SDCOBJ` for final set of SDC objects:

```
set allsdc [find -sdcobj]
```

- The following command returns `SDCOBJ` for all clock objects:

```
set clksdc [find -sdcobj -filter command_name=~create*clock]
```

- The following command returns `SDCOBJ` for all SDC objects in mode `m1`:

```
set m1_sdc [find -sdcobj -filter {sdcmode == m1}]
```

4.3.1 List of SDC Object Types Specific to Conformal Constraint Designer

- ccd_clock_group on page 78
- ccd_critical_path on page 79
- ccd_units on page 79

4.3.1.1 ccd_clock_group

`clocks`	List of clock objects that belong to clock_group Type: object/list Value: `<clock_objects>`
`clock_group_type`	Specifies the clock group type. Type: string Value: `<sync	phys_ex	logic_ex>`

4.3.1.2 ccd_critical_path

`end_clock`	Type: object Value: `<end_clock_object>`
`from`	Type: object/list Value: `<from	rise_from	fall_from>` `<object>`
`is_mrt`	Type: boolean Value: `<0	1>`
`mrt_path_id`	Type: string Value: `<path_name_in_mrt>`
`start_clock`	Type: object Value: `<start_clock_object>`
`through`	Type: object/list Value: `{{<through	rise_though	fall_through> <object>} {<through	rise_though	fall_through> <object>} …}`
`to`	Type: object/list Value: `<to	rise_to	fall_to> <object>`

4.3.1.3 ccd_units

`ref_object`	Type: object Value: `<main_library>` Use LIBRARY for main library.

`units`	Type: list Value: `{time {value <>} capacitance {} resistance {} voltage {} current {} static_power {}}` Time unit in second "s", otherwise NULL Capacitance unit in farad "F", otherwise NULL Resistance unit in ohm "Ohm", otherwise NULL Voltage unit in volt "V", otherwise NULL Current unit in amp "A", otherwise NULL Static power unit in watt "W",

4.3.2 Supported SDC Commands

The following lists the SDC commands supported for SDC objects:

- create_clock on page 82
- create_generated_clock on page 83
- create_voltage_area on page 84
- group_path on page 85
- set_annotated_transition on page 85
- set_case_analysis on page 86
- set_clock_gating_check on page 86
- set_clock_groups on page 86
- set_clock_latency on page 87
- set_clock_sense on page 87
- set_clock_transition on page 88
- set_clock_uncertainty on page 89
- set_data_check on page 89
- set_disable_timing on page 90
- set_dont_touch on page 90
- set_dont_touch_network on page 91
- set_drive on page 91

4.3.2.1 create_clock

`clock_groups`	Type: object/list Value: `sdcobj`
`latency`	Type: list/object Value: {`<sdcobj>` `<sdcobj>`….}
`log_ex_clocks`	Type: object/list Value: `sdcobj`
`period`	Clock period Type: float Value: `<clock_period>`
`phys_ex_clocks`	Type: object/list Value: `sdcobj`
`propagated`	Type: object Value: `sdcobj`
`ref_object`	Object/list of objects where the clock will be created Type: list/object Value: `<port	pin>`

transition

Type: object

Value: <sdcobj>

uncertainty

Type: list/object

Value: {<sdcobj> <sdcobj>….}

waveform

The clock time of the rising and falling edge

Type: list

Value: {<rising_time> <falling_time>}

4.3.2.2 create_generated_clock

divide_by

Value of divided_by; if not provided, value is NULL
Type: integer
Value: <factor>

duty_cycle

Percentage of duty cycle (such as, 60); if not provided, value is NULL
Type: float
Value: <percentage>

edge_shift

Specifies a list of floating numbers that represents the amount of shift
Type: list
Value: {<float> <float> <float> …..}

edges

List of clock edges to generate; NULL if not given
Type: list
Value: {<integer> <integer> <integer>….}

invert

Value is 1, if inverted; otherwise, 0

Type: boolean

Value: true | false

combinational

Type: boolean

Value: <0 |1>

latency

Type: list/object

Value:{<sdcobj> <sdcobj>….}

log_ex_clocks

Type: object/list

Value: <sdcobj>

`master_clock`

Specifies the master clock to be used for the generated clock
Type: object
Value: `<sdcobj>`

`multiply_by`

Value of multiply_by; if not provided, value is NULL
Type: integer
Value: `<factor>`

`p_clock_source`

Clocks that are defined on the pin specified in `-source`, or that propagate to it if no clocks are defined there.

Type: object/list

Value: `sdcobj`

`period`

Clock period

Type: float

Value: `<clock_period>`

`ref_object`

Object/list of objects where the clock will be created

Type: list/object

Value: `{<port | pin> .. }`

`source`

Type: object
Value: `<port | pin>`

`transition`

Type: list/object

Value: {`<sdcobj>` `<sdcobj>`....}

`uncertainty`

Type: list/object
Value: {`<sdcobj>` `<sdcobj>`....}

`waveform`

Clock time of the rising and falling edge
Type: list
Value: {`<rising_time>` `<falling_time>`}

4.3.2.3 create_voltage_area

`coordinate`

Type: list

Value: `{x1 y1 x2 y2}`

`guard_band`

Type: list

Value: `{x <> y <>}`

`ref_object`

Type: list

Value: `<cells_list>`

4.3.2.4 group_path

`critical_range`

Type: float

Value: `<critical_range><sdcstmt>`

Defaut is 0.0 (0.0 refers to the timing paths with the worst violation)

`from`

Type: list

Value: `<from | rise_from | fall_from>` `<object>`

`ref_object`

Type: object

Value: `<current_design>`

Use `DESIGN` for current design

`through`

Type: list

Value: `{{<through | rise_though | fall_through> <object> ... }`

`to`

Type: list

Value: `<to | rise_to | fall_to> <object>`

`weight`

Type: float

Value: `<weight><sdcstmt>`

Default is 1.0

4.3.2.5 set_annotated_transition

`corners`

Type: list

Value: `{max_rise { value <n> delta_only <0 | 1> sdcstmt <sdcstmt>} max_fall {} min_rise {} min_fall {}}`

`ref_object`

Type: object

Value: `<port | pin>`

4.3.2.6 set_case_analysis

`is_ignored`	Type: boolean Value: `<0	1>` 0 if used in propagation, 1 if ignored Settable
`ref_object`	Type: object Value: `<port	pin>`
`value`	Type: integer Value: `<0	1>`

4.3.2.7 set_clock_gating_check

`corners`	Type: list Value: `{setup_rise {value <> level <low	high	both> sdcstmt <>} setup_fall {} hold_rise {} hold_fall {}}` Note: Level is “both” if nothing is specified
`ref_object`	Type: object Value: `<clock	port	pin	cell>`

4.3.2.8 set_clock_groups

`allow_paths`	Type: boolean Value: `<0	1>` Available when type is asynchronous, otherwise 0
`groups`	Literal list of groups Type: list Value: `{{<sdcobj> ...} …}` Use `SDCOBJ` for `create*clock`

`ref_object`

Type: object

Value: `<current_design>`

Use `DESIGN` for current design.

`type`

Clock group types

Type: string

Value: `<physically_exclusive | logically_exclusive | asynchronous>`

4.3.2.9 set_clock_latency

`clock`

Type: object

Value: `<sdcobj>`

This will be null if ref_object is a clock object

`corners`

Type: list

Value: `{source_early_max_rise {value <> sdcstmt <>} source_early_max_fall {} source_early_min_rise {} source_early_min_fall {} source_late_max_rise {} source_late_max_fall {} source_late_min_rise {} source_late_min_fall {} network_max_rise {latency <> sdcstmt <>`

`ref_object`

Type: object

Value: `<clock | port | pin>`

4.3.2.10 set_clock_sense

`clock`

Reference clock objects. By default, all clocks passing through the given reference object will be considered

Type: list/object

Value: `{<sdcobj> ...}`

`is_ignored`

Type: boolean

Value: `<0 | 1>`

0 if used in propagation, 1 if ignored

Settable

`ref_object`

Type: object

Value: `<pin>`

`type`

Clock sense type

Type: string

Value: `{stop_propagation | positive | negative | rise_triggered_high_pulse | rise_triggered_low_pulse | fall_triggered_high_pulse | fall_triggered_low_pulse}`

4.3.2.11 set_clock_transition

`corners`

Clock transition value

Type: list

Value: `{max_rise {value <> sdcstmt <>} max_fall {} min_rise {} min_fall {}}`

`ref_object`

List of clock objects

Type: object

Value: {`<sdcobj>` …}
List of clock objects

4.3.2.12 set_clock_uncertainty

`corners`	(for simple uncertainty) Type: list Value: `{setup {value <> sdcstmt <>} hold {}}`

`corners`	(for inter-clock uncertainty)
	Type: list
	Value: `{setup_rise_from_rise_to {value <> sdcstmt <>} setup_rise_from_fall_to {}`
	`setup_fall_from_rise_to {}`
	`setup_fall_from_fall_to {}`
	`hold_rise_from_rise_to {}`
	`hold_rise_from_fall_to {}`
	`hold_fall_from_rise_to {}`
	`hold_fall_from_fall_to {} }`

`from`	Type: object Value: `<clock	port	pin>`
`to`	Type: object Value: `<clock	port	pin>`

4.3.2.13 set_data_check

`clock`	Type: object Value: `<clock>`

`corners`	Type: list
	Value: `{setup_rise_from_rise_to {value <> sdcstmt <>} setup_rise_from_fall_to {}`
	`setup_fall_from_rise_to {}`
	`setup_fall_from_fall_to {}`
	`hold_rise_from_rise_to {}`
	`hold_rise_from_fall_to {}`
	`hold_fall_from_rise_to {}`
	`hold_fall_from_fall_to {} }`

`from`	Type: object Value: `<port	pin>`

`to`	Type: object Value: `<port | pin>`

4.3.2.14 set_disable_timing

`from`	Full name of the pin or libpin object Type: list/object Value: G
`is_ignored`	Type: boolean Value: `<0 | 1>` 0 if used in propagation, 1 if ignored Settable
`ref_object`	Reference Type: object Value: `<cell | port | pin | libcell | libpin>`
`to`	Full name of the pin or libpin object Type: list/object Value: `<pin | libpin>`

4.3.2.15 set_dont_touch

`ref_object`	Type: object Value: `<design | instance | net | libcell>`
`value`	Type: boolean Value: `<0 | 1>`

4.3.2.16 set_dont_touch_network

`ref_object`	Type: object Value: `<clock	port	pin>`
`value`	Type: boolean Value: `<0	1>`

4.3.2.17 set_drive

`corners`	Type: list Value: `{max_rise {value <> sdcstmt <>} max_fall {} min_rise {} min_fall {} }`
`ref_object`	Input or output port Type: object Value: `<port>`

4.3.2.18 set_driving_cell

`clock`	Type: object Value: `<clock>`
`corners`	Type: list Value: `{clock_rise_max_rise {libcell <> library <> from_pin <> pin <> input_transition_rise <> input_transition_fall <> sdcstmt <>} clock_rise_max_fall {} clock_rise_min_rise {} clock_rise_min_fall {} clock_fall_max_rise {} clock_fall_max_fall {} clock_fall_min_fall`
`dont_scale`	Type: boolean Value: `<0	1>`
`multiply_by`	Type: float Value: `<float>`

`no_design_rule`

Type: boolean

Value: `<0 | 1>`

`ref_object`

Input or output port

Type: object

Value: `<port>`

4.3.2.19 set_false_path

`corners`

Type: list

Value: `{setup_rise {sdcstmt <>} setup_fall {} hold_rise {} hold_fall {}}`

`from`

Type: list

Value: `{<from | rise_from | fall_from> <object> }`

`through`

Type: list

Value: `{<through | rise_through | fall_through> <object>  …. }`

`to`

Type: list

Value: `{<to | rise_to | fall_to> <object> }`

4.3.2.20 set_fanout_load

`ref_object`

Output or inout port

Type: object

Value: `<port>`

`value`

Type: float

Value: `<load>`

4.3.2.21 set_ideal_latency

`corners`

Type: list

Value: `{max_rise {value <n> sdcstmt <sdcstmt>} max_fall {} min_rise {} min_fall {}}`

`ref_object`

Type: object

Value: `<port | pin>`

Hierarchical pins are NOT valid

4.3.2.22 set_ideal_net

`ref_object`

Type: object

Value: `<??>`

`value`

Type: boolean

Value: `<0 | 1>`

4.3.2.23 set_ideal_network

`ref_object`

Type: object

Value: `<port | net | net>`

`propagate`

Type: boolean

Value: `<0 | 1>`

Default is 1.

