Contents

Man Page Viewing and Printing Instructions ........ 1
User Commands ........................................ 3
Synthesis Commands ................................. 7
Synthesis Variables ................................. 154
Man Page Viewing and Printing Instructions

The following sections describe how to set up your UNIX environment so you can view and print man pages.

Setting Up the UNIX Environment

Edit your .cshrc file to contain these lines:

```
setenv SYN_MAN_DIR synopsys_root/doc/syn/man
setenv MANPATH ${MANPATH}:${SYN_MAN_DIR}
```

SYN_MAN_DIR is a variable that contains the path to the man page directories, and synopsys_root represents the specific path to the Synopsys synthesis software directory at your site.

Viewing Man Pages From UNIX

Command

```
% man command_name
```

Variable

```
% man variable_name
```

Variable group

```
% man variable_group_name
```

All attributes

```
% man attributes
```

Attribute group

```
% man attribute_group_name
```

Error, warning, or information message

```
% aman message_id
```

Viewing Man Pages From dc_shell

Command

```
dc_shell> man command_name
```

Variable

```
dc_shell> man variable_name
```
Variable group
\texttt{dc\_shell}\textgreater \texttt{man variable\_group\_name}

All attributes
\texttt{dc\_shell}\textgreater \texttt{man attributes}

Attribute group
\texttt{dc\_shell}\textgreater \texttt{man attribute\_group\_name}

Error, warning, or information message
\texttt{dc\_shell}\textgreater \texttt{man message\_id}

\section*{Printing Man Pages From UNIX}

User command
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt1/command\_name.1}

Synopsys command
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt2/command\_name.2}

Variable
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt3/variable\_name.3}

Variable group
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt3/variable\_group\_name.3}

All attributes
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt3/attributes.3}

Attribute group
\texttt{\% lpr -t -P printer\_name \textbackslash\textbackslash
\$SYN\_MAN\_DIR/fmt3/attribute\_group\_name.3}

You cannot print error, warning, or information message man pages from UNIX.

\section*{Printing Man Pages From dc\_shell}

You cannot print man pages from dc\_shell.
User Commands

Invoke user commands from a UNIX shell.

bc_view
Runs the BCView performance analysis tool.

bc_view
  [-f project_file]
  [-license_preference package | individual]

aman
Displays Synopsys extended error messages.

aman [error_message_code]

cache_ls
Lists elements in a Synopsys cache.

cache_ls
  cache_dir
  reg_expr

cache_rm
Removes elements from a Synopsys cache.

cache_rm
  cache_dir
  reg_expr

create_types
Extracts user-defined type information from VHDL package files.

create_types
  [-nc]
  [-w lib]
  [-v]
  [-o logfile]
  file_list
dc_shell-t
Invokes the Design Compiler shell in dctcl mode. For more information, see the man page for dc_shell.

```
dc_shell-t
[-f script_file]
[-x command_string]
[-no_init]
[-checkout feature_list]
[-wait wait_time]
[-timeout timeout_value]
[-version]
[-behavioral]
[-fpga]
[-syntax_check | -context_check]
```

dc_shell
Invokes the Design Compiler command shell.

```
dc_shell
[-f script_file]
[-x command_string]
[-no_init]
[-checkout feature_list]
[-tcl_mode]
[-wait wait_time]
[-timeout timeout_value]
[-version]
[-behavioral]
[-fpga]
[-syntax_check | -context_check]
```

design_analyzer
Runs the Design Analyzer menu interface in the X Window System.

```
design_analyzer
[-menu_script]
[-no_menu_script]
[-f script_file]
[-x command_string]
[-no_init]
[-checkout feature_list]
[-timeout timeout_value]
[-version]
[-behavioral]
[-fpga]
[-syntax_check | -context_check]
```
design_vision

Runs Design Vision visualization for Synopsys synthesis products.

design_vision
[-f script_file]
[-x command_string]
[-no_init]
[-checkout feature_list]
[-timeout timeout_value]
[-version]
[-behavioral]
[-syntax_check | -context_check]
[-tcl_mode]

lc_shell

Runs the Library Compiler command shell.

lc_shell
[-f script_file]
[-x command_string]
[-no_init]
[-version]

library_compiler

Runs the Library Compiler graphical interface in the X window system.

library_compiler
[-f script_file]
[-x command_string]
[-no_init]
[-version]

ra_shell

Runs the RTL Analyzer command shell.

ra_shell
[-f script_file]
[-x command_string]
[-no_init]
[-checkout feature_list]
[-timeout timeout_value]
[-version]
synenc

Runs the Synopsys Encryptor for HDL source code.

    synenc
    [-r synopsys_root]
    file_list

synopsys_users

Lists the current users of the Synopsys licensed features.

    synopsys_users [feature_list]
Synthesis Commands

This section contains the following subsections:

- Command Syntax
- Commands Specific to dcsh Mode
- Commands Specific to dctcl Mode

Command Syntax

Invoke these commands from within a synthesis tool. Unless otherwise noted, all commands are available in both dcsh and dctcl mode.

acs_check_directories (dctcl-mode only)
Checks Automated Chip Synthesis (ACS) directory structure settings for correctness. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int acs_check_directories
```

acs_compile_design (dctcl-mode only)
Compiles the constrained RTL to a netlist using constraints propagated from the top-level design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int acs_compile_design
[-destination pass_name]
[-prepare_only]
[-force]
[-update]
[-update_source source_pass]
design_name
```

acs_create_directories (dctcl-mode only)
Creates the project directory tree for Automated Chip Synthesis. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int acs_create_directories
```


acs_get_parent_partition (dctcl-mode only)

Creates a collection of designs that are compile partitions and that contain the specified subdesign. For use in dc_shell-t (Tcl mode of dc_shell) only.

string acs_get_parent_partition
    design_name
    [-hierarchy]
    [-list]

acs_get_path (dctcl-mode only)

Gets the path location for the specified file. To specify a file, specify its file type and, for pass-dependent files, its pass directory. For use in dc_shell-t (Tcl mode of dc_shell) only.

string acs_get_path
    -file_type filetype
    [-mode read | write]
    [-pass pass_name]
    [\[-name filename\]
    [-append]
    [-relative]

acs_merge_design (dctcl-mode only)

Preprocesses a design for incremental design update by merging the modified designs and their parent compile partitions with the mapped design being updated. For use in dc_shell-t (Tcl mode of dc_shell) only.

int acs_merge_design
    -update design_list
    [-unmapped source_dir]
    [-mapped data_dir]
    [-type pre | post]
    [-destination dest_dir]
    design_name
acs_read_hdl (dctcl-mode only)

Reads in the HDL source code of a design and generates the GTECH representation in memory. For use in dc_shell-t (Tcl mode of dc_shell) only.

```c
int acs_read_hdl
    [design_name]
    [-hdl_source file_or_dir_list]
    [-exclude_list file_or_dir_list]
    [-format {verilog | vhdl}]
    [-recurse]
    [-no_dependency_check]
    [-no_elaborate]
    [-ignore_analyze_errors]
    [-library design_lib_name]
    [-verbose]
```

acs_recompile_design (dctcl-mode only)

Compiles an unmapped constrained .db file using time budgets. The time budgets are created by using a previously-mapped design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```c
int acs_recompile_design
    [-budget_source budget_pass]
    [-destination destination_pass]
    [-source source_pass]
    [-prepare_only]
    [-force]
    [-update]
    [-update_source source_pass]
    design_name
```

acs_refine_design (dctcl-mode only)

Refines an already mapped design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```c
int acs_refine_design
    [-source pass_name]
    [-destination pass_name]
    [-prepare_only]
    [-force]
    [-update]
    [-update_source source_pass]
    design_name
```
**acs_report_directories** (dctcl-mode only)

Reports the current directory structure settings. For use in dc_shell-t (Tcl mode of dc_shell) only.

```tcl
int acs_report_directories
[-file_types type_list]
```

**add_module**

Reads in a specified library file containing module functional information and uses it to update an existing technology library.

```tcl
int add_module
[-overwrite]
[-permanent]
file_name
library_name
[-no_warnings]
```

**add_to_collection** (dctcl-mode only)

Adds objects to a collection, resulting in a new collection. The base collection remains unchanged.

```tcl
collection add_to_collection
base_collection
object_spec
[-unique]
```

**after** (dctcl-mode only)

Built-in Tcl command.

**alias**

Defines an alias for a command, or lists current alias definitions.

```tcl
int alias [identifier [expansion]]
```

**all_clocks**

Returns a list of all clocks in the current design. In dctcl mode, a collection is returned.

```tcl
list all_clocks()
```
**all_cluster_cells**
Returns a list of cells contained in the specified cluster. In dctcl mode, a collection is returned.

```python
list all_cluster_cells
[-hierarchy] cluster_name
```

**all_clusters**
Returns a list of subclusters associated with a specified cluster or with the current design. In dctcl mode, a collection is returned.

```python
list all_clusters
[-cluster cluster_name]
[-leaf]
```

**all_connected**
Returns the objects connected to a net, port, pin, or a net or pin instance.

```python
list all_connected object
```

**all_critical_cells**
Returns a list of critical leaf cells in the top hierarchy of the current design. In dctcl mode, a collection is returned.

```python
list all_critical_cells
[-slack_range range_value]
```

**all_critical_pins**
Returns a list of critical endpoints or startpoints in the current design. In dctcl mode, a collection is returned.

```python
list all_critical_pins
[-type endpoint | startpoint]
[-slack_range range_value]
```

**all_designs**
Returns a list of all designs in the current design. In dctcl mode, a collection is returned.

```python
list all_designs
```
all_fanin
Reports pins, ports, or cells in the fanin of specified sinks.

list all_fanin
 -to sink_list
 [-startpoints_only]
 [-exclude_bboxes]
 [-break_on_bboxes]
 [-only_cells]
 [-flat]
 [-levels count]

all_fanout
Returns a set of pins, ports, or cells in the fanout of the specified sources.

list all_fanout
 -clock_tree
 -from source_list
 [-endpoints_only]
 [-exclude_bboxes]
 [-break_on_bboxes]
 [-only_cells]
 [-flat]
 [-levels count]

all_inputs
Returns a list of input or inout ports in the current design. In dctcl mode, a collection is returned.

list all_inputs
 [-clock clock_name]
 [-edge_triggered | -level_sensitive]

all_outputs
Returns a list of output or inout ports in the current design. In dctcl mode, a collection is returned.

list all_outputs
 [-clock clock_name]
 [-edge_triggered | -level_sensitive]
**all_registers**

Returns a list of sequential cells or pins in the current design. In dcltcl mode, a collection is returned.

```bash
list all_registers
[-no_hierarchy]
[-clock clock_name]
[-cells]
[-data_pins]
[-clock_pins]
[-slave_clock_pins]
[-inverted_output]
[-output_pins]
[-level_sensitive | -edge_triggered]
[-master_slave]
```

**allocate_budgets (dcsh-mode only)**

Allocates budgets to the specified cells.

```bash
string allocate_budgets
[ -incremental]
[ -check_only]
[ -create_context]
[ -no_boundary_annotations]
[ -effort allocation_effort]
[ -min_register_to_output minimum_budget]
[ -min_input_to_output minimum_budget]
[ -min_input_to_register minimum_budget]
[ -no_environment]
[ -interblock_logic]
[ -file_format_spec format_spec]
[ -format script_format]
[ -separator name_separator]
[ -levels budget_levels]
[ -write_context]
[ -budget_design_ware]
[ cell_list]
```
allocate_partition_budgets (dctcl-mode only)

Creates budgets for the compile partitions in the specified design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```c
int allocate_partition_budgets
    -source src_pass
    -destination dst_pass
    [-absolute_paths]
    [-budget_shell budget_shell_exec]
    [-transcript_exec transcript_exec]
    [-format dcsh | dctcl]
    [-overconstrain overcstr]
    [-effort allocation_effort]
    [-interblock_logic]
    [-min_register_to_output minimum_budget]
    [-min_input_to_output minimum_budget]
    [-min_input_to_register minimum_budget]
    [-no_environment]
    [-generate_script_only]
    [-remove]
    design
```

analyze

Analyzes HDL files and stores the intermediate format for the HDL description in the specified library.

```c
int analyze
    [-library library_name]
    [-work library_name]
    [-format vhdl | verilog]
    [-create_update]
    [-update]
    [-define define_list]
    [schedule]
    file_list
```
**annotate_activity**

Sets (or resets) the toggle_rate and static_probability values for specified objects in the current design.

```c
int annotate_activity
    [-static_probability sp_value]
    [-toggle_rate tr_value]
    [-clock clock_port_name]
    [-period period_value]
    [-hier]
    [-reset]
    [-select ports|regs|all]
    [-objects object_list]
    [-instances instance_list]
```

**append** (dctcl-mode only)

Built-in Tcl command.

**apropos** (dctcl-mode only)

Search the command database for a pattern.

```c
string apropos
    [-symbols_only]
    pattern
```

**array** (dctcl-mode only)

Built-in Tcl command.

**attach_region**

Attaches cells to an existing region.

```c
int attach_region
    [-name region_name]
    cell_list
```

**auto_execok** (dctcl-mode only)

Built-in Tcl command.

**auto_import** (dctcl-mode only)

Built-in Tcl command.

**auto_load** (dctcl-mode only)

Built-in Tcl command.

**auto_load_index** (dctcl-mode only)

Built-in Tcl command.
**auto_qualify** (dctcl-mode only)

Built-in Tcl command.

**balance_buffer**

Builds a balanced buffer tree on user-specified nets and drivers.

```tcl
int balance_buffer
[-verify]
[-verify_hierarchically]
[-verify_effort low | medium | high]
[-from start_point_list]
[-to end_point_list]
[-net net_list]
[-force]
[-library library_name]
[-prefer buffer | inverter | lib_cell_name]
```

**balance_registers**

Moves the registers of a design to achieve a minimum cycle time.

```tcl
int balance_registers [design_name]
```

**bc_check_design**

Performs a quick check of Behavioral Compiler scheduling information on the current design and reports incorrect coding styles.

```tcl
int bc_check_design
[-io_mode cycle_fixed | superstate_fixed]
[-constraints]
```

**bc_dont_register_input_port**

For Behavioral Compiler, disables automatic register allocation for the specified input port.

```tcl
int bc_dont_register_input_port port_name
```
**bc_dont_ungroup**
Prevents the compile command from ungrouping cells grouped by Behavioral Compiler, for the specified group classes or for all grouped cells in the current design.

```plaintext
int bc_dont_ungroup [-register] [-fsm] [-random_logic] [-prio_logic] [-array_logic]
```

**bc_group_process**
For Behavioral Compiler, creates one level of hierarchy instead of flattening each process after scheduling.

```plaintext
int bc_group_process [-with_memory]
```

**bc_margin**
Sets the timing margin used by Behavioral Compiler.

```plaintext
```

**bc_report_arrays**
Reports conflicting and non-conflicting accesses to arrays mapped to register files within an elaborated behavioral design.

```plaintext
int bc_report_arrays [-conflicting] [-non_conflicting]
```
bc_report_memories
Reports specific information about the memories instantiated within an elaborated behavioral design and in the available synthetic libraries.

```plaintext
int bc_report_memories
[-synthetic_libraries]
[-bindings]
[-used_memories]
[-conflicting]
[-non_conflicting]
```

bc_time_design
Calculates timing and area estimates and annotates them on the current design for Behavioral Compiler and SystemC Compiler.

```plaintext
int bc_time_design
[-force]
[-fastest]
[-cache_preserved_functions library_name
[-except designs]]
[-use_cached_preserved_functions
library_name [-recompile designs]]
```

bc_view
Invokes BCView on the current design.

```plaintext
int bc_view
[-output out_db_file]
[-project_file project_file_name]
[-dont_start]
[-cossap]
[-search_additional path_list]
[-host machine_name]
[-arch architecture_name]
```

binary (dctcl-mode only)
Built-in Tcl command.

break
Immediately exits a loop structure.

```plaintext
int break
```
**calculate_rtl_load**
Calculates RTL load values using layout-based annotation.

```
int calculate_rtl_load
-capacitance
-delay pin_net_list
```

**catch** (dctcl-mode only)
Built-in Tcl command.

**cd**
Changes the current directory.

```
int cd [directory]
```

**cell_of**
Returns the cell objects for given pins in the current design.

```
list cell_of [object_list]
```

**chain_operations**
Specifies a list of operations to be scheduled by Behavioral Compiler for a specified process or for all processes.

```
int chain_operations
[-process process_name] operation_names
```

**change_link**
Changes the design to which a cell is linked.

```
int change_link
object_list
design_name
```

**change_names**
Changes the names of ports, cells, and nets in a design.

```
int change_names
[-rules name_rules]
[-hierarchy]
[-verbose]
[-names_file names_file]
[-log_changes log_file_name]```
characterize
Captures information about the environment of specific cell instances, and assigns the information as attributes on the design to which the cells are linked.

```
int characterize
cell_list
[-no_timing]
[-constraints]
[-connections]
[-power]
[-verbose]
```

check_bindings
Checks the bindings in a synthetic library module definition.

```
int check_bindings
[-bindings binding_list]
[-pin_widths pin_width_list]
module_name
```

check_bsd
Checks whether a design's boundary-scan implementation is compliant with IEEE Std 1149.1.

```
int check_bsd
[-verbose]
[-effort low | medium | high]
```

check_budget
Checks that user-specified budgets and fixed delays are consistent with path constraints.

```
string check_budget
[-verbose]
[-tolerance tolerance]
[-from object_list]
[-to object_list]
[-no_environment]
[-interblock_logic]
[cell_list]
```
check_design

Checks the current design for consistency.

```c
int check_design
[-summary]
[-no_warnings]
[-one_level]
[-post_layout | -only_post_layout]
```

cHECK DFT

Checks a design against the design rules of a scan test methodology with test points that have been inserted by either Autofix or Shadow LogicDFT.

```c
int check_dft
[-verbose]
[-check_contention true | false | scan_shift_only | capture_only]
[-check_float true | false | scan_shift_only | capture_only]
```

cHECK ERROR

Prints extended information on errors from last command.

```c
int check_error
[-verbose]
[-reset]
```

cHECKIMPLEMENTATIONS

Checks the implementations in a synthetic library module definition.

```c
int check_implementations
[-implementations implementation_list]
[-parameters parameter_list]
module_name
```

cHECK SCAN

Checks a design against the design rules of a scan test methodology.

```c
int check_scan
[-verbose]
[-check_contention true | false]
[-check_float true | false]
```
**check_synlib**
Performs semantic checks on synthetic libraries.

```bash
int check_synlib
```

**check_test**
Checks a design against the design rules of a scan test methodology.

```bash
int check_test
[-verbose]
[-check_contention true | false]
[-check_float true | false]
[-scan_shift_only | capture_only]
```

**check_timing**
 Warns about possible timing problems in the current design.

```bash
int check_timing
[-overlap_tolerance minimum_distance]
```

**check_unmapped** (dcsh-mode only)
Checks for any unmapped design below the current design.

```bash
int check_unmapped
```

**clean_buffer_tree**
Removes the buffer tree at a given driver pin on a mapped design.

```bash
int clean_buffer_tree
[-from start_point_list]
[-to end_point_list]
[-net net_list]
[-hierarchy]
```

**clock** (dctcl-mode only)
Built-in Tcl command.

**close** (dctcl-mode only)
Built-in Tcl command.
**compare_collections** (dctcl-mode only)

Compares the contents of two collections. If the same objects are in both collections, the result is 0 (like string compare). If they are different, the result is nonzero. The order of the objects can optionally be considered.

```c
int compare_collections
    [-order_dependent]
    collection1
collection2
```

**compare_design**

Compares two designs for functional equivalence.

```c
int compare_design
    [-effort low | medium | high]
    [-jtag]
    [-fsm]
    [-verbose]
    [-hierarchical]
design1 design2
```

**compare_fsm**

Compares the sequential behavior of two finite state machine designs.

```c
int compare_fsm
design1
design2
```

**compare_lib**

Performs a cross-reference check between a technology library and a symbol library.

```c
int compare_lib
library1
library2
```
compile

Performs logic-level and gate-level synthesis and optimization on the current design. Performs logic-level and gate-level synthesis and optimization on the current design.

```c
int compile
[-no_map]
[-map_effort low | medium | high]
[-area_effort none | low | medium | high]
[-incremental_mapping]
[-exact_map]
[-verify]
[-verify_hierarchically]
[-verify_effort low | medium | high]
[-ungroup_all]
[-boundary_optimization]
[-auto_ungroup area | delay]
[-no_design_rule | -only_design_rule | -only_hold_time]
[-scan]
[-background run_name]
[-host machine_name]
[-arch architecture]
[-xterm]
[-top]
```

compile_partitions (dctcl-mode only)

Distributes compile jobs for a design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```c
compile_partitions
-destination pass
```
**compile_preserved_functions**

Compiles and/or writes netlists for preserved functions.

```c
int compile_preserved_functions
    [preserved_functions]
    [-exclude preserved_functions]
    [-no_compile]
    [-compile_effort low | medium | high | 1 | 2 | 3]
    [-include_script script_name]
    [-exclude]
    [-filename filename]
    [-design_library library_name]
    [-force_recompile]
    [-stages number_of_stages]
    [-clock_port_name clock_port_name]
    [-output_delay delay_value]
    [-input_delay delay_value]
    [-sync_reset reset_port_name]
    [-async_reset reset_port_name]
    [-reset_polarity high | low]
```

**compile_systemc**

Reads a SystemC source file, checks for compliance with SystemC Compiler synthesis policy and syntax, and creates an internal database (.db) if there are no errors.

```c
int compile_systemc
    [-cpp cpp_program]
    [-cpp_options options]
    [-verbose]
    [-preserve functions]
    [-unroll loop_labels]
    [-trace]
    [-hls]
    [-rtl]
    [-rtl_format db | verilog]
    [-output output_file]
    [-single_file]
    [-w/-library directory_name]
    [-param module_spec_list]
    [-dont_rename module_list]
    file_name
```
**compileUltra**

Performs a two-pass, high-effort compile flow on the current design for better QOR results.

```bash
int compileUltra
[-scan]
[-no_uniquify]
[-no_autoungroup]
```

**concat (dctcl-mode only)**

Built-in Tcl command.

**connect_net**

Connects a specified net to specified pins or ports.

```bash
int connect_net
net
object_list
```

**context_check**

Enables or disables the Syntax Checker context_check mode in which commands issued are checked for context errors.

```bash
int context_check true | false
```

**continue**

Begins the next loop iteration.

```bash
int continue
```

**copy_collection (dctcl-mode only)**

Duplicates the contents of a collection, resulting in a new collection. The base collection remains unchanged.

```bash
collection copy_collection collection1
```

**copy_design**

Copies a design to a new design, or copies a list of designs to a new file in dc_shell memory.

```bash
int copy_design
source_design_name
target_design_name
```
create_bsd_patterns
Generates a set of functional patterns for a boundary-scan design.

int create_bsd_patterns
[-output test_program_name]
[-effort low | medium | high]
[-type vector_type_list]

create_bounds
Creates a fixed movebound or floating group bound in the design.

int create_bounds
[-name bound_name]
[-coordinate {llx1 lly1 urx1 ury1 ...}]
[-dimension {width height}]
[-effort low | medium | high | ultra]
[-type soft | hard]
[-cell_list]

create_bus
Creates a port bus or a net bus.

int create_bus
object_list
bus_name
[-type type_name]
[-sort]
[-no_sort]
[-start start_bit]
[-end end_bit]

create_cache
Populates the cache directories with instances of the requested synthetic modules.

int create_cache
-module module_list
[-implementation implementation_list]
[-parameters parameter_list]
[-operating_condition operating_condition]
[-wire_load list]
[-report]
create_cell

Creates cells in the current design.

```
int create_cell
    cell_list
    [reference_name]
    [-logic logic_value]
    [-only_physical]
```

create_clock

Creates a clock object and defines its waveform in the current design.

```
int create_clock
    [port_pin_list]
    [-name clock_name]
    [-period period_value]
    [-waveform edge_list]
```

create_cluster

Creates a cluster in the physical hierarchy of the design.

```
int create_cluster
    [-name cluster_name]
    [-keep]
    [-multibits]
    [-parent parent_cluster_object]
    object_list
```

create_command_group (dctcl-mode only)

Creates a new command group.

```
string create_command_group group_name
```

create_design

Creates a design in dc_shell memory.

```
int create_design
    design_name
    [file_name]
```
create_generated_clock

Creates a generated clock object.

string create_generated_clock
[-name clock_name]
[-source master_pin]
[-divide_by divide_factor | -multiply_by multiply_factor]
[-duty_cycle percent]
[-invert]
[-edges edge_list]
[-edge_shift edge_shift_list]
port_pin_list

create_multibit

Creates a multibit component for the specified list of cells in the current design.

int create_multibit
object_list
[-name multibit_name]
[-sort]
[-no_sort]

create_net

Creates nets in the current design.

int create_net
[-instance instance]
net_list

create_operating_conditions

Creates a new set of operating conditions in a library.

int create_operating_conditions
-name name
-library library_name
-process process_value
-temperature temperature_value
-voltage voltage_value
[-tree_type tree_type]
[-calc_mode calc_mode]
[-rail_voltages rail_value_pairs]

create_pass_directories (dctcl-mode only)

Creates the directory structure required for storing ACS data. For use in Tcl mode of dc_shell only.

int create_pass_directories pass_list
create_port
  Creates ports in the current design.

  int create_port
  port_list
  [-direction dir]

create_schematic
  Generates a schematic for the current design.

  int create_schematic
  [-hierarchy]
  [-size sheet_size]
  [-portrait]
  [-fill_percent fill_value]
  [-outputs_attract]
  [-order_outputs output_port_list]
  [-sort_outputs]
  [-dont_left_justify_inputs]
  [-schematic_view]
  [-symbol_view]
  [-hier_view]
  [-no_bus]
  [-bit_mappers]
  [-implicit]
  [-no_rippers]
  [-sge]
  [-no_type_mappers]
  [-reference]
  [-gen_database]

create_test_clock
  Defines the timing of a clock applied to a design during manufacturing test.

  int create_test_clock port_list
  -waveform two_value_rise_fall_edge_list
  [-period period_value]
  [-internal_clocks true | false | default]

create_test_patterns
  This command is obsolete. Use the command estimate_test_coverage for test coverage estimation and see the man page for syntax. To generate test vectors, use TetraMAX ATPG.
create_test_schedule

Define a series of test modes for the purpose of core test scheduling.

int create_test_schedule

create_wire_load

 Creates wire load models for the current design.

int create_wire_load
[-design design_name]
[-cell cell_list]
[-cluster cluster_name]
[-hierarchy]
[-this_level_nets_only]
[-mode top | enclosed]
[-name model_name]
[-output file_name]
[-update_lib library_name]
[-write_script script_file_name]
[-dont_smooth]
[-trim trim_value]
[-percentile percentile_value]
[-total_area area]
[-statistics]

current_design

Sets the working design in dc_shell.

string current_design [design]

current_design_name (dctcl-mode only)

Built-in Tcl command.

current_instance

Sets the working instance object in dc_shell and enables other commands to be used on a specific cell in the design hierarchy.

string current_instance [instance]

current_test_mode

Sets or gets the working test mode for the current design.

int current_test_mode [test_mode_label]
**date (dctcl-mode only)**
Returns a string containing the current date and time.

```
string date
```

**dbatt (dctcl-mode only)**
Built-in Tcl command.

**dbatt_db (dctcl-mode only)**
Built-in Tcl command.

**dc_allocate_budgets**
Allocates budgets to specified cells.

```
string dc_allocate_budgets
[cell_list]
[-write_script]
[-file_format_spec format_spec]
[-format dcsh | dctcl]
[-separator hierarchy_separator]
[-mode rtl | gate | mixed]
[-positive_slack_only]
[-budget_design_ware]
[-levels budget_levels]
[-no_interblock_logic]
```

**define_design_lib**
Maps a design library to a UNIX directory.

```
int define_design_lib
library_name
-path directory
```
**define_name_rules**

Defines a set of name rules for designs.

```c
int define_name_rules
    name_rules
    [-max_length length]
    [-target_bus_naming_style bus_naming_style]
    [-allowed allowed_chars]
    [-restricted restricted_chars]
    [-first_restricted first_chars]
    [-last_restricted last_chars]
    [-reserved_words reserves]
    [-replacement_char char]
    [-remove_chars]
    [-equal_ports_nets]
    [-inout_ports_equal_nets]
    [-collapse_name_space]
    [-caseInsensitive]
    [-special output_format]
    [-prefix prefix_name]
    [-map map_string]
    [-type object_type]
    [-reset]
```

**define_proc_attributes** *(dctcl-mode only)*

Defines attributes of a Tcl procedure, including an information string for help, a command group, a set of argument descriptions for help, and so on. The command returns the empty string.

```c
string define_proc_attributes
    proc_name
    [-info info_text]
    [-define_args arg_defs]
    [-command_group group_name]
    [-hide_body]
    [-hidden]
    [-dont_abbrev]
    [-permanent]
```

**define_test_mode**

Define a test mode for the purpose of model inference.

```c
int define_test_mode
    test_mode_label
    [-type mode_purpose]
    string test_mode_label
    string mode_purpose
```
delete_test (dctcl-mode only)
   Built-in Tcl command.

derive_clocks
   Creates clock objects for network source pins or ports in the current design.
   int derive_clocks

derive_constraints
   Propagates design environment, constraints, and attribute settings from the top-level design to the specified subdesigns.
   int derive_constraints
      [-attributes_only]
      [-verbose]
      [-budget]
      cell_list

derive_timing_constraints
   Derives timing requirements and places that constraint information on the current design.
   int derive_timing_constraints
      [-fix_hold]
      [-min_delay]
      [-no_max_delay]
      [-no_max_period]
      [-max_delay_scale max_delay_scale_factor]
      [-min_delay_scale min_delay_scale_factor]
      [-period_scale max_period_scale_factor]
      [-separate_rise_and_fall]

detach_region
   Detaches cells from an existing region.
   int detach_region
      [-name region_name]
      cell_list

disconnect_net
   Disconnects a net from pins or ports.
   int disconnect_net net object_list -all
**dont_chain_operations**
Constrains what operations are allowed to be chained by Behavioral Compiler for a specified process or for all processes.

```
int dont_chain_operations
[-process process_name]
[-into]
[-from] operation_names
```

**drive_of**
Returns the drive resistance value of the specified library cell pin.

```
float drive_of
library_cell_pin
[-rise | -fall]
[-wire_drive]
[-piece best | worst | average | int_value]
```

**echo**
Displays literal strings and values of variables and expressions in dc_shell or in dc_shell-t.

**dc_shell:**
```
int echo
[-n]
[text_string]
[variable]
[(expression)]
```

**dc_shell-t:**
```
int echo
[-n]
[text_string]
[$variable]
[[expr expression]]
```
elaborate
Builds a design from the intermediate format of a Verilog module, a VHDL entity and architecture, or a VHDL configuration.

```plaintext
int elaborate
design_name
[-library library_name | -work library_name]
[-architecture arch_name]
[-parameters param_list]
[-file_parameters file_list]
[-update]
[-schedule]
[-gate_clock]
```

encoding (dctcl-mode only)
Built-in Tcl command.

encrypt_lib
Encrypts a VHDL source library file.

```plaintext
int encrypt_lib
file_name
[-output encrypted_file]
```

eof (dctcl-mode only)
Built-in Tcl command.

error (dctcl-mode only)
Built-in Tcl command.

error_info (dctcl-mode only)
Prints extended information on errors from last command.

```plaintext
string error_info
```

estimate_test_coverage
Generates test coverage statistics on the current design.

```plaintext
int estimate_test_coverage
[-sample percent]
```

eval (dctcl-mode only)
Built-in Tcl command.
exec (dctcl-mode only)
Built-in Tcl command.

execute (dcsh-mode only)
Executes command arguments.

\[\text{int execute} \ [-s] \ \text{command} \ [\text{arguments}]\]

exit
Exits dc_shell.

\[\text{int exit} \ [\text{exit\_code\_value}]\]

expr (dctcl-mode only)
Built-in Tcl command.

externalize_cell
Makes a cell instance used by Behavioral Compiler external to the current design after scheduling.

\[\text{int externalize\_cell} \ \{\text{cell\_names}\}\]

extract
Extracts a state-machine representation from a netlist or HDL description.

\[\text{int extract} \ [-\text{minimize}] \ [-\text{reachable}]\]

extract_ilm
Transforms the current design into interface logic model (ILM) and writes out the model to the specified output file in .db format. After command execution, the design in memory is the interface logic model.

\[\text{int extract\_ilm} \ [-\text{include\_side\_load} \text{side\_load\_type}] \ [-\text{physical}] \ [-\text{verbose}] \ -\text{output} \ \text{file\_name}\]

fblocked (dctcl-mode only)
Built-in Tcl command.
**fconfigure** (dctcl-mode only)
Built-in Tcl command.

