SIEMENS EDA

Oasys-RTL™ Command Reference Manual

Software Version 2022.2.R1

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a leading global provider of product life cycle management (PLM) software and services with 7 million licensed seats and 71,000 customers worldwide. Headquartered in Plano, Texas, Siemens Digital Industries Software works collaboratively with companies to deliver open solutions that help them turn more ideas into successful products. For more information on Siemens Digital Industries Software products and services, visit www.siemens.com/plm.

Support Center: support.sw.siemens.com
Send Feedback on Documentation: support.sw.siemens.com/doc_feedback_form

Table of Contents

List of Tables

Chapter 1
Overview

This document lists all supported Oasys-RTL commands and invocation arguments for the tool.

oasys Tool Invocation

Invokes the Oasys-RTL tool from the command line in interactive, batch, or GUI mode.

Usage

oasys [-gui] [-title <name>] [-queue] [-log <name>] [-exec <cmds>] [-noinit] [-noexit] [-echo] [-nocheckout <keys>] [-help] [<tcl_file> ...]

Arguments

- -gui

  Launches the graphical user interface. If this option is omitted, you can invoke the GUI using the start_gui command from the oasys command prompt.

- -title <name>

  Sets window title to <name>. This option has effect only if the -gui option is also specified. It only applies to the first GUI window that is opened. Closing and restarting the GUI does not apply the <name> to the window title.

- -queue

  Specifies that a queue be started if a license is not available. When a queue exists, the tool waits for an available license instead of exiting immediately.

- -log <name>

  Specifies the prefix, <name>, for the generated log files.

- -exec <cmds>

  Specifies that the Tcl list of commands, <cmd>, be executed after reading the init files. The commands are executed before the interactive prompt appears and before any command file is executed.

- -connect <host>:<port>

  Connects to the specified host and port.

- -workdir <directory>

  Changes to the specified working directory.

- -noinit

  Prevents the loading of the initialization (init) files (~/.oasys/oasys.tcl and ./oasys.tcl).

- -noexit

  Directs the tool to enter interactive mode instead of exiting after loading Tcl files.

- -echo

  Echoes (displays to screen) Oasys-RTL commands from the command_file be echoed to the terminal as each one is executed. Built-in Tcl commands are not echoed.

- -nocheckout <keys>

  Runs the Oasys-RTL tool without checking out licenses for the features named in <keys>, where <keys> is a string containing space-separated feature keys. The valid keys that can be specified are rt_floorplan, rt_dft, rt_power, psynfloorplan, psyndft, and psynpower. The rt_designer or psyncore license must always be checked out. Specifying rt_designer or psyncore in the keys provided to the -nocheckout option has no effect. Excluded licenses are noted in the log file.

- -help

  Prints a short help message to standard output. If other options are specified with the -help option, they are ignored.

- <tcl_file> ...

  Specifies Tcl script files to source when the tool is invoked. The script files are executed in the order given in the list. The tool immediately exits after the last script is complete, unless the -noexit argument is specified.

Description

Invoke the Oasys-RTL tool in either the command-line or GUI mode.

To invoke the tool in command-line mode, enter the following (where $OASYS_HOME is the installation path for the Oasys-RTL tool):

```
> $OASYS_HOME/bin/oasys
```

To invoke the tool in GUI mode, do one of the following:

- Start GUI at invocation:

  ```
  > $OASYS_HOME/bin/oasys -gui
  ```

- Start GUI from inside the Oasys-RTL tool:

  ```
  > $OASYS_HOME/bin/oasys
  % start_gui
  ```

When the tool is invoked in command-line mode, you can use any standard UNIX shell commands, such as the following:

```
% ls
% cd <dir_name>
% pwd
```

The Oasys-RTL tool first searches for an internal tool command that matches your entry.

Log, debug and command files are created each time you invoke the Oasys-RTL tool. These files are named following one of two conventions:

Default File Naming Convention

The default log file prefix is *oasys*. The filenames of subsequent sessions are appended with ID numbers to prevent overwriting of the files. They are named using the following naming convention:

- **Log File —** *<logfile_prefix>.log.<run_ID>*

  Contains error, warning, and information messages from the session.

- **Debug File —** *<logfile_prefix>.dbg.<run_ID>*

  Contains any crash information that may be helpful in reproducing problems. encountered during operation. If no errors occur during the Oasys-RTL run, the debug file is deleted when the Oasys-RTL tool exits.

- **Command File —** *<logfile_prefix>.cmd.<run_ID>*

  Contains command history from the session.

Each time you start the Oasys-RTL tool, the *<run_ID>* is incremented.

User-Specified Naming Convention

You can specify the name of the log file with the -log option. The specified prefix is also used for the debug (*.dbg*) and command (*.cmd*) files.

Oasys-RTL Command Help

Command help is available from within the Oasys-RTL shell.

There are two ways to access help information for commands in the Oasys-RTL shell environment:

- Use the “man” command, similar to the UNIX utility as follows:

  ```
  % man {<command_name> | <parameter_name>}
  ```

  For example:

  ```
  % man get_selection
  ```

  The displayed information is also contained in this reference manual.

- Use the “-help” option for a command or parameter as follows:

  ```
  % {<command_name> | <parameter_name>} -help
  ```

For example:

```
% get_selection -help
```

Only command argument information is displayed with the "-help" option.

Documentation Access

You can access the Oasys-RTL documentation from within the tool.

Prerequisites

To access the documentation within the GUI or at the shell, ensure that you have set up Acrobat Reader:

- Install Acrobat Reader.
- Include the path to Acrobat Reader in your PATH environment variable, or set the MGC_PDF_READER environment variable to the Acrobat Reader executable.

Procedure

1. To open the documentation in the GUI, go to the Help menu.
2. To open the documentation at the command line, type the following:

```
% mgcdocs
```

Syntax Conventions

This documentation uses metacharacters and font properties as shown in Table 1-1 to describe this tool's command usage.

Table 1-1. Syntax Conventions

Convention	Example	Usage
`% [oasys-RTL]$`	% report_area [oasys-RTL]$ report_timing	Oasys-RTL prompt.
`>`	> oasys -gui &	A UNIX shell prompt.
`Courier`	% read_config cpu_config.tcl	The courier font indicates an example of a command, command option, or argument that you type.
Boldface	disable_timing_mode **-modes *mode_list***	A boldface font indicates a required argument.

Table 1-1. Syntax Conventions (cont.)

Convention	Example	Usage
<argument>	read_config <config_file>	The command argument appears between angle brackets when a user-defined input is required.
Italic	total_negative_slack [-group *group*] [-mode *name*]	Italics indicate a user-specified value.
[ ]	report_net [-detail] <pin>	Square brackets indicate optional arguments. Do not include the brackets when entering the command.
|	`create_blockage -type {hard | soft | partial | macro}`	Vertical bars indicate a set of possible choices. Specify one or none of the choices presented. Do not include the bar when entering commands.
Underline	set_dont_retime ***register_instance_list*** [true | false]	An underlined item indicates either the default argument or the default value of an argument.
{ }	`set_case_analysis { 0 | 1 | rise | fall}<port_pin_list>`	Braces enclose arguments to show grouping. Do not include the braces when entering the command.
...	set_clock_groups -group *clocks_list* ...	An ellipsis follows an argument or group of arguments that may appear more than once. When an ellipsis follows any style of brackets ({ }, [ ], ( ), < >), the entire contents in the bracket are repeated. Do not include the ellipsis when entering the command. If an ellipsis appears with brackets ([<argument>] ...), you can specify zero or more arguments. If an ellipsis appears without brackets (<argument> ...), you must specify at least one argument.
' '	read_library [-target_library '{'*targets*'}'] ***lib_files***	Single quotes enclose metacharacters that are intended to be entered literally. Do not include the quotes when entering the command.

Identifying Edges and Edge Partitions

Several physical commands involve actions relating to chip edges or edge partitions. These commands include: assign_pins, report_edges, and report_groups. The edge and edge partition identification syntax is as follows:

```
{<edge_id> [:<partition_id>]}
```

<edge_id> — identifies a particular chip edge

<parition_id> — identifies a particular edge subsection

The edge_id numbering convention defines edge 0 as the far left edge. In a clockwise manner, subsequent edges are numbered as 1, 2, 3, and so on. For finer-grain control and reporting, you may also identify a particular edge partition with the partition_id value. Specify partition_id as a fractional value such as 1/3 (the first of three equal portions) or 2/4 (the second of four equal portions).

Note

Edge paritions are also ordered in a clockwise fashion, similar to edge_id. The numerator must be greater than or equal to one, but may not exceed the denominator.

For example, to specify the second quarter of the third edge (right edge for a rectangular core), enter:

{3:2/4}

To specify the first third of the fourth edge (bottom edge for a rectangular core), enter:

{4:1/3}

In addition, you may also specify a contiguous range of subsections, such as:

{3:2/5-4/5}

The example above specifies partitions 2 of 5, 3 of 5, and 4 of 5 on the third edge.

Chapter 2
Read Commands

The read commands specify design and library input files for the synthesis flow.

Table 2-1. Read Commands

Command	Description
load_upf	Loads the power intent from the specified unified power format (UPF) file for a given design.
read_config	Reads and runs an existing configuration file to set up all input files and run a flow for a given design. The config file is generated from the GUI.
read_ctl	Reads test model for a macro (cell) or an instance that has pre-connected scan chains, in IEEE1450.6 (CTL) format.
read_db	Restores a previously saved Oasys-RTL database (ODB) design state.
read_def	Loads physical layout information from Design Exchange Format (DEF) files.
read_explore_checkpoint	Loads all relevant metrics for a session at the specified point in the exploration flow.
read_lef	Loads cell physical layout information from Library Exchange Format (LEF) files.
read_library	Loads library cell models defined in Liberty libraries.
read_ptf	Loads a PTF and sets the resistance and capacitance values per unit length for each layer.
read_saif	Reads the switching activity annotations on the nodes of the design. Switching activity and toggle counts in Switching Activity Interchange Format (SAIF) are passed to the tool using this command.
read_sdc	Loads design constraints from a Synopsys Design Constraint (SDC) file.
read_vcd	Loads value change information from a Value Change Dump (VCD) file.
read_verilog	Reads (or loads) one or more design source files in Verilog or System Verilog formats before the synthesize step.

Table 2-1. Read Commands (cont.)

Command	Description
read_vhdl	Reads (or loads) one or more design source files in VHDL format before the synthesize step.

load_upf

Loads the power intent from the specified unified power format (UPF) file for a given design.

Usage

load_upf[-scope *instance_name*] ***upf_file***

Arguments

- -scope *instance_name*

  The scope or hierarchical level of the design to which the UPF file applies. The default is the top level.

- ***upf_file***

  Specifies the name of the file that contains the unified power format (UPF) commands.

Description

Loads the power intent from the specified unified power format (UPF) file for a given design. The Oasys-RTL tool supports IEEE standard power intent specifications including UPF1.0 and most of IEEE 1801-2009 (UPF2.0).

The load_upf command must be run after the synthesize command.

Examples

The following example reads a unified power format file called *my_design.upf:*

```
% load_upf my_design.upf
```

Related Topics

Read Commands

commit_upf

synthesize

UPF Command Support

read_config

Reads and runs an existing configuration file to set up all input files and run a flow for a given design. The config file is generated from the GUI.

Usage

read_config *config_file*

Arguments

- *config_file*

  File that holds the inputs and flow configuration.

Examples

The following example reads a configuration file called *my_design.tcl:*

```
% read_config my_design.tcl
```

Related Topics

Read Commands

read_ctl

Reads test model for a macro (cell) or an instance that has pre-connected scan chains, in IEEE1450.6 (CTL) format.

Usage

read_ctl [-lib_cell *cell_name*] [-instance *instance_name*] [-verbose] ***ctl_file***

Arguments

- -lib_cell *cell_name*

  Test model applies to all instances of the specified macro (cell).

- -instance *instance_name*

  Apply only to the specified instances.

- -verbose

  Outputs details of the conversion from CTL format to the Oasys-RTL internal format.

- ***ctl_file***

  The name of the CTL file.

Description

Reads test model for a macro (cell) or an instance that has pre-connected scan chains, in IEEE 1450.6 (CTL) format. The CTL information is used when connecting scan chain. The connections made to the macro cells are chosen based on the CTL model.

Examples

The next example reads a CTL file:

```
% read_ctl my_model.ctl
```

Related Topics

Read Commands

read_db

Restores a previously saved Oasys-RTL database (ODB) design state.

Usage

read_db ***filename*** [-header_only] [-skip_tsdb]

Arguments

- ***filename***

  Required. Specifies the filename of an ODB design database. You can omit the *.odb* file extension and the tool searches for the specified name with the *.odb* file extension. If the complete filename does not have the *.odb* extension, the read_db command does not find it.

- -header_only

  Optional. Specifies to read and display only the header information of an ODB without loading the design.

- -skip_tsdb

  Skip reading the Tessent Database directory details.

Description

The read_db command loads the design, libraries, and other design setups that were saved to the ODB design database with the write_db command. Specifically, it performs the following, depending on the type of database:

- Reads in the integrated design database containing both the design and library (generated by the command “write_db -data all”).
- Reads in the library database (generated by the command “write_db -data library”) to set up the library environment.
- Reads in the design-only database (generated by the command “write_db -data design”) to populate the design database within the Oasys-RTL tool in a library environment that has already been set up.

If the library has already been loaded, the command automatically skips the step of loading the library.

For backward compatibility, the read_db command can also read in legacy design ODBs (with ASCII library setup).

Examples

The following are some examples of using read_db and write_db commands in a sample Oasys-RTL session:

```
# Read in a legacy ODB with explicit read_library commands
[oasys-RTL]$ read_db db_old.odb

# Write out the library database
[oasys-RTL]$ write_db -data library lib.odb

# Write a new ODB with library and design information included
[oasys-RTL]$ write_db db_new1.odb

# Write a new ODB without any library information
[oasys-RTL]$ write_db -data design db_new2.odb

## New Oasys-RTL session

# Restore the complete DB
[oasys-RTL]$ read_db db_new1.odb

## New Oasys-RTL session

# Restore the library
[oasys-RTL]$ read_db lib.odb

# Restore just the design, since the library database is already loaded
[oasys-RTL]$ read_db db_new1.odb

# Read and display only the header information
[oasys-RTL]$ read_db -header_only db_new2.odb
```

Related Topics

Read Commands

write_db

read_def

Loads physical layout information from Design Exchange Format (DEF) files.

Usage

read_def ***def_files*** [-flattened] [-ignore_unknown_component]
[-match_component {true | false}] [-module] [-sections *objects*] [-skip *objects*]

Arguments

- ***def_files***

  Required. Specifies one or more DEF files to load.

- -flattened

  Optional. Reads a DEF for flattened (ungrouped) design, compares it to the unflattened (or hierarchical) netlist, and recreates the hierarchical DEF for use within the Oasys-RTL tool. Use this option for DEFs from place and route tools if the resulting DEF file has been flattened before output, converting hierarchical paths A/B/C to A_B_C. This option is not applied by default.

- -ignore_unknown_component

  Optional. Ignores components found only in the DEF but not in the design database. By default, the tool creates a blockage for these unknown components.

- -match_component {true | false}

  Optional. Matches a component that is found in the DEF but not in the netlist. If an instance (component) in the DEF is not found in the netlist, this option matches the unknown component with an unmatched instance in the parent hierarchy that is of the same cell as specified in the DEF. This option is applied by default.

- -module

  Optional. Specifies the DEF design name in the accompanying DEFs to be matched and applied to the corresponding sub-module names in the design.

- -sections *objects*

  Optional. Specifies that only the specified sections of DEF are read.

- -skip *objects*

  Optional. Specifies which objects to skip when reading in the DEF file.

Description

The read_def command reads the specified DEF files into the database. By default, none of the optional arguments are applied, except for the -match option. If the Tcl variable search_path is specified, files are searched for in any of the specified paths. Well ties, filler cells, and decap cells are treated as blockages during the placement (just like fixed macros). They are not overlapped by any other standard cells.

Use the -sections option to read particular sections of the DEF. Use the -skip option to skip particular sections of the DEF. The following objects are valid for the -sections and -skip options:

- blockages
- components
- gcellgrid
- groups
- nets
- pins
- placement_blockages
- regions
- routing_blockages
- rows
- specialnets
- tracks
- vias

When reading in hierarchical DEF files with the -module option, the top cell of each DEF is resolved by associating the “DESIGN_NAME” with a corresponding sub-module in the design. The top-level DEF file must be read first.

If the DEF file is compressed with an extension *.gz* or *.bz2,* it uncompresses the file using gunzip and bunzip2 respectively.

This command can read an encrypted file.

Examples

Example

The following example sets the search path and then reads a design, TOP, that has modules A, B, C. The corresponding DEF files are *top.def, A.def, B.def,* and *C.def*, respectively and are located in the *…/path/to/def* directory. The location of the hierarchical DEF files is resolved by the search path that is set to the directory where they reside.

```
% set search_path {../path/to/libs ../path/to/def}
% read_def top.def
% read_def -module A.def B.def
% read_def -module C.def
```

Example

This example reads only the blockages (both placement and routing) from the DEF called *top.def*:

```
% read_def -section blockages top.def
```

Example

This example reads the DEF, *top.def*, but skips the nets section:

```
% read_def -skip nets top.def
```

Related Topics

Read Commands

encrypt

write_def

read_explore_checkpoint

Loads all relevant metrics for a session at the specified point in the exploration flow.

Usage

read_explore_checkpoint {-post_synthesize | -post_optimize | -checkpoint *checkpoint*} [-id *id*]

Arguments

- -post_synthesize

  Loads the exploration results after the synthesize command.

- -post_optimize

  Loads the exploration results after the optimization command.

- -checkpoint *checkpoint*

  Loads the results at the specified user-defined checkpoint.

- -id *id*

  Used to load a specific DSE session. The *id* parameter refers to the Job ID listed in the -status report.

Description

This command loads all relevant metrics and/or a design for a session at the specified point in the flow. This command can be used to load a specific DSE session.

Examples

Example

The following example looks at the status of the DSE runs to find a Job ID, which is used in the second example.

```
% report_explore -status
Report exploration status:
-----+------------+------+-------+---------------+--------+--------+-----+
     |clock_period|lib_vt|voltage|Job_ids        |Status  |Hostname|PID  |
-----+------------+------+-------+---------------+--------+--------+-----+
1    |1.2ns       |hvt   |0.85v  |explore.0.0.0.1|finished|o14sa28w|16638|
2    |1.2ns       |hvt   |0.95v  |explore.0.0.0.0|finished|o14sa28w|16548|
3    |1.2ns       |lvt   |0.85v  |explore.0.0.1.1|finished|o14sa28w|16799|
4    |1.2ns       |lvt   |0.95v  |explore.0.0.1.0|finished|o14sa28w|16625|
```

Example

The next example loads the results after post synthesis for the id explore.0.0.0.0.

```
% read_explore_checkpoint -post_synthesize -id explore.0.0.0.0
info:    no design loaded
info: loading /design/oasys.explore/explore.0.0.0.0/end_synthesize.odb
starting at 00:00:03(cpu)/0:11:49(wall) 37MB(vsz)
extracting odb        ... finished at 00:00:03(cpu)/0:11:49(wall) 37MB(vsz)
             Write Date   : Tue, 19 May 2015 23:11:44 -0700
             Host         : o14sa28 (64bit)
             Tool Version : 15.1-b026 (31-31)
             Tool Date    : Tue May 19 13:06:43 PDT 2015
             Tool Build   : 36956.0-O
             Design Name  : sparc
             Comment      :
loading environment... finished at 00:00:03(cpu)/0:11:49(wall) 37MB(vsz)
loading upf        ... finished at 00:00:03(cpu)/0:11:49(wall) 37MB(vsz)
loading libraries  ... finished at 00:00:06(cpu)/0:11:52(wall) 64MB(vsz)
loading physical   ... finished at 00:00:06(cpu)/0:11:52(wall) 64MB(vsz)
checking libraries ... finished at 00:00:06(cpu)/0:11:53(wall) 64MB(vsz)
loading design     ... finished at 00:00:08(cpu)/0:11:54(wall) 220MB(vsz)
loading sdc        ... finished at 00:00:08(cpu)/0:11:55(wall) 221MB(vsz)
all done
```

Related Topics

Read Commands

write_explore_checkpoint

explore

scale_utilization

set_explore

report_explore

scale_clock

read_lef

Loads cell physical layout information from Library Exchange Format (LEF) files.

Usage

read_lef ***lef_files***

Arguments

- ***lef_files***

  Required. Specifies one or more LEF files to load.

Description

Reads the specified LEF files into the database. If the Tcl variable search_path is specified, files are searched for in any of those paths. The technology file that contains the routing layer information should be read in first. Read LEF files prior to using the synthesize command.

LEF files should be read in after Liberty (*.lib*) files. The MACROS (library cells) in the LEF files are matched up against the library cells previously read in through the read_library command. Any macro or pin of a macro in LEF that was not found among the library cells that were previously read in through the read_library command are created as physical only and cannot be instantiated or used for synthesis.

If the LEF file is compressed with an extension *.gz* or *.bz2* it uncompresses the file using gunzip and bunzip2 respectively.

This command can read an Oasys-RTL encrypted file.

Examples

Example

The following example sets the search path and reads in the *hivt.lef* LEF file.

```
% set search_path {/path/to/libs /path/to/more/libs}
% read_lef hivt.lef
INFO: read lef file: '/path/to/more/libs/hivt.lef'
```

Example

The following example shows how to read multiple LEF files.

```
% set macro_lefs {tech.lef ram.lef io.lef pads.lef osc.lef}

% read_lef $macro_lefs
```

Related Topics

Read Commands

read_library

encrypt

read_library

Loads library cell models defined in Liberty libraries.

Usage

read_library [-target_library '{'*targets*'}'] ***lib_files***

Arguments

- -target_library '{'*targets*'}'

  Specifies names of the target libraries to which this library is added. The {} brackets are needed only if there are multiple targets specified. If a specified target does not exist, it is created. This option specifies that the <lib_files> that are read in are added to the specified target libraries. The default target library is named *default*.

- ***lib_files***

  One or more Liberty (*.lib*) technology files that are read.

Description

Reads the specified Liberty technology files into the database. If the Tcl variable search_path is specified, the files are searched for in any of these paths. Load library files prior to using the synthesize command.

A target library contains all the library cells to be used during synthesis. These cells can come from different library groups in different Liberty files. A target library is created either by using this command's -target_library option or by using the set_target_library command.

A default target library is always created by the database at start up. If -target_library is not specified in the read_library command, the library is added to a target library named default.

The set_target_library command needs to be specified before the synthesize command, if you want to use a target library different than the default target library. The name of a target library (<targets>), if using -target_library, can be any alphanumeric string and does not have anything to do with the names in the Liberty files.

A library can belong to more than one target library. As an example, the cells of the library can be used for synthesis when any of the target libraries it belongs to is chosen for synthesis. Cells that are from any library can be instantiated in the RTL without belonging to the target_library being used for synthesis and optimization.

If the Liberty file is compressed with an extension *.gz* or *.bz2,* it uncompresses the file using gunzip and bunzip2 respectively.

This command can read an Oasys-RTL encrypted file.

Examples

Example

The following example reads a Liberty timing file for a macro.

```
% read_library pads_slow.lib
```

Example

The following example sets a search path for the library (*.lib*) files and reads the Liberty library file *hivt.lib* into the target library *highVt*:

```
% set search_path {/path/to/libs /path/to/more/libs}
% read_library hivt.lib -target_library {highVt}
INFO: read lib file: '/path/to/more/libs/hivt.lib'

set_target_library highVt
```

Related Topics

Read Commands

set_target_library

encrypt

read_ptf

Loads a PTF and sets the resistance and capacitance values per unit length for each layer.

Usage

read_ptf ***ptf_filename*** [-layer_map_file *filename*] [-process_lib *library_name*]
[-temperature *double*]

Arguments

- ***ptf_filename***

  Required. Specifies the relative or absolute path name of a PTF for the tool to read.

- -layer_map_file *filename*

  Optional. Specifies the relative or absolute path name of a file containing the mapping between layer names in the PTF and the technology LEF.

- -process_lib *library_name*

  Optional. Specifies which process library name to use when multiple process libraries are defined in the PTF file. If multiple libraries are defined and you do not specify the -process_lib option, the tool returns all of the library names and an error message.

- -temperature *double*

  Optional. Specifies the temperature at which the tool needs to perform RC extraction. In general, the temperature corresponds to either the user-specified or library default operating conditions.

Description

The read_ptf command reads and computes the per-unit length layer resistance and layer capacitance values of all routing layers from the PTF and technology LEF file. You must load the technology LEF file containing the layer information before reading the PTF.

The -layer_map_file option maps the layer names specified in the PTF to the layers specified in the technology LEF. The layer map file defines the mapping using the following format:

```
<PTF layer name> <technology LEF layer name>
```

Examples

Example 1: Read a PTF

This example loads an LEF technology file and then a PTF.

```
[oasys-RTL]$ read_lef $tech_file
[oasys-RTL]$ set_operating_conditions
[oasys-RTL]$ read_ptf $ptf_file
[oasys-RTL]$ report_layer_rc

--+------+------+-------+-------+------+----------+---------+-----+-------------
  |Layer |Direc-|Width  |Spacing|ohm/sq|ohm/um    |cap ff/um|ecap |cap/Å
  |Name  |tion  |       |       |      |          |         |ff/um|
--+------+------+-------+-------+------+----------+---------+-----+-------------
1 |M1    |V     |0.07000|0.14000|     0| 4.3228574| 0.133407|    0|1.3340699e-05
```

Example 2: Read a PTF with a Layer Map File

This example reads a PTF and a layer map file, *map.txt*. The layer map file specifies the mapping of the layers in the PTF and the layers in the technology LEF.

```
[oasys-RTL]$ read_lef $tech_file
[oasys-RTL]$ set_operating_conditions
[oasys-RTL]$ read_ptf -layer_map_file map.txt $ptf_file
## After layer mapping
[oasys-RTL]$ report_layer_rc

--+------+------+-------+-------+------+----------+---------+-----+-------------
  |Layer |Direc-|Width  |Spacing|ohm/sq|ohm/um    |cap ff/um|ecap |cap/Å
  |Name  |tion  |       |       |      |          |         |ff/um|
--+------+------+-------+-------+------+----------+---------+-----+-------------
1 |metal1|V     |0.07000|0.14000|     0| 4.3228574| 0.133407|    0|1.3340699e-05
```

Following are the contents of the *map.txt* file:

```
#PTF  Tech LEF
 M1   metal1
 M2   metal2
 M3   metal3
 M4   metal4
 M5   metal5
 M6   metal6
 TM1  metal7
 TM2  metal8
 UM1  metal9
 UM2  metal10
```

Related Topics

report_layer_rc

read_lef

read_saif

Reads the switching activity annotations on the nodes of the design. Switching activity and toggle counts in Switching Activity Interchange Format (SAIF) are passed to the tool using this command.

Usage

read_saif [-scope *scope*] [-verbose] ***saif_file***

Arguments

- -scope *scope*

  Specifies the context to be extracted from the SAIF file to match the top-level design in the tool. This is necessary when the SAIF is generated at a higher level of hierarchy above the current design in the tool.

- -verbose

  Outputs detailed information as the file is read.

- ***saif_file***

  Specifies the path to an SAIF file. Specify the complete path of the SAIF file or add it to the search_path variable.

Description

The information from the SAIF file is used to calculate dynamic (switching) power. In the absence of an SAIF file, default switching probabilities and toggle counts are considered.

Examples

In this example, consider a top-level design, “chip_1”, with a corresponding SAIF file, *chip_1.saif*. The “chip_1” design has a module called “cpu_1”. Only the cpu_1 module is synthesized in the Oasys-RTL tool. The -scope option is used to extract and apply only the switching activities for the cpu module from the SAIF:

```
% read_saif -scope cpu_1 chip_1.saif
```

Related Topics

Read Commands

read_sdc

Loads design constraints from a Synopsys Design Constraint (SDC) file.

Usage

read_sdc ***sdc_file*** [-echo] [-mode *mode*] [-verbose]

Arguments

- ***sdc_file***

  Required. Specifies the Synopsys design constraint file (SDC) to read. If the Tcl variable search_path is specified, the file is searched for in any of these paths.

- -echo

  Optional. Displays each of the SDC commands as it reads them. Otherwise, only the errors are displayed.

- -mode *mode*

  Optional. Specifies the mode of the design. The *sdc_file* is applied to the design under the given mode.

- -verbose

  Optional. Similar to -echo but prints out additional information on how timing exceptions have been interpreted.

Description

This command parses the specified *sdc_file*. The file is parsed by a traditional SDC parser, which means it only handles limited Tcl support (such as variable expansion, [] sub commands, as well as the set, list, and expr commands). If your SDC contains more generic Tcl constructs, you need to source your SDC file. After reading the file, read_sdc prints out a summary of error/warning/information messages. Any SDC file can also be sourced (using Tcl native source command). However, in this case any resulting message are printed immediately after it occurs.

Examples

Example

This example reads the SDC file called *my_design.sdc*:

```
% read_sdc my_design.sdc
```

Example

This example reads the SDC file called *my_design.sdc* and applies it to the fast mode:

```
% read_sdc -mode fast my_design.sdc
```

Related Topics

Read Commands

source

SDC Command Support

read_vcd

Loads value change information from a Value Change Dump (VCD) file.

Usage

read_vcd [-scope *scope*] [-verbose] [-start *start_time*] [-end *end_time*] ***vcd_file***

Arguments

- -scope *scope*

  Specifies the scope to be extracted from the VCD file to match the top of the design in the tool. The SAIF is at a higher level of hierarchy than the design in the tool.

- -verbose

  Outputs detailed information as the file is read.

- -start *start_time*

  Specifies the time step in the VCD file at which to start accumulating the switching activities in the design. The default is 0.

- -end *end_time*

  Specifies the time step in the VCD file at which to end accumulating switching activities in the design. The default is the last time step in the VCD file.

- ***vcd_file***

  Specifies the VCD file to read.

Description

This command reads a VCD file to gather switching activities in a design. This information is used to calculate dynamic (switching) power.

Examples

The following example reads a vcd file called *design.vcd*.

```
% read_vcd design1.vcd
```

Related Topics

Read Commands

read_verilog

Reads (or loads) one or more design source files in Verilog or System Verilog formats before the synthesize step.

Usage

read_verilog [-include *include_dirs*] [-define *macro_defines*] [-library *library*] [-sv] [-sv2005] [-sv2009] [-y *path*] [-f *file*] ***verilog_files***

Arguments

- -include *include_dirs*

  <include_dirs> is either a string or a Tcl list of strings representing include directories. Include directories are used to search for files that have been included in the Verilog files.

- -define *macro_defines*

  <macro_defines> is a string or a Tcl list of strings. Each string is of the form <name> or <name>=<val>. The first form defines a macro <name>. The second form defines a macro <name> having value <val>.

- -library *library*

  Specifies the library in which to store the parse tree. This option is useful when multiple same-named Verilog modules exist in the design. Two same-named modules or entities cannot exist in the same library, so the -library option allows multiple same-named modules to coexist. Such modules can be distinguished in the rtl (from vhdl only) using lib.module syntax.

- -sv

  Reads in SystemVerilog 2009 files.

- -sv2005

  Reads in SystemVerilog 2005 files.

- -sv2009

  Reads in SystemVerilog 2009 files. This option is not necessary if you specify -sv.

- -y *path*

  Specifies a directory to search for unknown modules. You can specify this argument multiple times to search multiple directories.

- -f *file*

  Specifies a text file containing additional arguments to pass to the read_verilog command. See Example 4 in the Examples section below.

- ***verilog_files***

  Specifies either a string or a Tcl list of strings representing Verilog input files.

Description

Reads one or more RTL Verilog files. If -sv, -sv2005, or -sv2009 is not specified, the tool assumes the input files are normal Verilog files.

Run the read_verilog command before the synthesize command.

This command can read an encrypted Oasys-RTL file.

Examples

Example 1

This example reads the Verilog design and then synthesizes it:

```
% read_verilog -include proj1/iu/rtl "proj1/iu/rtl/funcX.v
proj1/iu/rtl/funcY.v proj1/cu/rtl/funcZ.v"
% synthesize
```

Example 2

This example reads Verilog files of a top module that instantiates two different implementations of the “bot” module. Then, run the synthesize command.

This is the *test_bar.v* Verilog file:

```
module mid_bar(output q, input d1, d2, d3);
   wire tmp;
   bot u1(.q(tmp), .d1(d1), .d2(d2));
   bot u2(.q(q), .d1(tmp), .d2(d3));
endmodule

module bot(output q, input d1, d2);
   assign q = d1 ~^ d2;
endmodule
```

This is the *test_foo.v* Verilog file:

```
module mid_foo(output q, input d1, d2, d3);
wire tmp;
   bot u1(.q(tmp), .d1(d1), .d2(d2));
   bot u2(.q(q), .d1(tmp), .d2(d3));
endmodule

module bot(output q, input d1, d2);
   assign q = ~(d1 & d2);
endmodule
```

This is the *top.v* Verilog file:

```
module top(output q, input d1, d2, d3, d4, d5);
   wire tmp;
   mid_foo b2(.q(q), .d2(d4), .d3(d5));
   mid_bar b1(.q(tmp), .d1(d1), .d2(d2));
endmodule
```

Based on the library in which the common module “bot” has been bound, the synthesize command uses that definition while binding the instance. Following are the commands:

```
% read_verilog -library foo test_foo.v
% read_verilog -library bar test_bar.v
% read_verilog top.v ;# instantiates both mid_foo and mid_bar
% synthesize
starting synthesize at 00:00:05(cpu)/0:01:35(wall) 68MB(vsz)
info:    design has multiple top modules 'mid_bar' ((/top.v:1)[7])  [VLOG-406]
info:    design has multiple top modules 'mid_foo' ((test_foo.v:1)[7])  [VLOG-406]
info:    design has multiple top modules 'mid_bar' ((/test_bar.v:1)[7])  [VLOG-406]

info:    binding instance 'u1' of module 'bot' in current library: 'foo.bot' ((/
test_foo.v:3)[7])  [VLOG-629]
info:    synthesizing module 'foo_bot' ((/test_foo.v:7)[7])  [VLOG-400]
info:    done synthesizing module 'foo_bot' (1#5) ((/test_foo.v:7)[7])  [VLOG-401]
info:    binding instance 'u2' of module 'bot' in current library: 'foo.bot' ((/
test_foo.v:4)[7])  [VLOG-629]
info:    done synthesizing module 'mid_foo' (2#5) ((/test_foo.v:1)[7])  [VLOG-401]
info:    synthesizing module 'mid_bar' ((/test_bar.v:1)[7])  [VLOG-400]
info:    binding instance 'u1' of module 'bot' in current library: 'bar.bot' ((/
test_bar.v:3)[7])  [VLOG-629]
info:    synthesizing module 'bar_bot' ((/test_bar.v:7)[7])  [VLOG-400]
info:    done synthesizing module 'bar_bot' (2#5) ((/test_bar.v:7)[7])  [VLOG-401]
info:    binding instance 'u2' of module 'bot' in current library: 'bar.bot' ((/
test_bar.v:4)[7])  [VLOG-629]
info:    done synthesizing module 'mid_bar' (3#5) ((/test_bar.v:1)[7])  [VLOG-401]
info:    done synthesizing module 'mid_bar__parameterized1' (depth 1) (2#6) ((/
test_bar.v:1)[7])  [VLOG-401]
finished synthesize at 00:00:18(cpu)/0:21:41(wall) 129MB(vsz)
```

Example 3

This example defines the macro, EO_STAGE, using the -define option.

```
++++++++++++++++++++++++++
set type_range "EO_STAGE"
read_verilog -define $type_range test.v
++++++++++++++++++++++++++

test.v
++++++++++++++++++++++++++
.......
genvar i;
generate
  for(i=1; i<=FFLEVEL; i=i+1) begin
        always @(posedge clk1) begin
                `ifdef ALL_STAGE
                        instorage[i] <= ~instorage[i-1] ;
                `elsif EO_STAGE
                        if ( i%2 == 0)
                                instorage[i] <= instorage[i-1] ;
                        else
                                instorage[i] <= ~instorage[i-1] ;
                `elsif NO_STAGE
                        instorage[i] <= instorage[i-1] ;
                `endif
        end
  end
endgenerate
```

Example 4

This example shows how to specify additional command line arguments with the -f argument:

File command.f contents

```
-sv file0.v
-sv file1.v
    file3.v -define {{PARAM1} {100}}
```

Executing read_verilog with arguments from *command.f*:

```
read_verilog -f command.f
```

This is functionally equivalent to:

```
read_verilog -sv file0.v
read_verilog -sv file1.v
read_verilog     file3.v -define {{PARAM1} {100}}
```

Related Topics

Read Commands

synthesize

encrypt

read_vhdl

Reads (or loads) one or more design source files in VHDL format before the synthesize step.

Usage

read_vhdl [-library {*lib* | *lib2=lib1*}] [-vhdl87 | -vhdl2008] [-f *file*] ***vhdl_files***

Arguments

- -library {*lib* | *lib2=lib1*}

  Optional. Specifies the library in which to store the VHDL parse tree. If you specify the *lib2=lib1* syntax format, the VHDL parse tree is stored in the *lib1* library, and *lib2* becomes an alias for library *lib1*. If you do not specify a library, Oasys-RTL stores the VHDL parse tree in the library defined by *default*.

- -vhdl87

  If specified, read_vhdl uses the this dialect. By default, the vhdl-1993 dialect is used.

- -vhdl2008

  If specified, read_vhdl uses this dialect. By default, the vhdl-1993 dialect is used.

- -f *file*

  Specifies a text file containing additional arguments to pass to the read_vhdl command. See Example 4 in the Examples section below.

- ***vhdl_files***

  Specifies either a string or a Tcl list of strings representing VHDL input files.

Description

Reads one or more RTL VHDL files into the database.

The read_vhdl command must be run before the synthesize command.

This command can read an encrypted Oasys-RTL file.

Examples

Example 1

This example reads the VHDL design my.vhdl, stores the VHDL parse tree in library *my_fast_lib*, and then synthesizes it:

```
% read_vhdl -library my_fast_lib my.vhdl
% synthesize
```

Example 2

This example reads the VHDL design *my.vhdl*, stores the VHDL parse tree in the library, and then synthesizes it:

```
% read_vhdl my.vhdl
% synthesize
```

Example 3

This example reads the VHDL design *mypack.vhdl*, stores it in VHDL library liby, and then defines libx to be an alias for liby.

```
% read_vhdl -library libx=liby mypack.vhdl
```

After reading the VHDL, mypack can be referenced within the VHDL code with either of the following:

```
library liby;
use liby.mypack.all
```

or

```
library libx;
use libx.mypack.all
```

Example 4

This example replaces multiple calls to read_vhdl with a single call. The parameters are passed with the -f *file* argument.

The text file contains the arguments to pass to the read_vhdl command:

```
# Contents of command_file.f
mypack.vhd
sub.vhd
top.vhd
```

The command accepts the file with the -f argument:

```
read_vhdl -f command_file.f
```

This is equivalent to entering the following commands:

```
read_vhdl mypack.vhd
read_vhdl sub.vhd
read_vhdl top.vhd
```

Related Topics

Read Commands

synthesize

encrypt

Chapter 3
Flow Commands

The main synthesis design flow is controlled by the synthesize and optimize commands.

Table 3-1. Flow Commands

Command	Description
synthesize	Synthesizes the RTL to the specified target technology.
optimize	Optimizes the loaded design to meet the timing, power, and physical constraints set on the design.
map_to_multibit	Performs mapping of single-bit registers to corresponding multi-bit registers.
retime	Performs register retiming optimization.

synthesize

Synthesizes the RTL to the specified target technology.

Usage

synthesize [-gate_clock *bool*] [-map_to_scan] [-module *module*]
[-parameters '{'*parameter_list*'}'] [-extract_pg_ports]

Arguments

- -gate_clock *bool*

  Inserts clock gates for load-enable flip-flops. The default value is true. See set_clock_gating_options.

- -map_to_scan

  Maps to scan flip-flops. The scan enable is always tied to its inactive state. The scan input is tied back to the scan output of the flip-flop. If -map_to_scan is not specified, the synthesize command does not use scan flops for register inference. If you synthesize using a library with only scan flops and do not provide this option, synthesize outputs a SYN-102 message:

  ```
  "no appropriate flip-flop".
  ```

  Oasys-RTL infers a scan cell from a liberty library file based on the test_cell attribute on the library cell. This attribute typically denotes the scan in, enable, and out signals using the signal_type attribute on the pin.

- -module *module*

  Synthesizes the specified user module only. The default is to synthesize all modules.

- -parameters '{'*parameter_list*'}'

  Modifies the defined Verilog parameters or VHDL generics with new values. The <parameter_list> must be enclosed in {} brackets. See example below.

- -extract_pg_ports

  Specifies to extract power/ground ports. This option extracts the power supply ports and block-level connections that already exist in the RTL and creates corresponding power supply ports in the database already constrained by UPF commands.

Description

Perform RTL optimizations and synthesize the RTL to the specified target technology. The target library is set using set_target_library. If no target library is set, the default target library is used.

Prior to running this command, read the technology library and the HDL design data.

All user-instantiated library cells are preserved (regardless of which library they come from), except for the following situations:

1. If the cell is a load-enable flip-flop and -gate_clock is not disabled, the Oasys-RTL tool attempts to remap such flip-flops to try to gate its clock.
2. If -map_to_scan was specified, the Oasys-RTL tool attempts to remap non-scan flip-flops to scan flip-flops. All other cells are preserved.

During synthesis, any undriven input pins are tied to zero.

Examples

Example

This example reads in the appropriate files, performs synthesis, and then optimizes the design:

```
% read_library myTechLib.lib
% read_lef myPhysLib.lef
% read_verilog myRTL.v
% synthesize -module myMod
% read_sdc myMod.sdc
% read_def myModFP.def
% optimize
```

Example

This example shows how to change the Verilog parameter called width to a value of 4 prior to synthesizing. Consider the Verilog module with parameters given below:

```
module comparator(x, y);
parameter width = 8;
input [width-1:0] x;
output [width-1:0] y;
...
```

By default, after the synthesize command, you get a module with 8-bit input and output ports. The -parameters option lets you override the default parameter values specified in the module:

```
% synthesize -module comparator -parameters { {width 4} }
```

The next example shows how to change multiple Verilog parameters prior to synthesizing:

```
% synthesize -module fir_filter -parameters { {width 8} {taps 32} ... }
```

Related Topics

Flow Commands

set_clock_gating_options

set_parameter

set_dont_scan

set_target_library

read_library

read_lef

read_verilog

read_sdc

read_def

optimize

optimize

Optimizes the loaded design to meet the timing, power, and physical constraints set on the design.

Usage

optimize [-place_only][-virtual]

Arguments

- -place_only

  Specifies to only do placement.

- -virtual

  Specify this option to check if a design can close timing by ignoring any potential physical problems. This causes optimization to only do a virtual placement. Virtual optimization can be followed by a regular optimization. However, if the timing after a virtual placement is not good enough, then there is no use in continuing with full optimization.

Description

Optimizes the design by trying to address timing issues in the design's physical context. The result is a globally placed design, optimized for timing, area, congestion, and power. The optimize command attempts to honor constraints. Some constraints may not be met due to unrealistic or conflicting values. The area cannot be constrained; the goal is to optimize for minimum area.

The input to this command is the design in technology gates, which is the output of the synthesize command. You can use the optimize command with no options set. Use the set_design_effort command to specify the effort levels for leakage optimization. By default, leakage optimization is not performed.

Examples

This is an example of a script to map and optimize a design from RTL to placed gates:

```
% read_library myTechLib.lib
% read_lef myPhysLib.lef
% read_verilog myRTL.v
% synthesize
% read_sdc myDesign.sdc
% read_def myFloorplan.def
% optimize
```

Related Topics

Flow Commands

set_design_effort

read_library

read_lef

read_verilog

synthesize

read_sdc

read_def

write_def

write_verilog

map_to_multibit

Performs mapping of single-bit registers to corresponding multi-bit registers.

Usage

map_to_multibit [-map_to_scan]

Arguments

- -map_to_scan

  Specifies that the tool converts single-bit non-scan registers to multi-bit scan registers.

Description

This command maps banks of single-bit registers to equivalent multi-bit registers. By default, it maps banks of single-bit non-scan registers to multi-bit non-scan registers and single-bit scan registers to corresponding multi-bit scan registers. With the -map_to_scan option, this command maps single-bit non-scan registers to multi-bit scan registers.

The multi-bit packing uses the implicit neighborhood property associated with RTL partitions to pack registers. It maps single-bit registers that are close together into the same multi-bit register.

The timing-driven mapping ensures that the multi-bit mapping step does not worsen the slack. It is recommended that you run this command after optimization. You can customize the multi-bit packing candidates and the packing strategy with the set_map_to_multibit and multi-bit grouping commands.

At the conclusion of executing the map_to_multibit command, Oasys generates a summary report. The following report is an example of the Multi-Bit summary:

```
info:    multibit mapping summary  [MBIT-100]
  Total number of single-bit FFs in design: 36946
    number of single-bit FFs marked as to be retimed: 0
    number of single-bit FFs with no pin compatible multi-bit libcells:241
    number of single-bit FFs marked as dont_touch, size_only, dont scan: 0
    number of single-bit FFs marked as set_map_to_multibit false: 0
    number of single-bit FFs that are shift registers: 0
    number of single-bit FFs excluded due to timing: 0
    number of single-bit FFs that are non-scan FFs: 0
    number of single-bit FFs that are scan FFs: 36946
  Total number of single-bit FFs eligible to be packed: 36705
    number of single-bit FFs packed: 36608
    number of single-bit FFs failed to pack due to timing: 32
    number of initial multi-bit instances: 0
    number of multi-bit instances created: 9208
    number of 2-bit instances created: 112
    number of 4-bit instances created: 9096
    number of 1-bit instances remaining: 338
    number of multi-bit groups defined: 0
    number of single-bit FFs packed in multi-bit groups: 0
  Total area saved: 11097.00squm
  Total leakage saved: 9.413uW
  Multi-bit FF packing percentage: 99.7% (w.r.t eligible candidates)
```

The following describe the report categories of ineligible flip-flops for which the command provides statistics:

- **Single-bit FFs marked as to be retimed —** The tool does not map flip-flops that you enable for retiming. The set_retime_module command enables a module for retiming.
- **Single-bit FFs with no pin-compatible multi-bit library cells —** The tool does not map flip-flops for which pin-compatible multi-bit library cells do not exist.
- **single-bit FFs marked as dont_touch, size_only —** The tool does not map flip-flops for which you have set either the dont_touch or size_only attribute to true.
- **single-bit FFs marked as set_map_to_multibit false —** The tool does not map flip-flops for which you have set the set_map_to_multibit attribute to false.
- **single-bit FFs excluded due to timing —** The tool does not pack flip-flops if doing so degrades timing.
- **single-bit FFs that are non-scan FFs —** If your library only contains scan cells and you do not use the -map_to_scan with the synthesize command, the tool does not map the non-scan flip-flops in the design. A non-zero value for this statistic may alert you that you intended but forgot to use the -map_to_scan with the synthesize command.

Note

For multi-bit mapping to scan cells, ensure that your library contains scan cells that are compatible with the flip-flops in your design.

The following describe the report categories of eligible flip-flops for which the command provides statistics:

- **single-bit FFs packed —** The tool reports the total number of single-bit flip-flops that the tool successfully packed into compatible multi-bit flip-flops.
- **multi-bit instances created —** The tool creates multi-bit instances when it successfully packs flip-flops during multi-bit mapping. The tool creates a multi-bit instance only if it can fully pack the multi-bit instance with compatible single-bit flip-flops.
- **n-bit instances created —** The tool can use multi-bit flip-flops of a variety of widths depending on what is available in the library. The command reports the number of multi-bit instances created for each width.
- **1-bit instances remaining —** The tool may not pack all single-bit flip-flops that are eligible for multi-bit mapping. Flip-flops remain if insufficient compatible single-bit flip-flops exist to fully pack a multi-bit instance. A partially packed multi-bit instance is not efficient enough to create a multi-bit instance. Flip-flops may also remain due to multi-bit group constraints.
- **multi-bit groups defined —** The tool reports the number of user-defined multi-bit groups. The tool only packs members of the same group together. The create_multibit_group command defines multi-bit groups.
- **single-bit FFs packed in multi-bit groups —** The tool may not be able to pack all of the eligible flip-flops that you assigned to multi-bit groups. The tool does not pack together flip-flops that are assigned to different multi-bit groups. In the case of unassigned flip-flops, the tool only packs unassigned flip-flops with other unassigned flip-flops.

The “Total number of single-bit FFs in design” equals the sum of each category of ineligible flip-flops and the “Total number of single-bit flip flops eligible to be packed”.

The “Multi-bit FF packing percentage” is the number of flip-flops that the tool successfully packed divided by the number of eligible flip-flops.

Examples

This example maps single-bit registers to scan multi-bit registers.

```
% synthesize -map_to_scan
% optimize
% map_to_multibit
```

This example maps single-bit non-scan registers to multi-bit scan registers.

```
% synthesize
% optimize -virtual
% map_to_multibit -map_to_scan
% optimize
```

Related Topics

Flow Commands

create_multibit_group

assign_to_multibit_group

remove_from_multibit_group

remove_multibit_group

set_map_to_multibit

retime

Performs register retiming optimization.

Usage

retime

Arguments

None.

Description

Register retiming optimization moves existing pipeline registers through deep cones of combinational logic such as wide multipliers and dividers within the retiming candidate modules to achieve the timing, area, power goals. Registers tagged with set_dont_retime are not moved during retiming. Retiming candidate modules are tagged using the set_retime_module command. If no module is tagged using by the set_retime_module command, the retime command stops with an error. The retime command must be run after the optimize command.

Examples

The following example specifies that all registers in the mod1 module are retimed, except for the reg1 register.

```
% ...
% synthesize
% set_retime_module mod1
% set_dont_retime mod1Instance/reg1
% read_sdc ...
% optimize
% retime
% ...
```

Related Topics

Flow Commands

synthesize

set_retime_module

set_dont_retime

read_sdc

optimize

report_retime

Chapter 4
Physical Commands

The physical commands are used to create or modify the design, including floorplanning.

Table 4-1. Physical Commands

Command	Description
add_power_guides	Specifies the domain and regions where level shifters and isolation cells can be placed by the Oasys-RTL tool.
assign_macros_to_edges	Assigns macros or macro groups to specific edge locations.
assign_pins	Restricts the floorplanning process by confining the specified pins to a particular side of the chip or block.
assign_to_group	Assigns instances to a group for placement.
assign_to_region	Assigns instances to a region that can be placed at a specific location on the die.
config_floorplan	Controls the macro placement strategies for floorplanning.
count_route_layer	Returns the maximum number of routing layers in the physical library.
create_blockage	Creates a rectangular blockage area that limits the available area for placement, or prevents the placement of macros, RTL partitions, or cells.
create_chip	Specifies the layout of the chip.
create_group	Creates a placement group. The tool places members of a group close together.
create_halo	Creates a placement halo for macros.
create_macro_blockages	Creates blockages around specified macros.
create_macro_groups	Creates macro groups based on connectivity, logical hierarchy, or thresholds for area or count.
create_macro_model	Creates a macro model cell to model a black box.
create_region	Creates a region for the placement of specific instances.
delete_physical	Clears the physical information that you previously created or loaded in the session.

Table 4-1. Physical Commands (cont.)

Command	Description
floorplan	Creates the die and performs macro and pin placement. Use when fine-tuning macro and pin placement before placing RTL partitions.
get_core_boundary	Returns the coordinates of the core boundary of the current floorplan.
get_groups	Generates a list of all groups, or searches for specific groups by name.
get_max_route_layer	Returns the maximum number of routing layers available for physical optimization.
get_regions	Generates a list of all regions, or searches for specific regions by name.
get_route_layer_direction	Gets the routing direction of the specified layer.
place_instance	Places the specified instance(s).
place_pins	Places I/O pins and physical module pins.
place_port	Provides the ability to preplace (fix) an I/O port of the design.
refine_macro	Modifies the floorplan of a macro instance by fixing, unfixing, or changing the orientation.
remove_blockage	Removes a placement blockage constraint that you previously created or loaded in the session.
remove_from_group	Removes instances from a placement group.
remove_from_region	Removes an instance from a placement region.
remove_group	Deletes the specified placement group.
remove_halo	Deletes halos from a design.
remove_macro_blockages	Removes macro blockages created with the create_macro_blockages command.
remove_macros_from_edges	Removes edge constraints from macro instances or macro groups.
remove_region	Deletes a placement region.
set_max_route_layer	Sets the maximum number of routing layers available for physical optimization.
set_route_layer_direction	Sets the routing direction of one or more layers.
set_route_layer_max_usage	Sets the fraction of the specified routing layer that can be used for routing.

Table 4-1. Physical Commands (cont.)

Command	Description
set_route_layer_spacing	Overwrites the default spacing provided in the technology LEF file for a given layer.
update_congestion	Recalculates the congestion estimates.

add_power_guides

Specifies the domain and regions where level shifters and isolation cells can be placed by the Oasys-RTL tool.

Usage

add_power_guides ***domain_name*** [-guides *region_names*]

Arguments

- ***domain_name***

  Specifies the power domain that the guide is contained in.

- -guides *region_names*

  Optionally specifies regions where level shifters or isolation cells can be placed. Currently the regions must be of FENCE (DEF) type.