4.3.2.24 set_ideal_transition

`corners`

Type: list

Value: `{max_rise {value <n> sdcstmt <sdcstmt>} max_fall {} min_rise {} min_fall {}}`

`ref_object`

Type: object

Value: `<port | pin>`

4.3.2.25 set_input_delay

`clock`

Reference clock objects

Type: object

Value: `<sdcobj>`

`corners`

Corners for the input delay

Type: list

Value: `{clock_rise_max_rise { value <n> level_sensitive <0 | 1> network_latency_included <0 | 1> source_latency_included <0 | 1> sdcstmt <sdcstmt>} clock_rise_max_fall {} clock_rise_min_rise {} clock_rise_min_fall {} clock_fall_max_rise {} clock_fall_max_fall {}`

`ref_object`

Type: object

Value: `<port | pin>`

`reference_pin`

`reference_pin` relative to the delay

Type: object

Value: `<port | pin>`

4.3.2.26 set_input_transition

`clock`

Reference clock object

Type: object

Value: `<sdcobj>`

`corners`

All the corners for output delay.

Type: list

Value: `{clock_rise_max_rise { value <n> sdcstmt <sdcstmt>} clock_rise_max_fall {} clock_rise_min_rise {} clock_rise_min_fall {} clock_fall_max_rise {} clock_fall_max_fall {} clock_fall_min_rise {} clock_fall_min_fall {}}`

`ref_object`

Input or output port

Type: object

Value: `<port>`

4.3.2.27 set_level_shifter_strategy

`rule`

Type: list

Value: `<all | low_to_high | high_to>`

Default is `all`.

4.3.2.28 set_level_shifter_threshold

`percent`

Type: float

Value: `<percentage>`

Default is 0

`voltage`

Type: float

Value: `<voltage>`

Default is 0

4.3.2.29 set_load

`corners`

(for ports)

Type: list

Value: `{max_rise { value <n> pin_load <0 | 1> wire_load <0 | 1> sdcstmt <sdcstmt>} max_fall {} min_rise {} min_fall {}}`

`corners`

(for nets)

Type: list

Value: `{max {value <n> subtract_pin_load <0 | 1> sdcstmt <sdcstmt>} min {}}`

`ref_object`

Lists where the object is created (for ports)

Type: object

Value: `<port>`

`ref_object`

Lists where the object is created (for nets)

Type: object

Value: `<net>`

4.3.2.30 set_logic_dc

`ref_object`

Type: object

Value: `<port>`

4.3.2.31 set_logic_one

`ref_object`

Type: object

Value: `<port>`

4.3.2.32 set_logic_zero

`ref_object`	Type: object Value: `<port>`

4.3.2.33 set_max_area

`value`	Type: float Value: `<area>`

4.3.2.34 set_max_dynamic_power

`value`	Type: float Value: `<power>`
`units`	Type: string Value: `<GW | MW | KW | W | mW | uW | nW | pW | fW | aW>`

4.3.2.35 set_max_capacitance

`corners`	For clock objects Type: list Value: `{clock_path_rise {value <> sdcstmt <>} clock_path_fall {} data_path_rise {} data_path_fall {} }`
`ref_object`	Type: object Value: `<current_design | port>`
`value`	Type: float Value: `<capacitance>`

4.3.2.36 set_max_delay

corners	Type: list Value: {rise {value <> sdcstmt <>} fall {}}
from	Type: list Value: {<from | rise_from | fall_from> <object> }
through	Type: list Value: {<through | rise_through | fall_through> <object> …. }
to	Type: list Value: {<to | rise_to | fall_to> <object> }

4.3.2.37 set_max_dynamic_power

value	Type: float Value: <power>
units	Type: string Value: <GW | MW | KW | W | mW | uW | nW | pW | fW | aW>

4.3.2.38 set_max_fanout

ref_object	Type: object Value: <current_design | port> Note: port has to be "input/inout" port
value	Type: float Value: <fanout>

4.3.2.39 set_max_leakage_power

`value`	Type: float Value: `<power>`
`units`	Type: string Value: `<GW	MW	KW	W	mW	uW	nW	pW	fW	aW>`

4.3.2.40 set_max_time_borrow

`ref_objecct`	Type: object Value: `<clock	cell	pin>`
`value`	Type: float

4.3.2.41 set_max_transition

`corners`	Type: list Value: `{clock_path_rise {value <> sdcstmt <>} clock_path_fall {} data_path_rise {} data_path_fall {} }`
`ref_object`	For design objects Type: object Value: `<port	design>`
`ref_object`	For clock objects Type: object Value: `<clock>`
`value`	Type: float Value: `<transition>`

4.3.2.42 set_min_capacitance

`ref_object`	Where the object is created Type: object Value: `<current_design	port>`
`value`	Type: float Value: `<capacitance>`

4.3.2.43 set_min_delay

`corners`	Type: list Value: `{rise {value <> sdcstmt <>} fall {}}`
`from`	Type: list Value: `{<from	rise_from	fall_from> <object> }`
`through`	Type: list Value: `{<through	rise_through	fall_through> <object> …. }`
`to`	Type: list Value: `{<to	rise_to	fall_to> <object> }`

4.3.2.44 set_multicycle_path

`corners`	Type: list Value: `{setup_rise {value <> is_start <0|1> sdcstmt <>} setup_fall {} hold_rise {} hold_fall {}}`
`from`	Type: list Value: `{<from	rise_from	fall_from> <object> }`

`through`

Type: list

Value: `{<through | rise_through | fall_through> <object>  …. }`

`to`

Type: list

Value: `{<to | rise_to | fall_to> <object> }`

4.3.2.45 set_output_delay

`clock`

Type: object

Value: `<sdcobj>`

Reference clock object

`corners`

All the corners for output delay

Type: list

Value: `{clock_rise_max_rise { value <n> level_sensitive <0 | 1> network_latency_included <0 | 1> source_latency_included <0 | 1> sdcstmt <sdcstmt>} clock_rise_max_fall {} clock_rise_min_rise {} clock_rise_min_fall {} clock_fall_max_rise {} clock_fall_max_fall {}`

`ref_object`

Type: object

Value: `<port | pin>`

`reference_pin`

`reference_pin` relative to the delay

Type: object

Value: `<port | pin>`

4.3.2.46 set_operating_conditions

`analysis_type`

Type: string

Value: `<single | bc_wc | on_chip_variation>`

If using `min` or `max`, default is `bc_wc;` otherwise, default is `single.`

condition	Type: string Value: <operating_condition>
corners	Type: list Value: {min {condition <> min_library <> sdcstmt <>} max {}}
library	Type: list/objects Value: list of libraries
ref_object	Type: object Value: <current_design | cell | port>

4.3.2.47 set_port_fanout_number

corners	Type: list Value: {min {value <> sdcstmt <>} max {value <> sdcstmt}}
ref_object	Type: object Value: <port>

4.3.2.48 set_propagated_clock

ref_object	Type: object Value: <clock | cell | port | pin> Note: "cell" is supported only by DC

4.3.2.49 set_resistance

corners	Type: list Value: {min {value <> sdcstmt <>} max {value <> sdcstmt}}

`ref_object`	Type: object Value: `<net>`

4.3.2.50 set_scan_signal

`chains`	Type: string Value: `{chain_name...}`
`hookup_pin`	Type: object Value: `<pin>`
`hookup_sense`	Type: string Value: `<non_inverted | inverted>`
`ref_object`	Type: object Value: `<port>`
`type`	Type: string Value: `<test_clock | test_scan_clock | test_scan_clock_a | test_scan_clock_b | test_scan_enable | test_scan_enable_inverted | test_scan_in | test_scan_out>`

4.3.2.51 set_timing_derate

`cell_check`	Type: list Value: `0`
`cell_check`	(for delay) Type: list Value: `1`

`corners`

(for delay)

Type: list

Value:

```
{cell_delay_clock_early_max_rise {value <>
sdcstmt <>} cell_delay_clock_early_max_fall {}
cell_delay_clock_early_min_rise {}
cell_delay_clock_early_min_fall {}
cell_delay_clock_late_max_rise {}
cell_delay_clock_late_max_fall {}
cell_delay_clock_late_min_rise {}
cell_delay_clock_late_min_fall {}
cell_delay_data_early_max_rise {}
cell_delay_data_early_max_fall {}
cell_delay_data_early_min_rise {}
cell_delay_data_early_min_fall {}
cell_delay_data_late_max_rise {}
cell_delay_data_late_max_fall {}
cell_delay_data_late_min_rise {}
cell_delay_data_late_min_fall {}
net_delay_clock_early_max_rise {}
net_delay_clock_early_max_fall {}
net_delay_clock_early_min_rise {}
net_delay_clock_early_min_fall {}
net_delay_clock_late_max_rise {}
net_delay_clock_late_max_fall {}
net_delay_clock_late_min_rise {}
net_delay_clock_late_min_fall {}
net_delay_data_early_max_rise {}
net_delay_data_early_max_fall {}
net_delay_data_early_min_rise {}
net_delay_data_early_min_fall {}
net_delay_data_late_max_rise {}
net_delay_data_late_max_fall {}
net_delay_data_late_min_rise {}
net_delay_data_late_min_fall {}}
```

`corners`	(for constraints) Type: list Value: `{early_max_rise {value <> sdcstmt <>} early_max_fall {} early_min_rise {} early_min_fall {} late_max_rise {} late_max_fall {} late_min_rise {} late_min_fall {}}`
`ref_object`	Type: object Value: `<current_design | instance | net | libcell>`

4.3.2.52 set_units

`ref_object`	Type: object Value: `<current_design>` Use `DESIGN` for current design

`units`	Type: list
	Value: `{time {value <> sdcstmt <>}`
	  `capacitance {}`
	  `resistance {}`
	  `voltage {}`
	  `current {}`
	  `static_power {}}`
	Time unit in second "s", otherwise NULL
	Capacitance unit in farad "F", otherwise NULL
	Resistance unit in ohm "Ohm", otherwise NULL
	Voltage unit in volt "V", otherwise NULL
	Current unit in amp "A", otherwise NULL
	Static power unit in watt "W",

4.3.2.53 set_voltage

`ref_object`

Type: object

Value: `<net | pin>`

`corners`

Type: list

Value:
`{max {value <> dynamic <> sdcstmt <>} min {}}`

4.3.2.54 set_wire_load_mbs

`value`

Type: float

Value: `<block_size>`

4.3.2.55 set_wire_load_mode

`value`

Type: string

Value: `<top | enclosed | segmented>`

4.3.2.56 set_wire_load_model

`corners`

Type: list

Value: `{min {wlm_name <> library <> sdcstnt <>} max {}}`

`ref_object`

Type: object

Value: `<current_design | instance | port>`

4.3.2.57 set_wire_load_selection_group

`corners`

Type: list

Value: `{min {wlmsg_name <> library <> sdcstmt <>} max {}}`

`ref_object`

Type: object

Value: `<current_design | cell>`

4.4 SDCSTMT

The following lists the attributes for SDCSTMT:

`command_args`

Lists all the interpreted command arguments. This list includes all the arguments that CCD supports--regardless of whether they are SDC compliant; it also includes non-SDC commands (commands that are not part of SDC).