**fcopy** (dctcl-mode only)
Built-in Tcl command.

**file** (dctcl-mode only)
Built-in Tcl command.

**fileevent** (dctcl-mode only)
Built-in Tcl command.

**filter**
Returns a list of design objects that satisfy a conditional attribute expression.

```
list filter
object_list
expression
[-dont_check_real_objects]
```

**filter_collection** (dctcl-mode only)
Filters a collection, resulting in a new collection. The base collection remains unchanged.

```
collection filter_collection
base_collection
expression
[-regexp]
[-nocase]
```

**find**
Finds a design or library object.

```
list find
type
[name_list]
[-hierarchy]
[-flat]
```

**flush** (dctcl-mode only)
Built-in Tcl command.

**for** (dctcl-mode only)
Built-in Tcl command.
foreach

Specifies the control structure for list traversal loop execution.

```tcl
int foreach
    (variable_name, list_expression)
    {
        loop_statement_block
    }
```

foreach_in_collection (dctcl-mode only)

Iterates over the elements of a collection.

```tcl
string foreach_in_collection
    itr_var
collections
    body
```

format (dctcl-mode only)

Built-in Tcl command.

get_attribute

Returns the value of an attribute on a list of design or library objects.

```tcl
list get_attribute
    object_list
    attribute_name
    [-bus]
    [-quiet]
```

get_cells (dctcl-mode only)

Creates a collection of cells from the current design relative to the current instance. You can assign these cells to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```tcl
string get_cells
    [-hierarchical]
    [-quiet]
    [-regexp]
    [-nocase]
    [-exact]
    [-filter expression]
patterns | -of_objects objects
```
get_clocks (dctcl-mode only)

Creates a collection of clocks from the current design. You can assign these clocks to a variable or pass them into another command. For use in Tcl-based dc_shell only.

string get_clocks
    [-quiet]
    [-regexp]
    [-nocase]
    [-filter expression]
    patterns

get_clusters (dctcl-mode only)

Creates a collection of one or more clusters loaded into dc_shell. You can assign these clusters to a variable or pass them into another command. For use in Tcl-based dc_shell only.

string get_clusters
    [-hierarchical]
    [-quiet]
    [-regexp]
    [-nocase]
    [-exact]
    [-filter expression]
    [patterns]

get_design_lib_path

Returns the directory to which the specified library is mapped.

int get_design_lib_path library_name

get_design_parameter

Returns the value of a parameter on a parameterized design object. The parameter can be a generic in VHDL or a parameter in Verilog.

int get_design_parameter
    parameter_name
    [-quiet]
get\_designs (dctcl-mode only)

Creates a collection of one or more designs loaded into dc\_shell. You can assign these designs to a variable or pass them into another command. For use in Tcl-based dc\_shell only.

```tcl
string get\_designs
[-hierarchical]
[-quiet]
[-regexp]
[-nocase]
[-exact]
[-filter expression]
patterns
```

get\_generated\_clocks (dctcl-mode only)

Creates a collection of generated clocks. For use in Tcl mode of dc\_shell only.

```tcl
string get\_generated\_clocks
[-quiet]
[-regexp]
[-nocase]
[-exact]
[-filter expression]
patterns
```

get\_ilm\_objects

Returns a collection of nets, cells, or pins that are part of the interface logic for the current design.

```tcl
collection get\_ilm\_objects
[-type {net | pin | cell}]
```

get\_lib\_cells (dctcl-mode only)

Creates a collection of library cells from libraries loaded into dc\_shell. You can assign these library cells to a variable or pass them into another command. For use in Tcl-based dc\_shell only.

```tcl
string get\_lib\_cells
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
patterns | -of\_objects
objects
```
**get_lib_pins** (dctcl-mode only)

Creates a collection of library cell pins from libraries loaded into dc_shell. You can assign these library cell pins to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```
string get_lib_pins
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
patterns | -of_objects
objects
```

**get_libs** (dctcl-mode only)

Creates a collection of libraries loaded into dc_shell. You can assign these libraries to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```
string get_libs
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
patterns | -of_objects
objects
```

**get_license**

Obtains a license for a feature.

```
int get_license feature_list
```

**get_message_info** (dctcl-mode only)

Returns information about diagnostic messages.

```
string get_message_info
[-error_count | -warning_count | -info_count]
```
get_multibits (dctcl-mode only)
Creates a collection of one or more multibits loaded into dc_shell. You can assign these multibits to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```tcl
string get_multibits
[-hierarchical]
[-quiet]
[-regexp]
[-nocase]
[-exact]
[-filter expression]
[patterns]
```

get_nets (dctcl-mode only)
Creates a collection of nets from the current design. You can assign these nets to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```tcl
string get_nets
[-hierarchical]
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
patterns | -of_objects
objects
```

get_object_name (dctcl-mode only)
Returns the name of the object in a single-object collection. For use in Tcl-based dc_shell only.

```tcl
string get_object_name collection
```
get_path_groups (dctcl-mode only)
Creates a collection of path groups from the current design. You can assign these path groups to a variable or pass them into another command. For use in Tcl mode of dc_shell only.

```tcl
string get_path_groups
[-quiet]
[-regexp]
[-nocase]
[-filter expression]
patterns
```

get_pins (dctcl-mode only)
Creates a collection of pins from the current design. You can assign these pins to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```tcl
string get_pins
[-hierarchical]
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
[-leaf]
patterns | -of_objects
objects
```

get_ports (dctcl-mode only)
Creates a collection of ports from the current design. You can assign these ports to a variable or pass them into another command. For use in Tcl-based dc_shell only.

```tcl
string get_ports
[-filter expression]
[-quiet]
[-regexp]
[-nocase]
[-exact]
patterns | -of_objects
objects
```
get_references (dctcl-mode only)
 Creates a collection of one or more references loaded into dc_shell. You can assign these references to a variable or pass them into another command. For use in dc_shell -tcl mode only.

```
string get_references
[-hierarchical]
[-quiet]
[-regexp]
[-nocase]
[-exact]
[-filter expression]
patterns
```

get_timing_paths (dctcl-mode only)
 Creates a collection of timing paths for custom reporting and other processing. You can assign these timing paths to a variable or pass them into another command.

```
string get_timing_paths
[-to to_list]
[-from from_list]
[-through through_list]
[-delay_type delay_type]
[-nworst paths_per_endpoint]
[-max_paths max_path_count]
[-enable_preset_clear_arcs]
[-group group_name]
[-true]
[-true_threshold path_delay]
[-greater greater_limit]
[-lesser lesser_limit]
[-slack_greater_than greater_slack_limit]
[-slack_lesser_than lesser_slack_limit]
[-include_hierarchical_pins]
```

get_unix_variable
 Returns the value of a UNIX environment variable.

```
string get_unix_variable variable_name
```

getenv (dctcl-mode only)
 Returns the value of a system environment variable.

```
string getenv variable_name
```
gets (dctcl-mode only)
   Built-in Tcl command.

glob (dctcl-mode only)
   Built-in Tcl command.

global (dctcl-mode only)
   Built-in Tcl command.

group
   Creates a new level of hierarchy.

   int group
   [cell_list | -logic | -pla | -fsm]
   [-soft | -hdl_block block_name | -hdl_all_blocks | -hdl_bussed]
   [-design_name design_name]
   [-cell_name cell_name]
   [-except exclude_list]

group_path
   Groups a set of paths for cost function calculations.

   int group_path
   [-weight weight_value]
   [-critical_range range_value]
   -default | -name group_name
   [-from from_list]
   [-to to_list]
   [-through through_list]

group_variable
   Adds a variable to the specified variable group. This command is typically used by the system administrator only.

   int group_variable group_name variable_name
help
In standard dc_shell (dcsb mode), the help command
displays man pages for Synopsys commands. In
Tcl-based dc_shell (dctl mode), it displays quick
help for one or more commands.

```
dc_shell:
  int help [topic]
dc_shell-t:
  string help
     [-verbose]
     pattern
```

highlight_path
Highlights timing paths in a schematic.

```
int highlight_path
  [pin_port_list]
  [-from source_pins_or_ports]
  [layer_name]
  [-critical_path]
  [-min]
  [-max]
  [-min_rise]
  [-min_fall]
  [-max_rise]
  [-max_fall]
  [-no_auto_cycle]
```

history
Displays the history list (an ordered list of previously
executed commands).

```
dc_shell:
  int history
     [-h]
     [-r]
     [n]
dc_shell-t:
  string history
     [-h]
     [-r]
     [n]
     [keep m]
     [advanced_args]
```
identify_interface_logic
Sets the is_interface_logic attribute on cells, nets and pins of the current design that are part of its interface logic.

```
int identify_interface_logic
  [-ignore_ports port_list]
  [-latch_level levels]
```

if
Conditional execution control structure.

dc_shell:
```
if ( expression ) {
  then_statement_block
}
```

dc_shell-t:
```
if { expression } {
  then_statement_block
}
```

ignore_array_loop_precedences
Removes loop-carried dependencies between array accesses for the Behavioral Compiler product.

```
int ignore_array_loop_precedences
  [-process process_name]
  operations
```

ignore_array_precedences
Removes dependency-related constraints between array accesses for the Behavioral Compiler product.

```
int ignore_array_precedences
  [-process process_name]
  -from_set from_operations
  -to_set to_operations
```

ignore_memory_loop_precedences
Removes loop carried dependencies between memory accesses for the Behavioral Compiler product.

```
int ignore_memory_loop_precedences
  [-process process_name]
  operations
```
ignore_memory_precedences
Removes dependency related constraints between memory accesses for the Behavioral Compiler product.

```
int ignore_memory_precedences
    [-process process_name]
    -from set from_operations
    -to set to_operations
```

include (dcsh-mode only)
Executes a script of dc_shell commands. For use in standard dc_shell (dcsh mode) only.

```
int include
    file_name
    [-quiet]
```

incr (dctcl-mode only)
Built-in Tcl command.

index_collection (dctcl-mode only)
Given a collection and an index into it, if the index is in range, extracts the object at that index and creates a new collection containing only that object. The base collection remains unchanged.

```
collection index_collection
collection1
index
```

info (dctcl-mode only)
Built-in Tcl command.

insert_bsd
Creates ANSI/IEEE Std 1149.1-compliant boundary scan circuitry using DesignWare macro cells.

```
int insert_bsd
```

insert_clock_gating
Performs clock gating on an appropriately-prepared GTECH netlist.

```
int insert_clock_gating -mc
```
**insert_dft**

Adds test points and scan chains to the current design.

```c
int insert_dft
    [-dont_fix_constraint_violations]
    [-map_effort low | medium | high]
    [-ignore_compile_design_rules]
    [-no_scan]
    [-background run_name]
    [-host machine_name]
    [-arch architecture]
    [-xterm]
    [-physical]
```

**insert_jtag** (dctcl-mode only)

Built-in Tcl command.

**insert_pads**

Inserts I/O pads in a design.

```c
int insert_pads
    [-verify]
    [-verify_hierarchically]
    [-verify_effort low | medium | high]
    [-thru_hierarchy]
    [-respect_hierarchy]
    [design_list]
```

**insert_scan**

Adds scan circuitry to the current design.

```c
int insert_scan
    [-dont_fix_constraint_violations]
    [-map_effort low | medium | high]
    [-ignore_compile_design_rules]
    [-physical]
    [-background run_name]
    [-host machine_name]
    [-arch architecture]
    [-xterm]
```

**interp** (dctcl-mode only)

Built-in Tcl command.
is_false (dctcl-mode only)
Tests the value of a specified variable, and returns a 1 if the value is 0 or the case-insensitive string is false; returns a 0 if the value is 1 or the case-insensitive string is true. For use in Tcl-based dc_shell only.

```tcl
int is_false value
```

is_true (dctcl-mode only)
Tests the value of a specified variable, and returns a 1 if the value is 1 or the case-insensitive string is true; returns a 0 if the value is 0 or the case-insensitive string is false. For use in Tcl-based dc_shell only.

```tcl
int is_true value
```

join (dctcl-mode only)
Built-in Tcl command.

lappend (dctcl-mode only)
Built-in Tcl command.

lib2saif
Creates a SAIF forward-annotation file from a specified library file or from a list of library files.

```tcl
int lib2saif
[-output file_name]
[-lib_pathname lib_path_name]
library_file_name
```

library_analysis (dctcl-mode only)
Built-in Tcl command.

license_users
Lists the current users of the Synopsys licensed features.

```tcl
license_users [feature_list]
```

lindex (dctcl-mode only)
Built-in Tcl command.
link
   Resolves design references.
   int link

linsert (dctcl-mode only)
   Built-in Tcl command.

list
   In standard dc_shell (dcs mode), lists information about the commands, variables, and licenses currently in Design Analyzer or standard dc_shell (dcs mode).
   In Tcl-based dc_shell, it creates a list.
   dc_shell:
   int list {-commands | -variables
variable_group | variable_name | -licenses |
-files}
   dc_shell-t:
   list ? arg1 arg2 arg arg ...

list_attributes (dctcl-mode only)
   Lists currently defined attributes. For use in dc_shell-t (Tcl mode of dc_shell) only.
   string list_attributes
   [-application]
   [-class class_name]

list_designs
   List the designs available in dc_shell.
   int list_designs
   [design_list]
   [-show_file]

list_files (dctcl-mode only)
   Lists the files that are loaded into dc_shell. For use in Tcl-based dc_shell only.
   int list_files
list_instances
   Lists the instances in dc_shell.
   int list_instances [instance_list] [-hierarchy] [-max_levels num_levels] [-full]

list_libs
   Lists the available libraries in dc_shell or psyn_shell.
   int list_libs [lib_list]

list_licenses (dctcl-mode only)
   Displays a list of licenses currently checked out by the user. For use in dc_shell -tcl_mode only.
   string list_licenses

list_test_models
   List the designs with test models available in dc_shell.
   int list_test_models

list_test_modes
   Displays all the test modes defined for the current design.
   int list_test_modes

llength (dctcl-mode only)
   Built-in Tcl command.

lminus (dctcl-mode only)
   Removes one or more named elements from a list and returns a new list.
   list lminus the_list elements

load_of
   Returns the capacitance of the specified library cell pin.
   float load_of library_cell_pin
**lrange** (dctcl-mode only)
Built-in Tcl command.

**lreplace** (dctcl-mode only)
Built-in Tcl command.

**ls** (dctcl-mode only)
Lists the contents of a directory.

`int ls [filename ...]`

**lsearch** (dctcl-mode only)
Built-in Tcl command.

**lsort** (dctcl-mode only)
Built-in Tcl command.

**man** (dctcl-mode only)
Displays reference manual pages.

`string man topic`

**merge_saif**
Reads a list of SAIF files with their corresponding weights, annotates switching activity attributes with merged toggle_rate and merged static_probability for nets, pins, and ports in the current design, and generates a merged output SAIF file.

`int merge_saif`
-`-input_list saif_file_and_weight_list`
-`-instance_name inst_name`
-`[-output merged_saif_name]`
-`[-simple_merge]`
-`[-ignore ignore_name]`
-`[-ignore_absolute ig_absolute_name]`
-`[-exclude exclude_file_name]`
-`[-exclude_absolute ex_absolute_file_name]`
-`[-unit_base unit_value]`
-`[-scale scale_value]`
-`[-khrate khrate_value]`

**minimize_fsm**
Performs state minimization on a state table design.

`int minimize_fsm`
namespace (dctcl-mode only)
Built-in Tcl command.

open (dctcl-mode only)
Built-in Tcl command.

optimize_bsd
Optimizes ANSI/IEEE Std 1149.1-compliant boundary-scan circuitry synthesized by the insert_bsd command using DesignWare macro cells.

int optimize_bsd

optimize_registers
Performs (register) retiming on a mapped gate-level netlist; determines the placement of registers in a design to achieve a target clock period; and minimizes the number of registers while maintaining that clock period.

int optimize_registers [design_name]
[-period period_value]
[-no_clock_correction]
[-minimum_period_only]
[-flatten]
[-no_incremental_map]
[-no_compile]
[-sync_transform multiclass | decompose | dont_retime]
[-async_transform multiclass | decompose | dont_retime]
[-sync_state preserve | dont_care]
[-async_state preserve | dont_care]
[-check_design [-verbose]]
[-print_critical_loop]

package (dctcl-mode only)
Built-in Tcl command.

parent_cluster
Returns the name of the parent cluster of the passed cluster.

string parent_cluster cluster_object
parse_proc_arguments (dctcl-mode only)
Parses the arguments passed into a Tcl procedure.

```
string parse_proc_arguments
-args arg_list
result_array
```

partition_dp
Transforms arithmetic operators (for example, addition, subtraction, and multiplication) into datapath blocks to be implemented by a datapath generator for the specified design or for the current design.

```
int partition_dp
[design_name]
[-duplicate]
[-dont_split]
```

pid (dctcl-mode only)
Built-in Tcl command.

pipeline_design
Pipelines combinational designs by adding registers at the outputs and retiming the circuit.

```
int pipeline_design
design_name
[-period period_value]
[-flatten]
[-stages number_of_stages ]
[-stall_ports port_list]
[-stall_polarity high | low]
[-sync_reset reset_port |
-async_reset reset_port]
[-reset_polarity high | low]
[-clock_port_name clock_port]
[-no_incremental_map]
[no_compile]
[-check_design [-verbose]]
[-print_critical_loop]
[-no_clock_correction]
[-minimum_period_only]
[-register_outputs]
```
pipeline_loop

Specifies the name of a loop for Behavioral Compiler to pipeline.

```
int pipeline_loop
  loop_name
  -initiation_interval interval_time
  -latency latency_time
```

plot

Plots the schematic or symbol view for the current design using PostScript format.

```
int plot
  [-hierarchy]
  [-sheet_list sheet_number_list]
  [-output file_name]
  [-symbol_view]
  [-schematic_view]
```

preschedule

Directs Behavioral Compiler to schedule an operation or operations in a specific cycle for a specified process or for all processes.

```
int preschedule
  [-process process_name]
  {operation_names}
  cycle
```

preview_bsd

Previews the boundary-scan design.

```
int preview_bsd
  [-show cells | data_registers | instructions | tap | all]
  [-script]
```
preview_dft

Previews, but does not implement, the test points and scan chains to be added to the current design, based on the current set of scan and test point specifications.

```c
int preview_dft
[-verbose]
[-script]
[-show bidirectionals | cells | scan | scan_clocks | scan_signals | segments | tristates | all]
[-test_points all]
[-no_scan]
[-physical]
```

preview_scan

Previews the scan design.

```c
int preview_scan
[-command insert_scan | reoptimize_design]
[-script]
[-physical]
[-show bidirectionals | cells | scan | scan_clocks | scan_signals | segments | tristates | all]
```

print_suppressed_messages (dctcl-mode only)

Displays an alphabetical list of message ids that are currently suppressed.

```c
string print_suppressed_message
```

print_variable_group (dctcl-mode only)

Lists the variables defined in a specified variable group, along with their current values. For use in Tcl-based dc_shell only.

```c
int print_variable_group variable_group
```

printenv (dctcl-mode only)

Prints the value of environment variables.

```c
string printenv [variable_name]
```
printvar (dctcl-mode only)
   Prints the values of one or more variables.

   string printvar
       [pattern]
       [-userDefined]
       [-application]

proc (dctcl-mode only)
   Built-in Tcl command.

proc_args (dctcl-mode only)
   Displays the formal parameters of a procedure.

   string proc_args proc_name

proc_body (dctcl-mode only)
   Displays the body of a procedure.

   string proc_body proc_name

propagate_annotated_delay_up
   Propagates annotated delay information for the
   specified list of hierarchical cells from their respective
   reference designs to the current design. If no cell list
   is specified, the default list is the set of Interface
   Logic Model (ILM) sub-designs in the current design.

   int propagate_annotated_delay_up

propagate_constraints
   Propagates timing constraints from lower levels of the
   design hierarchy to the current design.

   int propagate_constraints
       [-design design_list]

propagate_placement_up
   Propagates placement information for all leaf cells in
   the specified list of hierarchical cells from their
   respective reference designs to the current design. If
   no cell list is specified, the default list is the set of
   Interface Logic Model (ILM) sub-designs in the
   current design.

   int propagate_placement_up
puts (dctcl-mode only)
   Built-in Tcl command.

pwd
   Displays the pathname of the present working directory (pwd), also called the current directory.

   int pwd

query_objects (dctcl-mode only)
   Searches for and displays objects in the database. For use in dc_shell-t (Tcl mode of dc_shell) only.
   
   string query_objects
   [-verbose]
   [-class class_name]
   [-truncate elem_count]
   object_spec

quit
   Exits dc_shell.

   int quit
**read**

In standard dc_shell (dcsh mode), reads designs into memory in Synopsys internal database (.db) format. In Tcl-based dc_shell (dctcl mode), reads from a channel. In lc_shell, reads libraries into memory in Synopsys internal database (.db) format.

**dcsh:**

```
list read
[-library library_name]
[-work library_name]
[-define macro_names]
[-format input_format]
[-single_file file_name]
[-names_file name_mapping_files]
[-define define_list]
file_list
```

**dctcl:**

```
read ?-nonewline? channelId
read channelId numBytes
```

**lc_shell:**

```
list read file_list
```

**read_bsd_init_protocol**

Reads a boundary-scan initialization protocol file.

```
int read_bsd_init_protocol
[init_protocol_file]
```

**read_bsd_protocol**

Reads a custom boundary-scan protocol file.

```
int read_bsd_protocol [protocol_file]
```

**read_clusters**

Annotates a design with physical cluster hierarchy data from a file in Physical Design Exchange Format (PDEF).

```
int read_clusters
[-design design_name]
cluster_file_name
```
**read_db** (dctcl-mode only)
Reads in one or more design or library files in Synopsys database (.db) format.

```
string read_db file_names
```

**read_edif** (dctcl-mode only)
Reads in one or more design or library files in EDIF format.

```
string read_edif file_names
```

**read_file**
Reads designs or libraries into dc_shell, or reads libraries into lc_shell.

**dc_shell:**
```
list read_file
[-library library_name]
[-work library_name]
[-define macro_names]
[-format input_format]
[-netlist | -rtl]
[-single_file file_name]
[-names_file name_mapping_files]
file_list
```

**lc_shell:**
```
list read_file file_list
```

**read_init_protocol**
Reads an initialization protocol file.

```
int read_init_protocol
[init_protocol_file]
[-format tpf | stil]
```

**read_lib**
Reads a technology library, physical library, or symbol library into dc_shell or lc_shell. Creates a physical library for gdsii.

```
int read_lib
[-format format_name]
[-symbol intermediate_symbol_library_file]
[-plibrary physical_library_output_file]
[-pplibrary pseudo_physical_file_name]
```
read_parasitics
Reads net parasitics information from an SPEF file, and uses it to annotate the current design.

string read_parasitics
  -elmore
  [-increment]
  [-pin_cap_included]
  [-net_cap_only]
  [-complete_with zero | wlm]
  [-syntax_only]
  [-path path_name]
  [-strip_path path_name]
  [-quiet] file_names

read_partition (dctcl-mode only)
Reads the database for a design. For use only in dc_shell-t (Tcl mode of dc_shell).

int read_partition
  -source pass
  -type pre | post
design_name

read_pin_map
Reads in a port-to-pin mapping file, which defines the design port-to-package pin mapping for a boundary-scan design.

int read_pin_map path_name

read_preserved_function_netlist
Reads an external precompiled netlist for a preserved function.

int read_preserved_function_netlist
  [preserved_functions]
  [-exclude preserved_functions]
  [-design_library library_name]
  [-force_reload]
  [-filename filename]
  [-return_port return_port_name]
  [-clock_port_name clock_port_name]
  [-clock_edge positive | negative]
  [-sync_reset reset_port_name]
  [-async_reset reset_port_name]
  [-reset_polarity high | low]
**read_saif**
Reads a SAIF file and annotates the switching activity attributes toggle_rate and static_probability for nets, pins, and ports in the current design.

```c
int read_saif -input file_name
-instance_name name
[-ignore ignore_name]
[-ignore_absolute ig_absolute_name]
[-exclude exclude_file_name]
[-exclude_absolute ex_absolute_file_name]
[-names_file name_changes_log_file]
[-unit_base unit_value]
[-scale scale_value]
[-khrate khrate_value]
[-rtl_direct]
[-verbose]
```

**read_sdc** *(dctcl-mode only)*
Reads in a script in Synopsys Design Constraints (SDC) format.

```c
int read_sdc
file_name
[-echo]
[-syntax_only]
[-version sdc_version]
```

**read_sdf**
Reads leaf cell and net timing information from a file in Standard Delay Format (SDF) and uses that information to annotate the current design.

```c
string read_sdf
[-load_delay net | cell]
[-path path_name]
[-min_type sdf_min | sdf_typ | sdf_max]
[-max_type sdf_min | sdf_typ | sdf_max]
[-worst]

sdf_file_name
```

**read_test_model**
Reads a test model file.

```c
list read_test_model
[-format ctl | db]
[-design design_name]

model_files
```
read_test_protocol
Reads a custom test protocol file.

```c
int read_test_protocol
[protocol_file]
[-format tpf | stil]
```

read_toggle
Reads a TOGGLE file and annotates the switching activity attributes toggle_rate and static_probability onto nets, pins, and ports in the database file of the current design.

```c
int read_toggle
-input file_name
-instance_name name
[-unit_base unit_value]
[-scale scale_value]
[-khrate khrate_value]
[-names_file change_name_log_file]
[-case_insensitive]
[-output saif_file_name]
```

read_verilog (dctcl-mode only)
Reads in one or more design or library files in Verilog format.

```c
string read_verilog
[-netlist | -rtl]
file_names
```

read_vhdl (dctcl-mode only)
Reads in one or more design or library files in VHDL format.

```c
string read_vhdl file_names
```

redirect (dctcl-mode only)
Redirects the output of a command to a file.

```c
string redirect
[-append]
file_name
{command}
```
reduce_fsm
Reduces the logic for each state transition in a state table design.

int reduce_fsm

reg_global_var (dctcl-mode only)
Built-in Tcl command.

regexp (dctcl-mode only)
Built-in Tcl command.

regsub (dctcl-mode only)
Built-in Tcl command.

remove_analysis_info
Removes BCView analysis information from the specified design.

int remove_analysis_info [design]

remove_annotated_check
Removes annotated timing check information.

int remove_annotated_check
-all | -from from_list | -to to_list
[-rise | -fall]
[-clock rise | fall]
[-setup]
[-hold]
[-recovery]
[-removal]
[-nochange_low]
[-nochange_high]

remove_annotated_delay
Removes the annotated delay between two pins.

int remove_annotated_delay
-all | -cell_all | -net_all | -from from_list | -to to_list

remove_annotated_transition
Removes the annotated transition at a pin.