Examples

Example

This command specifies that level shifters and isolation cells can be placed in the previously defined FENCE regions, LS_Sp_Bot and LS_Sp_Top. These guides are contained in the PD_SparcCores domain.

```
% add_power_guides PD_SparcCores -guides {LS_Sp_Bot LS_Sp_Top}
```

Example

This example specifies that level shifters and isolation cells can be placed in the previously defined FENCE regions, LS_Default_Top, LS_Default_Bot, LS_Default_TL, LS_Default_TR, LS_Default_BL, and LS_Default_BR. These guides are contained in the PD_Default domain.

```
% add_power_guides PD_Default \
   -guides {LS_Default_Top LS_Default_Bot LS_Default_TL \
   LS_Default_TR LS_Default_BL LS_Default_BR}
```

To create a region, use the create_region command.

Related Topics

Physical Commands

assign_macros_to_edges

Assigns macros or macro groups to specific edge locations.

Usage

assign_macros_to_edges { -groups '{' *group_name ...* '}' |
-sequence '{{' *group_1_name ...* '}' '}'*group_2_name...* '}' ...'}' }
{-edges *edge* [:*edge_section*] ...} [-align [begin | end]] [-exclusive]
[-instances *instance_name ...*]

Arguments

- { -groups '{'*group_name ...* '}'

  Specifies macro groups that the tool places entirely in the location specified with the -edges option. Members of a group are not required to be placed in any particular order.

  For example, {A B C} does not require any placement ordering between the A, B, and C groups.

  The assignment of groups to the specified edge location is a hard directive. The placement algorithm puts a high cost penalty on placing the groups outside of the edge location.

  This option is required if the -sequence option is not used. It cannot be used with the -sequence option.

  Create macro groups using the create_group or the create_macro_groups command.

- -sequence '{{' *group_1_name ...* '}' '{'*group_2_name...* '}' ...'}'

  Specifies the sequence in which the tool places the groups of macros, beginning at the location specified with the -edges option and in the direction specified with the -align option.

  Specify the groups as a list of tuples. Each tuple can consist of one or more groups. The sequence of the tuples specifies the ordering of the tuples during macro placement. The tool does not necessarily place members within the tuple adjacent to each other. It must maintain the order among groups that are separated by the braces ({ }).

  For example, {{A} {B} {C}} specifies that the members of group A must be followed by the members of group B, and the members of group B must be followed by the members of group C. In the case of {{A B C} {D E}}, the tool places {A B C} and {D E} so that the members of the D and E groups follow the members of the A, B, and C groups. The tool does not impose any order within the tuples.

  A sequence is a soft directive that starts or ends at the edge specified with the -edges option.

  This option is required if the -groups option is not used. It cannot be used with the -groups option.

- -edges *edge* [:*edge_section*] ...

  Specifies the edges or edge subsections on which the tool places the specified groups. To specify an entire edge, simply use the edge number. Edge numbering begins at zero on the leftmost edge and increments in a clockwise direction.

  Optionally specify a subsection of an edge with the edge number followed by a colon and the specification of the subsection. For example, "-edge 1:2/6".

  The -edges option is required.

  You can specify multiple contiguous subsections of an edge or adjacent edges with the -groups or -instances option. If you specify multiple contiguous subsections, the tool spreads the groups over all specified subsections. A group of macros can occupy one or more subsections based on the edge specification and the -align option. The value of the -align option dictates the direction of macro packing for the -sequence option.

- -align [begin | end]

  Aligns the beginning or end of a sequence of groups during macro packing. The direction of macro packing depends on whether you specify begin or end.

  When begin is used, the tool places the first group of the sequence at the anchor position specified with the -edges option. The packing aligns left to the subsection and continues to adjacent subsections in a clockwise direction until the end of the edge or another anchor constraint is encountered.

  When end is used, the tool places the last group of the sequence at the anchor position specified with the -edges option. The packing aligns right to the subsection and continues to adjacent subsections in a counterclockwise direction until the start of the edge or another anchor constraint is encountered.

  In both cases, where the packing ends is flexible. However, macro placement beyond the edge or other anchor constraint boundary incurs a high penalty in the placement cost algorithm.

  The -align option is optional. The default is begin. This option can only be used with the -sequence option and cannot be used with the -groups or -instances option. It cannot be used with multiple edges or subsections.

- -exclusive

  Gives the highest priority to placing the groups in the location specified with the -edges option. Members of other groups cannot be placed in the that location.

  The exclusive assignment of groups is a hard directive. Placement of the macros outside of the subsections has a high penalty in the cost algorithm. The tool lowers the priority of placing macros that are not specified as exclusive.

  The -exclusive option is optional.

- -instances *instance_name ...*

  Specifies the instances that the tool must place in the location specified with the -edges option. The instances do not need to be assigned to a group. This option can be used with the -groups option but not with the -sequence option.

  The -instances option is optional.

Description

Controls the placement of groups of macros to specific edge locations. By default, the tool packs macros in a clockwise direction beginning at the edge location specified with the -edges option. If the "-align end" option is used, the tool packs the macros in the specified sequence in a counterclockwise direction.

Subsections are parts of an equally divided edge and are specified by a number indicating which division, followed by a slash and the total number of divisions. For example, if an edge is divided into three thirds, 1/3 refers to the first third of the edge, 2/3 refers to the middle third of the edge, and 3/3 refers to the last third of the edge. Edges can have different numbers of subsections. See Figure 4-1 for an illustration of edge and subsection numbering. Create groups using the create_group or create_macro_groups command.

The -groups and -sequence options are mutually exclusive. If both options are specified, the -groups option is ignored.

The -sequence and -exclusive options cannot be used together. If they are both specified, the -exclusive option is ignored. The -sequence and -instances options also cannot be used together.

If multiple assign_macros_to_edges commands are issued, the last invocation overwrites the previous one.

Figure 4-1. Edge Section Numbering Example

Examples

Example 1: Assign a Macro Group to an Edge

This example places high priority on assigning the *G1* user-created macro group to edge number 3.

```
% create_group G1 -type macro -instance [get_cells -hier * -filter \
   "is_hard_macro==true && full_name =~pipe1/*"]

% assign_macros_to_edges -groups G1 -edges 3 -exclusive
```

Example 2: Assign a Macro Group to Two Subsections of an Edge

This example assigns the *G2* user-created macro group to two sections of edge number 2: the first third and the middle third. The tool must place at least one part of the group in each section. The tool cannot place the entire group in only one of the subsections.

```
% create_group G2 -type macro -instance [get_cells -hier * -filter \
   "is_hard_macro==true && full_name =~pipe0/*"]

% assign_macros_to_edges -groups G2 -edges {2:1/3 2:2/3}
```

Example 3: Assign an Instance to a Subsection of an Edge

This example assigns the mem_0_0 macro to the third quarter of edge number 5.

```
% assign_macros_to_edges -instances mem_0_0 -edges 5:3/4
```

Example 4: Assign Macro Groups in Sequence to an Edge Subsection

This example specifies that the tool places the *{A B C}* and *{D E}* tuples (groups of macro groups) adjacent to each other in the specified sequence. The tool places the *{D E}* tuple following the *{A B C}* tuple. The tool does not impose any order for the macro groups that are within the tuples. The macro packing begins on the second sixth of edge 1 because the -align option is not specified and thus defaults to “-align begin”.

```
% assign_macros_to_edges -sequence {{A B C} {D E}} -edge 1:2/6
```

Example 5: Assign a Soft Constraint to a Single Group of Groups

This example uses a soft constraint to begin the macro packing of the *{A B C D E}* group of groups at the second sixth of edge 1. The tool does not impose any order among the *A*, *B*, *C*, *D*, and *E* groups.

```
% assign_macros_to_edges -sequence {{A B C D E}} -edges 1:2/6 \
   -align begin
```

Example 6: Assign Groups in Sequence that Ends at a Subsection

This example assigns the *A* and *C* groups in sequence. The tool place the *C* group at the fourth fifth of edge 1. The packing of the *A* group continues adjacent to the *C* group in a counterclockwise direction.

```
% assign_macro_to_edges -sequence {{A} {C}} -edges 1:4/5 -align end
```

Example 7: Exclusively Assign Groups in Sequence

This example exclusively assigns the *A* and *{B C}* groups to a location beginning at the first sixth of edge 1. The *{B C}* group follows the *A* group. The tool ensures that no other macro is placed in the subsections that A, B, and C occupy.

```
% assign_macros_to_edges -sequence {{A} {B C}} -edges 1:1/6 -align \
   begin -exclusive
```

Example 8: Assign Groups to a Subsection Using a Hard Constraint

This example is a hard assignment of the *A* and *C* groups of macros to the first half of edge 1.

```
% assign_macros_to_edges -groups {A C} -edge 1:1/2
```

Example 9: Exclusively Assign Groups to a Subsection Using a Hard Constraint

This example is an exclusive hard assignment of the *A* and *C* groups of macros to the first half of edge 1. The tool does not place macros that do not belong to the *A* and *C* groups in the subsection.

```
% assign_macros_to_edges -groups {A C} -edge 1:1/2 -exclusive
```

Example 10: Assign Macro Instances and Groups to a Subsection

This example assigns the *M1* and *M2* instances and the *A* and *C* groups to the middle fifth of edge 2.

```
% assign_macros_to_edges -instances {M1  M2} -groups {A C} -edges 2:3/5
```

Example 11: Assign Groups to a Contiguous Set of Subsections

This example assigns the *A*, *B*, *C*, *D*, and *E* groups to the following contiguous subsections: the second, third, and fourth sixths of the edge 1.

```
% assign_macros_to_edges -groups {A B C D E} -edges {1:2/6  1:3/6  1:4/6}
```

Example 12: Exclusively Assign Groups and Instances to Multiple Subsections.

This example exclusively assigns the *A*, *B*, *C*, *D*, and *E* groups and the *M1* and *M2* instances to the following contiguous subsections: the last third of edge 1 and the first third of edge 2. The tool does not place macros that do not belong to the *A, B, C, D,* and *E* groups or the *M1* and *M2* instances in the contiguous subsections.

```
% assign_macros_to_edges -groups {A B C D E} -instances {M1 M2} \
   -edges {1:5/6  1:6/6  2:1/3} -exclusive
```

Related Topics

Physical Commands

create_group

remove_macros_from_edges

assign_pins

Restricts the floorplanning process by confining the specified pins to a particular side of the chip or block.

Usage

assign_pins [-edge *edge_id* [:*partition_id*]] [-layer *layer_name*]
[-side {left | bottom | right | top}] [-pins *pin_list*]

Arguments

- -edge *edge_id* [:*partition_id*]

  Specifies an edge or edge partition of a rectilinear floorplan. See "Identifying Edges and Edge Partitions" on page 23 for details on how to identify a particular edge or edge partition.

- -layer *layer_name*

  Specifies the layer on which to place the pins.

- -side {left | bottom | right | top}

  Specifies the side of the chip or block to which the pins are restricted. The default is left, if no side is specified.

  Note

  This argument will be deprecated in a future release. Use the -edge option instead.

- -pins *pin_list*

  Specifies a list of pin names.

Examples

The following example restricts the BS_addr_0* pins to the second quarter of the third side (right side for a rectangular core):

```
% assign_pins -edge {3:2/4} -pins [get_ports BS_addr_0*]
```

Related Topics

Physical Commands

assign_to_group

Assigns instances to a group for placement.

Usage

assign_to_group ***string*** [-instance *string*]

Arguments

- ***string***

  Specifies the names of a group.

- -instance *string*

  Specifies the name of the instances to add to the group.

Description

This command assigns instances to a group that was created with create_group or imported from DEF as a GROUP construct.

Examples

The following example assigns CTL instances the A1 group:

```
% assign_to_group A1 -instances CTL
```

Related Topics

Physical Commands

report_groups

create_group

remove_group

get_groups

assign_to_region

Assigns instances to a region that can be placed at a specific location on the die.

Usage

assign_to_region ***region_name*** [-instance *inst_list*] [-group *group_name*]

Arguments

- ***region_name***

  The name of the region.

- -instance *inst_list*

  A list of instances or instance names that are to be placed in the region specified.

- -group *group_name*

  Specifies a group name to be assigned to the specified region.

Examples

The following example adds instance i0 to region r0:

```
% assign_to_region r0 -instance i0
```

Related Topics

Physical Commands

remove_from_region

create_region

config_floorplan

Controls the macro placement strategies for floorplanning.

Usage

config_floorplan [-create_blockages_for_pin_access {true | false}]
[-pack_similar_macros {true | false}] [-report]

Arguments

- -create_blockages_for_pin_access {true | false}

  Directs the tool to infer macro-only blockages if any congestion exists around I/O pins. These blockages improve pin access. They are persistent and have names with the __RTMOB prefix. The default is false.

- -pack_similar_macros {true | false}

  Directs the macro placement engine to group and place similar macros resulting in a regular packing pattern. The default is true.

- -report

  Reports the current settings for the command options.

Description

This command enables automatic macro grouping based on connectivity analysis and logical hierarchy for macro placement. It also enables congestion-driven macro packing to relieve any congestion caused by macros blocking access to I/O pins.

Use this command in conjunction with the create_group and create_macro_groups commands for macro placement.

Examples

Example 1

The following example removes the default directive to pack similar macros in a regular pattern.

```
% config_floorplan -pack_similar_macros false
```

Example 2

The following example directs the macro placement engine to relieve any congestion around the I/O pins by creating macro-only blockages.

```
% config_floorplan -create_blockages_for_pin_access true
```

Related Topics

Physical Commands

create_group

create_macro_groups

floorplan

optimize

count_route_layer

Returns the maximum number of routing layers in the physical library.

Usage

count_route_layer

Arguments

None.

Description

Returns the maximum number of routing layers in the physical library (number of routing layers in LEF).

Note

The number of layers reserved for power routing is always two. By default, optimization uses the effective number of routing layers shown below:

```
get_max_route_layer = count_route_layer - (# reserved for power routing)
```

Examples

The following example returns the total number of routing layers defined in the technology LEF:

```
% count_route_layer
10
```

Related Topics

Physical Commands

set_max_route_layer

get_max_route_layer

create_blockage

Creates a rectangular blockage area that limits the available area for placement, or prevents the placement of macros, RTL partitions, or cells.

Usage

create_blockage [-bottom *coordinate*] [-left *coordinate*] [-max_density *percentage*]
[-name *blockage_name*] [-right *coordinate*] [-top *coordinate*]
[-type {hard | macro | partial | soft}]

Arguments

- -bottom *coordinate*

  Specifies the bottom y-coordinate of the blockage. Use with the -left, -right, and -top options.

- -left *coordinate*

  Specifies the left x-coordinate of the blockage. Use with the -bottom, -right, and -top options.

- -max_density *percentage*

  Indicates the maximum percentage of the rectangle that is available for the placer. Must be used with the -type partial option.

- -name *blockage_name*

  Specifies the name of the rectangular specified blockage.

- -right *coordinate*

  Specifies the right x-coordinate of the blockage. Use with the -bottom, -left, and -top options.

- -top *coordinate*

  Specifies the top y-coordinate of the blockage. Use with the -bottom, -left, and -right options.

- -type {hard | macro | partial | soft}

  Specifies whether the placement blockage is hard, macro, partial, or soft. The default is hard. Details are provided in the description below.

  hard

  A blockage that prevents the placement of any RTL partition, cell, or macro by the placement engine in the Oasys-RTL tool. Previously placed cells with a FIXED or COVER status in the blockage area are not moved. Previously placed cells with a PLACED status in the blockage area are moved to a legal location. The write_def command's -floorplan option saves the hard blockage in the output floorplan file.

  macro

A blockage that prevents the placement of macros while allowing the placement of RTL partitions in the blockage area. Previously placed macros with a FIXED or COVER status are not moved. Previously placed macros with a PLACED status are moved to a legal location.

partial

A blockage that sets the available placement space as a percentage (-max_density) of area. The placer honors existing hard blockages in the density area.

soft

A blockage that prevents the placement of inverter or non-buffer logic by the implementation tool. Note that the Oasys-RTL tool treats hard and soft blockages in the same way with respect to placement. However, the “write_def -floorplan” command saves the soft blockage in the output floorplan file.

Description

Creates a rectangular placement blockage in microns. By default, a hard blockage is created.

Examples

The following example creates a blockage, “blockage0”, covering from (0, 0) to (500, 600):

```
% create_blockage -name blockage0 -left 0 -bottom 0 \
   -right 500 -top 600
```

Related Topics

Physical Commands

remove_blockage

create_chip

Specifies the layout of the chip.

Usage

create_chip {{-height *microns* -width *microns*} |
-points '{' '{'*x1 y1*'}' '{'*x2 y2*'}' ... '}' |
-utilization *percentage*}
[-aspect_ratio *ratio*]
[-bottom_clearance *microns*]
[-left_clearance *microns*]
[-origin '{'*left_coordinate bottom_coordinate*'}']
[-right_clearance *microns*]
[-standard_cell_utilization *percentage*]
[-top_clearance *microns*]

Arguments

- -height *microns* -width *microns*

  Required if neither the -points nor the -utilization argument is used. Specifies the height and width of the chip in microns.

- -points '{' '{'*x1 y1*'}' '{'*x2 y2*'}' ... '}'

  Required if neither the -height and -width argument pair nor the -utilization argument is used. Defines the coordinates of the rectilinear die area and corresponding standard cell rows. The coordinates must create a right-angle pattern. The coordinates can start at any point and go in either direction. The last coordinate point that closes the rectilinear pattern is optional. If you are defining a rectangle, then you have the option of only defining the lower left and upper right coordinates, in that order.

  Each x,y-coordinate pair must be enclosed in braces ({ }). The full list of coordinate pairs must be enclosed in braces as well.

- -utilization *percentage*

  Required if neither the -height and -width argument pair nor the -points argument is used. Specifies the chip utilization as a percentage of the die area. If you have problems with routing congestion, reducing this value should reduce the congestion.

  The chip utilization is based on the total cell area, which includes all the standard cells, macros, I/O pads, and physical-only cells.

  ```
  Chip Utilization = Total Cell Area / Die Area
  ```

- -aspect_ratio *ratio*

  Optional. Specifies the ratio of the height and width of the chip. The default is 1. The -height, -width, and -points options override -aspect_ratio.

- -bottom_clearance *microns*

  Optional. Specifies the clearance from the bottom of the die area to the bottom of the core area, reserving space for the placement of I/O pads. The default is 0.

- -left_clearance *microns*

  Optional. Specifies the clearance from the left side of the die area to the left side of the core area, reserving space for the placement of I/O pads. The default is 0.

- -origin '{'*left_coordinate bottom_coordinate*'}'

  Optional. Specifies the coordinates of the origin of the chip in microns. The coordinate pair must be enclosed in braces ({ }). The default is {0 0}.

- -right_clearance *microns*

  Optional. Specifies the clearance from the right side of the die area to the right side of the core area, reserving space for the placement of I/O pads. The default is 0.

- -standard_cell_utilization *percentage*

  Optional. Specifies the placement utilization as a percentage of the placeable area.

  The placement utilization is based on the total area of the standard cells only. The placeable area is the die area minus the fixed cell area. The fixed cell area includes the fixed macros, physical-only cells, I/O pads, and blockages.

  ```
  Standard Cell Utilization = Area of Standard Cells / Placeable Area

  Placeable Area = Die Area - Fixed Cell Area
  ```

- -top_clearance *microns*

  Optional. Specifies the clearance from the top of the die area to the top of the core area, reserving space for the placement of I/O pads. The default is 0.

Description

Specifies the chip origin, height, and width. You can also use this command to specify a rectilinear die, clearance, and utilization. Use this command after the synthesize command.

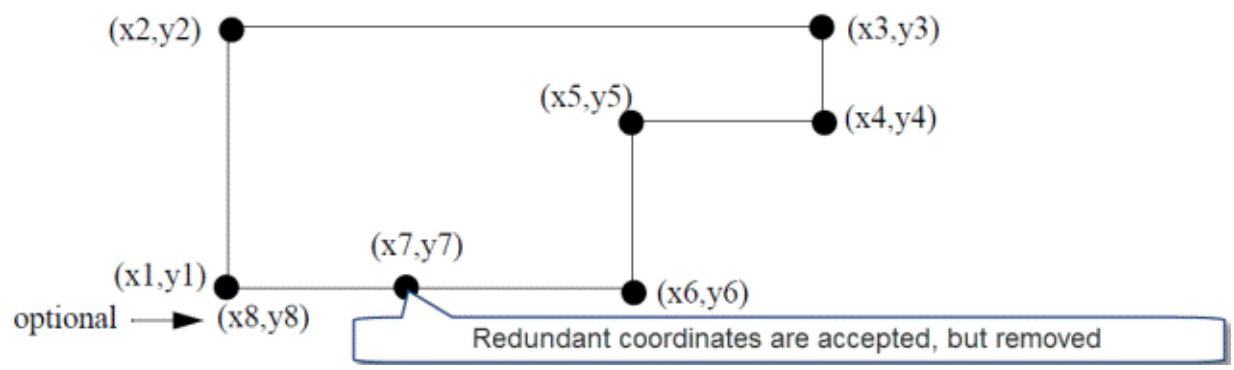
The required argument(s) must be one of the following:

- -height and -width (both)
- -points
- -utilization

If the -utilization and -standard_cell_utilization arguments are both specified then the total placement utilization percentage overrides the standard cell utilization percentage.

If both -points and -utilization arguments are used together, -utilization takes precedence.

Rectilinear dies are specified by a sequence of coordinates (points). You must specify an even number of points and create a rectangle at a minimum. Redundant points are removed. The last point is optional and is automatically added. All units are in microns.



Examples

Example 1: Create Chip with 100 Micron Clearance

This example specifies the origin, width, and height of a chip. It also specifies a 100 micron clearance between the chip core area and its I/O pads or pins.

```
[oasys-RTL]$ create_chip -origin { 0 0 } -width 2000 -height 2000 \
   -left_clearance 100 -bottom_clearance 100 -right_clearance 100 \
   -top_clearance 100
info: create new die (0.000000 0.000000) (2000.000000 2000.000000)[FP-110]
info: create new core (100.000000 100.000000) (1899.869950 1898.999950),
1285 core rows  [FP-109]
0
```

Example 2: Create Rectilinear Chip

This example specifies the rectilinear shape of the chip die area.

```
[oasys-RTL]$ create_chip -points {{0 0} {0 1000} {500 1000} {500 2000} \
   {2000 2000} {2000 0} }
info: create clearance left = 0.000000 bottom = 0.000000 right = 0.000000
top = 0.000000)  [FP-141]
info: new rectilinear die with bounding box (0.000000 0.000000)
(2000.000000 2000.000000) created, total 6 corner points  [FP-162]
1
```

Example 3: Create Chip with Origin at {100, 100}, 60% Utilization, and 0.6 Aspect Ratio

This example specifies the chip origin and aspect ratio. It also specifies the placement utilization.

```
[oasys-RTL]$ create_chip -origin {100 100} -utilization 60 -aspect_ratio 0.6
info: create new die (100.000000 100.000000) (1322.020000 833.210000)  [FP-110]
info: create new core (100.000000 100.000000) (1321.889950 832.199950), 523 core
rows  [FP-109]
0
```

Related Topics

Physical Commands

delete_physical

create_group

Creates a placement group. The tool places members of a group close together.

Usage

create_group ***string*** [-type {macro | instance}] [-instance *string*]
[-weight {high | low | medium}]

Arguments

- ***string***

  Specifies the name of a group.

- -type {macro | instance}

  Specifies the type of group. The type can be macro or instance (which does not include macros).

- -instance *string*

  Specifies the names of instances to include in the group.

- -weight {high | low | medium}

  Specifies the strength of the group. (default value: medium)

  Legal values: {low, medium, high}

Description

The create_group command creates a group with specified instances. The group becomes a placement constraint specifying that instances in the same group be placed close together. The create_group command can be used with macros only to control macro placement.

Examples

The following example creates a group called, A1, and assigns CTL instances to the group:

```
% create_group A1 -instances CTL
```

The following example creates a group called, A1, and assigns macros in the CTL module to the group. Leaf cells or RTL partitions in the CTL module are not assigned to the A1 group:

```
% create_group A1 -type macro -instances CTL
```

Related Topics

Physical Commands

report_groups

assign_to_group

remove_group

get_groups

create_halo

Creates a placement halo for macros.

Usage

create_halo {{-instance *inst_name*} | {-lib_cell *cell_name*}}
[-left *microns* -bottom *microns* -right *microns* -top *microns*]

Arguments

- -instance *inst_name*

  Specifies that a halo of the specified offset is applied to the specified instance. Either the -instance or -lib_cell option is required.

- -lib_cell *cell_name*

  Specifies that a halo of the specified offset is applied to all instances of <cell_name>. Either the -lib_cell or -instance option is required.

- -left *microns*

  Specifies the number of microns to extend from the left of the instance.

- -bottom *microns*

  Specifies the number of microns to extend below the instance.

- -right *microns*

  Specifies the number of microns to extend from the right of the instance.

- -top *microns*

  Specifies the number of microns to extend above the instance.

Description

Creates a DEF HALO for macro instances. If the -instance option is specified, a HALO is added only for the specified instance. If the -lib_cell option is specified, a HALO is added around all instances of that cell. The -left, -bottom, -right, and -top options are used to specify the HALO distance in microns. The default HALO distance is 10 microns. Either the -instance or -lib_cell option is required.

Examples

Example

All macros use a default HALO of 10 um on each side.

```
% create_halo
```

Example

This example creates a 20 micron HALO for instance i0.

```
% create_halo -instance i0 -left 20 -bottom 20 -right 20 -top 20
```

Example

This example creates the default 10 um HALO for all instances of library cell c0.

```
% create_halo -lib_cell c0
```

Related Topics

Physical Commands

delete_physical

remove_halo

create_macro_blockages

Creates blockages around specified macros.

Usage

create_macro_blockages [-bottom *distance*] [-halo *distance*] [-instances *instance_list*] [-layers *layer_name*] [-left *distance*] [-right *distance*] [-top *distance*] [-type {soft | routing | hard}]

Arguments

- -bottom *distance*

  Optional. Specifies a floating-point distance in microns that the macro blockage extends from the bottom edge of the macro. The default distance is 10 microns.

- -halo *distance*

  Optional. Specifies a floating-point distance in microns that the macro blockage extends from all edges of the macro. The default distance is 10 microns.

- -instances *instance_list*

  Optional. Specifies a list of instances around which to create macro blockages. The instances must be hard macros. If the -instances option is not specified, the command creates blockages around each macro, overwriting all existing blockages.

- -layers *layer_name*

  Optional. This option is mutually inclusive with the "-type routing" option. It specifies the layer for which the routing blockage needs to be created. You can specify more than one layer.

- -left *distance*

  Optional. Specifies a floating-point distance in microns that the macro blockage extends from the left edge of the macro. The default distance is 10 microns.

- -right *distance*

  Optional. Specifies a floating-point distance in microns that the macro blockage extends from the right edge of the macro. The default distance is 10 microns.

- -top *distance*

  Optional. Specifies a floating-point distance in microns that the macro blockage extends from the top edge of the macro. The default distance is 10 microns.

- -type {soft | routing | hard}

  Optional. Specifies how the place-and-route tool treats the new blockage. If you specify "-type soft", cells can be placed inside the blockage during detailed placement. If you specify "-type hard", no cells are allowed inside the blockage area. If you specify "-type routing" a routing blockage will be created of the layer specified in the "-layer" argument. The default is "-type hard".

Description

The blockage area provides channels for placing buffers and cells during detailed placement and additional space for wires during routing. If -instances is not specified, create_macro_blockages creates a blockage around each macro, overwriting any existing blockages.

The -left, -right, -top, -bottom options are mutually exclusive with the -halo option. If you use them together, the tool ignores the -halo option.

Examples

```
% create_macro_blockages -instances ram_1 -type hard -halo 1.1
```

Related Topics

Physical Commands

remove_macro_blockages

create_macro_groups

Creates macro groups based on connectivity, logical hierarchy, or thresholds for area or count.

Usage

create_macro_groups [-prefix *name*] [-ignore_connectivity {true | false}]
[-max_area_percentage *area_percentage*] [-max_macro_area *area*]
[-max_macro_count *count*] [-min_area_percentage *area_percentage*]

Arguments

- -prefix *name*

  Optional. Specifies the prefix for the names of all the macro groups that the tool creates.

- -ignore_connectivity {true | false}

  Optional. When the value is false, the tool creates groups based on the connectivity among macros as well as logical hierarchy. When the value is true, the grouping algorithm creates groups based on logical hierarchy and ignores connectivity. The default behavior is false.

- -max_area_percentage *area_percentage*

  Optional. Specifies that the tool groups together the macros in each logical hierarchy that has an area (including macros) equal to or less than the <area_percentage> of the total cell area. The default <area_percentage> is 20%.

- -max_macro_area *area*

  Optional. Specifies that the tool creates groups only when the maximum total area of the macros in the group is less than <area>. The default value is -1, which means that it is unlimited. This option is ignored when -ignore_connectivity is used.

- -max_macro_count *count*

  Optional. Specifies the maximum number of macros to include in each group. The default value is -1, which means that it is unlimited. This option is ignored when -ignore_connectivity is used.

- -min_area_percentage *area_percentage*

  Optional. Specifies that the tool groups together the macros in each logical hierarchy that has an area (including macros) equal to or greater than the <area_percentage> of the total cell area. The default <area_percentage> is 5%.

Description

This command uses connectivity among macros in addition to logical hierarchy to create macro groups. You can create groups based on threshold limits for macro count, macro area, and minimum and maximum area of logical hierarchies.

When you use the -ignore_connectivity option, the grouping algorithm performs clustering based only on logical hierarchy.

The created groups are like user-defined groups created with the create_group command. The report_groups command reports the groups and the groups are visible in the GUI. The create_macro_groups command does not require specific macro names like the create_group command.

To explore the macro grouping hierarchies in the GUI, change the minimum and maximum area percentages in the Physical view. In the Physical view, click the dropdown arrow next to the Normal view icon and choose **More View Options > View Setup**.

Examples

Example 1: Create Macro Groups With a Limited Number of Macros

This example creates groups with 15 or fewer macros. The names of the groups have an *LT15_* prefix.

```
create_macro_groups -prefix LT15 -max_macro_count 15
```

Example 2: Create Macro Groups In Logical Hierarchies Smaller Than 25% of Total Cell Area

This example creates macro groups based on logical hierarchies that have an area that is 25% or less of the total cell area of the design. The names of the groups have a *Max25* prefix.

```
create_macro_groups -prefix Max25 -max_area_percentage 25.0
```

Related Topics

Physical Commands

report_groups

create_group

create_macro_model

Creates a macro model cell to model a black box.

Usage

create_macro_model -width *width* -height *height* [-soft] ***name***

Arguments

- -width *width*

  Specifies the width of the macro model cell in microns.

- -height *height*

  Specifies the height of the macro model cell in microns.

- -soft

  Specifies that the macro model cell aspect ratio is adjustable.

- ***name***

  Specifies the name of the black box or empty model in the design.

Description

Creates a macro model cell of a specific size to model a black box or an empty module in the design. In addition, the name of the macro cell must match the name of the black box or empty module.

Examples

This example creates a macro model cell named macro1. The width and height of the cell are 1000 um and 500 um respectively.

```
% create_macro_model macro1 -width 1000 -height 500
```

Related Topics

Physical Commands

report_blackboxes

synthesize

create_region

Creates a region for the placement of specific instances.

Usage

create_region [-type {region | fence | guide} [-instance *inst_names*]
[-left *coordinate* -bottom *coordinate* -right *coordinate* -top *coordinate*] [-points '{' '{'*x1 y1*'}' '{'*x2 y2*'}' ...'}'] ***region_name***

Arguments

- -instance *inst_names*

  The name of one or more instances to be assigned to the region. Multiple instances are separated by spaces.

- -type {region | fence | guide}

  Specifies the DEF type for the region. The types are defined as follows:

  region - All instances assigned to the region are placed inside the region boundaries, and other cells can also be placed inside the region. This is the default if no -type is specified.

  fence - All instances assigned must be exclusively placed inside the region boundaries. No other instances are allowed inside the region.

  guide - All instances should be placed inside the region; however, it is a preference, not a hard constraint. Other constraints, such as wire length and timing can override this preference.

- -left *coordinate*

  Specifies the left x-coordinate of default region in microns.

- -bottom *coordinate*

  Specifies the bottom y-coordinate of default region in microns.

- -right *coordinate*

  Specifies the right x-coordinate of default region in microns.

- -top *coordinate*

  Specifies the top y-coordinate of default region in microns.

- -points '{' '{'*x1 y1*'}' '{'*x2 y2*'}' ...'}'

  Defines the coordinates of the rectilinear region area. The coordinates must only create a right angle patterns. The coordinates can start at any point and go in either direction. The last coordinate point that closes the rectilinear pattern is optional. If you are defining a rectangle, then you have the option of only defining the lower left and upper right coordinates, in that order. The units are in microns. Each x,y-coordinate pair must be enclosed in {} brackets. The full list of coordinate pairs must also be enclosed in brackets.

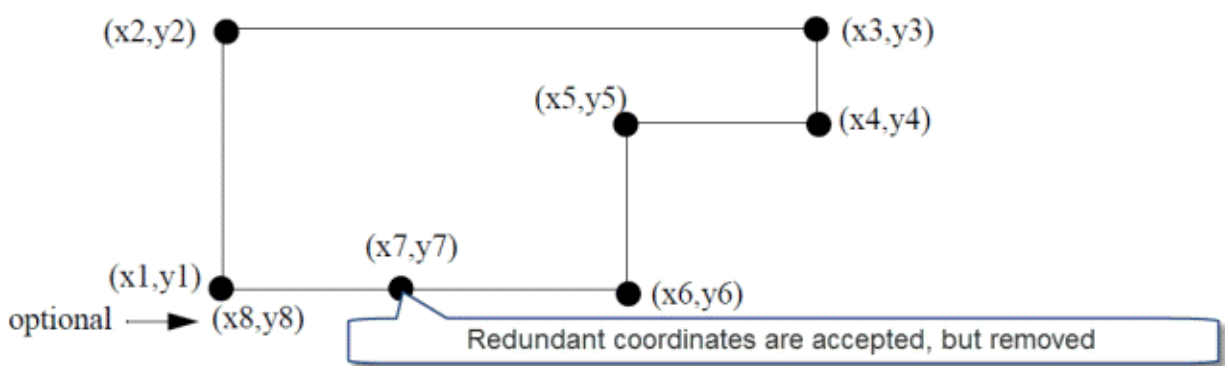
- ***region_name***

  Specifies the name of the rectilinear region.

Description

Creates a region for the placement. If the region of *region_name* does not exist, it creates a region with the given name and coordinates. If an instance is specified with the optional -instance argument, this command also assigns the instance to the region. You can assign instances to existing regions with the assign_to_region command.

The rectilinear region is specified by a sequence of coordinates (points) in microns. You must specify an even number of points, and minimally create a rectangle. Redundant points are removed. The last point is optional and is automatically added.



Examples

This example creates a region called region_1 of type fence from (500, 500) to (1000, 1000) in microns.

```
% create_region "region_1" -type fence -left 500 -bottom 500 \
   -right 1000 -top 1000
```

Example

This example creates a rectilinear region called region_2 of type guide:

```
% create_region "region_2" -type guide \
   -points { {100 100}{100 400} {300 400} {300 600} {600 600} {600 100}}
```

Related Topics

Physical Commands

remove_region

assign_to_region

remove_from_region

report_regions

delete_physical

Clears the physical information that you previously created or loaded in the session.

Usage

delete_physical [-all] [-rtlplacement] [-macro {placed | all}]
[-pin {placed | all}] [-blockage {routing | placement | all}] [-preroute]
[-region {region | fence | guide | all}] [-group] [-power_plan]

Arguments

- -all

  Specifies to delete all the physical information including placements, regions, blockages, die, and core area.

- -rtlplacement

  Specifies to delete the RTL placement.

- -macro {placed | all}

  Specifies to delete all macros or just the placed ones.

- -pin {placed | all}

  Specifies to delete all pins or just the placed pins.

- -blockage {routing | placement | all}

  Specifies to delete routing or placement blockages.

- -preroute

  Specifies to delete preroute blockage.

- -region {region | fence | guide | all}

  Specifies to delete the region, fence, or guide region.

- -group

  Specifies that all groups be deleted.

- -power_plan

  Specifies to delete the power rails.

Examples

Example

This example deletes all physical information.

```
% delete_physical -all
```

Example

This example deletes all RTL placement and regions.

```
% delete_physical -rtlplacement -region all
```

Example

This example deletes all placed macros, placed pins, and placement blockages.

```
% delete_physical -macro placed -pin placed -blockage placement
```

Related Topics

Physical Commands

floorplan

Creates the die and performs macro and pin placement. Use when fine-tuning macro and pin placement before placing RTL partitions.

Usage

floorplan [-place_macros {true | false}] [-place_pins {true | false}]

Arguments

- -place_macros {true | false}

  Places macros. The default is true.

- -place_pins {true | false}

  Places pins. The default is true.

Description

This command creates the die and places macros and pins but not RTL partitions. It adheres to placement constraints. If there are no placement constraints, macros and pins are placed for optimal timing and congestion. The floorplan command requires a design that has been mapped with the synthesize command.

Examples

The following example creates a die (if it does not already exist) and places macros and pins.

```
% floorplan
```

Related Topics

Physical Commands

optimize

synthesize

check_placement

get_core_boundary

Returns the coordinates of the core boundary of the current floorplan.

Usage

get_core_boundary

Arguments

None.

Description

Returns a string of coordinates for the boundary of the core region. You can create the core region using the create_chip command or read it in from a DEF file.

Examples

This example creates a core boundary that is a 600 x 600 rectangle with the origin at {0,0}.

```
% create_chip -origin {0.00 0.00} -points {{0 0} {600 600} }
% get_core_boundary
{ 0.000000 0.000000 }  { 600.000000 600.000000 }
```

Related Topics

Physical Commands

create_chip

get_groups

Generates a list of all groups, or searches for specific groups by name.

Usage

get_groups [*pattern*]

Arguments

- *pattern*

  Limits the search to the specified pattern. Supports glob-style patterns such as *, bar? and *foo*.

Description

Finds all groups that match the specified pattern. If no pattern is specified, returns a list of all groups.

Examples

In this example, there are two groups in the design: Grp1 and Grp2.

```
% get_groups
{Grp1 Grp2}
```

Related Topics

create_group

Physical Commands

report_groups

assign_to_group

remove_group

get_max_route_layer

Returns the maximum number of routing layers available for physical optimization.

Usage

get_max_route_layer

Arguments

None.

Description

Returns the maximum number of routing layers available for physical optimization. The maximum layer can be manually set with the set_max_route_layer command.

Note

By default, optimization uses the effective number of routing layers as follows:

get_max_route_layer = count_route_layer - (# reserved for power routing) where count_route_layer is the number of available routing layers in the technology LEF.

Examples

This example retrieves the maximum number of routing layers used in physical optimization:

```
% get_max_route_layer
6
```

Related Topics

Physical Commands

set_max_route_layer

count_route_layer

report_route_layers

get_regions

Generates a list of all regions, or searches for specific regions by name.

Usage

get_regions [*pattern*]

Arguments

- *pattern*

  Limits the search to the specific pattern. Supports glob-style patterns such as *, bar? and *foo*.

Description

Finds all regions that match the specified pattern. If no pattern is specified, returns a list of all regions.

Examples

Example 1: Get All Regions

This example returns a list of all regions in the design.

```
% get_regions
{region_1 region_2 region_3 ...}
```

Example 2: Get a Subset of Regions

This example searches for regions beginning with the letter ‘A’ and returns a list of all matching regions.

```
% get_regions A*
{A1 A2 A3 ...}
```

Related Topics

assign_to_region

create_region

remove_from_region

remove_region

get_route_layer_direction

Gets the routing direction of the specified layer.

Usage

get_route_layer_direction [-layer *layer_name*]

Arguments

- -layer *layer_name*

  Required. Specifies the layer for which to report the routing direction.

Description

The get_route_layer_direction command fetches the routing direction of the layer specified by the layer_name argument.

Examples

The following sequence of commands gets the current routing direction of the Metal1 layer (vertical), resets the direction using the set_route_layer_direction command, and then gets the new direction (horizontal).

```
[oasys-RTL]$get_route_layer_direction Metal1
vertical

[oasys-RTL]$set_route_layer_direction -layer Metal1 -direction horizontal

[oasys-RTL]$get_route_layer_direction Metal1
horizontal
```

place_instance

Places the specified instance(s).

Usage

place_instance [-name *string*] [-x *double*] [-y *double*]
[-orient {N | S | W | E | FN | FS | FW | FE}] [-fixed]

Arguments

- -name *string*

  Specifies the name of the instance to place.

- -x *double*

  Specifies the x-coordinate where the instance is to be placed.

- -y *double*

  Specifies the y-coordinate where the instance is to be placed.

- -orient {N | S | W | E | FN | FS | FW | FE}

  Specifies the orientation in which the instance should be placed. The default is N. This is optional.

- -fixed

  Marks the placed instance as fixed. If this is not set, instances are marked as placed.

Description

The place_instance command provides the ability to fine tune the placement of the instance based on precise coordinates.

Examples

```
[oasys-RTL]% place_instance -name i_cpu_mmu_ID_mem_0_0 -x 0.0 -y 840.0 \
   -fixed
```

Related Topics

Physical Commands

check_placement

place_pins

Places I/O pins and physical module pins.

Usage

place_pins

Arguments

None.

Description

This command places I/O pins and physical module pins.

Examples

The following example places pins:

```
% place_pins
```

Related Topics

Physical Commands

report_design_metrics

place_port

Provides the ability to preplace (fix) an I/O port of the design.

Usage

place_port **-name *port_name* -x *x_location* -y *y_location*** [-rect *llx lly urx ury*] [-layer *layer_name*]

Arguments

- **-name *port_name***

  Specifies the name of the port.

- **-x *x_location***

  Specifies the x location of the port.

- **-y *y_location***

  Specifies the y location of the port.

- -rect *llx lly urx ury*

  Controls the size of the port shape that is created. The port shape is defined by the lower left x, lower left y, upper right x, and upper right y-coordinates.

- -layer *layer_name*

  Specifies the layer on which the port is placed.

Description

Provides the ability to preplace (fix) an I/O port of the design. You can use the report_attribute command to see the port attribute details.

Examples

This command places port A at location (10, 20) on layer MET2.

```
% place_port -name A -x 10 -y 20 -layer MET2
```

Related Topics

Physical Commands

refine_macro

Modifies the floorplan of a macro instance by fixing, unfixing, or changing the orientation.

Usage

refine_macro [-fix | -unfix | -orient {N | S | W | E | FN | FS | FW | FE}]
[-instance *macro_name*]

Arguments

- -fix

  Fixes the instance(s) at the current position.

- -unfix

  Unfixes the instance(s).

- -orient {N | S | W | E | FN | FS | FW | FE}

  Changes the orientation of the instance(s). The default is N. The orientations are north, south, west, east, flip north, flip south, flip west, and flip east respectively.

- -instance *macro_name*

  Specifies the name of the macro instance.

Description

Refines the floorplan of a macro instance. All macros are refined if the -instance argument is not specified. Only one of the arguments (fix, unfix, or orient) can be specified at one time.

Examples

Example

This example fixes instance i0 at its current position.

```
% refine_macro -instance i0 -fix
```

Example

This example unfixes instance i0.

```
% refine_macro -instance i0 -unfix
```

Example

This example changes the orientation of i0 to FS.

```
% refine_macro -instance i0 -orient FS
```

Related Topics

Physical Commands

remove_blockage

Removes a placement blockage constraint that you previously created or loaded in the session.

Usage

remove_blockage ***blockage_name***

Arguments

- ***blockage_name***

  Specifies the name of the blockage to be removed.

Examples

This example removes the blockage0 blockage.

```
% remove_blockage blockage0
info:    remove placement blockage 'blockage0'  [FP-104]
```

Related Topics

Physical Commands

create_blockage

remove_from_group

Removes instances from a placement group.

Usage

remove_from_group ***string*** [-instance *string*] [-all]

Arguments

- ***string***

  Specifies the name of a group.

- -instance *string*

  Specifies the names of the instances to be removed.

- -all

  Specifies that all the instances in the group be removed.

Examples

The following example removes all the instances from group A1.

```
% remove_from_group A1 -all
```

Related Topics

Physical Commands

create_group

report_groups

remove_group

get_groups

remove_from_region

Removes an instance from a placement region.

Usage

remove_from_region [-instance *inst_name* | -group *group_name*] ***region_name***

Arguments

- -instance *inst_name*

  Specifies the name of the instance that is to be removed from the specified region.

- -group *group_name*

  Specifies a group name to be removed from the specified region.

- ***region_name***

  Specifies the name of the region.

Examples

This example removes the i0 instance from the r0 region.

```
% remove_from_region r0 -instance i0
```

Related Topics

Physical Commands

assign_to_region

create_region

remove_group

Deletes the specified placement group.

Usage

remove_group ***string***

Arguments

- ***string***

  Specifies the name of the group.

Examples

The following example creates a group A1, and then removes it.

```
% create_group A1
% remove_group A1
```

Related Topics

Physical Commands

create_group

report_groups

remove_from_group

get_groups

remove_halo

Deletes halos from a design.

Usage

remove_halo [-instance *inst_name*] [-lib_cell *cell_name*]

Arguments

- -instance *inst_name*

  Specifies the macro instance name from which the halo is deleted.

- -lib_cell *cell_name*

  Specifies the macro cell name from which halos are deleted.

Description

Depending on the specified option, remove_halo deletes either the halo of a specified macro instance, the halos of all macro instances with the specified lib_cell name, or all halos in the design. When no option is specified, all halos in the design are deleted.

Note

If -instance and -lib_cell are used simultaneously, -lib_cell is ignored.

Examples

Example 1

This example deletes the halo of the ifu/ict macro instance:

```
% remove_halo -instance ifu/ict
```

Example 2

This example deletes all halos for the macro instances of library cell, “bw_r_idct”:

```
% remove_halo -lib_cell bw_r_idct
```

Related Topics

Physical Commands

create_halo

remove_macro_blockages

Removes macro blockages created with the create_macro_blockages command.

Usage

remove_macro_blockages [-instances *instances*] [-layers *layer_names*]
[-type {routing | place | all}]

Arguments

- -instances *instances*

  Specifies a list of macros from which to remove previously defined blockages.

- -layers *layer_names*

  This option is mutually inclusive with the "-type routing" option. It specifies the layer(s) on which the routing blockages will be removed. You can specify more than one layer.

- -type {routing | place | all}

  Specifies how the place-and-route tool handles the blockage to be removed. If you specify "-type routing", routing blockages are removed. If you specify "-type place", placement blockages are removed. If you specify "-type all", all blockages are removed. The default is "-type all".

Examples

```
% remove_macro_blockages -instances ram_1 -type routing -layers Metal1
```

Related Topics

Physical Commands

create_macro_blockages

remove_macros_from_edges

Removes edge constraints from macro instances or macro groups.

Usage

remove_macros_from_edges [-instance *instance_name* ...] [-group *groupname* ...]

Arguments

- -instance *instance_name* ...

  Specifies the names of the instances from which the tool removes the edge constraints.

- -group *groupname* ...

  Specifies the names of the macro groups from which the tool removes the edge constraints.

Examples

This example creates the *G1* group and assigns it to edge number 3. Then it removes the edge constraint from the *G1* group.

```
% create_group G1 -type macro -instance \
   [get_cells -hier * -filter  "is_hard_macro==true && \
   full_name =~pipe1/*"]

% assign_macros_to_edges -group G1 -edges 3

% remove_macros_from_edges -group G1
```

Related Topics

Physical Commands

create_group

assign_macros_to_edges

remove_region

Deletes a placement region.

Usage

remove_region ***region_name***

Arguments

- ***region_name***

  Specifies the name of the region.

Description

The given region is removed and all the instances assigned to that region are no longer constrained by the region.

Examples

```
% remove_region r0
```

Related Topics

Physical Commands

create_region

remove_from_region

set_max_route_layer

Sets the maximum number of routing layers available for physical optimization.

Usage

set_max_route_layer ***num***

Arguments

- ***num***

  Specifies the maximum number of routing layers to use in physical optimization.

Description

The given number of routing layers for physical optimization must be less than or equal to the total number of routing layers available in the physical (LEF) library. By default, the two top routing layers are reserved for power routing. The remaining layers, (which include the maximum number of layers in the physical library minus two for power routing) are available for optimization.

By default:

get_max_route_layer = count_route_layer - 2

By using set_max_route_layer <num>:

get_max_route_layer = <num>

Examples

Example

This example sets the maximum routing layers to six.

```
% set_max_route_layer 6
```

Example

This example counts the maximum available routing layers in the physical library using count_route_layer. Next, get_max_route_layer returns the default maximum routing layers available for optimization, which is the total number of routing layers minus two layers that are reserved for power routing. Finally, the maximum number of layers available for physical optimization is changed to 7 with set_max_route_layer.

```
% count_route_layer
10
% get_max_route_layer
8
% set_max_route_layer 7
% get_max_route_layer
7
```

Related Topics

Physical Commands

get_max_route_layer

count_route_layer

report_route_layers

set_route_layer_direction

Sets the routing direction of one or more layers.

Usage

set_route_layer_direction [-layers *layer_names*] [-direction *direction*]

Arguments

- -layers *layer_names*

  Specifies the layer(s) for which to change the route direction.

- -direction *direction*

  Specifies the new routing direction for the specified layer(s).

Description

The set_route_layer_direction command changes the preferred routing direction of the layer used by the tool router. This command is used to change the layer direction from the direction specified in the technology LEF file.

Examples

This command sets the routing direction of the Metal1 layer to vertical:

```
set_route_layer_direction -layers Metal1 -direction vertical
```

This command sets the routing direction for layers Metal1, Metal3, and Metal5 to vertical:

```
set_route_layer_direction -layers { Metal1 Metal3 Metal5} \
   -direction vertical
```

set_route_layer_max_usage

Sets the fraction of the specified routing layer that can be used for routing.

Usage

set_route_layer_max_usage **<*def_routing_layer*>** [*fraction*]

Arguments

- <def_routing_layer>

  Specifies the name of the DEF routing layer.

- *fraction*

  Specifies the fraction of the <def_routing_layer> that can be used for routing. This is a number between 0 and 1. A number of 0.6 would signify that 60% of the layer can be used for routing. If there are pre-routes or blockages, then the usage is the fraction of the remaining routing layer. The default is 1, which means that all of the area is available for routing (excluding pre-routes and blockages).

Description

The Oasys-RTL tool considers the pre-routed information and blockages as non-routable when calculating the area that can be used by routing.

Examples

This example specifies that the maximum route layer usage of layer metal2 is 40%. This means that the 40% is the maximum available for each routing grid bin for the metal2 layer. It is less if pre-routes or blockages exist:

```
% set_route_layer_max_usage metal2 0.4
```

Related Topics

Physical Commands

report_route_layers

set_route_layer_spacing

Overwrites the default spacing provided in the technology LEF file for a given layer.

Usage

set_route_layer_spacing ***layer_name spacing***

Arguments

- ***layer_name***

  Specifies the name of the layer for which the spacing is defined.

- ***spacing***

  Specifies the spacing between the layer. This is a real number that uses the units defined in technology LEF file.

Examples

This command sets the layer spacing on layer M1 to be 1.5 LEF units.

```
% set_route_layer_spacing M1 1.5
```

Related Topics

Physical Commands

report_route_layers

update_congestion

Recalculates the congestion estimates.

Usage

update_congestion [-detail {-1 | 0 | 1}]

Arguments

- -detail {-1 | 0 | 1

  Specifies that detailed placement information is used for routing congestion estimation. The refine command must have been run to create the detailed placement before you use this option. The arguments are defined as follows:

  - -1 - The Oasys-RTL tool automatically determines whether to use detail placement information or not. The tool attempts to use detail placement information if all RTL partitions have been correctly detail placed. This is equivalent to not specifying the -detail option. This is the default.
  - 0 - Forces non-detail (global) placement for congestion estimation. The rtl partitioning is used to update the congestion calculations.
  - 1 - Forces detail placement. When 1 is specified, the tool uses global placement only on RTL partitions that do not have detail placement information.

Description

Updates the congestion calculations. Normally this command does not need to be run, as the tool automatically updates congestion after optimization. This command is only needed if the placement or netlist was changed manually.

Examples

This example updates the congestion calculations. It uses detailed placement if available in all of the RTL partitions:

```
% update_congestion
```

Related Topics

Physical Commands

Chapter 5
Report Commands

The report commands display information about the design.

Table 5-1. Report Commands

Command	Description
config_report	Configures the columns used by the report_timing, the “report_explore -floorplan -detail”, or “report_explore -detail” commands for the remainder of the session.
report_area	Reports the area information for all module instances.
report_attribute	Reports the attributes of a specified object. The report includes the value for each attribute.
report_blackboxes	Reports the number and names of blackbox instantiations in the design.
report_case_analysis	Reports ports and pins set to constant logic values.
report_cells	Reports the number and area of each library cell in the design.
report_clock_gating	Reports the information on clock gating of flip-flops done by the tool.
report_clock_gating_cell	Reports the integrated clock-gating cells used for clock-gating flip-flops.
report_clock_gating_options	Reports the options set for clock-gating flip-flops.
report_clocks	Reports the clock constraints that were defined with SDC commands.
report_congestion	Reports the routing congestion statistics for the design.
report_delay	Reports the delay associated with a particular cell instance or net.
report_design_metrics	Reports the design characteristics of the top-level design in the session. Design size, power, instance count, physical data including die size, and congestion are reported.
report_dft_partitions	Reports the instances for all DFT partitions.
report_dft_registers	Reports the DFT related information about the registers of the design in session.

Table 5-1. Report Commands (cont.)