Even-numbered fields contain option names; odd-numbered fields contain values.

Aguments that are considered flags should have their value set to true.

Type: list/string

Value: `{<option> <value> <option> <value>}`

Note: Even field names that start with "-" are options that are part of the command. Name "value" and "reference" are not options in the command, but they are introduced to make the list to be consistent. For example:

```
set_input_delay 3.0 -clock myclk [get_ports pi_a]

set sid [find -sdcstmt set_input_delay]

array set mycmdargs [get_attribute $sid cmd_args]

puts $mycmdargs (reference)
# this will print out pi_a

puts $mycmdargs (value)
# this will print out 3.0
```

`command_name`

Name of the SDC command

Type: string

Value: `<sdc_command_name>`

`command_string`

Original command, as is.

Type: string

Value: entire command, in string form

`compared_data`

SDC comparison results

Type: list

Value: `{id <no> status <pass | fail> match_sdcstmt {} withness {path}}`

`design_type`

Design type

Type: string

Value: `<golden | revised>`

`end_line`

End line number of the instance

Type: integer

Value: `<line_no_end>`

`id`

Identification of SDC statement

Type: integer

Value: `<sdcstmt_id>`

`is_obj_access`

Set to 1 when the command is an SDC object access command. Note: all collection commands are considered object access commands.

Type: boolean

Value: `<0 | 1>`

`is_overwritten`

1 if the SDC statement is overwritten, if not 0

Type: boolean

Value: `<0 | 1>`

`is_renamed`

1 if the reference has been renamed, if not 0

Type: boolean

Value: `<0 | 1>`

`is_sdc_compliant`

All commands, including object access commands, that are part of the SDC version (`sdc_version`) will have the value of 1.

Type: boolean

Value: <0 | 1>

`location`

Location of the SDC command

Type: string

Value:
`{<file_name>}`

`name`

Name of the SDCSTMT. This is derived from the SDC command name. The body of the SDC command is stored in the `cmd_args` attribute to avoid name collision between CCD attribute names and command arguments.

Type: string

Value: `<command_name>_<id>`

`object_type`

Type: string

Value: `sdcstmt`

`overwritten_history`

Overwritten history

Type: list

Value:
`{<latest(N)_statement> <N-1> <N-2> <first>}`

`renamed_objects`

Rename object list

Type: list

Value: `{<original_object_name> <renamed_object_name>}`

`sdc_lint_rules`

All lint occurences that relate to SDCSTMT

Type: list/objects

`sdc_version`

SDC version when the command is read in

Type: float

Value: `<sdc_version>`

`sdcdsgn`

sdcdsgn

Type: object

Value: `<sdcdsgn_object>`

`sdcmode`

Null if in single-mode; otherwise, SDC mode name

Type: object

Value: `<sdcmode_object>`

`start_line`

Start line number of the instance

Type: integer

Value: `<line_no_begin>`

`status`

Status of the command. A status of `unhandled` indicates that the software recognizes it is a command but does not do anything with it. Commands that are registered through `add_unhandled` should have the status of `unhandled`, (regardless if it comes from system defaults). A status of `unknown` indicates that the software is not aware of the command.

Type: string

Value: `<pass | fail | unhandled | unknown>`

Examples

- The following command returns SDC statements:

  ```
  find -sdcstmt -of_objects <list_of_sdcobj_objects>
  ```

- The following command returns `SDCSTMT` for all SDC statements read in:

  ```
  set allsdc [find -sdcstmt]
  ```

- The following command returns `SDCSTMT` for SDC statements with errors:

  ```
  set failsdc [find -sdcstmt -filter {status != pass}]
  ```

- The following command returns SDCSTMT for all `set_false_path` statements:

  ```
  set all_sfp [file -sdcstmt -filter {name == set_false_path}]
  ```

- All the SDC constructs read in with the `READ SDC` command are accessible through `SDCSTMT` objects, regardless their status (pass, fail, unhandled, or unknown). To find “pass” `SDCSTMT`s: you would run the following command:

  ```
  find -sdcstmt -filter {status == pass}
  ```

4.5 SDCMODE

The following lists the attributes for SDCMODE:

`design_type`	Design type Type: string Value: `<golden	revised>` Set to 1 if the design is used in an elaborated tree; otherwise, set to 0. Type: boolean Value: <0 | 1>
`library`	Library name where the design is stored. Type: string Value: <library_name>
`name`	Name of the SDCMODE. This is provided when creating a mode. Type: string Value: `<mode_name>`
`object_type`	Type: string Value: `sdcmode`

Examples

- The following commands return SDC modes:

  ```
  find -sdcmode -of_objects <list_of_sdcobj_objects>
  find -sdcmode -of_objects <list_of_sdcstmt_objects>
  ```

- The following command returns `SDCMODE` for all SDC modes in the design:

  ```
  set mymodes [find -sdcmode]
  ```

4.6 SDCDSGN

The following lists the attributes for SDCDSGN:

`hier_sep`	Type: string Value: `{@, #, | , ^}`
`name`	Name of the SDCDSN. This is derived from the `SET SDC DESIGN` command. Type: string Value: `</(root) | ||(glue) | instance_name>`
`object_type`	Type: string Value: `sdcmode`
`options`	Type: list Value: `{<option_name <value>......}`
`rename_rules`	Type: list Value: `{<rulename> {<original> <new>} <rulename>{....} ....}`
`sdcmode`	SDCMODE where SDCDSN belongs to. Type: string Value: `<current_mode_name>` Default is functional.
`units`	Type: list Value: `{{<unit> <number>} {...}}`

Examples

- The following commands return SDC designs:

  ```
  find -sdcdsgn -of_objects <list_of_sdcobj_objects>
  find -sdcdsgn -of_objects <list_of_sdcstmt_objects>
  ```

- The following command returns `SDCDSGN` for all nodes with SDCs attached:

```
    set mynodes [find -sdcdsgn]
```

5

FIFO Objects

You can add components to new FIFO instances using the `set_attribute` command. The following table describes the FIFO attributes. Some of these attributes are proved by the atomic checks described in *Conformal Constraint Designer Rule Check Reference*.

If an attribute is read-write, its value can be set through the set_attribute command. The value of read-only attributes is set by the tool.

- atomic_check_fifo on page 118
- atomic_check_memory on page 118
- atomic_check_raddr on page 118
- atomic_check_rgray on page 119
- atomic_check_waddr on page 119ff
- atomic_check_wdata on page 119
- atomic_check_wgray on page 120
- check_async_mem on page 120
- check_gray_comb_loop on page 121
- check_gray_func on page 121
- check_gray_size on page 122
- check_mem_out_exclusive on page 122
- check_mem_out_size on page 123
- check_mem_size on page 123
- check_mem_supported_cell_type on page 124
- check_mem_two_dimension on page 124
- check_mem_supported_cell_type on page 124

5.1 atomic_check_fifo

Lists the atomic checks at the FIFO level.

Read-only FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.2 atomic_check_memory

Lists the atomic checks related to memory.

Read-only FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.3 atomic_check_raddr

Lists the atomic checks related to read address registers.

Read-write FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.4 atomic_check_rgray

Lists the atomic checks related to read gray code registers.

Read-write FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.5 atomic_check_waddr

Lists the atomic checks related to write address registers.

Read-write FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.6 atomic_check_wdata

Lists the atomic checks related to write data.

Read-write FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.7 atomic_check_wgray

Lists the atomic checks related to write gray code registers.

Read-write FIFO attribute.

Type

list/list

Value

```
{1 <atomic_check_name> {status <> analysis_results <>}  .....
```

5.8 check_async_mem

When set to 1, the tool checks whether the memory and output registers are asynchronous.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_async_mem
```

Type

boolean

Values

< 1 | 0 >

5.9 check_gray_comb_loop

Set to 1 to check if there is a combinational loop for a gray code register.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_gray_comb_loop
```

Type

boolean

Value

< 1 | 0 >

5.10 check_gray_func

Set to 1 to run functional checks on FIFO read/write gray code registers.

Read-write FIFO attribute.

Default is 0.

Atomic Check

```
fifo_chk_atomic_gray_func
```

Type

boolean

Value

<0 | 1>

5.11 check_gray_size

When set to 1, the tool runs checks that the gray code size is equal to or greater than the minimum read graycode size.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_gray_size`

Type

`boolean`

Value

<1|0>

5.12 check_mem_out_exclusive

Set to 1 to check if the memory and output registers are exclusive.

Default is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_mem_out_exclusive
```

Type

boolean

Value

< 1 | 0 >

5.13 check_mem_out_size

Set to 1 to check if the memory size is a multiple of the output size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_out_size
```

Type

boolean

Value

< 1 | 0 >

5.14 check_mem_size

Set to 1 to checks that the memory size is equal to or larger than the minimum memory size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_size
```

Type

boolean

Value

< 1 | 0 >

5.15 check_mem_supported_cell_type

Set to 1 to check if all the element types are supported cell types.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_mem_supported_cell_type
```

Type

boolean

Value

< 1 | 0 >

5.16 check_mem_two_dimension

Set to 1 to check if the memory is a two-dimensional register array.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_mem_two_dimension`

Type

boolean

Value

< 1 | 0 >

5.17 check_out_size

Set to 1 to check that the output size is equal to or greater than the minimum output size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

`fifo_chk_atomic_out_size`

Type

boolean

Value

< 1 | 0 >

5.18 check_readptr_size

Set to 1 to checks that the read point size is equal to or larger than the minimum read pointer size.

Read-write FIFO attribute.

Default is 1.

Atomic Check

```
fifo_chk_atomic_readptr_size
```

Type

boolean

Value

< 1 | 0 >

5.19 check_readptr_sync

When set to 1, the tool checks if the read pointer is synchronous to the output.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

```
fifo_chk_atomic_readptr_sync
```

Type

boolean

Value

<1|0>

5.20 check_single_rgray

When set to 1, the tool checks if there is only one read graycode register candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_rgray`

Type

boolean

Value

<1|0>

5.21 check_single_rptr

When set to 1, the tool checks if there is only one read pointer candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_readptr`

Type

boolean

Value

<1|0>

5.22 check_single_synch

When set to 1, the tool checks if there is only one sync candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_sync`

Type

boolean

Value

<1|0>

5.23 check_single_wgray

When set to 1, the tool checks if there is only one write graycode register candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_wgray`

Type

boolean

Value

<1|0>

5.24 check_single_wptr

When set to 1, the tool checks if there is only one write pointer candidate.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_single_writeptr`

Type

boolean

Value

<1|0>

5.25 check_sync_size

When set to 1, the tool checks if the sync size is equal to the gray code candidate size.