int remove_annotated_transition
-all | -at pin_list
remove_attribute
Removes an attribute from a design or library object.

```c
list remove_attribute
object_list
attribute_name
[-bus]
[-quiet]
```

remove_bsd_instruction
Removes boundary-scan instructions from the instruction list to be used by insert_bsd for the current design.

```c
int remove_bsd_instruction instruction_list
```

remove_bsd_port
Removes from specified ports the attributes that identify those ports as IEEE Std 1149.1 test access ports (TAPs) in the current design.

```c
int remove_bsd_port port_list
```

remove_bsd_signal
Removes boundary-scan signal types from specified ports in the current design.

```c
int remove_bsd_signal port_list
```

remove_bsd_specification
Removes the IEEE 1149.1 specifications from a boundary-scan design.

```c
int remove_bsd_specification
[-all]
[-configuration]
[-compliance all|pattern_name]
[-intest]
[-runbist]
[-data_cell port_name]
[-control_cell hierarchical_cell_name]
[-path]
[-tap_element design_name]
[-bsr_element design_name]
[-linkage_port]
[-register]
[-instruction instruction_list]
```
remove_bsr_cell_type
   Removes the boundary-scan cell type specification from a set of design ports.
   int remove_bsr_cell_type port_list

remove_bus
   Removes a port bus or net bus.
   int remove_bus object_list

remove_cache
   Selectively removes elements from the synthetic library cache directories.
   int remove_cache
   [-design_lib list]
   [-module list]
   [-implementation list]
   [-parameters parameter_list]
   [-tech_lib list]
   [-wire_load list]
   [-operating_conditions list]
   [-directory dir_list]
   [-smaller size | -larger size]
   [-netlist_only | -model_only]
   [-accessed_since days | -accessed_beyond days]

remove_case_analysis
   Removes the case analysis value from specified input ports or pins.
   string remove_case_analysis
   port_or_pin_list | -all

remove_cell
   Removes cells from the current design.
   int remove_cell cell_list | -all

remove_cell_degradation
   Removes the cell_degradation attribute on specified ports or designs.
   int remove_cell_degradation object_list
**remove_clock**
Removes clocks from the current design.

```plaintext
int remove_clock clock_list | -all
```

**remove_clock_gating**
Directs the compile -incremental, physopt, and physopt -incremental commands to remove clock gating from registers clock-gated by Power Compiler.

```plaintext
int remove_clock_gating
[-gating_cells clock_gating_cells_list]
[-gated_registers gated_register_list]
[-all]
[-hier]
[-verbose]
[-undo]
```

**remove_clock_gating_check**
Removes setup and hold checks from the specified clock gating cells.

```plaintext
int remove_clock_gating_check
[-setup]
[-hold]
[-rise]
[-fall]
object_list
```

**remove_clock_latency**
Removes clock latency information from the specified objects.

```plaintext
string remove_clock_latency
[-fall]
[-min]
[-max]
[-source]
[-early]
[-late]
object_list
```

**remove_clock_transition**
Removes clock transition attributes on the specified clock objects.

```plaintext
int remove_clock_transition clock_list
```
**remove_clock_uncertainty**

Removes clock uncertainty information.

```tcl
string remove_clock_uncertainty
object_list
[-from from_clock]
[-to to_clock]
[-rise]
[-fall]
[-setup]
[-hold]
```

**remove_clusters**

Removes the physical cluster hierarchy associated with a design.

```tcl
int remove_clusters [designs_or_clusters]
```

**remove_constraint**

Removes all constraint attributes, clocks, and path delay information from the current design.

```tcl
int remove_constraint -all
```

**remove_design**

Removes a list of designs or libraries from dc_shell memory.

```tcl
int remove_design
[design_list | -designs | -all]
[-hierarchy]
[-quiet]
```

**remove_dft_configuration**

Removes the current DFT configuration from the current design.

```tcl
int remove_dft_configuration
```

**remove_dft_signal** (dctcl-mode only)

Built-in Tcl command.
**remove_disable_clock_gating_check**

For specified cells and pins, restores clock gating checks previously disabled by the `set_disable_clock_gating_check` command.

```
string remove_disable_clock_gating_check
object_list
```

**remove_driving_cell**

Removes driving cell attributes from the specified input or inout ports of the current design.

```
int remove_driving_cell [port_list]
```

**remove_from_collection** (dctcl-mode only)

Removes objects from a collection, resulting in a new collection. The base collection remains unchanged.

```
collection remove_from_collection
base_collection
```

**remove_generated_clock**

Removes a generated_clock object.

```
string remove_generated_clock
-all | clock_list
```

**remove_highlighting**

Removes highlighting from schematics of the current instance or the current design.

```
int remove_highlighting [-all | -hier]
```

**remove_ideal_latency**

Removes user-specified ideal latency information from specified objects.

```
string remove_ideal_latency
[-rise | -fall]
[-min]
[-max]
object_list
```

**remove_ideal_net**

Removes the ideal_net at -tribute from the specified nets in the current design.

```
int remove_ideal_net net_list
```
remove_ideal_network
Removes a set of ports or pins in an ideal network in the current design. Cells and nets in the transitive fanout of the specified objects are no longer treated as ideal.

int remove_ideal_network object_list

remove_ideal_transition
Removes ideal transition information from the specified objects.

string remove_ideal_transition
[-rise | -fall]
[-min]
[-max]
object_list

remove_input_delay
Removes input delay on pins or input ports.

int remove_input_delay
[-clock clock]
[-clock_fall]
[-level_sensitive]]
[-rise]
[-fall]
[-max]
[-min]
port_pin_list

remove_isolate_ports
Removes the specified ports from the list of ports that are isolated in the current design.

int remove_isolate_ports port_list

remove_license
Removes a licensed feature.

int remove_license feature_list

remove_multibit
Removes multibit components in the current design, or detaches cells in the current design from the multibit components that contain them.

int remove_multibit object_list
remove_net
Removes nets from the current design.

int remove_net
net_list | -all
[-only_physical]

remove_operand_isolation
Removes the isolation logic that Power Compiler inserted in the circuit during operand isolation.

int remove_operand_isolation
[-from from_list]
[-to to_list]

remove_output_delay
Removes output delay on pins or output ports.

int remove_output_delay
[-clock clock
[-clock_fall]
[-level_sensitive]]
[-rise]
[-fall]
[-max]
[-min]
port_pin_list

remove_pads
Removes I/O pads from a design.

int remove_pads
[-verify]
[-verify_effort effort]
[design_list]

remove_pass_directories (dctcl-mode only)
Removes the data directories associated with the specified passes. For use only in dc_shell-t (Tcl mode of dc_shell).

int remove_pass_directories pass_list

remove_pin_map
Removes a design port-to-package pin mapping for a boundary-scan design.

int remove_pin_map package_name
**remove_port**

Removes ports from the current design.

```plaintext
int remove_port port_list | -all
```

**remove_port_configuration**

Removes the Shadow LogicDFT port configuration from the specified port on the specified list of elements, so that insert_dft inserts the default test point circuitry.

```plaintext
int remove_port_configuration
-cell cell_design_ref_list
-port port_name
```

**remove_propagated_clock**

Removes a propagated clock specification.

```plaintext
string remove_propagated_clock object_list
```

**remove_rtl_load**

Removes previously-set capacitance and resistance RTL load values from pins, ports, and nets.

```plaintext
int remove_rtl_load -all | pin_net_list
```

**remove_scan_register_type**

Removes existing scan register types, previously set by set_scan_register_type, from specified cells or from the current design.

```plaintext
int remove_scan_register_type
-cell_or_design_list
```

**remove_scan_specification**

Removes scan specifications from the current design.

```plaintext
int remove_scan_specification
[-all] [-bidirectional] [-chain chain_name_list] [-configuration] [-link link_name_list] [-segment segment_name_list] [-signal port_name_list] [-tristates]
```
remove_scheduling_constraints
Removes explicit Behavioral Compiler scheduling constraints from the specified process or from all processes.

int remove_scheduling_constraints
[-process process_name]

remove_test_mode
Remove the mode declared by the define_test_mode command.

int remove_test_mode design_name

remove_test_model
Permanently removes the test model associated with a design.

int remove_test_model [-design design_name]

remove_unconnected_ports
Removes unconnected ports or pins from cells, references, and subdesigns.

int remove_unconnected_ports
cell_list
[-blast_buses]

remove_variable (dcsh-mode only)
Removes a variable from dc_shell.

int remove_variable variable_name

remove_wire_load_min_block_size
Removes the wire_load_min_block_size attribute from the current design.

int remove_wire_load_min_block_size
remove_wire_load_model
   Removes wire load model attributes from designs, ports, hierarchical cells, or the specified cluster of the current design.
   int remove_wire_load_model
   [-min]
   [-max]
   [-cluster cluster_name]
   [object_list]

remove_wire_load_selection_group
   Removes wire load model selection group from designs and cells, or from a specified cluster of the current design.
   int remove_wire_load_selection_group
   [-min]
   [-max]
   [-cluster cluster_name]
   [object_list]

remove_wrapper_element
   Removes the wrapper_element attribute from a list of elements.
   int remove_wrapper_element
   cell_design_ref_list

rename (dctcl-mode only)
   Rename or delete a user-defined procedure.
   string rename
   oldName
   newName

rename_design
   Renames a design in dc_shell, or moves a list of designs to a file.
   int rename_design
   design_name1
   design_name2
reoptimize_design
  Incrementally optimizes a design using layout/floorplanning information.

  int reoptimize_design
  [-map_effort low | medium | high]
  [-sizing]
  [-pin_swap]
  [-buffer_insertion]
  [-buffer_removal]
  [-cpr]
  [-tolerance_to_change low | medium | high]
  [-ignore_footprint]
  [-ignore_cell_area]
  [-no_design_rule | -only_design_rule]
  [-only_hold_time]
  [-area_recovery]

replace_fpga (dctcl-mode only)
  Built-in Tcl command.

replace_synthetic
  Implements all synthetic library parts of a design using generic logic.

  int replace_synthetic [-ungroup]

report_annotated_check
  Displays all annotated timing checks on the current design.

  int report_annotated_check [-nosplit]

report_annotated_delay
  Displays all annotated delays on cells and nets of the current design.

  int report_annotated_delay
  [-cell]
  [-net]
  [-min]
  [-nosplit]

report_annotated_transition
  Displays annotated transitions on all pins of the current design.

  int report_annotated_transition
**report_area**
Displays area information and statistics for the design of the current instance, if set; or for the current design otherwise. If the design is an Interface Logic Model (ILM), the report also displays area information and statistics of the original design for which the ILM was created.

```plaintext
int report_area
[-nosplit]
[-physical]
```

**report_attribute**
Displays attributes and their values associated with a cell, net, pin, port, instance, or design.

```plaintext
int report_attribute
[object_list] | [-net]
[-cell]
[-design]
[-hierarchy]
[-instance]
[-pin]
[-port]
[-reference]
[-nosplit]
```

**report_auto_ungroup**
Displays information about the cell hierarchies that have been ungrouped with compile -auto_ungroup area | delay.

```plaintext
int report_auto_ungroup
[-nosplit]
[-full]
```

**report_buffer_tree**
Displays the buffer tree and its level information at the given driver pin.

```plaintext
int report_buffer_tree
[-from start_point_list | -net net_list]
[-depth max_depth]
[-connections]
[-hierarchy]
[-physical]
[-nosplit]
```
**report_bus**

Lists the bused ports and nets in the current instance, if set; or in the current design otherwise.

```
int report_bus [-nosplit]
```

**report_cache**

Reports on the contents of synthetic library caches.

```
int report_cache
[-design_lib list]
[-module list]
[-implementation list]
[-parameters parameter_list]
[-tech_lib list]
[-wire_load list]
[-operating_conditions list]
[-directory dir_list]
[-smaller size | -larger size]
[-accessed_since days | -accessed_beyond days]
[-netlist_only | -model_only]
[-sort_largest | -sort_oldest | -sort_cache_key]
[-statistics]
```

**report_case_analysis**

Reports case analysis on ports or pins.

```
string report_case_analysis
[-all]
[-nosplit]
```

**report_cell**

Displays information about cells in the current instance, if set; or in the current design otherwise.

```
int report_cell
[-nosplit]
[-connections [-verbose]]
[-physical [-verbose]]
[-only_physical]
[cell_list]
```
report_clock
Displays clock-related information on the current design.

int report_clock
[-attributes]
[-skew]
[-nosplit]

report_clock_gating
Reports information about clock gating performed by Power Compiler.

report_clock_gating [-hier]
[-verbose]
[-gated]
[-ungated]
[-gating_elements]
[-only cell_list]
[-only]
[-nosplit]
[-physical]

report_clock_gating_check
Prints a report of the clock gating checks.

string report_clock_gating_check
[-nosplit]
[-type user | power | all]
[instance_list]

report_clusters
Reports on the physical cluster hierarchy associated with the current design.

int report_clusters
[-cluster cluster_name]
[-leaf]
[-nosplit]

report_compile_options
Displays information about the compile options for the design of the current instance, if set; or for the current design otherwise.

int report_compile_options
[-nosplit]
report_constraint
Displays constraint-related information about a design.

```bash
int report_constraint
[-all_violators]
[-verbose]
[-significant_digits digits]
[-max_area]
[-max_delay]
[-critical_range]
[-min_delay]
[-max_capacitance]
[-min_capacitance]
[-max_transition]
[-max_fanout]
[-cell_degradation]
[-min_porosity]
[-max_dynamic_power]
[-max_leakage_power]
[-connection_class]
[-multiport_net]
[-nosplit]
```

report_delay_calculation
Displays the actual calculation of a timing arc delay value for a cell or net.

```bash
int report_delay_calculation
-min
-max
-from from_pin
-to to_pin
[-nosplit]
```

report_design
Displays attributes of the current design.

```bash
int report_design
[-nosplit]
[-physical]
[-verbose]
```
**report_design_lib**
Lists the design units contained in the specified libraries.

```
int report_design_lib
[-libraries]
[-designs]
[-architectures]
[-packages]
[library_list]
```

**report_direct_power_rail_tie**
Reports all the library pins on which the attribute `direct_power_rail_tie` is set to `true`.

```
int report_direct_power_rail_tie
```

**report_disable_timing**
Reports disabled timing arcs in the current design.

```
string report_disable_timing [-nosplit]
```

**report_fpga** (dctcl-mode only)
Built-in Tcl command.

**report_fsm**
Displays state-machine attributes and information for the design of the current instance, if set; or for the current design otherwise.

```
int report_fsm [-nosplit]
```

**report_hierarchy**
Displays the reference hierarchy of the current instance, if set; or of the current design otherwise.

```
int report_hierarchy
[-nosplit]
[-full]
```
**report_ideal_network**
Displays information about ports, pins, nets, and cells on ideal networks in the current design.

```
int report_ideal_network
[-net]
[-cell]
[-load_pin]
[-timing]
[object_list]
```

**report_internal_loads**
Displays internal loads on the nets in the current design.

```
int report_internal_loads [-nosplit]
```

**report_isolate_ports**
Displays the status of port isolation on ports on which isolation was requested.

```
int report_isolate_ports [-nosplit]
```

**report_lib**
Displays information about technology or symbol libraries.

```
int report_lib
library_name
[-timing_arcs]
[-timing]
[-power]
[-em]
[-vhdl_name]
[-table]
[-full_table]
[-timing_label]
[-power_label]
[-routing_rule]
[-rwm]
[-fpga]
[-all]
[cell_list]
```
report_mode
Prints a report of the instance modes.

string report_mode
[-nosplit]
[instance_list]

report_multibit
Displays information about multibit components in the current design.

int report_multibit
[-nosplit]
[object_list]

report_multicycles
Reports on scheduled multicycle operations for Behavioral Compiler.

int report_multicycles
[-process process_name]

report_name_rules
Reports the values of name rules.

int report_name_rules [name_rules]

report_names
Reports potential name changes of ports, cells, and nets in a design.

int report_names
[-rules name_rules]
[-hierarchy]
[-original]
[-nosplit]
**report_net**

Displays net information for the design of the current instance, if set; or for the current design otherwise.

```
int report_net
[-nosplit]
[-noflat]
[-transition_times]
[-only_physical [-verbose]]
[-cell_degradation]
[-min]
[-connections [-verbose]]
[-physical [-verbose]]
[net_list]
[-max_toggle_rate]
```

**report_net_fanout**

Displays net fanout or buffer tree information for the design of the current instance, if set; or for the current design otherwise.

```
int report_net_fanout
[-nosplit]
[-high_fanout]
[-threshold lower]
[-bound upper]
[-verbose]
[-connections]
[-physical]
[-min]
[-tree [-depth level]]
[net_list]
```

**report_operand_isolation**

Reports the status of operand isolation cells in the current design.

```
int report_operand_isolation
```

**report_packages**

Displays the package names for all the port-to-pin mapping files that are read in for a boundary-scan design.

```
int report_packages
```
**report_partitions** (dctcl-mode only)
Lists the hierarchical designs and their associated attributes and relative size (estimated in RTL). For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int report_partitions [-nosplit]
```

**report_pass_data** (dctcl-mode only)
Reports the data files that are available for a design created by Automated Chip Synthesis. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int report_pass_data
[-hierarchy]
[-pass_list pass_list]
[design]
```

**report_path_budget**
Displays budgeting information about a design. The delay number shown for each block/cell shows the amount of budget allocated for that block/cell in the path. The overall budget for the path is the sum of these budgets (delays).

```
int report_path_budget
[-to to_list]
[-from from_list]
[-through through_list]
[-nworst paths_per_endpoint]
[-max_paths max_path_count]
[-input_pins]
[-nets]
[-transition_time]
[-capacitance]
[-significant_digits digits]
[-nosplit]
[-sort_by group | slack]
[-all]
[-verbose]
```

**report_path_group**
Reports information about path groups in the current design.

```
int report_path_group [-nosplit]
```
report_port
Displays information about ports for the design of the current instance, if set; or for the current design otherwise.

int report_port
[ -drive]
[ -verbose]
[ -physical]
[ -nosplit]
[ port_list]

report_power
Calculates and reports dynamic and static power for a design or instance.

int report_power
[ -net]
[ -cell]
[ -only cell_or_net_list]
[ -hier]
[ -hier_level level_value]
[ -verbose]
[ -cumulative]
[ -flat]
[ -exclude_boundary_nets]
[ -analysis_effort low | medium | high]
[ -nworst number]
[ -sort_mode mode]
[ -histogram [ -exclude_leq le_val | -exclude_geq ge_val]]
[ -nosplit]

report_reference
Displays information about references in the current instance, if set; or in the current design otherwise.

int report_reference [ -nosplit]

report_resource_estimates
Displays timing and area estimates for Behavioral Compiler operations in the current design.

int report_resource_estimates
report_resources
Lists the resources and datapath blocks used in the
design of the current instance, if set; or in the current
design otherwise.

int report_resources
[-nosplit]
[-hierarchy]

report_routability
Displays information about the routability of the
current design.

int report_routability [-nosplit]

report_saif
Reports switching activity annotations for nets, pins,
and ports of the current design or instance.

int report_saif
[-flat]
[-type rtl | gate]
[-missing]
[-only cell_or_net_list]

report_schedule
Displays the results of scheduling and allocation as
performed by Behavioral Compiler.

int report_schedule
[-process process_name]
[-operations [-mask [r][w][l][L][o][p]]
[-start start_cycle]
[-finish end_cycle]
[-delimiter character]]
[-variables [-min min_width]
[-max max_width]
[-start start_cycle]
[-finish end_cycle]
[-delimiter character]]
[-summary]
[-abstract_fsm [-mask [r][w][o][s]]]
[-verbose_fsm [-mask [r][w][o][d][s]]]
**report_scheduling_constraints**

Displays Behavioral Compiler scheduling constraints on the current design for a specified process or for all processes.

```c
int report_scheduling_constraints
[-process process_name]
[-dont_chain]
[-chains]
[-antichains]
[-two_point]
[-preschedule]
[-pipeline]
```

**report_synlib**

Displays information about synthetic libraries.

```c
int report_synlib library [module_list]
```

**report_test**

Displays test-related information about the current design.

```c
int report_test
[-assertions]
[-bsd]
[-bsd_configuration]
[-configuration]
[-constraints]
[-dft]
[-inst design_instance_list]
[-inst design_instance_list]
[-methodology]
[-port]
[-scan_path]
[-state]
[-trace_nets]
[-nosplit]
```

**report_test_mode**

Displays the test mode information attached to a model.

```c
int report_test_mode
[-design design_name]
[-all]
[-verbose]
[test modes]
```
**report_test_model**

Displays the test model information attached to a design.

```
int report_test_model [-design design_name]
```

**report_timing**

Displays timing information about a design.

```
int report_timing
 [-to to_list]
 [-from from_list]
 [-through through_list]
 [-path short | full | full_clock | only | end]
 [-delay min | min-rise | min-fall | max | max-rise | max-fall]
 [-nworst paths_per_endpoint]
 [-max_paths max_path_count]
 [-input_pins]
 [-nets]
 [-transition_time]
 [-capacitance]
 [-attributes]
 [-locations]
 [-physical]
 [-lesser_path max_path_delay]
 [-greater_path min_path_delay]
 [-loops]
 [-true [-true_threshold path_delay]]
 [-justify]
 [-enable_preset_clear_arcs]
 [-significant_digits digits]
 [-nosplit]
 [-sort_by group slack]
```

**report_timing_requirements**

Reports timing path requirements (user attributes) and related information.

```
int report_timing_requirements
 [-attributes]
 [-ignored]
 [-from from_list]
 [-through through_list]
 [-to to_list]
 [-expanded]
 [-nosplit]
```

---

90 Synthesis Commands
report_transitive_fanin
Reports logic in the transitive fanin of specified sinks.

    int report_transitive_fanin
    -to sink_list
    [-nosplit]

report_transitive_fanout
Reports logic in the transitive fanout of specified sources.

    int report_transitive_fanout
    -clock_tree | -from source_list
    [-nosplit]

report_ultra_optimization
Reports on whether or not the DC Ultra optimization mode is set and whether licenses have been checked out correctly.

    int report_ultra_optimization

report_wire_load
Displays the characteristics of the wire load models set on a design or in a library.

    int report_wire_load
    [-design design_name]
    [-name model_name]
    [-libraries]
    [-nosplit]

report_xref
Generates a cross-reference between schematic objects and sheets on which they occur.

    int report_xref [-nosplit]

reset_compare_design_script
Removes the compare_design script if it exists.

    int reset_compare_design_script
reset_design
Removes from the current design all user-specified objects and attributes, except those defined using set_attribute.
int reset_design

reset_mode
Resets the modes of the specified instances.
int reset_mode [instance_list]

reset_path
Resets specified paths to single cycle timing.
int reset_path
[-setup | -hold]
[-rise | -fall]
[-from from_list]
[-through through_list]
[-to to_list]

reset_switching_activity
Removes the toggle_rate and static_probability attributes, and/or the max_toggle_rate attribute, from nets, pins, cells and/or ports of the current design.
int reset_switching_activity
-switching_activity | -max_toggle_rate
| -all
[-verbose]

restore_test (dctcl-mode only)
Built-in Tcl command.

return (dctcl-mode only)
Built-in Tcl command.
rewire_clock_gating
Provides a mechanism for directing compile -incremental, physopt, and physopt -incremental to change the clock-gating cell implemented by Power Compiler for a particular register.

```
int rewire_clock_gating
 [-gating_cell new_clock_gating_cell]
 [-gated_registers gated_registers_list]
 [-verbose]
 [-undo]
```

rtl2saif
Creates a SAIF forward-annotation file starting from the top level of the design.

```
int rtl2saif
 [-output file_name]
 [-design design_name]
```

rtl_analyzer (dctcl-mode only)
Built-in Tcl command.

rtldrc
Analyzes the testability of a design at the RTL and gate level, using Verilog or VHDL RTL sources.

```
int rtldrc
 [-tristate]
 [-max_detail_lines n]
```

scan (dctcl-mode only)
Built-in Tcl command.

schedule
Invokes the scheduling and allocation functions of Behavioral Compiler.

```
int schedule
 [-effort quick | low | medium | high]
 [-io_mode cycle_fixed | superstate_fixed]
 [-extend_latency]
 [-host hostname]
 [-arch remote_host_architecture]
 [-allocation_effort quick | low | medium | high]
```
seek (dctcl-mode only)
Built-in Tcl command.

set (dctcl-mode only)
Built-in Tcl command.

set_annotated_check
Sets the setup, hold, recovery, or removal timing check value between two pins.

int set_annotated_check
  check_value
  -from from_pins -to to_pins
  -setup | -hold | -recovery | -removal |
  -nochange_high | -nochange_low
  [-rise | -fall]
  [-clock clock_check]
  [-worst]

set_annotated_delay
Sets the net or cell delay value between two pins.

int set_annotated_delay
  -net | -cell
  [-load_delay load_delay_type]
  [-rise | -fall]
  [-min]
  [-max]
  delay_value
  -from from_pins -to to_pins
  [-worst]

set_annotated_transition
Sets the transition time at a given pin.

int set_annotated_transition
  [-rise | -fall]
  [-min]
  [-max] transition
  port_pin_list
set_attribute
Sets the value of an attribute on a design or library object.

list set_attribute object_list attribute_name attribute_value
[-type boolean | integer | float | string]
[-bus]
[-quiet]

set_auto_disable_drc_nets
Sets the auto_disable_drc_net attribute on the current design, causing the specified networks to be have DRC disabled. This command was previously called set_auto_ideal_nets.

int set_auto_disable_drc_nets
[-default]
[-none]
[-all]
[-clock true | false]
[-constant true | false]
[scan true | false]

set_autofix_async
Specifies the asynchronous port to be used to automatically correct uncontrollable asynchronous violations of specified cells, during execution of preview_dft or insert_dft, when Autofix utility is enabled.

int set_autofix_async
async_port cell_list

set_autofix_clock
If Autofix is enabled, specifies a clock port to be used for specified cells, for automatic fixing of uncontrollable clock violations during execution of preview_dft or insert_dft.

int set_autofix_clock clock_port cell_list
set_autofix_configuration
If Autofix is enabled, optionally disables automatic fixing of all uncontrollable clock violations or all asynchronous preset/clear violations, during execution of preview_dft or insert_dft.

```text
int set_autofix_configuration
[-clock true | false]
[-async true | false]
[-fix_async_with_scan_en true | false]
```

set_autofix_element
If Autofix is enabled, optionally disables automatic fixing of uncontrollable clock violations or asynchronous preset/clear violations for specified sequential elements, during execution of preview_dft or insert_dft.

```text
int set_autofix_element
cell_list
-c clock true | false
-async true | false
```

set_balance_registers
Sets the balance_registers attribute on the specified designs or on the current design, so that the design is retimed during compile.

```text
int set_balance_registers
[true | false]
[-design design_list]
```

set_behavioral_reset
Directs Behavioral Compiler to set reset behavior for process, port, synchronicity, active state, direct connection or FSM generation.

```text
int set_behavioral_reset
[-process process_name]
[-port signal_name]
[-active high | low]
[-fsm]
[-async]
[-all]
```
set_boundary_optimization
Sets the boundary_optimization attribute on specified cells, references, or designs, thus allowing for optimization across hierarchical boundaries.

```c
int set_boundary_optimization
obj_list [true | false]
```

set_bsd_bsr_element
Characterizes a design cell as a boundary-scan register to be used for the boundary-scan insertion.

```c
int set_bsd_bsr_element
-type cell_type
-design design_name
-access access_list
```

set_bsd_compliance
Specifies an IEEE 1149.1 compliance-enable pattern for a boundary-scan design.

```c
int set_bsd_compliance
pattern_name
signal_port_bit_value_pairs
```

set_bsd_configuration
Specifies the boundary-scan configuration for a design.

```c
int set_bsd_configuration
[-asynchronous_reset true | false]
[-default_package package_name]
[-instruction_encoding default | one_hot]
[-ir_width instruction_register_length]
[-style asynchronous | synchronous]
[-infer_instructions true | false]
[-check_pad_designs none | all | pad_designs_list]
```

set_bsd_control_cell
Declares a boundary-scan control register cell to control tristate port pads.

```c
int set_bsd_control_cell
BSD_control_register_name -type cell_type
-port_list port_list
```
**set_bsd_data_cell**
Specifies a boundary-scan register cell type to be used on a specified list of ports in the current design.

```
int set_bsd_data_cell cell_type
    -port_list port_list
    [-direction in | out]
```

**set_bsd_instruction**
Specifies boundary-scan instructions to be used by `insert_bsd` for the current design or used by `check_bsd` in the verification flow.

```
int set_bsd_instruction instruction_set
    [-code inst_code_list]
    [-register user_data_reg]
    [-input_clock_condition clock_conditioning]
    [-output_condition output_conditioning]
    [-internal_scan pin_name]
```

**set_bsd_intest**
Specifies the parameters for the INTEST instruction.

```
int set_bsd_intest
    [-time real_numbers]
    [-clock_cycles clock_port_integer_pairs]
```

**set_bsd_linkage_port**
Identifies the linkage ports in your design.

```
int set_bsd_linkage_port
    -port_list list_of_ports
```

**set_bsd_pad_design**
Specifies and characterizes a design present in memory as a pad cell.

```
int set_bsd_pad_design
design_name
    -access access_list
    -type pad_type
    -differential differential
    -disable_res disable_result
    -lib_cell true | false
```
set_bsd_path
Specifies the order of the cells in the boundary scan register.

int set_bsd_path identifier_list

set_bsd_port
Identifies existing ANSI/IEEE Std. 1149.1 Test Access ports of the current design, for the check_bsd command.

int set_bsd_port port_type TAP_port

set_bsd_power_up_reset
Specifies and characterize the power up reset cell for the current design.

int set_bsd_power_up_reset
-cell_name cell_name
-reset_pin_name reset_pin_name
-active high | low
-delay power up reset delay

set_bsd_register
Declares a user-defined data register to be used for the boundary-scan insertion.

int set_bsd_register
register_identifier
-cell hierarchical_cell_name
-access access_list
[-style asynchronous | synchronous | global]

set_bsd_runbist
Specifies the parameters for the RUNBIST instruction.

int set_bsd_runbist
[-time runtime]
[-clock_cycles clock_port_integer_pairs]
[-signature pattern]

set_bsd_signal
Specifies a boundary scan signal type to be placed on a specified port in the current design.

int set_bsd_signal port_type port_name
set_bsd_tap_element
Characterizes a design cell as a boundary-scan tap controller to be used for boundary-scan insertion.

```
int set_bsd_tap_element
     -design design_name
     -access access_list
```

set_bsr_cell_type
Specifies the minimum acceptable boundary-scan cell implementation for a set of ports in the current design.

```
int set_bsr_cell_type
     -port port_list
     [-direction in | out]
     cell_type
```

set_case_analysis
Specifies that a port or pin is at a constant logic value 1 or 0.

```
string set_case_analysis
     value
     port_or_pin_list
```

set_cell_degradation
Sets the cell_degradation attribute to a specified value on specified ports or designs.

```
int set_cell_degradation
     cell_degradation_value
     object_list
```

set_cell_internal_power
Sets or removes the power_value attribute on or from specified pins; the value represents the power consumption for a single toggle of each pin.

```
int set_cell_internal_power
     pin_names
     [power_value [unit]]
```
**set_clock_gating_check**

Puts setup and hold checks on clock gating cells.

```plaintext
int set_clock_gating_check
[-setup setup_margin]
[-hold hold_margin]
[-rise]
[-fall]
[-high | -low] object_list
```

**set_clock_gating_registers**

Forces the enabling or disabling of clock gating for specified registers in the current design, overriding all conditions necessary for automatic RTL clock gating by the `insert_clock_gating` command.

```plaintext
int set_clock_gating_registers
[-include_instances register_list]
[-exclude_instances register_list]
[-undo register_list]
```

**set_clock_gating_signals**

Forces the enabling or disabling of clock gating for specified signals, overriding the conditions necessary for automatic RTL clock gating by `elaborate -gate_clock`.

```plaintext
int set_clock_gating_signals
[-design design_name]
[-include signal_list]
[-exclude signal_list]
```

**set_clock_gating_style**

Sets the clock gating style that HDL Compiler and the `insert_clock_gating` command use for clock gating.

```plaintext
int set_clock_gating_style
[-sequential_cell seq_cell]
[-minimum_bitwidth minsize_value]
[-setup setup_value]
[-hold hold_value]
[-positive_edge_logic gate_list]
[-negative_edge_logic gate_list]
[-control_point none | before | after]
[-control_signal scan_enable | test_mode]
[-observation_point true | false]
[-observation_logic_depth depth_value]
[-max_fanout max_fanout_count]
[-no_sharing]
```
set_clock_latency
Specifies clock network latency.

    string set_clock_latency
        [-rise]
        [-fall]
        [-min]
        [-max]
        [-source]
        [-early]
        [-late]
        delay
        object_list

set_clock_transition
Sets clock transition attributes on clock objects.

    int set_clock_transition
        transition
        [-rise | -fall]
        [-min]
        [-max]
        clock_list

set_clock_uncertainty
Specifies uncertainty (skew) of clock networks.

    string set_clock_uncertainty
        [-from from_clock]
        [-to to_clock]
        [-rise]
        [-fall]
        [-setup]
        [-hold] uncertainty
        [object_list]

set_combinational_type
Sets attributes on cell instances to specify which
combinational cells from the target library are to be
used by compile.

    int set_combinational_type
        -replacement_gate replacement_gate
        [cell_list]
set_common_resource
Specifies a group of operations to be scheduled on the same resources by Behavioral Compiler.

```
int set_common_resource
[-process process_name]
operation_names
[-min_count min_resources]
[-max_count max_resources]
[-force_sharing]
[-exclusive]
```

set_compare_design_script
A command to be added to the compare_design script, to be used during verification with the balance_buffer, compare_design, compile, insert_pads, reoptimize_design, and translate commands.

```
int set_compare_design_script
[-ignore endpoint_list]
[-only endpoint_list]
[-accept sub_design_list]
```

set_compile_partitions (dctcl-mode only)
Specifies the compile partitions for the current design. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
int set_compile_partitions
(-level level) | (-designs design_list) | -all | -auto
[-force]
[-no_reset]
```

set_connection_class
Sets the connection class value on ports.