Command	Description
report_dft_violations	Reports the type, the pin, and the number of affected registers for each DFT violation.
report_edges	Reports the assignment of macro groups and instances to various edges of the die.
report_electrical_violations	Reports the electrical violations for the specified power domain(s).
report_endpoints	Reports the worst endpoints and their slacks, sorted by criticality.
report_explore	Creates a table comparing output values of multiple design space exploration runs.
report_groups	Reports the groups used for placement.
report_hierarchy	Reports the area and cell count information about each hierarchy in the design. By default, reports the statistics of the top-level design in the session.
report_instances	Reports the immediate children belonging to a specified hierarchical instance.
report_layer_rc	Reports the unit resistance and capacitance values of each layer. These values are loaded from the captable file or PTF file.
report_leakage	Reports the leakage power of the design in session and the percentage of cells in each threshold voltage group.
report_leakage_setup	Reports the threshold voltage group characteristics.
report_lib_cell_delay	Reports the delay for the specified cell.
report_library_cells	Reports what library cells are in the loaded library.
report_logic_depth	Reports a text-based histogram with bins based on the number of logic levels in a timing path.
report_multibit	Reports a summary of the results of multi-bit mapping. The report includes statistics for the flip-flops that the tool packed successfully as well as the flip-flops that the tool deemed ineligible for multi-bit mapping.
report_mv	Reports the details of instances of multi-voltage cells.
report_name_rules	Reports the properties defined for a set of name rules. Name rules are created or modified by the define_name_rules command.
report_net	Reports net information in the current design.

Table 5-1. Report Commands (cont.)

Command	Description
report_operating_conditions	Reports the information for all operating conditions that exist in the database.
report_parameters	Reports the values and types of the Oasys-RTL parameters.
report_path_groups	Reports quantitative information about each existing path group.
report_power	Reports the power consumption information for the design.
report_power_domains	Reports the area and power information for specified power domains.
report_pst	Reports the power states (modes) in the current design.
report_regions	Reports physical information about specified regions.
report_retime	Generates a report on the number of registers in retiming modules.
report_route_layers	Reports how many resources are available for each routing layer.
report_rtl_partitions	Reports all RTL partitions that the tool has created.
report_scan_chains	Reports the information about the scan chains of the design in session.
report_switching_activity	Reports the switching activity values of the pins in the design.
report_test_clocks	Reports test clock-related information for test clocks in the design.
report_test_pins	Reports information about the pin (definition point) and its scan mode value for all the test pins defined in the design.
report_timing	Reports the worst paths (based on slack) in the design that satisfies the given constraints.
report_timing_exceptions	Reports the timing exceptions accepted by the tool. Timing exceptions that can be reported include false paths, multicycle paths, min/max delays, path groups, and reset_path.
report_timing_mode	Reports all timing constraint modes and operating conditions.
report_units	Outputs the default units for values when reading and writing SDC files
report_upf_status	Reports the status of the load_upf and commit_upf commands.

config_report

Configures the columns used by the report_timing, the “report_explore -floorplan -detail”, or “report_explore -detail” commands for the remainder of the session.

Usage

config_report {timing | explore_floorplan | explore} [-format <list>] [-reset]

Arguments

- {timing | explore_floorplan | explore}

  Configures the timing report, the floorplan exploration report, or the design space exploration report.

- -format <list>

  Specifies the columns for the table that report_timing generates. The list can include any of the following items:

  cell, slew, net_delay, arc_delay, delay, arrival, edge, net_load, load, fanout, location, power_domain

  Specifies the columns for the table that “report_explore -detail” generates. The list can include any of the following default and optional items:

  default: explore_vars, job_ids, wall_clock, cpu_time, frequency, clock_period, worst_slack, worst_slack_path_group, worst_slack_all, total_negative_slack, total_power, instance_count, instance_area, instance_leakage, utilization, wire_length, chip_size, congestion_global, view_phy_criticality, view_phy_hierarchical, view_phy_congestion, view_phy_leakage_power, view_phy_dynamic_power, status, hostname, pid, log_file

  optional: internal_power, switching_power, leakage_power, cell_area, cell_count, cell_leakage, chip_height, chip_width, congestion_local

  Note

  For the “report_explore -detail” command, if a specified -format <list> does not contain either explore_vars or job_ids, the job_ids column is automatically included in the report.

- -reset

  Restores default report settings. This option applies only to the floorplan exploration report and the design space exploration report.

Examples

Example

The following example configures the timing report to include columns named cell, net_delay, arc_delay, and arrival.

```
% config_report timing -format {cell net_delay arc_delay arrival}
% report_timing
-------------------------------------------------------------------------
Startpoint: exu/alu/addsub/sub_dff/q_reg[63]/Q
    (Clocked by GCLK R)
Endpoint: exu/ecl/dff_mem_invalid_e2m_q_reg[0]/D
    (Clocked by GCLK R)
Path Group: default
Data required time: 1181.8
    (Clock period: 1500.0, minus Uncertainty: 100.0, plus Latency 0.0,
minus Setup time: 218.2)
Data arrival time: 1244.6
Slack: -62.8
Logic depth: 41
-------------------------------------------------------------------------
                                                    Net      Arc  Arrival
Path                      Module/Cell             Delay    Delay     Time
                                                   (ps)     (ps)     (ps)
-------------------------------------------------------------------------
gclk                      {create_clock}                     0.0      0.0
spc_hdr/I0/i_0_0/A2->ZN   AND2_X4_HVT               0.0      0.0      0.0
exu/alu/addsub/sub_dff/q_reg[63]/CK->Q
                          SDFF_X1_HVT#*             0.0    282.5    282.5
exu/alu/addsub/i_0_1_74/A2->ZN
                          NAND2_X4_HVT              0.0     15.6    298.1
exu/alu/addsub/i_0_1_72/A2->ZN
                          NAND2_X4_HVT              0.0     40.4    338.5
exu/alu/addsub/adder/i_0_0_442/A->ZN
                          INV_X8_HVT                0.0      8.6    347.1
exu/alu/addsub/adder/i_0_0_440/A1->ZN
                          NAND2_X4_HVT              0.0     25.1    372.2
exu/alu/addsub/adder/i_0_0_437/A2->ZN
                          NAND2_X4_HVT              0.0     13.5    385.7
```

Example

The following example configures the report for the “report_explore -detail” command to include columns named switching_power, leakage_power, total_negative_slack, total_power, and job_ids.

```
% config_report explore -format {switching_power leakage_power \
total_negative_slack total_power job_ids}
% report_explore -detail
Report Exploration metrics (details):
-----+-------------+----------+----------+-------------+---------------
     |Switching    |Leakage   |TNS(ns)   |Total        |Job_ids
     |_power(uw)   |_power(uw)|          |_power(uw)   |
-----+-------------+----------+----------+-------------+---------------
1    | 10143.236328|1865.978882| 0.000000| 39349.835938|explore.1.1.1.1
2    | 10148.872070|1865.973511| 0.000000| 39393.574219|explore.0.1.1.1
3    |102153.007812| 992.744019| 0.000000|137978.484375|explore.2.0.1.0
4    | 10293.923828|  12.960711|44.698521| 40937.628906|explore.0.2.0.1
5    | 10295.111328|  12.960733|44.698521| 40928.664062|explore.1.2.0.1
```

Related Topics

Report Commands

report_timing

report_explore

report_area

Reports the area information for all module instances.

Usage

report_area [-no_macros]

Arguments

- -no_macros

  Specifies that the area report should not include macros.

Description

Reports the area information of all of the module instances. In the column headers, the Cells column reports the number of standard cells in each module instance. The Area column reports the total area of cells in each module instance in square microns.

Examples

```
% report_area
Report Instance Areas:
-----+---------------------------------+-------------------------------+------+---------
     |Instance                         |Module                         |Cells |Cell Area
-----+---------------------------------+-------------------------------+------+---------
1    |top                              |                               |152307|   540252
2    |  ifu                            |sparc_ifu                      | 29280|   107003
3    |    fdp                          |sparc_ifu_fdp                  |  9827|    16858
4    |      t2pcs_reg                  |dff_s__parameterized7__1       |    99|      352
5    |      npcs_mux                   |dp_mux4ds__parameterized1__1   |   155|      177
6    |      pcs_mux                    |dp_mux4ds__parameterized1__2   |   155|      177
```

Related Topics

Report Commands

report_attribute

Reports the attributes of a specified object. The report includes the value for each attribute.

Usage

report_attribute [-class <class>] [-quiet] <object_name>

Arguments

- -class <class>

  Specifies the type of the specified <object_name>. This option is required if the name is specified instead of an object. The following classes are supported: cell, clock, design, inst, lib, lib_cell, lib_pin, module, net, pin, port, and power_domain.

- -quiet

  Specifies that no messages should be issued.

- <object_name>

  Specifies the name or attribute object.

Examples

Example

This example reports attribute information on the DRAM23_N_REF_RES pin.

```
% report_attribute -class pin DRAM23_N_REF_RES
Report Port/Pin Attributes:
---+-----------------+----------------+-------
   |Attribute Name   |Value           |Writable
---+-----------------+----------------+--------
1  |direction        |in              |false
2  |pin_direction    |in              |false
3  |port_direction   |in              |false
4  |is_clock_pin     |false           |false
5  |is_clock_port    |false           |false
6  |is_hierarchical  |true            |false
7  |name             |DRAM23_N_REF_RES|false
8  |full_name        |DRAM23_N_REF_RES|false
9  |fanout           |1               |false
10 |location         |                |yes
11 |layer            |                |yes
12 |route_layer      |                |yes
13 |is_physical      |false           |false
14 |is_placed        |placed          |false
-----+---------------+----------------+--------
```

Example

The following example outputs a list of the valid port attributes.

```
% report_attribute [get_ports DRAM0_RAS_L]

Report Port/Pin Attributes:
---+---------------+-----------+--------
   |Attribute Name |Value      |Writable
---+---------------+-----------+--------
1  |direction      |out        |false
2  |pin_direction  |out        |false
3  |port_direction |out        |false
4  |is_clock_pin   |false      |false
5  |is_clock_port  |false      |false
6  |is_hierarchical|true       |false
7  |name           |DRAM0_RAS_L|false
8  |full_name      |DRAM0_RAS_L|false
9  |fanout         | 1         |false
10 |location       |           |yes
11 |layer          |           |yes
12 |route_layer    |           |yes
13 |is_physical    |false      |false
14 |is_placed      |placed     |false
---+---------------+-----------+--------
```

Example

The following example outputs a list of the valid net attributes.

```
% report_attribute [get_nets DRAM0_RAS_L]

Report Net Attributes:
--+--------------+-----------+--------
  |Attribute Name|Value      |Writable
--+--------------+-----------+--------
1 |name          |DRAM0_RAS_L|false
2 |full_name     |DRAM0_RAS_L|false
3 |dont_touch    |true       |false
4 |num_pins      | 2         |false
5 |route_length  | 865.200012|false
6 |net_type      |signal     |false
7 |driver_count  | 1         |false
--+--------------+-----------+--------
```

Example

The following example outputs a list of the valid cell attributes.

```
% report_attribute [get_cells spc_hdr]
Report Instance Attributes:
-----+-----------------+-------------------+--------
     |Attribute Name   |Value              |Writable
-----+-----------------+-------------------+--------
1    |name             |spc_hdr            |no
2    |full_name        |spc_hdr            |no
3    |number_of_pins   |                12 |no
4    |ref_name         |bw_clk_cl_sparc_cmp|no
5    |is_black_box     |false              |no
6    |is_combinational |false              |no
7    |is_sequential    |false              |no
8    |is_physical_only |false              |no
9    |is_mapped        |false              |no
10   |is_placed        |global             |no
11   |is_fixed         |false              |no
12   |is_hard_macro    |false              |no
13   |is_hierarchical  |true               |no
14   |area             |                 0 |no
15   |width            |                 0 |no
16   |height           |                 0 |no
17   |dont_touch       |false              |no
18   |is_dont_touch    |false              |no
19   |power_domain     |/PD_Default        |no
20   |is_buffer        |false              |no
21   |is_inverter      |false              |no
22   |is_latch         |false              |no
23   |is_ff            |false              |no
24   |is_clock_gate    |false              |no
25   |preserve_boundary|false              |no
26   |is_scan          |false              |no
27   |is_dont_scan     |false              |no
28   |is_size_only     |false              |no
29   |object_class     |cell               |no
30   |is_multibit      |false              |no
-----+-----------------+-------------------+--------
```

Example

The following example outputs a list of the valid library cell attributes.

```
% report_attribute [get_lib_cell OR2_X1]
Report LibCell Attributes:
-----+--------------------+-----------------------------+--------
     |Attribute Name      |Value                        |Writable
-----+--------------------+-----------------------------+--------
1    |dont_use            |false                        |yes
2    |dont_touch          |false                        |yes
3    |name                |OR2_X1                       |no
4    |full_name           |NangateOpenCellLibrary/OR2_X1|no
5    |function            |ZN=(A1 | A2)                |no
6    |number_of_pins      |                            3|no
7    |ref_name            |OR2_X1                       |no
8    |ref_lib_name        |NangateOpenCellLibrary       |no
9    |is_black_box        |false                        |no
10   |is_combinational    |true                         |no
11   |is_sequential       |false                        |no
12   |is_physical_only    |false                        |no
13   |is_hard_macro       |false                        |no
14   |is_hierarchical     |false                        |no
15   |area                |                     1.064000|no
16   |is_isolation_cell   |false                        |no
17   |is_level_shifter    |false                        |no
18   |is_clock_gating_cell|false                        |no
19   |is_logical_only     |false                        |no
20   |is_buffer           |false                        |no
21   |is_inverter         |false                        |no
22   |is_latch            |false                        |no
23   |is_ff               |false                        |no
24   |user_bloat          |                            0|yes
25   |is_scan             |false                        |no
26   |width               |                     0.760000|no
27   |height              |                     1.400000|no
28   |symmetry_x          |true                         |yes
29   |symmetry_y          |true                         |yes
30   |symmetry_r90        |false                        |yes
31   |object_class        |lib_cell                     |no
32   |is_multibit         |false                        |no
33   |mbit_width          |                            1|no
34   |is_map_to_multibit  |true                         |no
35   |is_memory           |false                        |no
36   |default_orientation |N                            |yes
37   |is_pad_cell         |false                        |no
-----+--------------------+-----------------------------+-------
```

Example

The following example outputs a list of the valid library pin attributes.

```
% report_attribute [get_lib_pin OR2_X1/ZN]
Report LibPin Attributes:
-----+--------------+-----------------------------------------+--------
     |Attribute Name|Value                                    |Writable
-----+--------------+-----------------------------------------+--------
1    |direction     |out                                      |no
2    |pin_direction |out                                      |no
3    |name          |ZN                                       |no
4    |full_name     |NangateOpenCellLibrary/OR2_X1/ZN         |no
5    |function      |(A1 | A2)                                |no
6    |is_clock_pin  |false                                    |no
7    |max_fanout    |340282346638528859811704183484516925440.00|no
8    |is_scan_enable|false                                    |no
9    |is_scan_in    |false                                    |no
10   |is_scan_out   |false                                    |no
11   |object_class  |lib_pin                                  |no
-----+--------------+-----------------------------------------+--------
```

Related Topics

Report Commands

get_attribute

set_attribute

set_user_attribute

define_user_attribute

report_blackboxes

Reports the number and names of blackbox instantiations in the design.

Usage

report_blackboxes

Arguments

None.

Examples

```
[oasys-RTL]$ report_blackboxes
Report BlackBoxes:
-----+-------------+---------
     |BlackBox name|Instances
-----+-------------+---------
1    |decoder      |        2
2    |encoder      |        1
-----+-------------+---------
```

Related Topics

Report Commands

create_macro_model

report_case_analysis

Reports ports and pins set to constant logic values.

Usage

report_case_analysis [-modes *mode_list*] [-pins *pin_list*] [-all *bool*]

Arguments

- -modes *mode_list*

  Specifies the timing modes for which to report. If you do not specify this argument, reports case analysis for all modes.

- -pins *pin_list*

  Specifies a list of pins for which to report the constant values. If you do not specify this argument, reports all pins with constant values.

- -all *bool*

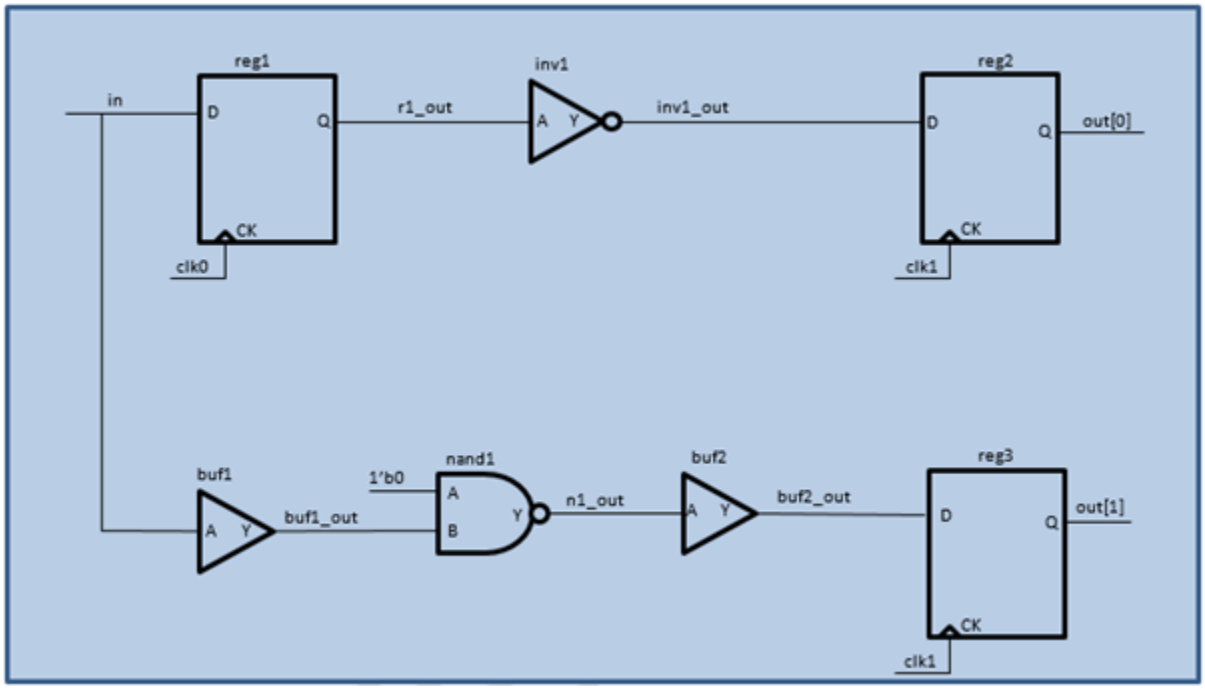
  Includes pins marked constant due to either ‘0’ or ‘1’ signal value propagated from one of its connected pins. (Default: false)

Examples

Example 1

Consider the following design netlist and SDC file.

Figure 5-1. Design Netlist 1



```
## SDC File 1 ##
create_clock -period 2.0 -waveform {0.0 1.0} -name clk0 [get_ports clk0]
create_clock -period 3.0 -waveform {0.0 1.0} -name clk1 [get_ports clk1]

set_input_delay 0.01 -clock clk0 [get_ports in]
set_output_delay 0.02 -clock clk1 [get_ports out*]

set_case_analysis 1 [get_ports in]
```

Executing the report_case_analysis command with no arguments generates a report of only the pins explicitly set to a constant value.

```
[oasys-RTL]$ report_case_analysis
```

Table 5-2. Command Results with No Arguments

Pin	Value	Constant Type
in	1	Constraint
nand1/A	0	Netlist

Executing the command with the -all argument includes pins set through logic propagation.

```
[oasys-RTL]$ report_case_analysis -all
```

Table 5-3. Command Results with -all Argument

Pin	Value	Constant Type
in	1	Constraint
buf1/A	1	Propagation
reg1/D	1	Propagation
nand1/A	0	Netlist
nand1/Y	1	Propagation
buf2/A	1	Propagation
buf1/Y	1	Propagation
nand1/B	1	Propagation
buf2/Y	1	Propagation
reg3/D	1	Propagation

Example 2

Consider the design netlist shown in Figure 5-1 along with the following mulit-mode SDC file:

```
create_clock -period 2.0 -waveform {0.0 1.0} -name clk0 [get_ports clk0]
create_clock -period 3.0 -waveform {0.0 1.0} -name clk1 [get_ports clk1]
set_input_delay 0.01 -clock clk0 [get_ports in]
set_output_delay 0.02 -clock clk1 [get_ports out*]

set_timing_mode m1
set_case_analysis 1 [get_ports in]

set_timing_mode m2
set_case_analysis 0 [get_ports in]
```

Executing the report_case_analysis command with no arguments generates a report of only the pins explicitly set to a constant value.

```
[oasys-RTL]$ report_case_analysis
```

Table 5-4. Command Results with No Arguments (Multi-mode Design)

Pin	Value	Constant Type	Mode
in	1	Constraint	m1
nand1/A	0	Netlist	m1
in	0	Constraint	m2
nand1/A	0	Netlist	m2

Executing the command with the -all argument includes pins set through logic propagation.

```
[oasys-RTL]$ report_case_analysis -all
```

Table 5-5. Command Results with -all Argument (Multi-mode Design)

Pin	Value	Constant Type	Mode
in	1	Constraint	m1
nand1/A	0	Netlist	m1
in	0	Constraint	m2
nand1/A	0	Netlist	m2
buf1/Y	1	Propagation	m1
buf1/A	1	Propagation	m1
nand1/Y	1	Propagation	m1
nand1/B	1	Propagation	m1
buf2/Y	1	Propagation	m1

Table 5-5. Command Results with -all Argument (Multi-mode Design) (cont.)

Pin	Value	Constant Type	Mode
buf2/A	1	Propagation	m1
reg3/D	1	Propagation	m1
reg1/D	1	Propagation	m1
buf1/Y	0	Propagation	m2
buf1/A	0	Propagation	m2
nand1/Y	1	Propagation	m2
nand1/B	0	Propagation	m2
buf2/Y	1	Propagation	m2
buf2/A	1	Propagation	m2
reg3/D	1	Propagation	m2
reg1/D	0	Propagation	m2

Related Topics

Report Commands

remove_case_analysis

report_cells

Reports the number and area of each library cell in the design.

Usage

report_cells

Arguments

None.

Description

Reports the family, type, data, count, area, power, and library of each library cell used in the design. The Data column in the report indicates whether a cell used in the design has only timing and functional information (*.lib*), physical information (LEF), or both.

Examples

```
% report_cells

Report Cells:
--+---------+---------+-----+----+------+
  |Cell     |Family   |Type |Data|Count |
--+---------+---------+-----+----+------+
1 |ACHCINX2 |ACHCINX2 |comb |Both| 26   |
2 |ACHCONX2 |ACHCONX2 |comb |Both| 16   |
3 |AND2X2   |AND2X2   |comb |Both| 33416|
4 |AND3X2   |AND3X2   |comb |Both| 4022 |
5 |AND4X2   |AND4X2   |comb |Both| 1266 |
6 |AO21X2   |AO21X2   |comb |Both| 1268 |
7 |AO22X2   |AO22X2   |comb |Both| 21503|
8 |AO2B2BX2 |AO2B2BX2 |comb |Both| 785  |

----------+-----------+-----------+-----------------+----------
Area (squm)|Total (squm)|Leakage (nw)|Total Leakage(nw)|Library
----------+-----------+-----------+-----------------+----------
14.8       | 385.3     | 50.668    | 1317.358        |typical
14.8       | 237.1     | 48.897    | 782.357         |typical
3.5        | 117891.6  | 7.826     | 261507.266      |typical
4.2        | 17027.5   | 8.697     | 34980.664       |typical
5.6        | 7146.3    | 9.638     | 12201.467       |typical
4.9        | 6262.9    | 12.926    | 16390.549       |typical
5.6        | 121380.1  | 13.021    | 279988.438      |typical
8.5        | 6646.8    | 21.913    | 17201.861       |typical
```

Related Topics

Report Commands

report_clock_gating

Reports the information on clock gating of flip-flops done by the tool.

Usage

report_clock_gating [-detail] [-short] [-type {all |gated | ungated}] [-module <module_name>] [-instance <instance_name>] [-clock_domain] [-hier_names]

Arguments

- -detail

  Reports the name of each clock gated flip-flop (or not clock gated, if -type is all or ungated). Includes additional columns such as “# occur”, “cell”, and “S”. The “S” column identifies the source of the clock gating instance. “U” denotes that the clock gate was instantiated by the user, and “R” denotes that the clock gate was instantiated by the tool. By default, this command only reports a summary for each module.

- -short

  Specifies that the report summarizes the clock gating coverage at the module level.

- -type {all |gated | ungated}

  Reports clock gating information for the type of flip-flop specified. Can be gated, ungated, or all. The default is all.

- -module <module_name>

  Reports clock gating information only for the module specified. The default reports for all modules in the design.

- -instance <instance_name>

  Specifies the flip-flop instance for which clock gating information is to be reported. The instance name is a hierarchical name relative to the top module by default. If the -module option is specified, the name is relative to the module name. This option only applies to a clock gating cell or a flip-flop.

- -clock_domain

  Generates a report classified by clock domains. A clock domain is defined by a base clock and contains all instances that are driven by this clock. These instances may be directly connected to the clock source or connected through clock gating cells and buffers as part of the clock tree.

  With the -clock_domain option, the report_clock_gating command reports the information about the domain name, the number of clock gating instances in the clock domain, the number of flip-flops that are connected through clock gating instances, and the number of flip-flops that are not gated along with their respective percentages.

  If -clock_domain is used with the -detail option, it provides a similar report with names of clock gating cells and flip-flops. As a clock domain can encompass multiple hierarchical modules, the names in the report are full hierarchical names.

- -hier_names

  Specifies that the reported instance names be full hierarchical names relative to the top module by default. If the -module option is specified, the names are relative to the specified module name.

Description

Provides information on clock gating of flip-flops done by the Oasys-RTL tool. A module or instance can be specified. Based on the options provided, it can also provide information on the flip-flops that are not clock gated. The default report format is a short summary report.

The “fanout clock gates” column contains information about clock gating instances that are connected to the output of a clock gate. It contains the number of clock gates in the immediate fanout (which is the next stage) of a clock gate. The subsequent names are the next stage clock gates, which are followed by the flip-flops connected to the clock gates.

Examples

Example 1: Report with Default Options

This example generates a clock gating report.

```
[oasys-RTL]$ report_clock_gating
Report Clock Gating:
Clock Gating Summary for    : sparc

Number of total flops       : 19821
Number of gated flops       : 1255 ( 6.33 % )
User Clock gating Instances : 0
RT   Clock Gating Instances : 130
Maximum_number of stages    : 1
Minimum_number of stages    : 1
Number of ungated flops     : 18566 ( 93.67 % )
Gating prevented due to     :
- no enable logic           : 18201 ( 98.03 %)
- small width               : 365 ( 1.97 %)
- dont_clock_gate           : 0 ( 0.00 %)
- no clock-gate cell found  : 0 ( 0.00 %)
```

Example 2: Report for a Specific Instance

This example generates a clock gating report for the instance clk_gate_out5_reg.

```
[oasys-RTL]$ report_clock_gating -instance clk_gate_out5_reg
warning: cannot find instance 'clk_gate_out5_reg' in module 'onereg'  [CMD-103]
legend:  S - Source of clock gating  U - User instantiated clock gate in RTL  R - Oasys-RTL
inserted clock gate
Report Clock Gating:
------+-----------------+-----+-----------------+--------------
module|clock-gate insts |Stage|fanout clock-gates|clock-gated ffs
------+-----------------+-----+-----------------+--------------
top   |clk_gate_out5_reg|    2|                0|             4
      |                 |     |                 |out5_reg
      |                 |     |                 |out6_reg
      |                 |     |                 |out7_reg
      |                 |     |                 |out8_reg
------+-----------------+-----+-----------------+--------------
```

Example 3: Report with Default Options for a Design with Hierarchy

This example generates a clock gating report for a design with hierarchy.

```
[oasys-RTL]$ report_clock_gating
Report Clock Gating:
Clock Gating Summary for    : top

Number of total flops       : 12
Number of gated flops       : 12 ( 100.00 % )
User Clock gating Instances : 0
RT   Clock Gating Instances : 3
Maximum_number of stages    : 2
Minimum_number of stages    : 1
Number of ungated flops     : 0 ( 0.00 % )

[oasys-RTL]$ report_clock_gating -module onereg
Report Clock Gating:
Clock Gating Summary for    : onereg

Number of total flops       : 4
Number of gated flops       : 4 ( 100.00 % )
User Clock gating Instances : 0
RT   Clock Gating Instances : 1
Maximum_number of stages    : 1
Minimum_number of stages    : 1
Number of ungated flops     : 0 ( 0.00 % )
```

Example 4: Report with the -detail

This example generates a detailed report of the clock gates.

```
[oasys-RTL]$ report_clock_gating -detail
Report Clock Gating:
Clock Gating Summary for    : top

Number of total flops       : 12
Number of gated flops       : 12 ( 100.00 % )
User Clock gating Instances : 0
RT   Clock Gating Instances : 3
Maximum_number of stages    : 2
Minimum_number of stages    : 1
Number of ungated flops     : 0 ( 0.00 % )
legend:  S - Source of clock gating  U - User instantiated clock gate in RTL  R - Oasys-RTL
inserted clock gate
legend:  fanout - total fanout (clock-gates)
Report Clock Gating:
--------------------+-+-----+------
clock-gate insts    |S|Stage|fanout
--------------------+-+-----+------
bank4/clk_gate_q_reg|R|    1|4 (0)
clk_gate_always_reg |R|    1|5 (1)
clk_gate_out5_reg   |R|    2|4 (0)
--------------------+-+-----+------
legend:  clock-gate insts - total (user instantiated)
Report Clock Gating:
------------+---+---------------+--------+-+-----+------+-------+------+------+----+-------
module      |#  |clock-gate     |cell    |S|Stage|fanout|clock- |%     |non-  |%   |reason
            |oc-|insts          |        | |     |clock-|gated  |%     |clock-|%   |for
            |cur|               |        | |     |gates |ffs    |%     |gated |%   |not
            |   |               |        | |     |      |       |%     |ffs   |%   |gating
------------+---+---------------+--------+-+-----+------+-------+------+------+----+-------
design total|   |3 (0)          |        | |    2|      |     12|100.00|     0|0.00|
onereg      |  1|1( 0)          |        | |    1|      |      4| 33.33|     0|0.00|
            |   |clk_gate_q_reg |TLATNCX2|R|    1|     0|      4| 33.33|      |    |
            |   |               |        | |     |      |reg[0] |      |      |    |
            |   |               |        | |     |      |reg[1] |      |      |    |
            |   |               |        | |     |      |reg[2] |      |      |    |
            |   |               |        | |     |      |reg[3] |      |      |    |
top         |  1|2 (0)          |        | |    2|      |      8| 66.67|     0|0.00|
            |   |clk_g_always_reg|TLATNCX2|R|    1|     1|      4| 33.33|      |    |
            |   |               |        | |     |ck_reg|       |      |      |    |
            |   |               |        | |     |      |ot1_reg|      |      |    |
            |   |               |        | |     |      |ot2_reg|      |      |    |
            |   |               |        | |     |      |ot3_reg|      |      |    |
            |   |               |        | |     |      |ot4_reg|      |      |    |
            |   |clk_gt_out5_reg|TLATNCX2|R|    2|     0|      4| 33.33|      |    |
            |   |               |        | |     |      |ot5_reg|      |      |    |
            |   |               |        | |     |      |ot6_reg|      |      |    |
            |   |               |        | |     |      |ot7_reg|      |      |    |
            |   |               |        | |     |      |ot8_reg|      |      |    |
------------+---+---------------+--------+-+-----+------+-------+------+------+----+-------
```

Related Topics

Report Commands

report_clock_gating_options

report_clock_gating_cell

synthesize

set_parameter

report_clock_gating_cell

Reports the integrated clock-gating cells used for clock-gating flip-flops.

Usage

report_clock_gating_cell

Arguments

None.

Description

This command displays the integrated clock-gating cells that Oasys places in the design during synthesis. You can configure the clock-gating parameters with the set_clock_gating_options command.

Note

Oasys places integrated clock-gating cells by default. If you do not want to include clock-gating cells in the design, set the synthesize -gate_clock option to false.

Examples

```
% report_clock_gating_cell

info: clock-gating cell for posedge FFs = TLATNCAX2 in target library
'default' [POWER-112]

info: no clock-gating cell found in target library 'default' for negedge
FFs for the given specification [POWER-113]
```

Related Topics

Report Commands

report_clock_gating_options

report_clock_gating

set_parameter

synthesize

report_clock_gating_options

Reports the options set for clock-gating flip-flops.

Usage

report_clock_gating_options [-instance <inst>]

Arguments

- -instance <inst>

  Specifies that the tool return the clock-gating options for the specified instance, <inst>.

Description

Reports the options for clock-gating flip-flops, which you can configure with the set_clock_gating_options command.

Examples

```
% report_clock_gating_options

info: clock_gating minimum_width = 4, sequential_cell = latch,
control_port = (null), control_point = none, observability = no [POWER-
111]
```

Related Topics

Report Commands

synthesize

report_clock_gating

report_clock_gating_cell

set_clock_gating_options

set_parameter

report_clocks

Reports the clock constraints that were defined with SDC commands.

Usage

report_clocks [-mode <mode>] [-clocks <clock_name>] [-detail]

Arguments

- -mode <mode>

  Displays clocks from the specified timing mode.

- -clocks <clock_name>

  Displays clocks matching the specified name.

- -detail

  Includes the following additional columns in the report: Total Negative Slack, Violating End Points, Total End Points, Waveform, Edges, and Pin.

Description

Reports the information about the clocks defined in SDC: worst slack for each clock domain, clock period, uncertainty values, transition values, latency, and master clock. If there is no path with negative slack, the smallest positive slack value is reported in the Worst Slack column. If there are generated clocks, the Master clock of these is shown. Values are in picoseconds (ps). Use the -detail option to display additional information as described above.

Examples

Example 1: Sample Output for report_clocks

The following example shows a multimode example with two timing modes: m1 and m2.

```
% report_clocks
Report Clocks For Mode "m1":
-----+-----+-----+------+-----------+----------+-------+------+------
     |Clock|Worst|Period|Uncertainty|Transition|Latency|Master|Timing
     |     |Slack|      |           |          |       |      | Mode
-----+-----+-----+------+-----------+----------+-------+------+------
1    |clk1 |521.4|1000.0|        0.0|       0.0|    0.0|      |m1
2    |clk2 |383.5|2000.0|        0.0|       0.0|    0.0|      |m1
-----+-----+-----+------+-----------+----------+-------+------+------
Report Clocks For Mode "m2":
-----+-----+-----+------+-----------+----------+-------+------+------
     |Clock|Worst|Period|Uncertainty|Transition|Latency|Master|Timing
     |     |Slack|      |           |          |       |      | Mode
-----+-----+-----+------+-----------+----------+-------+------+------
1    |clk1 |121.4| 500.0|        0.0|       0.0|    0.0|      |m2
2    |clk2 |-16.5|1000.0|        0.0|       0.0|    0.0|      |m2
-----+-----+-----+------+-----------+----------+-------+------+------
```

Related Topics

Report Commands

report_congestion

Reports the routing congestion statistics for the design.

Usage

report_congestion [-hotspot <double>] [-instance <string>] [-leaf] [-max_level <integer>] [-max_trace <integer>] [-style {histogram | hierarchical}] [-threshold <double>]

Arguments

- -hotspot <double>

  Optional. Specifies the threshold for identifying a hot spot. If an instance happens to be in such a region, it will be flagged as a hot spot. The value for this option must be between 0 and 1.0, inclusive. When set to 0, only the spot that has the highest routing usage is treated as a hot spot. When set to 1, all places that have congestion overflow are treated as hot spots. The default value is 0.5.

- -instance <string>

  Optional. Limits reporting to the specified instance argument.

- -leaf

  Optional. Includes leaf-level instances in the report. The default value is false, where only hierarchical instances are included.

- -max_level <integer>

  Optional. Specifies the depth of hierarchy for the reporting. The default value is 100. You can use a much smaller number, in order to limit the reporting to a few of the higher hierarchy levels.

- -max_trace <integer>

  Optional. Controls the maximum RTL traces to be shown against each instance in the table. For the first row in table, which is the top module, there is not any trace. The default limit for the number of RTL traces to be displayed under each instance is 10. If the -max_trace option is set to 0, then the last “RTL” column disappears from the table.

- -style {histogram | hierarchical}

  Optional. Specifies the style of the report. The styles are defined as follows:

  - **histogram** - Provides a report that groups congestion by ranges of track overflow. This is the default.
  - **hierarchical** - Provides a report that traverses the logical hierarchy and reports congestion metrics such as the average congestion and average congestion over

congested area for all the instances that have congestion. In this style, the following metrics are reported in the table generated by the command:

Table 5-6. Parameters Reported by the "-style hierarchal" Option

Parameters	Method of Calculation
Average overflow (%)	(horizontal_overflow + vertical_overflow)/ total_tracks_of_design
Congested area (%)	Percentage of congested area of the total area of the instances
Average overflow over congested area (%)	(horizontal_overflow + vertical_overflow)/ total_congested_tracks_of_design
Local usage (%)	(horizontal_usage + vertical_usage)/ total_tracks_of_instance
Local usage over congested area (%)	(horizontal_usage + vertical_usage)/ total_tracks_of_instance_in_congested_area
Local reserved usage(%)	(horizontal_reserved_tracks + vertical_reserved_tracks)/total_tracks_of_instance

- -threshold <double>

  Optional. Limits the reporting to instances having a congestion beyond the specified threshold value. The default threshold value is 0.01, where all instances greater than 0.01 are reported.

  Reporting is also dependent on two more options (-hotspot and -max_level). The report reports the hierarchical instances in the entire hierarchy tree to a depth specified by the -max_level option only if the instance satisfies *both* of the following conditions:

  - Its congestion value is larger than the threshold value, or its hotspot value is larger than the value specified by the -hotspot option, *and*
  - The hierarchy level of the instance is lower than the value specified by the -max_level option.

Description

The report_congestion command reports routing congestion statistics in ASCII format for the current design. Using the physical viewer in the GUI, you can see congestion distribution in the design. The tabular representation of the report_congestion command provides detailed quantitative metrics of the congestion in the design and more granular information at the module level. This report also directs you to the specific section of the RTL source file responsible for the congestion. This enables you to cross-probe from the physical viewer to the RTL source in order to identify RTL issues and fix them in RTL.The design must be placed prior to running this command.

Examples

Note

Some columns in tables in the following examples area truncated or abbreviated due to space limitations.

Example

```
[oasys-RTL]$ report_congestion

% bins with overflow
% overflow   Both         H            V
    30+      0.000      0.000        0.002
    20+      0.000      0.005        0.041
    10+      0.026      0.005        0.103
     0+      0.363      0.019        0.214
   none     99.612     99.970       99.641info: total congestion : 9571
tracks (958 horizontal, 8613 vertical) [PLACE-119]

info: average percentage of overflow: 0.0166645% overall,
4.29% over congested areas (0.75% horizontal,
9.01% vertical) [PLACE-120]

info: peak bin congestion : horizontal 43 tracks, vertical 41 tracks, max
43 tracks [PLACE-121]

info: peak bin congestion : horizontal 234 tracks,
vertical 168 tracks, max 370 tracks [PLACE-121]
```

Example

This example uses the “-style hierarchical” option to report congestion in hierarchical mode. It also shows the “RTL” trace column added in the report (last column).

```
[oasys-RTL]$ report_congestion -style hierarchical

Report Congestions:
```

instance	module	average overflow (%)	congested area (%)	average overflow over congested area (%)	local usage (%)	local usage over congested area (%)	local reserved usage(%)	RTL
sparc		0.27598	4.438	6.15369	4.89436	21.47652	0.00000	
exu	sparc_exu	0.20138	18.374	4.49037	8.26829	31.49178	0.00000	/<path>
bypass	sparc_exu_byp	0.17739	63.636	3.95532	24.88831	35.74966	0.00000	/<path>
*rcc_data_dff	dff_s__param…	0.03902	100.000	0.87012	38.85714	38.85714	0.00000	/<path>
*rs2_data_dff	dff_s__param…	0.03542	100.000	0.78986	64.38095	64.38095	0.00000	/<path>
div	sparc_exu_div	0.14895	48.905	3.32129	21.42510	39.12580	0.00000	/<path>
alu	sparc_exu_alu	0.02849	69.697	0.63537	19.84848	23.92340	0.00000	/<path>
addsub	sparc_exu…	0.01290	56.818	0.28759	15.50866	19.48571	0.00000	/<path>
tlu	tlu	0.02915	6.418	0.65008	5.54370	13.07812	0.00000	/<path>
tdp	tlu_tdp	0.02592	22.575	0.57785	8.22939	15.71822	0.00000	/<path>
ifu	sparc_ifu	0.02717	4.363	0.60594	5.33355	23.62890	0.00000	/<path>
mbist	sparc_ifu…	0.01725	85.714	0.38456	6.44403	7.30159	0.00000	/<path>
lsu	lsu	0.01887	2.099	0.42068	2.34261	6.00221	0.00000	/<path>

Example

This example reports congestion with the -max_trace option set to 0. When -max_trace is 0, the “RTL” column disappears from report.

```
[oasys-RTL]$ report_congestion -style hierarchical -max_trace 0

Report Congestions:
```

instance	module	average overflow (%)	congested area (%)	average overflow over congested area (%)	local usage (%)	local usage over congested area (%)	local reserved usage(%)
sparc		0.27598	4.438	6.15369	4.89436	21.47652	0.00000
exu	sparc_exu	0.20138	18.374	4.49037	8.26829	31.49178	0.00000
bypass	sparc_exu_byp	0.17739	63.636	3.95532	24.88831	35.74966	0.00000
*rcc_data_dff	dff_s__parameterized19__83	0.03902	100.000	0.87012	38.85714	38.85714	0.00000
*rs2_data_dff	dff_s__parameterized19__83	0.03542	100.000	0.78986	64.38095	64.38095	0.00000
div	sparc_exu_div	0.14895	48.905	3.32129	21.42510	39.12580	0.00000
alu	sparc_exu_alu	0.02849	69.697	0.63537	19.84848	23.92340	0.00000
addsub	sparc_exu_aluaddsub	0.01290	56.818	0.28759	15.50866	19.48571	0.00000
tlu	tlu	0.02915	6.418	0.65008	5.54370	13.07812	0.00000
tdp	tlu_tdp	0.02592	22.575	0.57785	8.22939	15.71822	0.00000
ifu	sparc_ifu	0.02717	4.363	0.60594	5.33355	23.62890	0.00000
mbist	sparc_ifu_mbist	0.01725	85.714	0.38456	6.44403	7.30159	0.00000
lsu	lsu	0.01887	2.099	0.42068	2.34261	6.00221	0.00000

Example

This example reports congestion with the -hotspot option set to 1 and the -threshold option set to 0.1. When the -hotspot and -threshold options are used simultaneously, all congested modules having overflow more than the threshold are reported along with “rcc_data…” and “rs2_data…” instances. This is because setting the -hotspot option to 1 shows all places that have congestion overflow.

```
[oasys-RTL]$ report_congestion -style hierarchical -hotspot 1 -threshold
0.1

Report Congestions:
```

```
-----------------+-----------------+--------+--------+---------------+--------+-------------+--------+-----------
                 |                 |average |congested|average overflow|local   | local usage  |  local |
instance         |module           |overflow|   area  |over congested  |usage   |over congested|reserved|     RTL
                 |                 |  (%)   |   (%)   |   area (%)     | (%)    |   area (%)   |usage(%)|
-----------------+-----------------+--------+--------+---------------+--------+-------------+--------+-----------
sparc            |                 | 0.27598|   4.438|         6.15369| 4.89436|      21.47652|
0.00000|
  exu            |sparc_exu        | 0.20138|  18.374|         4.49037| 8.26829|      31.49178| 0.00000|/<path>
    bypass       |sparc_exu_byp    | 0.17739|  63.636|         3.95532|24.88831|      35.74966| 0.00000|/<path>
      *rcc_data_dff|dff_s__parameterized| 0.03902|  100.000|         0.87012|38.85714|      38.85714| 0.00000|/<path>
      *rs2_data_dff|dff_s__parameterized| 0.03542|  100.000|         0.78986|64.38095|      64.38095| 0.00000|/<path>
    div          |sparc_exu_div    | 0.14895|  48.905|         3.32129|21.42510|      39.12580| 0.00000|/<path>
-----------------+-----------------+--------+--------+---------------+--------+-------------+--------+-----------
```

Example

This example reports congestion with the -hotspot option set to 0 and the -threshold option set to 0.1. When the -hotspot option is 0, the two instances “rcc_data…” and “rs2_data…” are dropped from report because the highest routing usage is treated as a hot spot.

```
[oasys-RTL]$ report_congestion -style hierarchical -hotspot 0 -threshold
0.1

Report Congestions:
```

```
----------+-------------+--------+--------+---------------+--------+-------------+--------+-----------
          |             |average |congested|average overflow|local   | local usage  |  local |
instance  |module       |overflow|   area  |over congested  |usage   |over congested|reserved|     RTL
          |             |  (%)   |   (%)   |   area (%)     | (%)    |   area (%)   |usage(%)|
----------+-------------+--------+--------+---------------+--------+-------------+--------+-----------
sparc     |             | 0.27598|   4.438|         6.15369| 4.89436|      21.47652| 0.00000|
  exu     |sparc_exu    | 0.20138|  18.374|         4.49037| 8.26829|      31.49178| 0.00000|/<path>
    bypass|sparc_exu_byp| 0.17739|  63.636|         3.95532|24.88831|      35.74966| 0.00000|/<path>
    div   |sparc_exu_div| 0.14895|  48.905|         3.32129|21.42510|      39.12580| 0.00000|/<path>
----------+-------------+--------+--------+---------------+--------+-------------+--------+-----------
```

Example

This example reports congestion with the -max_level option set to 5. Using “-max_level 5” causes instances down to the fifth level of the hierarchy to be reported.

```
[oasys-RTL]$ report_congestion -style hier -max_level 5

Report Congestions:
```

instance	module	average overflow (%)	congested area (%)	average overflow over congested area (%)	local usage (%)	local usage over congested area (%)	local reserved usage(%)	RTL
tf_crcv		1.41464	34.564	10.71719	9.57348	17.00917	17.93914	
tf_crcvd	tf_crcv_d	0.74254	40.549	5.62538	8.46242	13.49507	17.95431	<path>
dpath	tf_crcv_dpath_JDcell_JVdat	0.46025	53.958	3.48680	4.68239	5.70070	17.97403	/<path>
always__246_47691068428384	[always__246]	0.32048	88.480	2.42794	3.34400	3.67269	18.08080	<path>0
crcv_mmgr_payload_r1_reg	[crcv_mmgr_payload_r1_reg]	0.20877	96.577	1.58164	4.50734	4.66709	18.12959	<path>
block_47691068429600	[block]	0.13719	88.306	1.03931	3.12702	3.32192	18.08266	<path>
crcv_mmgr_payload_reg	[crcv_mmgr_payload_reg]	0.12294	94.237	0.93136	0.43729	0.46403	18.08813	<path>
always_47691070505440	[always]	0.09376	50.398	0.71034	1.60477	2.98421	17.87135	<path>
rrau_payload_s0_reg	[rrau_payload_s0_reg]	0.09364	54.046	0.70943	1.65896	3.02139	17.91763	<path>
always__243_47691067676896	[always__243]	0.07012	38.843	0.53124	4.59504	6.26950	18.00413	<path>
block_47691068245792	[block]	0.02082	62.712	0.15773	12.10170	11.82432	18.07627	<path>
per_port_cnt	tf_stat_counter_array_JDdep…	0.06967	44.964	0.52782	7.65647	11.22800	17.98381	<path>
always_47691067481184	[always]	0.04030	45.562	0.30528	3.67751	3.40260	17.92899	<path>
fctl	tf_crcv_fctl	0.28579	33.819	2.16509	11.39113	22.31495	17.92456	<path>

Related Topics

Report Commands

report_delay

Reports the delay associated with a particular cell instance or net.

Usage

report_delay **-delay_type** { **instance** | **net** } **-from** ***pin_name*** **-to** ***pin_name*** [-min|-max] [-derate *bool*] [-compute_ideal *bool*]

Arguments

- **-delay_type** { **instance** | **net** }

  Specifies to report a cell/instance delay or a net delay.

- **-from** ***pin_name***

  Specifies the start pin for the delay calculation.

- **-to** ***pin_name***

  Specifies the destination pin for the delay calculation.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Specifies to report the maximum delay. This is the default.

- -derate *bool*

  Optional argument specifying whether to apply a derating factor to the cell/instance. This option is only valid when '-delay_type instance' is also applied. If a derate value exists for the cell/instance and you do not specify -derate, the default derate value is applied.

  Note

  Set derating factors with the set_timing_derate command.

- -compute_ideal *bool*

  Optional argument that specifies to report an actual delay for ideal cells/instances. If you do not specify this argument, ideal cells (clock-gating cells, for example) will report zero delay. Default: false.

Description

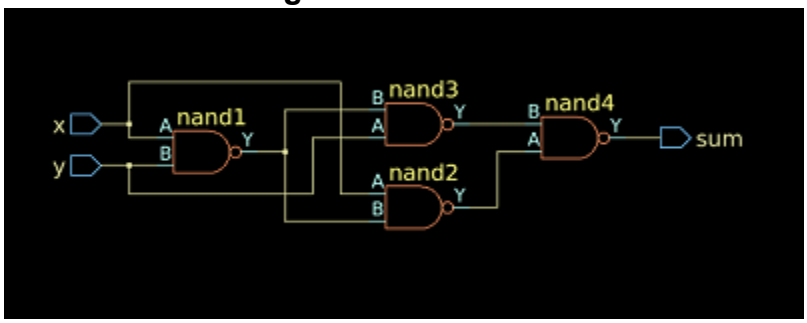
Report the delay associated with the specified cell instance or net. Prior to running this command, Oasys-RTL must generate the timing calculations using either the report_timing or worst_slack commands.

Examples

Example 1

The following example shows reports the instance delay from input pin B to output pin Y on the nand3 cell.

Figure 5-2. Circuit 1




```
% report_delay -delay_type instance -from nand3/B -to nand3/Y
Report Delay Instance:
------------------------+----------------
Attribute               |Value
------------------------+----------------
Time/Cap Unit           |(ps)/(ff)
From Pin                |nand3/B
To Pin                  |nand3/Y
Timing Mode             |default
Arc Type                |comb
Arc Sense               |negative_unate
Master Cell             |NAND2X1
Pin Library             |typical
Total Delay (Rise, Fall)|-{28.1ps,23.0ps}
Total Capacitance       |8.76ff
Input Transition        |{Fall, Rise}
Output Transition       |{Rise, Fall}
Input Slew              |{36.1ps,37.5ps}
Output Slew             |{30.1ps,22.2ps}
Derating Factor         |           1.000
------------------------+----------------
```

Example 2

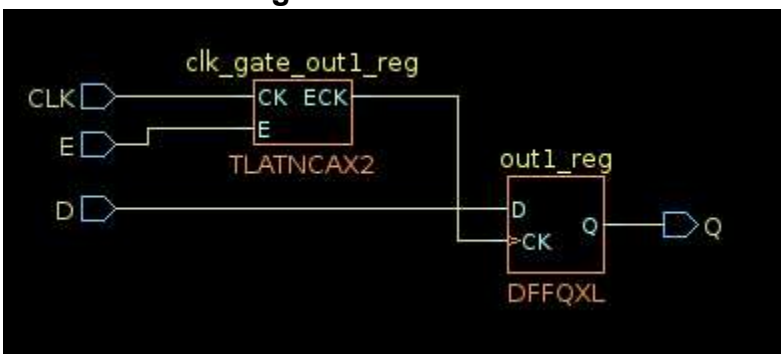
Using the same circuit, the following example reports the net delay between the output pin of the nand2 cell and the ‘A’ input of the nand4 cell.

```
% report_delay -delay_type net -from nand2/Y -to nand4/A
Report Delay Net:
---------------------+----------------
Attribute            |Value
---------------------+----------------
Time/Cap/Res Unit    |(ps)/(ff)/(ohm)
From Pin             |nand2/Y
To Pin               |nand4/A
Net Delay            |             0.0
Slew (Rise/Fall)     |  72.100/91.700
Wire Length          |           7.655
Pin Capacitance      |             1.7
Total Capacitance    |             1.6
Total Resistance     |           6.350
---------------------+----------------
```

Example 3

The following example reports the cell/instance delay for an ideal clock gate. Notice that the delay is reported as zero.

Figure 5-3. Circuit 2




```
% report_delay -delay_type instance -from clk_gate_out1_reg/CK -to
clk_gate_out1_reg/ECK
Report Delay Instance:
------------------------+--------------------
Attribute               |Value
------------------------+--------------------
From Pin                |clk_gate_out1_reg/CK
To Pin                  |clk_gate_out1_reg/ECK
Mode                    |default
Arc Type                |comb
Arc Sense               |positive_unate
Master Cell             |TLATNCAX2
Pin Library             |typical
Total Delay (Rise, Fall)|+{0.0ps,0.0ps}
Total Capacitance       |0.00ff
Time/Cap Unit           |(ps)/(ff)
Output Transition       |{Rise, Fall}
Input Transition        |{Rise, Fall}
Output Transition Time  |{9.0ps,9.0ps}
Input Transition Time   |{9.0ps,9.0ps}
Derating Factor         |               1.000
------------------------+--------------------
```

To report the actual delay for an ideal cell, specify the -compute_ideal argument.

```
% report_delay -delay_type instance -from clk_gate_out1_reg/CK -to
clk_gate_out1_reg/ECK -compute_ideal
Report Delay Instance:
------------------------+--------------------
Attribute               |Value
------------------------+--------------------
From Pin                |clk_gate_out1_reg/CK
To Pin                  |clk_gate_out1_reg/ECK
Mode                    |default
Arc Type                |comb
Arc Sense               |positive_unate
Master Cell             |TLATNCAX2
Pin Library             |typical
Total Delay (Rise, Fall)|+{43.0ps,52.6ps}
Total Capacitance       |0.00ff
Time/Cap Unit           |(ps)/(ff)
Output Transition       |{Rise, Fall}
Input Transition        |{Rise, Fall}
Output Transition Time  |{22.9ps,25.8ps}
Input Transition Time   |{9.0ps,9.0ps}
Derating Factor         |                 1.000
------------------------+--------------------
```

Related Topics

Report Commands

set_timing_derate

report_design_metrics

Reports the design characteristics of the top-level design in the session. Design size, power, instance count, physical data including die size, and congestion are reported.