Default value is 1.

Read-write FIFO attribute.

Atomic Check

`fifo_chk_atomic_sync_size`

Type

boolean

Value

<1|0>

5.26 check_wdata_size

Set to 1 to checks if the minimum wdata size has been exceeded.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

```
fifo_chk_atomic_wdata_size
```

Type

boolean

Value

< 1 | 0 >

5.27 check_writeptr_size

Set to 1 to checks if the write point size is equal to or greater than the minimum write pointer size.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

```
fifo_chk_atomic_writeptr_size
```

Type

boolean

Value

< 1 | 0 >

5.28 check_writeptr_sync

Set to 1 to checks if the write pointer is synchronuos to the FIFO memory.

Read-write FIFO attribute.

Default value is 1.

Atomic Check

```
fifo_chk_atomic_writeptr_sync
```

Type

boolean

Value

< 1 | 0 >

5.29 design_type

Specifies whether the design where FIFO was extracted is a Golden or Revised design.

Read-only FIFO attribute.

Type

string

Value

```
<golden | revised>
```

5.30 end_line

Specifies the end line number of the memory.

Read-only FIFO attribute.

Type

integer

Value

```
<line_no_end>
```

5.31 location

Specifies the file name where memory is instantiated.

Read-only FIFO attribute.

Type

string

Value

```
<full_path_to_file_name>
```

5.32 memory

Lists the objects inferred for memory.

Read-write FIFO attribute.

Type

list/object

Value

```
<list_of_memory_objects>
```

5.33 min_mem_size

Minimum FIFO memory size.

Read-write FIFO attribute.

Type

integer

5.34 min_out_size

Minimum FIFO output size.

Read-write FIFO attribute.

Type

integer

5.35 name

Name of the FIFO object.

Read-write FIFO attribute.

Type

string

Value

```
<object_name>
```

5.36 object_type

Specifies the object type.

Read-only FIFO attribute.

Type

string

Value

```
fifo
```

5.37 raddr

List of objects inferred for read address.

Read-write FIFO attribute.

Type

list/object

Value

```
<list_of_objects_for_read_address>
```

5.38 rdata

List of objects inferred for read data.

Read-write FIFO attribute.

Type

list/object

Value

```
<list_of_objects_for_read_data>
```

5.39 rgraycode

List of objects inferred for read gray-code.

Read-write FIFO attribute.

Type

list/object

Value

```
<list_of_objects_for_readadr_gray-code>
```

5.40 rsync

List of objects inferred for read synchronizer.

Read-write FIFO attribute.

Type

list/object

Value

```
<list_of_objects_for_readadr_synchronizer>
```

5.41 start_line

Start line number of the memory.

Read-only FIFO attribute.

Type

integer

Value

`<line_no_begin>`

5.42 status

Indicates the status of the FIFO, where 1 indicates pass.

Read-only FIFO attribute.

Type

boolean

Value

<0 | 1>

5.43 waddr

List of objects inferred for write address.

Read-write FIFO attribute.

Type

list/object

Value

`<list_of_objects_for_write_address>`

5.44 wdata

List of objects inferred for write data.

Read-write FIFO attribute.

Type

list/object

Value

`<list_of_objects_for_write_data>`

5.45 wgraycode

List of objects inferred for write gray-code.

Read-write FIFO attribute.

Type

list/object

Value

`<list_of_objects_for_writeadr_gray-code>`

5.46 wsync

List of objects inferred for write synchronizer.

Read-write FIFO attribute.

T

ype

list/object

Value

```
<list_of_objects_for_writeadr_synchronizer>
```

6

GUI Objects

This section describes the attributes that fall under the GUI object type.

- SRCVIEW on page 140
- SCHVIEW on page 142

6.1 SRCVIEW

Describes attributes that control the Source Code Viewer.

- location on page 141
- focus_line on page 140
- location on page 141
- location on page 141
- name on page 141
- object_type on page 141

6.1.1 focus_color

Read-write attribute.

Type

string

Value

```
<valid_color>
```

6.1.2 focus_line

Read-write attribute.

Type

integer

Value

```
<line_no>
```

6.1.3 location

Type

string

Value

```
<full_path_to_the_file>
```

6.1.4 name

Type

string

Value

```
<name_of_the_window>
```

6.1.5 object_type

Type

string

Value

```
<srcview | schview | rulemngr | textview | cfmwdgt>
```

6.2 SCHVIEW

Describes the attributes that control the Schematic Viewer.

6.2.1 highlight_objects

Specifies which design objects to highlight in the Schematic Viewer and what color to use.

Type

list/list

Value

```
<color>{<list_of_design_objects>} <color
{<list_of_design_objects>}..}
```

6.2.2 message

Specifies a message to display in the message box for the design object in the Schematic Viewer.

Type

string

Value

`<message>`

6.2.3 name

Specifies the window name.

Type

string

Value

`<name_of_the_window>`

6.2.4 object_type

Type

string

Value

`<srcview | schview | rulemngr | textview | cfmwdgt>`

6.2.5 show_info_box

Specifies the cells for which an info box should be displayed.

Type

list/object

Value

```
{<info_box> {<list_obj_names>} <info_box> {<list_obj_names>}...}>
```

6.2.6 show_inst_names

Controls whether instance names are displayed.

Type

boolean

Value

```
<on | off>
```

6.2.7 show_net_names

Controls whether net names are displayed.

Type

boolean

Value

```
<on | off>
```

6.2.8 show_next_level

Controls whether the next-level view is displayed.

Type

list/object

Value

```
<objects_inside_to_be_shown>
```

6.2.9 show_pin_names

Controls the display of pin names.

Type

boolean

Value

`<on | off>`

6.2.10 show_port_names

Controls whether port names are displayed.

Type

boolean

Value

`<on | off>`

6.2.11 show_sdc_refs

Displays the SDC reference points in the schematic.

Type

boolean

Value

`<on | off>`

6.2.12 tag

Specifies a tag to attach to the design object in the Schematic Viewer.

Type

string

Value

<tag>

7

Design Objects

Design objects are unchanged after modules are synthesized and the complete design hierarchy is created and the design is elaborated (using the ELABORATE command).

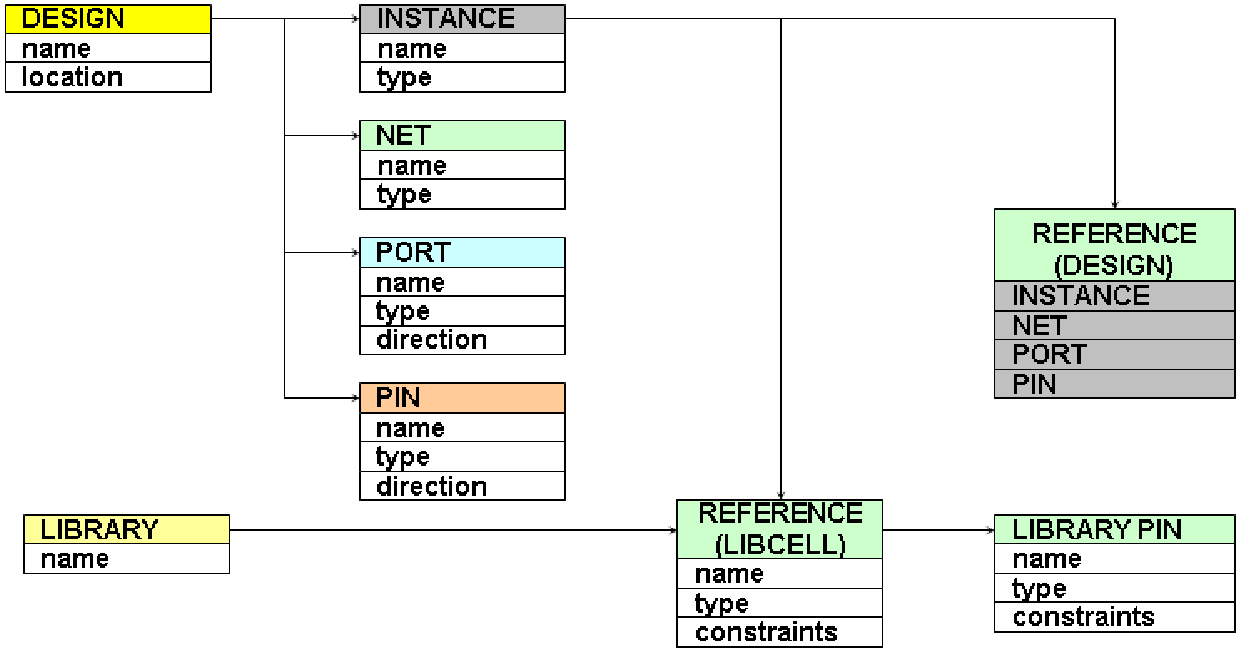
Note: If re-elaboration is performed after accessing the design objects, pointers to the design objects from the previous elaboration will be invalid.

If an attribute is described as **read-write**, you can set its value using the set_attribute command. The tool sets the values for **read-only** attributes.

See also Common Attributes for Design Objects on page 149.

7.1 Design Object Overview

The following figure illustrates how design objects relate to each other.



7.2 Common Attributes for Design Objects

The following lists the common attributes for design object types.

- design_type on page 149
- end_line on page 149
- location on page 150
- name on page 150
- object_type on page 150
- start_line on page 151
- sdc_constraints on page 151

7.2.1 design_type

Read-only attribute. Specifies whether this is the Golden or Revised design

Type

string

Value

```
<golden | revised>
```

7.2.2 end_line

Read-only attribute. End line number of the object

Type

integer

Value

```
<line_no_end>
```

7.2.3 location

Read-only attribute. Location of the file where it is instantiated

Type

string

Value

```
<file_name>
```

7.2.4 name

Name of the object

Type

string

Value

```
<object_name>
```

7.2.5 object_type

Read-only attribute.