```
int set_connection_class
connection_class_value
object_list
```
**set_context_margin**
Specifies the margin by which to tighten or relax constraints.

```
string set_context_margin [-percent] [-relax] [-min] [-max] value [object_list]
```

**set_cost_priority**
Sets the cost_priority attribute to a specified value on the current design.

```
```

**set_critical_range**
Sets the critical_range attribute to a specified value on a list of designs.

```
int set_critical_range range_value designs
```

**set_cycles**
Sets the number of cycles between two Behavioral Compiler operations and/or loop boundaries for a specified process or for all processes.

```
int set_cycles [-process process_name] cycle_offset -from_option start_operation -to_option end_operation
```
**set_datapath_optimization**
Sets the datapath_optimization attribute on arithmetic operations in the current design to prevent them from being transformed by the built-in datapath optimization feature in the compile command.

This command is not currently supported.

```
int set_datapath_optimization
object_list
[true | false]
```

**set_default_drive**
Sets the default driving strength for specified objects, to be used by Top-Down Environmental Propagation (TDEP).

```
int set_default_drive
[-min]
[-max]
[-rise]
[-fall]
[-none]
[resistance]
[cell_or_pin_list]
```

**set_default_driving_cell**
Sets the default driving cell for specified objects, to be used by Top-Down Environmental Propagation (TDEP).

```
int set_default_driving_cell
[-lib_cell lib_cell_name]
[-library lib]
[-rise]
[-fall]
[-pin pin_name]
[-from_pin from_pin_name]
[-dont_scale]
[-no_design_rule]
[-multiply_by factor]
[-none]
[cell_or_pin_list]
```
set_default_fanout_load
Sets the default fanout load to be used by Top-Down Environmental Propagation (TDEP).

```
int set_default_fanout_load
[-none]
[fanout_load_value]
[cell_or_pin_list]
```

set_default_input_delay
Sets the value of the input delay as a percentage of the clock period to be assigned during environment propagation.

```
int set_default_input_delay
[-none]
percent_delay
[cell_or_pin_list]
```

set_default_load
Sets the default load to be used by Top-Down Environmental Propagation (TDEP).

```
int set_default_load
[-min]
[-max]
[-pin_load]
[-wire_load]
[-none]
[value]
[cell_or_pin_list]
```

set_default_output_delay
Sets the output delay as a percentage of the clock period to be assigned during environment propagation.

```
int set_default_output_delay
[-none]
percent_delay
[cell_or_pin_list]
```
set_design_license

Adds license information to the current design. The set_design_license command can be used to require a license before a design can be read in.

```plaintext
int set_design_license
[-dont_show references]
[-quiet]
[-limited limited_keys]
regular_keys
```

set_dft_configuration

Sets the DFT configuration for the current design.

```plaintext
int set_dft_configuration
[-order list_of_clients]
```

set_dft_signal

Specifies DFT signals for insert_dft to use for implementing test point signals of a specified signal type.

```plaintext
int set_dft_signal
dft_signal_type
-port port_list
[-hookup pin [-sense sense]]
```

set_direct_power_rail_tie

Sets the direct_power_rail_tie attribute on library pins.

```plaintext
int set_direct_power_rail_tie
lib_pin_list
[true | false]
```

set_disable_clock_gating_check

Disables the clock gating check for specified objects in the current design.

```plaintext
string set_disable_clock_gating_check
object_list
```
set_disable_timing
Disables timing arcs in the current design.

int set_disable_timing
object_list
[-from from_pin_name -to to_pin_name]
[-restore]

set_dont_touch
Sets the dont_touch attribute on cells, nets, references,
and designs in the current design, and on library cells,
to prevent these objects from being modified or
replaced during optimization.

int set_dont_touch object_list
[true | false]

set_dont_touch_network
Sets the dont_touch_network attribute on clocks, pins,
or ports in the current design to prevent cells and nets
in the transitive fanout of the set_dont_touch_network
objects from being modified or replaced during
optimization.

int set_dont_touch_network object_list

set_dont_use
Sets the dont_use attribute on library cells to exclude
them from the target library during optimization.

int set_dont_use object_list

set_drive
Sets the rise_drive or fall_drive attributes to specified
resistance values on specified input and inout ports.

int set_drive
resistance
[-rise]
[-fall]
[-min]
[-max]
port_list
set_driving_cell
Sets attributes on input or inout ports of the current design, specifying that a library cell or pin will drive the ports.

```plaintext
int set_driving_cell
[-lib_cell lib_cell_name]
[-library lib]
[-rise]
[-fall]
[-pin pin_name]
[-from_pin from_pin_name]
[-dont_scale]
[-no_design_rule]
[-input_transition_rise rtran]
[-input_transition_fall ftran]
[-multiply_by factor]
port_list
```

set_electromigration_drc
Sets or resets the electromigration DRC constraint for the current design by setting the electromigration_drc attribute.

```plaintext
int set_electromigration_drc
on | off
[-use_switching_activity]
```

set_equal
Defines two input ports as logically equivalent.

```plaintext
int set_equal
port1
port2
```

set_exclusive_use
Maps a specified HDL variable to a unique Behavioral Compiler register for a specified process or for all processes.

```plaintext
int set_exclusive_use
[-process process_name]
variable_name
[-shared]
```
**set_false_path**

Removes timing constraints from particular paths.

```
int set_false_path
[-rise | -fall]
[-setup | -hold]
[-from from_list]
[-through through_list]
[-to to_list]
[-reset_path]
```

**set_fanout_load**

Sets the fanout_load attribute to a specified value on specified output ports of the current design.

```
int set_fanout_load value port_list
```

**set_fix_hold**

Sets a fix_hold attribute on clocks in the current design.

```
int set_fix_hold clock_list
```

**set_fix_multiple_port_nets**

Sets the fix_multiple_port_nets attribute to a specified value on the current design.

```
int set_fix_multiple_port_nets
-default | -all | [-feedthroughs]
[-outputs]
[-constants]
[-buffer_constants]
```

**set_flatten**

Sets or removes the flatten attribute on specified designs or on the current design, to enable or disable the flattening optimization step during compile.

```
int set_flatten
[true | false]
[-effort low | medium | high]
[-minimize single_output | multiple_output | none]
[-phase true | false]
[-design design_list]
[-quiet]
```
set_fpga

Configures the current design for behavioral synthesis targeting an FPGA.

```c
int set_fpga
[-target FPGA_target]
[-device FPGA_device]
[-speed FPGA_speed_grade]
[-module]
[-list]
```

set_fsm_encoding

Specifies the bit encodings for states in the current design.

```c
int set_fsm_encoding encoding_list
```

set_fsm_encoding_style

Defines the encoding style for assigning unencoded states.

```c
int set_fsm_encoding_style
one_hot | binary | gray | auto
```

set_fsm_minimize

Determines whether or not state minimization is to be performed on the state machine design during compile.

```c
int set_fsm_minimize true | false
```

set_fsm_order

Sets the ordering of states in a state machine design.

```c
int set_fsm_order state_list
```

set_fsm_preserve_state

Specifies states to be preserved during state minimization.

```c
int set_fsm_preserve_state state_list
```

set_fsm_state_vector

Specifies the instance names for flip-flops used to implement the state vector.

```c
int set_fsm_state_vector vector_list
```
set_ideal_latency
Specifies ideal network latency.

string set_ideal_latency
[-rise | -fall]
[-min]
[-max]
delay
object_list

set_ideal_net
Sets the ideal_net attribute on specified individual
nets in the current design.

int set_ideal_net net_list

set_ideal_network
Marks a set of ports or pins in the current design as
sources of an ideal network. This disables timing
update and optimization of cells and nets in the
transitive fanout of the specified objects.

int set_ideal_network
[-dont_care_placement]
object_list

set_ideal_transition
Specifies ideal transition for the ideal network and
ideal nets.

string set_ideal_transition
[-rise | -fall]
[-min]
[-max]
transition_time
object_list

set_impl_priority
Sets the formula attribute of the priority parameter
and/or the set_id attribute for implementations in
synthetic libraries.

int set_impl_priority
[-priority formula]
[-set_id id]
implementation_list
set_implementation
  Specifies the implementation to use for synthetic library cell instances in a design.

  int set_implementation
  implementation_name
  cell_list
  [-check_impl]

set_input_delay
  Sets input delay on pins or input ports relative to a clock signal.

  int set_input_delay
  delay_value
  [-clock clock]
  [-clock_fall]
  [-level_sensitive]
  [-network_latency_included]
  [-source_latency_included]
  [-rise]
  [-fall]
  [-max]
  [-min]
  [-add_delay]
  port_pin_list

set_input_transition
  Sets the max_transition_rise, max_transition_fall, min_transition_rise, or min_transition_fall attributes to the specified transition values on the specified input and inout ports.

  int set_input_transition
  transition
  [-rise]
  [-fall]
  [-min]
  [-max]
  port_list
set_isolate_ports
Specifies the ports that are to be isolated from internal fanouts of their driver nets.

int set_isolate_ports
[-type buffer]
[-type inverter]
[-driver cell_name]
[-force]
port_list

set_isolation_operations
Specifies a group of operations for operand isolation by Behavioral Compiler.

int set_isolation_operations
[-process process_name]
[-exclude_operations operation_names]
[-include_operations operation_names]

set_jtag_implementation (dctcl-mode only)
Built-in Tcl command.

set_jtag_instruction (dctcl-mode only)
Built-in Tcl command.

set_jtag_manufacturer_id (dctcl-mode only)
Built-in Tcl command.

set_jtag_part_number (dctcl-mode only)
Built-in Tcl command.

set_jtag_port (dctcl-mode only)
Built-in Tcl command.

set_jtag_port_mode (dctcl-mode only)
Built-in Tcl command.

set_jtag_port_routing_order (dctcl-mode only)
Built-in Tcl command.

set_jtag_port_type (dctcl-mode only)
Built-in Tcl command.
set_jtag_version_number (dctcl-mode only)
  Built-in Tcl command.

set_layer
  Defines features of a schematic layer.
  
  int set_layer
  layer_name
  attribute
  value

set_libcell_dimensions
  Sets the width and height of a library cell.
  
  int set_libcell_dimensions
  -cell cell_name
  -width width
  -height height

set_libpin_location
  Sets the location of a pin of a library cell relative to
  the origin of the library cell.
  
  int set_libpin_location
  -cell library_cell_name
  -pin pin_name_of_the_library_cell
  -coordinate {x_coordinate y_coordinate}

set_load
  Sets the load attribute to a specified value on specified
  ports and nets.
  
  int set_load
  value
  objects
  [-subtract_pin_load]
  [-min]
  [-max]
  [][-pin_load]
  [-wire_load]]

set_local_link_library
  Sets the local_link_library attribute to specified files
  and libraries on the current design.
  
  int set_local_link_library
  local_link_library
set_logic_dc
    Specifies one or more input ports in the current design that are to be driven by don't care. The set_logic_one and set_logic_zero commands are used the same way as this command.

    int set_logic_dc port_list

set_logic_one
    Specifies one or more input ports in the current design that are to be driven by logic 1. The set_logic_zero and set_logic_dc commands are used the same way as this command.

    int set_logic_one port_list

set_logic_zero
    Specifies one or more input ports in the current design that are to be driven by logic 0. The set_logic_one and set_logic_dc commands are used the same way as this command.

    int set_logic_zero port_list

set_map_only
    Sets the map_only attribute on specified objects so that they can be excluded from logic-level optimization during compile.

    int set_map_only
    object_list
    flag

set_max_area
    Sets the max_area attribute to a specified value on the current design.

    int set_max_area
    [-ignore_tns]
    area_value
set_max_capacitance
Sets the max_capacitance attribute to a specified value
on the specified ports and designs.

```c
int set_max_capacitance
    capacitance_value
    object_list
```

set_max_cycles
Sets the maximum number of cycles between two
Behavioral Compiler operations and loop boundaries
for a specified process or for all processes.

```c
int set_max_cycles
    [-process process_name]
    cycle_offset
    -from_option start_operation
    -to_option end_operation
```

set_max_delay
Specifies a maximum delay target for paths in the
current design.

```c
int set_max_delay
    delay_value
    [-rise | -fall]
    [-from from_list]
    [-through through_list]
    [-to to_list]
    [-group_path group_name]
    [-reset_path]
```

set_max_dynamic_power
Sets the target dynamic power for the current design
by setting the max_dynamic_power attribute to a
specified value.

```c
int set_max_dynamic_power
    dynamic_power
    [GW | MW | KW | W | mW | uW | nW | pW | fW | aW]
```
**set_max_fanout**

Sets the max_fanout attribute to a specified value on specified input ports and/or designs.

```c
int set_max_fanout
fanout_value
object_list
```

**set_max_leakage_power**

Sets the target leakage power for the current design by setting the max_leakage_power attribute to a specified value.

```c
int set_max_leakage_power leakage_power
[GW | MW | KW | W | mW | uW | nW | pW | fW | aW]
```

**set_max_peak_noise**

Sets the max_peak_noise attribute to a specified value on specified ports or designs.

```c
int set_max_peak_noise
noise_value
object_list
```

**set_max_time_borrow**

Sets the max_time_borrow attribute to a specified value on clocks, latch cells, data pins, or clock (enable) pins, to constrain the amount of time borrowing possible for level-sensitive latches.

```c
int set_max_time_borrow
delay_value
object_list
```

**set_max_toggle_rate**

Sets or resets the max_toggle_rate attribute on designs, cells, nets, pins, and ports of the current design.

```c
int set_max_toggle_rate
object_list
[value max_tr_value]
[=clock clock_name]
set_max_transition
Sets the max_transition attribute to a specified value on specified ports or designs.

    int set_max_transition
    transition_value
    object_list

set_memory_input_delay
Sets the input delay on a memory to be used by Behavioral Compiler.

    int set_memory_input_delay
    [delay_value]
    [-external ext_delay_value]
    [-name mem_name]

set_memory_output_delay
Sets the output delay on a memory to be used by Behavioral Compiler, and enables operation chaining on the outputs of the memory.

    int set_memory_output_delay
    delay_value
    [-external ext_delay_value]
    [-name mem_name]

set_min_capacitance
Sets the min_capacitance attribute to a specified value on specified input ports in the current design.

    int set_min_capacitance
    capacitance_value
    object_list

set_min_cycles
Sets the minimum number of cycles between two Behavioral Compiler operations and loop boundaries for a specified process or for all processes.

    int set_min_cycles
    [-process process_name]
    cycle_offset
    -from_option start_operation
    -to_option end_operation
**set_min_delay**
Specifies a minimum delay target for paths in the current design.

```
int set_min_delay
delay_value
[-rise | -fall]
[-from from_list]
[-through through_list]
[-to to_list]
[-reset_path]
```

**set_min_fault_coverage** (dctcl-mode only)
Built-in Tcl command.

**set_min_library**
Sets an alternate library to use for minimum delay analysis.

```
int set_min_library
max_library
-min_version
min_library | -none
```

**set_min_porosity**
Sets the minimum_porosity attribute on specified designs or on the current design.

```
int set_min_porosity
porosity_value
[design_list]
```

**set_minimize_tree_delay**
Sets the minimize_tree_delay attribute on a design or designs, thus determining whether an arithmetic expression tree will be restructured to minimize delay during compile. By default, all expression trees are candidates for tree height minimization if timing constraints are specified.

```
int set_minimize_tree_delay
[true | false]
[design design_list]
```
set_mode

Selects the mode of a component.

```c
int set_mode
[mode_list]
[instance_list]
```

set_model_drive

Sets the model_drive attribute to a specified value on specified input or inout ports to set their drive values during synthetic library modeling.

```c
int set_model_drive
drive_value
port_list
```

set_model_load

Sets the model_load attribute to a specified value on specified ports to set their load values during synthetic library modeling.

```c
int set_model_load
load_value
port_list
```

set_model_map_effort

Sets the model_map_effort attribute to a specified value on the current design, to specify the relative amount of CPU time to use during synthetic library modeling.

```c
int set_model_map_effort low | medium | high
```

set_model_scale

Sets the model_scale attribute to a specified value on the current design, to use as a scale factor in calculating timing constraints during synthetic library modeling.

```c
int set_model_scale scale
```
set_multibit_options

Sets the multibit_mode and minimum_multibit_width attributes to specified values on the current design.

```plaintext
int set_multibit_options
[-default]
[-mode multibit_mode]
[-minimum_width width]
```

set_multicycle_path

Modifies the single-cycle timing relationship of a constrained path.

```plaintext
int set_multicycle_path
path_multiplier
[-rise | -fall]
[-setup | -hold]
[-start | -end]
[-from from_list]
[-to to_list]
[-through through_list]
[-reset_path]
```

set_operand_isolation_cell

Specifies a list of GTECH cells to be operand isolation candidates.

```plaintext
int set_operand_isolation_cell
[object_list]
```

set_operand_isolation_slack

Sets the timing threshold below which the automatic isolation roll back operation is not triggered.

```plaintext
int set_operand_isolation_slack slack_number
```

set_operand_isolation_style

Sets the operand isolation style that Power Compiler uses for operand isolation.

```plaintext
int set_operand_isolation_style
[-logic logic_style]
```
set_operating_conditions
Defines the operating conditions for the current design.

```
int set_operating_conditions
[-min min_condition]
[-max max_condition]
[-min_library min_lib]
[-max_library max_lib]
[-min_phys min_proc]
[-max_phys max_proc]
[-library lib]
[condition]
```

set_opposite
Defines two input ports as logically opposite.

```
int set_opposite
port1
port2
```

set_optimize_registers
Sets the optimize_registers attribute on the specified design or on the current design, so that compile automatically invokes the DC-Ultra optimize_registers command to retime the design during optimization.

```
int set_optimize_registers
[true | false]
[-design design_list]
```

set_output_delay
Sets output delay on pins or output ports relative to a clock signal.

```
int set_output_delay
delay_value
[-clock clock
[-clock_fall] ]
[[-level_sensitive]]
[-network_latency_included]
[-source_latency_included]
[-rise]
[-fall]
[-max]
[-min]
[-add_delay]
[[-group_path group_name] port_pin_list]
```
**set_pad_type**

Indicates the type of I/O pads needed on given ports.

```plaintext
int set_pad_type
[-example example_pad]
[-exact exact_pad]
[-schmitt | -hysteresis]
[-pullup]
[-pulldown]
[-opendrain]
[-opensource]
[-vil vil]
[-vih vih]
[-vol vol]
[-voh voh]
[-vimin vimin]
[-vimax vimax]
[-vomin vomin]
[-vomax vomax]
[-currentlevel currentlevel]
[-clock]
[-no_clock]
[-slewrate slewratevalue]
design_or_port_list
```

**set_pipeline_stages**

Sets directives to control the implementation of DW03_mult_n_stage operators referenced by the specified cells.

```plaintext
int set_pipeline_stages
[cell_list]
[-min min_stages]
[-fixed fixed_stages]
[-auto]
```

**set_port_configuration**

Provides the Shadow LogicDFT utility with information about the input and output ports of the specified elements that are to receive wrappers.

```plaintext
int set_port_configuration
-cell cell_design_ref_list
-port port_name
[-tristate]
[-wrapper_exclude]
[-clock clock_name]
[-read signal_value_pin_pairs]
[-write signal_value_pin_pairs]
```
set_port_fanout_number
Sets the number of external fanout points driven by
specified ports in the current design.

```
int set_port_fanout_number
fanout_number port_list
```

set_port_is_pad
Sets the port_is_pad attribute on specified ports
and/or designs, to indicate that those ports are to have
I/O pads attached.

```
int set_port_is_pad [port_design_list]
```

set_port_location
Annotates the specified top-level port with X, Y
coordinates and layer geometry, to be used by
Floorplan Manager and Physical Compiler during
execution of reoptimize_design.

```
int set_port_location
[-coordinate {x_coordinate y_coordinate}]
[-layer_name {layer_name}]
[use only in Physical Compiler]
[-layer_area {lower_x_coordinate
lower_y_coordinate
upper_x_coordinate upper_y_coordinate}]
[use only in Physical Compiler]
```

set_prefer
Sets the preferred attribute on specified library gates.

```
int set_prefer
[-min]
gate_list
```

set_propagated_clock
Specifies propagated clock latency.

```
string set_propagated_clock object_list
```
set_register_type
Sets the latch_type or flip_flop_type attributes on designs or cell instances, to specify which sequential cells from the target library are to be used by compile.

```plaintext
int set_register_type
    [[-exact] -latch example_latch]
    [[-exact] -flip_flop example_flip_flop]
    [cell_or_design_list]
```

set_resistance
Sets the resistance value on nets.

```plaintext
int set_resistance
    value
    [-min]
    [-max]
    net_list
```

set_resource_allocation
Sets the resource_allocation attribute on the current design, thus specifying the type of resource allocation to be used by compile.

```plaintext
int set_resource_allocation
    none | area_only | area_no_tree_balancing | constraint_driven
```

set_resource_implementation
Sets the resource_implementation attribute on the current design, thus specifying the type of resource implementation to be used by compile.

```plaintext
int set_resource_implementation
    area_only | constraint_driven | use_fastest
```

set_rtl_load
Sets an RTL load value for capacitance and resistance on pins, ports and nets.

```plaintext
int set_rtl_load
    -cap cvalue
    -res rvalue
    [-min]
    [-max]
    pin_net_list
```
set_scan_bidi
Determine whether insert_scan and insert_dft configure specified bidirectional ports as inputs or outputs, or leave them untouched, during scan shift.

```c
int set_scan_bidi
  bidir_mode
  -port port_list
```

set_scan_configuration
Specifies the scan chain design.

```c
int set_scan_configuration
  [-add_lockup true | false]
  [-area_critical false]
  [-bidir_mode input | output | no_disabling ]
  [-chain_count integer_or_default]
  [-longest_chain_length integer_or_default]
  [-clock_gating entire_design | leaf_cell | superbuffer]
  [-clock_mixing no_mix | mix_edges | mix_clocks | mix_clocks_not_edges]
  [-create_test_clocks_by_system_clock_domain
    true | false ]
  [-dedicated_scan_ports true | false]
  [-disable true | false]
  [-existing_scan true | false]
  [-external_tristates disable_all | enable_one | no_disabling]
  [-hierarchical_isolation true | false]
  [-internal_clocks true | false ]
  [-internal_tristates disable_all | enable_one | no_disabling]
  [-insert_end_of_chain_lockup_latch true | false]
  [-methodology full_scan | partial_scan | none]
  [-multibit_segments true | false]
  [-physical true | false]
  [-prfile report_file_name]
  [-prtool cadence | avant]
  [-rebalance true | false]
  [-replace true | false]
  [-route true | false]
  [-route_signals all | global | serial | clocks | scan_enables]
  [-style multiplexed_flip_flop | clocked_scan
    | lssd | aux_clock_lssd | combinational | none]
set_scan_element
Sets the scan_element attribute on specified design objects, to determine whether or not insert_scan replaces them with scan cells.

int set_scan_element
ture | false
cell_design_ref_list
-multibit multi-bit_list

set_scan_link
Declares a scan link for the current design.

int set_scan_link
scan_link_name wire | scan_out_lockup

set_scan_path
Specifies a scan chain for the current design.

int set_scan_path
scan_chain_name
[ordered_list]
[dedicated_scan_out true | false]
[complete true | false]
[chain_length integer_or_default]
[clock clock_name]

set_scan_register_type
Specifies a list of scan sequential cells from the target library that are to be used by insert_scan or compile -scan when scan replacing designs or cell instances.

int set_scan_register_type
[-exact]
-type example_scan_seq_cell_list
[cell_or_design_list]

set_scan_segment
Identifies existing logic in the current design that is to be designated a scan segment.

int set_scan_segment
scan_segment_name
[-access signal_type_pin_pairs]
[contains member_list]
[-synthesizable true | false | default]
[reverse_order true | false]
set_scan_signal
Specifies one or more scan signals for the current design.

int set_scan_signal
scan_signal_type
   -port port_list
   [-hookup pad_instance_name/port [-sense sense]]
   [-chain chain_list]

set_scan_state
Sets the scan state status for the current db design.

int set_scan_state
test_ready | scan_existing

set_scan_style (dctcl-mode only)
Built-in Tcl command.

set_scan_transparent
Sets the scan_latch_transparent attribute on specified design objects, to determine whether or not level-sensitive sequential cells are modeled as transparent latches during automatic test pattern generation (ATPG).

int set_scan_transparent
ture | false
cell_design_ref_list
   -multibit multi-bit_list
   -existing

set_scan_tristate
For specified tristate nets, determines whether insert_scan and insert_dft disable all drivers, enable exactly one driver, or leave the nets untouched, during scan shift.

int set_scan_tristate
disabling_option
   -net net_list
set_share_cse
Sets the share_cse attribute, which determines whether common subexpressions are shared during compile. By default, all common subexpressions are shared unless otherwise specified.

```c
int set_share_cse
  [true | false]
  [-design design_list]
```

set_signal_type
Sets the signal type on a list of pins or ports.

```c
int set_signal_type
  signal_type port_list
  [-associated_clock clk]
  [-index signal_index]
```

set_simple_compile_mode
Places Design Compiler into simple compile mode.

```c
int set_simple_compile_mode
  [true | false]
  [-verbose]
  [-budget]
```

set_state_for_retiming
Sets the state_for_retiming attribute on cells in the current design. This command can effect both hierarchical cells and sequential leaf cells.