Usage

report_design_metrics

Arguments

None.

Description

If this command is run after placement, the physical information is also provided to show the utilization and wire length. It also indicates whether the design is placed or unplaced.

The chip utilization is based on the total cell area, which includes all the standard cells, macros, I/O pads, and physical-only cells.

```
Chip Utilization = Total Cell Area / Die Area
```

The placement utilization is based on the total area of the standard cells only. The placeable area is the die area minus the fixed cell area. The fixed cell area includes the fixed macros, physical-only cells, I/O pads, and blockages.

Well ties, filler cells, and decap cells are treated like blockages during the placement (just like fixed macros). They are not overlapped by any other standard cells.

```
Standard Cell Utilization = Area of Standard Cells / Placeable Area

Placeable Area = Die Area - Fixed Cell Area
```

Examples

```
% report_design_metrics

Report Design Metrics:
------------------------+-------------+-----------+------------
                        |             |Area (squm)|Leakage (uW)
------------------------+-------------+-----------+------------
Design Name             |sparc        |           |
  Total Instances       |       155345|     542157|      13.679
    Macros              |           18|     294036|      12.180
    Pads                |            0|          0|       0.000
    Phys                |            0|          0|       0.000
    Blackboxes          |            0|          0|       0.000
    Cells               |       155327|     248121|       1.499
      Buffers           |          184|        147|       0.001
      Inverters         |        35845|      19070|       0.127
      Combinational     |        99311|     106835|       0.678
      Latches           |          166|        442|       0.002
      Registers         |        19821|     121628|       0.691
        Load-Enabled    |            0|           |
        Clock-Gated     |         1255|           |
  Tristate Pin Count    |            0|           |
Physical Info           |Placed       |           |
  Chip Size (mm x mm)   |0.881 x 0.881|     776937|
  Fixed Cell Area       |             |     294036|
    Phys Only           |            0|          0|
  Placeable Area        |             |     350976|
  Movable Cell Area     |             |     248121|
  Utilization (%)       |           70|           |
  Chip Utilization (%)  |           72|           |
  Total Wire Length (mm)|     5884.709|           |
  Longest Wire (mm)     |        1.742|           |
  Average Wire (mm)     |        0.244|           |
------------------------+-------------+-----------+------------
```

Related Topics

Report Commands

report_dft_partitions

Reports the instances for all DFT partitions.

Usage

report_dft_partitions

Arguments

None.

Description

DFT partitions are sections of the design created by the define_dft_partition command for scan connection. To associate each DFT partition with its own set of scan chains, apply the -partition option of the define_scan_chain command.

Examples

In this example, the R0 DFT partition contains the r0 instance. The R1S1 partition R1S1 contains the r1 and s1 instances.

```
% report_dft_partitions

Report DftPartitions:
--+-----------+----------
  | Partition | Instance
--+-----------+----------
1 | R0        | r0
2 | R1S1      | r1
3 |           | s1
--+-----------+----------
```

Related Topics

Report Commands

define_dft_partition

define_scan_chain

report_dft_registers

Reports the DFT related information about the registers of the design in session.

Usage

report_dft_registers

Arguments

None.

Examples

```
% report_dft_registers

Report DftRegisters:
--+------------------+--------+-------+--------+--------------
  | Register         | Status | Clock | Edge   | Dft Violation
ID
--+------------------+--------+-------+--------+--------------

1 | i_0/s0/r_reg\[0\]| pass   | ck[3] | rising |
2 | i_0/s0/r_reg\[1\]| pass   | ck[3] | rising |
3 | i_0/s0/r_reg\[2\]| pass   | ck[3] | rising |
4 | i_0/s0/r_reg\[3\]| pass   | ck[3] | rising |
5 | i_0/s0/r_reg\[4\]| pass   | ck[3] | falling|
…
```

Related Topics

Report Commands

report_dft_violations

Reports the type, the pin, and the number of affected registers for each DFT violation.

Usage

report_dft_violations [-detail]

Arguments

- -detail

  Includes additional details.

Examples

```
% report_dft_violations

Report DftViolations:
---+----------------------------------------+-------------+-------------
   | Type                                   | Pin         | # Affected
   |                                        |             |   Registers
---+----------------------------------------+-------------+-------------
1  |primary pin not declared as a test-clock|ck           | 2
2  |internal clock driver                   |i_5/i_0/Y    | 1
3  |internal clock driver                   |i_4/i_0/Y    | 1
4  |internal clock driver                   |r0/out_reg/Q | 1
5  |internal clock driver                   |i_3/i_0/Y    | 1
6  |internal clock driver                   |i_2/i_0/Y    | 1
7  |internal clock driver                   |i_1/i_0/Y    | 1
8  |internal clock driver                   |i_0/i_0/Y    | 1
9  |primary pin not declared as a test-clock|asr          | 1
10 |constant 0/1 clock                      |r9/ck        | 1
11 |blocking clock gate                     |g10/clk_gate_|
                                            |ireg_reg/ECK | 8
---+----------------------------------------+-------------+-------------
```

Related Topics

Report Commands

report_edges

Reports the assignment of macro groups and instances to various edges of the die.

Usage

report_edges

Arguments

None.

Description

Reports the edges along with the length of each edge, their begin and end coordinates, and corresponding edge assignments.

Examples

Assume the edge assignments are as follows:

```
assign_macros_to_edges -groups {G1 G5} -edges {0:1/3 0:2/3 0:3/3} \
   -exclusive
assign_macros_to_edges -sequence {{G3} {G4}} -edges 1 -align begin
assign_macros_to_edges -instance [get_cells ffu/frf] -groups G2 -edges 2
```

Running the report_edges command generates a report corresponding to these assignments:

```
[oasys-RTL]$ report_edges

Report Edges:
--+---------+-----+------+---------------------------+-----------------------------
  |Floorplan|Edges|Length|Edge Begin/End Points      |Edge Assignments
  |Area     |     |      |                           |
--+---------+-----+------+---------------------------+-----------------------------
1 |Core     |    0|881.44|(0.00 0.00)(0.00 881.44)   |{G1 0:1/3 0:2/3 0:3/3 X}
  |         |     |      |                           |   {G5 0:1/3 0:2/3 0:3/3 X}
2 |Core     |    1|881.44|(0.00 881.44)(881.44 881.44)|{G3 1:1 B}{G4 1:2 B}
3 |Core     |    2|881.44|(881.44 881.44)(881.44 0.00)|{G2 2}{ffu/frf 2}
4 |Core     |    3|881.44|(881.44 0.00)(0.00 0.00)   |
--+---------+-----+------+---------------------------+-----------------------------
Edge assignment notation -  edge_number:subsection:sequence {B|E|X},
B:packing from begining, E:packing from end, X exclusive
```

For details on specifying and interpreting edge assignments, see “Identifying Edges and Edge Partitions” on page 23.

report_electrical_violations

Reports the electrical violations for the specified power domain(s).

Usage

report_electrical_violations [-domain <domain_name>] [-show_violation_only]

Arguments

- -domain <domain_name>

  This optional argument specifies the source domain name for which the electrical violations are reported. If you do not specify this argument, reports electrical violations for all power domains.

- -show_violation_only

  Outputs only the violations (results marked NOTOK).

Description

Reports electrical violations for the specified power domain(s). The report shows the source domain, boundary driver pin, driver cell, voltage shift, Always-On (AO) relation, boundary load pin, load cell, and sink domain.

The voltage shift column shows the minimum and maximum voltage shift between the domains. The column is marked similar to the OK(0.00,0.00) if there is no violation and NOTOK(0.10,0.15) if there is one. The two numbers are for the minimum and maximum respectively. The AO relation choices are shown below. The AO relation column is marked with an OK if there is no violation.

>= More or equally always-on

== Equally always-on

<= Less or equally always-on

<> Independently always-on

Examples

This example shows an electrical violations report. The report only has “==” AO relations. You can see NOTOK(0.15,0.15) that shows minimum and maximum voltage shifts of 0.15 volts, which is a violation. These violations go away when level shifters are automatically added.

```
% report_electrical_violations -domain PD_Default

Report Electrical Violations:
--+-------------+----------------------------------------+------------+
  |source domain|driver pin                              |driver cell |
--+-------------+----------------------------------------+------------+

1 |PD_Default   |ccx/cpx/buf_top/ff0/cpx_spc_data_cx2[0]  | - ...
2 |PD_Default   |ccx/cpx/buf_top/ff0/cpx_spc_data_cx2[0]  | - ...
3 |PD_Default   |ccx/cpx/buf_top/ff0/cpx_spc_data_cx2[100]| - ...
4 |PD_Default   |ccx/cpx/buf_top/ff0/cpx_spc_data_cx2[101]| - ...
5 |PD_Default   |ccx/cpx/buf_top/ff0/cpx_spc_data_cx2[102]| - ...
...
+----------------+----------+-----------------------------------+---
|voltage shift |AO relation|load pin          |load cell |sink domain
.+----------------+----------+-----------------------------------+---
|OK (0.00,0.00)|OK (==)|sparc0/i_0/cpx_spc_data_cx2[0]| |PD_SparcCores
|OK (0.00,0.00)|OK (==)|sparc0/ffu/i_0/cpx_fpu_data[0]| |PD_SparcCores
|OK (0.00,0.00)|OK (==)|sparc0/i_0/cpx_spc_data_cx2[100]|PD_SparcCores
|OK (0.00,0.00)|OK (==)|sparc0/i_0/cpx_spc_data_cx2[101]|PD_SparcCores
|OK (0.00,0.00)|OK (==)|sparc0/i_0/cpx_spc_data_cx2[102]|PD_SparcCores
.
.
|NOTOK(0.15,0.15)|OK (==)|sparc7/CPF_LS/i_153/A |LSLVHV |PD_SparcCores
|NOTOK(0.15,0.15)|OK (==) |sparc7/CPF_LS/i_154/A |LSLVHV |PD_SparcCores
|NOTOK(0.15,0.15)|OK (==) |sparc7/CPF_LS/i_155/A |LSLVHV |PD_SparcCores
```

Related Topics

Report Commands

report_endpoints

Reports the worst endpoints and their slacks, sorted by criticality.

Usage

report_endpoints [-mode <mode>] [-count <number>] [-slack <value>] [-group <group_name>]

Arguments

- -mode <mode>

  Displays endpoints from the specified timing mode.

- -count <number>

  Specifies the maximum number of endpoints to report. The default is 3. Specifying 0 results in printing all endpoints. This value must be a non-negative integer.

- -slack <value>

  Specifies the maximum slack value to report. This argument supports floating-point numbers.

- -group <group_name>

  Specifies a group for which the endpoints are reported.

Description

Generates a report showing the endpoints with the worst slack. The report includes timing information such as shift, delay, and launch and capture clocks. For designs that contain multiple path groups, the results are listed on a group basis by default. You can configure time units with the time_units_for_reports parameter.

Examples

Example

This example outputs the 5 worst case endpoints and their slack times.

```
% report_endpoints -count 5
Report End Points:
-----+-------+------+-------+------+-----+-----+-----+-----------+-----------
     |Group  |Launch|Capture|Shift |Delay|Depth|Slack|Begin Point |End Point
-----+-------+------+-------+------+-----+-----+-----+-----------+-----------
1    |default|GCLK  |GCLK   |1400.0|964.8|   41| 65.9|exu/alu ...|exu/ecl ...
2    |default|GCLK  |GCLK   |1400.0|971.5|   16|-63.9|ifu/fcl ...|tlu/tcl ...
3    |default|GCLK  |GCLK   |1400.0|971.5|   16|-63.9|ifu/fcl ...|tlu/tcl ...
4    |default|GCLK  |GCLK   |1400.0|959.2|   33|-62.4|exu/alu ...|ifu/fdp ...
5    |default|GCLK  |GCLK   |1400.0|958.6|   31|-61.8|exu/alu ...|ifu/fdp ...
-----+-------+------+-------+------+-----+-----+-----+-----------+-----------
```

The following example shows a multimode example with two timing modes: m1 and m2.

```
% report_endpoints
Report End Points:
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
     |Group  |Launch|Capture|Shift|Delay|Depth|Slack|Mode|Begin Point|End Point
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
1    |default|clk1  |clk2   |500.0| 51.0|    1|-16.5|m2  |reg2/Q     |out[0]
2    |default|clk1  |clk1   |500.0|106.4|    3|287.2|m2  |reg2/Q     |reg6/D
3    |default|clk1  |clk1   |500.0| 88.4|    2|304.5|m2  |reg2/Q     |reg5/D
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
Report End Points:
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
     |Group  |Launch|Capture|Shift|Delay|Depth|Slack|Mode|Begin Point|End Point
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
1    |def_grp|clk1  |clk2   |500.0| 23.6|    1|  1.8|m2  |reg5/Q     |out[1]
-----+-------+------+-------+-----+-----+-----+-----+----+-----------+---------
Report End Points:
-----+------+------+-------+------+-----+-----+-----+----+-----------+---------
     |Group |Launch|Capture|Shift |Delay|Depth|Slack|Mode|Begin Point|End Point
-----+------+------+-------+------+-----+-----+-----+----+-----------+---------
1    |m1_grp|clk2  |clk1   |1000.0| 36.7|    2|521.4|m1  |in[2]      |reg6/D
2    |m1_grp|clk2  |clk1   |1000.0| 18.7|    1|538.7|m1  |in[2]      |reg5/D
3    |m1_grp|clk2  |clk1   |1000.0| 12.8|    1|546.0|m1  |in[1]      |reg2/D
-----+------+------+-------+------+-----+-----+-----+----+-----------+---------
Report End Points:
-----+------+------+-------+-----+-----+-----+-----+----+-----------+---------
     |Group |Launch|Capture|Shift|Delay|Depth|Slack|Mode|Begin Point|End Point
-----+------+------+-------+-----+-----+-----+-----+----+-----------+---------
1    |m2_grp|clk2  |clk1   |500.0| 36.7|    2|121.4|m2  |in[2]      |reg6/D
2    |m2_grp|clk2  |clk1   |500.0| 18.7|    1|138.7|m2  |in[2]      |reg5/D
3    |m2_grp|clk2  |clk1   |500.0| 12.8|    1|146.0|m2  |in[1]      |reg2/D
-----+------+------+-------+-----+-----+-----+-----+----+-----------+---------
```

Example

This example only prints out negative slacks below -500.

```
% report_endpoints -slack -500
Report End Points:
--+-------+-------+-------+------+------+-----+------+--------+-------
  |Group  |Launch |Capture|Shift |Delay |Depth|Slack |Begin Pt|End Pt
--+-------+-------+-------+------+------+-----+------+--------+-------
1 |default|DCO_CLK|DCO_CLK|5000.0|5001.3|   51|-630.1|i/r[3]/Q|i/r[0]/D
2 |default|DCO_CLK|DCO_CLK|5000.0|4982.7|   51|-609.4|i/r[3]/Q|i/r[0]/D
3 |default|DCO_CLK|DCO_CLK|5000.0|4963.3|   51|-590.2|i/r[3]/Q|i/r[0]/D
4 |default|DCO_CLK|DCO_CLK|5000.0|4970.0|   48|-547.8|i/r[6]/Q|i/r[0]/D
5 |default|DCO_CLK|DCO_CLK|5000.0|4921.5|   51|-547.1|i/r[3]/Q|i/r[0]/D
6 |default|DCO_CLK|DCO_CLK|5000.0|4911.9|   51|-532.2|i/r[3]/Q|i/r[0]/D
7 |default|DCO_CLK|DCO_CLK|5000.0|4822.4|   48|-508.5|i/r/Q   |i/r[0]/D
--+-------+-------+-------+------+------+-----+------+--------+-------
```

Example

```
This example only prints out negative slacks below -600 for the same
design as the previous example.

% report_endpoints -slack -600
Report End Points:
--+-------+-------+-------+------+------+-----+------+--------+-------
  |Group  |Launch |Capture|Shift |Delay |Depth|Slack |Begin Pt|End Pt
--+-------+-------+-------+------+------+-----+------+--------+-------
1 |default|DCO_CLK|DCO_CLK|5000.0|5001.3|   51|-630.1|i/r[3]/Q|i/r[0]/D
2 |default|DCO_CLK|DCO_CLK|5000.0|4982.7|   51|-609.4|i/r[3]/Q|i/r[0]/D
--+-------+-------+-------+------+------+-----+------+--------+-------
```

Related Topics

Report Commands

time_units_for_reports

report_explore

Creates a table comparing output values of multiple design space exploration runs.

Usage

report_explore [-checkpoint <checkpoint>] [-csv] [-detail] [-directory <path>]
[-format floorplan] [-post_synthesize | -post_optimize] [-status]

Arguments

- -checkpoint <checkpoint>

  Optional. Specifies a user-defined checkpoint if you want a time different from post-synthesize or post-optimize.

- -csv

  Optional. Specifies that a comma-separated value file is output. The data in this file can be imported into Microsoft Excel or similar products.

- -detail

  Optional. Specifies that the generated table includes the following default columns:

  explore_vars, job_ids, wall_clock, cpu_time, frequency, clock_period, worst_slack, worst_slack_path_group, worst_slack_all, total_negative_slack, total_power, instance_count, instance_area, instance_leakage, utilization, wire_length, chip_size, congestion_global, view_phy_criticality, view_phy_hierarchical, view_phy_congestion, view_phy_leakage_power, view_phy_dynamic_power, status, hostname, pid, log_file

  The columns that are reported can be configured using the config_report command.

- -directory <path>

  Optional. Specifies the root directory used for parallel exploration runs.

- -format floorplan

  Optional. Specifies the column headings for the report of a floorplan exploration session. The column headings depend on the other specified s:

  - **No other specified option** - Frequency, WNS, TNS, Wire_length, Qualified, Macro_out_of_bound, Macro_overlap, Total_power, Inst_area, Inst_count, Congestion_global, Status, Host Name, PID
  - **-status** - Job_ids, Status, Hostname, PID, Log
  - **-detail**
    - **Default** - explore_vars, job_ids, wall_clock, cpu_time, qualified, macro_out_of_bound, macro_overlap, frequency, clock_period, worst_slack, worst_slack_path_group, worst_slack_all, total_negative_slack, total_power, instance_count, instance_area, instance_leakage, utilization, wire_length, chip_size, congestion_global, view_phy_criticality, view_phy_hierarchical,

view_phy_congestion, view_phy_leakage_power, view_phy_dynamic_power, status, hostname, pid, log_file

- **Other Available Columns** - internal_power, switching_power, leakage_power, cell_area, cell_count, cell_leakage, chip_height, chip_width, congestion_local

- -post_optimize

  Optional. Reports the exploration results after optimization.

- -post_synthesize

  Optional. Reports the exploration results after synthesis.

- -status

  Optional. Reports the status of parallel runs. See the report_explore -status example below.

Description

Creates a table from a DSE session that compares the output values of the multiple design space exploration runs. This command must not be used in a DSE script. Apply the report_explore command from the same run directory, but in a different Oasys-RTL session. The table columns for the -detail option are configurable with the config_report command.

Examples

Example 1: Report with Default Options

This example creates a report of the DSE results. The WNS, total power, instance area, and instance count are shown for each combination of vt library, voltage, and clock period.

```
[oasys-RTL]$ report_explore
Report exploration metrics (summary):
--+-------+------+-------+-----------+-------+--------------+...+------+------+--------+...
  |clock_ |lib_vt|voltage|Frequency  |WNS(ns)|Total_power   |...|Inst_ |Inst_ |Status  |...
  |period |      |       |(mhz)      |       |(uw)          |...|area  |count |        |...
  |       |      |       |           |       |              |...|(um2) |      |        |...
--+-------+------+-------+-----------+-------+--------------+...+------+------+--------+...
1 |1.2ns  |hvt   |0.85v  | 833.333313|-0.2750| 190626.156250|...|541868|155102|finished|...
2 |1.2ns  |hvt   |0.95v  | 833.333313| 0.0009| 189227.625000|...|526381|125423|finished|...
3 |1.2ns  |lvt   |0.85v  | 833.333313| 0.0124| 159888.375000|...|519258|113595|finished|...
4 |1.2ns  |lvt   |0.95v  | 833.333313| 0.2436| 199254.593750|...|518152|110912|finished|...
5 |1.35ns |hvt   |0.85v  | 740.740723|-0.1852| 168545.546875|...|539921|151408|finished|...
6 |1.35ns |hvt   |0.95v  | 740.740723| 0.0025| 162913.671875|...|521649|117615|finished|...
7 |1.35ns |lvt   |0.85v  | 740.740723| 0.0234| 142857.578125|...|519240|113498|finished|...
8 |1.35ns |lvt   |0.95v  | 740.740723| 0.3936| 178100.734375|...|518152|110912|finished|...
9 |1.5ns  |hvt   |0.85v  | 666.666687|-0.0204| 149700.765625|...|538666|148396|finished|...
10|1.5ns  |hvt   |0.95v  | 666.666687| 0.0130| 142508.468750|...|520448|115258|finished|...
11|1.5ns  |lvt   |0.85v  | 666.666687| 0.1734| 129373.242188|...|519239|113497|finished|...
12|1.5ns  |lvt   |0.95v  | 666.666687| 0.5436| 161177.953125|...|518152|110912|finished|...
13|1.65ns |hvt   |0.85v  | 606.060608| 0.0003| 118437.906250|...|530751|134027|finished|...
14|1.65ns |hvt   |0.95v  | 606.060608| 0.0127| 129019.507812|...|519733|114339|finished|...
15|1.65ns |lvt   |0.85v  | 606.060608| 0.0039| 118195.390625|...|519109|113041|finished|...
16|1.65ns |lvt   |0.95v  | 606.060608| 0.6936| 147331.859375|...|518152|110912|finished|...
--+-------+------+-------+-----------+-------+--------------+...+------+------+--------+...
```

Example 2: Report with the -status Option

This example creates a report of the status of the DSE runs. In this case, all of the runs have finished. This command was applied from the same run directory, but in a different Oasys-RTL session.

```
[oasys-RTL]$ report_explore -status

Report exploration status:
--+------+-------+------------+---------------+--------+--------+-----+-----------------
  |lib_vt|voltage|clock_period| Job IDs       | Status |Hostname| PID | Log
--+------+-------+------------+---------------+--------+--------+-----+-----------------
1 |hvt   |0.85v  |0.9ns       |explore.0.0.0.1|finished|supra7  | 2944| ...
2 |hvt   |0.85v  |1.0ns       |explore.0.1.0.1|finished|supra7  | 3040| ...
3 |hvt   |0.85v  |1.1ns       |explore.0.2.0.1|finished|supra7  | 3046| ...
4 |hvt   |0.85v  |1.2ns       |explore.0.3.0.1|finished|supra7  | 3024| ...
5 |hvt   |0.95v  |0.9ns       |explore.0.0.0.0|finished|supra7  | 2865| ...
6 |hvt   |0.95v  |1.0ns       |explore.0.1.0.0|finished|supra7  | 2904| ...
7 |hvt   |0.95v  |1.1ns       |explore.0.2.0.0|finished|supra7  | 2908| ...
8 |hvt   |0.95v  |1.2ns       |explore.0.3.0.0|finished|supra7  | 2919| ...
9 |lvt   |0.85v  |0.9ns       |explore.0.0.1.1|finished|supra7  | 3034| ...
10|lvt   |0.85v  |1.0ns       |explore.0.1.1.1|finished|supra7  | 3086| ...
11|lvt   |0.85v  |1.1ns       |explore.0.2.1.1|finished|supra7  | 3106| ...
12|lvt   |0.85v  |1.2ns       |explore.0.3.1.1|finished|supra7  | 3101| ...
13|lvt   |0.95v  |0.9ns       |explore.0.0.1.0|finished|supra7  | 2936| ...
14|lvt   |0.95v  |1.0ns       |explore.0.1.1.0|finished|supra7  | 2987| ...
15|lvt   |0.95v  |1.1ns       |explore.0.2.1.0|finished|supra7  | 3039| ...
16|lvt   |0.95v  |1.2ns       |explore.0.3.1.0|finished|supra7  | 2999| ...
--+------+-------+------------+---------------+--------+--------+-----+-----------------
```

Example 3: Report with the “-format floorplan” Option

This example creates a report of a floorplan DSE run.

```
[oasys-RTL]$ report_explore -format floorplan
Report exploration metrics (summary):
--+-----+-------+-------+---------+------------+---+-----------+------+------+---------+---
  |FP   |Freq-  |WNS(ns)|TNS(ns)  |Wire_length |...|Total_power|Inst_ |Inst_ |Conges   |...
  |     |ency   |       |         |(mm)        |...|(uw)       |area  |count |tion_    |...
  |     |(mhz)  |       |         |            |...|           |(um2) |      |global   |...
--+-----+-------+-------+---------+------------+---+-----------+------+------+---------+---
1 |case1| 1250.0|-0.0200|-2.841699| 4899.028809|...| 515514.750|520007|115590| 0.348627|...
2 |case2| 1250.0|-0.0667|-1.447500| 4295.763672|...| 477887.218|520007|115590| 0.119150|...
3 |case3| 1250.0|-0.0200|-2.841699| 4899.028809|...| 515514.750|520007|115590| 0.348627|...
--+-----+-------+-------+---------+------------+---+-----------+------+------+---------+---
```

Related Topics

Report Commands

Design Space Exploration Commands

explore

set_explore

config_report

report_groups

Reports the groups used for placement.

Usage

report_groups [-group <group_name>] [-interface]

Arguments

- -group <group_name>

  Optional. Specifies a group name. The default is all groups.

- -interface

  Optional. Reports timing information at the boundary of the existing groups.

Description

This command generates a table with group information such as region, area, instance count, and edge assignment. You can create groups of macros or instances using the create_group command. This command can report one or all groups.

The -interface option generates a report with the following headings:

- **input/output WNS (ps) —** Worst negative slack of the group
- **input/output TNS (ps) —** Total negative slack of the group
- **input/output (#) —** Total number of input/output signals in the group
- **Critical input/output (#) —** Total number of input/output paths that do not meet the timing constraints

The edge assignment column is not included in the report with this option.

Examples

Example 1: Report Placement Groups Using Default Options

This example generates a report of existing groups.

```
[oasys-RTL]$ report_groups
Report Groups:
-----+-----+------+-----------+-------------+--------------
     |group|region|area (sq um)|instance count|edge assignment
-----+-----+------+-----------+-------------+--------------
1    |X1   |X1    |       93419|         28002|             2
2    |X2   |X2    |      165841|         26010|         3:1/3
-----+-----+------+-----------+-------------+--------------
```

Example 2: Report Timing at the Interface of the Placement Groups

This example reports the timing at the boundary of each placement group.

```
[oasys-RTL]$ report_groups -interface
Report Groups:
-----+-----+------+--------+--------+-------------+-------------+------------+-----------
     |group|region|area    |instance|input/output  |input/output  |input/output|critical
     |     |      |(sq um) |count   |WNS (ps)      |TNS (ps)      |(#)         |input/output
     |     |      |        |        |              |              |            |(#)
-----+-----+------+--------+--------+-------------+-------------+------------+-----------
1    |G1   |      |  105607|   27138|-67.00/-44.00 |339.90/427.50 |997/562     |13/13
2    |G2   |      |  164409|   23128|-48.10/-51.20 |200.50/239.70 |1111/980    |6/9
-----+-----+------+--------+--------+-------------+-------------+------------+-----------
```

For details on interpreting edge assignments, see “Identifying Edges and Edge Partitions” on page 23.

Related Topics

Report Commands

assign_macros_to_edges

create_group

remove_group

remove_from_group

report_hierarchy

Reports the area and cell count information about each hierarchy in the design. By default, reports the statistics of the top-level design in the session.

Usage

report_hierarchy [-leaf] [-all] [-dp] [-name <hierarchy_name>]

Arguments

- -leaf

  Reports the leaf cell statistics for the specified partition.

- -all

  Reports design information for all partitions. Each hierarchy is listed in a separate table.

- -dp

  Reports the datapath area for each partition. A Datapath Area column is added to the report table.

- -name <hierarchy_name>

  Specifies the name of a hierarchy to report. The default hierarchy is the top module.

Description

Following are the report_hierarchy table headings and their descriptions:

- Module - An individual user-defined block of design
- Count - Number of instantiations of the module in the design
- #Cells - Number of leaf cells instantiated in the module
- H#Cells - Total number of leaf cells instantiated in the hierarchical instances instantiated in the module
- Macro Area (sqμm) - Area of macro leaf cells in the module
- Seq Area (sqμm) - Area of sequential leaf cells in the module
- Comb Area (sqμm) - Area of combinational leaf cells in the module
- H. Area (sqμm) - Area of hierarchical cells in the module
- Total Area (sqμm) - Sum of the cell area of all the leaf macros, leaf combinational cells, leaf sequential cells, and hierarchical cells in the module.

Examples

Example

This example reports the top module.

```
%report_hierarchy
Report Hierarchy  demo_chip:
--+-----------------------+-----+------+-------+------+------+------+-------+-------
  |Module                 |Count|#Cells|H#Cells|Macro |Seq   |Comb  |H.     |Total
  |                       |     |      |       |Area  |Area  |Area  |Area   |Area
  |                       |     |      |       |(squm)|(squm)|(squm)|(squm) |(squm)
--+-----------------------+-----+------+-------+------+------+------+-------+-------
1 |cpu_sys                |    1|     0|     22|     0|     0|     0| 185011| 185011
2 |nova_wrapper__1        |    1|     1| 147335|     0|     7|     0| 302860| 302867
3 |hpdmc                  |    1|     8|    171|     0|     0|     4|   6512|   6516
4 |hpdmc__parameterized0__1|    1|     8|    171|     0|     0|     4|   6512|   6516
5 |nova_wrapper           |    1|     1| 147182|     0|     7|     0| 302761| 302768
6 |hpdmc__parameterized1  |    1|     8|    171|     0|     0|     4|   6512|   6516
7 |hpdmc__parameterized0  |    1|     8|    171|     0|     0|     4|   6512|   6516
8 |usb_sys                |    1|     0|   2928|     0|     0|     0| 377978| 377978
9 |usb_phy                |    1|     0|    299|     0|     0|     0|    800|    800
10|demo_chip              |    1|   775| 298484| 72044|   616|   413|1195489|1268562
--+-----------------------+-----+------+-------+------+------+------+-------+-------
```

Example

This example reports the information about a sub-partition (nova_wrapper) .

```
%report_hierarchy -name nova_wrapper
Report Hierarchy  nova_wrapper:
--+------------+-----+------+-------+------+------+------+--------+------
  |Module      |Count|#Cells|H#Cells|Macro |Seq   |Comb  |H.      |Total
  |            |     |      |       |Area  |Area  |Area  |Area    |Area
  |            |     |      |       |(squm)|(squm)|(squm)|(squm   |(squm)
--+------------+-----+------+-------+------+------+------+--------+------
1 |nova        |    1|     0| 147182|     0|     0|     0|  302761|302761
2 |nova_wrapper|    1|     1| 147182|     0|     7|     0|  302761|302768
--+------------+-----+------+-------+------+------+------+--------+------
```

Example

This example reports the information about same sub-partition including datapath.

```
%report_hierarchy -name nova_wrapper
Report Hierarchy  nova_wrapper:
--+------------+-----+------+-------+------+------+------+--------+--------+-----
  |Module      |Count|#Cells|H#Cells|Macro |Seq   |Comb  |Datapath|H.      |Total
  |            |     |      |       |Area  |Area  |Area  |Area    |Area    |Area
  |            |     |      |       |(squm)|(squm)|(squm)|(squm)  |(squm)  |(squm)
--+------------+-----+------+-------+------+------+------+--------+--------+------
1 |nova        |    1|     0| 147182|     0|     0|     0|  25049|   302761|302761
2 |nova_wrapper|    1|     1| 147182|     0|     7|     0|  25049|   302761|302768
--+------------+-----+------+-------+------+------+------+--------+--------+------
```

Example

This example reports the information about same sub-partition including leaf cells.

```
% report_hierarchy -name nova_wrapper -leaf
Report Hierarchy  nova_wrapper:
--+-------------+-----+------+-------+------+------+------+-------+-------
  |Module       |Count|#Cells|H#Cells|Macro |Seq   |Comb  |H.Area |Total
  |             |     |      |       |Area  |Area  |Area  |Area   |Area
  |             |     |      |       |(squm)|(squm)|(squm)|(squm) |(squm)
--+-------------+-----+------+-------+------+------+------+-------+-------
1 |nova         |    1|     0| 147182|     0|     0|     0| 302761| 302761
2 |CLKGATETST_X1| 1278|     0|      0|     0|     0|  5099|      0|   5099
3 |SDFF_X1_LVT  |14748|     0|      0|     0| 90228|     0|      0|  90228
4 |INV_X1       |23060|     0|      0|     0|     0| 12268|      0|  12268
5 |NOR4_X1      |  845|     0|      0|     0|     0|  1124|      0|   1124
6 |OR2_X1       |  670|     0|      0|     0|     0|   713|      0|    713\
7 |NOR3_X1      |  633|     0|      0|     0|     0|   674|      0|    674
8 |NAND2_X1     |36382|     0|      0|     0|     0| 29033|      0|  29033
9 |AOI222_X1    |  464|     0|      0|     0|     0|   987|      0|    987
10|AOI22_X1     |15869|     0|      0|     0|     0| 21106|      0|  21106
11|NAND4_X1     | 3918|     0|      0|     0|     0|  5211|      0|   5211
12|NOR2_X1      | 4594|     0|      0|     0|     0|  3666|      0|   3666
13|SDFF_X2_HVT  | 9849|     0|      0|     0| 62876|     0|      0|  62876
14|OAI21_X1     | 4876|     0|      0|     0|     0|  5188|      0|   5188
15|NAND3_X1     | 7381|     0|      0|     0|     0|  7853|      0|   7853
16|XOR2_X1      |  626|     0|      0|     0|     0|   999|      0|    999
17|XNOR2_X1     | 3507|     0|      0|     0|     0|  5597|      0|   5597
18|OAI22_X1     | 2436|     0|      0|     0|     0|  3240|      0|   3240
19|OAI211_X1    | 1792|     0|      0|     0|     0|  2383|      0|   2383
20|OR3_X1       |  118|     0|      0|     0|     0|   157|      0|    157
21|AOI21_X1     | 1796|     0|      0|     0|     0|  1911|      0|   1911
22|OAI221_X1    |  696|     0|      0|     0|     0|  1111|      0|   1111
23|AOI211_X1    |  371|     0|      0|     0|     0|   493|      0|    493
24|MUX2_X1      |  710|     0|      0|     0|     0|  1322|      0|   1322
25|AND2_X1      | 2746|     0|      0|     0|     0|  2922|      0|   2922
26|AND3_X1      |  136|     0|      0|     0|     0|   181|      0|    181
27|AOI221_X1    | 1082|     0|      0|     0|     0|  1727|      0|   1727
28|OR4_X1       |   62|     0|      0|     0|     0|    99|      0|     99
29|OAI33_X1     |  319|     0|      0|     0|     0|   594|      0|    594
30|AND4_X1      |   34|     0|      0|     0|     0|    54|      0|     54
31|OAI222_X1    |  105|     0|      0|     0|     0|   223|      0|    223
32|HA_X1        |  362|     0|      0|     0|     0|   963|      0|    963
33|FA_X1        |  531|     0|      0|     0|     0|  2260|      0|   2260
34|BUF_X1_LVT   |  618|     0|      0|     0|     0|   493|      0|    493
35|SDFF_X1_HVT  |   43|     0|      0|     0|   263|     0|      0|    263
36|BUF_X1       |   40|     0|      0|     0|     0|    32|      0|     32
37|SDFFR_X1_LVT | 4310|     0|      0|     0| 28662|     0|      0|  28662
38|DLH_X1       |   37|     0|      0|     0|    98|     0|      0|     98
39|SDFFR_X2_LVT |   21|     0|      0|     0|   145|     0|      0|    145
40|SDFFR_X2_HVT |   98|     0|      0|     0|   678|     0|      0|    678
41|SDFFS_X1_LVT |   18|     0|      0|     0|   120|     0|      0|    120
42|SDFFS_X2_LVT |    2|     0|      0|     0|    14|     0|      0|     14
43|nova_wrapper |    1|     1| 147182|     0|     7|     0| 302761| 302768
--+-------------+-----+------+-------+------+------+------+-------+-------
```

Related Topics

report_area

report_instances

report_instances

Reports the immediate children belonging to a specified hierarchical instance.

Usage

report_instances [<instance>]

Arguments

- <instance>

  Specifies the instance name. If you do not specify this option, the default is the top of the design.

Examples

Example

```
% report_instances

Report Instances:
--+---------+-------+------------+----------------
  |instance |module |area (sq um)|leakage power (nW)
--+---------+-------+------------+----------------
1 |i_6      |[none] | 0          | 0
2 |ccx      |ccx    | 546964     | 916429
3 |ctu      |ctu    | 98101      | 185804
4 |dram02   |dram__1| 421487     | 766184
5 |dram13   |dram   | 421487     | 766184
6 |efc      |efc    | 12922      | 18552
7 |fpu      |fpu    | 203170     | 434449
…
```

Example

```
% report_instances sparc0/*

Report Instances:
--+---------+------------------+------------+----------------
  |instance |module            |area (sq um)|leakage power (nW)
--+---------+------------------+------------+----------------
1 |exu      |sparc_exu__key0__1 | 51990      | 5870114
2 |ff_cpx   |cpx_spc_rpt__key0  | 694        | 6104
3 |ffu      |sparc_ffu__key0__1 | 46779      | 7039480
4 |ifu      |sparc_ifu__key0__1 | 100240     | 14049400
5 |
…
```

Related Topics

Report Commands

report_layer_rc

Reports the unit resistance and capacitance values of each layer. These values are loaded from the captable file or PTF file.

Usage

report_layer_rc

Arguments

None.

Examples

This example reports the unit resistance and capacitance values that were previously loaded:

```
% report_layer_rc
Report Layers RC:
--+------+------+-------+-------+------+----------+---------+-----+-------------
  |Layer |Direc-|Width  |Spacing|ohm/sq|ohm/um    |cap ff/um|ecap |cap/Å
  |Name  |tion  |       |       |      |          |         |ff/um|
--+------+------+-------+-------+------+----------+---------+-----+-------------
1 |M1    |V     |0.07000|0.14000|     0| 4.3228574| 0.133407|    0|1.3340699e-05
```

Related Topics

Report Commands

report_leakage

Reports the leakage power of the design in session and the percentage of cells in each threshold voltage group.

Usage

report_leakage [-detail] [-instance <instance>] [-module <module>]

Arguments

- -detail

  Specifies leakage information on each of the cell types.

- -instance <instance>

  Reports leakage for the specified instance. The default is the whole design.

- -module <module>

  Reports leakage for the specified module.

Examples

Example

```
% report_leakage

Report Leakage Efficiency:
-----------------------+---------+----------+-------+------+-------+-------+-------+------
                       |# cells  |Leakage   |hvt    |hvt   |mvt    |mvt    |lvt    | lvt
                       |         |(uW)      |%#cells|%     |%#cells|%      |%#cells| %
-----------------------+---------+----------+-------+------+-------+-------+-------+------
Design Name            |OpenSPARC|          |       |      |       |       |       |
  Macros               |421      |13019.464 |100.000|100.00|       |       |       |
  Pads                 |261      |3300.849  |100.000|100.00|       |       |       |
  Cells                |1694593  |23999.955 |0.000  |0.000 |87.775 |81.466 |12.225|18.534
    Buffers/Inverters  |457891   |5671.185  |0.000  |0.000 |100.000|100.000|0.000 |0.000
    Combinational      |935544   |14500.631 |0.000  |0.000 |77.857 |69.324 |22.143|30.676
    Latches            |2075     |75.408    |0.000  |0.000 |100.000|100.000|0.000 |0.000
    Registers          |299083   |3752.730  |0.000  |0.000 |100.000|100.000|0.000 |0.000
-----------------------+---------+----------+-------+------+-------+-------+------+------
```

Related Topics

Report Commands

report_leakage_setup

Reports the threshold voltage group characteristics.

Usage

report_leakage_setup

Arguments

None

Description

Generates a report of threshold voltage group (Vth group) characteristics. The Vth group can be from a library or user-created. The “Num cells” column is the number of library cells that belong to the Vth group. A library cell can belong to only one Vth group. The “Num cell families” column is the number of unique cell classes such as AND, OR, and MUX. All cells in a cell family have the same logical function.

Examples

```
% report_leakage_setup
Report Leakage Setup:
-----+---------+---------+-----------------+----------------------
     |Vth group|Num cells|Num cell families|Avg leakage/area (pw/squm)
-----+---------+---------+-----------------+----------------------
1    |HVT      |      134|               45|                7.000000
2    |SVT      |      134|               45|               16.000000
3    |LVT      |      134|               45|             5864.000000
-----+---------+---------+-----------------+----------------------
```

Related Topics

Report Commands

report_lib_cell_delay

Reports the delay for the specified cell.

Usage

report_lib_cell_delay **-lib_cell *cell_name* -from *pin_name* -to *pin_name*** [-inSlew *value*] [-outSlew *value*] [-load *value*] [{-min | -max}] [-derate]

Arguments

- **-lib_cell *cell_name***

  Specifies the library cell for which to report delay.

- **-from *pin_name***

  Specifies the input pin selected for the delay calculation.

- **-to *pin_name***

  Specifies the output pin selected for the delay calculation.

- -inSlew *value*

  This optional argument specifies the slew on the input pin. If you do not specify this argument, the worst slew is automatically selected.

- -outSlew *value*

  This optional argument specifies the slew on the output pin. If you do not specify this argument, the worst slew is automatically selected.

- -load *value*

  This optional argument specifies the load value for the output pin. The units are determined by the technology library. If you do not specify this argument, the maximum value specified in the technology library is automatically selected.

- {-min | -max}

  This optional argument specifies whether to report the minimum or maximum delay. The default is maximum.

- -derate

  Adjusts the reported delay times based on the cell's derate factor. If you do not specify this argument, the reported delay times are not derated.

  Note

  Set derating factors with the set_timing_derate command.

Examples

Example 1

The following example reports the delay between input pin A and output pin Y on the NAND2X1 library cell.

```
[oasys-RTL]$ report_lib_cell_delay -lib_cell NAND2X1 -from A -to Y -
outSlew 0.54 -load 0.78 -derate
Warning: OutSlew will be discarded since calculating delay for non setup
arc
Info: Required Value of InSlew not provided, will be picked from library
Report Library Cell Delay For Cell "NAND2X1":
--------------------------+--------------------
Attribute                 |Value
--------------------------+--------------------
From Pin                  |A
To Pin                    |Y
Cell Library              |typical
Arc Type                  |comb
Arc Sense                 |negative unate
Time/Cap Unit             |(ps)/(ff)
Delay (Rise/Fall)         |-{3586.2ps,4345.0ps}
Load Capacitance          |             780.000
Output Pin Transition     |Fall
Input Pin Transition      |Fall
Output Pin Transition Time|<max>
Input Pin Transition Time |{660.0ps,660.0ps}
Derating Factor           |               1.000
--------------------------+--------------------
```

Example 2

The following example repeats Example 1, but modifies the outslew to 0.54, specifies a load, and enables derating.

```
[oasys-RTL]$ report_lib_cell_delay -lib_cell NAND2X1 -from A -to Y
-outSlew 0.54 -load 0.78 -derate

Warning: OutSlew will be discarded since calculating delay for non setup
arc
Info: Required Value of InSlew not provided, will be picked from library
Report Library Cell Delay For Cell "NAND2X1":
--------------------------+--------------------
Attribute                 |Value
--------------------------+--------------------
From Pin                  |A
To Pin                    |Y
Cell Library              |typical
Arc Type                  |comb
Arc Sense                 |negative unate
Time/Cap Unit             |(ps)/(ff)
Delay (Rise/Fall)         |-{3586.2ps,4345.0ps}
Load Capacitance          |             780.000
Output Pin Transition     |Fall
Input Pin Transition      |Fall
Output Pin Transition Time|<max>
Input Pin Transition Time |{660.0ps,660.0ps}
Derating Factor           |               1.000
--------------------------+--------------------
```

Related Topics

Report Commands

report_library_cells

Reports what library cells are in the loaded library.

Usage

report_library_cells { -isolation | -level_shifter | -multibit | -pin_count <count> | -retention | -standard | -target_library <library> } ... [<cell_name>]

Arguments

- { -isolation | -level_shifter | -multibit | -pin_count <count> | -retention | -standard | -target_library <library>} ...

  One or more of the following options are required:

  -isolation - List all isolation cells that are in the design.

  -level_shifter - List all level shifters that are in the design.

  -multibit - List all multi-bit cells in the library.

  -pin_count <count> - List all cells of the specified types that have a pin count of <count>.

  -retention - List all retention cells that are in the design.

  -standard - List all standard cells that are in the target library.

  -target_library <library> - Specifies a target library.

- <cell_name>

  Optional. Specifies the name of the library cell. Glob-style pattern matching is performed on the instance or module names.

Description

The report_library_cells command does not have any default options.You must specify at least one option, or else the command reports nothing. The -pin_count argument must be used with another argument that lists cells (-isolation, -level_shifter, -multibit, -retention, -standard) or with the <cell_name> argument.

Examples

Example 1: List All Cells Named “iso*”

This example reports a list of all cells that start with iso in all target libraries:

```
% report_library_cells iso* -target_library ALL
```

Example 2: List All Standard Cells

This example reports all standard library cells:

```
% report_library_cells -standard true
Report Standard Cells:
---+-------+-------+--------------------------------+------+--------+--------+-----+-----
   |       |       |                                |Area  |Leakage |Function|Power|
   |Cell   |Family |Library                         |(squm)|(pW)    |Type    |Type |Rails
---+-------+-------+--------------------------------+------+--------+--------+-----+-----
1  |SDFF_X1|SDFF_X1|NangateOpenCellLib_PDKv1_10_typi |6.1   |15376.25|ff      | -   | -
2  |SDFF_X1|SDFF_X1|NangateOpenCellLib_PDKv1_10_fast |6.1   |51124.80|ff      | -   | -
3  |SDFF_X1|SDFF_X1|NangateOpenCellLib_PDKv1_10_slow |6.1   |10275.64|ff      | -   | -
4  |SDFF_X2|SDFF_X1|NangateOpenCellLib_PDKv1__10_typi|6.1   |16966.44|ff      | -   | -
. . .
. . .
. . .
12 |AND2_X1|AND2_X2|NangateOpenCellLib_PDKv1_10_fast |1.1   |28205.18|comb    | -   | -
13 |AND2_X1|AND2_X2|NangateOpenCellLib_PDKv1_10_slow |1.1   | 5107.49|comb    | -   | -
14 |AND2_X4|AND2_X2|NangateOpenCellLib_PDKv1_10_typi |1.1   |19176.88|comb    | -   | -
15 |AND2_X4|AND2_X2|NangateOpenCellLib_PDKv1_10_fast |1.1   |68260.13|comb    | -   | -
```

Example 3: List All Isolation Cells

This example reports all isolation cells in all target libraries.

```
% report_library_cells * -target_library ALL -isolation true
```

Example 4: List Standard Cells With a Specified Pin Count

This example reports all standard library cells with a pin count of 4.

```
% report_library_cells -pin_count 4 -standard
```

Related Topics

Report Commands

report_logic_depth

Reports a text-based histogram with bins based on the number of logic levels in a timing path.

Usage

report_logic_depth [-clock <string>] [-group <string>]

Arguments

- -clock <string>

  Filters the histogram report to only consider endpoints driven by the specified clock. If the -clock or -group options are not specified, then Oasys RTL considers all constrained endpoints. The report_clocks command generates a list of clock names in the design.

- -group <string>

  Filters the histogram to only consider endpoints contained within the specified path group. If the -clock or -group option are not specified, then Oasys RTL considers all valid endpoints. The report_path_groups command generates a list of path group names in the design.

Description

The report_logic_depth command creates a text-based histogram table with bins based on the number of levels of logic. Each bin contains the number of constrained, timing endpoints with the defined levels of logic. By default, Oasys RTL generates a text-based histogram for all valid, constrainted endpoints in the design. You can filter these results based on the clock or path group.

The logic depth histogram does not include the following objects in the report:

- Modules specified using the set_driving_cell command
- I/O Delays
- Clock latency
- Instances with timing arcs through registers or clock paths
- Buffers and Inverters

Note

Endpoints with false path constraints will not be reported.

Examples

The following example creates a path group called ‘I2R’ that contains timing paths that begin at the primary inputs of the design. The report filters the logic depth histogram to only consider the endpoints in this *I2R* path group.

```
[oasys-RTL]$ group_path -name I2R -from [all_inputs]
[oasys-RTL]$ report_path_groups
Report Path Groups:
-----+-------+------+---------+---------
     | Path  |Weight|Critical |Worst
     | Group |      |Range(ps)|Slack(ps)
-----+-------+------+---------+---------
1    |default| 1.000|      0.0|     -6.6
2    |I2R    | 1.000|      0.0|    123.3
3    |I2O    | 1.000|      0.0|<ill>
4    |R2O    | 1.000|      0.0|    115.2
-----+-------+------+---------+---------
[oasys-RTL]$ report_logic_depth -group I2R
info:    Target library/cell information has changed that further may
change timing results.  [TA-159]
Report Logic Depth:
-----+--------+------
Depth|EP Count| %
-----+--------+------
    1|     258|50.097
    2|     257|49.903
-----+--------+------
```

The following example shows how to filter the histogram of logic depth based on the endpoints driven by the ‘sysclk’ clock. For brevity, the report shown in this example was truncated at 20 logic levels:

```
[oasys-RTL]$ report_clocks
Report Clocks:
-----+-----------+---------+--------+----------+-------+--------
     |Clock      |  Worst  |Period  |Transition|Latency|Master
     |           |Slack(ps)| (ps)   |   (ps)   |  (ps) |
-----+-----------+---------+--------+----------+-------+--------
1    |lfxt_clk   |<unc>    |100000.0|       0.0|    0.0|
2    |sysclk     |     -6.6|  2500.0|       0.0|    0.0|lfxt_clk
3    |usbclk     |  15287.1| 16666.7|       0.0|    0.0|lfxt_clk
4    |sysclk_byp |<unc>    |100000.0|       0.0|    0.0|
5    |usbclk_byp |<unc>    |100000.0|       0.0|    0.0|
6    |vsysclk    |    454.6|  2500.0|       0.0|    0.0|
7    |vsysclk_ddr|<unc>    |  1250.0|       0.0|    0.0|
-----+-----------+---------+--------+----------+-------+--------

[oasys-RTL]$ report_logic_depth -clock sysclk
info:    Target library/cell information has changed that further may
change timing results.  [TA-159]
Report Logic Depth:
-----+--------+------
Depth|EP Count| %
-----+--------+------
    0|    1350| 2.208
    1|     518| 0.847
    2|     397| 0.649
    3|     208| 0.340
    4|     146| 0.239
    5|     413| 0.675
    6|   23239|38.005
    7|     714| 1.168
    8|    4126| 6.748
    9|     915| 1.496
   10|     635| 1.038
   11|    1030| 1.684
   12|     128| 0.209
   13|      36| 0.059
   14|     104| 0.170
   15|     170| 0.278
   16|     168| 0.275
   17|     268| 0.438
   18|     380| 0.621
   19|     122| 0.200
   20|      62| 0.101
```

Related Topics

Report Commands

report_multibit

Reports a summary of the results of multi-bit mapping. The report includes statistics for the flip-flops that the tool packed successfully as well as the flip-flops that the tool deemed ineligible for multi-bit mapping.

Usage

report_multibit [-instance <instance_list>]

Arguments

- -instance <instance_list>

  Specifies multi-bit instances to include in the report.

Description

Reports multi-bit packing statistics. Packing statistics are available after you run the map_to_multibit command. The report includes statistics for eligible and ineligible flip-flops.

The following example is truncated version of the Multi-Bit Report. For brevity, this example only shows about half of the hierarchical blocks of the design. As such, the column values of the 'TOP' level row are greater than the sum of the values in the remaining columns.