Type

string

Value

```
{<design | instance | port | pin |
net | library | libcell | libpin>}
```

7.2.6 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to the specified object. Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_attached_to_the_object>}
```

7.2.7 start_line

Read-only attribute. Start line number of the object

Type

integer

Value

```
<line_no_begin>
```

7.3 DESIGN

The following lists the attributes for the current design objects.

- is_in_elaborated_tree on page 152
- is_protected on page 152
- library on page 152

See also Common Attributes for Design Objects on page 149.

7.3.1 is_in_elaborated_tree

Read-only attribute. When set to 1, it means the design is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

```
<0 | 1>
```

7.3.2 is_protected

Read-only attribute. When set to 1, it means the design is protected.

Type

boolean

Value

```
<0 | 1>
```

7.3.3 library

Read-only attribute. Library name where the design in stored

Type

string

Value

`<library_name>`

7.4 INSTANCE

The following lists the attributes for the instance (cell) design objects. See also Common Attributes for Design Objects on page 149.

- bbox_type on page 155
- clock_gating_cell_type on page 155
- full_name on page 156
- is_bbox on page 156
- is_clock_gating_cell on page 156
- is_dont_touch on page 157
- is_dont_use on page 157
- is_in_elaborated_tree on page 157
- is_leaf_cell on page 158
- is_pad_cell on page 158
- is_sdc_node on page 158
- is_tool_generated_name on page 159
- library on page 159
- power_domain on page 159
- parent on page 160
- ref_name on page 160
- sdc_constraints on page 160
- type on page 161

Instance Examples

The following commands return instances cells in the current level of hierarchy, all the cells in the design, and all the flip-flops in the design:

```
set mycells [find -instance]
set mycells [find -instance -hierarchical]
set allffs [find -instance -hierarchical -filter {type == flipflop}]
```

The following commands return instances:

```
find -instance -of_objects <list_of_pin_objects>
find -instance -of_objects <list_of_net_objects>
```

7.4.1 bbox_type

Read-only attribute. Specifies the blackbox type.

Type

string

Value

```
<user | timing | undefined | empty
      | unsupported | notranslate>
```

where the values are defined as:

- `user`: blackboxes added previously with the `ADD BLACK BOX` command
- `timing`: cells that do not have a function defined but have timing arcs
- `undefined`: cells that do not have a module definition and were blackboxed due to the ’`SET UNDEFINED CELL Black_box`’ command
- `empty`: blackboxes for empty modules
- `unsupported`: blackboxes for unsupported modules
- `notranslate`: cells that were blackboxed due to the `ADD NOTRANSLATE MODULES` command

7.4.2 clock_gating_cell_type

Read-only attribute. String value from libcell attribute `clock_gating_integrated_cell`

Type

string

Value

`<string | generic>`

7.4.3 full_name

Read-only attribute. Hierarchical name of the instance from root.

Type

string

Value

`<hierarchical_instance_name>`

7.4.4 is_bbox

Read-only attribute. When set to 1, the object is a blackbox. Refer to the `bbox_type` attribute for a list of situations where an object is considered a blackbox.

Type

boolean

Value

`<0 | 1>`

7.4.5 is_clock_gating_cell

Read-only attribute. When set to 1, the libcell has `clock_gating_integrated_cell` in techlib.

Type

boolean

Value

`<0 | 1>`

7.4.6 is_dont_touch

Read-only attribute. When set to 1, the instance or reference has `dont_touch` in techlib.

Type

boolean

Value

`<0 | 1>`

7.4.7 is_dont_use

Read-only attribute. When set to 1, the instance or reference has `dont_use` in techlib.

Type

boolean

Value

`<0 | 1>`

7.4.8 is_in_elaborated_tree

Read-only attribute. When set to 1, the design is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

`<0 | 1>`

7.4.9 is_leaf_cell

Read-only attribute. When set to 1, the object is a leaf-level cell (libcell of primitive).

Type

boolean

Value

`<0 | 1>`

7.4.10 is_pad_cell

Read-only attribute. When set to 1, the instance or reference has `pad_cell` in techlib.

Type

boolean

Value

`<0 | 1>`

7.4.11 is_sdc_node

Read-only attribute. When set to 1, `SDCGDGN` attached to the instance.

Type

boolean

Value

`<0 | 1>`

7.4.12 is_tool_generated_name

Read-only attribute. When set to 1, the name is assigned by the tool; 0 if the name is from the design.

Type

boolean

Value

```
<0 | 1>
```

7.4.13 library

Read-only attribute. Specifies the library name where reference of this instance is stored.

Type

string

Value

```
<library_name>
```

7.4.14 power_domain

Read-only attribute. Specifies the power domain in which this instance belongs.

Type

object

Value

```
<power_domain>
```

7.4.15 parent

Read-only attribute. Specifies the object where this object resides. For instances defined at the top of the hierarchy, returns the top design object. For other instances, returns the containing hierarchical instance.

Type

object

Value

```
<parent_object>
```

7.4.16 ref_name

Read-only attribute. Specifies the reference (design or libcell) name.

Type

string

Value

```
<reference_name>
```

7.4.17 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to this instance

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_attached_to_instance>}
```

7.4.18 type

Read-only attribute. Type of the instances. Anything that the software does not understand will be type `bbox`.

Type

string

Value

For the cell type values, see the Conformal Primitive Gate Types appendix in the *Conformal Constraint Designer User Guide*.

7.5 PORT

The following lists the attributes for the port design objects, in alphabetical order:

- bit_width on page 163
- bus_idx on page 163
- bus_name on page 163
- capture_clocks on page 164
- clocks on page 164
- constant_value on page 164
- direction on page 165
- fanin on page 165
- fanout on page 165
- full_name on page 166
- is_constant on page 166
- is_end_point on page 166
- is_in_elaborated_tree on page 167
- is_undriven on page 167
- is_start_point on page 167
- is_tool_generated_name on page 168
- is_timing_end_ point on page 168
- is_timing_start_ point on page 168
- launch_clocks on page 169
- lsb on page 169
- msb on page 169
- p_blocked_clocks on page 170
- p_capture_clocks on page 170
- p_clocks on page 170

- p_constant_value on page 171
- p_is_constant on page 171
- p_launch_clocks on page 171
- parent on page 172
- power_domain on page 172
- sdc_constraints on page 172

7.5.1 bit_width

Read-only attribute. Specifies the bitwidth of the port.

Type

integer

Value

```
<bit_width>
```

7.5.2 bus_idx

Read-only attribute. Specifies the index number for the bus to which the port belongs.

Type

integer

Value

```
<index_number>
```

7.5.3 bus_name

Read-only attribute. Specifies the bus name.

Type

string

Value

`<name_of_the_array>`

7.5.4 capture_clocks

Read-only attribute. Specifies the capture clock of the port. For the port objects, this attribute contains the capture clocks defined using the `set_output_delay` command.

Type

list/list

Value

```
{{<sdcobj> <phase>} ....}
```

Where phase can be `pos`, `neg`, or `both`.

7.5.5 clocks

Read-only attribute. Specifies the clock defined on this object and includes the phase information.

Type

list/sdcobj

Value

```
{<clock><phase><clock><phase>...}
```

7.5.6 constant_value

Read-only attribute. Contains the constant value followed by the SDC constraint that caused the constant value or by the object itself if it is tied in the design.

Type

boolean

Value

`{<0 | 1><constraint_or_constant>...}`

7.5.7 direction

Read-only attribute. Specifies the direction of the port.

Type

string

Value

`<in | out | inout>`

Example

The following command finds all the input ports that are not bidirectional:

```
set clkports [find -port -filter {direction==in}]
```

7.5.8 fanin

Read-only attribute. Specifies the fanin of the port.

Type

list

Value

`{<list_of_pin/port_objects>}`

7.5.9 fanout

Read-only attribute. Specifies the fanout of the port.

Type

list

Value

`{<list_of_pin/port_objects>}`

7.5.10 full_name

Read-only attribute. This attribute is similar to attribute “`name`”; it has been provided for consistency with pin objects.

Type

string

Value

`<port_name>`

7.5.11 is_constant

Read-only attribute. When set to 1, it means a constant value is set on this object. The constant value is specified by the constant_value attribute.

Type

boolean

Value

<0 | 1>

7.5.12 is_end_point

Read-only attribute. When set to 1, the object is a structural endpoint.

Type

boolean

Value

`<0 | 1>`

7.5.13 is_in_elaborated_tree

Read-only attribute. When set to 1, the design is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

`<0 | 1>`

7.5.14 is_undriven

Read-only attribute. When set to 1, the port is not driven by a driver.

Type

boolean

Value

`<0 | 1>`

7.5.15 is_start_point

Read-only attribute. When set to 1, the object is a structural startpoint.

Type

boolean

Value

`<0 | 1>`

7.5.16 is_tool_generated_name

Read-only attribute. When set to 1, the name is assigned by the tool; 0 if the name is from the design.

Type

boolean

Value

`<0 | 1>`

7.5.17 is_timing_end_ point

Read-only attribute. When set to 1, the object is a valid timing endpoint.

Type

boolean

Value

`<0 | 1>`

7.5.18 is_timing_start_ point

Read-only attribute. When set to 1, the object is a valid timing startpoint.

Type

boolean

Value

`<0 | 1>`

7.5.19 launch_clocks

Read-only attribute. Specifies the launch clock of the port. For the port objects, this attribute contains the clocks derived using the `set_input_delay` command.

Type

list/list

Value

`{{<sdcobj> <phase>} .....}`

Where phase can be `pos`, `neg`, or `both`.

7.5.20 lsb

Read-only attribute. Specifies the LSB port.

Type

integer

Value

`<lsb>`

7.5.21 msb

Read-only attribute. Specifies the MSB port.

Type

integer

Value

`<msb>`

7.5.22 p_blocked_clocks

Read-only attribute. Specifies the clocks whose propagation was blocked by constraints or constants from the design. The resulting value will show the tied pin or net causing the clock propagation to be blocked.

Type

list/list

Value

```
{<blocked_clock><blocking_constraints_or_constants>...}
```

7.5.23 p_capture_clocks

Read-only attribute. Specifies the capture clock that propagates to the port.

Type

list/list

Value

```
{{<sdcobj> <phase>} .....}
```

Where phase can be `pos`, `neg`, or `both`.

7.5.24 p_clocks

Read-only attribute. Specifies the propagated clock(s) and their phase information.

Type

list/sdcobj

Value

```
{<clock><phase><clock><phase>...}
```

7.5.25 p_constant_value

Read-only attribute. Specifies the constant values propagated to this object, followed by the SDC constraints and/or objects whose values determine the propagated constant value.

Type

value/list

Value

```
{<0|1><constraint_or_constant>...}
```

7.5.26 p_is_constant

Read-only attribute. Constant, due to a propagated value

Type

boolean

Value

```
<0 | 1>
```

7.5.27 p_launch_clocks

Read-only attribute. Specifies the launch clock that propagates to the port.

Type

list/list

Value

```
{{<sdcobj> <phase>} .....}
```

Where phase can be `pos`, `neg`, or `both`.

7.5.28 parent

Read-only attribute. Specifies the object where this object resides. For ports, this is the top-level design object.

Type

object

Value

<parent_object>

7.5.29 power_domain

Read-only attribute. Specifies the power domain for which this instance belongs (this attribute relates to Low Power).

Type

object

Value

`<power_domain>`

7.5.30 sdc_constraints

Read-only attribute. Specifies the list of SDC objects (SDCOBJ) attached to this port.