```c
int set_state_for_retiming
  cell_list preserve | dont_care
```

set_structure
Sets various structure attributes on a design or on a list of designs, to determine whether and how the designs are structured during compile.

```c
int set_structure
  [true | false]
  [-design design_list]
  [-boolean true | false]
  [-boolean_effort low | medium | high]
  [-timing true | false]
```
set_switching_activity
Sets (or resets) the toggle_rate and static_probability values for nets, pins, and ports of the current design.

```
int set_switching_activity
[-static_probability sp_value]
[-toggle_rate tr_value]
[-state_dep boolean_eq_of_pins]
[-path_dep sources_of_path]
[-transition_type rising | falling]
[-period period_value | -clock clock_name]
object_list
```

set_synlib_dont_get_license
Specifies a list of synthetic library part licenses that are not automatically checked out.

```
int set_synlib_dont_get_license
license_list
```

set_test_assume
Sets the test_assume attribute to a logic value to be assumed on specified cell output pins throughout test design rule checking.

```
int set_test_assume
zero_or_one_value
pin_list
```

set_test_dont_fault (dctcl-mode only)
Built-in Tcl command.

set_test_hold
Sets the test_hold attribute to a logic value to be assumed on specified input ports during testing.

```
int set_test_hold
zero_or_one_value
port_list
```

set_test_initial
Sets the test_initial attribute to a logic value to be assumed on specified cell output pins at the start of test design rule checking and fault simulation.

```
int set_test_initial
zero_or_one_value
pin_list
```
**set_test_isolate**
Sets the test_isolate attribute on the specified cells, pins or ports, indicating that they are to be logically isolated and considered untestable during test design rule checking.

```tcl
int set_test_isolate pin_cell_port_list
```

**set_test_mask_fault** (dctcl-mode only)
Built-in Tcl command.

**set_test_methodology** (dctcl-mode only)
Built-in Tcl command.

**set_test_require** (dctcl-mode only)
Built-in Tcl command.

**set_test_signal**
Specifies test-mode signals for the current design.

```tcl
int set_test_signal
  test_signal_type
  -port port_name
```

**set_test_target**
Define the core wrapper modes for a specific mode in the test schedule.

```tcl
int set_test_target
[-purpose core_internal_test | top_internal_test | mission_mode]
[-cores core_list]
[-test_mode mode_name]
```

**set_test_unmask_fault** (dctcl-mode only)
Built-in Tcl command.

**set_timing_ranges**
Sets timing ranges for the current design.

```tcl
int set_timing_ranges
[timing_ranges]
[-library library_name]
**set_transform_for_retiming**

Sets the transform_for_retiming attribute on cells in the current design. This can effect both hierarchical cells and sequential leaf cells.

```c
int set_transform_for_retiming
   cell_list
   multiclass | decompose | dont_retime
```

**set_true_delay_case_analysis**

Sets the true_delay_case_analysis attribute, which specifies the input vector value to use for specified pins or ports of the current design for the -true and -justify options of report_timing.

```c
int set_true_delay_case_analysis
   0 | 1 | r | f | none port_pin_list
```

**set_ultra_optimization**

Sets the DC Ultra optimization mode and checks out the DC Ultra license, if available, for the current Design Compiler session.

```c
int set_ultra_optimization
   [true | false]
   [-force]
```

**set_unconnected**

Lists output ports to be unconnected.

```c
int set_unconnected port_list
```

**set_ungroup**

Sets the ungroup attribute on specified designs, cells, or references, indicating that they are to be ungrouped during compile.

```c
int set_ungroup
   object_list
   true | false
```

**set_unix_variable**

Sets the value of a UNIX environment variable.

```c
string set_unix_variable
   variable_name
   new_value
```
set_user_budget
Sets user budgets or budget ratios.

string set_user_budget
-from object_list
-to object_list
[-percent]
value

set_wire_load_min_block_size
Sets the wire load min_block_size attribute on the current design.

int set_wire_load_min_block_size size

set_wire_load_mode
Sets the wire_load_model_mode attribute on the current design, specifying how wire load models are to be used to calculate wire capacitance in nets.

int set_wire_load_mode mode_name

set_wire_load_model
Set the wire_load_attach_name attribute on designs, ports, hierarchical cells of current design, or the specified cluster of the current design, for selecting a wire load model to use in calculating wire capacitance.

int set_wire_load_model
-name model_name
[-library lib]
[-min]
[-max]
[-cluster cluster_name]
[object_list]
set_wire_load_selection_group
Specify a selection group to use for determining a wire load model to be assigned to designs and cells or to a specified cluster. This command is supported only for the enclosed wire load mode.

```c
int set_wire_load_selection_group
[-library lib]
[-min]
[-max]
[-cluster cluster_name]
group_name
[object_list]
```

set_wired_logic_disable
Sets the wired_logic_disable attribute on the specified ECL designs, to indicate whether or not the creation of wired OR logic is to be disabled when optimizing the designs using compile.

```c
int set_wired_logic_disable
object_list
[true | false]
```

set_wrapper_element
Sets the wrapper_element attribute on a list of elements around which preview_dft and insert_dft are to insert a wrapper when the Shadow LogicDFT utility is enabled.

```c
int set_wrapper_element
cell_design_ref_list
-type wrapper_type
```

setenv (dctcl-mode only)
Sets the value of a system environment variable.

```c
string setenv variable_name new_value
```

sh
Sends a command to the UNIX operating system.

```c
int sh command
```
simplify_constants
Propagates constants and other information in the current design.

```
int simplify_constants
[-verify]
[-verify_hierarchically]
[-verify_effort low | medium | high]
[-boundary_optimization]
```

sizeof_collection (dctcl-mode only)
Returns the number of objects in a collection.

```
int sizeof_collection collection1
```

socket (dctcl-mode only)
Built-in Tcl command.

sort_collection (dctcl-mode only)
Sorts a collection based on one or more attributes, resulting in a new, sorted collection. The sort is ascending by default.

```
collection sort_collection
[-descending]
collection1
criteria
```

source (dctcl-mode only)
Read a file and evaluate it as a Tcl script.

```
string source
[-echo]
[-verbose] file
```

split (dctcl-mode only)
Built-in Tcl command.

string (dctcl-mode only)
Built-in Tcl command.
**sub_designs_of (dctcl-mode only)**

Gets the subdesigns according to the options. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
collection sub_designs_of
    [-hierarchy]
    [-in_partition | -partition_only]
    [-dt_only | -ndt_only]
    [-multiple_instances | -single_instances]
    [-names_only]
    design
```

**sub_instances_of (dctcl-mode only)**

Gets the subinstances according to the options. For use in dc_shell-t (Tcl mode of dc_shell) only.

```
collection sub_instances_of
    [-hierarchy]
    [-in_partition]
    [-partition_only]
    [-dt_only]
    [-ndt_only]
    [-of_references reference_list]
    [-master_instance]
    [-names_only]
    design
```

**subst (dctcl-mode only)**

Built-in Tcl command.

**suppress_message (dctcl-mode only)**

Disables printing of one or more informational or warning messages.

```
string suppress_message [message_list]
```

**switch (dctcl-mode only)**

Built-in Tcl command.

**syntax_check**

Enables or disables Syntax Checker syntax_check mode, which checks commands for syntax errors without executing them.

```
int syntax_check true | false
```
**tclLog** (dctcl-mode only)

Built-in Tcl command.

**tell** (dctcl-mode only)

Built-in Tcl command.

**time** (dctcl-mode only)

Built-in Tcl command.

**trace** (dctcl-mode only)

Built-in Tcl command.

**trace_nets**

Enables global net tracing during check_test on the specified nets in the current design.

```tcl
int trace_nets hierarchical_net_list
```

**transform_csa**

Transforms arithmetic operators (for example, addition, subtraction, and multiplication) into the carry save adder (CSA) operator, for the specified design or for the current design.

```tcl
int transform_csa
[design_name]
[-labelled_only]
[-decompose_multiply all | none | constant]
[-duplicate]
[-area]
[-group]
[-dont_split]
```

**translate**

Translates a design from one technology to another.

```tcl
int translate
[-verify]
[-verify_hierarchically]
[-verify_effort low | medium | high]
[-preserve_structure]
```
unalias

Removes alias definitions.

```
list unalias [-all] [alias_list]
```

ungroup

Removes a level of hierarchy.

```
int ungroup
    cell_list | -all
    [-prefix prefix_name]
    [-flatten]
    [-simple_names]
    [-soft]
    [-small n]
    [-force]
```

uniquify

Removes multiply-instantiated hierarchy in the current design by creating a unique design for each cell instance.

```
int uniquify
    [-force]
    [-base_name base_name]
    [-cell cell_list]
    [-reference design_name]
    [-new_name new_design_name]
```

unschedule

Permits Behavioral Compiler to reschedule I/O operations into cycles different from those defined in the original HDL description.

```
int unschedule operation_names
```

unset (dctcl-mode only)

Built-in Tcl command.

unset_fpga

Removes the behavioral synthesis FPGA configuration on the current design. The design is restored to targeting an ASIC implementation.

```
int unset_fpga
```
unsuppress_message  (dctcl-mode only)
   Enables printing of one or more suppressed informational or suppressed warning messages.
   string unsuppress_message [messages]

untrace_nets
   Disables global net tracing during check_test on any specified nets for which net tracing had been previously enabled by trace_nets.
   int untrace_nets
   hierarchical_net_list | -all

update  (dctcl-mode only)
   Built-in Tcl command.

update_bounds
   Updates an existing bound by adding or removing objects. The bound should be of type movebound.
   int update_clusters cell_list

update_clusters
   Updates the clusters associated with the current design to reflect the changes made to a subdesign.
   int update_bounds
   [-name bound_name]
   [-bound bound_object]
   [-add]
   [-remove]cell_list
   cell_list

update_lib
   Reads in a specified library file and uses it to update an existing technology, synthetic, or symbol library.
   int update_lib
   [-overwrite]
   [-permanent]
   library_name
   file_name
   [-no_warnings]
update_script
   Modifies an old dc_shell script to use current dc_shell commands.

   int update_script
   [-from_version version]
   script_file_name
   [-output_file output_file_name]

update_timing
   Updates timing information on the current design.

   int update_timing

uplevel (dctcl-mode only)
   Built-in Tcl command.

upvar (dctcl-mode only)
   Built-in Tcl command.

variable (dctcl-mode only)
   Built-in Tcl command.

vwait (dctcl-mode only)
   Built-in Tcl command.

which
   Displays the pathname of one or more files, in dc_shell or in dc_shell-t (Tcl mode of dc_shell).

   list which file_names

while
   Loop execution control structure.

   while ( expression ) {
      loop-statement-block
   }
write

Writes a design netlist or schematic from dc_shell to a file.

int write
[-format output_format]
[-hierarchy]
[-no_implicit]
[-modified]
[-output output_file_name]
[-library library_name]
[design_list]
[-names_file name_mapping_files]
[-donot_expand_dw]

write_bsd_protocol

Writes a boundary-scan protocol file.

int write_bsd_protocol
[-out protocol_file]
[-format tpf | stil]

write_bsd

Generates the boundary-scan description language (BSDL) file for a boundary-scan design.

int write_bsd
[-naming_check VHDL | BSDL | none]
[-output file_name]
[-effort low | medium | high]

write_clusters

Writes to a file in Physical Design Exchange Format (PDEF) the physical cluster annotations associated with a design.

int write_clusters
[-design design_name]
[-output new_cluster_file_name]
[-no_attributes]
[-hier_cells]
[-new_cells_only original_cluster_file_name]
write_compare_design_script

Saves the compare_design script, which contains dc_shell commands to be used during verification with balance_buffer, compare_design, compile, insert_pads, reoptimize_design, and translate.

int write_compare_design_script

write_compile_script (dctcl-mode only)

Writes a compile script for the specified design. For use in dc_shell-t (Tcl mode of dc_shell) only.

write_compile_script
[-absolute_paths]
[-hierarchy]
[-format dcsh | dctcl]
[-no_reports]
[-refining]
-destination pass
[-update design_list]
[design]

write_constraints

Writes constraints for the place and route tools.

int write_constraints
[-output file_name]
[-format synopsys | sdf | sdf-v2.1]
[-max_paths max_path_number]
[-nworst nworst_number]
[-max_path_slack slack_value]
[-cover_design | -cover_nets]
[-net_priorities]
[-min_net_priority min_priority_number]
[-max_net_priority max_priority_number]
[-low_priorities]
[-max_path_timing]
[-net_timing]
[-load_delay net | cell]
[-net_capacitance]
[-subtract_pin_cap]
[-cell_groups]
[-hierarchy]
[-by_input_pin_name]
[-by_output_pin_name]
[-max_nets max_net_number]
[-from start_point_list]
[-to end_point_list]
[-through through_point_list]
write_design_lib_paths
Writes to a file the paths to which design libraries are mapped.

```
int write_design_lib_paths
[-filename file_name]
[-dc_setup]
```

write_designlist (dcsh-mode only)
Writes a list of designs referenced by the specified design or by the current design.

```
int write_designlist
[-output listfile]
[design]
```

write_environment
Writes the variable settings and constraints for the specified cells or designs.

```
write_environment
[-cells cell_list | -designs design_list]
[-format dcsh | dctcl]
[-output file_name]
[-suffix suffix]
[-environment_only]
[-constraints_only]
[-no_lib_info]
```

write_file (dctcl-mode only)
Built-in Tcl command.

write_ibm_constraints
Writes Synopsys Design Constraints (SDC) in a neutral format for import into the IBM EinsTimer static timing analysis tool.

```
int write_ibm_constraints
[-hierarchy]
[-full_path_lib_names]
[-output output-file-name]
[-ignored_file ignored_commands-file-name]
```
write_layout_scan

Writes scan chain information for performing scan chain reordering using third-party place and route tools.

This command is part of Scan Planner, which is available with a separate license.

int write_layout_scan
[-output output_command_file]
[-noclockdomain]

write_lib

Writes a compiled library to disk in Synopsys database, EDIF, or VHDL format.

int write_lib library_name
[-format db | edif | vhdl]
[-output file_name]
[-names_file file_list]
[-macro_only]

write_makefile (dctcl-mode only)

Writes a makefile that defines the dependencies and commands required to compile the specified design. For use in dc_shell-t (Tcl mode of dc_shell) only.

write_makefile
[-absolute_paths]
[-dependencies depends]
[-dc_shell exec_name]
[-format dcsh | dctcl]
-destination pass
[-target target_name]
[-lsf [-bsubargs bsub_args]]
[-update design_list]
[design]

write_parasitics

Writes parasitics in SPEF format to a disk file for the delay calculation tools.

int write_parasitics
[-output file_name]
[-format reduced | distributed]
[-min]
[-ratio ratio_number]
[-script]
**write_partition** (dctcl-mode only)

 Writes the database for a design into the Automated Chip Synthesis data structure. For use only in dc_shell-t (Tcl mode of dc_shell).

```bash
int write_partition
- type pre | post
 [-destination pass]
 [-hierarchy]
 [design]
```

**write_partition_constraints** (dctcl-mode only)

 Writes out the timing constraints for a design. For use only in dc_shell-t (Tcl mode of dc_shell).

```bash
int write_partition_constraints
 [-hierarchy]
 [-format dcsh | dctcl]
 [-destination pass]
 [-update design_list]
 [design]
```

**write_power**

 Calculates and saves dynamic and static power information of a design or instance for interface with RTL Analyzer.

```bash
int write_power
 [-net]
 [ -cell]
 [-only cell_or_net_list]
 [-cumulative]
 [-flat]
 [-exclude_boundary_nets]
 [-analysis_effort low | medium | high]
```
write_rtl

Writes the design synthesized by Behavioral Compiler to a file in a hardware description language (VHDL, Verilog, SystemC).

```c
int write_rtl
[-format output_format]
[-output output_file_name]
[-use_packages vhdl_packages]
[-include_files verilog_or_systemc_include_files]
[-simulation]
[-debug_mode]
[-rtl_script script_name]
[-ignore_rtl_processes]
```

write_rtl_load

Writes a script of RTL load commands for the current design.

```c
int write_rtl_load
[-format dctcl | dcsh]
[-output file_name]
```

write_script

Writes dc_shell commands to save the current settings.

```c
int write_script
[-hierarchy]
[-no_annotated_check]
[-no_annotated_delay]
[-full_path_lib_names]
[-format dctcl | dcsh]
[-output file_name]
```

write_sdc

Writes out a script in Synopsys Design Constraints (SDC) format.