	Instance	Module	1-bit FF	2-bit FF	4-bit FF	Total FF	Total Bits	% Multibit	Bit Ratio
1	TOP		338	112	9096	9546	36946	99.1	3.87
2	u_f	md_fetch	51	49	5060	5160	20389	99.7	3.95
3	u_bcc	md_bcc	0	1	525	526	2102	100.0	4.00
4	pht	md_bcc_ram2p	0	0	516	516	2064	100.0	4.00
5	p_fifo	md_fifo__p0	0	0	53	53	212	100.0	4.00
6	buf_fifo	md_fifo__p1	0	0	161	161	644	100.0	4.00
7	i_fifo	md_ififo	33	34	887	954	3649	99.1	3.82
8	ififo_cpx	md_ififo_cpx	6	1	221	228	892	99.3	3.91
9	ififo_wdatax	md_ififo_wdatax	0	0	0	0	0	0.0	0.00
10	ififo_insnx	md_ififo_insnx	0	0	0	0	0	0.0	0.00
11	u_btb	md_btb	1	12	3256	3269	13049	100.0	3.99
12	btb0	md_btb2w__01	0	3	812	815	3254	100.0	3.99
13	btbw1_ram2p	md_btb_ram2p__02	0	1	386	387	1546	100.0	3.99
14	btbw0_ram2p	md_btb_ram2p__03	0	1	386	387	1546	100.0	3.99
15	btb1	md_btb2w__04	0	3	812	815	3254	100.0	3.99
16	btbw1_ram2p	md_btb_ram2p__05	0	1	386	387	1546	100.0	3.99
17	btbw0_ram2p	md_btb_ram2p__06	0	1	386	387	1546	100.0	3.99
18	btb2	md_btb2w__07	0	3	812	815	3254	100.0	3.99
19	btbw1_ram2p	md_btb_ram2p__08	0	1	386	387	1546	100.0	3.99
20	btbw0_ram2p	md_btb_ram2p__09	0	1	386	387	1546	100.0	3.99
21	btb3	md_btb2w	0	3	812	815	3254	100.0	3.99
22	btbw1_ram2p	md_btb_ram2p__10	0	1	386	387	1546	100.0	3.99
23	btbw0_ram2p	md_btb_ram2p	0	1	386	387	1546	100.0	3.99
24	ad_fifo	md_fifo	0	0	35	35	140	100.0	4.00
25	u_ras	md_ras	0	1	129	130	518	100.0	3.98
26	u_rn	md_rename	165	0	305	470	1385	88.1	2.95
27	u_fifox	md_rename_fifox	4	0	280	284	1124	99.6	3.96
28	u_decode0	md_rn_decode__11	0	0	0	0	0	0.0	0.00
29	u_decode1	md_rn_decode__12	0	0	0	0	0	0.0	0.00
30	u_decode2	md_rn_decode	0	0	0	0	0	0.0	0.00

The following list describes the column heading in the report:

- **Instance —** This value is the instance name of the hierarchical block. The indentation implies the hierarchical nesting of the blocks.
- **Module —** This value denotes the corresponding module name of the instance.
- **N-bit FFs —** This value is the summation of the flop count with exactly N bits contained within the hierarchical blocks of this instance.
- **Total FFs —** This value is the summation of the total number of flops (single and multibit) contained within the hierarchical blocks of this instance.
- **Total Bits —** This value is the number of bits represented by the "Total FF" count.
- **% Multibit —** This values is the percentage of the flops that have more than 1 bit compared to the total number of flops.
- **Bit Ratio —** This value shows the average number of bits that each flop has.

report_mv

Reports the details of instances of multi-voltage cells.

Usage

report_mv [-level_shifter] [-isolation] [-domain <string>]

Arguments

- -level_shifter

  Specifies that the report be limited to level-shifter associations.

- -isolation

  Specifies that the report be limited to isolation cell associations.

- -domain <string>

  Specifies the power domain for reporting the multi-voltage cells.

Description

This command generates a report of multi-voltage cells (such as level-shifter and isolation cells) that are in the design. Included in the report are the power domain associations specified in the UPF rules. By default all multi-voltage instances are reported. Options can be used to limit the scope of reporting.

Examples

```
% report_mv -domain PD_TOP -isolation
Report Multi-Voltage Special Cells:
-----------+---------+---------+------+--------+---------+----------------+--------------
|instance  |domain   |cell     |driver|load    |input    |output          |control
|          |         |         |domain|domains |power    |power           |signal
|LS_NOVA1LS|PD_NOVA1 |LS_HL_X1 |PD_TOP|PD_NOVA1|VDD(VDD) |VDDNOVA1SW(VDDL)|iso_con[1]
```

Related Topics

Report Commands

report_name_rules

Reports the properties defined for a set of name rules. Name rules are created or modified by the define_name_rules command.

Usage

report_name_rules [-name_rules <string>]

Arguments

- -name_rules <string>

  Optional. Specifies the name of the rules to be reported. If the name_rules option is not specified, all currently defined name rules will be reported.

Examples

Example 1: Report Without Any Options

This example creates a report of all the currently defined name rules.

```
[oasys-RTL:/]$ define_name_rules my_rules1 \
   -flatten_multi_dimension_busses -type port
[oasys-RTL:/]$ change_names -rule my_rules1 -hier

info:    Applying name rules: my_rules1  [NL-178]
info:    Names of 132 object(s) changed  [NL-179]
132

[oasys-RTL:/]$ define_name_rules my_rules2 -map {{"pack", "GROUP"}}
[oasys-RTL:/]$ change_names -rule my_rules2 -hier

info:    Applying name rules: my_rules2  [NL-178]
info:    Names of 236 object(s) changed  [NL-179]
236

[oasys-RTL:/]$ define_name_rules my_rules3 -first_restricted "A-Z" \
   -replacement_char "@"
[oasys-RTL:/]$ change_names -rule my_rules3 -hier

info:    Applying name rules: my_rules3  [NL-178]
info:    Names of 44 object(s) changed  [NL-179]
44

[oasys-RTL:/]$ report_name_rules
```

Figure 5-4. Output of report_name_rules Command With No Options

```
[oasys-RTL:/]$ report_name_rules

Report Name Rules:
---------+------------------------------+-----------------+-----------------+-----------------
Name     |Rule                          |Port             |Cell             |Net
---------+------------------------------+-----------------+-----------------+-----------------
my_rules1|                              |                 |                 |
         |-flatten_multi_dimension_busses|true             |                 |
my_rules2|                              |                 |                 |
         |-map                          |{"pack", "GROUP"} |{"pack", "GROUP"} |{"pack", "GROUP"}
my_rules3|                              |                 |                 |
         |-replacement_char             |@                |@                |@
         |-first_restricted             |A-Z              |A-Z              |A-Z
---------+------------------------------+-----------------+-----------------+-----------------
```

Example 2: Report With the -name_rules Option

This example creates a report for the specified rule.

```
[oasys-RTL:/]$ report_name_rules -name_rules my_rules3

Report Name Rules:
---------+-----------------+----+----+----
Name     |Rule             |Port|Cell|Net
---------+-----------------+----+----+----
my_rules3|                 |    |    |
         |-replacement_char|@   |@   |@
         |-first_restricted|A-Z |A-Z |A-Z
---------+-----------------+----+----+----
```

Related Topics

Report Commands

change_names

define_name_rules

report_net

Reports net information in the current design.

Usage

report_net <pin> [-physical]

Arguments

- <pin>

  Required. Specifies a pin to which the net is connected.

- -physical

  Optional. Reports the connectivity between RTL partitions only.

Description

By default, the report_net command reports all logically connected cells and any top level ports. For each connected pin, it reports the following:

- **pin —** the name of the pin
- **dir —** the direction of the pin
- **t —** the type, which is one of (P)ort, (R)TL partition, (M)acro, or (C)ell
- **leaf —** the leaf name of the RTL partition, port, macro, or cell
- **RTL partition —** the RTL partition to which the pin belongs
- **xLoc, yLoc —** the x and y coordinates of the location
- **pinCap —** the pin capacitance in femtofarads
- **slack —** the slack of the connected leaf in picoseconds
- **area —** the area of the connected leaf in square microns

The -physical option reports all connections on the physical net such as all connected RTL partitions and any connected top-level instances and ports. For each connected pin, it reports the default columns without “RTL partition” and with following additional columns:

- **pc —** the number of internally connected pins if it is an RTL partition (1 for others)
- **wLen —** estimated wire length in microns
- **netRes —** estimated resistance to the pin
- **netDly —** estimated net delay to the pin in picoseconds

Examples

Example 1: Report Information About Pins Connected to a Specified Net

This example reports information about the pins connected to the *exu/rt_shieldBuf/A* pin.

```
[oasys-RTL]$ report_net exu/rt_shieldBuf/A
Report Net: netCap = 9.85ff:
-+---------------+---+-+------------+---------------+----+----+------+-----------+----
 |pin            |dir|t|leaf        |RTL partition  |xLoc|yLoc|pinCap|slack      |area
-+---------------+---+-+------------+---------------+----+----+------+-----------+----
1|q_reg[5]/Q     |out|C|SDFF_X1_HVT |dff_param19__83| 617| 544|  0.00|      -39.1| 6.1
2|q_reg[5]/SI    |in |C|SDFF_X1_HVT |dff_param19__83| 617| 544|  0.71|     2147.0| 6.1
3|i_0_5/A        |in |C|MUX2_X1_HVT |sparc_exu_GC0  | 603| 543|  0.97|      546.2| 1.9
4|i_0_0_5/A      |in |C|INV_X1_HVT  |sparc_exu_GC0  | 603| 543| 17.83|      -34.0| 0.5
5|A              |in |C|BUF_X1_HVT  |               | 623| 554|  0.69|      101.2| 0.8
6|i_1_6/A2       |in |C|NAND2_X1_HVT|sparc_exu__GCB0| 559| 580|  2.85|      -39.1| 0.8
-+---------------+---+-+------------+---------------+----+----+------+-----------+----
```

Example 2: Report Physical Information About Pins Connected to a Net

This example reports physical information about the net connected to the *exu/rt_shieldBuf/A* pin.

```
[oasys-RTL]$ report_net -physical exu/rt_shieldBuf/A
Report Net: netCap = 9.85ff:
-+---------------+---+-+--+------+----+----+----+------+------+------+-----------+----
 |pin            |dir|t|pc|leaf  |xLoc|yLoc|wLen|netRes|netDly|pinCap|slack      |area
-+---------------+---+-+--+------+----+----+----+------+------+------+-----------+----
1|q_reg[5]/Q     |out|C| 1|SDF...| 617| 544|    |      |      |  0.00|      -39.1| 6.1
2|q_reg[5]/SI    |in |C| 1|SDF...| 617| 544|   0|   0.0|   0.0|  0.71|     2147.0| 6.1
3|i_0_5/A        |in |C| 1|MUX...| 603| 543|  15|   0.0|   0.0|  0.97|      546.2| 1.9
4|i_0_0_5/A      |in |C| 1|INV...| 603| 543|  15|   0.0|   0.0| 17.83|      -34.0| 0.5
5|A              |in |C| 1|BUF...| 623| 554|  16|  71.8|   0.0|  0.69|      101.2| 0.8
6|i_1_6/A2       |in |C| 1|NAN...| 559| 580|  94|   0.0|   0.0|  2.85|      -39.1| 0.8
-+---------------+---+-+--+------+----+----+----+------+------+------+-----------+----
```

Related Topics

Report Commands

report_operating_conditions

Reports the information for all operating conditions that exist in the database.

Usage

report_operating_conditions

Arguments

None.

Examples

```
% report_operating_conditions

Report Operating conditions:
-+------+--------+-------------+---------+---------+--------+------------
 | Name |Default?|   Type      | Library | Process |Voltage | Temperature
-+------+--------+-------------+---------+---------+--------+------------
1| WCCOM|  yes   |standard cell| my64.lib| 1.00000 |0.990000|125.0000000
-+------+--------+-------------+---------+---------+--------+------------
```

Related Topics

Report Commands

create_operating_condition

report_parameters

Reports the values and types of the Oasys-RTL parameters.

Usage

report_parameters [-group <group_name>] [-modified]

Arguments

- -group <group_name>

  Optional. Shows only those parameters that are in the specified group. Valid groups are: dft, main, messages, naming, optimize, physical, power, retime, syn, timer, and write.

- -modified

  Optional. Shows only those parameters that were modified from their default values.

Description

Reports the parameter values that can be used during the run. The read/write parameters can be modified with the set_parameter command. See the parameters described in “Oasys-RTL Parameters” on page 647.

Examples

Example 1: Report with Default Options

```
[oasys-RTL]$ report_parameters

Report Parameters:
-+-----+------------------------------+-------+-------+-------+----------
 |     |                              |Current|Default|       |
 |Group|Parameter                     |Value  |Value  |Type   |Read/Write
-+-----+------------------------------+-------+-------+-------+----------
1|dft  |auto_define_derived_test_clocks|true   |false  |boolean|read/write
2|dft  |dft_rising_edge_head_lockup    |false  |false  |boolean|read/write
3|dft  |dft_rising_edge_tail_lockup    |false  |false  |boolean|read/write
4|dft  |scan_lockup_instance_prefix    |lockup |lockup |string |read/write
…
```

Example 2: Report with the -modified Option

```
[oasys-RTL]$ report_parameter -modified
---+-------+------------------+--------+-------+-------+----------
                              |Current |Default|       |
   |Group  |Parameter         |Value   |Value  |Type   |Read/Write
---+-------+------------------+--------+-------+-------+----------
1  |naming |array_naming_style|[%d]    |[%d]   |format |read/write
```

Example 3: Report with the -group Option

```
[oasys-RTL]$ report_parameter -group naming

Report Parameters:
-+------+-------------------------+-------+-------+-------+----------
 |      |                         |Current|Default|       |
 |Group |Parameter                |Value  |Value  |Type   |Read/Write
-+------+-------------------------+-------+-------+-------+----------
1|naming|always_naming_style      |       |       |format |read/write
2|naming|architecture_naming_style|%s_%s  |%s_%s  |format |read/write
3|naming|array_naming_style       |[%d]   |[%d]   |format |read/write
4|naming|block_naming_style       |%s_    |       |format |read/write
```

Related Topics

Report Commands

Oasys-RTL Parameters

get_parameter

reset_parameter

set_parameter

report_path_groups

Reports quantitative information about each existing path group.

Usage

report_path_groups [-mode <mode>] [-detail]

Arguments

- -mode <mode>

  Displays path groups from the specified timing mode.

- -detail

  Includes the following additional columns: Total Negative Slack, Total End Points, Violating End Points, and (%) Violating End Points. The Violating End Points column reports the number of end-points with negative slack.

Description

Displays information for each path group, including: path group name, weight, critical range, worst slack, and timing mode. Use the -detail option to display additional information as described above.

Examples

The following example shows path groups for a multimode design with two timing modes: m1 and m2.

```
% report_path_groups
Report Path Groups For Mode "m1":
-----+-------+------+---------+---------+------
     | Path  |Weight|Critical |Worst    |Timing
     | Group |      |Range(ps)|Slack(ps)| Mode
-----+-------+------+---------+---------+------
1    |default| 1.000|      0.0|    383.5|m1
2    |def_grp| 1.000|      0.0|    401.8|m1
3    |m1_grp | 1.000|      0.0|    521.4|m1
-----+-------+------+---------+---------+------
Report Path Groups For Mode "m2":
-----+-------+------+---------+---------+------
     | Path  |Weight|Critical |Worst    |Timing
     | Group |      |Range(ps)|Slack(ps)| Mode
-----+-------+------+---------+---------+------
1    |default| 1.000|      0.0|    -16.5|m2
2    |def_grp| 1.000|      0.0|      1.8|m2
3    |m2_grp | 1.000|      0.0|    121.4|m2
-----+-------+------+---------+---------+------
```

Related Topics

Report Commands

group_path

report_power

Reports the power consumption information for the design.

Usage

report_power [-all] [-instance <instance>] [-hierarchical] [-internal_power { true | false }] [-switching_power { true | false }] [-leakage_power { true | false }] [[-total { true | false }] | [-total_only]]

Arguments

- -all

  Reports the power values of all instances in the design.

- -instance <instance>

  Reports the power for the specified instance.

- -hierarchical

  Reports the power of each hierarchical instance.

- -internal_power { true |false }

  Reports internal power. By default, this column is included in the report.

- -switching_power { true | false }

  Reports switching power. By default, this column is included in the report.

- -leakage_power { true | false }

  Reports leakage power. By default, this column is included in the report.

- -total { true | false }

  Reports the total power. By default, this column included in the report.

- -total_only

  Reports only the total power. Cannot be used with -internal_power, -switching_power, -leakage_power, or -total arguments.

Description

Power can be shown for a specific instance or at each hierarchical level. By default, -internal_power, -switching_power, -leakage_power, and -total arguments are all set.

Examples

This example reports the internal, switching, leakage, and total power.

```
% report_power

Report Power (instances with prefix '*' are included in total) :
--+-------------+----------------+-----------------+---------------+------------
  | Instance    | Internal Power | Switching Power | Leakage Power | Total Power
  |             | (uw)           | (uw)            | (uw)          | (uw)
--+-------------+----------------+-----------------+---------------+------------
1 |*IR_q_reg[7] | 3.381459       | 0.006728        | 0.057166      | 3.445353
2 |*IR_q_reg[6] | 3.708899       | 0.009105        | 0.057166      | 3.775170
3 |*IR_q_reg[5] | 3.531503       | 0.629139        | 0.057166      | 4.217808
4 |*IR_q_reg[4] | 3.362934       | 0.269173        | 0.057166      | 3.689274
5 |*IR_q_reg[3] | 3.610359       | 0.190609        | 0.057166      | 3.858135
6 |*IR_q_reg[2] | 3.627044       | 0.755787        | 0.057166      | 4.439998
7 |*IR_q_reg[1] | 3.438082       | 0.335269        | 0.057166      | 3.830518
8 |*IR_q_reg[0] | 3.558173       | 0.172637        | 0.057166      | 3.787976
87|*TOTAL       | 588.247375     |16.076899        | 15.91239      | 620.236694
--+-------------+----------------+-----------------+---------------+--------------
```

Related Topics

Report Commands

report_power_domains

Reports the area and power information for specified power domains.

Usage

report_power_domains [-domain <domain_name>] [-details]

Arguments

- -domain <domain_name>

  Specifies the name of the power domain. By default, all power domains are listed along with the leakage, dynamic, and total power for each domain. If -domain <domain_name> is specified, then each of the instances in that domain are shown along with their power values. The module column specifies the module for that instance or partition for instances of rtl partitions generated by the Oasys-RTL tool.

- -details

  Specifies to include region and supply details in the report.

Description

Reports the area in square microns and power in nanowatts for the power domains. Reports the power domains defined in the power constraints from the UPF file. You can run the command after the load_upf command.The report includes leakage, dynamic, and total power, along with the instance area.

Examples

Example 1

This example creates a report that lists each power domain along with its leakage, dynamic, and total power:

```
% report_power_domains

  |power domain |area   |leakage power|dynamic power|total power
--+-------------+-------+-------------+-------------+------------
1 |PD_Default   |6331076| 850469568   | 2935573504  | 37860431362
2 |PD_SparcCores|4037094| 538140928   | 2312752128  | 2850893056
--+-------------+-------+-------------+-------------+------------
```

Example 2

The below example creates a report for the PD_Default power domain. It shows each instance in that domain along with leakage, dynamic, and total power.

```
% report_power_domains -domain PD_Default

Report PowerDomains:
--+--------+--------------+-------+------------+-------------+-----------
  |instance|module        |area   |leakage     |dynamic      |total
  |        |              |       |power       |power        |power
1 |ccx     |ccx__key0     | 217556| 6150610    | 180818000   | 186968608
2 |ctu     |ctu__key0     | 139816| 21551522   | 99012872    | 120564392
3 |dram02  |dram__key0__1 | 154771| 6438656    | 110707344   | 117146000
4 |dram13  |dram__key0    | 154771| 6438656    | 108232160   | 114670816
5 |efc     |efc__key0     | 44185 | 7178602    | 14099128    | 21277730
…
```

Related Topics

Report Commands

report_pst

Reports the power states (modes) in the current design.

Usage

report_pst

Arguments

None.

Description

Reports the power state table definitions that you previously read into the Oasys-RTL tool with the load_upf command. The power states are defined in the UPF file. The Oasys-RTL tool uses the power states when it performs low power synthesis. You must run this command after the load_upf command.

The asterisk (*) in st0* in the Power Modes column signifies the default mode.

Note

This command is for UPF only.

Examples

```
% report_pst

Report PST:
-----+-----------+-----------+--------------+---
     |Power modes|VDD_Default|VDD_SparcCores|VSS
-----+-----------+-----------+--------------+---
1    |st0*       |on85(0.85) |on85(0.85)    |*
2    |st1        |on85(0.85) |on75(0.75)    |*
-----+-----------+-----------+--------------+---
```

Related Topics

Report Commands

report_regions

Reports physical information about specified regions.

Usage

report_regions [-region <regions>]

Arguments

- -region <regions>

  Specifies a region to report.

Examples

```
% report_regions

--+------+-----------+---------------+----+-------------+---------------
  |region|region type|location       |area|physical size|utilization (%)
--+------+-----------+---------------+----+-------------+---------------
1 |__r__0|DEFAULT    |(260.73,329.96)|0   |35686        |0.00
--+------+-----------+---------------+----+-------------+---------------
```

Related Topics

Report Commands

create_region

report_retime

Generates a report on the number of registers in retiming modules.

Usage

report_retime

Arguments

None.

Examples

```
% report_retime

Report Retime:
--+----------+----------------+---------------+----------------
  |          |                | Pre-retime    | Post-retime
  | Module   |    Instance    | register count| register count
--+----------+----------------+---------------+----------------
1 | mult_pipe| mult_pipe_inst1|             32|              57
2 | mult_pipe| mult_pipe_inst2|             32|              43
3 | mult_pipe| mult_pipe_inst3|             32|              71
4 | mult_pipe| mult_pipe_inst4|             32|              54
--+----------+----------------+---------------+----------------
```

Related Topics

Report Commands

report_route_layers

Reports how many resources are available for each routing layer.

Usage

report_route_layers

Arguments

None.

Examples

The following example creates a report showing the percentage of each routing layer that is available for use. The set_route_layer_max_usage command can be used to set the max usage.

```
% report_route_layers

Report Layers:
---+----------+------------+--------------------+------------+------------
   |Layer Name|Max Usage(%)|PWR/GND Pre-Routes(%)|Blockages(%)|Available(%)
---+----------+------------+--------------------+------------+------------
1  |AP        |    0.00    |               0.00 |        0.00|      100.00
2  |M10       |     100.00 |              30.98 |       72.65|       17.28
3  |M9        |     100.00 |              21.33 |       76.60|       17.26
4  |M9        |      20.00 |              26.69 |       60.08|       27.04
5  |M9        |     100.00 |              21.77 |       66.80|       14.85
.....
```

- **Layer Name —** Name of layer from the technology LEF file.
- **Max Usage —** Allowable usage percentage of layer or available percentage of layer for routing.
- **PWR/GND Pre-Routes —** Percentage of layer space used by power and ground nets based on the SPECIALNETS section of the DEF file.
- **Blockages —** Percentage of layer space used by blockages based on the LEF and DEF files.
- **Available —** Percentage of total available space calculated by subtracting the number of blockages from the total tracks divided by the total tracks. The available tracks percentage only considers the routing blockage. It does not consider the Max Usage.

  Available (%) = ((total tracks - blocked tracks)/total tracks) * 100.0

Related Topics

Report Commands

set_route_layer_max_usage

set_max_route_layer

report_rtl_partitions

Reports all RTL partitions that the tool has created.

Usage

report_rtl_partitions

Arguments

None.

Description

Reports the RTL partitions in the design, the number of times it is replicated, the area in square microns, and the number of instances each partition contains.

Note

The Oasys-RTL tool automatically breaks the design into RTL partitions, eliminating the need to divide the design into small blocks.

Examples

```
% report_rtl_partitions

Report RTL Partitions:
----+--------------------------------+-----------+------------+---------
    |RTL Partition                   |Replication|Area (um2)  |Instances
----+--------------------------------+-----------+------------+---------
1   |ctu_clsp GB0                    |          1 |22232.750000|    2964
2   |ctu_clsp_clkgn                  |          1 |16736.126953|    2741
3   |ctu_clsp GB2                    |          1 |15699.599609|    2377
4   |ctu_dft_creg GB0                |          1 |18947.476562|    2087
5   |ctu_dft_creg GB1                |          1 | 7007.313477|     674
6   |ctu_dft GC0                     |          1 |16849.728516|    2166
7   |ctu GC0                         |          1 | 621.633606 |      84
8   |efc GC0                         |          1 | 7922.476562|    1438
9   |c2i GB0                         |          1 |18030.902344|    2089
10  |c2i GB1                         |          1 |14265.115234|    1707
11  |c2i GB2                         |          1 |18800.007812|    1978
12  |c2i GB3                         |          1 |11281.838867|    1172
13  |i2c_sdp                         |          1 |11232.447266|    1956
14  |i2c_buf key0 GP                 |          1 |11110.376953|    1013
.
.
.
800 |bw_io_ddr_impctl_pullup GC0 1   |          1 | 2057.529541|     470
801 |ddr_ch_b GC0 1                  |          1 | 653.385620 |     114
```

Related Topics

Report Commands

report_scan_chains

Reports the information about the scan chains of the design in session.

Usage

report_scan_chains [-name <string>] [-detail] [-physical] [-show_shift_registers] [-mode <string>]

Arguments

- -name <string>

  Specifies the name of the scan chain to report.

- -detail

  Includes a list of all flip-flops in each chain. Reports multi-bit scan registers in the scan chain. In the case of a serial multi-bit, the width appears next to the multi-bit instance names.

- -physical

  Reports physical wire length estimates for scan connections.

- -show_shift_registers

  Reports the composition details of the inferred shift registers in the scan chains.

- -mode <string>

  Limits report to only include scan chains defined in the specified mode. If this switch is not specified, all modes are reported.

Examples

Example

```
% report_scan_chains
Report ScanChains:

Index|Chain|ScanInstance|Length|TestClock|ClockEdge|Lockup|ScanIn|ScanOut
-----+-----+------------+------+---------+---------+------+------+-------
1    |c1   |  20        |  19  |clk[0],clk[1]|rise | 1    |si1   |so1
2    |c2   |  19        |  19  |clk[1]   |rise     | 0    |si2   |so2
3    |c3   |  20        |  20  |clk[2]   |rise     | 0    |si3   |so3
4    |c4   |  19        |  19  |clk[2]   |rise     | 0    |si4   |so4
5    |c5   |  19        |  19  |clk[2]   |rise     | 0    |si5   |so5
6    |c6   |  19        |  19  |clk[2]   |rise     | 0    |si6   |so6
7    |c7   |  19        |  19  |clk[2]   |rise     | 0    |si7   |so7
8    |c8   |  11        |  11  |clk[3]   |rise     | 0    |si8   |so8
```

Note

The value in the Scan Instance column is equal to the sum of the Length and Lockup column values. The Length field denotes the number of scan flip-flops.

Example

```
% report_scan_chains -detail
Report ScanChains:

Index|Chain|ScanInstance     |Length|TestClock    |ClockEdge|Lockup |ScanIn|ScanOut
-------------------------------------------------------------------------------
1    |c1   | 20              |19    |clk[0],clk[1]|rise     |   1   |si1   | so1
1.1  |     |PCcntr/cnt_reg_0 |1     |clk[0]       |rise     |       |      |
1.2  |     |PCcntr/cnt_reg_1 |1     |clk[0]       |rise     |       |      |
…    |
```

Example

```
% report_scan_chains -show_shift_register
Report ScanChains:
--+-----+------------+------+---------+---------+---+----------+-----------
  |Chain|ScanInstance|Length|TestClock|ClockEdge|...|ScanIn    | ScanOut
--+-----+------------+------+---------+---------+---+----------+-----------
 1|core2|     2      |    12|ck       |rise     |   |scan_in[2]|scan_out[2]
 2|core3|     2      |    12|ck       |rise     |   |scan_in[3]|scan_out[3]
 3|core4|     2      |    12|ck       |rise     |   |scan_in[4]|scan_out[4]
 4|core5|     2      |    12|ck       |rise     |   |scan_in[5]|scan_out[5]
 5|core6|     3      |     3|ck       |rise     |   |scan_in[6]|scan_out[6]
 6|core7|     0      |     0|         |mixed    |   |scan_in[7]|scan_out[7]
--+-----+------------+------+---------+---------+---+----------+-----------
Shift Register 'SR_u1_a_reg7':
    u1_a_reg7
    u1_b_reg7
    u1_c_reg7
    u2_a_reg7
    u2_b_reg7
    u2_c_reg7
Shift Register 'SR_u1_a_reg6':
    u1_a_reg6
    u1_b_reg6
    u1_c_reg6
    u2_a_reg6
    u2_b_reg6
    u2_c_reg6
Shift Register 'SR_u1_a_reg5':
    u1_a_reg5
    u1_b_reg5
    u1_c_reg5
    u2_a_reg5
    u2_b_reg5
    u2_c_reg5
Shift Register 'SR_u1_a_reg4':
    u1_a_reg4
    u1_b_reg4
    u1_c_reg4
    u2_a_reg4
    u2_b_reg4
    u2_c_reg4
Shift Register 'SR_u1_a_reg3':
    u1_a_reg3
    u1_b_reg3
    u1_c_reg3
    u2_a_reg3
    u2_b_reg3
    u2_c_reg3
```

Related Topics

Report Commands

define_scan_chain

remove_scan_chain

report_switching_activity

Reports the switching activity values of the pins in the design.

Usage

report_switching_activity [-probability] [-toggle_percentage] [-toggle_rate] [-pin <pin_list>] [-hierarchical] [-average] [-all]

Arguments

- -probability

  Optional. Reports the percentage probability that a pin has a value of 1. This is reported by default.

- -toggle_percentage

  Optional. Reports the toggle percentage of a pin. This is the number of toggles per clock cycle.

- -toggle_rate

  Optional. Reports the toggle rate of a pin, in toggles per time unit. This is reported by default.

- -pin <pin_list>

  Optional. Specifies a list of pins for which the activity is reported.

- -hierarchical

  Optional. Reports the activity for all outputs if the specified instances are not hierarchical instances. For hierarchical instances, the specified activity value is reported to the outputs of the leaf instances in the specified hierarchical instances without traversing the hierarchy.

- -average

  Optional. Reports the specified activity value for the outputs of all leaf instances in the hierarchy of each specified hierarchical instance.

- -all

  Optional. Reports the switching activity for all pins in the design.

Examples

This example reports the switching activity. For each of the pins, the table shows the probability that the pin value is one and the toggle rate in toggles per unit of time.

```
% report_switching_activity

Report Switching activities:
-----+-------------+------------+------------
     | Pin         | Probability| Toggle rate
-----+-------------+------------+------------
1    |wr[7]        |    0.671534|    0.352922
2    |wr[6]        |    0.523928|    0.399084
3    |wr[5]        |    0.524368|    0.399050
4    |wr[4]        |    0.533911|    0.398160
5    |wr[3]        |    0.631218|    0.372451
6    |wr[2]        |    0.485250|    0.399652
7    |wr[1]        |    0.484743|    0.399628
8    |wr[0]        |    0.482801|    0.399527
9    |PC[7]        |    0.407684|    0.386365
10   |PC[6]        |    0.312500|    0.343750
11   |PC[5]        |    0.312500|    0.343750
. . .
```

Related Topics

Report Commands

set_switching_activity

report_test_clocks

Reports test clock-related information for test clocks in the design.

Usage

report_test_clocks

Arguments

None.

Description

The report_test_clocks command displays test clock-related information in a tabular format for all the test clocks of the design. This includes the pin, associated test domain, root clock, rise/fall and inverted/non-inverted information about these clocks.

Examples

```
% report_test_clocks

Report TestClocks:
-----+-------+-----------+----------+-----+------|------------
     | Pin   | TestDomain| RootClock| Rise| Fall | Inverted
-----+-------+-----------+----------+-----+------|------------
1    | clk[0]| A         | clk[0]   | 25  | 75   |  no
2    | clk[1]| A         | clk[1]   | 25  | 75   |  no
3    | clk[2]| B         | clk[2]   | 25  | 75   |  no
4    | clk[3]| C         | clk[3]   | 25  | 75   |  no
```

Related Topics

Report Commands

define_test_clock

remove_test_clock

report_test_pins

Reports information about the pin (definition point) and its scan mode value for all the test pins defined in the design.

Usage

report_test_pins

Arguments

None.

Examples

```
% report_test_pins

Report TestPins:
--+-----+----------
  | Pin | ScanMode
--+-----+----------
1 | se  | 1
2 | tm  | 1
3 | rst | 0
--+-----+----------
```

Related Topics

Report Commands

define_test_pin

remove_test_pin

report_timing

Reports the worst paths (based on slack) in the design that satisfies the given constraints.

Usage

report_timing [-hierarchical] [-net] [{-from | -rise_from | -fall_from} <from_list>]
[{{-through | -rise_through | -fall_through} <through_list>} ...]
[{-to | -rise_to | -fall_to} <to_list>] [-rise] [-fall] [-max_paths <number_of_paths>] [-mode]
[-combined] [-group <group>] [-format <column_list>]

Arguments

- -hierarchical

  Displays hierarchical information and shows additional timing details.

- -net

  Adds net name and net delay information to the report. By default, only pin information is displayed.

- {-from | -rise_from | -fall_from} <from_list>

  Specifies a list of clocks, pins, or instances. Report timing for paths that start at any given object in the list. These paths can start at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from). Zero or one of -from, -rise_from, or -fall_from arguments can be used at once. If an instance is specified, this means the source can be any output of that instance.

- {{-through | -rise_through | -fall_through} <through_list>} ...

  Reports timing for paths through any given object in the list. The <through_list> is a list of pins, nets or instances. Multiple -through, -rise_through, or -fall_through arguments can be specified, in which case the report applies to any path that goes through at least one object in each <through_list>, in exactly the specified order. These paths can go through the rising edge (-rise_through), falling edge (-fall_through), or both edges (-through).

- {-to | -rise_to | -fall_to} <to_list>

  Reports timing for paths that end at any given object in the list. The <to_list> is a list of clocks, pins or instances. Zero or one of arguments -to, -rise_to, or -fall_to can be used at once.

- -rise

  Reports a path that has a rising edge at the end point of the path satisfying construct -to, -through, and -from.

- -fall

  Reports a path that has a falling edge at the end point of the path satisfying construct -to, -through, and -from.

- -max_paths <number_of_paths>

  Specifies the maximum number of paths to report for each path group. By default, one path is reported per path group.

- -mode <mode>

  Specifies the timing mode. The -mode option is only effective when specified with the from, through, or to lists. If a mode and a from, through, or to lists are specified, the report satisfies both conditions.

- -combined

  Specifies to combine path groups into one report. The -combined option has an effect only if there are path groups in the design. Without this option, this command reports separately for each path group. For instance, if there are two path groups, there are two reports in report_timing results. When the -combined option is used, report_timing only has one report.

- -group <group>

  Specifies that the report only contains the worst endpoints of the specified <group>. Groups can be defined using the group_path command.

- -format <column_list>

  Specifies what columns should be printed out in the report. The <list> is a Tcl list that can include any of the following items: cell, slew, net_delay, arc_delay, delay, arrival, edge, net_load, pin_load, load, fanout, location, power_domain, wire_length, and wire_load_model.

Description

Reports the timing paths in the design ordered by the slack time at the endpoints for the given constraints. The following items are reported:

- Startpoint
- Endpoint
- Data required time
- Setup time
- Data arrival time - The arrival time includes cell delay, physical net delay based on cell locations, capacitive load, and any virtual buffers.
- Slack - Required time / arrival time.
- Table showing detailed information about the instances/nets in the timing path

The delay in report_timing shows the combination of cell and net delay. Parameters control the precision (timing_report_significant_digits) and units (timing_units_for reports) of the timing values in the report. The pin count is also reported in report_timing -hierarchical.

When instances have high fanout or high capacitive load, Oasys-RTL timing analysis accounts for buffer delays that would be required. These virtual buffers are not actually inserted in the design. You must insert buffers and fix design rules in the physical implementation tool. Instances that are virtually buffered are indicated with an asterisk (*) after their names. The report_timing command indicates various details of a path by suffixing symbols to the end of the library cell on the path. The following legend can be used to infer the attributes:

Table 5-7. Suffix Definitions for the report_timing Command

Symbol	Definition
*	An output has been virtually buffered for timing.
DU	A library cell is dont_use.
DT	A cell is dont_touch.
DR	Indicates that the delay of a cell has been derated.
retime borrow	A state cell has a retime constraint and timing was borrowed.

The following example shows a timing path for which the Oasys-RTL tool has added a virtual buffer delay denoted with an *:

```
fpu/fpu_add/fpu_add_frac_dp/i_0_26_0/A1->ZN
AND2_X2* 15.1 94.6 1236.7 ff 46.2 3098, 1982 PD_Default (1.25)
```

Note

Unconstrained paths are not reported.

Examples

The following command reports the worst path in the design with any edge for the worst endpoint pin. The path goes from the startpoint to the endpoint. Data required time (clock period uncertainty), data arrival time, and slack time are reported. Finally, the complete path from startpoint to endpoint is shown.

Example

```
% report_timing

---------------------------------------------------------------------------
Startpoint: ctu/ctu_clsp/.../i0/q_reg/Q (Clocked by PLL_RAW_CLK R)
Endpoint: ctu/ctu_clsp/.../i0/q_reg/D (Clocked by PLL_RAW_CLK F)
Data required time: 342.5 (Clock period: 416.5, minus
Uncertainty: 30.0, plus Latency 0.0, minus
Setup time: 44.0)
Data arrival time: 521.6
Slack: -179.1

---------------------------------------------------------------------------
Arrival
Path                          Slew   Delay   Time Edge   Load   Location
(ps)    (ps)     (ps)         (ff)   (um,um)
---------------------------------------------------------------------------
ctu/ctu_clsp/.../u_muxi210/i_0/i_0/B->Y (MXI2X1)
0.0      0.0    400.0    rf      0.0  862, 2732 ctu/ctu_clsp/.../u_inv0/i_0/i_0/A->Y (INVX2)
0.0      0.0    400.0    fr      0.0  825, 2731 ctu/ctu_clsp/.../u_synchronizer_ff1_nsr0/i0/
q_reg/CK->Q
(DFFSRXL)
0.0     98.7    498.7    rf      5.5  819, 2734
ctu/ctu_clsp/.../u_synchronizer_neg_ff_nsr0/i0/i_0/i_1/A->Y (INVX2)
30.9     7.6    521.6    rf      2.4  797, 2729 ctu/ctu_clsp/.../u_synchronizer_neg_ff_nsr0/i0/
q_reg/D
(DFFSRXL)
14.1     0.0    521.6    f            793, 2731
---------------------------------------------------------------------------
```

Example

The following command reports the worst path in the design with a falling edge for the worst endpoint pin.

```
% report_timing -fall

------------------------------------------------------------------------
Startpoint:
ctu/ctu_clsp/.../u_synchronizer_ff1_nsr0/i0/q_reg/Q (clocked by PLL_CLK R)
Endpoint: ctu/ctu_clsp/.../u_synchronizer_neg_ff_nsr0/i0/q_reg/D (Setup time: 44.000,
clocked by
PLL_CLK F)
Data required time: 342.500 (Clock period: 416.500, minus Uncertainty: 30.000, minus Setup
time:
44.000)
Data arrival time: 520.700
Slack: -178.200
------------------------------------------------------------------------
Arrival
Path                        Slew   Delay    Time Edge    Load    Location
(ps)    (ps)    (ps)          (ff)    (um,um)
------------------------------------------------------------------------
ctu/ctu_clsp/.../u_nand2/i_0/i_0/B->Y (NAND2X2)
0.0 0.0 400.0 rf 0.0 2235,2375
ctu/ctu_clsp/.../u_inv/i_0/i_0/A->Y (INVX2)
0.0 0.0 400.0 fr 0.0 2235,2375
ctu/ctu_clsp/.../i0/q_reg/CK->Q (DFFSRXL)
0.0 98.1 498.1 rf 4.9 2235,2375
ctu/ctu_clsp/.../u_synchronizer_neg_ff_nsr0/i0/i_0/i_0/A->Y (NAND2X2)
34.5 14.8 512.9 fr 4.1 2235,2375
ctu/ctu_clsp/u.../u_synchronizer_neg_ff_nsr0/i0/i_0/i_1/A->Y (INVX2)
29.9 7.8 520.7 rf 3.4 2235,2375
ctu/ctu_clsp/.../u_synchronizer_neg_ff_nsr0/i0/q_reg/D (DFFSRXL)
14.4 0.0 520.7 f 2235,2375
------------------------------------------------------------------------
```

Example

This example reports the worst path from reg/Q with a falling edge for reg/Q:

```
% report_timing -fall -from reg/Q
```

Example

This example reports the worst path from reg/Q through i6/A pin with a falling edge for i6/A:

```
% report_timing -fall -from reg/Q -through i6/A
```

Example

This example reports the worst path in the design launched by CLKA, going through either i2/A or i3/A, and then going through i7/Y with a rising edge at i7/Y:

```
% report_timing -from CLKA -through {i2/A i3/A} -through i7/Y -rise
```

Example

This command also reports the hierarchical paths (-hierarchical) between startpoint and endpoint along with net information (-net). Starting at the end of this report and going backwards, you can trace the clock path as follows:

```
% report_timing -net -hierarchical
---------------------------------------------------------------------------
Startpoint:
ctu/ctu_clsp/u_ctu_clsp_clkgn/.../u_synchronizer_ff1_nsr0/i0/q_reg/Q
clocked by PLL_CLK R)

Endpoint:
ctu/ctu_clsp/u_ctu_clsp_clkgn/.../u_synchronizer_neg_ff_nsr0/i0/q_reg/D (Setup time: 44.000,
clocked by PLL_CLK F)

Data required time: 342.500 (Clock period: 416.500, minus

Uncertainty: 30.000, minus Setup time:
44.000)

Data arrival time: 520.700

Slack: -178.200
---------------------------------------------------------------------------
Net    Arc
Arrival
Path             Slew  Delay  Delay    Time Edge    Load  Pins Location(
                 (ps)  (ps)   (ps)       (ps)       (ff)  (#)  (um,um)
---------------------------------------------------------------------------
ctu/ctu_clsp/i_1/pll_clk_out_pre (net)
ctu/ctu_clsp/u_ctu_clsp_clkgn/pll_clk_out (module:
ctu_clsp_clkgn)

ctu/ctu_clsp/u_ctu_clsp_clkgn/pll_clk_out (net)
ctu/ctu_clsp/ ... /pll_clk_out (module: ctu_clsp_clkgn_fstlog)
…
ctu/ctu_clsp/ ... /u_cmp_div_bypass_gated_bar_gated/z (net) ctu/ctu_clsp/
... /u_cmp_div_bypass_gated_bar_gated/a (module: ctu_and2  2)
ctu/ctu_clsp/ ... /cmp_div_bypass (net) ctu/ctu_clsp/ ... /pll_clk_out
(module: ctu_clsp_clkgn_fstlog) ctu/ctu_clsp/u_ctu_clsp_clkgn/pll_clk_out
(net) ctu/ctu_clsp/u_ctu_clsp_clkgn/pll_clk_out (module: ctu_clsp_clkgn)
```

Example

This example specifies the columns of the timing report.

```
% report_timing -format {cell net_delay arc_delay arrival}
--------------------------------------------------------------------------
Startpoint: exu/alu/addsub/sub_dff/q_reg[63]/Q
    (Clocked by GCLK R)
Endpoint: exu/ecl/dff_mem_invalid_e2m_q_reg[0]/D
    (Clocked by GCLK R)
Path Group: default
Data required time: 1181.8
    (Clock period: 1500.0, minus Uncertainty: 100.0, plus Latency 0.0, minus Setup time:
218.2)
Data arrival time: 1247.7
Slack: -65.9
Logic depth: 41
--------------------------------------------------------------------------
                                                  Net      Arc  Arrival
Path                     Module/Cell            Delay    Delay     Time
                                                 (ps)     (ps)     (ps)
--------------------------------------------------------------------------
gclk                     {create_clock}                    0.0      0.0
spc_hdr/I0/i_0_0/A2->ZN  AND2_X4_HVT             0.0      0.0      0.0
exu/alu/addsub/sub_dff/q_reg[63]/CK->Q
                         SDFF_X1_HVT#*           0.0    282.9    282.9
exu/alu/addsub/i_1_1_74/A2->ZN
                         NAND2_X4_HVT            0.1     15.6    298.6
exu/alu/addsub/i_1_1_72/A2->ZN
                         NAND2_X4_HVT            0.0     41.7    340.3
exu/alu/addsub/adder/i_0_0_442/A->ZN
                         INV_X8_HVT              0.1      8.7    349.1
exu/alu/addsub/adder/i_0_0_440/A1->ZN
                         NAND2_X4_HVT            0.0     25.1    374.2
exu/alu/addsub/adder/i_0_0_437/A2->ZN
                         NAND2_X4_HVT            0.0     13.5    387.7
exu/alu/addsub/adder/i_0_0_436/A1->ZN
                         NAND2_X4_HVT            0.0     16.7    404.4
exu/alu/addsub/adder/i_0_0_435/A1->ZN
                         NAND2_X4_HVT            0.0     12.0    416.4
exu/alu/addsub/adder/i_0_0_427/A1->ZN
                         NAND3_X4_HVT            0.0     63.7    480.1
exu/alu/addsub/adder/i_0_0_412/A1->ZN
                         NAND2_X4_HVT            0.0     14.4    494.5
exu/alu/addsub/adder/i_0_0_403/A1->ZN
                         NAND2_X4_HVT            0.0     37.0    531.5
```

Related Topics

Report Commands

config_report

report_timing_exceptions

Reports the timing exceptions accepted by the tool. Timing exceptions that can be reported include false paths, multicycle paths, min/max delays, path groups, and reset_path.

Usage

report_timing_exceptions [-type <enum>] [-setup <bool>] [-hold <bool>]
[-modes <mode_list>] [-from <pin_list/src_clock_list>] [-through <pin_list>]
[-to <pin_list/target_clock_list>] [-filter <string>]

Arguments

- -type <enum>

  Optional. Specifies the type of exception to report. The following are legal values:

  - all (default)
  - false_path
  - multicycle_path
  - min_max_delay
  - group_path
  - reset_path

  At least one of the -type, -from, -through, or -to options must be specified.

- -setup <bool>

  Optional. Reports only setup exceptions.

- -hold <bool>

  Optional. Reports only hold exceptions.

- -modes <mode_list>

  Optional. Specifies a list of design modes.

- -from <pin_list/src_clock_list>

  Optional. Specifies a list of “from” pins or a list of source clocks. At least one of the -type, -from, -through, or -to options must be specified.

- -through <pin_list>

  Optional. Specifies a list of “through” pins. At least one of the -type, -from, -through, or -to options must be specified.

- -to <pin_list/target_clock_list>

  Optional. Specifies a list of “to” pins or target clocks. At least one of the -type, -from, -through, or -to options must be specified.

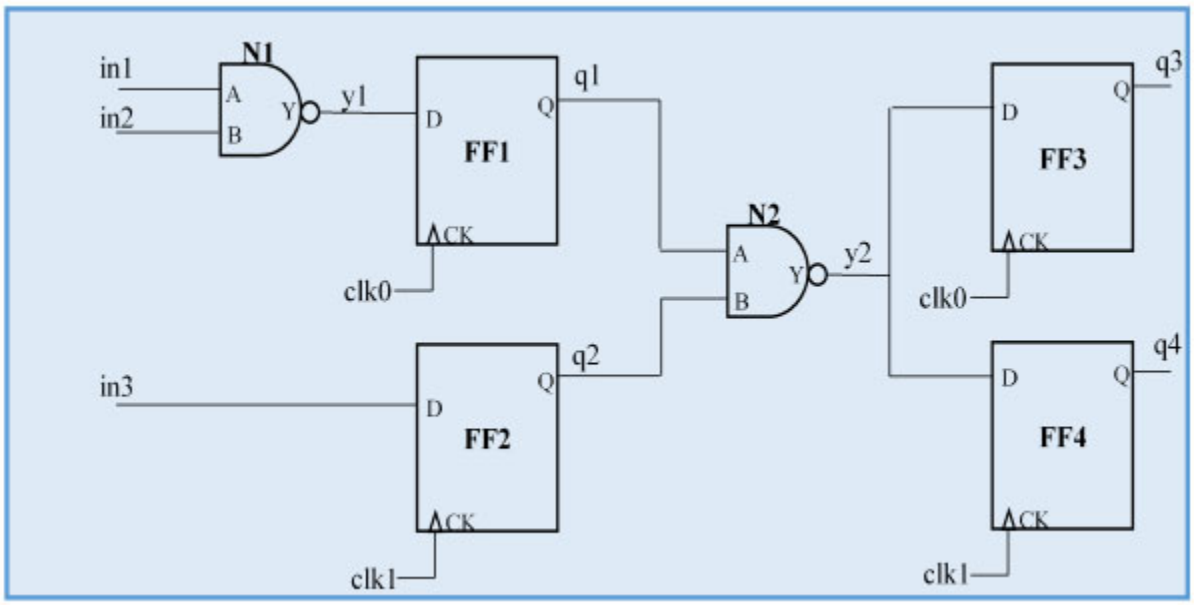
- -filter <string>

  Optional. Filters objects using the specified expressions. Supports only the ==, !=, &&, and || operators.

Examples

Assume the following sample design netlist is loaded:

Figure 5-5. Sample Design Netlist



and that the timing exemptions in the following sample SDC file are loaded:

```
create_clock -period 6 -waveform {0 3} -name clk0 [get_ports clk0]
create_clock -period 10 -waveform {0 5} -name clk1 [get_ports clk1]
...
...
##Exceptions##
set_false_path -from [get_clocks clk0] -to [get_clocks clk1]
set_multicycle_path -setup 3 -from [get_pins FF2/CK] -to [get_pins FF4/D]
```

The following examples show a few scenarios for reporting timing exceptions:

Scenario 1:

Report all timing exceptions that pass through the pin N2/Y:

```
% report_timing_exceptions -through [get_pins N2/Y]

Timing Exception #0:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | False Path
Setup           | True
Hold            | True
Mode            | common_mode
(1) From        | Clock: clk0
(2) To          | Clock: clk1
----------------+--------------------

Timing Exception #1:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | Multicycle Path
Setup           | Mult = 3
Hold            | Mult = 0 (Default)
Mode            | common_mode
(1) From        | Pin: FF2/Q
(2) To          | Pin: FF4/D
----------------+--------------------
```

Scenario 2:

Report all exceptions ending at the pin FF4/D

```
% report_timing_exceptions -to [get_pins FF4/D]

Timing Exception #1:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | Multicycle Path
Setup           | Mult = 3
Hold            | Mult = 0 (Default)
Mode            | common_mode
(3) From        | Pin: FF2/Q
(4) To          | Pin: FF4/D
----------------+--------------------
```

Scenario 3:

Report all exceptions whose endpoints are the clock clk1:

```
% report_timing_exceptions -to [get_clocks clk1]

Timing Exception #0:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | False Path
Setup           | True
Hold            | True
Mode            | common_mode
(1) From        | Clock: clk0
(2) To          | Clock: clk1
----------------+--------------------

Timing Exception #1:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | Multicycle Path
Setup           | Mult = 3
Hold            | Mult = 0 (Default)
Mode            | common_mode
(1) From        | Pin: FF2/Q
(2) To          | Pin: FF4/D
----------------+--------------------
```

Scenario 4:

Report all exceptions except false_path exceptions whose endpoints are the clock clk1:

```
% report_timing_exceptions -to [get_clocks clk1] \
   -filter "@type!=false_path"

Timing Exception #1:
----------------+--------------------
Attribute       | Value
----------------+--------------------
Type            | Multicycle Path
Setup           | Mult = 3
Hold            | Mult = 0 (Default)
Mode            | common_mode
(3) From        | Pin: FF2/Q
(4) To          | Pin: FF4/D
----------------+--------------------
```

Scenario 5:

Report all min/max delay exceptions:

```
% report_timing_exceptions -type min_max_delay

Warning: No timing exception found!!!
```

Related Topics

Report Commands

remove_timing_exceptions

report_timing_mode

Reports all timing constraint modes and operating conditions.

Usage

report_timing_mode

Arguments

None.

Examples

The following example reports two timing modes: fast and normal. The normal timing mode is currently selected, as indicated by the “active” tag. If you specify additional timing constraints, they are added to the active mode.

Figure 5-6. Command Output

```
[oasys-RTL]$ report_timing_mode
Report Modes:
-----+----------------------
     | Name
-----+----------------------
   1 | fast
-----+----------------------
   2 | normal (active)
-----+----------------------

Report Operating conditions:
-----+--------------+--------+------------+---------------------------------+--------+----------------+-----------
     |Name          |Default?|Type        |Library                          |Process |Voltage         |Temperature
-----+--------------+--------+------------+---------------------------------+--------+----------------+-----------
1    |worst_low_0p85V|yes     |standard cell|NangateOpenCellLibrary_45nm_HVT_0p85|1.000000|         0.850000| -40.000000
2    |worst_low      |        |standard cell|NangateOpenCellLibrary_45nm_HVT     |1.000000|         0.950000| -40.000000
3    |typical        |        |standard cell|IO                               |1.000000|         1.100000|  27.000000
4    |TYP            |        |standard cell|PLL_TYP                          |1.000000|         0.900000|  25.000000
5    |typical        |        |standard cell|MemGen_16_10                     |1.000000|         1.800000|  25.000000
6    |worst_low      |        |multi-voltage|NangateOpenCellLibrary_45nm_HVT     |1.000000|0.000000:0.000000| -40.000000
7    |worst_low      |        |multi-voltage|NangateOpenCellLibrary_45nm_HVT     |1.000000|0.950000:0.800000| -40.000000
8    |worst_low      |        |multi-voltage|NangateOpenCellLibrary_45nm_HVT     |1.000000|0.800000:0.950000| -40.000000
-----+--------------+--------+------------+---------------------------------+--------+----------------+-----------
```

Related Topics

Report Commands

get_timing_modes

report_units

Outputs the default units for values when reading and writing SDC files

Usage

report_units

Arguments

None.

Description

This command reports the default units for time, capacitance, resistance, power, voltage, and current that Oasys uses when reading and writing SDC files.

The “Input Value” column denotes the default units when Oasys reads SDC files. You can change the units in this column using the *set_unit* command.