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_port_is_constrained>}
```

7.6 PIN

The following lists the attributes for the pin design objects, in alphabetical order:

- active_phase on page 174
- bit_width on page 175
- bus_idx on page 175
- capture_clocks on page 175
- clocks on page 176
- constant_value on page 176
- direction on page 176
- fanin on page 177
- fanout on page 177
- full_name on page 177
- ideal_source on page 178
- is_clock on page 178
- is_constant on page 178
- is_constant_pin on page 179
- is_end_point on page 179
- is_floating on page 179
- is_in_elaborated_tree on page 180
- is_leaf_pin on page 180
- is_set_reset on page 180
- is_start_point on page 181
- is_tool_generated_name on page 181
- is_timing_end_point on page 181
- is_timing_start_point on page 182
- is_undriven on page 182

7.6.1 active_phase

Read-only attribute. Specifies the active phase of the clock/enable/set/reset pin.

Type

string

Value

```
<posedge | negedge | bothedge | high | low
| null ("")>
```

7.6.2 bit_width

Read-only attribute. Specifies the bitwidth of the pin.

Type

integer

Value

```
<bit_width>
```

7.6.3 bus_idx

Read-only attribute. Specifies the index number of the bus to which the pin belongs.

Type

integer

Value

```
<index_number>
```

7.6.4 bus_name

Read-only attribute. Specifies the name of the bus.

Type

string

Value

```
<name_of_the_array>
```

7.6.5 capture_clocks

Read-only attribute. Specifies the capture clock of the pin.

Type

list/list

Value

```
{{<sdcobj> <phase>} .....}
```

Where phase can be `pos`, `neg`, or `both`.

7.6.6 clocks

Read-only attribute. Specifies the clock defined on this object and includes the phase information.

Type

list/sdcobj

Value

```
{<clock><phase><clock><phase>...}
```

7.6.7 constant_value

Read-only attribute. Contains the constant value followed by the SDC constraint that caused the constant value or by the object itself if it is tied in the design.

Type

boolean

Value

```
{<0 | 1><constraint_or_constant>...}
```

7.6.8 direction

Read-only attribute. Specifies the direction of the pin

Type

string

Value

```
<in | out | inout | internal>
```

7.6.9 fanin

Read-only attribute. Specifies the fanin of the pin.

Type

list

Value

```
{<list_of_pin/port_objects>}
```

7.6.10 fanout

Read-only attribute. Specifies the fanout of the pin.

Type

list

Value

```
{<list_of_pin/port_objects>}
```

7.6.11 full_name

Read-only attribute. Specifies the full path to the pin.

Type

string

Value

`<hierarchical_pin_name>`

7.6.12 ideal_source

Read-only attribute. Specifies the SDCOBJ from which the `is_ideal` attribute is derived.

Type

object

Value

`<sdcobj_of_set_ideal_network/set_ideal_net>`

7.6.13 is_clock

Read-only attribute. When set to 1, the pin is a clock pin.

Type

boolean

Value

`{<0 | 1>}`

7.6.14 is_constant

Read-only attribute. When set to 1, a constant value is set on this object. The constant value is specified by the constant_value attribute.

Type

boolean

Value

<0 | 1>

7.6.15 is_constant_pin

Read-only attribute. When set to 1, the pin is tied to constant.

Type

boolean

Value

`{<0 | 1>}`

7.6.16 is_end_point

Read-only attribute. When set to 1, the object is a structural endpoint.

Type

boolean

Value

`<0 | 1>`

7.6.17 is_floating

Read-only attribute. When set to 1, the pin is not connected.

Type

boolean

Value

`<0 | 1>`

7.6.18 is_in_elaborated_tree

Read-only attribute. When set to 1, the design is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

```
<0 | 1>
```

7.6.19 is_leaf_pin

Read-only attribute. When set to 1, the pin is a leafcell (libcell or primitive) pin.

Type

boolean

Value

```
{<0 | 1>}
```

7.6.20 is_pad

Read-only attribute. When set to 1, the pin has "`is_pad:true`" specified in the techlib.

Type

boolean

Value

```
{<0 | 1>}
```

7.6.21 is_set_reset

Read-only attribute. When this is set to 1, the pin is set or reset.

Type

boolean

Value

```
{<0 | 1>}
```

7.6.22 is_start_point

Read-only attribute. When set to 1, the object is a structural startpoint.

Type

boolean

Value

```
<0 | 1>
```

7.6.23 is_tool_generated_name

Read-only attribute. When set to 1, the name is assigned by the tool; 0 if the name is from the design.

Type

boolean

Value

```
<0 | 1>
```

7.6.24 is_timing_end_point

Read-only attribute. When set to 1, the object is a valid timing end point.

Type

boolean

Value

```
<0 | 1>
```

7.6.25 is_timing_start_point

Read-only attribute. When set to 1, the object is a valid timing start point.

Type

boolean

Value

```
<0 | 1>
```

7.6.26 is_undriven

Read-only attribute. When set to 1, the pin is not driven by a driver.

Type

boolean

Value

```
<0 | 1>
```

7.6.27 launch_clocks

Read-only attribute. Specifies the launch clock of the pin.

Type

list/list

Value

```
{{<sdcobj> <phase>} .....}
```

Where phase can be `pos`, `neg`, or `both`.

7.6.28 lower_net

Read-only attribute. Specifies the inside net that is connected to the hierarchical cell pin

Type

object

Value

`<net_inside_of_the_hierarchical_cell>`

7.6.29 lsb

LSB pin

Type

integer

Value

`<lsb>`

7.6.30 msb

Read-only attribute. Specifies the MSB pin.

Type

integer

Value

`<msb>`

7.6.31 p_blocked_clocks

Read-only attribute. Specifies the clocks whose propagation was blocked by constraints.

Type

list/list

Value

```
{<blocked_clock><constraints_blocked>...}
```

7.6.32 p_capture_clocks

Read-only attribute. Specifies the capture clock that propagates to the pin.

Type

list/list

Value

```
{{<sdcobj> <phase>} .....}
```

Where phase can be `pos`, `neg`, or `both`.

7.6.33 p_clocks

Read-only attribute. Specifies the propagated clock(s) with phase information

Type

list/sdcobj

Value

```
{<clock><phase><clock><phase>...}
```

7.6.34 p_constant_value

Read-only attribute. Lists the constant values propagated to this object, followed by the SDC constraints and/or objects whose values determine the propagated constant value.

Type

value/list

Value

`{<0|1><constraint_or_constant>...}`

7.6.35 p_is_constant

Read-only attribute. Indicates a propagated constant.

Type

boolean

Value

`<0 | 1>`

7.6.36 p_launch_clocks

Read-only attribute. Specifies the launch clock that propagates to the pin.

Type

list/list

Value

`{{<sdcobj> <phase>} .....}`

Where phase can be `pos`, `neg`, or `both`.

7.6.37 parent

Read-only attribute. Specifies the instance object where this object resides

Type

object

Value

```
<parent_instance>
```

7.6.38 power_domain

Read-only attribute. Specifies the power domain for which this instance belongs (this attribute relates to Low Power).

Type

object

Value

```
<power_domain>
```

7.6.39 ref_name

Read-only attribute. Specifies the reference pin name.

Type

string

Value

```
<reference_pin_name>
```

7.6.40 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to this pin

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_pin_is_constrained>}
```

7.6.41 test_cell_signal_type

Read-only attribute. Specifies the pin type of the test cell signal.

Type

string

Value

```
<test_cell_signal_pin_type>
```

7.6.42 timing_arc

7.6.43 type

Read-only attribute. Specifies the pin type of the reference cell.

Type

string

Value

```
<pin_type>
```

7.6.44 timing_arc

Read-only attribute. Specifies the timing arc type and its condition.

Type

list

Value

```
{{<related_pin> <when> <timing_sense> <timing_type>} ...}
```

Examples

- The following command returns pins of the library cell l1:

  ```
  set mylibpin [find -libpin l1/*]
  ```

- The following command returns pins of the clock pin of libcell l1:

  ```
  set clklibpin [find -libpin l1/* -filter {is_clock == 1}]
  ```

- The following command returns library pins:

  ```
  find -libpin -of_objects <libcell_object>
  ```

7.6.45 upper_net

Read-only attribute. Specifies the outside net that is connected to the hierarchical cell pin

Type

object

Value

```
<net_outside_of_the_hierarchical_cell>
```

7.7 NET

The following lists the attributes for the net design objects, in alphabetical order:

- bit_width on page 190
- bus_idx on page 190
- bus_name on page 190
- constant_value on page 191
- fanin on page 191
- fanout on page 191
- full_name on page 192
- is_clock on page 192
- is_constant on page 192
- is_floating on page 192
- is_in_elaborated_tree on page 193
- is_set_reset on page 193
- is_tool_generated_name on page 193
- is_undriven on page 194
- lsb on page 194
- msb on page 194
- p_constant_value on page 195
- p_is_constant on page 195
- parent on page 195
- sdc_constraints on page 196

Net Examples

- The following command returns pins in the cell `u1`:

```
set u1_pins [find -pin u1/*]
```

- The following commands return pins:

```
find -pin -of_objects <instance_object>
find -pin -of_objects <net_object>
```

7.7.1 bit_width

Read-only attribute. Specifies the bitwidth of the net.

Type

integer

Value

`<bit_width>`

7.7.2 bus_idx

Read-only attribute. Specifies the index number of the bus to which the net belongs.

Type

integer

Value

`<index_number>`

7.7.3 bus_name

Read-only attribute. Specifies the name of the bus.

Type

string

Value

```
<name_of_the_array>
```

7.7.4 constant_value

Read-only attribute. Contains the constant value followed by the SDC constraint that caused the constant value or by the object itself if it is tied in the design.

Type

boolean

Value

```
{<0 | 1><constraint_or_constant>...}
```

7.7.5 fanin

Read-only attribute. Lists the fanin of the net.

Type

list

Value

```
{<list_of_pin/port_objects>}
```

7.7.6 fanout

Read-only attribute. Lists the fanout of the net

Type

list

Value

```
{<list_of_pin/port_objects>}
```

7.7.7 full_name

Read-only attribute. Specifies the full path to the net from the root.

Type

string

Value

```
<hierarchical_net_name>
```

7.7.8 is_clock

Read-only attribute. When set to 1, the net is inferred as a clock.

Type

boolean

Value

```
{<0 | 1>}
```

7.7.9 is_constant

Read-only attribute. When set to 1, the net is tied to constant.

Type

boolean

Value

```
{<0 | 1>}
```

7.7.10 is_floating

Read-only attribute. When set to 1, the net is not connected.

Type

boolean

Value

`<0 | 1>`

7.7.11 is_in_elaborated_tree

Read-only attribute. When set to 1, the design is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

`<0 | 1>`

7.7.12 is_set_reset

Read-only attribute. When set to 1, the net is set or reset.

Type

boolean

Value

`{<0 | 1>}`

7.7.13 is_tool_generated_name

Read-only attribute. When set to 1, the name is assigned by the tool; 0 when the name is from the design.

Type

boolean

Value

`<0 | 1>`

7.7.14 is_undriven

Read-only attribute. When set to 1, the net is not driven by a driver.

Type

boolean

Value

`<0 | 1>`

7.7.15 lsb

Read-only attribute. Specifies the least significant bit (LSB) net.

Type

integer

Value

`<lsb>`

7.7.16 msb

Read-only attribute. Specifies the most significant bit (MSB) net.

Type

integer

Value

```
<msb>
```

7.7.17 p_constant_value

Read-only attribute. List that contains the constant value propagated to this object, followed by the SDC constraints and/or objects whose values determine the propagated constant value.

Type

value/list

Value

```
{<0|1><constraint_or_constant>...}
```

7.7.18 p_is_constant

Read-only attribute. When set to 1, the net is a constant, due to a propagated value.

Type

boolean

Value

```
<0 | 1>
```

7.7.19 parent

Read-only attribute. Specifies the object where this object resides. For nets defined at the top of the hierarchy, returns the top design object. For other instances, returns the containing hierarchical instance.

Type

object

Value

```
<parent_object>
```

7.7.20 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to this net.