```c
int write_sdc
file_name
[-version sdc_version]
```
write_sdf

Writes a Standard Delay Format (SDF) back-annotation file.

string write_sdf
[-version sdf_version]
[-instance inst_name]
file_name

write_test

Formats the test patterns for the current design into one or more test vector files.

int write_test
[-input test_program_name]
[[-output output_vector_file_name]
[-cumulative]]
[-format test_program_format]
[-first n_patterns]
[-parallel]
[-part_number part_number]
[-revision revision]

write_test_model

Writes a test model file.

int write_test_model
[-format ctl | db]
[-output model_file]
[design_name]

write_test_protocol

 Writes a test protocol file.

int write_test_protocol
[-out file_name]
[-format tpf | stil]

write_testsim_lib

Note: TestSim is obsolete with 1999.10, and has been replaced by the TetraMax fault simulator. For more information, see the TetraMax documentation.
write_timing

Writes leaf cell pin-to-pin timing information to a disk file.

int write_timing
[-output timing_file_name]
[-load_delay net | cell]
[-design design_name]
Commands Specific to dcsh Mode

The following commands are available only in dcsh mode:

- allocate_budgets
- check_unmapped
- execute
- include
- remove_variable
- write_designlist

Commands Specific to dctcl Mode

The following commands are available only in dctcl mode:

- acs_check_directories
- acs_compile_design
- acs_create_directories
- acs_get_parent_partition
- acs_get_path
- acs_merge_design
- acs_read_hdl
- acs_recompile_design
- acs_refine_design
- acs_report_directories
- add_to_collection
- after
- allocate_partition_budgets
- append
- apropos
- array
- auto_execok
- auto_import
- auto_load
- auto_load_index
- auto_qualify
- binary
- catch
- clock
- close
- compare_collections
- compile_partitions
- concat
- copy_collection
- create_command_group
create_pass_directories
current_design_name
date
define_proc_attributes
encoding
eof
error
error_info
eval
exec
expr
fblocked
fconfigure
fcopy
file
fileevent
filter_collection
flush
for
foreach_in_collection
format
get_cells
get_clocks
get_clusters
get_designs
get_generated_clocks
get_lib_cells
get_lib_pins
get_libs
get_message_info
get_multibits
get_nets
get_object_name
get_path_groups
get_pins
get_ports
get_references
get_timing_paths
getenv
gets
glob
global
incr
index_collection
info
interp
is_false
is_true
join
lappend
lindex
linsert
list_attributes
list_files
list_licenses
llength
lminus
lrange
lreplace
ls
lsearch
lsort
man
namespace
open
package
parse_proc_arguments
pid
print_suppressed_messages
print_variable_group
printenv
printvar
proc
proc_args
proc_body
puts
query_objects
read_db
read_edif
read_partition
read_sdc
read_verilog
read_vhdl
redirect
reg_global_var
regexp
regsub
remove_from_collection
remove_pass_directories
rename
report_partitions
report_pass_data
return
scan
seek
set
set_compile_partitions
setenv
sizeof_collection
socket
sort_collection
source
split
string
sub_designs_of
sub_instances_of
subst
suppress_message
switch
tclLog
tell
time
trace
unset
unsuppress_message
update
uplevel
upvar
variable
vwait
write_compile_script
write_makefile
write_partition
write_partition_constraints
Synthesis Variables

Synthesis tools define variables that are used to control the behavior of the tools.

**access_internal_pins**
Controls the creation, deletion, and user access of internal pins.
Default value for this variable is false.

**acs_area_report_suffix**
Specifies the suffix for area reports generated during the automated compile process. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is area.

**acs_autopart_max_area**
Defines partition threshold; used with other acs variables to control chip-level partitioning. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is 0.0 (no maximum area specified).

**acs_autopart_max_percent**
Controls chip-level partitioning; used with other acs variables. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is 0.0% (no maximum percentage specified).

**acs_bs_exec**
Specifies the location of the budget_shell executable. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is $SYNOPSYS/$arch/syn/bin/budget_shell.
acs_bsub_args
Specifies the command arguments for the batch submission command in the makefile.
Default value for this variable is "-K -P".

acs_bsub_exec
Specifies the batch submission command used to run the dc_shell command when compiling the design.
Default value for this variable is bsub.

acs_budget_output_file_suffix
Specifies the default suffix for log files generated by the allocate_partition_budgets command. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is btcl.out.

acs_budget_script_file_suffix
Specifies the default suffix for design budgeting script files generated by the allocate_partition_budgets command. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is btcl.

acs_budgeted_cstr_suffix
Specifies the suffix for constraint files generated by the derive_partition_budgets command. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is con.

acs_compile_script_suffix
Specifies the default suffix for script files generated by the write_compile_script command, sourced in the makefile generated by the write_makefile command, and located by the report_pass_data command. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is autoscr.
**acs_constraint_file_suffix**

Specifies the default suffix for constraint files generated by `write_partition_constraints` during the automated compile process. For use in `dc_shell-t` (Tcl mode of `dc_shell`) only.

Default value for this variable is `con`.

**acs_cstr_report_suffix**

Specifies the default suffix for constraint reports generated during the automated compile process. For use in `dc_shell-t` (Tcl mode of `dc_shell`) only.

Default value for this variable is `cstr`.

**acs_db_suffix**

Specifies the default suffix for `.db` files that are read or written during the automated compile process. For use in `dc_shell-t` (Tcl mode of `dc_shell`) only.

Default value for this variable is `db`.

**acs_dc_exec**

Specifies the location of the `dc_shell` executable. This variable is used by the `acs_compile_design`, `acs_refine_design`, and `acs_recompile_design` commands to generate the makefile. For use in `dc_shell-t` (Tcl mode of `dc_shell`) only.

Default value for this variable is `$SYNOPSYS/$arch/syn/bin/dc_shell`.

**acs_default_pass_name**

Specifies the prefix for the default data directory names. The pass number (either 0 or 1) is added to the prefix to generate the directory name.

Default value for this variable is `pass`.

**acs_exclude_extensions**

Specifies the file endings of files you do not want the `acs_read_hdl` command to analyze.

Default value for this variable is `""`. 

---

156 Synthesis Variables
acs_exclude_list
Specifies files and directories you do not want the acs_read_hdl command to analyze.
Default value for this variable is "[list $synopsys_root]".

acs_global_user_compile_strategy_script
Specifies the base file name for the user-defined default compile strategy. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is default.

acs_hdl_source
Specifies the location of the source code files analyzed by the acs_read_hdl command.
Default value for this variable is "".

acs_lic_wait
Specifies the maximum wait time for checking out all the licenses required by a compile job. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is 0.

acs_log_file_suffix
Specifies the default suffix for log files generated during the automated compile process. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is log.

acs_make_args
Specifies the command arguments for the make utility command (gmake, by default).
Default value for this variable is "-j".

acs_make_exec
Specifies the make utility command used to run the compile jobs.
Default value for this variable is gmake.
acs_makefile_name
Specifies the file name for the makefile generated by the write_makefile command and run by the compile_partitions command. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is Makefile.

acs_num_parallel_jobs
Specifies the number of compile jobs to run in parallel when using gmake as the make utility. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is 1.

acs_override_report_suffix
Specifies the suffix for user-defined partition report scripts. For use in Tcl mode of dc_shell (dc_shell-t) only.
Default value for this variable is report.

acs_override_script_suffix
Specifies the suffix for user-defined partition compile scripts. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is scr.

acs_qor_report_suffix
Specifies the suffix for QOR reports generated during the automated compile process; the default is qor. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is qor.

acs_script_mode
Specifies the dc_shell mode used by Automated Chip Synthesis for the compile process when running one of the pass commands (acs_compile_design, acs_refine_design, or acs_recompile_design). For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is dctcl.
acs_timing_report_suffix
Specifies the suffix for timing reports generated during the automated compile process. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is tim.

acs_tr_exec
Specifies the location of the transcript executable. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is $SYNOPSYS/$arch/syn/bin/dc-transcript.

acs_use_autopartition
Sets autopartitioning as the default partitioning strategy for the chip-level compile commands. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is false.

acs_use_dc_gate_level_budgeting
Specifies that ACS should use dc_shell budgeting to perform gate-level budgeting. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is true.

acs_use_default_delays
Sets top-down environment propagation as the constraint generation method for GTECH designs (the acs_compile_design command). When this variable is false (the default), the acs_compile_design command uses RTL budgeting to generate constraints for the compile partitions.
Default value for this variable is false.

acs_use_lsf
Specifies how the dc_shell command is run within the makefile. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is false.
acs_user_budgeting_script
   Specifies the file name for the user-defined budgeting script. For use in dc_shell-t (Tcl mode of dc_shell) only.
   Default value for this variable is budget.scr.

acs_user_compile_strategy_script_suffix
   Specifies the suffix for user-defined partition compile strategies. For use in dc_shell-t (Tcl mode of dc_shell) only.
   Default value for this variable is compile.

acs_verilog_extensions
   Specifies the file endings of files analyzed as Verilog source code by the acs_read_hdl command.
   Default value for this variable is "v".

acs_vhdl_extensions
   Specifies the file endings of files analyzed as VHDL source code by the acs_read_hdl command.
   Default value for this variable is "vhd".

acs_work_dir
   Specifies the root of the Automated Chip Synthesis project directory. For use in dc_shell-t (Tcl mode of dc_shell) only.
   Default value for this variable is the current working directory.

auto_link_disable
   Specifies whether the code to perform an auto_link during any Design Compiler command should be disabled.
   Default value for this variable is false.
**auto_link_options**

Specifies the link command options to be used when link is invoked automatically by various Design Compiler and DFT Compiler commands (for example, create_schematic and compile).

Default value for this variable is -all.

**auto_wire_load_selection**

Turns the automatic selection of the wire load model on or off.

Default value for this variable is true.

**bc_add_io_trace**

Inserts additional code to write out I/O debugging information during the RTL simulation run.

Default value for this variable is false.

**bc_allow_shared_memories**

Enables sharing of memories between processes when using Behavioral Compiler.

Default value for this variable is false.

**bc_chain_read_into_mem**

Enables Behavioral Compiler to chain input reads directly into a memory even if the inputs to the memory have been specified as non-chainable.

Default value for this variable is true.

**bc_chain_read_into_oper**

Enables Behavioral Compiler to chain input reads directly into an operation even if the inputs to the operation have been specified as non-chainable.

Default value for this variable is true.

**bc_constrain_signal_memories**

Determines whether shared memory accesses are scheduled as signals or variables.

Default value for this variable is false.
**bc_detect_array_accesses**
Determines whether precedences (dependency-related constraints) between nonconflicting pairs of array accesses to the same register file are automatically ignored.

Default value for this variable is true.

**bc_detect_memory_accesses**
Determines whether precedences (dependency related constraints) between nonconflicting pairs of memory accesses are automatically ignored.

Default value for this variable is true.

**bc_enable_analysis_info**
Enables the generation of BCView analysis information.

Default value for this variable is false.

**bc_enable_chaining**
Enables chaining of synthetic library operations.

Default value for this variable is true.

**bc_enable_multi_cycle**
Enables inference of multicycle synthetic library operations.

Default value for this variable is true.

**bc_enable_speculative_execution**
Controls whether speculative execution is enabled.

Default value for this variable is false.

**bc_estimate_mux_input**
Specifies the number of mux inputs for bc_time_design to use when estimating the combinational delay through muxes introduced by resource sharing.

Default value for this variable is 4.
**bc_estimate_timing_effort**
Specifies the level of timing estimation effort to be used in addition to the operator timing performed by bc_time_design.
Default value for this variable is high.

**bc_fsm_coding_style**
Controls the state assignment for the controller generated by Behavioral Compiler.
Default value for this variable is one_hot.

**bc_group_eql_logic**
Controls whether Behavioral Compiler groups all logic gates for the equal/non-equal operation.
Default value for this variable is true.

**bc_group_index_logic**
Controls whether Behavioral Compiler groups all logic gates for the array indexing operation.
Default value for this variable is true.

**bc_report_filter**
Contains a string to be used to filter the operators and variables for report_schedule -op and report_schedule -var.
Default value for this variable is "".

**bc_time_all_sequential_op_bindings**
Controls the delay calculation for operations mapped onto DW03_mult_n_stage.
Default value for this variable is false.

**bc_use_registerfiles**
Selects the array style.
Default value for this variable is false.
**bsd_max_in_switching_limit**

Specifies the maximum number of design inputs that may switch simultaneously while generating input DC parametric tests using the `create_bsd_patterns` command.

Default value for this variable is 60000.

**bsd_max_out_switching_limit**

Specifies the maximum number of design outputs that may switch simultaneously while generating output DC parametric tests using the `create_bsd_patterns` command.

Default value for this variable is 60000.

**bus_dimension_separator_style**

Specifies the style to use in naming an individual port member, net member, or cell instance member of a multi-dimensional EDIF array or of a multi-dimensional Verilog or VHDL vector.

Default value for this variable is "][]".

**bus_extraction_style**

Specifies the style used to extract the base name of net arrays, port arrays, and cell instance arrays in EDIF files.

Default value for this variable is %s[%%d-%d].

**bus_inference_descending_sort**

Specifies that the members of that port bus are to be sorted in descending order rather than in ascending order.

Default value for this variable is true.

**bus_inference_style**

Specifies the pattern used to infer individual bits into a port bus.

Default value for this variable is "."
**bus_minus_style**

Controls the naming of individual members of bit-blasted port, instance, or net buses with negative indices.

Default value for this variable is -%d.

**bus_multiple_separator_style**

Determines the name of a multibit cell which implements bits that do not form a range.

Default value for this variable is ,.

**bus_naming_style**

Specifies the style to use in naming an individual port member, net member, or cell instance member of an EDIF array or of a Verilog or VHDL vector.

Default value for this variable is %s[%d].

**bus_range_separator_style**

 Specifies the style to use in naming a net connected to the wire end of a ripper in the EDIF file.

Default value for this variable is :.

**cache_dir_chmod_octal**

Specifies the value of the mode bits for created cache directories.

Default value for this variable is 777.

**cache_file_chmod_octal**

Specifies the value of the mode bits for created cache files.

Default value for this variable is 664.

**cache_read**

Specifies a list of directories that contain cache files that will be read from whenever a cache entry is needed.

Default value for this variable is 
{}remote/dac1, 
 remotely/dac1\".\}
cache_read_info
Specifies whether an informational message will be printed each time a cache element is read.
Default value for this variable is true.

cache_write
Specifies the directory where optimized and unoptimized synlib parts will be written, if they are not already in the cache.
Default value for this variable is /remote/dac1.

cache_write_info
Specifies whether an informational message will be printed each time a cache element is written.
Default value for this variable is true.

case_analysis_log_file
Specifies the name of a log file generated during propagation of constant values, from case analysis or from nets tied to logic zero or logic one.
Default value for this variable is "".

case_analysis_with_logic_constants
When true, enables constant propagation, even if a design contains only logic constants.
Default value for this variable is false.

change_names_dont_change_bus_members
Controls how the change_names command modifies the names of bus members.
Default value for this variable is false.

change_names_update_inst_tree
Determines whether the instance trees for all designs in dc_shell are updated whenever names change.
Default value for this variable is true.
check_error_list
   Specifies the error codes that the check_error command checks for.
   Default value for this variable is EQN-16 EQN-18 EQN-20.

collection_result_display_limit
   Sets the maximum number of objects that can be displayed by any command that displays a collection.
   Default value for this variable is 100.

command_log_file
   Specifies the name of the file to which a log of the initial values of variables and commands executed is written. If the value is an empty string, a command log file is not created.
   Default value for this variable is /.command.log.

company
   Specifies the name of the company where Synopsys software is installed. The company name is displayed on the schematics.
   Default value for this variable is Synopsys.

compatibility_version
   Sets the default behavior of the system to be the same as the Synopsys software version specified in the variable.
   Default value for this variable is current release.

compile_assume_fully_decoded_three_state_busses
   Specifies whether the compile and translate commands can assume that three-state busses are fully decoded.
   Default value for this variable is false.
**compile_auto_ungroup_area_num_cells**
Defines the minimum number of cells all subdesigns in a hierarchy must have so that compile -auto_ungroup area does not ungroup the hierarchy.
Default value for this variable is 30.

**compile_auto_ungroup_delay_num_cells**
Defines the minimum number of cells all subdesigns in the hierarchy must have so that the compile -auto_ungroup delay command does not ungroup the hierarchy.
Default value for this variable is 500.

**compile_auto_ungroup_override_wlm**
Specifies whether the compiler considers a cell instance for automatic ungrouping, if the cell's wire load model differs from that of its parent.
Default value for this variable is false.

**compile-automatic_clock_phase_inference**
Specifies the method used to determine clock phase during sequential mapping.
Default value for this variable is strict.

**compile_checkpoint_cpu_interval**
Specifies a time, in minutes, to be used as the interval between each automatic checkpoint.
Default value for this variable is 0.0.

**compile_checkpoint_filename**
Specifies the name of the file to which the database containing all hierarchy of the checkpointed design is to be written.
Default value for this variable is ./CHECKPOINT.db.
**compile_checkpoint_phases**

Determines whether checkpoints are generated during execution of the compile command.

Default value for this variable is false.

**compile_checkpoint_pre_area_filename**

Specifies the name of the file to which the .db file containing all hierarchy of checkpointed design is written before the Area-Recovery phase.

Default value for this variable is ./CHECKPOINT_PRE_AREA.db.

**compile_checkpoint_pre_delay_filename**

Specifies the name of the file to which the database containing all hierarchy of the checkpointed design is to be written before the delay optimization phase.

Default value for this variable is ./CHECKPOINT_PRE_DELAY.db.

**compile_checkpoint_pre_drc1_filename**

Specifies the name of the file to which the database containing all hierarchy of the checkpointed design is written before Design Rule Fixing Phase 1.

Default value for this variable is ./CHECKPOINT_PRE_DRC1.db.

**compile_checkpoint_pre_drc2_filename**

Specifies the name of the file to which the database containing all hierarchy of the checkpointed design is written before Design Rule Fixing Phase 2.

Default value for this variable is ./CHECKPOINT_PRE_DRC2.db.
**compile_cpu_limit**

Specifies a time, in minutes, to be used as the limit for the amount of time to be spent in the phases after structuring and mapping. Optimization cancels when the limit is reached.

Default value for this variable is 0.0.

**compile_create_mux_op_hierarchy**

Controls whether MUX_OP implementations have their own level of hierarchy.

Default value for this variable is true.

**compile_create_wire_load_table**

Controls the type of wire load model generated by the create_wire_load command.

Default value for this variable is false.

**compile_delete_unloaded_sequential_cells**

Controls whether the compile command deletes unloaded sequential cells.

Default value for this variable is true.

**compile_disable_hierarchical_inverter_opt**

Controls whether inverters can be moved across hierarchical boundaries during boundary optimization.

Default value for this variable is false.

**compile_dont_touch_annotated_cell_during_inplace_opt**

Controls whether cells that have annotated delays can be optimized.

Default value for this variable is false.
**compile_dont_use_dedicated_scanout**
Controls whether test-ready compile (compile -scan), and subsequent compiles, use a scan cell's dedicated scan-out pin for functional connections.
Default value for this variable is 1.

**compile_dw_simple_mode**
This variable is for use only with the set_simple_compile_mode command. Controls whether DesignWare parts are compiled in simple compile mode.
Default value for this variable is false.

**compile_fast_optimization**
Controls whether fast algorithms are used in lieu of the default algorithms, for delay and area optimization.
Default value for this variable is true in low and medium effort, false in high effort.

**compile_fix_cell_degradation**
Controls whether the algorithms for fixing cell degradation violation are activated.
Default value for this variable is false.

**compile_hold_reduce_cell_count**
Controls whether the logic used to fix hold time violations is selected based on minimum cell count or minimum area.
Default value for this variable is false.

**compile_implementation_selection**
Controls whether the compile command reevaluates the current implementation of a synthetic module during optimization.
Default value for this variable is true.
**compile_instance_name_prefix**
Specifies the prefix used in generating cell instance names when compile is executed.
Default value for this variable is U.

**compile_instance_name_suffix**
Specifies the suffix used for generating cell instance names when compile is executed.
Default value for this variable is ".".

**compile_log_format**
Controls the format of the columns to be displayed during the mapping phases of compile and reoptimize_design.
Default value for this variable is "/elap_time %area \%wns \%tns \%drc \%endpoint".

**compile_mux_no_boundary_optimization**
Controls whether the compile command performs boundary optimization on MUX_OP implementations.
Default value for this variable is false.

**compile_negative_logic_methodology**
Specifies the logic value connected to floating inputs by the compile and translate commands.
Default value for this variable is false.

**compile_new_Boolean_structure**
Controls which Boolean structure optimization algorithm is used.
Default value for this variable is false.

**compile_new_optimization**
Controls which optimization algorithms Design Compiler uses.
Default value for this variable is false.
compile_no_new_cells_at_top_level
Controls whether the compile command adds new cells to the top-level design.
Default value for this variable is false.

compile_power_opto_only
Enables or disables power optimization in the presence of timing violations during incremental compile.
Default value for this variable is false.

compile_preserve_subdesign_interfaces
Controls whether the compile command preserves the subdesign interface.
Default value for this variable is false.

compile_retime_license_behavior
Controls how the compile command behaves when the optimize_registers or balance_registers attributes are set on a design or parts of a design and the required license(s) (BOA-BRT or DC-Expert) are not available immediately.
Default value for this variable is wait.

compile_seqmap_enable_output_inversion
Controls whether the compile command allows sequential elements to have their output phase inverted.
Default value for this variable is asynchronous set and asynchronous reset sequential cells.

compile_sequential_area_recovery
Controls whether the compile command tries to reduce area by remapping sequential elements.
Default value for this variable is false.
compile_simple_mode_block_effort
  Specifies the map effort value for compiling the lower level blocks in simple compile mode.
  Default value for this variable is none.

compile_top_acs_partition
  Controls whether the compile -top command only fixes all violations that cross top-level partitions.
  Default value for this variable is false.

compile_top_all_paths
  Controls whether the compile -top command fixes all violations in the design, or only those that cross top-level hierarchical boundaries.
  Default value for this variable is false.

compile_update_annotated_delays_during_inplace_opt
  Controls whether compile -in_place can update annotated delay values in the neighborhood of swapped cells. It has no effect for reoptimize_design and physopt, which always update annotated delay values.
  Default value for this variable is true.

compile_use_fast_delay_mode
  Selects the algorithm used for delay calculations when using the CMOS2 or nonlinear delay models.
  Default value for this variable is true.

compile_use_low_timing_effort
  Controls the tradeoff between runtime and accuracy for delay calculations when using technology libraries in which the input transition time affects the output transition time.
  Default value for this variable is false.
context_check_status
Reports whether the context_check mode is enabled (read-only).
Default value for this variable is false.

cps_default_sp
Specifies the default static probability value to be used by Power Compiler for modeling switching activity.
Default value for this variable is 0.5.

cps_default_tr
Specifies the default toggle rate value to be used by Power Compiler for modeling switching activity.
Default value for this variable is 0.5.

create_clock_no_input_delay
Affects delay propagation characteristics of clock sources created using create_clock.
Default value for this variable is false.

current_design
Specifies the design being worked on. This variable is used by most of the Synopsys commands.
Default value for this variable is "".

dc_shell_mode
Reports the mode of the current dc_shell session.
Default value for this variable is default or tcl.

default_input_delay
Specifies the global default input delay value to be used for environment propagation.
Default value for this variable is 30.0.
default_name_rules
Contains the name of a name_rule to be used as a
default by change_names if a name_rule is not
specified using the -rules name_rules option.
Default value for this variable is "".

default_output_delay
Specifies the global default output delay value to be
used for environment propagation.
Default value for this variable is 30.0.

default_port_connection_class
Contains the value of the connection class to be
assigned to ports that do not have a connection class
assigned to them.
Default value for this variable is universal.

default_schematic_options
Specifies options to use when schematics are
generated.
Default value for this variable is -size infinite.

design_library_file
Specifies the name of a file that contains design
library mappings.
Default value for this variable is ".synopsys_vss.setup".

designer
Specifies the name of the current user.
Default value for this variable is "".

disable_auto_time_borrow
Determines whether the report_timing command and
other commands will use automatic time borrowing.
Default value for this variable is false.
**disable_case_analysis**
When true, disables constant propagation from both logic constants and set_case_analysis command constants.
Default value for this variable is false.

**disable_library_transition_degradation**
Controls whether the transition degradation table is used to determine the net transition time.
Default value for this variable is false.

**do_operand_isolation**
Enables or disables automatic operand isolation by the Pragma or GTECH methods, for a design.
Default value for this variable is false.

**dpcm_arc_sense_mapping**
Controls whether Design Compiler maps half unate arcs to preset and clear arcs for sequential cells.
Default value for this variable is true.

**dpcm_debuglevel**
Determines the level of debugging for Design Compiler.
Default value for this variable is 0.

**dpcm_functionscope**
Controls how DPCM determines the FunctionalMode value when doing timing calculations.
Default value for this variable is global.

**dpcm_level**
Controls the mode used by DPCM for timing calculations.
Default value for this variable is performance.
dpcm_libraries
Specifies the libraries of the link_path that use DPCM
delay calculation.
Default value for this variable is "".

dpcm_rulepath
Specifies a list of paths, similar to a search path, for
locating the DPCM main rule.
Default value for this variable is "".

dpcm_rulespath
Locates the DPCM subrules when provided a list of
paths, the list being similar to a search path.
Default value for this variable is {}.

dpcm_slewlimit
Determines Design Compiler behavior when input pin
slew exceeds library limits.
Default value for this variable is true.

dpcm_tablepath
Specifies a list of paths, similar to a search path, for
locating the DPCM tables.
Default value for this variable is "".

dpcm_temperaturescope
Controls whether DPCM requests Temperature for
each delay calculation.
Default value for this variable is global.

dpcm_version
Specifies the version of the API used by the DPCM
delay calculation.
Default value for this variable is IEEE-P1481.
**dpcm_voltagescope**

Controls whether DPCM requests RailVoltage values for every delay calculation.

Default value for this variable is global.

**dpcm_wireloadsce**

Controls whether DPCM requests WireLoadModel values for every delay calculation.

Default value for this variable is global.

**duplicate_ports**

Specifies whether ports are to be drawn on every sheet for which an input or output signal appears.

Default value for this variable is false.

**dw_prefer_mc_inside**

Enables Synopsys Module Compiler to generate arithmetic DesignWare parts.

Default value for this variable is false.

**echo_include_commands**

Controls whether the contents of a script file is printed as it executes.

Default value for this variable is true.

**edifin_autoconnect_offpageconnectors**

Controls whether the EDIF reader can input a file containing an off-page connector and a port with the same name.

Default value for this variable is false.

**edifin_autoconnect_ports**

Controls whether the EDIF reader connects a port to a net with the same name (if there is one), even if there is no explicit connection statement.

Default value for this variable is false.
edifin_dc_script_flag
Controls whether the EDIF reader ignores comments in the input file, or parses them to look for Design Compiler commands.
Default value for this variable is "".

edifin_delete_empty_cells
Controls whether the EDIF reader deletes empty cells when reading in files.
Default value for this variable is true.

edifin_delete_ripper_cells
Controls whether the EDIF reader deletes designs of cellType RIPPER when reading in EDIF files.
Default value for this variable is true.

edifin_ground_net_name
Specifies EDIF ground nets.
Default value for this variable is "".

edifin_ground_net_property_name
Specifies EDIF ground nets.
Default value for this variable is "".

edifin_ground_net_property_value
Specifies EDIF ground nets.
Default value for this variable is "".

edifin_ground_port_name
Specifies EDIF ground ports.
Default value for this variable is "".

edifin_instance_property_name
Specifies the EDIF property used to determine the value of the cell_property attribute on a cell.
Default value for this variable is "".
edifin_lib_in_osc_symbol
Specifies the name of the input off-sheet connector symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_in_port_symbol
Specifies the name of the input port symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_inout_osc_symbol
Specifies the name of the input/output off-sheet connector symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_inout_port_symbol
Specifies the name of the input/output port symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_logic_0_symbol
 Specifies the name of the ground symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_logic_1_symbol
Specifies the name of the power symbol in the EDIF symbol library.
Default value for this variable is "".

edifin_lib_mentor_netcon_symbol
Specifies the name of the Mentor $netcon symbol in the EDIF symbol library.
Default value for this variable is "".
**edifin_lib_out_osc_symbol**
Specifies the name of the output off-sheet connector symbol in the EDIF symbol library.
Default value for this variable is "".

**edifin_lib_out_port_symbol**
Specifies the name of the output port symbol in the EDIF symbol library.
Default value for this variable is "".

**edifin_lib_ripper_bits_property**
Specifies the ripped_bits_property attribute of the ripper symbol (which is specified by the edifin_lib_ripper_cell_name variable).
Default value for this variable is "".

**edifin_lib_ripper_bus_end**
Specifies the ripped_pin attribute of the ripper symbol (which is specified by the edifin_lib_ripper_cell_name variable).
Default value for this variable is "".

**edifin_lib_ripper_cell_name**
Specifies the name of the ripper symbol in the EDIF symbol library.
Default value for this variable is "".

**edifin_lib_ripper_view_name**
Specifies the view of the ripper cell (as specified by the edifin_lib_ripper_cell_name variable) used in the Synopsys symbol library.
Default value for this variable is "".

**edifin_lib_route_grid**
Specifies the size of the route grid of the library.
Default value for this variable is 1024.
edifin_lib_templates
Specifies the sheet sizes available in a library.
Default value for this variable is {}.

edifin_portinstance_disabled_property_name
Specifies the EDIF property used to determine the value of the disabled attribute on a pin.
Default value for this variable is "".

edifin_portinstance_disabled_property_value
Controls whether the disabled attribute should be placed on the pin of the cell by specifying the required value of the corresponding EDIF property.
Default value for this variable is "".

edifin_portinstance_property_name
Specifies the EDIF property used to determine the value of the pin_properties attribute on a pin.
Default value for this variable is "".

edifin_power_net_name
Specifies EDIF power nets.
Default value for this variable is "".

edifin_power_net_property_name
Specifies EDIF power nets.
Default value for this variable is "".

edifin_power_net_property_value
Specifies EDIF power nets.
Default value for this variable is "".

edifin_power_port_name
Specifies EDIF power ports.
Default value for this variable is "".
**edifin_use_identifier_in_rename**
Controls whether objects are named using the identifier value or the name value.
Default value for this variable is false.

**edifin_view_identifier_property_name**
Allows multiple views (representations) of a cell to be created in the Synopsys database.
Default value for this variable is "".

**edifout_dc_script_flag**
Controls whether the EDIF writer embeds comments containing Design Compiler commands into the EDIF file.
Default value for this variable is "".

**edifout_design_name**
Controls the identifier in the design construct written at the end of EDIF files.
Default value for this variable is Synopsys_edif.

**edifout_designs_library_name**
Specifies the name that is given to the EDIF library construct that contains the cell constructs for the designs being written.
Default value for this variable is DESIGNS.

**edifout_display_instance_names**
Controls whether cell names are displayed in the schematic.
Default value for this variable is false.

**edifout_display_net_names**
Controls whether net names are displayed in the schematic.
Default value for this variable is false.
**edifout_external**
Controls whether the EDIF output contains complete symbol definitions or only interface descriptions.
Default value for this variable is true.

**edifout_external_graphic_view_name**
Specifies the name of an EDIF view used in a library contained in an external construct in a schematic EDIF file.
Default value for this variable is Graphic_representation.

**edifout_external_netlist_view_name**
Specifies the name of an EDIF view used in a library contained in an external construct in a netlist EDIF file.
Default value for this variable is Netlist_representation.

**edifout_external_schematic_view_name**
Specifies the name of an EDIF view used in a library contained in an external construct in a schematic EDIF file.
Default value for this variable is Schematic_representation.

**edifout_ground_name**
Specifies the name used by the EDIF writer for the Synopsys built-in ground cell.
Default value for this variable is logic_0.

**edifout_ground_net_name**
Specifies the name of the ground net.
Default value for this variable is ".".
**edifout_ground_net_property_name**
Specifies the name of the property used to identify ground nets.
Default value for this variable is "".

**edifout_ground_net_property_value**
Specifies the value of the property used to identify ground nets.
Default value for this variable is "".

**edifout_ground_pin_name**
Specifies the name of the ground cell pin.
Default value for this variable is logic_0_pin.

**edifout_ground_port_name**
Specifies the name of the ground port.
Default value for this variable is GND.

**edifout_instance_property_name**
Specifies the name of the EDIF property used to represent the value of the cell_property attribute on an EDIF instance.
Default value for this variable is "".

**edifout_instantiate_ports**
Controls whether symbol constructs for ports and off-sheet connectors are included in a library (or external) construct in an EDIF schematic.
Default value for this variable is false.

**edifout_library_graphic_view_name**
Specifies the name of an EDIF view used in a library contained in a library construct (not an external construct) in a schematic EDIF file.
Default value for this variable is Graphic_representation.
edifout_library_netlist_view_name
Specifies the name of an EDIF view used in a library contained in a library construct (not an external construct) in a netlist EDIF file.
Default value for this variable is Netlist_representation.

edifout_library_schematic_view_name
Specifies the name of an EDIF view used in a library contained in a library construct (not an external construct) in a schematic EDIF file.
Default value for this variable is Schematic_representation.

edifout_merge_libraries
Controls whether the EDIF writer writes all cells for an EDIF file in the same library.
Default value for this variable is false.

edifout_multidimension_arrays
Controls whether the EDIF writer can represent a bus as a multi-dimensional array.
Default value for this variable is false.

edifout_name_oscs_different_from_ports
Controls whether each off-sheet connector that is connected to a port is given a different name from the port in the descriptions of designs written in EDIF format.
Default value for this variable is false.

edifout_name_rippers_same_as_wires
Controls whether bus ripper cell instances are named based on the order they are created during schematic generation or based on the names of the extracted wires.
Default value for this variable is false.
**edifout_netlist_only**
Controls whether the EDIF writer generates both a netlist and a schematic, or a netlist only.
Default value for this variable is false.

**edifout_no_array**
Controls whether the EDIF writer uses array constructs.