The “Output Value” column denotes the default units when Oasys generates reports and writes SDC files. You can change the units in this column by setting the corresponding **_units_for_reports* parameter using hte set_parameter command (eg. *set_parameter time_unites_for_reports ns*).

Examples

```
% report_units
Report SDC units:
-----+-----------+-----------+------------
     |Unit       |Input Value|Output Value
-----+-----------+-----------+------------
1    |Time       |ns         |ps
2    |Capacitance|pf         |ff
3    |Resistance |kohm       |ohm
4    |Power      |nW         |uW
5    |Voltage    |V          |V
6    |Current    |uA         |uA
-----+-----------+-----------+------------
```

Related Topics

Report Commands

get_leakage_power_unit

get_cap_unit

get_time_unit

get_voltage_unit

get_res_unit

get_current_unit

set_units

report_upf_status

Reports the status of the load_upf and commit_upf commands.

Usage

report_upf_status [-detail] [-load_only] [-commit_only]

Arguments

- -detail

  Provides additional level of detail in the report. This argument is not compatible with the -load_only argument.

- -load_only

  Reports the status of UPF commands read with the load_upf command. This argument is not compatible with the -detail or -commit_only arguments.

- -commit_only

  Reports a summary of retention cells inserted with the commit_upf command. Includes the domain, retention rule name, total number of instances, success/fail status, and details of any failures. Also reports a summary of multi-voltage cell inserted with the commit_upf command. Includes the source domain, sink domain, isolation rule name, level_shifter rule name, total number of boundary nets, success/fail status, count of multi-voltage cells inserted and details of any failures.

Examples

Example 1

The following example shows the output of the report_upf_status command:

```
> report_upf_status
UPF Files:
-------------------------------------------------------------------------------------------
<test-case-path>/test.upf
-------------------------------------------------------------------------------------------
Load UPF Status Summary:
-----+--------------------+-----+------+-------+--------+-------
     |command             |total|failed|ignored|warnings|status
-----+--------------------+-----+------+-------+--------+-------
1    |create_power_domain |    4|     0|      0|       0|success
2    |create_supply_port  |    5|     0|      0|       0|success
3    |create_supply_net   |   11|     0|      0|       0|success
4    |connect_supply_net  |    5|     0|      0|       0|success
5    |set_domain_supply_net|   4|     0|      0|       0|success
6    |add_port_state      |    7|     0|      0|       0|success
7    |create_pst          |    1|     0|      0|       1|success
8    |add_pst_state       |    2|     0|      0|       0|success
9    |set_isolation       |    2|     0|      0|       0|success
10   |set_isolation_control|   2|     0|      0|       0|success
11   |set_retention       |    2|     0|      0|       0|success
12   |set_retention_control|   2|     0|      0|       0|success
13   |set_level_shifter   |    2|     0|      0|       0|success
-----+--------------------+-----+------+-------+--------+-------
Retention Cells Insertion Summary:
-----+-------+--------+---------+-------+----+--------------
     |domain |rule    |instances|success|fail|failure reason
-----+-------+--------+---------+-------+----+--------------
1    |PD_CPU1|ret_cpu1|        4|      4|   0|
2    |PD_CPU3|ret_cpu3|        4|      4|   0|
-----+-------+--------+---------+-------+----+--------------
Multi-Voltage Cells Insertion Summary:
-----+-------------+-----------+--------+-------+-------------+-------+----+----+---+
     |source       |sink       |iso     |ls     |boundary|success|fail|ISO|LS  |ELS|failure
     |domain       |domain     |rule    |rule   |nets    |       |    |   |    |   |reason
-----+-------------+-----------+--------+-------+-------------+-------+----+----+---+
1    |/PD_CPU1     |/PD_TOP    |iso_cpu1|--     |       4|      4|   0|  4|   0|  0|
2    |/PD_TOP      |/PD_CPU2   |--      |ls_cpu2|       5|      5|   0|  0|   5|  0|
3    |/PD_TOP      |/PD_CPU3   |--      |ls_cpu3|       5|      5|   0|  0|   5|  0|
4    |/PD_CPU2     |/PD_TOP    |--      |ls_cpu2|       4|      4|   0|  0|   4|  0|
5    |/PD_CPU3     |/PD_TOP    |iso_cpu3|ls_cpu3|       4|      4|   0|  0|   0|  4|
-----+-------------+-----------+--------+-------+-------------+-------+----+----+---+
```

Related Topics

Report Commands

commit_upf

load_upf

Chapter 6
Check Commands

The check commands verify and report aspects of your design.

Table 6-1. Check Commands

Command	Description
check_dft	Checks each flip-flop to see if it passes standard DFT rules before including them in a scan chain.
check_library	Checks that the library contains the necessary cells for synthesis.
check_mv	Checks for multi-voltage domain errors.
check_netlist	Checks the netlist for multi-driven and un-driven nets.
check_placement	Checks for illegally placed macros and outputs the results.
check_timing	Checks for any timing problems with the design.

check_dft

Checks each flip-flop to see if it passes standard DFT rules before including them in a scan chain.

Usage

check_dft [-items <list_of_items>] [-ignore_clock_gating] [-auto_test_clocks] [-auto_test_pins] [-verbose]

Arguments

- -items <list_of_items>

  Specifies a list of items to check. The default is all items.

- -ignore_clock_gating

  Specifies to temporarily ignore any clock-gating cells on the clock path when check_dft is issued. This allows you to connect the test pins of the clock-gating cells after optimize, as connect_clock_gating_test_pins uniquifies the design.

- -auto_test_clocks

  Specifies to automatically define any top-level pin (that controls the clock of a flop) as a test clock.

- -auto_test_pins

  Automatically defines any top-level pin (that controls the asynchronous set or reset of a flop) as a test pin.

- -verbose

  Specifies that all warning and error messages associated with the checking are shown. You can specify the limit for the number of times that a specific message is output by setting the message_suppress_limit parameter.

Description

Checks for each flop to see if it passes standard DFT rules before inclusion in a scan chain. Should be run after the synthesize, define_test_pin, and remove_test_clock commands.

Only those flip-flops that are mapped to scan cells are checked for DFT rules. See set_dont_scan and synthesize -map_to_scan to control scan mapping.

If a design has clock-gating cells, they must have their test control pin connected to a test pin so as to allow the clock to go through in scan-mode. See set_clock_gating_options for more details.

Also checks DFT rules for scan models that are identified using the read_ctl or define_scan_model commands.

Supports scan-mapping and scan-stitching for multi-bit registers. Handles serial scan multi-bit mapping. Handles multi-bit registers by treating them as scan segments.

The parallel_scan_clock check reports a violation on multibit scan flops when they contain more than one SI/SO pair (parallel scan flops) and no scan-model has been defined on them. This check is only applicable in the Tessent flow.pwd

A successful check_dft command generates the following report:

```
% check_dft

starting check_dft at 00:18:30(cpu)/23:00:35(wall) 630MB(vsz)
Checking DFT rules for 'OpenSPARCT1' ...
Report Check DFT:
--+-------------------------------------------------------------------
  |Item                  |Errors|Warnings|Status|Description
--+-------------------------------------------------------------------
1 |internal_clock        |0     |0       |Passed|Internal Clock
2 |constant_clock        |0     |0       |Passed|Constant Clock
3 |non_clock_PI          |0     |0       |Passed|Non-Clock PI
4 |blocking_clock_gate   |0     |0       |Passed|Blocking clock gate
5 |internal_async        |0     |0       |Passed|Internal Async. Set/Reset
  |                      |      |        |      |control
6 |constant_active_async|0     |0       |Passed|Constant active Async.
  |                      |      |        |      |Set/Reset signal
7 |non_test_PI           |0     |0       |Passed|Unconstrained PI driving
  |                      |      |        |      |Async/ Set/Reset
8 |async_clock_conflict  |0     |0       |Passed|Async. Set/Reset signal
  |                      |      |        |      |and Clock conflict
9 |parallel_scan_clock   |0     |0       |Passed|Clock pin of unsupported
  |                      |      |        |      | parallel-scan flop
--+-------------------------------------------------------------------
Design has 0 DFT violation(s)
finished check_dft at 00:19:05(cpu)/23:01:14(wall) 641MB(vsz)
```

The tool -auto_test_clock and -auto_test_pins can resolve errors or warnings on non_clock_PIs or non_test_PIs. In cases of errors or warnings on clocks or asynchronous signals, use the fix_dft_violations command to fix the violations.

Examples

Example

This example shows warnings and error messages when automatically defining test clocks and test pins.

```
% check_dft -verbose -auto_test_clocks -auto_test_pins
-------> Message [DFT-204] suppressed 6 times
warning: detected a blocking clock-gate 'test_stub/bist_ctl_reg_5_1_clk_gate_q_reg/GCK' -
connect a proper test pin to its enable logic  [DFT-204]
Report Check DFT:
-+--------------------+------+--------+------+----------------------------------------
 |Item                |Errors|Warnings|Status|Description
-+--------------------+------+--------+------+----------------------------------------
1|internal_clock      |     0|     168|Failed|Internal Clock
2|constant_clock      |     0|       0|Passed|Constant Clock
3|non_clock_PI        |     0|       0|Passed|Non-Clock PI
4|blocking_clock_gate |     0|     130|Failed|Blocking clock gate
5|internal_async      |     0|       0|Passed|Internal Async. Set/Reset control
6|constant_active_async|     0|       0|Passed|Constant active Async. Set/Reset signal
7|non_test_PI         |     0|       0|Passed|Unconstrained PI driving Async/ Set/Reset
8|async_clock_conflict|     0|       0|Passed|Async. Set/Reset signal and Clock conflict
9|parallel_scan_clock |     0|       0|Passed|Clock pin of unsupported parallet-scan flop
-+--------------------+------+--------+------+----------------------------------------
Design has 298 DFT violation(s)
finished check_dft at 00:00:14(cpu)/0:01:15(wall) 247MB(vsz)
298
```

Related Topics

Check Commands

DFT Commands

set_clock_gating_options

read_ctl

define_scan_model

synthesize

remove_test_clock

set_dont_scan

define_test_pin

fix_dft_violations

check_library

Checks that the library contains the necessary cells for synthesis.

Usage

check_library [-items <list_of_items>] [-verbose]

Arguments

- -items <list_of_items>

  Optional. Specifies a list of items to check. Valid values are logical_only_cell, physical_only_cell, no_basic_gates, no_clock_gate_cells, and bad_physical_lib. The default behavior is to check all items.

- -verbose

  Optional. Specifies that all warning and error messages associated with the checking are shown. You can specify the limit for the number of times that a specific message is generated by setting the message_suppress_limit parameter.

Examples

Example 1

```
% check_library
Report Library Warnings:
--+------------------+----+------+------+---------------------------
  |Item              |Info|Errors|Status|Description
--+------------------+----+------+------+-----------------------------
1 |logical_only_cell | 0  | 0    |OK    |Logical only cells exist in the
  |                  |    |      |      |libraries
2 |physical_only_cell| 0  | 0    |OK    |Physical only cells exist in the
  |                  |    |      |      |libraries
3 |no_basic_gates    | 0  | 0    |OK    |No basic gates for synthesis or
  |                  |    |      |      |mapping
4 |no_clock_gate_cells| 0 | 0    |OK    |No clock-gating cells for
  |                  |    |      |      |clock-gating
5 |bad_physical_lib  | 0  | 0    |OK    |Bad physical libraries
  |                  |    |      |      |(no layer etc.)
--+------------------+----+------+------+---------------------------
```

Example 2

```
% check_library -verbose
info: clock-gating cell for posedge FFs = CLKGATE_X4 in target library
'default' [POWER-112]

info: no clock-gating cell found in target library 'default' for negedge
FFs for the given specification [POWER-113]
Report Library Warnings:
--+------------------+----+------+------+-----------------------------
  |Item              |Info|Errors|Status|Description
--+------------------+----+------+------+-----------------------------
1 |logical_only_cell | 0  | 0    |OK    |Logical only cells exist in the
  |                  |    |      |      |libraries
…
```

Example 3

```
% check_library -items no_clock_gate_cells -verbose
info: clock-gating cell for posedge FFs = CLKGATE_X4 in target library
'default' [POWER-112]

info: no clock-gating cell found in target library 'default' for negedge
FFs for the given specification [POWER-113]

Report Library Warnings:
--+------------------+----+------+------+--------------------------
  |Item              |Info|Errors|Status|Description
--+------------------+----+------+------+--------------------------
1 |no_clock_gate_cells| 0  | 0    |OK    |No clock-gating cells for
  |                  |    |      |      |clock-gating
--+------------------+----+------+------+--------------------------
```

Related Topics

Read Commands

Check Commands

check_mv

Checks for multi-voltage domain errors.

Usage

check_mv [-items <list_of_items>] [-verbose]

Arguments

- -items '{'<list_of_items>'}'

  Specifies a list of items to check. The list must be enclosed in {} brackets. Valid values are no_power_domain, no_top_power_domain, no_region_for_domain, non_uniform_voltage_region, inst_outside_region, mv_timing_err, and unconnected_iso. The default behavior is to list all items.

- -verbose

  Specifies that all warning and error messages associated with the checking are shown. You can specify the limit for the number of times that a specific message is output by setting the message_suppress_limit parameter.

Examples

Example

```
% check_mv
Report Check Multi-Voltage:
--+--------------------------+------+--------+------+------------------
  |Item                      |Errors|Warnings|Status|Description
--+--------------------------+------+--------+------+------------------
1 |no_power_domain           |     0|       0|Passed|No power domain ...
2 |no_top_power_domain       |     0|       0|Passed|No top/default ...
3 |no_region_for_domain      |     0|       1|Passed|No voltage ...
4 |non_uniform_voltage_region|     0|       0|Passed|Voltage region ...
5 |inst_outside_region       |     0|       0|Passed|Cell placed ...
6 |mv_timing_err             |     0|       0|NA    |No library cell ...
7 |unconnected_iso           |     0|       0|Passed|Unconnected or ...
--+--------------------------+------+--------+------+------------------
```

Example

```
% check_mv -items {inst_outside_region}
Report Check Multi-Voltage:
-----+------------------+------+--------+------+-----------------------------------
     |Item              |Errors|Warnings|Status|Description
-----+------------------+------+--------+------+-----------------------------------
1    |inst_outside_region|     0|       0|Passed|Cell placed outside of assigned region
-----+------------------+------+--------+------+-----------------------------------
```

Related Topics

Low Power Commands

check_netlist

Checks the netlist for multi-driven and un-driven nets.

Usage

check_netlist [-items <list_of_nets>] [-verbose]

Arguments

- -items <list_of_nets>

  Specifies a list of nets to check. The default is all nets.

- -verbose

  Specifies that all warning and error messages associated with the checking are shown. You can specify the limit for the number of times that a specific message is generated by setting the message_suppress_limit parameter.

Examples

Example

```
% check_netlist

Report Check Netlist:
-+-----------------------+------+--------+------+-----------------------
 |Item                   |Errors|Warnings|Status|Description
-+-----------------------+------+--------+------+-----------------------
1|bidir_ports            |  0   |   4    |Passed|Top level bidir ports
2|non_driving_ports      |  0   |   3    |Passed|Top level input ports
 |                       |      |        |      | not driving a load
3|undriven_ports         |  0   |   1    |Passed|Undriven top level
 |                       |      |        |      | output ports
4|hierarchical_bidir_pins|  0   |   32   |Passed|Hierarchical bidir pins
5|non_driving_instances  |  0   |   896  |Passed|Instances not driving
 |                       |      |        |      | anything
6|undriven_pins          |  0   |   1    |Passed|Undriven instance input
 |                       |      |        |      | pins
7|multi_driven_nets      |  0   |   21   |Passed|Multi driven nets
8|wrong_port_direction   |  0   |   8    |Passed|Wrong port direction
-+-----------------------+------+--------+------+-----------------------
```

Example

```
% check_netlist -verbose
```

Related Topics

read_verilog

read_vhdl

check_placement

Checks for illegally placed macros and outputs the results.

Usage

check_placement [-include_placement_blockage { true | false }] [-items <list_of_items>] [-verbose]

Arguments

- -include_placement_blockage { true | false }

  Specifies that overlaps with blockages be reported. Reported blockages include: macros overlapping with blockages, and pads overlapping with blockages. The default behavior is to not report overlaps.

- -items <list_of_items>

  Specifies a list of items to check. Valid values for the -items argument include macro_unplaced, macro_overlap, macro_out_of_bound, and cells_overlap. The default is all items.

- -verbose

  Specifies that all warning and error messages associated with the checking are shown. You can specify the limit for the number of times that a specific message is generated by setting the message_suppress_limit parameter.

Description

Checks and reports placement errors such as macro and halo overlaps, and illegal standard cell locations.

Examples

Example 1: Check Placement Using the Default Output Format

This command checks for placement errors.

```
[oasys-RTL]$ check_placement
Report Check Placement:
-----+------------------+------+--------+------+-----------------
     |Item              |Errors|Warnings|Status|Description
-----+------------------+------+--------+------+-----------------
1    |macro_unplaced    |     0|       0|Passed|Macro unplaced
2    |macro_overlap     |     8|       0|Failed|Macro overlap
3    |macro_out_of_bound|     1|       0|Failed|Macro out of bound
4    |cells_overlap     |     7|       0|Failed|Overlapping cells
-----+------------------+------+--------+------+-----------------
```

Example 2: Check Placement Using the -verbose Option

This example checks for placement errors with the -verbose option.

```
[oasys-RTL]$ check_placement -verbose
Detected 8 macros overlapped with other macros or blockage

error:   macro_overlap : 4   [PLACE-149]
error:     macro: us_core/inst/ctxram0/RF at (-200000, 13740000)  [PLACE-150]
error:     macro: us_core/ofifo/ram_a/RF at (100000, 14026670)  [PLACE-150]
error:     macro: us_core/ofifo/ram_r/RF at (100000, 17496670)  [PLACE-150]
error:     macro: us_core/ofifo/ram_w/RF at (6083500, 14026600)  [PLACE-150]

error:   macro_overlap : 2   [PLACE-149]
error:     macro: us_core/inst/iram_addr_alpha/RF at (0, 63834000)  [PLACE-150]
error:     macro: us_core/inst/iram_inst_cmn/RF at (100000, 59823210)  [PLACE-150]

error:   macro_overlap : 2   [PLACE-149]
error:     macro: pipe2/alub/data_ramb/RF at (7893800, 62365300)  [PLACE-150] error:
macro: pipe2/aluc/data_rama/RF at (8023200, 64537510)  [PLACE-150]

Detected 1 macros placed out of bound

error:   macro_out_of_bound : 1   [PLACE-151]
error:     macro: us_core/inst/ctxram0/RF at (-200000, 13740000)  [PLACE-152]

error:   placement conflict with instance us_core/td_fifo/data_out_reg[510] at (35691599,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[509] at (35708399,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[508] at (35708399,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[504] at (35708399,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[502] at (35708399,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[501] at (35708399,
41806798)  [PLACE-140]
error:   placement conflict with instance us_core/td_fifo/data_out_reg[492] at (35708399,
41806798)  [PLACE-140]
error:   7 placement overlaps are detected.  [PLACE-142]
Report Check Placement:
-----+------------------+------+--------+------+------------------
     |Item              |Errors|Warnings|Status|Description
-----+------------------+------+--------+------+------------------
1    |macro_unplaced    |     0|       0|Passed|Macro unplaced
2    |macro_overlap     |     8|       0|Failed|Macro overlap
3    |macro_out_of_bound|     1|       0|Failed|Macro out of bound
4    |cells_overlap     |     7|       0|Failed|Overlapping cells
-----+------------------+------+--------+------+------------------
```

Related Topics

Check Commands

check_timing

Checks for any timing problems with the design.

Usage

check_timing [-items <list_of_items>] [-verbose]

Arguments

- -items <list_of_items>

  Optional. Specifies a list of scenarios to check. The default is all items.

  The following are valid values for <list_of_items>:

  - clock_pin_with_data
  - clock_pin_with_multiple_clocks
  - clock_pin_without_clock
  - setup_pin_with_clock
  - setup_pin_without_data
  - trigger_pin_without_required
  - unconstrained_IO
  - unexpected_assertion

- -verbose

  Optional. Shows all warning and error messages from the check. You can limit the number of times that a specific message is shown by setting the message_suppress_limit parameter.

Description

The check_timing command checks for each top level port or sequential element in the following timing scenarios:

- **Clock pin has data signal arriving —** Clock pins that are driven by data.
- **Clock pin has multiple clock signals —** Clock pins that are driven by multiple clocks. This may be valid if all clocks are defined to be exclusive.
- **Clock pin does not have clock signal —** Clocks pins that do not have valid timing paths from a clock definition.
- **Setup pin has clock signal arriving —** Data pins that are driven by a clock signal. This may be valid for clock divider logic.
- **Setup pin does not get arriving data —** Flip-flops that do not have a valid timing path from a starting point.

- **Trigger pin does not get required data —** Flip-flops that do not have a valid timing path from the Q pin to an end-point. The flip-flop launches data but it is never captured.
- **Unconstrained IO pin —** Inputs that do not have an input delay timing constraint and outputs that do not have an output delay timing constraint.
- **Found unexpected timing assertion —** Tool internal error.

Examples

Example 1: Default Output Format

This example shows the default output format of the check_timing command.

```
[oasys-RTL]$ check_timing
Report Check Timing:
--+------------------------------+----+------+------+-----------------------------------
  |Item                          |Er- |Warn- |Status|Description
  |                              |rors|ings  |      |
--+------------------------------+----+------+------+-----------------------------------
1 |unconstrained_IO              |   0|     8|Passed|Unconstrained IO pin
2 |unexpected_assertion          |   0|     0|Passed|Found unexpected timing assertion
3 |trigger_pin_without_required  |   0|  6804|Passed|Trigger pin does not get required data
4 |setup_pin_without_data        |   0| 14680|Passed|Setup pin does not get arriving data
5 |setup_pin_with_clock          |   0|     0|Passed|Setup pin has clock signal arriving
6 |clock_pin_with_multiple_clocks|   0|     0|Passed|Clock pin has multiple clock signals
7 |clock_pin_without_clock       |   0|     0|Passed|Clock pin does not have clock signal
8 |clock_pin_with_data           |   0|   172|Passed|Clock pin has data signal arriving
--+------------------------------+----+------+------+-----------------------------------
```

Example 2: Check I/O Timing Constraints and Clock Pins with Multiple Clocks

This example reports inputs and outputs without timing constraints as well as clock pins that are driven by multiple clocks.

```
[oasys-RTL]$ check_timing -items { unconstrained_IO clock_pin_with_multiple_clocks }
Report Check Timing:
--+------------------------------+----+------+------+-----------------------------------
  |Item                          |Er- |Warn- |Status|Description
  |                              |rors|ings  |      |
--+------------------------------+----+------+------+-----------------------------------
1 |unconstrained_IO              |   0|     8|Passed|Unconstrained IO pin
2 |clock_pin_with_multiple_clocks|   0|     0|Passed|Clock pin has multiple clock signals
--+------------------------------+----+------+------+-----------------------------------
```

Example 3: Check I/O Timing Constraints and Clock Pins with Multiple Clocks with the -verbose Option

This example reports inputs and outputs without timing constraints as well as clock pins that are driven by multiple clocks. It also prints the associated warning messages.

```
[oasys-RTL]$ check_timing -items { unconstrained_IO clock_pin_with_multiple_clocks }
-verbose
info:    No external delay asserted at input pin 'TEST_MODE'  [TA-904]
info:    No external delay asserted at input pin 'SI_0'  [TA-904]
info:    No external delay asserted at input pin 'SI_1'  [TA-904]
info:    No external delay asserted at input pin 'SI_2'  [TA-904]
info:    No external delay asserted at input pin 'SI_3'  [TA-904]
info:    No external delay asserted at input pin 'SI_4'  [TA-904]
info:    No external delay asserted at input pin 'SI_5'  [TA-904]
info:    No external delay asserted at input pin 'SI_6'  [TA-904]
Report Check Timing:
--+------------------------------+----+------+------+-----------------------------------
  |Item                          |Er- |Warn- |Status|Description
  |                              |rors|ings  |      |
--+------------------------------+----+------+------+-----------------------------------
1 |unconstrained_IO              |   0|     8|Passed|Unconstrained IO pin
2 |clock_pin_with_multiple_clocks|   0|     0|Passed|Clock pin has multiple clock signals
--+------------------------------+----+------+------+-----------------------------------
```

Related Topics

Check Commands

Timing Commands

Chapter 7
Write Commands

The write commands generate output from the Oasys-RTL tool.

Table 7-1. Write Commands

Command	Description
save_upf	Saves UPF commands in the design to a file. By default, Oasys saves a hierarchical UPF file.
write_ctl	Writes out the abstract DFT model for the current top design in IEEE 1450.6 CTL format. It captures information on scan chains, tests clocks, and tests pins for the design.
write_db	Saves a snapshot of the design currently loaded in the Oasys-RTL session. The snapshot may optionally include library data in binary format.
write_def	Writes out the physical information for the design in DEF format.
write_explore_checkpoint	Saves a report of all relevant metrics and an Oasys-RTL database for each session at the specified point in the design space exploration flow.
write_optimized_registers	Exports a file with a list of registers that are optimized.
write_saif	Writes out the dynamic power information to a switching activity interchange format (SAIF) file.
write_scandef	Writes out chain elements in scandef format to reorder after physical synthesis.
write_sdc	Writes out the design constraints information in the SDC format.
write_stil	Writes out DFT information in the Standard Test Interface Language (STIL) format.
write_verilog	Outputs the design netlist to a Verilog file.

save_upf

Saves UPF commands in the design to a file. By default, Oasys saves a hierarchical UPF file.

Usage

save_upf [-scope <instance_name>] [-version <version>] [-hierarchical <true|false>] <filename>

Arguments

- -scope <instance_name>

  Specifies the scope of the design to save.

- -version <version>

  Specifies the UPF version. 1.0 and 2.0 are supported.

- -hierarchical <true|false>

  Saves as a hierarchical UPF.

- <filename>

  Specifies the file name to which the UPF commands are saved.

Examples

```
% save_upf designx.upf -version 2.0
```

Related Topics

Low Power Commands

Write Commands

write_ctl

Writes out the abstract DFT model for the current top design in IEEE 1450.6 CTL format. It captures information on scan chains, tests clocks, and tests pins for the design.

Usage

write_ctl [-detail] <filename>

Arguments

- -detail

  Specifies that a list of all scan cells to output.

- <filename>

  Specifies the file name to which the CTL commands are saved.

Examples

```
% write_ctl ctl_file.ctl
```

Related Topics

Write Commands

write_db

Saves a snapshot of the design currently loaded in the Oasys-RTL session. The snapshot may optionally include library data in binary format.

Usage

write_db <filename> [-data [library | design | all]] [-comment <string>]

Arguments

- <filename>

  Required. Specifies the filename to which the design snapshot is saved. By default, the Oasys-RTL tool adds a *".odb"* extension to the filename. If the specified file already exists, it will be overwritten.

- -data [design | library | all]

  Optional. Specifies the type of database to be saved. Details are provided in the description below.

  design

  Writes out the database for the design only. No library information is included.

  library

  Writes out the library database only. No design data is included.

  all

  This is the default. Writes out an integrated database that includes both design and library data.

- -comment <string>

  Optional. Provides the ability to write a comment with the saved ODB.

Description

The write_db command provides the capability to save an integrated database that includes both the design data and the technology libraries (timing (LIB), physical (LEF and PTF), and so on), or a database with the design data only, or a standalone library database with just the libraries in a binary format. The command can be used during any stage in the synthesis session.

The output file will have a default suffix of *.odb*; therefore, if separate databases are generated for libraries and designs, it is recommended to name the databases to distinguish their types (for example, design, library, or all, as shown in the examples section). All the library attributes (such as set_dont_use, set_dont_touch, and so on) that are set on the library cells during the session will be saved in the database.

The write_db command will check the completeness of the imported physical libraries. If the check_library command reports any failures, then the write_db command will not write out the database, and will error out.

You can use the read_db command to read back any of the database *.odb* files generated from the Oasys-RTL tool.

Examples

Exporting a Design DB With and Without Libaries

For example, consider a design *Top* and library *MyLib.lib*, *MyLib.lef*, and so on. Either of the following commands generates the design database in the file *Top.odb*:

```
write_db -data design Top.odb
write_db -data design Top
```

Note that if the *.odb* extension is not provided, the tool adds it.

The following command generates the library database as the file *Libs4Top.odb*:

```
write_db -data library Libs4Top
```

Either of the following commands generates the integrated design and library database in *TopNLib.odb*:

```
write_db -data all TopNLib
write_db TopNLib
```

In the last command, the "-data all" option is assumed by default.

Synthesize and Write a Design

This example synthesizes the top module and then writes the design to the *saved_design.odb* file with the comment "Example 2".

```
synthesize -module ${top_module} -map_to_scan -gate_clock
write_db -comment "Example 2" saved_design
```

Trying to Write Out an Incomplete Imported Library

The write_db command errors out when trying to write out an incomplete imported library.

```
[oasys-RTL]$ write_db -data lib lib.db
-------> Message [NL-130] suppressed 14 times
error: no usable 2-input basic gate found in target library 'default'
[NL-102]
note:  the above message has more detailed information, see "message NL-
102"
error: 1 error message(s) issued while executing command 'write_db'
Invalid Library. Cannot write odb.
```

Related Topics

Write Commands

read_db

write_def

Writes out the physical information for the design in DEF format.

Usage

write_def [-floorplan] [-skip <objects>] [-sections <objects>] [-module <name>] [-fix_all_pins] <filename>

Arguments

- -floorplan

  Writes only floorplan information, including die area, pin placement, macro placement, rows, and blockages.

- -skip <objects>

  Specifies which objects to skip when writing to the DEF. Objects can be one or more of the following: blockages, components, gcellgrid, groups, nets, pins, regions, rows, specialnets, tracks, or vias.

- -sections <objects>

  Specifies the only objects to write to the DEF. Objects can be one or more of the following: blockages, components, gcellgrid, groups, nets, pins, regions, rows, specialnets, tracks, or vias. If the -skip option is used with the -sections option for the same object, the -skip option has priority and the -sections option is ignored.

- -module <name>

  Specifies that the top level to be written to the DEF is the specified module instead of the current top level.

- -fix_all_pins

  Specifies that all pins in the DEF file be marked as FIXED. This prevents downstream tools from moving the pins.

- <filename>

  Specifies the file name of the generated DEF file. Existing files are overwritten.

Examples

```
% write_def mydef.def
```

Related Topics

Write Commands

read_def

write_explore_checkpoint

Saves a report of all relevant metrics and an Oasys-RTL database for each session at the specified point in the design space exploration flow.

Usage

write_explore_checkpoint [-no_odb] <checkpoint>

Arguments

- -no_odb

  Specifies that a *.odb* file is not saved at the checkpoint. The default is to save it.

- <checkpoint>

  The name of the checkpoint.

Description

Saves a report of all relevant metrics and a *.odb* for each session at the specified point in the flow. There is already a built-in checkpoint at the end of the synthesize and optimize commands.

Examples

This example specifies a checkpoint. This should be placed at an interesting point in your Oasys-RTL script, other than post-synthesis or post-optimize.

```
write_explore_checkpoint check1
```

Related Topics

Write Commands

set_explore

scale_clock

scale_utilization

explore

synthesize

optimize

write_optimized_registers

Exports a file with a list of registers that are optimized.

Usage

write_optimized_registers <string>

Arguments

- <string>

  Specifies the name of the file to export the list of optimized registers.

Description

The write_optimized_registers command exports a detailed list of all design registers that are optimized as a part of being constant or are merged during the flow.

Examples

```
[oasys-RTL]% write_optimized_registers opt_regs.rpt
```

Related Topics

Write Commands

merge_flipflops

merge_latches

preserve_constant_flipflops

preserve_constant_latches

write_saif

Writes out the dynamic power information to a switching activity interchange format (SAIF) file.

Usage

write_saif [filename] [-duration [integer]] [-all]

Arguments

- filename

  Specifies the name of the output SAIF file. The tool overwrites existing files.

- -duration [integer]

  Specifies the simulation time duration for which the switching activity is captured.

- -all

  Specifies that the tool write the switching activity information for all pins and ports.

Examples

This example writes the dynamic power information for all pins and ports to a file named, *design.saif*.

```
% write_saif -all design.saif
```

write_scandef

Writes out chain elements in scandef format to reorder after physical synthesis.

Usage

write_scandef <file_name>

Arguments

- <file_name>

  Specifies the name of the file to which the scandef format is written.

Description

Writes out chain elements in scandef format to reorder after physical synthesis. Each chain is broken as necessary to prevent physical synthesis from moving flops across lockup-latches and clock-domain or edge crossings in a chain. This command should be run after the connect_scan_chains command.

Supports serial multi-bit registers. Multi-bit register instance names include a “bits n” statement.

Examples

```
% write_scandef top.scandef
```

Related Topics

Write Commands

connect_scan_chains

DFT Commands

write_sdc

Writes out the design constraints information in the SDC format.

Usage

write_sdc [-mode <mode>] <file_name>

Arguments

- -mode <mode>

  Writes the constraints from the specified timing mode. If you do not specify this argument, the command writes the constraints into separate files for each timing mode.

- <file_name>

  Specifies the name of the file to which the SDC commands are written. If the file already exists, its contents are overwritten.

Description

Generates the SDC information for the design constraints that you have set.

You may have set design constraints by using the read_sdc command and the source command for timing constraints. Some of these commands and options may be ignored by the timer because they are not supported by the tool or are not relevant to physical synthesis. The options that are not used by the tool are still parsed, checked for correct syntax, and stored in the database so that they are written out with subsequent write_sdc commands.

Examples

Example 1

In the following example, the design contains two timing modes: fast and normal. Oasys creates a separate file for each timing mode.

```
% write_sdc top
info: writing Sdc file 'top_fast.sdc' for design 'demo_chip' [WRITE-104]
info: writing Sdc file 'top_normal.sdc' for design 'demo_chip' [WRITE-104]
0
```

Related Topics

Write Commands

read_sdc

set_timing_mode

write_stil

Writes out DFT information in the Standard Test Interface Language (STIL) format.

Usage

write_stil [-compression] [-skip_io] <filename>

Arguments

- -compression

  Specifies that the scan chains should be output in DFT compression mode.
- -skip_io

  Specifies to skip the inputs and outputs.
- <filename>

  Specifies the name of the file to which the file is saved.

Examples

```
% write_stil stil_file.stil
```

Related Topics

Write Commands

write_verilog

Outputs the design netlist to a Verilog file.

Usage

write_verilog [-no_hierarchical] [-module <module>] <filename> [-no_inferred_modules]

Arguments

- -no_hierarchical

  Writes only the specified module and does not write sub-modules. By default all the sub-modules are written to the <filename> output.

- -module <module>

  Writes the netlist starting from <module>. By default, the top module is written.

- -no_inferred_modules

  By default, Oasys-RTL writes the datapath inferred modules to the verilog netlist. If this option is specified, the design will be written without the hierarchy of the datapath inferred modules. The modules will still exist in the design but, in the netlist, these inferred modules will appear as flattened hierarchy.

- <filename>

  Specifies the name of the file to which the verilog netlist is written.

Examples

Example

This example writes out a Verilog file called *myverilog.v*, including all hierarchy in the top level module.

```
% write_verilog myverilog.v
```

Example

This example writes out a Verilog file, but does not write out submodules:

```
% write_verilog -no_hierarchical myverilog.v
```

Example

This example writes out a Verilog file starting from module1:

```
% write_verilog -module module1 myverilog.v
```

Related Topics

Write Commands

read_verilog

Chapter 8
Design Editing and Optimization Control Commands

The design editing commands modify properties of the library and RTL components in the design.

Table 8-1. Design Editing and Optimization Control Commands

Command	Description
Design Edit Commands	These commands edit the design objects.
Design Object Access Commands	These Oasys-RTL commands provide an interface to retrieve objects and information from a loaded design.
Library Database Commands	These commands access library cells and their attributes.
Optimization Control Commands	These commands direct the synthesize and optimize commands during optimization.

Design Edit Commands

These commands edit the design objects.

Table 8-2. Design Edit Commands

Command	Description
change_link	Provides the ability to reference a library cell instance to a different library cell.
change_names	Changes the names of cells, ports, and nets in a design.
connect_net	Connects the specified net to the specified pins or ports.
connect_pin	Connects the specified set of pins or ports together. You can only specify a single driving pin, but multiple load pins that span hierarchies are allowed.
create_cell	Creates cells in the current design or its subdesigns.
create_net	Creates signal nets in the current design or its subdesign. Only scalar nets can be created with this command.
create_port	Creates ports in the current design or its subdesign. Only scalar ports can be created with this command.
delete_design	Removes all design information from the database.
disconnect_net	Disconnects a net in the current design from its pins or ports.
group	Creates a new level of hierarchy.
insert_buffer	Inserts a buffer or a pair of inverters at a specified location in the netlist.
remove_buffer	Removes a specified buffer from the netlist.
remove_cell	Removes cells in the current design or its subdesigns.
ungroup	Removes and flattens logic within the specified set of hierarchical instances.
uniquify	Makes unique copies of modules that have more than one instance in the design.

change_link

Provides the ability to reference a library cell instance to a different library cell.

Usage

change_link <instance_list> <reference> [-force] [-verbose]

Arguments

- <instance_list>

  Required. Specifies a list of instances of a library cell.

- <reference>

  Required. Specifies the library cell that the library instance is referenced to.

- -force

  Optional. Forces the Oasys-RTL tool to link the library cell instance to a new library cell even if the number of pins does not match.

- -verbose

  Optional. Echoes information about what was set.

Examples

Example 1: Change Link of an Instance to a Different Library Cell

This example links the library cell inst1 to the library cell AND2.

```
% change_link inst1 AND2
```

Example 2: Force Change of Link Despite Pin Mismatch

This example links the library cell inst1 to the library cell AND4 even if the number of pins is different.

```
% change_link inst1 AND4 -force
```

Related Topics

General Commands

change_names

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

Usage

change_names [-rules <name_rule_list>] [-hierarchy] [-instance <instance>]
[-new_name <name>]

Arguments

- -rules <name_rule_list>

  Specifies a name rule set that details the rules for modifying names. The <name_rule_list> file is defined by using the define_name_rules command. The default is DEFAULT_NAME_RULES.

- -hierarchy

  Specifies that all names in the design hierarchy are to be modified. By default only those objects in the current hierarchy are changed.

- -instance <instance>

  Specifies an instance in the design to change. The full path to the instance must be specified. This option requires the option -new_name.

- -new_name <name>

  Specifies the new name for the object provided in the -instance option. The full path of the instance is not required.

Description

The change_names command changes the names of cells, ports, and nets in a design. The rules that are used, are defined by the define_name_rules command.

Examples

Example

This example converts names to lower case for the whole hierarchy of the design:

```
% define_name_rules TOLOWER -allowed {a-z 0-9 _ [ ]}
% change_names -rules TOLOWER -hierarchy
```

Example

This example converts ports and nets that are busses to individual bits (bit blasting):

```
% define_name_rules BITBLAST -remove_port_bus \
      -remove_internal_net_bus
% change_names -rules BITBLAST -hierarchy
```

Example

This example changes the name of an instance in the u_decoder module.

```
% change_names -instance u_decoder/u_counter/u_inst1 -new_name u_inst2
```

Example

This example removes square brackets, [], from only cells in the netlist.

```
% define_name_rules no_brackets -type cell -allowed { a-z A-Z 0-9 _ }
% change_names -rules no_brackets -hierarchy
```

Related Topics

Design Editing and Optimization Control Commands

define_name_rules

report_name_rules

connect_net

Connects the specified net to the specified pins or ports.

Usage

connect_net [-verbose] <net> <pin_port_list>

Arguments

- -verbose

  Echoes details of the connected net.

- <net>

  Specifies the net that to be connected.

- <pin_port_list>

  Specifies a list of pins and ports to which the net is to be connected.

Examples

This example shows how to connect net1 to port1:

```
%connect_net net1 port1
```

Related Topics

Design Editing and Optimization Control Commands

disconnect_net

connect_pin

Connects the specified set of pins or ports together. You can only specify a single driving pin, but multiple load pins that span hierarchies are allowed.

Usage

connect_pin -from <pin_name> -to <pin_list> [-verbose] [-invert] [-in_prefix <prefix>] [-out_prefix <prefix>] [-port_name <port_basename>]

Arguments

- -from <pin_name>

  Specifies a pin or port from which to connect at any level of hierarchy. The direction of the pin or port cannot be inout.

- -to <pin_list>

  Specifies a list of pins and ports to which a connection is made. Pins and ports can be at any level of hierarchy. The direction of the pins or ports cannot be inout.

- -verbose

  Echoes details of the pin that was connected.

- -invert

  Specifies to add an inverter when connecting the pin.

- -in_prefix <prefix>

  Specifies a prefix for a punched input pin that is needed to make the connection.

- -out_prefix <prefix>

  Specifies a prefix for a punched output pin that is needed to make the connection.

- -port_name <port_basename>

  Specifies the base name to use as names of ports when a new port is created on subdesigns to make the connection.

Description

Connects the pins or ports in a design. This command makes global connections between the from object and the to object list. The pins or ports can be at any level of hierarchy. Do not specify inout pins.

Examples

This example connects pin A1 to pins A/a1 and A/a2:

```
% connect_pin -from A1 -to {A/a1 A/a2}
```

Related Topics

Design Editing and Optimization Control Commands

all_connected

create_cell

Creates cells in the current design or its subdesigns.

Usage

create_cell [-verbose] <cell_list> <reference_name>

Arguments

- -verbose

  Echoes details of the cell that was created.

- <cell_list>

  Specifies the names of cells created in the current design. Each cell name must be unique within the current design.

- <reference_name>

  Specifies the design or library cell that new cells reference. Ports on the reference determine the name, number, and direction of pins on the new cell.

Description

Creates new cells in the current design or its subdesigns. New cells are the instantiation of an existing design or library cell.

Examples

Example

The following creates a cell called *cellx* in subdesign *blocka* using a library cell as a reference.

```
% create_cell {blocka/cellx} lib1/NAND2
```

Example

The following creates cells *U1* and *U2* using a library cell as a reference.

```
% create_cell {U1 U2} lib1/NAND2
```

Related Topics

Design Editing and Optimization Control Commands

create_net

Creates signal nets in the current design or its subdesign. Only scalar nets can be created with this command.

Usage

create_net [-verbose] <net_list>

Arguments

- -verbose

  Echoes details of the net.

- <net_list>

  Specifies the names of the created nets. The net name must be unique within the current design or the subdesign where it is created.

Examples

Example

This command creates signal nets *net1* and *net2*.

```
% create_net {net1, net2}
```

Example

This command creates signal nets *net1* and *net2* in *blocka*.

```
% create_net {blocka/net1, blocka/net2}
```

Related Topics

Design Editing and Optimization Control Commands

create_port

Creates ports in the current design or its subdesign. Only scalar ports can be created with this command.

Usage

create_port -direction {in | out | inout} [-verbose] <port_list>

Arguments

- -direction {in | out | inout}

  Specifies the signal flow direction of the created ports. The default is in.

- -verbose

  Echoes details of the port that was created.

- <port_list>

  Specifies names of ports created in the current design. Each port name must be unique within the current design.

Examples

Example

This command creates two input ports, P1 and P2.

```
% create_port -direction "in" {P1 P2}
```

Example

This command creates a port P3 and then echoes what was done.

```
% create_port -verbose {P3}
Created port 'P3' in design 'OpenSPARCT1'
```

Related Topics

Design Editing and Optimization Control Commands

delete_design

Removes all design information from the database.

Usage

delete_design

Arguments

None.

Description

Deletes the current design (netlist, SDC constraints, UPF database, floorplan and placement) from the database. LEF and Liberty libraries and parameters remain in the database.

Examples

The following example deletes the design after synthesizing it:

```
% read_verilog foo.v
% synthesize
% delete_design
% read_verilog bar.v
% synthesize
```

Related Topics

General Commands

disconnect_net

Disconnects a net in the current design from its pins or ports.

Usage

disconnect_net [-verbose] <net> <pin_port_list> [-all]

Arguments

- -verbose

  Echoes details of the net that is disconnected.

- <net>

  Specifies the net in the current design to be disconnected.

- <pin_port_list>

  Specifies a list of pins or ports to which are disconnected from the net.

- -all

  Specifies that all pins or ports are disconnected from the specified net.

Description

Disconnects a net in the current design from its pins or ports. Only supports scalar nets; you can specify a bit of a bus, but not a bus name.

Examples

This example shows how to disconnect netA from all its pins and ports:

```
% disconnect_net netA -all
```

Related Topics

Design Editing and Optimization Control Commands

connect_net

group

Creates a new level of hierarchy.

Usage

group -design_name <design_name> -instance_name <inst_name> <cell_list>

Arguments

- -design_name <design_name>

  Names the design containing the new level of hierarchy. This name cannot already exist in the current design.

- -instance_name <inst_name>

  Names the instance of the new design.

- <cell_list>

  Specifies the list of cells grouped together to create another level of hierarchy.

Examples

In this example, the command is specified to create an instance called I2 of a design called new, created from the cells d1 and d2:

```
% group {d1 d2} -design_name new -instance_name I2
```

Related Topics

Design Editing and Optimization Control Commands

ungroup

insert_buffer

Inserts a buffer or a pair of inverters at a specified location in the netlist.

Usage

insert_buffer <lib_cell> <object_list> [-inverter_pair]
[-new_cell_names <cell_names>] [-new_net_names <net_name>] [-verbose]

Arguments

- <lib_cell>

  Required. Specifies the library cell to insert on the net. The cell must be a buffer or inverter.

- <object_list>

  Required. Specifies a net using one of two methods of identification:

  - a single net or a single driving pin. The tool connects all existing loading pins of the net or driving pin to the output of the inserted buffer.
  - a collection of loading pins that are all connected to the net. These pins must connect to the output of the inserted buffer.

- -inverter_pair

  Optional. Inserts a pair of inverters instead of a single buffer.

- -new_cell_names <cell_names>

  Optional. Specifies the name of the new buffer or names of the new inverter pair cells.

- -new_net_names <net_name>

  Optional. Specifies the name of the new net that is connected to the output of the inserted buffer. The input net of the buffer is connected to the existing net.

  If the new net is connected to a module port, the tool does not apply the net name to the new net. Instead, the tool applies the net name to the old net provided that the old net does not have a user-specified name and is no longer connected to any other module port.

- -verbose

  Optional. Prints the atomic netlist changes as the tool makes them. The following are examples of the printed messages:

  - “Created cell ...”
  - “Connected pin ... to net ...”

Description

Use this command to insert an isolation buffer, for example:

- To isolate critical logic from non-critical logic.
- To isolate logic from an I/O pad.

- To fix a net loading issue.
- To prevent synthesis from removing a net.

This command does not insert a buffer on a multi-driven net if doing so puts one or more of the drivers (or bidirectional drivers) on the output side of the buffer. To insert a buffer on a multi-driven net, provide an explicit list of loading (non-bidirectional) pins/ports.

To insert multiple buffers on a net, use multiple instances of this command.

Note

The tool keeps a link from the netlist to the RTL throughout the optimization process. Breaking the link between the netlist and the RTL inhibits the optimization process from obtaining optimal QoR. In the unlikely case that the insertion of a buffer leads to breaking the link, the tool does not insert the buffer and returns a Tcl error. To bypass this restriction, use low-level edit commands such as the create_cell and connect_net commands.

Examples

The following example inserts a BUFX2 cell.

```
[oasys-RTL]$ insert_buffer -verbose [lappend lPins [get_pin mod/z]] BUFX2
```

remove_buffer

Removes a specified buffer from the netlist.

Usage

remove_buffer [-verbose] <buffer>

Arguments

- -verbose

  Optional. Prints the names of the pins that the tool disconnects and connects.

- <buffer>

  Required. Specifies the hierarchical instance name of a buffer in the design to remove.

Examples

Example 1: Remove a Buffer Using the -verbose Option

The following example removes the *ifu/ifqdp/rt_shieldBuf__21* buffer using the remove_buffer -verbose option. The tool prints the names of the pins that it disconnects or connects.

```
[oasys-RTL]$ remove_buffer ifu/ifqdp/rt_shieldBuf__21 -verbose
>Disconnecting pin 'ifu/ifqdp/rt_shieldBuf__21/Z'
>Connecting pin 'ifu/ifqdp/ifq_bypmux/iN0[143]' to net 'ifu/ifqdp/
ifd_iNv_ifqop_i2[143]'
>Disconnecting pin 'ifu/ifqdp/rt_shieldBuf__21/A'
>Deleting buffer 'ifu/ifqdp/rt_shieldBuf__21
```

Related Topics

Design Editing and Optimization Control Commands

remove_cell

remove_cell

Removes cells in the current design or its subdesigns.

Usage

remove_cell [-verbose] <cell_list>

Arguments

- -verbose

  Echoes details of the cell that was removed.

- <cell_list>

  Specifies the names of cells to remove in the current design.

Examples

Example

The following removes a cell called *cellx* in subdesign *blocka*.

```
% remove_cell {blocka/cellx}
```

Related Topics

Design Editing and Optimization Control Commands

remove_buffer

ungroup

Removes and flattens logic within the specified set of hierarchical instances.

Usage

ungroup [-flatten] [-all] [-start_level <level>] [-small <number>] [-force] <instances>

Arguments

- -flatten

  Flattens all levels within the instances.

- -all

  Specifies to flatten the whole design.

- -start_level <level>

  Specifies to ungroups all hierarchical instances deeper than <level>.

- -small <number>

  Specifies to ungroup all levels containing less than the <number> leaf cells.

- -force

  Specifies to force an ungroup operation if there is a dont_ungroup attribute set (or dont_modify on any parent).

- <instances>

  Specifies a list of instances to be ungrouped.

Description

Use this command to dissolve certain instantiations of hierarchical modules. This can be useful to create more opportunity for optimization.

For example, if the critical path looks like this:

```
% A/B/C/i_0/i_... A/D/i_1/i_... A/D/F/i_3/i_...
```

then ungrouping A/D/F would put i_3 in the same hierarchical level as i_1 (that is, the same level as A/D) and create opportunities for optimization. On the other hand, dissolving A/B/C would not help, because there would still be a hierarchical boundary to the next instance (A/B/i_1).

To create a further opportunity for optimization, dissolve all of A/B/C, A/B, and A/D instances, resulting in i_0, i_1, and i_3 all in the same hierarchy level (that is, at level A). The goal is to create a flatter hierarchy where i_* instances are next to each other at the same hierarchy level.

By default, this command ungroups only the (hierarchical) instance(s) given as an argument. Specifying -flatten flattens the whole hierarchy below the given instance(s).

Specifying -all flatten the whole design. Specifying -start_level <level> and/or -small <size> ungroups all hierarchical instances which are deeper than <level> and contain less than the <number> of leaf cells. The command operates relative to the current_design.

Examples

The following example ungroups hierarchies A/B and A/D*:

```
% ungroup {A/B A/D*}
```

Related Topics

set_dont_ungroup

Design Editing and Optimization Control Commands

uniquify

Makes unique copies of modules that have more than one instance in the design.

Usage

uniquify [-force]

Arguments

- -force

  Optional. Uniqufies all modules in the design except the top-level design with the naming style specified by the uniquify_naming_style parameter. One of the uses for this option is to add a prefix or suffix to all modules in a synthesis partition to avoid name collisions during chip integration.

Examples

The following example uniquifies all modules with a prefix MYSPARC:

```
[oasys-RTL]$ get_designs *dp_mux2es*

{dp_mux2es__parameterized6__13 dp_mux2es__parameterized6__14
dp_mux2es__parameterized6__15 dp_mux2es__parameterized6__16
dp_mux2es__parameterized1 dp_mux2es__parameterized1__1
dp_mux2es__parameterized1__2 dp_mux2es__parameterized1__3
dp_mux2es__parameterized1__4 dp_mux2es__parameterized20 ...(42 more)}

[oasys-RTL]$ set_parameter uniquify_naming_style MYSPARC_%s__%d
[oasys-RTL]$ uniquify -force
[oasys-RTL]$ get_designs *dp_mux2es*

{MYSPARC_dp_mux2es__parameterized6__13_0
MYSPARC_dp_mux2es__parameterized6__14_0
MYSPARC_dp_mux2es__parameterized6__15_0
MYSPARC_dp_mux2es__parameterized6__16_0
MYSPARC_dp_mux2es__parameterized1_0
MYSPARC_dp_mux2es__parameterized1__1_0
MYSPARC_dp_mux2es__parameterized1__2_0
MYSPARC_dp_mux2es__parameterized1__3_0
MYSPARC_dp_mux2es__parameterized1__4_0
MYSPARC_dp_mux2es__parameterized20_0 ...(42 more)}
```

Related Topics

Design Editing and Optimization Control Commands

Design Object Access Commands

These Oasys-RTL commands provide an interface to retrieve objects and information from a loaded design.

Design object access commands that have a name beginning with *get_* have a -filter option to narrow the set of returned results based on a user-specified expression. The “filter_collection <expression>” argument works in a similar way. Oasys-RTL supports the following operators in the filter expressions:

- ==
- !=
- >
- <
- >=
- <=
- &&
- ||

For pattern matching, the tool supports the following operators:

- =~
- !~

This example returns all cells whose ref_name attribute value contains the string “DFFQX”.

```
[oasys-RTL]$ get_cells -filter ref_name=~*DFFQX*
```

This example returns all cells whose ref_name attribute value is exactly “DFFQXL”.

```
[oasys-RTL]$ get_cells -filter ref_name==DFFQXL
```

This example returns a list of all input pins on the *y_reg* block and its sub-hierarchies. It combines multiple expressions using the && (and) and || (or) operators.

```
[oasys-RTL]$ get_pins -hier -filter "full_name=~y_reg/* && direction==in"
```

The following get_pins and filter_collection commands are equivalent:

```
[oasys-RTL]$ get_pins -hier -filter "full_name=~y_reg/* && direction==in"

[oasys-RTL]$ set a [get_pins -hier]
[oasys-RTL]$ filter_collection $a {@full_name=~y_reg/* && direction==in"
```

Table 8-3. Design Access Commands

Command	Description
add_to_collection	Adds objects to a collection, resulting in a new collection.
all_connected	Returns the objects connected to a net, port, or pin.
all_designs	Returns a list of all modules in the design.