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_port_is_constrained>}
```

Examples

- The following command returns clock nets:

  ```
  set clk_nets [find -net -filter {is_clock == 1}]
  ```

- The following command returns nets:

  ```
  find -net -of_objects <pin_object>
  ```

7.8 LIBRARY

The following lists the attributes for the library design objects, in alphabetical order:

- default_max_capacitance on page 197
- default_max_fanout on page 197
- default_max_transition on page 198
- default_operating_condition on page 198
- default_wire_load_mode on page 198
- default_wire_load_model on page 199
- default_wire_load_selection on page 199
- library_set_names on page 199
- operating_conditions on page 200
- units on page 200
- wire_load_model on page 200
- wire_load_selection on page 201
- wire_load_table on page 201

7.8.1 default_max_capacitance

Read-only attribute. Specifies the default maximum capacitance. NULL if there is no maximum.

Type

float

Value

```
<max_capacitance>
```

7.8.2 default_max_fanout

Read-only attribute. Specifies the default maximum number of fanouts. NULL if there is no maximum.

Type

integer

Value

```
<max_fanout>
```

7.8.3 default_max_transition

Read-only attribute. Specifies the default maximum number of transitions. NULL if there is no maximum.

Type

float

Value

```
<max_transition>
```

7.8.4 default_operating_condition

Read-only attribute. Specifies the default operating condition. NULL if there is no default specified.

Type

string

Value

```
<operating_condition>
```

7.8.5 default_wire_load_mode

Read-only attribute. Specifies the default wire load mode. NULL if there is no default specified.

Type

string

Value

```
<wlm_mode>
```

7.8.6 default_wire_load_model

Read-only attribute. Default wire load model, NULL if it does not

Type

string

Value

```
<wlm_model>
```

7.8.7 default_wire_load_selection

Read-only attribute. Default wire load selection, NULL if it does not

Type

string

Value

```
<wlm_selection>
```

7.8.8 library_set_names

Read-only attribute. Library set to which this library belongs

Type

object/list

Value

```
<list_of_library_set_names>
```

7.8.9 operating_conditions

Read-only attribute. Operating conditions in library, NULL if it does not

Type

list

Value

```
{<list_of_operating_condition_names>}
```

7.8.10 units

Read-only attribute. Specifies library units for time, capacitance, resistance, voltage, current and static power; otherwise, NULL

Type

list

Value

```
{<time_unit> <capacitive_load_unit> <pulling_resistance_unit>
<voltage_unit> <current_unit> <leakage_power_unit>}
```

7.8.11 wire_load_model

Read-only attribute. Wire load models in library, NULL if it does not

Type

list

Value

```
{<list_of_wlm_names>}
```

7.8.12 wire_load_selection

Read-only attribute. Wire load selections in library, NULL if it does not

Type

list

Value

```
{<list_of_wlm_selection_names>}
```

7.8.13 wire_load_table

Read-only attribute. Wire load tables in library, NULL if it does not

Type

list

Value

```
{<list_of_wlm_names>}
```

Examples

- The following command returns libraries:

  ```
  set alllibs [find -library]
  ```

- The following commands return libraries:

  ```
  find -lib -of_objects <list_of_instance_objects>
  find -lib -of_objects <list_of_libcell_objects>
  ```

7.9 LIBCELL

The following lists the attributes for the library cell design objects, in alphabetical order:

- bbox_type on page 202
- clock_gating_cell_type on page 203
- has_mod_instance on page 203
- is_bbox on page 203
- is_clock_gating_cell on page 204
- is_dont_touch on page 204
- is_dont_use on page 204
- is_in_elaborated_tree on page 205
- is_pad_cell on page 205
- library on page 205
- sdc_constraints on page 206
- type on page 206

7.9.1 bbox_type

Read-only attribute. Specifies the blackbox type.

Type

string

Value

```
<user | timing | undefined | empty
      | unsupported | notranslate>
```

where the values are defined as:

`user`: blackboxes added previously with the `ADD BLACK BOX` command

`timing`: cells that do not have a function defined but have timing arcs

`undefined`: cells that do not have a module definition and were blackboxed due to the ’`SET`

UNDEFINED CELL Black_box' command

empty: blackboxes for empty modules

unsupported: blackboxes for unsupported modules

notranslate: cells that were blackboxed due to the ADD NOTRANSLATE MODULES command

7.9.2 clock_gating_cell_type

Read-only attribute. String value from libcell attribute clock_gating_integrated_cell

Type

string

Value

<string | generic>

7.9.3 has_mod_instance

Read-only attribute. When set to 1, the libcell is a hierarchical libcell.

Type

boolean

Value

<0 | 1>

7.9.4 is_bbox

Read-only attribute. When set to 1, the libcell is a blackbox. Refer to the bbox_type attribute for a detailed list of all situations where an object is considered a blackbox.

Type

boolean

Value

`<0 | 1>`

7.9.5 is_clock_gating_cell

Read-only attribute. When set to 1, the libcell has `clock_gating_integrated_cell` in techlib.

Type

boolean

Value

`<0 | 1>`

7.9.6 is_dont_touch

Read-only attribute. When set to 1, the libcell has `dont_touch` in techlib.

Type

boolean

Value

`<0 | 1>`

7.9.7 is_dont_use

Read-only attribute. When set to 1, libcell has `dont_use` in techlib.

Type

boolean

Value

`<0 | 1>`

7.9.8 is_in_elaborated_tree

Read-only attribute. When set to 1, it means the library cell is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

`<0 | 1>`

7.9.9 is_pad_cell

Read-only attribute. When set to 1, libcell has `pad_cell` in techlib.

Type

boolean

Value

`<0 | 1>`

7.9.10 library

Read-only attribute. Specifies the name of the library that contains `libcell`.

Type

string

Value

`<library_name>`

7.9.11 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to this libcell.

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_libcell_is_constrained>}
```

7.9.12 type

Read-only attribute. Specifies the type of the libcell. Anything that CCD does not understand will be type of `bbox`.

Type

string

Value

For the cell type values, see the Conformal Primitive Gate Types appendix in the *Conformal Constraint Designer User Guide*.

Examples

- The following command returns all library cells read in:

  ```
  set libcells [find -libcells]
  ```

- The following command returns all library cells in library mylib:

  ```
  set libcells [find -libcells mylib/*]
  ```

- The following command returns all library cells with `dont_use` attribute:

  ```
  set libcells [find -libcells -filter {is_dont_use == 1}]
  ```

- The following command returns library cells:

  ```
  find -libcell -of_objects <list_of_libpin_objects>
  ```

7.10 LIBPIN

The following lists the attributes for the library pin design objects, in alphabetical order:

- active_phase on page 207
- bit_width on page 208
- bus_idx on page 208
- bus_name on page 208
- direction on page 209
- function on page 209
- is_clock_pin on page 209
- is_in_elaborated_tree on page 209
- is_pad on page 210
- libcell on page 210
- library on page 210
- lsb on page 211
- msb on page 211
- sdc_constraints on page 211
- test_cell_signal_type on page 212
- timing_arc on page 212
- type on page 213

7.10.1 active_phase

Read-only attribute. Specifies the active phase of the clock/enable/set/reset pin.

Type

string

Value

```
<posedge | negedge | bothedge | high | low
| null ("")>
```

7.10.2 bit_width

Read-only attribute. Specifies the bitwidth of the libpin.

Type

integer

Value

```
<bit_width>
```

7.10.3 bus_idx

Read-only attribute. Specifies the index number of the bus to which the pin belongs.

Type

integer

Value

```
<index_number>
```

7.10.4 bus_name

Read-only attribute. Specifies the name of the bus.

Type

string

Value

```
<name_of_the_array>
```

7.10.5 direction

Read-only attribute. Specifies the direction of the libpin.

Type

string

Value

```
<in | out | inout |internal >
```

7.10.6 function

Read-only attribute. Specifies the function of the libpin (input will have NULL string)

Type

string

Value

```
<function>
```

7.10.7 is_clock_pin

Read-only attribute. When set to 1, the pin has `clock: true` in techlib.

Type

boolean

Value

```
{<0 | 1>}
```

7.10.8 is_in_elaborated_tree

Read-only attribute. When set to 1, it means the library cell is instantiated in another module that is part of the currently elaborated design.

Type

boolean

Value

```
<0 | 1>
```

7.10.9 is_pad

Read-only attribute. When set to 1, the pin has `is_pad: true` in techlib.

Type

boolean

Value

```
{<0 | 1>}
```

7.10.10 libcell

Read-only attribute. Specifies the name of the libcell of the libpin.

Type

string

Value

```
<libcell_of_the_pin>
```

7.10.11 library

Read-only attribute. Specifies the name of the library that contains the libpin.

Type

string

Value

`<library_name>`

7.10.12 lsb

Read-only attribute. Specifies the least significant bit (LSB) pin.

Type

integer

Value

`<lsb>`

7.10.13 msb

Read-only attribute. Specifies the most significant bit (MSB) pin.

Type

integer

Value

`<msb>`

7.10.14 sdc_constraints

Read-only attribute. List of `SDCOBJ` attached to this libcell

Note: Constraints in design objects contain only the SDCs that refer to that design object in the SDC statement.

Type

list

Value

```
{<list_of_sdcobj_libpin_is_constrained>}
```

7.10.15 test_cell_signal_type

Read-only attribute. Specifies the possible values for this attribute are those defined in the Liberty language for `signal_type` inside the `test_cell` group:

`test_scan_in`, `test_scan_in_inverted`, `test_scan_out`, `test_scan_out_inverted`, `test_scan_enable`, `test_scan_enable_inverted`, `test_scan_clock`, `test_scan_clock_a`, `test_scan_clock_b`, `test_clock`

If a library cell pin does not have this information specified in the Liberty file, then `test_cell_signal_type` has the value `""`.

7.10.16 timing_arc

Read-only attribute. Specifies the timing arc type and its condition.

Type

list

Value

```
{{<related_pin> <when> <timing_sense> <timing_type>} ...}
```

Examples

- The following command returns pins of the library cell l1:

  ```
  set mylibpin [find -libpin l1/*]
  ```

- The following command returns pins of the clock pin of libcell l1:

  ```
  set clklibpin [find -libpin l1/* -filter {is_clock == 1}]
  ```

- The following command returns library pins:

  ```
  find -libpin -of_objects <libcell_object>
  ```

7.10.17 type

Read-only attribute. Specifies the type of the libpin.

Type

string

Value

```
<clock | set | reset | data | unknown>
```

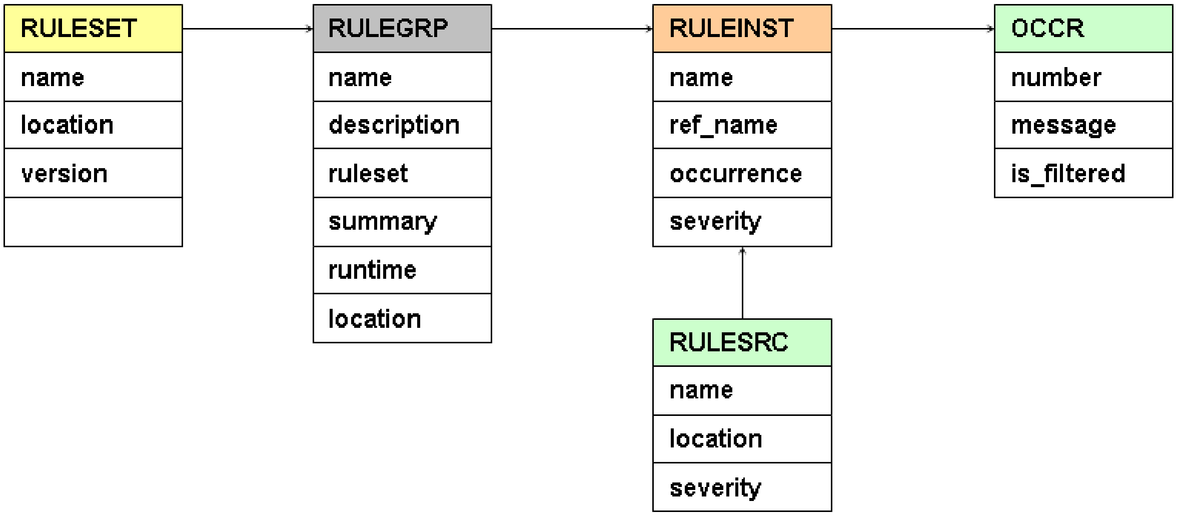
8

Rule Objects

Rule objects consist of following object types:

8.1 Rule Related Object Overview

The following figure illustrates how rule objects relate to each other.