Default value for this variable is false.

**edifout_numerical_array_members**
Controls the method used to generate the array index values.
Default value for this variable is false.

**edifout_pin_direction_in_value**
Includes an EDIF property with the same name as edifout_pin_direction_property_name, and value the same as this variable string, on input pins.
Default value for this variable is "".

**edifout_pin_direction_inout_value**
Includes an EDIF property with the same name as edifout_pin_direction_property_name, and value the same as this variable, on input/output pins.
Default value for this variable is "".

**edifout_pin_direction_out_value**
Includes an EDIF property with the same name as edifout_pin_direction_property_name, and value the same as this variable, on output pins.
Default value for this variable is "".

**edifout_pin_direction_property_name**
Specifies the name of the EDIF pin direction property.
Default value for this variable is "".
**edifout_pin_name_property_name**
Includes on pins an EDIF property with the same name as this variable string, and value the same as the pin name.
Default value for this variable is "".

**edifout_portinstance_disabled_property_name**
Creates in the portInstance construct a property with the same name as this variable string and a value that is the same as that of edifout_portinstance_disabled_property_value, if the pin of a cell instance has the disabled attribute on it.
Default value for this variable is "".

**edifout_portinstance_disabled_property_value**
Creates in the portInstance construct a property with the same name as edifout_portinstance_disabled_property_name and the same value as this variable string, if the pin of a cell instance has the disabled attribute on it.
Default value for this variable is "".

**edifout_portinstance_property_name**
Creates in the portInstance construct a property with the same name as this variable string and the same value as the pin_properties attribute, if the pin_properties attribute is on the pin of a cell instance.
Default value for this variable is "".

**edifout_power_and_ground_representation**
Determines power and ground representations in EDIF files.
Default value for this variable is cell.

**edifout_power_name**
Specifies the name used by the EDIF writer for the Synopsys built-in power cell.
Default value for this variable is logic_1.
**edifout_power_net_name**  
Specifies the name of the power net.  
Default value for this variable is "".

**edifout_power_net_property_name**  
Specifies the name of the property used to identify power nets.  
Default value for this variable is "".

**edifout_power_net_property_value**  
Specifies the value of the property used to identify power nets.  
Default value for this variable is "".

**edifout_power_pin_name**  
Specifies the name of the power cell pin.  
Default value for this variable is logic_1_pin.

**edifout_power_port_name**  
Specifies the name given to the power port.  
Default value for this variable is VDD.

**edifout_skip_port_implementations**  
Controls whether the EDIF writer writes portImplementation constructs for ports in the contents constructs of EDIF schematics.  
Default value for this variable is false.

**edifout_target_system**  
Specifies the target system for the generated EDIF files.  
Default value for this variable is "".

**edifout_top_level_symbol**  
Controls whether the EDIF writer writes the top-level symbol in EDIF schematic files.  
Default value for this variable is true.
**edifout_translate_origin**
Specifies the origin for the EDIF schematics.
Default value for this variable is "".

**edifout_unused_property_value**
Specifies the property value to be output for unused pins.
Default value for this variable is "".

**edifout_write_attributes**
Controls whether the EDIF writer embeds comments containing Design Compiler attribute definitions into the EDIF file.
Default value for this variable is false.

**edifout_write_constraints**
Controls whether the EDIF writer embeds comments containing Design Compiler constraint commands into the EDIF file.
Default value for this variable is false.

**edifout_write_properties_list**
Specifies a list of library, cell, or port properties to write into the EDIF description.
Default value for this variable is {}.

**enable_instances_in_report_net**
Enables report_net to report on instances in the current design.
Default value for this variable is false.
When true, long reports are displayed one page at a time (similar to the UNIX more command).
Commands affected by this variable include list, help, and the report commands.
Default value for this variable is true.
enable_recovery_removal_arcs
Controls whether Design Compiler accepts recovery and removal arcs that are specified in the technology library.
Default value for this variable is false.

enable_slew_degradation
Determines whether the transition degradation is taken into account for nets with physical information.
Default value for this variable is true.

equationout_and_sign
Specifies the and sign to use when writing a design in equation format.
Default value for this variable is *.

equationout_or_sign
Specifies the or sign to use when writing a design in equation format.
Default value for this variable is +.

equationout_postfix_negation
Controls whether a single quote (`) or an exclamation mark (!) is used as the negation operator.
Default value for this variable is true.

exit_delete_filename_log_file
Controls whether the file specified by the variable filename_log_file is deleted after design_analyzer or dc_shell exits normally.
Default value for this variable is true.

filename_log_file
Specifies the name of the filename log file to be used in case a fatal error occurs during execution of design_analyzer or dc_shell.
Default value for this variable is filenames.log.
**find_converts_name_lists**
Controls whether the find command converts the
name_list string to a list of strings before searching
for design objects.
Default value for this variable is false.

**fsm_auto_inferring**
Determines whether or not to automatically extract
finite state machine during the compile.
Default value for this variable is false.

**fsm_enable_state_minimization**
Determines whether or not the state minimization is
performed for all finite state machines (FSMs) in the
design.
Default value for this variable is false.

**fsm_export_formality_state_info**
Determines whether or not state machine encoding
information is exported into the files that will be used
by Formality.
Default value for this variable is false.

**gen_bussing_exact_implicit**
Controls whether schematics generated using the
create_schematic -implicit command should contain
implicit bus names instead of bus rippers.
Default value for this variable is false.

**gen_cell_pin_name_separator**
Specifies the character used to separate cell names
and pin names in the bus names generated by the
create_schematic command.
Default value for this variable is ./.
**gen_create_netlist_busses**
Controls whether create_schematic creates netlist buses whenever it creates buses on the schematic.
Default value for this variable is true.

**gen_dont_show_single_bit_busses**
Controls whether single-bit buses are generated in the schematic.
Default value for this variable is false.

**gen_match_ripper_wire_widths**
Controls whether the create_schematic command generates rippers whose width always equals the width of the ripped net.
Default value for this variable is false.

**gen_max_compound_name_length**
Controls the maximum length of compound names of bus bundles (for the create_schematic -sge command).
Default value for this variable is 256.

**gen_max_ports_on_symbol_side**
Specifies the maximum allowed size of a symbol created by create_schematic.
Default value for this variable is 0.

**gen_open_name_postfix**
Specifies the postfix to be used by create_schematic -sge when creating placeholder net names for unconnected pins.
Default value for this variable is ".".

**gen_open_name_prefix**
Specifies the prefix to be used by create_schematic -sge when creating placeholder net names for unconnected pins.
Default value for this variable is Open.
**gen_show_created_busses**
Controls whether a message is printed out every time a schematic bus is created from cell pins for which no equivalent net bus exists in the netlist.
Default value for this variable is false.

**gen_show_created_symbols**
Controls whether create_schematic prints a warning message every time it generates a new symbol for a cell because an appropriate symbol could not be found in the symbol libraries.
Default value for this variable is false.

**gen_single_osc_per_name**
Controls whether more than one off-sheet connector with any particular name is drawn on any schematic sheet.
Default value for this variable is false.

**generic_symbol_library**
Specifies the generic symbol library used for schematics.
Default value for this variable is generic.sdb.

**hdl_keep_licenses**
Controls whether HDL licenses that are checked out remain checked out throughout the dc_shell session or are released after use.
Default value for this variable is true.

**hdl_naming_threshold**
Determines the maximum character length that a parameter can have in order to be included in the design name.
Default value for this variable is 20.
hdl_preferred_license
Selects an hdl license to check out, if none is currently checked out.
Default value for this variable is "".

hdlin_allow_mixed_blocking_and_nonblocking
Controls whether blocking and nonblocking assignments to the same variable are considered an error. Effective only for Presto HDL Compiler and Verilog.
Default value for this variable is true.

hdlin_array_instance_naming_style
Sets the naming style for Verilog arrays of instances.
Default value for this variable is %s[%d].

hdlin_auto_full_case
Controls whether HDL Compiler attempts to detect full-case statements.
Default value for this variable is true.

hdlin_auto_netlist_reader
Controls whether the read -f verilog command attempts to automatically determine the type of input file and select the most efficient reader.
Default value for this variable is true.

hdlin_auto_parallel_case_early
Controls whether HDL Compiler tries to auto-detect easy to find parallel cases earlier in the elaborate phase.
Default value for this variable is true.

hdlin_auto_save_templates
Controls whether HDL designs containing parameters are read in as templates.
Default value for this variable is false.
**hdlin_black_box_pin_hdlc_style**
Controls whether to follow HDL Compiler pin naming style when performing black box module instantiation.
Default value for this variable is true.

**hdlin_build_selectop_for_var_index**
Controls whether the Presto HDL Compiler implements variable indexing expressions using a MUX_OP or SELECT_OPs with additional decode logic.
Default value for this variable is false.

**hdlin_call_stack_depth**
Limits the depth of nested or recursive function calls.
Default value for this variable is 1000.

**hdlin_check_no_latch**
Controls whether a warning message is issued if a latch is inferred from a design.
Default value for this variable is false.

**hdlin_check_user_full_case**
Causes a warning to be issued if the full_case pragma is placed on a case statement that is not full.
Default value for this variable is true.

**hdlin_check_user_parallel_case**
Causes a warning to be issued if the parallel_case pragma is placed on a case statement that is not parallel.
Default value for this variable is true.

**hdlin_compare_const_with_gates**
Specifies whether comparisons with constants are implemented with gates or synthetic operators.
Effective only for Presto HDL Compiler.
Default value for this variable is true.
**hdlin_compare_eq_with_gates**
Specifies whether equality comparisons are implemented with gates or synthetic operators. Effective only for Presto HDL Compiler.
Default value for this variable is true.

**hdlin_decoder_max_input_width**
Controls the circumstances in which the Presto HDL Compiler attempts to use a decoder to implement a series of comparisons involving an expression and a number of constants.
Default value for this variable is 31.

**hdlin_decoder_min_input_width**
Controls the circumstances in which the Presto HDL Compiler attempts to use a decoder to implement a series of comparisons involving an expression and a number of constants.
Default value for this variable is 5.

**hdlin_decoder_min_use_percentage**
Controls the circumstances in which the Presto HDL compiler attempts to use a decoder to implement a series of comparisons involving an expression and a number of constants.
Default value for this variable is 90.

**hdlin_dont_check_param_width**
Controls whether HDL Compiler checks that a Verilog parameter is large enough to contain the value assigned to it.
Default value for this variable is false.

**hdlin_dont_infer_mux_for_resource_sharing**
Controls whether HDL Compiler infers a MUX_OP for a signal-variable assigned in a case statement.
Default value for this variable is true.
**hdlin_dyn_array_bnd_check**
Controls whether logic is added to check the validity of array indices.
Default value for this variable is false.

**hdlin_enable_analysis_info**
Controls whether RTL Analyzer creates analysis information for designs processed by subsequent dc_shell commands.
Default value for this variable is false.

**hdlin_enable_presto**
Controls whether the two commands read and analyze use the Presto HDL Compiler for Verilog input files.
Default value for this variable is true.

**hdlin_enable_rtlsrc_info**
Controls whether RTL TestDRC creates file name and line number information for HDL constructs and instances for designs processed by subsequent dc_shell commands.
Default value for this variable is false.

**hdlin_enable_vpp**
Enables Verilog Preprocessing (VPP).
Default value for this variable is false.

**hdlin_escape_special_names**
Causes a backslash \ character to be prepended to Verilog names that contain special characters.
Default value for this variable is false.

**hdlin_ff_always_async_set_reset**
Controls whether HDL Compiler checks and reports asynchronous set and reset conditions of flip-flops.
Default value for this variable is true.
hdlin_ff_always_sync_set_reset
Controls whether every constant 0 loaded on a flip-flop under the clock event is used for synchronous reset, and every constant 1 loaded on a flip-flop under the clock event is used for synchronous set.
Default value for this variable is false.

hdlin_generate_naming_style
Specifies the naming style for generated design instances.
Default value for this variable is "%s_%d".

hdlin_generate_separator_style
Specifies the separator string for instances generated in multiple-nested loops.
Default value for this variable is "__".

hdlin_group_selectors
Enables grouping of single-bit selectors into multi-bit selectors.
Default value for this variable is true.

hdlin_hide_resource_line_numbers
Controls whether (V)HDL Compiler appends the HDL source line number to the inferred cell's name when inferring a synthetic library or DesignWare part.
Default value for this variable is false.

hdlin_infer_block_local_latches
Controls whether latches are allowed to be inferred for function- and task-scope variables.
Default value for this variable is true.
**hdlin_infer_comparators**

Controls whether the Presto HDL compiler will attempt to infer 6-output comparators. A 6-output comparator may save area when there are several different types of comparisons between the same two expressions in the design.

Default value for this variable is true.

**hdlin_infer_decoders**

Controls whether the Presto HDL compiler attempts to infer synthetic decoders. When an expression is compared against a series of different constants, that set of comparisons can optionally be implemented using a decoder.

Default value for this variable is false.

**hdlin_inferEnumeratedTypes**

Controls whether the Presto HDL Compiler tries to infer variables that have enumerated types.

Default value for this variable is false.

**hdlin_infer_fsm**

Controls whether the compiler automatically detects finite state machines (FSMs) in the HDL code in the Presto HDL Compiler flow.

Default value for this variable is true.

**hdlin_infer_function_local_latches**

Controls whether the Presto HDL Compiler infers latches inside functions and tasks.

Default value for this variable is false.

**hdlin_infer_multibit**

Specifies inference of multibit components for an entire design.

Default value for this variable is default_none.
**hdlin_infer_mux**

Determines whether and how HDL Compiler infers a MUX_OP.

Default value for this variable is default.

**hdlin_keep_feedback**

Removes (when false) all flip-flop feedback loops.

Default value for this variable is false.

**hdlin_keep_inv_feedback**

When true, retains all inverted flip-flop feedback loops.

Default value for this variable is true.

**hdlin_latch_always_async_set_reset**

Uses, for asynchronous reset, every constant 0 loaded on a latch, and uses, for asynchronous set, every constant 1 loaded on a latch, for a design subsequently analyzed.

Default value for this variable is false.

**hdlin_link_design**

Controls whether the Presto HDL Compiler links a design as it is elaborated.

Default value for this variable is false.

**hdlin_loop_invariant_code_motion**

Controls whether the Presto HDL Compiler attempts to move expressions calculated inside a loop to a location outside the loop, when it is safe to do so.

Default value for this variable is true.

**hdlin_map_to_entity**

Determines whether the Presto HDL Compiler will support map_to_entity synopsys pragmas.

Default value for this variable is true.
**hdlin_map_to_module**
Determines whether the Presto HDL Compiler will support map_to_module synopsys pragmas.
Default value for this variable is true.

**hdlin_map_to_operator**
Determines whether the Presto HDL Compiler will support map_to_operator synopsys pragmas.
Default value for this variable is true.

**hdlin_merge_nested_conditional_statements**
Infers a SELECT_OP (a generic logic component inferred for conditional statements) for each if or case statement in a nested conditional construct. Effective only in original HDLC flow, not in Presto.
Default value for this variable is false.

**hdlin_module_arch_name_splitting**
Controls whether Presto HDL Compiler recognizes a special format of Verilog module names, which allows users to specify both a module and an implementation architecture.
Default value for this variable is false.

**hdlin_mux_oversize_ratio**
Prevents inference of a sparse multiplexer, when the ratio of MUX_OP data inputs to unique data inputs is above the hdlin_mux_oversize_ratio.
Default value for this variable is 100.

**hdlin_mux_size_limit**
Limits the number of inputs of an inferred multiplexer.
Default value for this variable is 32.
**hdlin_mux_size_min**
Sets the lower bound for the number of inputs required to infer a multiplexer.
Default value for this variable is 2.

**hdlin_netlist_transform**
Determines whether the Presto HDL Compiler performs netlist transformation steps during elaboration.
Default value for this variable is true.

**hdlin_no_adder_feedthroughs**
Controls whether Presto HDL Compiler performs feed-through optimizations on adders or subtractors.
Default value for this variable is true.

**hdlin_no_sequential_mapping**
Determines whether Presto HDL Compiler performs the sequential mapping step.
Default value for this variable is false.

**hdlin_one_hot_one_cold_on**
Determines whether to use one_hot and one_cold attributes to simplify circuits.
Default value for this variable is true.

**hdlin_optimize_array_references**
Enables the optimization of array references when there are constants in the index expression.
Default value for this variable is true.

**hdlin_optimize_case_default**
Enables HDL Compiler to optimize the control logic for the default branch of a case statement.
Default value for this variable is true.
**hdlin_optimize_enum_types**  
Controls whether the Presto HDL compiler attempts to simplify designs by using the information that certain variables have an enumerated type. 
Default value for this variable is false.

**hdlin_optimize_shift_expressions**  
Enables the optimizing of shift expressions when the input and output widths differ. 
Default value for this variable is true.

**hdlin_preserve_vpp_files**  
Controls deletion of files with pp extension created by Verilog Preprocessing.  
Default value for this variable is false.

**hdlin_print_modfiles**  
Controls the printing of informational messages whenever modules are read from or written to disk during Presto HDL Compiler activities.  
Default value for this variable is true.

**hdlin_prohibit_nontri_multiple_drivers**  
Controls whether the Presto HDL compiler issues an error, or only a warning, when it finds multiple drivers of a net.  
Default value for this variable is true.

**hdlin_redundancy_elimination**  
Controls whether the Presto HDL Compiler applies a series of transformations to remove redundant computations from a design.  
Default value for this variable is true.
**hdlin_reg_report_length**
Sets the maximum length, in characters, of the Boolean formulas reported when `hdlin_report_inferred_modules` is set to verbose.
Default value for this variable is 60.

**hdlin_register_report_depth**
Controls the effort the Presto HDL Compiler puts into the accuracy of the sequential inference report.
Default value for this variable is true.

**hdlin_replace_synthetic**
Processes synthetic library parts in HDL designs during the read and elaborate commands. Actual gate-level implementations are inserted during the read command.
Default value for this variable is false.

**hdlin_report Enumerated Types**
Controls whether the Presto HDL Compiler prints a report of the variables in a design that are known to have an enumerated type. This includes both user-declared and automatically detected enumerated types.
Default value for this variable is true.

**hdlin_report fsm**
Controls whether the compiler reports finite state machines (FSMs) discovered in the HDL code in the Presto HDL Compiler flow.
Default value for this variable is true.

**hdlin_report Inferred Modules**
Generates a brief report for inferred latches, flip-flops, three-state, and multiplexer devices.
Default value for this variable is true.
**hdlin_report_mux_op**
Determined whether mux_op information is reported during HDL Compiler activity.
Default value for this variable is true.

**hdlin_report_syn_cell**
Determined whether the Presto HDL Compiler prints a report summary of synthetic cells.
Default value for this variable is false.

**hdlin_report_tri_state**
Controls whether the Presto HDL Compiler prints a report of three-state buffers inferred. Three-state buffers are inferred when a variable is assigned to the three-state value in the design.
Default value for this variable is true.

**hdlin_selector_simplify_effort**
Controls the effort the Presto HDL Compiler puts into simplifying selectors with constant inputs.
Default value for this variable is 1.

**hdlin_seqmap_search_depth**
Controls how deep Presto HDL Compiler looks for sets and resets of flip-flops and latches.
Default value for this variable is 3.

**hdlin_share_all_operators**
Controls whether the Presto HDL compiler attempts to share all identical arithmetic expressions or defers the sharing decision to the compile command.
Default value for this variable is false.

**hdlin_subprogram_default_values**
Determines which value the compiler will use as the default value for variables, 'LEFT of its type or 0s.
Default value for this variable is false.
hdlin_translate_off_on
   Ignores the translate_off and translate_on directives, when disabled by setting the value to false.
   Default value for this variable is true.

hdlin_translate_off_skip_text
   Causes VHDL Compiler to treat as comments the text between the translation directives translate_off and translate_on.
   Default value for this variable is false.

hdlin_unsigned_integers
   Controls whether the Presto HDL Compiler generates signed arithmetic operators for signed operations.
   Default value for this variable is false.

hdlin_upcase_names
   Controls whether identifiers in the Verilog source code are converted to uppercase letters or left in their original case.
   Default value for this variable is false.

hdlin_use_carry_in
   Controls whether the Presto HDL Compiler attempts to use the carry-in input of adders and subtracters, in conjunction with the hdlin_use_cin variable.
   Default value for this variable is false.

hdlin_use_syn_shifter
   Maps DesignWare shifter parts to shift operators in your HDL code. When this variable is disabled (false, the default), a flattened implementation is supplied.
   Default value for this variable is false.

hdlinVerboseCellNaming
   Enables/disables the verbose naming style of cells.
   Default value for this variable is false.
**hdlin_vhdl93_concat**
Controls the concatenation behavior the compiler uses to conform to the VHDL '93 Standard or the '87 Standard.
Default value for this variable is false.

**hdlin_vhdl_93**
Controls whether Presto-VHDL follows VHDL '93 Standard or '87 Standard.
Default value for this variable is true.

**hdlin_vpp_temporary_directory**
Determines where to create intermediate files, which Verilog preprocessing constructs.
Default value for this variable is "".

**hdlin_vrlg_std**
Controls whether Presto Verilog enforces Verilog 1995 or Verilog 2000.
Default value for this variable is 2000.

**hdlin_warn_array_bound**
Causes warning messages to be issued or suppressed when the index of an array element is out of bounds.
Default value for this variable is true.

**hdlin_warn_implicit_wires**
Controls whether the compiler warns users about implicitly declared wires.
Default value for this variable is true.
**hdlin_warn_mixed_blocking_and_nonblocking**

Controls whether warnings are issued when blocking and nonblocking assignments to the same variable are present. Effective only for Presto HDL compiler and Verilog and only if the variable hdlin_allow_mixed_blocking_and_nonblocking is set to true.

Default value for this variable is true.

**hdlin_warn_sens_list**

Controls whether the Presto HDL Compiler issues a warning when it detects that a variable or signal in a combinational always block is read but does not appear in the sensitivity list.

Default value for this variable is true.

**hdlin_while_loop_iterations**

Places an upper bound on the number of times a loop is unrolled (to prevent potential infinite loops).

Default value for this variable is 1000.

**hdlin_work_directory**

Specifies the directory used to store analyzed Verilog modules. Effective only for Presto HDL Compiler.

Default value for this variable is "./WORK".

**hdlin_write_gtech_design_directory**

Specifies the directory in which to place the RTL Analyzer intermediate files.

Default value for this variable is ".".

**hdlout_internal_busses**

Controls the way in which the write-format verilog command and the write-format vhdl command write out internal bused nets by parsing the names of the nets.

Default value for this variable is false.
hier_dont_trace_ungroup
Disables ungroup tracing set on the design with the
ungroup command.
Default value for this variable is 0.

high_fanout_net_pin_capacitance
Specifies the pin capacitance used to compute the
loading of high-fanout nets.
Default value for this variable is 1.0.

high_fanout_net_threshold
Specifies the minimum number of loads for a net to be
classified as a high-fanout net.
Default value for this variable is 1000.

hlo_disable_datapath_optimization
Disables built-in data-path optimization feature in
compile.
Default value for this variable is false.

hlo_ignore_priorities
Determines whether priorities set for synthetic
implementations are to be observed during resource
sharing and implementation selection under compile.
Default value for this variable is false.

hlo_minimize_tree_delay
During compile, enables tree height minimization
during resource sharing, if the minimize_tree_delay
attribute is not set.
Default value for this variable is true.

hlo_resource_allocation
Sets the default resource sharing type to be used by
the compile command, if the resource_allocation
attribute is not set.
Default value for this variable is constraint_driven.
**hlo_resource_implementation**
Sets the default implementation selection type to be used by the compile command, if the resource_implementation attribute is not set.
Default value for this variable is use_fastest.

**hlo_share_common_subexpressions**
Enables (during compile) sharing of common subexpressions during resource sharing.
Default value for this variable is true.

**hlo_share_effort**
Sets the relative amount of CPU time to be spent on resource sharing during compile.
Default value for this variable is low.

**hlo_transform_constant_multiplication**
Replaces, during elaborate command activity, all multiplications that have one constant input, by a series of shift, add, and subtract operations. The default is false, meaning that the multiplications are not replaced.
Default value for this variable is false.

**ilm_derive_keepout**
Enables and disables automatic inference of placement and wiring keepouts for interface logic models (ILMs) in Physical Compiler.
Default value for this variable is true.

**insert_dft_clean_up**
Causes the insert_dft command to use area recovery techniques to reduce the amount of test point logic.
Default value for this variable is true.
**insert_test_design_naming_style**
Specifies how the insert_dft command names new designs created during the addition of test circuitry.
Default value for this variable is %s_test_%d.

**lbo_cells_in_regions**
Puts new cells at specific locations within a cluster.
Default value for this variable is false.

**lib_thresholds_per_lib**
Causes tripPoint values in the Synopsys library to override user-specified values.
Default value for this variable is true.

**libgen_max_differences**
Specifies to the read_lib command the maximum number of differences to list between the v3.1 format description of a library cell and its statetable description.
Default value for this variable is -1.

**link_force_case**
Controls the case-sensitive or case-insensitive behavior of the link command.
Default value for this variable is check_reference.

**link_library**
Specifies the list of design files and libraries used during linking.
Default value for this variable is {
"*" your_library.db}.

**lsiin_net_name_prefix**
Contains the prefix added to names assigned to unnamed nets within a design read in the LSI/NDL format. Only alphanumeric characters and the underscore are allowed in the string.
Default value for this variable is NET_.

---

**Synthesis Variables** 213
lsiout_inverter_cell
Prevents the write-format lsi command from directly connecting logic zero or logic one to internal pins or ports.
Default value for this variable is "".

lsiout_upcase
Converts to uppercase all of the names within LSI/NDL netlists.
Default value for this variable is true.

ltl_obstruction_type
Controls the routing blockage type for the named obstructions, without route type being specified.
Default value for this variable is placement_only.

mentor_bidirect_value
Specifies the value of the property that identifies the design's pins or primary ports as being bidirectional.
Default value for this variable is INOUT.

mentor_do_path
Specifies a non-default directory that contains the NETED DO macros referenced by the write-f mentor output.
Default value for this variable is "".

mentor_input_output_property_name
Specifies the name of the property whose value represents the direction of pins and primary ports of the design.
Default value for this variable is PINTYPE.
**mentor_input_value**
Specifies the value of the property that identifies input pins or primary input ports. You can specify the property name using the variable `mentor_input_output_property_name`.
Default value for this variable is IN.

**mentor_logic_one_value**
Specifies the value of the property that identifies logic one nets. You can specify the name of the property using the variable `mentor_logic_zero_one_property_name`.
Default value for this variable is 1SF.

**mentor_logic_zero_one_property_name**
Specifies the name of the property whose value determines whether the net is connected to logic one (power) or logic zero (ground).
Default value for this variable is INIT.

**mentor_logic_zero_value**
Specifies the value of the property that identifies logic zero nets. You can specify the property name using the variable `mentor_logic_zero_one_property_name`.
Default value for this variable is OSF.

**mentor_output_value**
Specifies the value of the property that identifies output pins or primary output ports. You can specify the name of the property using the variable `mentor_primitive_property_name`.
Default value for this variable is OUT.

**mentor_primitive_property_name**
Specifies the name of the property placed on all automatically generated symbols.
Default value for this variable is PRIMITIVE.
**mentor_primitive_property_value**
Specifies the value of the property placed on all automatically generated symbols. You can specify the name of the property using the variable `mentor_primitive_property_name`.

Default value for this variable is MODULE.

**mentor_reference_property_name**
Specifies the name of the property that contains the reference name of every instance. The reference name is the name of the cell of which this object is an instantiation.

Default value for this variable is COMP.

**mentor_search_path**
Causes the write -f mentor command to add a NETED SEARCH command to the beginning of an output file. This variable contains the list of directories to be searched for cells to be instantiated while executing the NETED DO macro.

Default value for this variable is "".

**mentor_write_symbols**
Causes the write -f mentor command to include symbol (SYMED) information in the output file for all automatically generated symbols.

Default value for this variable is true.

**mgi_scratch_directory**
Specifies a directory in which to store the intermediate files created by external generators. By default, the scratch directory is set to designware_generator in the current working directory.

Default value for this variable is designware_generator.
**pdefout_full_path_name**  
Controls the output PDEF file during the writing out of hierarchical designs.  
Default value for this variable is false.

**pdefout_diff_original**  
Used in conjunction with the option -new_cells_only of the command write_clusters.  
Default value for this variable is true.

**physical_library**  
Specifies the list of technology physical libraries of components used in physical synthesis.  
Default value for this variable is "".

**pla_read_create_flip_flop**  
Affects read -f pla and when set to true, enables output register information in PLA files to be read in and stored, so that the output registers are instantiated within the design.  
Default value for this variable is false.

**plot_box**  
Causes a box to be drawn around the plot. The default is false.  
Default value for this variable is false.

**plot_command**  
Specifies the operating system command that produces a hard copy of the plot.  
Default value for this variable is lpr -Plw -h.

**plot_orientation**  
Specifies whether the schematic is vertical or horizontal.  
Default value for this variable is best_fit.
plot_scale_factor
Specifications a scaling factor for the schematic.
Default value for this variable is 100.

plotter_maxx
Specifies the x coordinate of the upper right corner of
the plot output device.
Default value for this variable is 584.

plotter_maxy
Specifies the y coordinate of the upper right corner of
the plot output device.
Default value for this variable is 764.

plotter_minx
Specifies the x coordinate of the lower left corner of
the plot output device.
Default value for this variable is 28.

plotter_miny
Specifies the y coordinate of the lower left corner of
the plot output device.
Default value for this variable is 28.

port_complement_naming_style
Defines the convention the compile command uses to
rename ports complemented as a result of using the
set_boundary_optimization command.
Default value for this variable is %s_BAR.

power_cg_flatten
Specifies to the ungroup command whether to flatten
Synopsys clock-gating cells.
Default value for this variable is false.
power_do_not_size_icg_cells
When true, compile does not size the integrated clock-gating cells in the design to correct DRC violations, because doing so results in lower area and power. The default is false.
Default value for this variable is false.

power_hdlc_do_not_split_cg_cells
When true, elaborate does not split clock-gating cells to limit their fanout.
Default value for this variable is false.

power_keep_license_after_power_commands
Affects the amount of time a DesignPower license is checked out during a dc_shell (Design Compiler) session.
Default value for this variable is false.

power_preserve_rtl_hier_names
Preserves the hierarchy information of the RTL objects in the RTL design.
Default value for this variable is false.

power_rtl_saif_file
Defines for the rtl2saif command where to store the forward-annotation SAIF file, if you do not specify the -output option.
Default value for this variable is power_rtl.saif.

power_sdpd_saif_file
Defines for the lib2saif command where to store the forward-annotation SAIF file, if you do not specify the -output option.
Default value for this variable is power_sdpd.saif.
**rc_input_threshold_pct_fall**
Specifies the threshold voltage that defines the startpoint of the falling cell or net delay calculation.
Default value for this variable is 50.

**rc_input_threshold_pct_rise**
Specifies the threshold voltage that defines the startpoint of the rising cell or net delay calculation.
Default value for this variable is 50.

**rc_output_threshold_pct_fall**
Specifies the threshold voltage that defines the startpoint of the falling cell or net delay calculation.
Default value for this variable is 50.

**rc_output_threshold_pct_rise**
Specifies the threshold voltage that defines the startpoint of the rising cell or net delay calculation.
Default value for this variable is 50.

**rc_slew_derate_from_library**
Specifies the derating needed for the transition times in the Synopsys library to match the transition times between the characterization trip points.
Default value for this variable is 1.0.

**rc_slew_lower_threshold_pct_fall**
Specifies the threshold voltage that defines the endpoint of the falling slew calculation.
Default value for this variable is 20.

**rc_slew_lower_threshold_pct_rise**
Specifies the threshold voltage that defines the startpoint of the rising slew calculation.
Default value for this variable is 20.
**rc_slew_upper_threshold_pct_fall**
Specifies the threshold voltage that defines the startpoint of the falling slew calculation.
Default value for this variable is 80.

**rc_slew_upper_threshold_pct_rise**
Specifies the threshold voltage that defines the endpoint of the rising slew calculation.
Default value for this variable is 80.

**read_db_lib_warnings**
Indicates that warnings are to be printed while a technology db library is being read in with the read command. When false (the default), no warnings are given.
Default value for this variable is false.

**read_name_mapping_nowarn_libraries**
Specifies a list of libraries for which no warning messages are to be issued by read-f edif -names_file if the libraries are not found.
Default value for this variable is {}.

**read_translate_msff**
Indicates (when true, the default) that master-slave flip-flops (specified with the clocked_on_also syntax) are to be automatically translated to master-slave latches. When false, both master and slave remain flip-flops.
Default value for this variable is true.

**reoptimize_design_changed_list_file_name**
Creates a file in which to store the list of cells that changed and cells and nets that were added during post-layout or in-place optimization.
Default value for this variable is "".
rtl_load_resistance_factor
Specifies a factor to be used by the set_rtl_load command to calculate resistance values from capacitance values for RTL loads.
Default value for this variable is 0.0.

sdc_write_unambiguous_names
Ensures that cell, net, pin, lib_cell, and lib_pin names that are written to the SDC file are not ambiguous.
The default value is true.
Default value for this variable is true.

sdfout_allow_non_positive_constraints
Writes out PATHCONSTRAINT constructs with nonpositive (<= 0) constraint values.
Default value for this variable is false.

sdfout_min_fall_cell_delay
Specifies the minimum non-back-annotated fall cell delay that the write_timing command writes to a timing file in SDF format.
Default value for this variable is 0.000000.

sdfout_min_fall_net_delay
Specifies the minimum non-back-annotated fall net delay that write_timing can write to a timing file in SDF format.
Default value for this variable is 0.000000.

sdfout_min_rise_cell_delay
Specifies the minimum non-back-annotated rise cell delay that write_timing can write to a timing file in SDF format.
Default value for this variable is 0.000000.
sdfout_min_rise_net_delay
Specifies the minimum non-back-annotated rise net
delay that the write_timing command can write to a
timing file in SDF format.
Default value for this variable is 0.000000.

sdfout_time_scale
Specifies the time scale of the delays written to timing
files in SDF format.
Default value for this variable is 1.000000.

sdfout_top_instance_name
Specifies the name prepended to all instance names
when writing timing files in SDF format.
Default value for this variable is "".

sdfout_write_to_output
Specifies whether the write_timing -f sdf command
writes interconnect delays between cells and top-level
output ports.
Default value for this variable is false.