Table 8-3. Design Access Commands (cont.)

Command	Description
all_fanin	Returns a list of pins, ports, or cells in the fanin of specified sinks.
all_fanout	Returns a list of pins, ports, or cells in the fanout of specified sources.
append_to_collection	Appends objects to the specified collection.
compare_collections	Compares the contents of two collections.
copy_collection	Duplicates a collection.
define_name_rules	Defines naming rules, restrictions, and styles, for object names that the tool generates.
define_user_attribute	Creates a user-defined attribute and specifies the properties of the attribute.
filter_collection	Returns a subset of a collection by using a regular expression to filter the collection.
foreach_in_collection	Iterates over elements in a collection.
get_attribute	Returns a value for an attribute.
get_cells	Returns a list of instances of cells in the design based on the selection criteria.
get_clocks	Returns a list of instances in the design based on the specified options.
get_design_effort	Returns the settings configured with the set_design_effort command.
get_designs	Returns a list of the specified designs.
get_lib_cells	Returns the pointer to the library cells with a specific pattern in the cell name.
get_lib_pins	Returns library pins that match a specified pattern.
get_libs	Returns libraries that match a specified pattern.
get_nets	Returns the specified nets in the design.
get_macros	Returns a list of all macros in the design.
get_object_names	Returns the names of specified objects. The names are strings that you can pass as arguments to commands or use in scripts.
get_operating_process	Returns the process of the current operating condition.
get_operating_temperature	Returns the operating temperature of the design.

Table 8-3. Design Access Commands (cont.)

Command	Description
get_operating_voltage	Returns the operating voltage of the design.
get_pg_info	Returns power and ground related information.
get_pins	Returns the specified pins in the design.
get_ports	Returns the specified ports in the design.
get_power_domains	Returns specified power domains.
get_references	Returns the instances of the given module or library cell name that has the specified pattern.
get_selection	Returns the names of one or more objects selected in the GUI window.
get_supply_nets	Returns power/ground supply net objects.
get_supply_ports	Returns power/ground supply port objects.
index_collection	Creates a collection of one object that is the nth object of another collection.
remove_from_collection	Removes objects from a collection creating a new collection.
sizeof_collection	Determines the size of a collection.
sort_collection	Creates a sorted collection for the specified collection.

add_to_collection

Adds objects to a collection, resulting in a new collection.

Usage

add_to_collection [-unique] <base_collection> <object_list>

Arguments

- -unique

  Specifies that if an object is already in the collection, it is not added to the specified collection.

- <base_collection>

  The name of the collection.

- <object_list>

  Specifies which items are added to the collection.

Examples

Example

This example finds cells that start with the letter n and adds them to collection1:

```
% set collection2 [add_to_collection collection1 [get_cells n*] ]
```

Example

This next example shows the difference between add_to_collection and append_to_collection:

```
% set foo {a b}
% add_to_collection $foo {c d}
{a b c d}
% puts $foo
{a b}
% append_to_collection foo {e f}
% puts $foo
{a b e f}
```

Related Topics

append_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

all_connected

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

Usage

all_connected <object> [-comment <string>] [-leaf]

Arguments

- <object>

  Specifies a pin, port, or net object.

- -comment <string>

  Specifies a string to be used for commenting the operation for tracking purposes.

- -leaf

  Returns all connected leaf pins across hierarchies. This argument can only be used if the <object> argument is a net.

Description

The all_connected command returns a list of objects connected to the specified net, port, or pin. If the object is a pin or port, the command returns the connected net. If the object is a net, it returns all connected pins. If the -leaf option is specified, it returns all pins connected across hierarchical boundaries.

Examples

This command returns all objects connected to the net named “my_net”:

```
% all_connected [get_net my_net]
```

Related Topics

all_inputs

all_outputs

connect_net

General Commands

all_designs

Returns a list of all modules in the design.

Usage

all_designs

Arguments

None.

Examples

This example lists all modules in the design:

```
% all_designs
{core_exec_unit core_ffu core_ifu lsu … }
```

Related Topics

General Commands

all_fanin

Returns a list of pins, ports, or cells in the fanin of specified sinks.

Usage

all_fanin -to <sink_list> [-startpoints_only] [-exclude_bboxes] [-break_on_bboxes] [-only_cells] [-flat] [-levels <count>] [-port_only] [-clock <clock>]

Arguments

- -to <sink_list>

  Returns a list of sink pins, ports, or nets in the design along with the timing fanin of each sink in the <sink_list>. Specifying a net is equivalent to listing all driver pins on that net.

- -startpoints_only

  Returns only the timing startpoints.

- -exclude_bboxes

  Specifies to exclude black boxes on traversal.

- -break_on_bboxes

  Specifies to break on black boxes on traversal.

- -only_cells

  Returns only the cells in the timing fanin of the <sink_list>.

- -flat

  Returns objects from any level of hierarchy. This is the default. A hierarchical mode is not supported.

- -levels <count>

  Stops traversal when reaching <count> levels of cells from the sink.

- -port_only

  Returns only the inputs ports found during the fanin traversal. The -port_only switch can only be specified with the -clock and -startpoints_only options.

- -clock <clock>

  Traces the fanin of registers fed by this clock starting from the data pins, for the specified SDC clock name.

Description

The all_fanin command reports the timing fanin of the specified sink pins, ports, or nets in the design. A pin is considered to be in the timing fanin of a sink if there is a timing path through the combinational logic from the pin to that sink. The fanin report stops at the clock pins of registers.

Examples

The following example returns the fanin of the specified pins:

```
% all_fanin -to [get_pins spc_pcx_req_pq[1]]

{spc_pcx_req_pq[1] lsu/qct11/rq_stgpq/q_reg[1] lsu/qct11/rq_stgpq/
q_reg[1]/CK}
```

Related Topics

all_fanout

General Commands

all_fanout

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

Usage

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

Arguments

- -clock_tree

  Generates a report that displays the clock trees or networks in the design. The option specifies to use all clock source pins or ports in the design as the list of sources. If there are no clocks, or if the clocks have no sources, the report is empty. The -clock_tree and -from options are mutually exclusive.

- -from <source_list>

  Returns a list of source pins, ports, or nets in the design along with the timing fanout of each source specified in <source_list>. Specifying a net is equivalent to listing all load pins on that net.

- -endpoints_only

  Returns only the timing endpoints.

- -exclude_bboxes

  Specifies to exclude black boxes on traversal.

- -break_on_bboxes

  Specifies to break on black boxes on traversal.

- -only_cells

  Returns only the cells in the timing fanout of the <source_list>.

- -flat

  Returns objects from any level of hierarchy. This is the default and a hierarchical mode is not supported.

- -levels <count>

  Stops traversal when reaching <count> levels of cells from the source.

Description

The all_fanout command reports the timing fanout of specified source pins, ports, or nets in the design. A pin is considered to be in the timing fanout of a source if there is a timing path through combinational logic from that source to the pin. The fanout report stops at the inputs to registers. The source pins or ports are specified by using the -clock_tree option or -from <source_list> option.

Examples

Example

```
% all_fanout -clock_tree
{gclk spc_hdr/I0/sync_cluster_master/q_r_reg/CK spc_hdr/I0/
sync_cluster_slave/i_1_0/A spc_hdr/I0/sync_cluster_slave/i_1_0/ZN
spc_hdr/I0/sync_cluster_slave/so_l_reg/G spc_hdr/I0/rst_repeater/lockup/
i_1_0/A spc_hdr/I0/rst_repeater/lockup/i_1_0/ZN spc_hdr/I0/rst_repeater/
lockup/so_l_reg/G spc_hdr/I0/rst_repeater/repeater/i0/q_reg/CK spc_hdr/
I0/dbginit_repeater/lockup/i_1_0/A ...(21048 more)}
```

Example

```
% all_fanout -from [get_ports *rst*]
{cmp_grst_l spc_hdr/I0/rst_repeater/repeater/i0/i_0_2/A spc_hdr/I0/
rst_repeater/repeater/i0/i_0_2/ZN spc_hdr/I0/rst_repeater/repeater/i0/
i_0_3/A1 spc_hdr/I0/rst_repeater/repeater/i0/i_0_3/ZN spc_hdr/I0/
rst_repeater/repeater/i0/q_reg/D cmp_arst_l ctu_tst_pre_grst_l i_0/
ctu_tst_pre_grst_l test_stub/scan_ctls/i_0_7/A2 ...(56838 more)}
```

Example

```
% all_fanout -from [get_ports *rst*] -levels 1
{cmp_grst_l spc_hdr/I0/rst_repeater/repeater/i0/i_0_2/A spc_hdr/I0/
rst_repeater/repeater/i0/i_0_2/ZN cmp_arst_l ctu_tst_pre_grst_l i_0/
ctu_tst_pre_grst_l test_stub/scan_ctls/i_0_7/A2 test_stub/scan_ctls/
i_0_7/ZN test_stub/scan_ctls/i_0_5/A2 test_stub/scan_ctls/i_0_5/ZN}
```

Related Topics

all_fanin

General Commands

append_to_collection

Appends objects to the specified collection.

Usage

append_to_collection [-unique] <base_collection> <object_list>

Arguments

- -unique

  Specifies that if an object is already in the collection, it is not added to the specified collection.

- <base_collection>

  The name of the collection.

- <object_list>

  Specifies the items that are appended to the collection.

Description

This command appends objects to the specified collection. This command modifies the <base_collection> by adding new members to it.

Examples

Example

This example finds cells that start with the letter n and appends them to collection1:

```
% append_to_collection collection1 [get_cells n*]
```

Example

This example shows the difference between add_to_collection and append_to_collection:

```
% set foo {a b}
% add_to_collection $foo {c d}
{a b c d}
% puts $foo
{a b}
% append_to_collection foo {e f}
% puts $foo
{a b e f}
```

Related Topics

add_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

compare_collections

Compares the contents of two collections.

Usage

compare_collections [-order_dependent] <collection1> <collection2>

Arguments

- -order_dependent

  Specifies that the comparison is done in order.

- <collection1>

  The first of the two collections that are to be compared.

- <collection2>

  The second of the two collections that are to be compared.

Description

Compares the contents of two collections, object for object and optionally, in order.

Examples

This example compares collection1 to collection2 independent of the order of the items:

```
% compare collection1 collection2
```

Related Topics

add_to_collection

append_to_collection

copy_collection

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

copy_collection

Duplicates a collection.

Usage

copy_collection <base_collection>

Arguments

- <base_collection>

  The name of the collection that is to be copied.

Examples

This example makes a copy of collection1 called collection2:

```
% set collection2 [copy_collection $collection1]
```

Related Topics

General Commands

append_to_collection

delete_design

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

add_to_collection

define_name_rules

Defines naming rules, restrictions, and styles, for object names that the tool generates.

Usage

define_name_rules [<name_rules>] [-add_dummy_nets] [-allowed <allowed_chars>]
[-case_insensitive] [-collapse_name_space] [-dont_change_ports]
[-equal_port_nets] [-first_restricted <chars>]
[-flatten_multi_dimension_busses] [-last_restricted <chars>] [-map <string>]
[-max_length <length>] [-prefix <name>] [-remove_internal_net_bus] [-remove_port_bus]
[-replacement_char <replacement_char>] [-reserved_words <object_names>]
[-restricted <chars>] [-target_bus_naming_style <string>] [-type [port | cell | net]]

Arguments

- <name_rules>

  Optional. Specifies the name of the rules being defined. If <name_rules> is not specified, the default is DEFAULT_NAME_RULES.

- -add_dummy_nets

  Optional. Adds dummy nets on unconnected pins. This is done by default.

- -allowed <allowed_chars>

  Optional. Specifies the set of characters allowed in names. By default, any printable character is allowed in a name. This option cannot be used with the -restricted option.

- -case_insensitive

  Optional. Specifies to ignore character case when comparing names. The default behavior is to perform case comparison.

- -collapse_name_space

  Optional. Combines the namespaces of different objects such as instances, ports, nets, and modules. When the namespaces are collapsed, each object must have a unique name. By default, objects have unique namespaces and different objects can have the same name.

- -dont_change_ports

  Optional. Specifies that naming rules do not apply to port names.

- -equal_port_nets

  Optional. Make the name of net connected to a port same as the port name. This is done by default.

- -first_restricted <chars>

  Optional. Specifies the set of characters that are not allowed as the first character in a name.

- -flatten_multi_dimension_busses

  Optional. Flattens multi-dimension ports and net busses. This is done by default.

Caution

If you use the formal verification flow, ensure that the resulting names match those that are checked by the verify command. Changes you make using the define_name_rules command do not propagate to the verify command.

- -last_restricted <chars>

  Optional. Specifies the set of characters that are not allowed as the last character in a name.

- -map <string>

  Optional. Specifies the replacement rules for mapping a pattern of one or more characters to a different set. The pattern matching expression can use regular expressions specifiers: caret (^) and dollar ($). The format of the string option is { { “<character pattern to be replaced”>, “<replacement character pattern>”} {…} }.

  This option is supported for cell, port, and net type objects.

- -max_length <length>

  Optional. Specifies the maximum length of a name, which should be more than 8 characters. Specifying 0 restores the default behavior, which puts no restriction on the name length.

- -prefix <name>

  Optional. Specifies a string to prefix to the names. The default prefixes are “u” for cells, “p” for ports, and “n” for nets.

- -remove_internal_net_bus

  Optional. Bit-blasts all internal net buses (converts busses to bits).

- -remove_port_bus

  Optional. Bit-blasts all port buses. Net buses that connect to ports are also bit-blasted.

- -replacement_char <replacement_char>

  Optional. Specifies the character used to replace characters not allowed in names, as specified by the -allowed option. By default, the replacement character is the underscore character (_).

- -reserved_words <object_names>

  Optional. Specifies a list of names that are not allowed as design object names.

- -restricted <chars>

  Optional. Specifies the set of characters that are not allowed in a name. By default, all characters are allowed. This option replaces a restricted character by changing its case (from upper to lower, or vice versa), if allowed. Otherwise, it replaces a restricted character with the replacement character. The replacement character is defined with the -replacement_char option. The -restricted option cannot be used with the -allowed option.

- -target_bus_naming_style <string>

  Optional. Specifies the naming style for bit when flattening ports and net busses. Currently, only {%s[%d]} style is supported.

- -type [port | cell | net]

  Optional. Specifies that the defined rules apply only to the specified objects types. The object type choices are port, cell, or net. By default, the defined rules apply to all object types.

Description

Defines a set of name rules for designs. The change_names command converts the names of cells, ports, or nets in a design, based on these rules.

If you allow lower case letters (a-z) and not uppercase letters (A-Z), uppercase letters are converted to lower case letters, when the change_names command is invoked. See the example below.

Use the -reserved_words option to prohibit specified object names from being used.

Examples

Example 1: Convert Names to Lower Case

The following example converts names to lower case for the whole hierarchy of the design:

```
[oasys-RTL]$ define_name_rules TOLOWER -allowed {a-z 0-9 _ [ ]}
[oasys-RTL]$ change_names -rules TOLOWER -hierarchy
```

Example 2: Bit Blast Port and Net Busses

The next example converts ports and nets that are busses to individual bits (bit blasting):

```
[oasys-RTL]$ define_name_rules BITBLAST-remove_port_bus \
   -remove_internal_net_bus
[oasys-RTL]$ change_names -rules BITBLAST -hierarchy
```

Example 3: Prohibit Use of Reserved Words

This example prohibits a list of object names from being used.

```
[oasys-RTL]$ define_name_rules object_names \
   -reserved_words [list vdd gnd always and assign begin]
```

Example 4: Replace Single Characters

This example performs multiple single character replacements.

```
[oasys-RTL]$ define_name_rules -map {{{"/",  "."}, {"[",  "-"}}} my
```

Example 5: Replace Multiple Characters

This example performs two multiple character replacements.

```
[oasys-RTL]$ define_name_rules ff_rule -map {{"_reg$",  "_ff"}}
[oasys-RTL]$ define_name_rules remove_XX_prefix -map {{"^XX",  ""}}
```

Related Topics

Design Editing and Optimization Control Commands

change_names

report_name_rules

define_user_attribute

Creates a user-defined attribute and specifies the properties of the attribute.

Usage

define_user_attribute <name> -type {string | int | float | double | boolean}
[-classes {design | port | pin | cell | net | lib | lib_cell | lib_pin}]
[-range_min <min_value>] [-range_max <max_value>] [-one_of <string list>] [-quiet]

Arguments

- <name>

  Specifies the name of the user defined attribute. You can only define user attributes once.

- -type {string | int | float | double | boolean}

  Specifies the type of the attribute. Only one type can be specified.

- -classes {design | port | pin | cell | net | lib | lib_cell | lib_pin}

  Specifies the class. One or more classes can be specified.

- -range_min <min_value>

  Specifies the minimum value that this attribute can have. The legal values depend on the type.

- -range_max <max_value>

  Specifies the maximum value that this attribute can have. The legal values depend on the type.

- -one_of <string_list>

  Specifies a list of possible values that the attribute can have. Requires the -type argument to have a value of string.

- -quiet

  Suppresses output of warnings.

Examples

The below example defines a user-defined attribute named, *my_attr2*, of type string, in the class design. The attribute can have one of the values abc, pqr or lmn. All warning are suppressed.

```
% define_user_attribute my_attr2 -class design -type string
-quiet -one_of "abc pqr lmn"
```

This example defines a user-defined attribute named, *my_attr3*, of type string, for both classes: port and pin.

```
% define_user_attribute my_attr3 -type string -class {port pin}
```

Related Topics

set_user_attribute

report_attribute

General Commands

filter_collection

Returns a subset of a collection by using a regular expression to filter the collection.

Usage

filter_collection [-regexp] [-nocase] <base_collection> <expression>

Arguments

- -regexp

  Uses the <expression> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Does not distinguish between lower and upper case. This argument may be used without the -regexp argument.

- <base_collection>

  Specifies the base collection to be filtered.

- <expression>

  Specifies an expression used to filter the base collection.

Examples

Example 1

This example creates a collection called col and places all input ports that have the letter x at the beginning of their name into the collection:

```
% set col [filter_collection [get_ports x*] {@port_direction==in}]
```

Example 2

This example shows how you can use filter_collection in a loop in a *.tcl* file:

```
foreach reg $all_registers {
   set target [filter_collection [all_fanout -end -from $reg] \
      "full_name=~*vdma*"]
}
```

Example 3

These examples illustrate syntax supported by the -regexp argument and the resulting output:

```
% get_object_names [filter_collection [get_designs -quiet *par32*] \
   -regexp {name =~ (.*)[0-9]+}]
sparc_ifu_par32__27 sparc_ifu_par32__28 sparc_ifu_par32__29
sparc_ifu_par32__30 sparc_ifu_par32__31 sparc_ifu_par32__32
sparc_ifu_par32__33  sparc_ifu_par3 ...(message truncated)

% get_object_names [filter_collection [get_designs -quiet *par32*] \
    -regexp {name =~ (.*)\d+}]
sparc_ifu_par32__parameterized0__27 sparc_ifu_par32__parameterized0__28
sparc_ifu_par32__parameterized0__29 sparc_ifu_par32__parameterized0__30
sparc_ifu_par32__parameterized0__8 sparc_ifu_par32__parameterized0__9
sparc_ifu_par32__parameterized0__10 sparc_ifu_par32__parameterized0__1
...(message truncated)
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

foreach_in_collection

Iterates over elements in a collection.

Usage

foreach_in_collection <collection_item> <base_collection> <commands>

Arguments

- <collection_item>

  Specifies the current element in the collection.

- <base_collection>

  Specifies the collection name.

- <commands>

  Specifies commands that are executed while iterating through each element of the <base_collection>.

Examples

This example loops through each member of collection1 and executes a set of commands during each iteration:

```
foreach_in collection member collection1 { <commands> }
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

filter_collection

index_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

get_attribute

Returns a value for an attribute.

Usage

get_attribute [-quiet] [-class <class>] <object> <attribute>

Arguments

- -quiet

  Suppresses output of warnings.

- -class <class>

  Specifies the class. The following classes are supported: cell, clock, design, inst, lib, lib_cell, lib_pin, module, net, pin, port, and power_domain.

- <object>

  Specifies the object, such as a net or pin.

- <attribute>

  Specifies an attribute. These attributes can only be read and not set.

Description

This command gets a value for an attribute. These attributes can only be read and not set. See the report_attribute command examples for a list of valid attributes.

Examples

Example

This example reports all the attributes of a net and returns the value of a single attribute.

```
% report_attribute [get_net efc_spc_fuse_clk1]
Report Net Attributes:
--+--------------+-----------------+--------
  |Attribute Name|Value            |Writable
--+--------------+-----------------+--------
1 |name          |efc_spc_fuse_clk1|no
2 |full_name     |efc_spc_fuse_clk1|no
3 |dont_touch    |false            |no
4 |num_pins      | 3               |no
5 |route_length  | 1234.520020     |no
6 |net_type      |signal           |no
7 |driver_count  | 1               |no
8 |object_class  |net              |no
--+--------------+-----------------+--------

% get_attribute -class net [get_net efc_spc_fuse_clk1] route_length
1234.520020
```

Example

The following example shows how to get various attributes of the pin DRAM23_N_REF_RES.

```
% get_attribute -class pin DRAM23_N_REF_RES fanout
1
% get_attribute -class pin DRAM23_N_REF_RES direction
in
% get_attribute -class pin DRAM23_N_REF_RES port_direction
in
% get_attribute -class pin DRAM23_N_REF_RES is_clock_pin
false
```

Example

This example shows how to get the value of the attribute without specifying the class of the object.

```
% get_attribute [get_lib_cell NAND2] threshold_voltage_group
vt1
```

Related Topics

report_attribute

set_attribute

General Commands

get_cells

Returns a list of instances of cells in the design based on the selection criteria.

Usage

get_cells [-quiet] [-regexp] [-nocase] [-filter <expression>] [-hierarchical]
[-of_objects <objects>] [<patterns>] [-hsc <separator>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any cells that match the specified criteria, the expression is evaluated based on the cell's attributes. If the expression evaluates to true, it is included in the result.

- -hierarchical

  Searches down the design hierarchy. The default is not to do this.

- -of_objects <objects>

  Limits the search to the specific <objects>. You cannot use the -hierarchical option with the -of_objects option.

- <patterns>

  Limits the search to the specific <patterns>. Glob-style patterns such as *, bar?, or *foo* are supported.

- -hsc <separator>

  This option is ignored.

Description

Returns a list of all instances in the design which match, given glob style patterns. This command should only be used in combination with other SDC commands.

Examples

Example 1

The following example sets all instances that have inst at the beginning of their name in the directory path/to/some as a false path:

```
% set_false_path -through [get_cells path/to/some/inst*]
```

Example 2

The following example sets all instances that have inst at the beginning of their name in the top level of the hierarchy as a false path.

```
% set_false path -through [get_cells inst*]
```

Example 3

The following example sets all instances that have inst at the beginning of their name in the level below the top level of hierarchy as a false path.

```
% set_false path -through [get_cells */inst*]
```

Example 4

The following example uses get_pins and get_cells to report input clock pins and output pins of specified registers by using multiple filters.

```
% get_pins -of_objects [ get_cells -hier sub_o* \
-filter "is_ff==true"] -filter "direction==in && is_clock_pin==true \
|| direction==out”

{sub_o_reg/Q sub_o_reg/CK}
```

Example 5

The following example uses get_cells to set_preserve_boundary on all user-defined modules.

```
foreach el [get_cells * -hier -filter "is_user_hierarchy==true"] {
  set_preserve_boundary -verbose [get_object_name $el]
}
```

Related Topics

get_lib_cells

filter_collection

SDC Commands

get_clocks

Returns a list of instances in the design based on the specified options.

Usage

get_clocks [-quiet] [-regexp] [-nocase] [-filter <expression>] [<patterns>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any clocks that match the specified criteria, the expression is evaluated based on the clock's attributes. If the expression evaluates to true, it is included in the result.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

Examples

```
% set_false_path -from [get_clocks Clk*] -to fp_reg
```

Related Topics

create_clock

all_clocks

SDC Commands

get_design_effort

Returns the settings configured with the set_design_effort command.

Usage

get_design_effort

Arguments

None.

Examples

```
% get_design_effort
leakage effort: 0
```

Related Topics

set_design_effort

get_designs

Returns a list of the specified designs.

Usage

get_designs [<patterns>] [-comment <string>] [-exact] [-filter <expression>] [-hierarchical] [-match_rtl_name] [-nocase] [-quiet] [-regexp]

Arguments

- <patterns>

  Optional. Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -comment <string>

  Optional. Specifies a string for commenting the operation. The <string> is displayed in standard output before the list of designs.

- -exact

  Optional. Searches for any object with a name that matches the <patterns> exactly. The default is true.

- -filter <expression>

  Optional. Returns a list of every design for which <expression> evaluates to true.

- -hierarchical

  Optional. Searches down the design hierarchy. The default is false.

- -match_rtl_name

  Optional. Returns a list of every design with an RTL name matching <patterns>.

- -nocase

  Optional. Specifies that the search for objects is case-insensitive.

- -quiet

  Optional. Suppresses the output of warnings.

- -regexp

  Optional. Interprets the <patterns> argument as containing regular expressions rather than simple wildcard patterns.

Examples

Example 1: List Designs in the Current Design

This example lists all designs in the current design.

```
[oasys-RTL]$ get_designs *
```

Example 2: List Every Design with a Name that Starts with a Specific String

This example lists every design whose name starts with *mux3d.*.

```
[oasys-RTL]$ get_designs mux3d.* -regexp
```

Example 3: List the Attributes of a Design with a Specific RTL Name

This example lists every design whose RTL name is *hpdmc_iddr32* then reports the attributes of the *hpdmc_iddr32__4* design.

```
[oasys-RTL]$ get_designs hpdmc_iddr32 -match_rtl_name
{hpdmc_iddr32__6 hpdmc_iddr32__4 hpdmc_iddr32__2 hpdmc_iddr32__5}

[oasys-RTL]$ report_attribute [get_designs hpdmc_iddr32__4]
Report Module Attributes:
-----+------------------+--------------------------------+--------
     |Attribute Name    |Value                           |Writable
-----+------------------+--------------------------------+--------
1    |name              |hpdmc_iddr32__4                 |no
2    |full_name         |hpdmc_iddr32__4                 |no
3    |rtl_name          |hpdmc_iddr32                    |no
4    |number_of_pins    |                             106|no
5    |dont_touch        |false                           |no
6    |preserve_boundary |false                           |no
7    |object_class      |design                          |no
8    |parameterized_name|hpdmc_iddr32_</PD_TOP,default,>|no
9    |is_placed         |false                           |no
10   |is_area_optimized |true                            |no
11   |is_dont_ungroup   |false                           |no
-----+------------------+--------------------------------+--------
```

Example 3: List Every Design with a Name that Exactly Matches a Specific String

This example lists every design with the exact name of *mux3d_32*.

```
[oasys-RTL]$ get_designs mux3d_32 -exact
```

Example 4: Filter By Name and Ignore Case

This example lists every design whose name starts with *rr64* using the -filter option. The filter for *rr64* is case-insensitive.

```
[oasys-RTL]$ get_designs -filter name=~RR64* -nocase
```

Example 5: Filter Designs with a Specific Attribute Value

This example lists every design whose *dont_touch* attribute value is true. The -hierarchy option searches designs hierarchically and the -quiet option suppresses any warning messages.

```
[oasys-RTL]$ get_designs -filter "dont_touch==true" -hierarchical -quiet
```

Related Topics

General Commands

get_lib_cells

Returns the pointer to the library cells with a specific pattern in the cell name.

Usage

get_lib_cells [-quiet] [-regexp] [-nocase] [-filter <expression>]
[-of_objects <objects> | <patterns>] [-hsc <separator>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any library cells that match the specified criteria, the expression is evaluated based on the library cell's attributes. If the expression evaluates to true it is included in the result.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -hsc <separator>

  This option is ignored.

Examples

```
% get_lib_cell AND*
TechLib45nm/AND2_X1_HV TechLib45nm/AND2_X2_HV TechLib45nm/AND2_X4_HV …
```

Related Topics

SDC Commands

get_cells

get_lib_pins

Returns library pins that match a specified pattern.

Usage

get_lib_pins [-quiet] [-regexp] [-nocase] [-filter <expression>]
[-of_objects <objects> | <patterns>] [-hsc <separator>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any library cell pins that match the specified criteria, the expression is evaluated based on the library cell pin's attributes. If the expression evaluates to true it is included in the result.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -hsc <separator>

  This option is ignored.

Examples

```
% get_lib_pins D*
```

Related Topics

SDC Commands

filter_collection

get_libs

Returns libraries that match a specified pattern.

Usage

get_libs [-quiet] [-regexp] [-nocase] [-filter <expression>] [-of_objects <objects> | <patterns>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any library cell pins that match the specified criteria, the expression is evaluated based on the library cell pin's attributes. If the expression evaluates to true it is included in the result.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

Examples

```
% get_libs D*
```

Related Topics

SDC Commands

get_lib_cells

get_nets

Returns the specified nets in the design.

Usage

get_nets [-quiet] [-regexp] [-nocase] [-filter <expression>] [-hierarchical]
[-of_objects <objects> | <patterns>] [-hsc <separator>] [-segments]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any nets that match the specified criteria, the expression is evaluated based on the net's attributes. If the expression evaluates to true it is included in the result.

- -hierarchical

  Searches down the design hierarchy.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -hsc <separator>

  This option is ignored.

- -segments

  This option is ignored.

Description

Finds all nets in the design which match given glob style patterns. This command should only be used in combination with other SDC commands.

Examples

```
% set_false_path -through [get_nets path/to/some/net*]
```

Related Topics

SDC Commands

filter_collection

get_pins

get_macros

Returns a list of all macros in the design.

Usage

get_macros

Arguments

None.

Examples

```
% get_macros
tlb_sram32x8  lsu_sram512x8
```

Related Topics

General Commands

get_object_names

Returns the names of specified objects. The names are strings that you can pass as arguments to commands or use in scripts.

Usage

get_object_names <objects>

Arguments

- <objects>

  Specifies the objects.

Examples

```
% get_object_names [get_cells o*]

% get_object_names [filter_collection [get_designs -quiet *par32*] \
   -regexp {name =~ (.*)[0-9]+}]
sparc_ifu_par32__27 sparc_ifu_par32__28 sparc_ifu_par32__29
sparc_ifu_par32__30 sparc_ifu_par32__31 sparc_ifu_par32__32
sparc_ifu_par32__33 sparc_ifu_par3 ...(message truncated)

% get_object_names [filter_collection [get_designs -quiet *par32*] \
    -regexp {name =~ (.*)\d+}]
sparc_ifu_par32__parameterized0__27 sparc_ifu_par32__parameterized0__28
sparc_ifu_par32__parameterized0__29 sparc_ifu_par32__parameterized0__30
sparc_ifu_par32__parameterized0__8 sparc_ifu_par32__parameterized0__9
sparc_ifu_par32__parameterized0__10 sparc_ifu_par32__parameterized0__1
...(message truncated)
```

Related Topics

General Commands

get_operating_process

Returns the process of the current operating condition.

Usage

get_operating_process

Arguments

None.

Examples

```
% get_operating_process

1.0
```

Related Topics

General Commands

get_operating_voltage

get_operating_temperature

get_operating_temperature

Returns the operating temperature of the design.

Usage

get_operating_temperature

Arguments

None.

Examples

```
% get_operating_temperature

-40
```

Related Topics

General Commands

get_operating_process

get_operating_voltage

get_operating_voltage

Returns the operating voltage of the design.

Usage

get_operating_voltage

Arguments

None.

Examples

```
% get_operating_voltage

0.947
```

Related Topics

General Commands

get_operating_process

get_operating_temperature

get_pg_info

Returns power and ground related information.

Usage

get_pg_info [-primary_power_of <power_domain_name>]
[-primary_ground_of <power_domain_name>]
[-type_of <cell_name> | <pg_pin_name>]
[-voltage_of <cell_name> | <pg_pin_name>]
[-signal_pins_of <cell_name> | <pg_pin_name>]
[-related_power_of <cell_name> | <pin_name>]
[-related_ground_of <cell_name> | <pin_name>]

Arguments

- -primary_power_of <power_domain_name>

  Returns the primary power of the specified power domain.
- -primary_ground_of <power_domain_name>

  Returns the primary ground of the specified power domain.
- -type_of <cell_name> | <pg_pin_name>

  Returns the power/ground type of a power/ground pin of a library cell.
- -voltage_of <cell_name> | <pg_pin_name>

  Returns the voltage of a power/ground pin of a library cell.
- -signal_pins_of <cell_name> | <pg_pin_name>

  Returns pins that are associated with the power/ground pin of a library cell.
- -related_power_of <cell_name> | <pin_name>

  Returns the related power pin for a pin of a library cell.
- -related_ground_of <cell_name> | <pin_name>

  Returns the related ground pin for a pin of a library cell.

Examples

```
% get_pg_info -primary_power_of PD_TOP
```

Related Topics

General Commands

Low Power Commands

get_pins

Returns the specified pins in the design.

Usage

get_pins [-leaf] [-quiet] [-regexp] [-nocase] [-filter <expression>] [-hierarchical] [-of_objects <objects> | <patterns>] [-hsc <separator>]

Arguments

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -leaf

  Specifies to include only pins on leaf cells connected to any nets in the objects argument. This option can only be used if the -of_objects is also specified option.

- -filter <expression>

  Filters the collection using the specified <expression>. For any pins that match the specified criteria, the expression is evaluated based on the pin's attributes. If the expression evaluates to true it is included in the result.

- -hierarchical

  Searches down the design hierarchy.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -hsc <separator>

  This option is ignored.

Examples

Example

```
% set_false_path -through [get_pins pinX*]
```

Example

Use get_pins to report input clock pins and output pins of specified registers by using multiple filters.

```
% get_pins -of_objects [ get_cells -hier sub_o* \
-filter "is_ff==true"] -filter "direction==in && is_clock_pin==true \
|| direction==out”

{sub_o_reg/Q sub_o_reg/CK}
```

Related Topics

SDC Commands

filter_collection

get_ports

Returns the specified ports in the design.

Usage

get_ports [-hierarchical] [-quiet] [-regexp] [-nocase] [-filter <expression>]
[-of_objects <objects> | <patterns>] [-comment <string>]

Arguments

- -hierarchical

  Searches down the design hierarchy. The default is not to do this.

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Specifies the search to be case-insensitive for the specified objects.

- -filter <expression>

  Filters the collection using the specified <expression>. For any ports that match the specified criteria, the expression is evaluated based on the port's attributes. If the expression evaluates to true it is included in the result.

- -of_objects <objects>

  Limits the search to the specific <objects>.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -comment <string>

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

```
% set_false_path -through [get_ports pX*]

% get_ports -regexp {data_[ml]sb[][0-9][]]}
{data_lsb[7] data_lsb[6] data_lsb[5] data_lsb[4] data_lsb[3] data_lsb[2]
data_lsb[1] data_lsb[0] data_msb[7] data_msb[6] ...(6 more)}

% get_ports -regexp {DRAM[0-9]_C[SK]_[LP][][0-9][]]}
{DRAM0_CS_L[3] DRAM0_CS_L[2] DRAM0_CS_L[1] DRAM0_CS_L[0] DRAM0_CK_P[3]
DRAM0_CK_P[2] DRAM0_CK_P[1] DRAM0_CK_P[0] DRAM1_CS_L[3] DRAM1_CS_L[2]
...(22 more)}
```

Related Topics

SDC Commands

filter_collection

all_inputs

all_outputs

get_power_domains

Returns specified power domains.

Usage

get_power_domains <domains>

Arguments

- <domains>

  Specifies a list of defined power domains.

Examples

```
% get_power_domains PD_TOP
```

Related Topics

General Commands

Low Power Commands

get_references

Returns the instances of the given module or library cell name that has the specified pattern.

Usage

get_references [-hierarchical] [-quiet] [-regexp] [-nocase] [-exact] [-filter <expression>] <patterns> [-comment <string>]

Arguments

- -hierarchical

  Searches down the design hierarchy. The default is not to do this.

- -quiet

  Suppresses output of warnings.

- -regexp

  Uses the <patterns> argument as real regular expressions rather than simple wildcard patterns.

- -nocase

  Does not distinguish between lower and upper case.

- -exact

  An exact reference. This is the default.

- -filter <expression>

  If the expression evaluates to true, it is included in the result.

- <patterns>

  Limits the search to the specific <patterns>. Glob style patterns such as *, bar?, or *foo* are supported.

- -comment <string>

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

This example returns the names of the instances of decoder:

```
% get_object_names [get_references decoder]
dec
```

Related Topics

General Commands

get_selection

Returns the names of one or more objects selected in the GUI window.

Usage

get_selection

Arguments

None.

Examples

Example 1: Get the Name of the Object That Is Selected in the GUI

This example shows how to get the hierarchical name of a selected instance in the GUI and outputs it in the shell window. Select an instance first, then type the following in the shell window:

```
[0] get_selection
{flop_rptrs/xc4/jbussync1_ff}
```

Example 2: Set a Variable to the Names of the Objects That Are Selected in the GUI

This example sets the *my_selection* variable to the *tlu/tlu_scpd*, *tlu/tsa0*, and *tlu/tsa1* macros that are selected in the GUI.

```
[1] set my_selection [get_selection]
{tlu/tlu_scpd tlu/tsa0 tlu/tsa1}
```

Related Topics

set_selection

General Commands

get_supply_nets

Returns power/ground supply net objects.

Usage

get_supply_nets <net_name>

Arguments

- <net_name>

  Specifies a a supply net name to retrieve. Wildcards are supported with an asterisk.

Examples

```
% get_supply_nets VSS

/VSS
```

Related Topics

General Commands

get_supply_ports

Low Power Commands

get_supply_ports

Returns power/ground supply port objects.

Usage

get_supply_ports <port_name>

Arguments

- <port_name>

  Specifies a supply port name to retrieve. Wildcards are supported with an asterisk.

Examples

```
% get_supply_ports *

/VDD /VDD_1
```

Related Topics

General Commands

get_supply_nets

Low Power Commands

index_collection

Creates a collection of one object that is the nth object of another collection.

Usage

index_collection <base_collection> <index>

Arguments

- <base_collection>

  Specifies the collection.

- <index>

  Specifies the index into the collection.

Examples

This example creates a collection of the first object in collection c1:

```
% set c1 [get_cell {cell1 cell2}]
% set first [index_collection $c1 0]
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

remove_from_collection

sizeof_collection

sort_collection

General Commands

remove_from_collection

Removes objects from a collection creating a new collection.

Usage

remove_from_collection <base_collection> <object_list>

Arguments

- <base_collection>

  Specifies the name of the collection.

- <object_list>

  Specifies the items that are removed from the collection.

Examples

This example returns a collection of all cells except those from the top level:

```
% set xcells [remove_from_collection \
[get_cells * -hierarchical] [get_cells *]]
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

index_collection

sizeof_collection

sort_collection

General Commands

sizeof_collection

Determines the size of a collection.

Usage

sizeof_collection <base_collection>

Arguments

- <base_collection>

  Specifies the name of the collection that you want to determine the size of.

Examples

This example determines if the collection scanNets is not empty and if so, executes a set of commands:

```
set scanNets [get_nets SE*]
if {[sizeof_collection $scanNets] != 0} {
   set_multicycle_path -setup 3 -from [get_ports $scanEnPorts]
}
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sort_collection

General Commands

sort_collection

Creates a sorted collection for the specified collection.

Usage

sort_collection [-descending] <base_collection> [<criteria>

Arguments

- -descending

  Specifies to sort the collection in descending order.

- <base_collection>

  Specifies the name of the collection to sort.

- <criteria>

  Specifies to sort the collection based on the given criteria. Currently, only full_name is supported as the criteria.

Examples

This examples shows how to create a new sorted collection in ascending order, called sortedcollection:

```
% set sortedcollection [sort_collection collection1]
```

Related Topics

add_to_collection

append_to_collection

copy_collection

delete_design

filter_collection

foreach_in_collection

index_collection

remove_from_collection

sizeof_collection

General Commands

Library Database Commands

These commands access library cells and their attributes.

Table 8-4. Library Database Commands

Command	Description
copy_target_library	Copies an existing target library.
get_equivalent_lib_cells	Returns all usable library cells from the target library that have equivalent functionality to a specified cell.
get_lef_files	Returns the names of the LEF files that were read in for the current session.
get_lib_files	Outputs the name of the .lib files that were read in for the current session.
get_ptf_file	Returns the complete name of the PTF file that is loaded for the current session.
get_target_library	Returns the name of the current target library.
set_dont_use	Changes the dont_use attribute of a library cell. The attribute is applied to all target libraries to which the cell belongs if a list of target libraries is not specified.
set_target_library	Specifies a target library to be used to synthesize and optimize a design.

copy_target_library

Copies an existing target library.

Usage

copy_target_library <existing_target_library> <new_target_library>

Arguments

- <existing_target_library>

  Specifies the name of target library to copy.

- <new_target_library>

  Specifies the name of the copy.

Description

This command creates a copy of the target library. You can customize the copy for specific hierarchical instances or modules. The customization can be applied after the synthesize command.

Examples

This example copies the current target library and names the copy with the new library name. Then it sets the dont_use attribute for the cells of the new library. It modifies the new library by setting the dont_use attribute to true for cells with names that match the following patterns: NAN*6*X, AO*, and OA*X*. Finally, it specifies that the tool use the modified new library to map the instances with the name TcTModInst and the modules with the name DbXMod.

```
% copy_target_library  $current_target_library  $new_target_library

% set_dont_use -target_library $new_target_library  NAN*6*X* true
% set_dont_use -target_library $new_target_library  AO* true
% set_dont_use -target_library $new_target_library  OA*X* true
% synthesize
% set_target_library $new_target_library -instances [get_cells TgtModInst ]
% set_target_library $new_target_library -module [get_design DbXMod ]
```

get_equivalent_lib_cells

Returns all usable library cells from the target library that have equivalent functionality to a specified cell.

Usage

get_equivalent_lib_cells <cell>

Arguments

- <cell>

  Specifies a cell for which to return cells with equivalent functionality.

Examples

```
% get_equivalent_lib_cells xor2x1
xor2x1 xor2x2 xor2x4
```

Related Topics

General Commands

get_lef_files

Returns the names of the LEF files that were read in for the current session.

Usage

get_lef_files

Arguments

None.

Examples

```
% get_lef_file
.../openCore/nangate/Nangate_45/techLef.lef
.../openCore/sparcT1/lef/levelshifter.lef
...
```

Related Topics

read_lef

General Commands

get_lib_files

Outputs the name of the .lib files that were read in for the current session.

Usage

get_lib_files

Arguments

None.

Examples

```
% get_lib_files
.../Nangatetypical_conditional_ecsm.lib
...
.../levelshifter_fast.lib
```

Related Topics

read_library

General Commands

get_ptf_file

Returns the complete name of the PTF file that is loaded for the current session.

Usage

get_ptf_file

Arguments

None.

Examples

This example gets the name of the *original.ptf* file.

```
% get_ptf_file
/home/user/original.ptf
```

Related Topics

Library Database Commands

read_ptf

get_target_library

Returns the name of the current target library.

Usage

get_target_library [-all] [-instance <instance_list>] [-module <name>]

Arguments

- -all

  Optional. Returns a list of all defined target libraries.

- -instance <instance_list>

  Optional. Returns the target library set for a given instance.

- -module <name>

  Optional. Returns the target library set for a given module.

Description

Returns the list of target libraries, the target library specified on a module, or the current target library.

Examples

```
% get_target_library
default

% get_target_library -all
low_vt std_vt high_vt default

%get_target_library -module dff_s
high_vt

%get_target_library -instance ifu mix_vt
```

Related Topics

General Commands

set_dont_use

Changes the dont_use attribute of a library cell. The attribute is applied to all target libraries to which the cell belongs if a list of target libraries is not specified.

Usage

set_dont_use [-target_library <target_lib_list>] <lib_cell_list> {true | false}

Arguments

- -target_library <target_lib_list>

  Applies the set_dont_use command to a list of cells from target libraries in the <target_lib_list> list. The list is a space-separated Tcl list of target library names. If this option is not specified, the command applies to the specified cells in all of the target libraries to which the cells belong.

- <lib_cell_list>

  Specifies a space-separated Tcl list of library cell names. Glob-style pattern matching is performed on the library cell names.

- {true | false}

  Specifies a Boolean value for the set_dont_use attribute on the specified objects. A value of true sets the set_dont_use attribute, while a value of false removes the attribute. The default is true.

Description

Excludes cells from the specified target libraries so that these cells are not used for synthesis when the target libraries are selected to be used during synthesis. A given library cell can belong to multiple target libraries. The dont_use attribute can be applied to one or more specified libraries rather than all target libraries by applying the -target_library argument.

Examples

Example

This example reads a library into two different target libraries and sets different cells to dont_use in the two target libraries:

```
% read_library Lib65nm -target_library {hivt lovt}
% set_dont_use -target_library hivt {AND2X1lvt AND2X2lvt ...}
% set_dont_use -target_library lovt {AND2X1hvt AND2X2hvt ...}
```

Example

In this example, the dont_use attribute is disabled for the specified cell:

```
% read_library Lib65nm.lib
% set_dont_use {AND2X1lvt} false
```

Example

In this example, clock driver cells (CLKBUF* and CLKINV*) from the library are excluded from consideration during synthesis:

```
% read_library Lib65nm -target_library {hivt lovt}
% set_dont_use -target_library hivt [get_lib_cell CLKBUF*]
% set_dont_use -target_library hivt [get_lib_cell CLKINV*]
```

Related Topics

Design Editing and Optimization Control Commands

read_library

set_target_library

Specifies a target library to be used to synthesize and optimize a design.

Usage

set_target_library <target_lib_name> [-instances <instance_list>] [-modules <modules>]

Arguments

- <target_lib_name>

  Required. Specifies the name of the target library to be used for synthesis. Before running this command, create the target library with the read_library command.

- -instances <instance_list>

  Optional. Specifies instances to be mapped using this target library. It does not prohibit other instances from using this target library. If this option is not specified, then the <target_lib_name> is applied to the top design.

- -modules <modules>

  Optional. Specifies that the target library be restricted to the modules in the argument.

Description

Specifies the target library to be used for synthesis. If the set_target_library is not applied to specify a target library, a target library named default is used.

Target libraries must be populated with library cells using the read_library command. Libraries must contain at least one inverter and one of the following 2-input logic cells: AND, OR, NAND, or NOR. This is so the target library can be used for synthesis. Corresponding physical information must also exist. This information is read in with the read_lef command.

Examples

This example reads two libraries (Lib90nmA, Lib90nmB) and assigns each one to a target library (targetA, targetB). The target library is then set to targetA and synthesis is performed using only cells from library Lib90nmA.lib:

```
% read_library Lib90nmA.lib -target_library {targetA}
% read_library Lib90nmB.lib -target_library {targetB}
% read_lef myPhysLib.lef
% set_target_library targetA

% synthesize
```

Related Topics

read_library

General Commands

Optimization Control Commands

These commands direct the synthesize and optimize commands during optimization.

Table 8-5. Optimization Control Commands

set_dont_remove	Directs the tool to not remove specified instances during synthesis.
set_dont_touch	Marks specified instances or cells as not to be modified or removed during synthesis.
set_dont_touch_network	Preserves specified signal networks during optimization.
set_dont_ungroup	Marks specified hierarchical instances as not to be ungrouped.
set_fix_multiple_port_nets	Instructs the optimize command on what needs to be done for nets connecting to multiple ports of a module. The command can apply to feedthroughs, outputs, and buffer constants.
set_preserve_boundary	Preserves the boundary on the specified hierarchical instance during optimization.
set_route_layer	Assigns a routing layer to a specified list of pins.
set_route_layer_capacitance	Specifies the capacitance per unit length for a technology layer.
set_route_layer_edge_capacitance	Specifies the edge capacitance per unit length for a routing layer.
set_route_layer_resistance	Specifies the resistance per unit length for a routing layer.
set_size_only	Marks a specified set of cell instances for sizing only. These marked cells are not remapped or removed during optimization.

set_dont_remove

Directs the tool to not remove specified instances during synthesis.

Usage

set_dont_remove [-flat] [-all_instances] [-verbose] <obj_list> [true | false]

Arguments

- -flat

  Sets a dont_remove attribute on the cell at all levels of design hierarchy. This option only applies to cells.

- -all_instances

  Specifies that all instances should not be removed.

- -verbose

  Echoes information about what was set.

- <obj_list>

  A space-separated Tcl list of user-instantiated cells not to be removed during synthesis. Glob-style pattern matching is performed on the instance or module names.

- [true | false]

  Specifies a Boolean value for the dont_remove attribute on the specified objects. A value of true sets the dont_remove attribute, while a value of false removes the attribute. The default is true.

Description

Instances specified with this command are not removed after synthesis. This command must be issued after the synthesize command.

The set_dont_remove command does not restrict the tool from replacing the specified cells with equivalent functionality cells, unlike the set_dont_touch command. Low voltage threshold cells could be replaced with their high voltage threshold equivalent. Small drivers could be replaced with larger ones if the set_dont_remove command is used.

The set_dont_remove command accepts an object list as input. This means you can explicitly specify objects as follows:

```
set_dont_remove [get_cells */abc*]
```

In this case, the objects are instances (module or cell instances).

You can also specify a string, as follows:

```
set_dont_remove abc
```

In this case it searches for the object name abc until it is found. The tool searches for instances then modules.

Examples

Example

In this example, all instances with the string reg in their names are marked as not removed:

```
% set_dont_remove [get_cell *reg*]
```

Example

In this example, all instances with the string DnT in their names, in any module, are marked as not removed:

```
% set_dont_remove [get_cell */*DnT*]
```

Related Topics

Design Editing and Optimization Control Commands

set_dont_touch

set_dont_touch

Marks specified instances or cells as not to be modified or removed during synthesis.

Usage

set_dont_touch [-flat] [-verbose] <obj_list> [true | false]

Arguments

- -flat

  Sets a dont_touch attribute on the net at all levels of design hierarchy. This option only applies to nets.

- -verbose

  Echoes information about what was set.

- <obj_list>

  A space-separated Tcl list of hierarchical instances or library cells not to be modified or removed. Glob-style pattern matching is performed on the instance or module names.

- [true | false]

  Specifies a Boolean value for the dont_touch attribute on the specified objects. A value of true sets the dont_touch attribute, while a value of false removes the attribute. The default is true.

Description

Instances specified with this command are not modified or removed. This command must be issued after the synthesize command.

The set_dont_touch command accepts an object list as input. This means you can explicitly specify objects as follows:

```
set_dont_touch [get_cells */abc*]
```

In this case, the objects are instances (module or cell instances).

You can also specify a string, as follows:

```
set_dont_touch abc
```

In this case it searches for the object name abc until it is found. The search order is as follows: instance, net, module.

Note

For a net that is connected to standard cells, if the dont_remove attribute is set, then each combinational cell connected to that net inherits the dont_remove setting.

Examples

Example

In this example, all instances with the string reg in their names are marked so that they are not modified or removed:

```
% set_dont_touch [get_cell *reg*]
```

Example

In this example, all instances with the string DnT in their names, in any module, are marked so that they are not modified or removed:

```
% set_dont_touch [get_cell */*DnT*]
```

Example

In this example, the command looks for instances called X in the top-level module B1 and for an instance called Y at the top level. If found, they are marked as dont_touch. If not found, the search continues for nets, and then finally modules:

```
% set_dont_touch {B1/X Y}
```

Example

In this example, the command marks all cell instances connected to the net net1 at all levels of hierarchy as dont_touch:

```
% set_dont_touch -flat {net1}
```

Example

In the following example, the command looks for all instances with the string DnT in their names, in any module, and if found marks them so that they are not modified or removed. If no instances are found, it continues searching nets and then finally modules:

```
% set_dont_touch {*/*DnT*} false
```

Related Topics

Design Editing and Optimization Control Commands

set_dont_remove

set_dont_touch_network

Preserves specified signal networks during optimization.

Usage

set_dont_touch_network <network> [-no_propagate]

Arguments

- <network>

  Specifies a signal tree (in general, a global net such as a clock or reset) that you want to keep unmodified.

- -no_propagate

  Specifies to stop propagating the set_dont_touch_network attribute at the first combination cell encountered. By default, the attribute is propagated throughout all combinational cells in the fanout.

Description

This command instructs the Oasys-RTL tool not to modify the specified signal network (such as the pins and ports of a clock or reset tree in the current design) during optimization. The transitive fanout from the specified collection of pins or ports is marked with dont_touch attribute. This is useful for trees that you want to maintain as unmodified during optimization because the refinements are performed with downstream physical implementation tools.

Examples

```
set_dont_touch_network [all_clocks]
set_dont_touch_network $reset
```

Related Topics

Design Editing and Optimization Control Commands

set_dont_ungroup

Marks specified hierarchical instances as not to be ungrouped.

Usage

set_dont_ungroup [-verbose] <obj_list> [true | false]

Arguments

- -verbose

  Echoes information about what was set.

- <obj_list>

  A space-separated Tcl list of hierarchical instances not to be ungrouped. Glob-style pattern matching is performed on the instance or module names.

- [true | false]

  Specifies a Boolean value for the dont_remove attribute on the specified objects. A value of true sets the dont_ungroup attribute, while a value of false removes the attribute. The default is true.

Description

Specifies not to ungroup instances of hierarchical modules. This command must be issued after the synthesize command.

The set_dont_ungroup command accepts an object list as input. This means you can explicitly specify objects as follows:

```
set_dont_ungroup [get_cells */abc*]
```

In this case, the objects are instances (module or cell instances).

You can also specify a string, as follows:

```
set_dont_ungroup abc
```

In this case, it searches for the “abc” object name until it is found. The search order is as follows: instance, net, module.

Examples

Example

In this example, all instances with “reg” the string in their names are marked so that they are not ungrouped:

```
% set_dont_ungroup [get_cell *reg*]
```

Example

In this example, all instances with “DnT” the string in their names (in any module) are marked so that they are not ungrouped:

```
% set_dont_ungroup [get_cell */*DnT*]
```

Example

In this example, the command looks for instances called X in the top-level module B1 and for an instance called Y at the top level. If found, they are marked as dont_ungroup. If not found, the search continues for nets, and then finally for modules:

```
% set_dont_ungroup {B1/X Y}
```

Example

In the following example, the command looks for all instances with the “DnT” string in their names (in any module) and if found, marks them so that they are not modified or removed. If no instances are found, it continues searching nets and then finally modules:

```
% set_dont_ungroup {*/*DnT*} false
```

Related Topics

Design Editing and Optimization Control Commands

ungroup

set_fix_multiple_port_nets

Instructs the optimize command on what needs to be done for nets connecting to multiple ports of a module. The command can apply to feedthroughs, outputs, and buffer constants.

Usage

set_fix_multiple_port_nets [-all] [-buffer_constants] [-feedthroughs] [-outputs]

Arguments

- -all

  Optional. Specifies all options: -buffer_constants, -feedthroughs, and -outputs.

- -buffer_constants

  Optional. Inserts a buffer to prevent a constant from driving an output port directly.

- -feedthroughs

  Optional. Specifies that for every feedthrough net in a module, a buffer is inserted.

- -outputs

  Optional. Adds a buffer to prevent any nets from directly driving more than one output port.

Examples

Example 1: Insert Buffers for Feedthrough Nets

The following example inserts a buffer for every feedthrough net.