Rule objects contain information and manipulate the following:

- Rule sets (RULESET)
- Rule groups (RULEGRP)
- Rule instances (RULEINST)
- Rule source (RULESRC)
- Rule occurrences (RULEOCCR)

8.2 Common Attributes for Rule Objects

The following lists the common attributes for rule object types—they do not apply to rule occurrences.

`desc`

Read-write attribute. Short description of this rule object. This can be modified with the `set attribute` command.

Type: string

Value: `<description>`

`full_desc`	**Read-write** attribute. Detailed description or short help for rule object. This can be modified with the `set attribute` command. Type: string Value: `<detailed_description_of_the_rule>`
`full_name`	Full name of the rule object Type: string Value: `<object_full_name>`
`location`	Location of the definition Type: string Value: `<full_path_to_file_name>`
`name`	Name of the rule object. Type: string Value: `<object_name>`
`object_type`	Type of this rule object Type: string Value: `<ruleset | rulegrp | ruleinst | rulesrc | rulefilter>`
`start_line`	Line number of the definition Type: integer Value: `<line_number>`
`version`	**Read-write** attribute. Version of this rule source Type: floating Value: `<version>`

8.3 RULEFILTER

The following lists the attributes for rule filters. See also “Common Attributes for Rule Objects” on page 216.

`applied_to`	List of rule objects to which this applies Type: list/object Value: `<list_of_rule_objects>`
`condition`	Condition of the filter Type: string Value: `<condition>`
`end_line`	End line number of the filter. Type: integer Values: `<line_no_end>`

8.4 RULEGRP

The following lists the attributes for rule groups. See also “Common Attributes for Rule Objects” on page 216.

`filters`	List of filters applied to occurrences of this rule group Type: list/object Value: `<list_of_filter_objects>`
`mem_usage`	Total memory usage of the rules in this rule set Type: list/list Value: `<memory_usage>`
`ruleinsts`	List of rule instances in this rule groups Type: list/object Value: `<list_of_ruleinsts>`

`ruleset`	List of rule sets for this rule group Type: object Value: `<ruleset_this_belongs_to>`
`runtime`	Total runtime of the rules in this rule set Type: floating Value: `<total_runtime>`
`status`	Status of this rule group Type: string Value: `<pass	fail>`

8.5 RULEINST

The following lists the attributes for rule instances. See also “Common Attributes for Rule Objects” on page 216.

`category`	Rule category Type: string Value:
`check_proc`	Name of the Tcl procedure executed when the `RUN RULE CHECK` command is executed. Type: string Value: `<name_of_check_proc>`

`continue_on_error`	**Read-write** attribute. Specifies whether the tool should continue after encountering occurrences of a rule instance. 1 to continue; otherwise, 0. For RTL rules, the default value is 0. For SDC lint rules, the default value is 1. Type: boolean Value: <0 | 1> Note: This attribute is settable for RTL rules and SDC lint rules, as long as their severity level is not Error.
`diagnose_proc`	Name of the Tcl procedure to diagnose occurrences Type: string Value: `<name_of_diagnose_proc>`
`filters`	List of filters applied to occurrences of this rule instance Type: list/object Value: `<list_of_filter_objects>`
`get_occr_msg_proc`	The Tcl procedure executed when asked for a message specific to an occurrence of this rule check. Type: string Value: `<name_of_message_proc>`
`help_file`	**Read-write** attribute. Path to the help file Type: string Value: `<help_file_name>`
`include_files`	List of all the files to load together with this rule instance Type: list Value: `<list_of_include_file>`
`is_run`	1 if rule instance has been run, if not 0 Type: boolean Value: `<0 | 1>`

is_virtual	**Read-write** attribute. Set to 1 if this rule instance is marked as virtual. Type: boolean Value: <0 | 1>
mem_usage	Total memory usage of this rule instance Type: list/list Value: <memory_usage>
occr_count	Number of occurrences associated with this instance per status Type:list/key_value Value: {<status_key><occurrence_count_per_status_key>}
occrs	List of occurrences of this rule instance Type: list/object Value: <list_of_occurrences>
occr_limit	Maximum number of occurrences that this rule instance can create Type: integer Value: 2147483647
open_hdl_src_proc	Name of the procedure for opening the HDL source Type: string Value: <name_of_hdl_src_proc>
open_schematics_proc	
	Name of the procedure for opening schematics Type: string Value: <name_of_schematics_proc>
open_sdc_src_proc	Name of the procedure to call when opening up the SDC source Type: string Value: <name_of_sdc_src_proc>

`options`	Lists the list of options passed to its rule source. Type: list/list Value: `<options_of_ruleinstance>`
`report_rule_proc`	Name of the procedure to call when the `REPORT RULE CHECK` command is executed. Type: string Value: `<name_of_report_rule_proc>`
`required_license`	License required to run this rule Type: string Value: `<L | XL | MCC | GXL>`
`required_state`	Required state to run this rule source Type: string Value: `<setup | verify | read_library | read_design | read_sdc | propagate_clock (verify) | commit_clock>`
`rulegrp`	Rule group to which this rule instance belongs Type: object Value: `<rulegrp_this_belongs_to>`
`ruleset`	Rule set to which this rule instance belongs Type: object Value: `<ruleset_this_belongs_to>`
`rulesrc`	Reference rule source for this rule instance Type: object Value: `<reference_rulesrc>`
`runtime`	Total runtime of this rule instance Type: floating Value: `<total_runtime>`
`rule_type`	Specifies the rule type Value: `{ standard | multimode | hierarchical | sdcintegration }`

severity	**Read-write** attribute. Severity of this rule instance Type: string Value: <error | warning | info | ignore | <userdefined>>
status	Status of this rule instance Type: string Value: <pass | fail | notrun>

8.6 RULESET

The following lists the attributes for rule sets. See also "Common Attributes for Rule Objects" on page 216.

filters	List of filters applied to occurrences of this rule set Type: list/object Value: <list_of_filter_objects>
mem_usage	Total memory usage of the rules in this rule set Type: list/list Value: <memory_usage>
rulegrps	List of rule groups in this rule set Type: list/object Value: <list_of_rulegrps_in_ruleset>
runtime	Total runtime of the rules in this rule set Type: floating Value: <total_runtime>
status	Status of this rule set Type: string Value: <pass | fail>

8.7 RULESRC

The following lists the attributes for rule source.

See also “Common Attributes for Rule Objects” on page 216.

`category`	Rule category Type: string
`check_proc`	Name of the Tcl procedure to check rule Type: string Value: `<name_of_check_proc>`
`diagnose_proc`	Name of the Tcl procedure to diagnose occurrences Type: string Value: `<name_of_diagnose_proc>`
`get_occr_msg_proc`	Name of the Tcl procedure to create a message for an occurrence of this rule Type: string Value: `<name_of_message_proc>`
`help_file`	Help file for this rule Type: string Value: `<name_of_help_file>`
`include_files`	**Read-write** attribute. List of required Tcl files Type: list Value: `<list_of_include_file>`
`open_hdl_src_proc`	Type: string Value: `<name_of_hdl_src_proc>`
`open_schematics_proc`	

	Name of the Tcl procedure to open schematics for occurrences Type: string Value: `<name_of_schematics_proc>`
`open_sdc_src_proc`	Type: string Value: `<name_of_sdc_src_proc>`
`options`	Lists the options for this rule source. Type: list/list Value: `<options_of_rulesrc>`
`required_license`	License required to run this rule Type: string Value: `<L | XL | MCC | GXL>`

`required_state`	Required state to run this rule source
	Type: string
	Value: `<setup | verify | read_library`
	`| read_design | read_sdc`
	`| propagate_clock (verify) | commit_clock>`

`ruleinsts`	List of instances associated with this rule source Type: list/object Value: `<list_of_ruleinsts_referred>`
`rule_type`	Specifies the rule type Value: `{ standard | multimode | hierarchical | sdcintegration }`
`severity`	**Read-write** attribute. Severity of this rule Type: string Value: `<error | warning | info | ignore` `| <userdefined>>`

8.8 OCCR

The following lists the attributes for occurrences. See also "Common Attributes for Rule Objects" on page 216.

Attribute	Description
`atomic_checks`	Type: list/list
`design_type`	SDC design in which this occurrence belongs Type: string Value: `<golden | revised>`
`dsgnobjs`	Type: list/object
`filtered_by`	List of filters that makes this occurrence to be filtered Type: object Value: `<filter_object>`
`full_name`	Number ID of this occurrence Type: string Value: `<full_name_of_occur>`
`is_filtered`	**Read-write** attribute. Set to 1 if this occurrence is filtered. Type: boolean Value: `<0 | 1>`
`message`	Message for this occurrence Type: string
`name`	Return value occr Type: integer Value: `<occur_number>`
`occr_type`	Occurrence type Type: string Value: `<Library | Design>`

Attribute	Description
rulegrp	Rule group to which this occurrence belongs. Type: object Value:
ruleinst	Rule instance to which this occurrence belongs. Type: object Value:
ruleset	Rule set to which this occurrence belongs. Type: object Value:
rulesrc	Rule source to which this occurrence belongs. Type: object Value:
sdcdsgn	Type: object
sdcmode	SDC mode in which this occurrence belongs Type: object
sdcobjs	List of relevant sdcobj objects to this occurrence Type: list/object
sdcstmts	List of relevant SDC statements to this occurrence Type: list/object
status	Type: string Value: `<pass | fail>`
witness	Type: list/list

A

FIFO-Related Conformal Object Attributes

The following table lists all the FIFO-related Conformal object attributes and their corresponding FIFO object attributes. These FIFO-related Conformal object attributes are used as default configuration for all FIFOs. You, however, also use FIFO object attributes to configure checks for a particular FIFO. Please refer to Chapter 5, “FIFO Objects” for more details about each attribute.

Conformal object attributes	FIFO object attributes
`fifo_check_mem_size`	`check_mem_size`
`fifo_check_out_size`	`check_out_size`
`fifo_check_mem_out_size`	`check_mem_out_size`
`fifo_check_async_mem`	`check_async_mem`
`fifo_check_mem_out_exclusive`	`check_mem_out_exclusive`
`fifo_check_mem_supported_cell_type`	`check_mem_supported_cell_type`
`fifo_check_readptr_size`	`check_readptr_size`
`fifo_check_readptr_sync`	`check_readptr_sync`
`fifo_check_writeptr_size`	`check_writeptr_size`
`fifo_check_writeptr_sync`	`check_writeptr_sync`
`fifo_check_gray_comb_loop`	`check_gray_comb_loop`
`fifo_check_single_sync`	`check_single_synch`
`fifo_check_wdata_size`	`check_wdata_size`
`fifo_check_gray_size`	`check_gray_size`
`fifo_check_synch_size`	`check_sync_size`
`fifo_check_gray_func`	`check_gray_func`

Conformal object attributes	**FIFO object attributes**
`fifo_check_single_rptr`	`check_single_rptr`
`fifo_check_single_wptr`	`check_single_wptr`
`fifo_check_single_rgray`	`check_single_rgray`
`fifo_check_single_wgray`	`check_single_wgray`
`fifo_skip_two_dimensional_check`	`check_mem_two_dimension`