search_path
Specifies directories that Design Compiler and DFT
Compiler search for files specified without directory
names.
Default value for this variable is {search_path + .}.

sh_arch
The sh_arch variable is set by the application to
indicate the current system architecture of the
machine you are using; such as sparcOS5, hpxx10,
and so on. This is a read-only variable.
Default value for this variable is Platform-dependent.
sh_command_abbrev_mode
Command abbreviation is meant as an interactive convenience. Script files should probably not use any command or option abbreviation because these files are then susceptible to command changes in subsequent versions of the application.
Default value for this variable is Anywhere.

sh_command_log_file
Specifies the name of the file to which is written a log of the initial values of variables and executed commands. For use in dc_shell-t (Tcl mode of dc_shell) only.
Default value for this variable is ./command.log.

sh_continue_on_error
Under normal circumstances, when executing a script with source, Tcl errors (syntax and semantic) cause the execution of the script to terminate. Setting sh_continue_on_error to true allows processing to continue when errors occur. By default, this variable is set to false.
Default value for this variable is false.

sh_dev_null
The sh_dev_null variable is set by the application to indicate the current null device. For example, on Unix machines, this is set to /dev/null. This is a read-only variable.
Default value for this variable is Platform-dependent.

sh_enable_page_mode
When true, long reports are displayed one page at a time (similar to the UNIX more command). Consult the man page for various commands that generate reports to see if they are affected by sh_enable_page_mode.
Default value for this variable is false.
sh_new_variable_message
The sh_new_variable_message variable controls a debugging feature for tracing the creation of new variables. Its primary debugging purpose is to catch the misspelling of an application-owned global variable. When true, an informational message (CMD-041) is displayed when a variable is defined for the first time at the command line. When false, no message is displayed.

Default value for this variable is true.

sh_new_variable_message_in_proc
The sh_new_variable_message_in_proc variable controls a debugging feature for tracing the creation of new variables in a Tcl procedure. Its primary debugging purpose is to catch the misspelling of an application-owned global variable.

Default value for this variable is false.

sh_new_variable_message_in_script
The sh_new_variable_message_in_script variable controls a debugging feature for tracing the creation of new variables within a sourced script. Its primary debugging purpose is to catch the misspelling of an application-owned global variable.

Default value for this variable is false.

sh_product_version
This variable is set to the version of the application currently running. The variable is read only.
sh_script_stop_severity
When a script is executed with the source command, there are several ways to get it to stop executing before it completes. One is to use the sh_script_stop_severity variable. This variable can be set to none, W, or E. When set to E, the generation of one or more error messages by a command will cause a script to stop. When set to W, the generation of one or more warning or error messages will cause a script to stop. Note that sh_script_stop_severity is ignored if sh_continue_on_error is set to true.

Default value for this variable is none.

sh_source_emits_line_numbers
When a script is executed with the source command, error and warning messages can be emitted from any command within the script. Using the sh_source_emits_line_numbers variable, you can help isolate where errors and warnings are occurring.

Default value for this variable is none.

sh_source_logging
When you source a script, the source command is echoed to the command log file. By default, each command in the script is logged to the command log file as a comment. You can disable this logging by setting sh_source_logging to false.

Default value for this variable is true.

sh_source_uses_search_path
Causes the search command to use the search_path variable to search for files. This variable is for use in dc_shell-t (Tcl mode of dc_shell) only.

Default value for this variable is true.
sh_tcllib_app_dirname
The sh_tcllib_app_dirname variable is set by the application to indicate the directory where application-specific Tcl files and packages are found. This is a read-only variable.

single_group_per_sheet
Specifies to the tool to put only one logic group on a sheet.
Default value for this variable is false.

site_info_file
Contains the path to the site information file for licensing.
Default value for this variable is ""

sort_outputs
Sorts output ports on the schematic by port name.
Default value for this variable is false.

suppress_errors
Specifies a list of error codes for which messages are to be suppressed during the current Design Analyzer/dc_shell session.
Default value for this variable is {}.

symbol_library
Specifies the symbol libraries to use during schematic generation.
Default value for this variable is {lsi_10k.sdb}.

synlib_disable_limited_licenses
Disables or enables limited licenses for synthetic library parts.
Default value for this variable is true.
**synlib_dont_get_license**

Specifies a list of synthetic library part licenses that the compiler does not automatically check out.

Default value for this variable is `{}`.

**synlib_evaluation_mode**

Evaluates synthetic library parts, when no DesignWare-Basic or DesignWare-FPGA-Basic keys are available.

Default value for this variable is false.

**synlib_hiis_force_on_cells**

Specifies a list of design cells on which the compiler is to force hierarchical incremental implementation selection (hiis).

Default value for this variable is `{}`.

**synlib_iis_use_netlist**

Allows netlists of the DesignWare parts to be used as opposed to timing models for cost comparison during the Incremental Implementation Selection step. Timing models are created without considering the design context while netlists are mapped in the context of the design.

Default value for this variable is false.

**synlib_model_map_effort**

Determines the map_effort used during modeling of synthetic library parts.

Default value for this variable is medium.

**synlib_optimize_non_cache_elements**

Controls whether non-cached models are optimized or used in an unoptimized form.

Default value for this variable is true.
**synlib_prefer_ultra_license**
Determines the use by Design Compiler of DesignWare Foundation library parts and DesignWare-Foundation and DesignWare-Foundation-Ultra licenses.
Default value for this variable is false.

**synlib_replace_synthetic_oani**
Replaces non-inferable, single implementation synthetic parts after elaboration.
Default value for this variable is false.

**synlib_sequential_module**
Controls the amount of processing the compile command does during resource sharing and implementation selection, on synthetic library modules that have implementations with sequential elements.
Default value for this variable is default.

**synlib_wait_for_design_license**
Specifies a list of authorized synthetic library licenses that Design Compiler is to wait for.
Default value for this variable is {}.

**synopsys_program_name**
This variable is read only, and is set by the application to indicate the name of the program you are running. This is useful when writing scripts that are mostly common between some applications, but contain some differences based on the application.

**synopsys_root**
This variable is read only, and is set by the application to indicate the root directory from which the application was run.
syntax_check_status
Reports whether the syntax_check mode is enabled.
Default value for this variable is false.

synthetic_library
Specifies a list of synthetic libraries to use when compiling. Default is {}.
Default value for this variable is {}.

systemcout_debug_mode
This variable is to be used only when the systemcout_levelize variable is set to true, to generate debug information.
Default value for this variable is false.

systemcout_levelize
Levelizes and flattens the netlist and replaces standard DesignWare operations with simulatable SystemC, before writing out the netlist, during write -f systemc command activity.
Default value for this variable is true.

target_library
Specifies the list of technology libraries of components to be used when compiling a design.
Default value for this variable is {your_library.db}.

tdlout_upcase
Converts to uppercase all names within TDL netlists.
Default value for this variable is true.

template_naming_style
Generates automatically a unique name when a module is built.
Default value for this variable is %s_%p.
template_parameter_style
Generates automatically a unique name when a module is built.
Default value for this variable is %s%d.

template_separator_style
Generates automatically a unique name when a module is built.
Default value for this variable is _.

test_allow_clock_reconvergence
Allows reconvergent nets to originate from the same top-level clock port.
Default value for this variable is true.

test_bsd_allow_tolerable_violations
Allows the optimize_bsd command to replace observe_and_control BSR cells with observe_only cells or remove BSR cells during timing-driven or area-driven optimization.
Default value for this variable is false.

test_bsd_control_cell_drive_limit
Specifies the number of cells a single BSR control cell can drive while optimizing control cell allocation during optimize_bsd command activity.
Default value for this variable is 0.

test_bsd_manufacturer_id
Specifies the manufacturer ID to use to create the value captured in the device identification register during execution of the insert_bsd command.
Default value for this variable is 0.
test_bsd_optimize_control_cell
   When true, allows the optimize_bsd command to optimize allocation of BSR control cells during area-driven optimization, using the value of the test_bsd_control_cell_drive_limit variable.
   Default value for this variable is false.

test_bsd_part_number
   Specifies the part number to use to create the value captured in the device identification register during execution of the insert_bsd command.
   Default value for this variable is 0.

test_bsd_version_number
   Specifies the version number to use to create the value captured in the device identification register during execution of the insert_bsd command.
   Default value for this variable is 0.

test_bsdl_default_suffix_name
   Specifies the default suffix for the name of the BSDL file generated by the write_bsdl command.
   Default value for this variable is bsdl.

test_bsdl_max_line_length
   Specifies the maximum number of characters per line for the output BSDL file the write_bsdl command produces.
   Default value for this variable is 72.

test_capture_clock_skew
   Specifies a qualitative measure of clock skew.
   Default value for this variable is small_skew.
**test_cc_ir_masked_bits**
Identifies instruction register (IR) bits to be masked during the search by the check_bsd command for all possible implemented instructions.
Default value for this variable is 0.

**test_cc_ir_value_of_masked_bits**
Specifies values to be forced into bits of the instruction register (IR) that are masked, during the search by the check_bsd command for all possible implemented instructions.
Default value for this variable is 0.

**test_check_port_changes_in_capture**
Checks (through the check_test command) for changes in values applied to bidirectional ports in the parallel measure cycle.
Default value for this variable is true.

**test_clock_port_naming_style**
Specifies the naming style used by the insert_scan command for global test signal ports created in designs during the addition of test circuitry.
Default value for this variable is test_c%s.

**test_dedicated_subdesign_scan_outs**
Instructs DFT Compiler to create dedicated scan-out ports on subdesigns.
Default value for this variable is true.

**test_default_bidir_delay**
Defines the default switching time of bidirectional ports in a tester cycle.
Default value for this variable is 55.0.
test_default_client_order
   Enables or disables test point utilities and determines
   the order in which they are invoked.
   Default value for this variable is {}.

test_default_delay
   Defines the default time in a tester cycle to apply
   values to input ports.
   Default value for this variable is 5.0.

test_default_period
   Defines the default length of a test vector cycle.
   Default value for this variable is 100.0.

test_default_scan_style
   Defines the default scan style for the insert_dft
   command if a scan style is not specified with the
   set_scan_style command.
   Default value for this variable is
   multiplexed_flip_flop.

test_default_strobe
   Defines the default strobe time in a test cycle for
   output ports and bidirectional ports in output mode.
   Default value for this variable is 95.0.

test_default_strobe_width
   Defines the default strobe pulse width, which is the
   default time that specifies how long after invocation
   the strobe pulse needs to be held active.
   Default value for this variable is 0.000000.

test_design_analyzer_uses_insert_scan
   Executes (when true) the insert_scan command
   through a Design Analyzer menu.
   Default value for this variable is true.
test_disable_find_best_scan_out
Selects (when true) the scan-out pin on a scan cell based on availability instead of timing slack.
Default value for this variable is false.

test_dont_fix_constraint_violations
Minimizes performance constraint violations.
Default value for this variable is false.

test_enable_capture_checks
Controls checking for capture violations during execution of the check_dft command.
Default value for this variable is true.

test_infer_slave_clock_pulse_after_capture
Guides protocol inference for master/slave test design methodologies during execution of the check_test command.
Default value for this variable is infer.

test_isolate_hier_scan_out
Prevents the insert_dft command inserting logic that isolates scan connections at hierarchical boundaries during functional operation.
Default value for this variable is 0.

test_jump_over_bufs_invs
Determines whether or not insert_scan and preview_scan consider output pins of buffers and inverters to be internal clocks.
Default value for this variable is true.

test_mode_port_inverted_naming_style
Specifies the naming style to use for the test_hold_logic_zero type of test mode signal ports to be created in the design.
Default value for this variable is test_mode_i%s.
**test_mode_port_naming_style**
Specifies the naming style to use for the
*test_hold_logic_one* type of test mode signal ports
created in the design.
Default value for this variable is *test_mode%s*.

**test_mux_constant_si**
Specifies how scan insertion uses a port you declare
as scan input, when the port is tied high or to the
ground in functional mode.
Default value for this variable is false.

**test_mux_constant_so**
Specifies how scan insertion uses a port you declare
as scan output, when the port is tied high or to the
ground in functional mode.
Default value for this variable is false.

**test_non_scan_clock_port_naming_style**
Specifies the style the *insert_dft* command uses to
name the ports that clock gating creates for nonscan
clocks.
Default value for this variable is *test_nsc_%s*.

**test_point_keep_hierarchy**
Synthesizes (when true) test points and ungroups the
test point design, during execution of the *insert_dft*
command.
Default value for this variable is false.

**test_preview_scan_shows_cell_types**
Shows (when true) cell instance types, during
execution of the *preview_scan* command.
Default value for this variable is false.
**test_protocol_add_cycle**

Adds an extra cycle (when true) after the shift cycle, in the test protocol, during execution of the `check_test` command.

Default value for this variable is true.

**test_rtl_drc_latch_check_style**

Specifies the latch check style to use during rtl_drc command activities.

Default value for this variable is default.

**test_scan_clock_a_port_naming_style**

Determines the naming style to be used by the `insert_scan` command for test scan clock a ports created in designs during the addition of test scan circuitry.

Default value for this variable is `test_sca%s`.

**test_scan_clock_b_port_naming_style**

Determines the naming style to be used by the `insert_scan` command for test scan clock b ports created in designs during the addition of test scan circuitry.

Default value for this variable is `test_scb%s`.

**test_scan_clock_port_naming_style**

Determines the naming style used by the `insert_scan` command for global test scan signal ports created in designs during the addition of test circuitry.

Default value for this variable is `test_scs%s`.

**test_scan_enable_inverted_port_naming_style**

Determines the naming style to be used by the `insert_scan` command for scan enable inverted ports created in designs during the addition of test scan circuitry.

Default value for this variable is `test_sei%s`.
**test_scan_enable_port_naming_style**
Determines the naming style to be used by the insert_scan command for test scan enable ports created in designs during the addition of test scan circuitry.
Default value for this variable is test_se%s.

**test_scan_in_port_naming_style**
Specifies the naming style used by the insert_scan command for serial test-signal ports created in designs during the addition of test circuitry.
Default value for this variable is test_si%s%s.

**test_scan_link_so_lockup_key**
Indicates to the preview_scan command what key to use to identify cells with scan-out lock-up latches in reports.
Default value for this variable is l.

**test_scan_link_wire_key**
Indicates to the preview_scan command the key to use to identify cells that drive wire-scan links in reports.
Default value for this variable is w.

**test_scan_out_port_naming_style**
Used the same as test_scan_in_port_naming_style.
Default value for this variable is test_so%s%s.

**test_scan_segment_key**
Tells the preview_scan command what key to use to identify scan segments in reports.
Default value for this variable is s.

**test_scan_true_key**
Specifies to the preview_scan command the key to use to identify in reports cells with true scan attributes.
Default value for this variable is t.
test_stil_max_line_length
Specifies the maximum line length for the file written
by the write_test_protocol -format stil command.
Default value for this variable is 72.

test_stil_multiclock_capture_procedures
Indicates to the write_test_protocol -format stil
command to create capture procedures in the STIL
protocol, with multiple clocks active in each
procedure.
Default value for this variable is false.

test_stil_netlist_format
Indicates to the write_test_protocol command what
netlist format to use when writing out STIL protocol
files.
Default value for this variable is db.

test_user_defined_instruction_naming_style
Indicates to the check_bsd command and the
write_bsd command the naming style to use for the
user-defined (nonstandard) instructions inferred by
these commands.
Default value for this variable is USER%d.

test_user_test_data_register_naming_style
Indicates to the check_bsd command and the
write_bsd command the naming style to use for the
user-defined (nonstandard) test data registers inferred
by these commands.
Default value for this variable is UTDR%d.

test_write_four_cycle_stil_protocol
Instructs the write_test_protocol -format stil
command to insert in the output STIL protocol file a
dummy cycle between all measure and capture cycles
in the STIL protocol.
Default value for this variable is false.
**text_editor_command**
Specifies the command that executes when the Edit/File menu is selected in the Design Analyzer text window.
Default value for this variable is xterm.

**text_print_command**
Specifies the command that executes when the File/Print menu is selected in the Design Analyzer text window.
Default value for this variable is lpr.

**timing_disable_internal_inout_cell_paths**
Disables bidirectional feedback paths in a cell if set to true.
Default value for this variable is true.

**timing_disable_internal_inout_net_arcs**
Disables bidirectional feedback paths that involve more than one cell if the variable is true.
Default value for this variable is true.

**timing_report_attributes**
Specifies the list of attributes to be reported with the report_timing -attributes command.
Default value for this variable is `{dont_touch, dont_use, map_only, size_only, ideal_net}`.

**timing_self_loops_no_skew**
Affects the behavior, runtime, and CPU usage of report_timing and compile.
Default value for this variable is false.

**true_delay_prove_false_backtrack_limit**
Specifies to the report_timing -true command the number of backtracks to use in searching for false paths.
Default value for this variable is 1000.
true_delay_prove_true_backtrack_limit
Specifies the number of backtracks the report_timing
-true command is to use in searching for true paths.
Default value for this variable is 1000.

uniquify_naming_style
Specifies the naming convention to be used by the
uniquify command.
Default value for this variable is %s_%d.

use_port_name_for_oscs
Specifies that when off-sheet connectors for nets also
have ports on them, they are given the name of the
port.
Default value for this variable is true.

verbose_messages
Causes more explicit system messages to be displayed
during the current Design Analyzer dc_shell session.
Default value for this variable is true.

verilogout_debug_mode
Instructs Behavioral Compiler (under specific
conditions) to insert additional code to write out
depthceptional debugging information during the RTL simulation
run. This variable (when set to true) is to be used only
when the variable verilogout_levelize is also set to
true.
Default value for this variable is false.

verilogout_equation
Writes Verilog assign statements (Boolean equations)
for combinational gates, rather than gate
instantiations.
Default value for this variable is false.
verilogout_higher_designs_first

Writes Verilog modules so that the higher level designs come before lower level designs, as defined by the design hierarchy.

Default value for this variable is false.

verilogout_ignore_case

Instructs the compiler not to consider case when comparing identifiers to Verilog reserved words.

Default value for this variable is false.

verilogout_include_files

Specifies to the write -f verilog command (when the verilogout_levelize variable is set to true) to write an include statement that will have the name of the value you set for this variable.

Default value for this variable is {}.

verilogout_levelize

Specifies, when true, to the write -f verilog command to levelize and flatten the netlist and to replace standard DesignWare operations with simulatable Verilog, before writing out the netlist.

Default value for this variable is false.

verilogout_no_negative_index

Shifts the negative range to the positive range starting 0. For example, if you have 0 downto -7, it becomes 7 downto 0.

Default value for this variable is false.

verilogout_no_tri

Declares three-state nets as Verilog wire instead of tri. This variable is useful in eliminating assign primitives and tran gates in the Verilog output.

Default value for this variable is false.
verilogout_show_unconnected_pins
Instructs the Verilog writer in dc_shell to write out all of the unconnected instance pins, when connecting module ports by name. For example, modb b1 (.A(in),.Q(out),.Qn()).
Default value for this variable is false.

verilogout_single_bit
Instructs the compiler not to output vectored ports in the Verilog output. All vectors are written as single bits.
Default value for this variable is false.

verilogout_unconnected_prefix
Instructs the Verilog writer in dc_shell to use the name (SYNOPSYS_UNCONNECTED_) to create unconnected wire names. The general form of the name is SYNOPSYS_UNCONNECTED_%d.
Default value for this variable is SYNOPSYS_UNCONNECTED_.

vhdllib_architecture
Determines the VHDL model types for the write_lib command to generate.
Default value for this variable is {FTGS,VITAL}.

vhdllib_glitch_handle
Determines if timing hazards have glitch-forced (on-detect) or spike-forced (on-event) Xs. When true (the default value), Xs are glitch-forced; when false, the Xs are spike-forced.
Default value for this variable is true.
vhdllib_logic_system
Selects the logic system in which to create the VHDL libraries. Currently only ieee-1164, for the IEEE 1164.1 nine value (std_logic) logic system, is allowed. Do not change the value of this variable to anything other than ieee-1164.
Default value for this variable is ieee-1164.

vhdllib_logical_name
This variable defines the logical name to be used by the VHDL libraries. If the variable is set to an empty string, the file base name is used as the default.
Default value for this variable is "".

vhdllib_negative_constraint
Determines whether a generated VITAL model is to have negative constraint handling capability.
Default value for this variable is false.

vhdllib_pulse_handle
Determines the algorithm used to handle timing hazards for the FTGS model. Values are glitch, spike, transport, inertial, or use_vhdllib_glitch_handle (the default).
Default value for this variable is use_vhdllib_glitch_handle.

vhdllib_tb_compare
Controls library testbench generation. No testbenches are created if this variable is set to 0 (the default).
Default value for this variable is 0.

vhdllib_tb_x_eq_dontcare
Specifies the default value for the testbenches X_Eq_DontCare generic Boolean flag. If X_Eq_DontCare is true, X states are ignored during output comparisons. The default is false.
Default value for this variable is false.
**vhdllib_timing_checks**

Determines the default value for the cell
TimingChecksOn generic Boolean flag in the VITAL model.

Default value for this variable is true.

**vhdllib_timing_mesg**

Determines the default value for the cell
Timing_mesg generic Boolean flag in FTSM, UDSM, and FTGS models. It also determines the value of the GlitchMode parameter for the VitalPropagatePathDelay procedure in the VITAL model.

Default value for this variable is true.

**vhdllib_timing_xgen**

Determines the default value for the cell Timing_xgen generic Boolean flag in FTSM, UDSM, and FTGS models. It also determines the default value for the cell XGenerationOn generic Boolean flag in the VITAL model.

Default value for this variable is false.

**vhdlout_architecture_name**

Determines the name to be used for the architecture the write -f vhdl command writes out.

Default value for this variable is SYN_%a_%u.

**vhdlout_bit_type**

Sets the basic bit type in a design written to VHDL.

Default value for this variable is std_logic.

**vhdlout_bit_type_resolved**

Prevents the VHDL writer (VHDLout) creating new bus resolution functions when writing wired logic.

Default value for this variable is true.
vhdlout_bit_vector_type
Sets the basic bit vector type in a design written to
VHDL.
Default value for this variable is std_logic_vector.

vhdlout_conversion_functions
Overrides conversion functions that are written out.
Default value for this variable is {}.

vhdlout_debug_mode
Instructs Behavioral Compiler (under specific
conditions) to insert additional code to write out
debugging information during the RTL simulation
run.
Default value for this variable is false.

vhdlout_dont_create_dummy_nets
Instructs the VHDL writer not to create dummy nets
for connecting unused pins or ports.
Default value for this variable is false.

vhdlout_dont_write_types
Specifies to the compiler not to write type
declarations for any types declared in the original
VHDL.
Default value for this variable is false.

vhdlout_equations
Specifies to the compiler how to write combinational
logic and sequential logic.
Default value for this variable is false.

vhdlout_follow_vector_direction
Specifies to the compiler to use the original range
direction when writing out an array.
Default value for this variable is false.
vhdlout_levelize
Specifies to the write -f vhdl command to levelize and flatten a netlist.
Default value for this variable is false.

vhdlout_one_name
Determines the literal name for constant bit value 1 in a design written in VHDL.
Default value for this variable is 1.

vhdlout_package_naming_style
Determines the name to be used for the type conversion packages written out by the VHDL writer (VHDLout).
Default value for this variable is CONV_PACK_%d;.

vhdlout_preserve_hierarchical_types
Affects the way in which the write -f vhdl command writes out ports on lower-level designs.
Default value for this variable is VECTOR.

vhdlout_separate_scan_in
Affects the way in which the scan chain is written out in VHDL.
Default value for this variable is false.

vhdlout_single_bit
Affects the way in which the write -f vhdl command writes out ports on the top-level design.
Default value for this variable is USER.

vhdlout_synthesis_off
Instructs the write -f vhdl command to write out synthesis_off and synthesis_on at the beginning and end of the configuration statement, but only when the vhdlout_write_top_configuration variable is also true.
Default value for this variable is true.
vhdlout_target_simulator
Names the target simulator to which the VHDL file is written. Currently, the only valid value is xp.
Default value for this variable is "".

vhdlout_three_state_name
Names the high impedance bit value used for three-state device values.
Default value for this variable is `Z.'

vhdlout_three_state_res_func
Names a user-supplied three-state resolution function that must be in one of the packages specified by the vhdlout_use_packages variable.
Default value for this variable is "".

vhdlout_time_scale
Specifies the scaling of the delays the compiler writes to timing files in Synopsys VHDL format.
Default value for this variable is 1.

vhdlout_top_configuration_arch_name
Determines the name of the outside architecture, depending on the setting of the vhdlout_write_top_configuration variable, and causes the VHDL writer (VHDLout) to write out a configuration statement.
Default value for this variable is A.

vhdlout_top_configuration_entity_name
Determines the name of the outside entity, depending on the setting of the vhdlout_write_top_configuration variable, and causes the VHDL writer (VHDLout) to write out a configuration statement.
Default value for this variable is E.
**vhdlout_top_configuration_name**
Determines the name of the configuration statement the write command writes out when the `vhdlout_write_top_configuration` variable is set to true.
Default value for this variable is CFG_TB_E.

**vhdlout_unknown_name**
Specifies the value the compiler uses to drive a signal to the unknown state.
Default value for this variable is X.

**vhdlout_upcase**

**vhdlout_use_packages**
Instructs the write command to write into the VHDL file a use clause containing a list of package names, for each of these packages, for all entities.
Default value for this variable is {IEEE.std_logic_1164}.

**vhdlout_wired_and_res_func**
Specifies the name of a wired and resolution function.
Default value for this variable is "".

**vhdlout_wired_or_res_func**
Specifies the name of a wired or resolution function.
Default value for this variable is "".

**vhdlout_write_architecture**
Instructs the write -format vhdl command to write out architecture declarations.
Default value for this variable is true.
vhdlout_write_components
Instructs the write-format vhdl command to write out component declarations for cells mapped to a technology library.
Default value for this variable is true.

vhdlout_write_entity
Instructs the write-format vhdl command to write out entity declarations.
Default value for this variable is true.

vhdlout_write_top_configuration
Instructs the write-format vhdl command to write out a configuration statement if necessary, such as when ports on the top-level design are written as vectors instead of user types.
Default value for this variable is false.

vhdlout_zero_name
Determines the literal name for constant bit value "0" in a design written to VHDL.
Default value for this variable is 0.

view_analyze_file_suffix
Specifies, in a list of file extensions, the files shown in the File/Analyze dialog box of Design Analyzer.
Default value for this variable is {v, vhd, vhdl}.

view_arch_types
Sets the contents of the architecture option menu. Contains a list of host machine architectures you can use for background jobs from the Design Analyzer viewer.
Default value for this variable is {sparcOS5, hpux10, rs6000, sgimips}.
view_background
Specifies the background color of the Design Analyzer viewer.
Default value for this variable is white.

view_cache_images
Specifies to Design Analyzer that the tool is to cache bitmaps for fast schematic drawing.
Default value for this variable is true.

view_command_log_file
Names a file and its location that is to contain all text written to the Design Analyzer Command window.
Default value for this variable is "".

view_command_win_max_lines
Contains the maximum number of lines to be saved in the Design Analyzer command window.
Default value for this variable is 1000.

view_dialogs_modal
Requires that the question and error dialogs in Design Analyzer be confirmed, before you can continue entering commands.
Default value for this variable is true.

view_disable_cursor_warping
Causes the cursor to be automatically warped (moved). When false, the cursor is automatically warped (or moved) to dialog boxes.
Default value for this variable is true.

view_disable_error_windows
Instructs Design Analyzer not to post the error windows when errors occur.
Default value for this variable is false.
view_disable_output
Disables output to the Design Analyzer command window.
Default value for this variable is false.

view_error_window_count
Specifies the maximum number of errors Design Analyzer reports for a command.
Default value for this variable is 6.

view_execute_script_suffix
Displays only files with the stated suffixes, from directories you select in the Execute Script option window of the Setup menu of Design Analyzer.
Default value for this variable is {.script, .scr, .dcs, .dcv, .dc, .con}.

view_info_search_cmd
Invokes, if set, the online information viewer through the optional menu item On-Line Information.
Default value for this variable is ".".

view_log_file
Specifies the file in which the tool stores events that occur in the viewer.
Default value for this variable is ".".

view_on_line_doc_cmd
Invokes, if set, the online documentation viewer, through the optional menu item On-Line Documentation.
Default value for this variable is ".".

view_read_file_suffix
Displays only files with the stated suffixes, from directories you select with the Read option of the File menu of Design Analyzer.
Default value for this variable is v, vhd, vhdl, xnf.
view_report_append
   Specifies to the tool to append to the specified file the
   reports the Design Vision menus generate.
   Default value for this variable is true.

view_report_interactive
   Specifies to the tool to send to the command line view
   the reports generated by Design Vision menus.
   Default value for this variable is true.

view_report_output2file
   Specifies to the tool to send to the specified file the
   reports generated by Design Vision menus.
   Default value for this variable is false.

view_script_submenu_items
   Allows users to add to the Design Analyzer Setup
   pulldown menu valid items to invoke user scripts.
   Default value for this variable is {}.

view_tools_menu_items
   Permits partial configuration of the Tools pulldown
   menu to add a new menu item for invoking user
   scripts.
   Default value for this variable is {}.

view_use_small_cursor
   Specifies to the tool that the X display is to support
   only 16 x 16-bit map size cursors.
   Default value for this variable is "".

view_use_x_routines
   Enables the use of internal arc-drawing routines
   (instead of X routines).
   Default value for this variable is true.
**view_write_file_suffix**
Displays only files with the stated suffixes, from directories you select with the Save As option of the File menu of Design Analyzer.
Default value for this variable is vhd, vhdl, xnf}.

**write_name_mapping_nowarn_libraries**
Specifies a list of libraries for which no warning messages are to be issued by write -f edif -names_file if the command does not find the libraries.
Default value for this variable is {}.

**write_name_nets_same_as_ports**
Specifies to the tool that nets are to receive the same names as the ports the nets are connected to.
Default value for this variable is false.

**write_test_formats**
Specifies the test vector formats recognized and created by the write_test command.
Default value for this variable is {synopsys, tssi_ascii, tds, verilog, vhdl, wgl}.

**write_test_include_scan_cell_info**
Specifies to the write_test command to include in the vector files scan-chain, cell, and inversion information for vector formats.
Default value for this variable is true.

**write_test_input_dont_care_value**
Controls the logic value the write_test command outputs when you have an input with a don't care condition.
Default value for this variable is X.
write_test_max_cycles
    Controls the automatic partitioning of long test sets
    across multiple files, by specifying the maximum
    number of tester cycles any one vector file can
    contain.
    Default value for this variable is 0.

write_test_max_scan_patterns
    Controls the automatic partitioning of long test sets
    across multiple files, by specifying the maximum
    number of scan test patterns any one vector file can
    contain.
    Default value for this variable is 0.

write_test_pattern_set_naming_style
    Specifies how to name pattern sets when long test sets
    are partitioned across multiple files.
    Default value for this variable is TC_Syn_%d.

write_test_round_timing_values
    Specifies to the write_test command to round to the
    nearest integer all timing values.
    Default value for this variable is true.

write_test_scan_check_file_naming_style
    Specifies how to name the file containing the vectors
    that test the scan chain logic.
    Default value for this variable is %s_schk.%s.

write_test_vector_file_naming_style
    Specifies how to name scan vector files, when long
    test sets are partitioned across multiple files.
    Default value for this variable is %s_%d.%s.

x11_set_cursor_background
    Specifies background color of the cursor in the Design
    Analyzer menus and viewer.
    Default value for this variable is "."
**x11_set_cursor_foreground**
Specifications foreground color of the cursor in the Design Analyzer menus and viewer.
Default value for this variable is "".

**x11_set_cursor_number**
Specifies the cursor, from the standard X cursor font used by the Design Analyzer menus and viewer.
Default value for this variable is -1.

**xnfin_dff_clock_enable_pin_name**
Instructs the read -format xnf command to assume that the clock enable pin name on the DFF (D flip-flop) component has the given string name.
Default value for this variable is CE.

**xnfin_dff_clock_pin_name**
Instructs the read -format xnf command to assume that the clock pin name on the DFF (D flip-flop) component has the given string name.
Default value for this variable is C.

**xnfin_dff_data_pin_name**
Instructs the read -format xnf command to assume that the data pin name on the DFF (D flip-flop) component has the given string name.
Default value for this variable is D.

**xnfin_dff_q_pin_name**
Instructs the read -format xnf command to assume that the Q pin name on the DFF (D flip-flop) component has the given string name.
Default value for this variable is Q.
xnfin_dff_reset_pin_name
Instructs the read -format xnf command to assume that the reset pin name on the DFF (D flip-flop) component has the given string name. Use this variable to support different versions of the Xilinx library, which can have differing pin names for these sequential devices.

Default value for this variable is RD.

xnfin_dff_set_pin_name
Instructs the read -format xnf command to assume that the set pin name on the DFF (D flip-flop) component has the given string name.

Default value for this variable is SD.

xnfin_family
Instructs the read -format xnf command to assume that the file being read is a design of the specified Xilinx family. The XNF reader supports only the 4000 family of Xilinx parts, so 4000 is the only valid value.

Default value for this variable is 4000.

xnfin_ignore_pins
Instructs the read -format xnf command to leave unconnected the given Xilinx component pin names.

Default value for this variable is GTS, GSR, GR.

xnfout_clock_attribute_style
Controls the style in which the write -format xnf command writes XNF netlist timing constraints.

Default value for this variable is CLK_ONLY.

xnfout_constraints_per_endpoint
Specifies the number of constraints the write -format xnf command will write out for each endpoint in the design for a constraint type. An endpoint is either an output port or an input pin to a sequential element.

Default value for this variable is 50.
xnfout_default_time_constraints
Instructs the write -format xnf command (when set to true, the default) to write out default time constraints for the paths in the design that are not covered by a specific path timing constraint.
Default value for this variable is true.

xnfout_library_version
Instructs the write -format xnf command to write out the LIBVER= attribute with this string, on symbol records.
Default value for this variable is "".

xterm_executable
Specifies the path to an xterm program spawned to run Synopsys analysis tools (for example, RTL Analyzer or BCView). The default is xterm.
Default value for this variable is xterm.