```
[oasys-RTL]$ set_fix_multiple_port_nets -feedthroughs
```

Related Topics

Design Editing and Optimization Control Commands

set_preserve_boundary

Preserves the boundary on the specified hierarchical instance during optimization.

Usage

set_preserve_boundary [-hierarchical] [-verbose] <obj_list> [true | false]

Arguments

- -hierarchical

  Optional. Specifies that boundaries going down the hierarchy are preserved. The default is not to do this.

- -verbose

  Optional. Echoes information about what was set.

- <obj_list>

  Required. Specifies a space-separated Tcl list of hierarchical instances or library cells not to be modified or removed. Glob-style pattern matching is performed on the instance or module names.

- [true | false]

  Optional. Specifies a Boolean value for the dont_remove attribute on the specified objects. A value of true sets the preserve_boundary attribute, while a value of false removes the attribute. The default is true.

Description

Preserves the boundary on the specified hierarchical instance during optimization. If set_preserve_boundary is not specified, the instance hierarchical boundary may change because of the following:

- Propagation of constants across the boundary
- Propagation of unconnected or un-driven ports
- Propagation of equivalent or inverted-equivalent signals
- Inversion is pushed across the boundary

The set_preserve_boundary command accepts an object list as input. This means you can explicitly specify objects as follows:

```
set_preserve_boundary [get_cells */abc*]
```

In this case, the objects are instances (module or cell instances).

You can also specify a string, as follows:

```
set_preserve_boundary abc
```

In this case it searches for the object name abc until it is found. The search order is as follows: instance, module.

Examples

Example

This example preserves the boundary of all instances with the string ccx in their name:

```
% set_preserve_boundary [get_cell *ccx*]
```

Example

This example searches for an instance named abc and if found preserves its boundary. If an instance is not found, the tool continues searching for a module named abc and preserves its boundary.

```
% set_preserve_boundary abc
```

Related Topics

Design Editing and Optimization Control Commands

set_route_layer

Assigns a routing layer to a specified list of pins.

Usage

set_route_layer [-layer <layer_number>] <port_pin_list> [-comment "<comment>"]

Arguments

- -layer <layer_number>

  Specifies an existing routing layer number.

- <port_pin_list>

  Specifies a Tcl list of pin objects.

- -comment "<comment>"

  Specifies a string used for commenting the operation.

Examples

```
% set_route_layer -layer 5 [get_pins pcx*] \
   -comment "pcx pins must use layer 5"
```

Related Topics

General Commands

set_route_layer_capacitance

set_route_layer_edge_capacitance

set_route_layer_resistance

set_route_layer_capacitance

Specifies the capacitance per unit length for a technology layer.

Usage

set_route_layer_capacitance <layer_name> <capacitance>

Arguments

- <layer_name>

  Specifies the layer name to which the capacitance value is set.

- <capacitance>

  Specifies a floating-point capacitance per unit length in fF/um.

Examples

```
% set_route_layer_capacitance metal3 0.03
```

Related Topics

General Commands

set_route_layer

set_route_layer_edge_capacitance

set_route_layer_resistance

set_route_layer_edge_capacitance

Specifies the edge capacitance per unit length for a routing layer.

Usage

set_route_layer_edge_capacitance <layer_name> <capacitance>

Arguments

- <layer_name>

  Specifies the layer name to which the capacitance value is set.

- <capacitance>

  Specifies a floating-point edge capacitance per unit length in fF/um.

Examples

```
% set_route_layer_edge_capacitance metal3 0.0122
```

Related Topics

General Commands

set_route_layer_capacitance

set_route_layer

set_route_layer_resistance

set_route_layer_resistance

Specifies the resistance per unit length for a routing layer.

Usage

set_route_layer_resistance <layer_name> <resistance>

Arguments

- <layer_name>

  Specifies the layer name to which the resistance value is set.

- <resistance>

  Specifies a floating-point resistance per unit length in ohms/um.

Examples

```
% set_route_layer_resistance metal3 14
```

Related Topics

General Commands

set_route_layer_capacitance

set_route_layer_edge_capacitance

set_route_layer

set_size_only

Marks a specified set of cell instances for sizing only. These marked cells are not remapped or removed during optimization.

Usage

set_size_only [-flat] [-all_instances] [-verbose] <object_list> [true | false]

Arguments

- -flat

  Sets a set_size_only attribute on the net at all levels of hierarchy.

- -all_instances

  Sets a set_size_only attribute on all instances.

- -verbose

  Echoes information about what was set.

- <object_list>

  Specifies a list of leaf cell instances on which the size_only attribute is to be set.

- [true | false]

  Specifies a Boolean value for the size_only attribute on the specified objects. A value of true sets the size_only attribute, while a value of false removes the attribute. The default is true.

Examples

The below example sets the attributes on the leaf cell instance U2, so it can be sized during optimization:

```
% set_size_only U2
```

Related Topics

get_attribute

Design Editing and Optimization Control Commands

Chapter 9
Multi-Bit Mapping Commands

Multi-bit commands direct the multi-bit mapping with multi-bit groups and attributes.

Table 9-1. Multi-bit Mapping Commands

Command	Description
assign_to_multibit_group	Assigns single-bit register instances to an existing multi-bit group.
create_multibit_group	Creates a multi-bit group to contain single-bit registers instances for multi-bit mapping.
get_multibit_groups	Returns a list of available multi-bit groups that are used for multi-bit mapping.
remove_from_multibit_group	Removes instances from an existing multi-bit group.
remove_multibit_group	Removes the specified multi-bit group.
set_map_to_multibit	Identifies sequential library cells or sequential instances as candidates for multi-bit mapping.

assign_to_multibit_group

Assigns single-bit register instances to an existing multi-bit group.

Usage

assign_to_multibit_group <group_name> -instances <instance_list>

Arguments

- <group_name>

  A required argument that specifies the name of the group to which the tool assigns the register instances. You must create groups using the create_multibit_group command.

- -instances <instance_list>

  A required argument that specifies a list of leaf-level instances to assign to the <group_name>. Use the get_cells command to generate the <instance_list>. This option only assigns single-bit register instances to <group_name>. It ignores all instances in the <instance_list> that are not single-bit registers.

  When the tool assigns an instance to a multi-bit group, it assigns the value of the group name to the instance's multibit_group attribute.

Description

You can assign instances to multi-bit groups to control the selection of single-bit registers the tool can pack together during multi-bit mapping. The tool packs an instance only with other instances that are assigned to the same group. It does not pack instances that belong to different groups into the same multi-bit instance. Unassigned instances are packed only with other unassigned instances.

An instance can belong to only one multi-bit group. Assigning an instance to a multi-bit group does not guarantee that the tool is able to pack the instance.

You must use the "synthesize -map_to_scan" command to map the design to scan flops if you want to use the map_to_multibit command to pack the flops into multi-bit registers.

This command requires that you have already created <group_name> with the create_multibit_group command.

Examples

This example creates the *mgroup_A* multi-bit group, assigns the single-bit register instances in the hierarchical *ifu* cell to *mgroup_A*, then performs multi-bit mapping for the design.

```
% create_multibit_group mgroup_A
% assign_to_multibit_group mgroup_A -instances [get_cells ifu/*reg*]
% map_to_multibit
```

Related Topics

Multi-Bit Mapping Commands

create_multibit_group

map_to_multibit

remove_from_multibit_group

create_multibit_group

Creates a multi-bit group to contain single-bit registers instances for multi-bit mapping.

Usage

create_multibit_group <group_name> [-instances <instance_list>]

Arguments

- <group_name>

  A required argument that specifies the name of the group to create.

- -instances <instance_list>

  An optional argument that specifies the list of instances to include in the group.

Description

Creates a multi-bit group with the instances specified in the -instances option. If the -instances option is not used, the tool creates the empty group. You can later add instances to the group with the assign_to_multibit_group command. This command creates a single group at a time. To create multiple multi-bit groups, repeat the command for each group.

You can assign instances to multi-bit groups to control the selection of single-bit registers the tool can pack together during multi-bit mapping. The tool packs an instance only with other instances that are assigned to the same group. It does not pack instances that belong to different groups into the same multi-bit instance. Unassigned instances are packed only with other unassigned instances.

An instance can belong to only one multi-bit group. Assigning an instance to a multi-bit group does not guarantee that the tool is able to pack the instance.

You must use the “synthesize -map_to_scan” command to map the design to scan flops if you want to use the map_to_multibit command to pack the flops into multi-bit registers.

Examples

This example creates the *mgroup_A* multi-bit group with the *lsu/load_store_reg** and *lsu/save_data_reg** cells.

```
% create_multibit_group mgroup_A -instances {[get_cells \
   lsu/load_store_reg*] [get_cells lsu/save_data_reg*]}
```

This example create the *mgroup_B* group, assigns the *exu/exec_stat_reg** cell to it, then performs multi-bit mapping.

```
% create_multibit_group mgroup_B
% assign_to_multibit_group mgroup_B -instances [get_cells \
   exu/exec_stat_reg*]
% map_to_multibit
```

Related Topics

Multi-Bit Mapping Commands

map_to_multibit

remove_multibit_group

get_multibit_groups

Returns a list of available multi-bit groups that are used for multi-bit mapping.

Usage

get_multibit_groups

Arguments

None.

Description

Returns the name of the multi-bit groups that were created using the create_multibit_group command.

Examples

This example creates the *mgroup_A* multi-bit group, then lists the multi-bit groups.

```
% create_multibit_group mgroup_A
% get_multibit_groups
mgroup_A
```

Related Topics

Multi-Bit Mapping Commands

create_multibit_group

map_to_multibit

remove_from_multibit_group

Removes instances from an existing multi-bit group.

Usage

remove_from_multibit_group <group_name> [-instances <instance_list>]

Arguments

- <group_name>

  Specifies the name of the group from which the tool removes the instances.

- -instances <instance_list>

  Specifies the list instances to be removed from <group_name>.

Description

Removes instances from a multi-bit group that was created with the create_multibit_group command.

Examples

This example creates the *mgroup_A* multi-bit group, assigns the *ifu* instance to the group, then removes the *ifu* instance from the group.

```
% create_multibit_group mgroup_A -instances ifu
% remove_from_multibit_group mgroup_A -instances ifu
```

Related Topics

Multi-Bit Mapping Commands

assign_to_multibit_group

create_multibit_group

map_to_multibit

remove_multibit_group

Removes the specified multi-bit group.

Usage

remove_multibit_group <group_name>

Arguments

- group_name

  Specifies the name of the multi-bit group to remove.

Description

Removes the multi-bit group that was created using the create_multibit_group command.

Examples

This example creates the *ifu_grp* multi-bit group and then removes it.

```
% create_multibit_group ifu_grp -instances ifu_reg*
% remove_multibit_group ifu_grp
```

Related Topics

Multi-Bit Mapping Commands

create_multibit_group

map_to_multibit

set_map_to_multibit

Identifies sequential library cells or sequential instances as candidates for multi-bit mapping.

Usage

set_map_to_multibit {-lib_cell <library_cell_name> | <instance_list>} [true | false]

Arguments

- -lib_cell <library_cell_name> | <instance_list>

  Specifies either flip-flop library cells or flip-flop instances to be considered for or excluded from multi-bit mapping.

  <library_cell_name> specifies the single-bit library cells.

  <instance_list> specifies the particular instances in the design.

  -lib_cell <library_cell_name> and <instance_list> are mutually exclusive.

- true | false

  An optional argument that specifies whether or not the cell is a candidate for multi-bit mapping. The default is true.

Description

Enables or disables mapping of single-bit library cells or instances in the netlist to multi-bit registers during multi-bit mapping. This command sets the is_map_to_multibit attribute of the specified instance to the specified boolean value. By default, all library flip-flop cells and flip-flop instances are candidates for mapping. The map_to_multibit command performs the multi-bit mapping.

Examples

This example excludes all instances of the *DFFQ* library cell from multi-bit mapping.

```
% set_map_to_multibit -lib_cell "DFFQ*" false
```

This example excludes instances with *state_reg* and *bus_reg* in their names from multi-bit mapping.

```
% set_map_to_multibit [list[get_cells top/control/state_reg*] \
   get_cells top/dcache/control/bus_reg*]] false
```

Related Topics

Multi-Bit Mapping Commands

map_to_multibit

Chapter 10
Low Power Commands

The low power commands specify design attributes that control multi-voltage and power properties.

Table 10-1. Low Power Commands

Command	Description
add_to_threshold_voltage_group	Appends library cells to existing voltage threshold groups.
commit_upf	Implements the commands specified in the UPF file.
create_operating_condition	Creates a new voltage scaling operating condition based on existing operating conditions.
create_threshold_voltage_group	Creates a threshold voltage group used for optimizing leakage.
create_upf_pg_ports	Creates power and ground ports defined in the UPF for all user hierarchies.
instance_leakage	Outputs the power leakage of the design.
remove_clock_gate	Incrementally removes specified clock gates from the design by merging the enable logic into the fanout registers or clock gates (for multi-stage clock gating).
remove_threshold_voltage_group	Removes a threshold voltage group.
remove_upf	Removes the UPF constraints in the internal database.
rename_threshold_voltage_group	Renames a threshold voltage group.
reset_switching_activity	Restores the default values for switching activity for the specified pins.
set_clock_gating_options	Sets clock-gating options.
set_clock_gating_stages	Sets the maximum number of clock-gating stages for specific registers.
set_dont_gate_clock	Marks specified registers that should not be clock-gated during synthesis.
set_switching_activity	Specifies the switching activity values for the specified objects.

Table 10-1. Low Power Commands (cont.)

Command	Description
set_voltage	Defines the voltage on a specified supply net.

add_to_threshold_voltage_group

Appends library cells to existing voltage threshold groups.

Usage

add_to_threshold_voltage_group **group_name -lib_cells *lib_cells***

Arguments

- **group_name**

  Specifies the name of the voltage threshold group to which you want to add cells.

- **-lib_cells *lib_cells***

  Specifies a list of library cells to add to the voltage threshold group.

Description

Appends the user-specified library cells to the existing threshold voltage group. This is applicable to threshold groups automatically created by the tool or to threshold groups you have created using the create_threshold_voltage_group command.

Examples

```
% create_threshold voltage_group -lib_cells {cell1 cell2} Vth_low1
% add_to_threshold voltage_group -lib_cells {cellA cellB} Vth_low1
```

Related Topics

Low Power Commands

remove_threshold_voltage_group

create_threshold_voltage_group

rename_threshold_voltage_group

report_leakage

commit_upf

Implements the commands specified in the UPF file.

Usage

commit_upf

Arguments

None.

Description

Implements the commands specified in the unified power format (UPF) file. This command must be entered after the synthesize and load_upf commands.

Examples

```
#synthesize design
% synthesize

#read UPF file
% load_upf powerFile.upf

#Implement UPF file and output to a log using the append option
% commit_upf >> upf.log
```

Related Topics

Low Power Commands

check_library

load_upf

report_electrical_violations

report_pst

synthesize

create_operating_condition

Creates a new voltage scaling operating condition based on existing operating conditions.

Usage

create_operating_condition ***operating_condition*** [-voltage *value*] [-from *operating_condition_1* [*operating_condition_2*]]

Arguments

- ***operating_condition***

  Specifies a name for the voltage scaling operating condition.

- -voltage *value*

  Specifies a floating-point target voltage value for the new operating condition.

- -from *operating_condition_1* [*operating_condition_2*]

  Specifies one or two operating conditions that the tool uses to create the new voltage scaling operating condition. Each item in the list must be either an operating condition name, such as op_cond_1 or a library name prefixed to an operating condition, such as design_lib/op_cond_1. The library is not required if the operating condition is unique across all libraries.

Description

The create_operating_condition command allows you to specify that the whole or part of the design is voltage scaled. After the new operating conditions are created, the Oasys-RTL tool sets the new operating conditions as follows

- **UPF designs —** at the design-level
- **Non-UPF designs —** at both the instance and design-level (the same as any operating condition created in the library).

The create_operating_condition command is usually used in cases where a limited number of libraries are available and the voltages need to be scaled. An example is when there is a set of libraries that needs to map to a different voltage than is specified in the UPF. If the UPF is set at 1.2 volts and the available libraries are set at .9 volts, you can use the command to create an operating condition based on the libraries. In this case, you use the command to inform the tool to use .9 volts for the specified operating condition (instead of 1.2 volts). In addition, all the delays, which are normally extracted from the libraries, are manipulated based on the updated voltage.

Examples

```
% create_operating_condition vscale_op_1 -voltage 0.5 \
   -from {op_cond_1, lib2/op_cond}
```

Related Topics

Low Power Commands

report_operating_conditions

set_domain_operating_conditions

set_operating_conditions

create_threshold_voltage_group

Creates a threshold voltage group used for optimizing leakage.

Usage

create_threshold_voltage_group **group_name -lib_cells *lib_cells***

Arguments

- **group_name**

  Specifies the name of the voltage threshold group to create.

- **-lib_cells *lib_cells***

  Specifies a list of library cells to include in the voltage threshold group.

Description

Creates a threshold voltage groups used for optimizing leakage. By default, the Oasys-RTL tool determines the low, medium, and high leakage groups based on the average leakage per unit area of all the cells.

The following conditions apply to the usage of this command:

- If you specify a cell that already exists in another previously defined voltage group, then it is removed from the previous group and added to this newly created group.
- You cannot create a new threshold voltage group with the same name as a previously defined threshold voltage group.
- If a cell is not added to any group, by default it gets added to the lowest leakage group.
- If a group is removed and the cells are not added to any group, by default the cells get added to the lowest leakage group.

To redefine library-specified or pre-defined voltage groups, delete the existing groups using the “remove_threshold_voltage_group *” command before issuing the create_threshold_voltage_group command to create new ones.

Examples

```
% create_threshold voltage_group -lib_cells {cell1 cell2} Vth_low1
```

Related Topics

Low Power Commands

add_to_threshold_voltage_group

remove_threshold_voltage_group

rename_threshold_voltage_group

report_leakage

create_upf_pg_ports

Creates power and ground ports defined in the UPF for all user hierarchies.

Usage

create_upf_pg_ports

Arguments

None.

Description

Uses the power and ground ports in the UPF to create signal ports in the design for all hierarchies that require ports. Nets are created automatically to connect the ports if necessary. The ports are connected as defined by connect_supply_net commands in the UPF.

Ports created with this command use the same names as the corresponding power/ground ports of the top-level power domain.

This command should be applied after the load_upf command. Depending on the design flow, you may need to set the power_ignore_pg_for_leaf_cells parameter to false. This allows the Oasys-RTL tool to read in the power pins for the standard cells and macros. If this parameter is set to true, the power information is only read in for special UPF cells, such as level shifters, isolation cells, and retention registers.

Examples

```
[oasys-RTL]$ set_parameter power_ignore_pg_for_leaf_cells false

[oasys-RTL]$ read_library power.lib

[oasys-RTL]$ synthesize

[oasys-RTL]$ load_upf constraints/power.upf

[oasys-RTL]$ create_upf_pg_ports
```

Related Topics

Low Power Commands

commit_upf

load_upf

power_ignore_pg_for_leaf_cells

create_operating_condition

instance_leakage

Outputs the power leakage of the design.

Usage

instance_leakage [-no_macros]

Arguments

- -no_macros

  An optional argument to specify not to include macros in the leakage value.

Examples

This example returns the value of the power leakage of the design with and without macros included in the leakage value.

```
[oasys-RTL]$ instance_leakage
16386364.00
[oasys-RTL]$ instance_leakage -no_macros
16040826.00
```

Related Topics

Low Power Commands

report_leakage

remove_clock_gate

Incrementally removes specified clock gates from the design by merging the enable logic into the fanout registers or clock gates (for multi-stage clock gating).

Usage

remove_clock_gate ***instance_list***

Arguments

- ***instance_list***

  Specifies the list of clock-gating instances to be removed. The full hierarchical names of the instances are required.

Description

This command incrementally removes clock-gates. If the specified clock-gate cell drives registers directly, this clock-gate is removed, and the flip-flop is remapped. Otherwise, the specified clock-gate will be merged with its fanout clock-gates.

Examples

Example

This example removes the specified clock-gates from the design.

```
% remove_clock_gate {clk_gate_1 clk_gate_2 clk_gate_3}
```

Example

This example removes clock-gates from registers with names beginning with “clk_gate_out” and ending with “reg”.

```
% remove_clock_gate [get_cells clk_gate_out*reg]
```

Related Topics

Low Power Commands

set_clock_gating_options

remove_threshold_voltage_group

Removes a threshold voltage group.

Usage

remove_threshold_voltage_group ***group_name***

Arguments

- ***group_name***

  Specifies the name of the threshold voltage group that to remove.

Return Values

Description

Removes a threshold voltage group. If a group is removed and the cells are not added to any group, by default the cells get added to the lowest leakage group.

Examples

```
% remove_threshold_voltage_group Vth_low1
```

Related Topics

Low Power Commands

add_to_threshold_voltage_group

create_threshold_voltage_group

rename_threshold_voltage_group

report_leakage

remove_upf

Removes the UPF constraints in the internal database.

Usage

remove_upf [-level_shifters] [-isolation_cells] [-enable_level_shifters] [-force]

Arguments

- -level_shifters

  Removes existing level shifter cells that do not have the dont_touch attribute.
- -isolation_cells

  Removes existing isolation cells that do not have the dont_touch attribute.
- -enable_level_shifters

  Removes existing enable level shifter cells that do not have the dont_touch attribute.
- -force

  Forces removal of cells even if they have the dont_touch attribute.

Description

The remove_upf command removes existing boundary cells based on the specified arguments. If no arguments are specified, it removes the UPF intent from the design.

Examples

```
% remove_upf
```

Related Topics

Low Power Commands

commit_upf

load_upf

rename_threshold_voltage_group

Renames a threshold voltage group.

Usage

rename_threshold_voltage_group ***current_name new_name***

Arguments

- ***current_name***

  Specifies the name of the threshold voltage group to rename.

- ***new_name***

  Specifies the new name of the threshold voltage group.

Examples

```
% rename_threshold_voltage_group Vth_low1 Vth_low2
```

Related Topics

Low Power Commands

add_to_threshold_voltage_group

create_threshold_voltage_group

remove_threshold_voltage_group

report_leakage

reset_switching_activity

Restores the default values for switching activity for the specified pins.

Usage

reset_switching_activity [-all] [*pin_list*] [-verbose]

Arguments

- -all

  An optional argument that esets the switching activity settings on all primary input and register output pins.

- *pin_list*

  An optional argument that specifies a list of pin to reset. The list should contain pin names.

- -verbose

  Outputs a summary of pins/instances for which switching activity is reset.

Description

Resets the probability, toggle percentage, and toggle rate settings to the default. The probability and toggle percentage default value is 50%. The toggle rate default value is a function of the toggle percentage and clock frequency.

This command removes any previous settings that were set using the set_switching_activity, read_saif, or read_vcd command. If the command is successful, the tool does not return any message.

All arguments are optional. However, if you do not specify an argument, no pins are reset. If you specify both -all and <pin_list>, the command ignores <pin_list>.

Examples

The following example sets the probability and the toggle percentage of the *Ctl1* and *Ctl2* pins to 10% and 20%, respectively. Next, it reports the settings. Then, it resets the probability and the toggle percentage to the default values and reports the settings again.

```
[oasys-RTL:/]$ set_switching_activity -pins {Ctl1 Ctl2} -probability 0.1 -toggle_rate 0.2

[oasys-RTL:/]$ report_switching_activity -pin {Ctl1 Ctl2} -toggle_percentage

Report Switching activities:
-----+-----+-------------+------+-------------------+------+------------------+------
     | Pin | Probability | Type | Toggle percentage | Type | Toggle rate (GHz)| Type
-----+-----+-------------+------+-------------------+------+------------------+------
1    |Ctl1 |     0.100000| USER |           0.250000|  SYS |          0.200000| USER
2    |Ctl2 |     0.100000| USER |           0.250000|  SYS |          0.200000| USER
-----+-----+-------------+------+-------------------+------+------------------+------

[oasys-RTL:/]$ reset_switching_activity {Ctl1 Ctl2}

[oasys-RTL:/]$ report_switching_activity -pin {Ctl1 Ctl2} -toggle_percentage

Report Switching activities:
-----+-----+-------------+------+-------------------+------+------------------+------
     | Pin | Probability | Type | Toggle percentage | Type | Toggle rate (GHz)| Type
-----+-----+-------------+------+-------------------+------+------------------+------
1    |Ctl1 |     0.500000|  SYS |           0.500000|  SYS |          0.400000|  SYS
2    |Ctl2 |     0.500000|  SYS |           0.500000|  SYS |          0.400000|  SYS
-----+-----+-------------+------+-------------------+------+------------------+------
```

Related Topics

read_saif

read_vcd

report_power

report_switching_activity

set_switching_activity

set_clock_gating_options

Sets clock-gating options.

Usage

set_clock_gating_options [-minimum_bitwidth *bitwidth*]
[-sequential_cell {none | ff | latch}] [-control_port *port_name*]
[-control_point *control_point_value*] [-observation_point] [-maximum_fanout *value*]
[-num_stages *number*] [-use_discrete_cells *cell_list*] [-string_tag *prefix*]
[-create_multi_stage] [-exclude_instantiated_clock_gates]
[-merge_multi_stage] [-enhanced_coverage] [-instances *instance_name*] [-log *filename*]

Arguments

- -minimum_bitwidth *bitwidth*

  An optional argument that specifies the minimum number of flip-flops that share common enable logic for the flip-flops to be clock-gated. The *bitwidth* value must be an integer. If this option is not specified, the default is 4.

- -sequential_cell {none | ff | latch}

  An optional argument that specifies the type of sequential element to use for the clock_gating_integrated_cell. The type corresponds with the first string in the four-string clock_gating_integrated_cell attribute of a library cell. Only those library cells with the first string equal to the sequential type are considered for clock-gating during synthesis. If -sequential_cell is not specified, the default is null, which signifies that any clock-gating cell can be chosen from the library. This is different from specifying the value none, where the tool only chooses a clock-gating cell with its first string equal to none in the Liberty attribute.

- -control_port *port_name*

  An optional argument that specifies the name of the test control port of each RTL module to which the control pin of the clock_gating_integrated_cell should be connected. If the specified port does not exist in an RTL module being synthesized, it is created in the resulting netlist. If -control_port is not specified, the default is null and if a clock-gating cell with control pin is used, the control pin is tied to 0. The Oasys-RTL tool only supports active high control signals.

- -control_point *control_point_value*

  An optional argument to specify whether the test control point should be available in the clock_gating_integrated_cell used for clock-gating. If it is available, use this option to specify that the test control point is before or after the sequential element of the clock_gating_integrated_cell. The *control_point_value* must be one of the following strings:

  - none
  - before

- after

This argument corresponds to the third string in the four-string clock_gating_integrated_cell attribute of a library cell in Liberty. A *control_point_value* equal to before or after corresponds to the values of precontrol or postcontrol, respectively, of this third string in the Liberty attribute. A matching clock_gating_integrated_cell is used for clock-gating. If this option is not specified and a control port is specified, any clock-gating cell with a control pin is used.

- -observation_point

  An optional argument to specify that a test observability output pin should be available in the clock_gating_integrated_cell for clock-gating. This argument corresponds to the fourth string in the four-string clock_gating_integrated_cell attribute of a library cell in Liberty. Specifying this argument corresponds to the value *obs* in the fourth string of the Liberty attribute. Similar to the rules in Liberty, this argument can only be specified if a control_point other than none is specified. If this argument is not specified, a clock_gating_integrated_cell with the value *obs* in its four-string attribute may still be used if this is the best available clock_gating_integrated_cell in the target library. The observability output pin of clock-gating cells is always left unconnected after synthesis. If this argument is not specified, the default is no, which is equivalent to the fourth string of the Liberty attribute being absent.

- -maximum_fanout *value*

  An optional argument to specify the maximum number of load connections for the cell. The *value* is an integer.

- -num_stages *number*

  An optional argument that limits the number of clock-gating stages (levels) to *number*. It does not undo any clock-gating that already exists in the incoming RTL if the -exclude_instantiated_clock_gates option is specified. For finer control to specify the maximum number of clock-gating stages for specific registers, see set_clock_gating_stages.

- -use_discrete_cells *cell_list*

  An optional argument that provides the ability for the Oasys-RTL tool to automatically create a clock-gating cell, rather than use one from the library. The Oasys-RTL tool creates a clock-gating cell module that uses the cells from the *cell_list*. By using this option, the clock-gating cell does not need to be put in the library. By creating these clock-gating cells, you have access to the internals of the cell. This provides the ability to reuse, optimize, or share these internal cells.

- -string_tag *prefix*

  An optional argument that specifies a prefix for internal cell instance names when applying the -use_discrete_cells option. The instance name takes the form: rt_<*prefix*>_<*type*>_<*number*>

  where:

  - <*prefix*> is user-specified

  - <*type*> is a latch, ff, or gate
  - <*number*> is automatically generated

- -create_multi_stage

  An optional argument that specifies that multi-stage clock-gating structures be created based on the -num_stages option at the end of timing optimization using placement information.

- -exclude_instantiated_clock_gates

  An optional argument that specifies that RTL-instantiated clock-gates are not counted in the total number of stages when creating or merging multi-stage clock-gates.

- -merge_multi_stage

  An optional argument that specifies that multi-stages of clock-gating be merged to meet the number of stages specified by -num_stages. If the -exclude_instantiated_clock_gates option is specified, the number of stages does not include RTL-instantiated stages.

- -enhanced_coverage

  An optional argument that specifies to improve clock-gating coverage by using advanced techniques to combine the gating of banks of registers that would otherwise not meet the minimum bit width threshold.

- -instances *instance_name*

  An optional argument that specifies the hierarchical instance name of the cell to which clock-gating options are applied. The -instances option can only be used with a subset of other clock-gating options.The following subset of options can be used with the -instances option:

  - -max_fanout
  - -min_bitwidth
  - -num_stages
  - -create_multi_stage
  - -enhanced_coverage
  - -exclude_instantiated_clock_gates
  - -merge_multi_stage
  - -power
  - -log

  If the -instances option is used with an option that is not in the subset, the tool issues a POWER-151 warning and ignores the option.

- -log *filename*

  An optional argument that specifies to save a log file. To specify thelog file name, provide a string *filename*. If you do not provide the string *filename*, the log file is named *$designName.clockgate.log*. The log file details the power savings analysis for each clock-gating move accepted or rejected during the creation of the multi-stage clock-gating structures.

Description

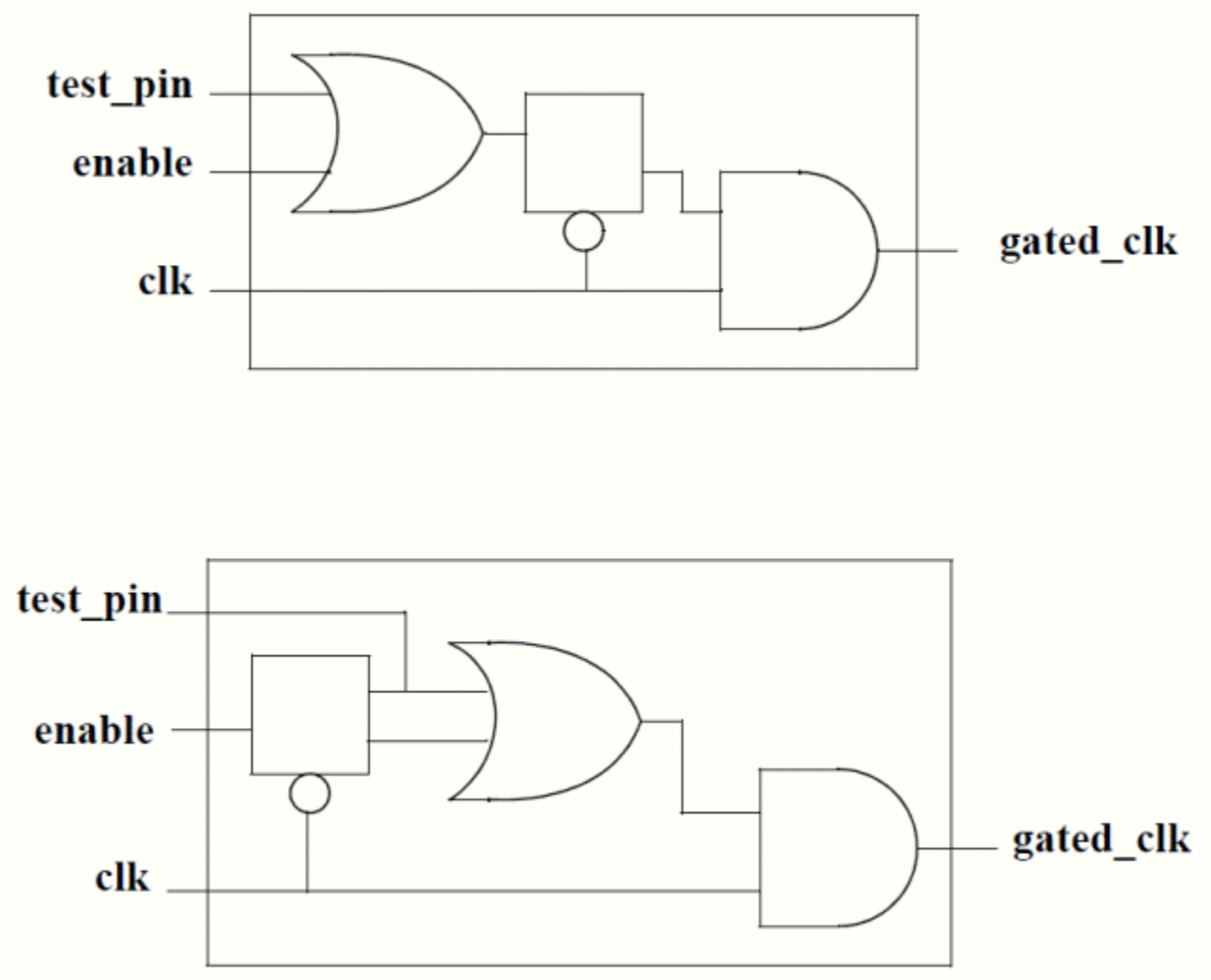
This command sets the options that control the following:

- The selection of clock_gating_integrated_cells from the target library to be used for clock-gating
- The amount of clock-gating to be done during synthesis

You must issue this command before synthesis. If you set the -gate_clock option to false in the synthesize command, these options have no effect. Set all required parameters with a single call of this command; all non-specified parameters are configured to their default value.

To check the values of the clock-gating options, use the report_clock_gating_options command. To check the clock_gating_integrated_cell chosen for clock-gating, use the report_clock_gating_cell command. The values of the clock-gating options revert to their

defaults upon issuing a set_clock_gating_options command with no arguments or a delete_design command (if a design has been loaded).



DFT Options for Clock-Gating

Adding clock-gating is one method to reduce power, but you must also consider the impact on design for test (DFT). To increase test coverage, you must ensure that the inserted clock-gating logic is controllable and observable. The -control_point option of the set_clock_gating_options command determines whether the test control point should be before (precontrol) or after (postcontrol) the sequential element.

Examples

Example 1

```
% set_clock_gating_options -minimum_bitwidth 8 -sequential_cell none

% report_clock_gating_options
info: clock_gating minimum_width = 8, sequential_cell = none,
control_port = (null), control_point = none,
observability = no [POWER-112]
```

Example 2

```
% set_clock_gating_options -control_port scan_enable -control_point before

% report_clock_gating_options
info: clock_gating minimum_width = 4, sequential_cell = (null),
control_port = scan_enable, control_point = before,
observability = no [POWER-112]
```

Example 3

```
% set_clock_gating_options -observation_point

% report_clock_gating_options
info: clock_gating minimum_width = 4, sequential_cell = (null)
control_port = scan_enable, control_point = before,
observability = no [POWER-112]
```

Example 4

```
% set_clock_gating_options -control_point before -observation_point

% report_clock_gating_options
info: clock_gating minimum_width = 4, sequential_cell = (null),
control_port = scan_enable, control_point = before,
observability = no [POWER-112]
```

Example 5

This example uses the unsupported combination of the -control_port and -instances options. The tool ignores the -control_port and generates the POWER-151 warning.

```
% set_clock_gating_options -control_port SE -instances MID
warning: option '-control_port' is used together with -instances, this
option '-control_port' will be ignored.  [POWER-151]
0
```

Example 6

This example uses -use_discrete_cells to create discrete clock gating cells.

```
% set_clock_gating_options \
   -use_discrete_cells "AND2X2 INVX2 BUFX4 TLATX1" -sequential_cell latch
info:   target library 'rt_clock_gating_3' is created for clock gating
[POWER-125]
0

synthesize -gate_clock
starting synthesize at 00:00:12(cpu)/23:12:43(wall) 68MB(vsz)/235MB(peak)
info:   clock-gating cell is built from discrete cells in target library
'rt_clock_gating_3'  [POWER-121]
i
```

The result appears in the netlist as follows:

```
...
DFFQXL \q_tmp_reg[0]  (.D(n_29), .CK(rt_clk_gate_clknet), .Q(n_28));
TLATX1 rt_clk_gate_latch (.D(en), .G(n_31), .Q(n_30), .QN());
INVX2 rt_clk_gate_gate (.A(clk), .Y(n_31));
AND2X2 rt_clk_gate_gate__0 (.A(clk), .B(n_30), .Y(rt_clk_gate_clknet));
...
```

Related Topics

Low Power Commands

synthesize

report_clock_gating_options

report_clock_gating_cell

report_clock_gating

set_clock_gating_stages

set_dont_gate_clock

set_clock_gating_stages

Sets the maximum number of clock-gating stages for specific registers.

Usage

set_clock_gating_stages ***list_of_regs num_stages***

Arguments

- ***list_of_regs***

  Specifies registers to which the maximum number of stages applies.

- ***num_stages***

  Specifies the maximum number of clock-gating stages and includes any RTL-instantiated clock-gates unless the -ignore_instantiated_clock_gates argument was specified with the set_clock_gating_options command.

Description

Sets the maximum number of stages for clock-gating with finer control compared to the "set_clock_gating_options -num_stages" command. If the "-num_stages" option is specified in the set_clock_gating_options command, the set_clock_gating_stages command has higher priority. The number of stages includes RTL-instantiated clock-gates. The number of stages includes RTL-instantiated clock-gates.

Examples

This example sets the maximum number of stages for specified registers to 2.

```
% set_clock_gating_stages [get_cells *reg*] 2
```

Related Topics

Low Power Commands

report_clock_gating

report_clock_gating_cell

report_clock_gating_options

set_clock_gating_options

set_dont_gate_clock

synthesize

set_dont_gate_clock

Marks specified registers that should not be clock-gated during synthesis.

Usage

set_dont_gate_clock [-verbose] ***inst_list*** [true | false]

Arguments

- -verbose

  Echoes information about what was set.

- ***inst_list***

  Specifies a list of instances that should not be clock-gated during synthesis. The instances are specified as a white space-separated Tcl list in the following form:

  ```
  <module_name>/<instance_name>
  ```

  Glob-style pattern matching is performed on the module names and instance names.

- true | false

  Specifies whether to set the dont_gate_clock attribute on the specified instances. A value of true sets the dont_gate_clock attribute on the specified objects, while a value of false removes the attributes. The default is true.

Description

Registers specified in this command are not clock-gated during synthesis. This command applies only to registers that are inferred as flip-flops. The command takes an object list, which can be explicitly specified as follows:

"set_dont_gate_clock [get_cells */abc*]"

You can also specify a string that searches for the object until one is found, such as the following:

"set_dont_gate_clock abc".

The tool searches the string with the order instance, net, module.

Examples

The following example marks instances with the string abc in their name as not to be clock-gated:

```
% set_dont_gate_clock [get_cell *abc*]
```

The next example searches for an instance abc and when found marks it as not to be clock-gated. If no instance are found, it continues checking for a module abc and marks the module as not be clock-gated.

```
% set_dont_gate_clock abc
```

Related Topics

Low Power Commands

report_clock_gating_options

set_clock_gating_options

report_clock_gating_cell

synthesize

report_clock_gating

set_switching_activity

Specifies the switching activity values for the specified objects.

Usage

set_switching_activity [-all | -pins *pin_list* | -clock_pins *clock_pin_list* | -instances *inst_list*] [-hierarchical] [-probability *probability*] [-toggle_rate *toggle_rate*] [-verbose]

Arguments

- -all

  Set the specified switching activity on all primary inputs and register output pins.

- -pins *pin_list*

  Indicates that the specified activity value is applied only to the pins in the <pin_list>.

- -clock_pins *clock_pin_list*

  Indicates that the specified activity value is applied only to the data signals associated with the specified clock pins.

- -instances *inst_list*

  Indicates that the specified activity value is applied to all outputs if the specified instances are not hierarchical instances. The specified activity value set on the inputs gets modified by the Boolean function of the gates to propagate it to the outputs.

- -hierarchical

  Indicates that the specified activity value is applied to the outputs of all leaf instances in the hierarchy of each specified hierarchical instance. The specified activity value set on the inputs gets modified by the Boolean function of the gates to propagate it to the outputs.

- -probability *probability*

  The probability that the signal value is at logic 1. The number must be between 0 and 1. A default of 0.5 is assumed if a value is not specified.

- -toggle_rate *toggle_rate*

  Specifies the number of toggles per time unit. The time unit is in nanoseconds. A default of 2 toggles per clock period is assumed if a value is not specified. When time constraints are specified in an SDC file, the clock frequency and toggle is taken from the specified waveform.

- -verbose

  Outputs a summary of pins/instances for which switching activity is set.

Description

Specifies the switching activity values (toggle rate and probability) for the specified pins. If no clock is related to the data pin, the worst clock of the design is used. If you also specify clock pins through the -clock_pins option, the computed toggle rate is only applied to the data pins

related to those clock pins. If you did not specify any clock pins, a computed toggle rate is applied to all data pins and the value for each data pin is based on its related clock pin.

Switching activity is set on the inputs and is propagated to the outputs only during reporting or power analysis. The switching activity set on the inputs gets modified by the Boolean function of the gate to propagate it to the outputs.

In general, if you specify a switching activity value at one level of the hierarchy, and then again at a lower level, the lower level overrides the higher one.

Note

Switching activity values set by the set_switching_activity command become the defaults until they are overridden by a subsequent invocation of the command (assuming different values are specified).

When multiple objects need to be specified (using the -pins, -clock_pins, and -instance arguments), use multiple commands starting with the more generic one.

Accurate switching activities can also be obtained by reading VCD or SAIF files.

Note

If VCD or SAIF files are not present and the set_switching_activity command is not used, default values are assumed for switching activity. The switching probability defaults to 0.5 and the toggle rate defaults to 2 toggles/ns. When switching activity goes to different hierarchy levels, the switching activity number will change. It is not 50% at all levels.

If VCD or SAIF files are present and the set_switching_activity command is used, the switching values of the set_switching_activity command will take precedence.

Examples

This example sets the default switching probability to 0.4 and the default toggle rate to 0.4:

```
% set_switching_activity -probability 0.4  -toggle_rate 0.4
```

Related Topics

Low Power Commands

read_saif

read_vcd

report_power

report_switching_activity

write_scandef

set_voltage

Defines the voltage on a specified supply net.

Usage

set_voltage ***voltage*** [-object_list *supply_nets*]

Arguments

- ***voltage***

  Specifies the target supply voltage. The voltage is a floating-point number in volts.

- -object_list *supply_nets*

  Specifies a list of supply nets to which the target voltage is set.

Description

This command must be specified during or after UPF constraints are loaded. If the supply net is a primary power/ground net of a domain, this command applies the specified operating voltage to elements in the domain. There must be corresponding libraries available for the operating voltage. The voltage must also be defined in the power state table.

Examples

```
% set_voltage 0.95 -object_list [get_nets VDD1]
```

Related Topics

Low Power Commands

commit_upf

load_upf

Chapter 11
Timing Commands

The timing commands define timing characteristics of the design.

Table 11-1. Timing Commands

Command	Description
config_timing	Configures the timing properties of the timer.
delete_timing	Deletes timing information, including SDC information.
disable_timing_mode	Disables the specified timing modes.
get_slack	Returns the slack value for a design object such as a path group or a specific pin, or for the entire design.
get_timing	Returns timing information or components that impacts timing.
get_timing_modes	Returns a list of all timing modes.
get_timing_paths	Returns a list of timing paths.
get_timing_precision	Returns the value of the internal timing scale set using the set_timing_precision command.
remove_case_analysis	Removes a previously defined case analysis for timing.
remove_clock_uncertainty	Removes previously set clock skew characteristics.
remove_driving_cell	Removes a previously specified driving cell for the specified pins.
remove_input_delay	Removes the input delay for the specified ports.
remove_input_transition	Removes the specified input transition on the specified pin list.
remove_load	Removes loads on the specified pin list.
remove_output_delay	Removes output delay ports.
remove_timing_exceptions	Removes timing exceptions.
reset_timing	Resets the internal timing calculation created in the database.
set_domain_operating_conditions	Sets the operating conditions for the specified power domain.
set_dont_retime	Specifies register instances that should not be retimed.

Table 11-1. Timing Commands (cont.)

Command	Description
set_retime_module	Identifies the modules that can be retimed.
set_timing_mode	Sets the timing mode of the design.
set_timing_precision	Sets the internal timing scale.
total_negative_slack	Returns the sum of negative slack values.
worst_slack	Returns the worst slack value in the current design.

Note

Min/Hold Timing Analysis Support: The timer of the Oasys-RTL tool does not perform early timing or hold time analysis; therefore, the -hold and -min options are ignored by the timer. However, the -hold and -min options in any SDC command are parsed, retained in the database and can be passed through to the output SDC with the write_sdc command.

config_timing

Configures the timing properties of the timer.

Usage

config_timing [-report] [-auto_group_paths { true | false }] [-interface_pg { true | false }] [-zero_rc_delay { true | false }]

Arguments

- -auto_group_paths { true | false }

  When set to true, specifies that register-to-register path groups are created for every clock defined thereafter. Oasys-RTL generates the resulting path group by concatenating the clock name (defined by create_clock) and the "_R2R" string. For example, if a clock was defined as "PLLclk", then the resulting path group would be named "PLLclk_R2R".

  All group paths are generated with equal weighting (1.0). In this case, the optimizer will give equal priority to all path groups. The weighting on a path group can be changed using the group_path command (eg. group_apth -name PLLclk_R2R -weight 0.4)

  Setting this switch to false disables the feature. If you do not specify the option, the default is false.

- -interface_pg { true | false }

  When set to true, specifies that path groups are generated for the timing paths on interface IO ports. Oasys-RTL generates three path groups:

  - I2R : all inputs to all registers/macros
  - R20 : all registers/macros to all outputs
  - I2O : all inputs to all outputs

  All group paths are generated with equal weighting (1.0). In this case, the optimizer will give equal priority to all path groups. The weighting on a path group can be changed using the group_path command (eg. group_apth -name I2) -weight 0.1)

- -report

  Generates a report of the current config_timing settings.

- -zero_rc_delay { true | false }

  Specifies that timing is configured with zero RC delays on nets. Specify this option to configure the timer to set zero resistance and capacitance values for the nets. If you do not specify the option, the default is false.

Examples

Example

The following example configures the Oasys-RTL tool to create path groups for every create_clock command executed thereafter, reads SDC constraints, and then shows the path groups for all clocks.

```
% config_timing -auto_group_path
% read_sdc block.sdc
% report_path_groups
```

Example

Following example configures the Oasys-RTL tool to report current settings.

```
% config_timing -report
auto_group_paths : false
zero_rc_delay    : false
```

Example

Following example configures Oasys-RTL for zero wireload synthesis.

```
% config_timing -zero_rc_delay
% synthesize
```

Related Topics

Timing Commands

report_timing

delete_timing

Deletes timing information, including SDC information.

Usage

delete_timing

Arguments

None.

Examples

```
% delete_timing
```

Related Topics

Timing Commands

disable_timing_mode

Disables the specified timing modes.

Usage

disable_timing_mode **-modes *mode_list***

Arguments

- **-modes *mode_list***

  Specifies a list of timing modes to disable.

Description

Disables the specified timing modes. Disabled timing modes do not appear in the results for get_timing_mode or any of the reporting commands. In addition, disabled timing modes are not written to SDC files with the write_sdc command, unless they are explicitly specified with the -mode argument.

Examples

Example 1

The following example disables timing mode m2.

```
[oasys-RTL]$ disable_timing_mode -modes m2
```

Related Topics

Timing Commands

get_timing_modes

report_timing_mode

set_timing_mode

get_slack

Returns the slack value for a design object such as a path group or a specific pin, or for the entire design.

Usage

get_slack [-tns | -pin *string* | -group *path_group*] [-mode *mode*]

Arguments

- -tns | -pin *string* | -group *path_group*

  Specifies the slack measurement to query.

  - -tns - Returns the total negative slack for the design.
  - -pin *string* - Returns the slack associated with the selected pin.
  - -group *path_group* - Returns the slack associated with the selected path group.
  - none specified - Returns the Worst Negative Slack (WNS).

- -mode *mode*

  Specifies the timing mode for which slack is reported.

Description

Reports the slack values corresponding with the specified arguments. You can configure the time units with the time_units_for_reports parameter.

Examples

The following examples use a single-clock design with a default path_group.

Get the WNS for the design:

```
% get_slack
-125.8
```

Gets the TNS for the design:

```
% get_slack -tns
-10635.2
```

Gets the worst slack on the pin *mul/dpath/mulcore/out_dff/i_0_103/A2* on the critical path:

```
% get_slack -pin mul/dpath/mulcore/out_dff/i_0_103/A2
-125.8
```

Gets the worst slack for the path_group default:

```
% get_slack -group default
-125.8
```

Related Topics

Timing Commands

get_timing_paths

report_path_groups

report_timing

time_units_for_reports

get_timing

Returns timing information or components that impacts timing.

Usage

get_timing ***type pin_name*** [-mode *mode*] [-rise | -fall] [-launch_clock *clock*] [-bidir {in | out}]

Arguments

- ***type***

  Specifies the type of timing information to retrieve, which must be one of the following types: arrival, cap, case_analysis, clocks, constant, end_point_slack, fanout, pin_cap, required, slack, transition, wire_cap, or wire_delay.

- ***pin_name***

  Specifies the name of a pin or pin object.

- -mode *mode*

  Specifies the mode.

- -rise | -fall

  Specifies whether the timing is for the rising or falling edge.

- -launch_clock *clock*

  Specifies the name of the launch clock.

- -bidir {in | out}

  Specifies if the timing needs to be retrieved from the input or the output part of a bidirectional pin.

Examples

Example

This example outputs the delay for the specified pin:

```
[oasys-RTL]$ get_timing arrival sparc7/test_stub/global_shift_enable
201ps
```

Example

This example gets the fanout of a pin:

```
[oasys-RTL]$ get_timing fanout scdata1/header/so
1
```

Example

This example gets the capacitance of a pin in units specified by the capacitance_units_for_reports parameter:

```
[oasys-RTL]$ get_timing cap sparc7/test_stub/global_shift_enable
173999.72ff
```

Related Topics

Timing Commands

report_timing

get_timing_modes

Returns a list of all timing modes.

Usage

get_timing_modes

Arguments

None.

Examples

The following example shows that two timing modes have been created for the design: fast and normal.

```
% get_timing_modes
fast normal
```

Related Topics

Timing Commands

disable_timing_mode

read_sdc

report_timing_mode

set_timing_mode

get_timing_paths

Returns a list of timing paths.

Usage

get_timing_paths [{-from | -rise_from | -fall_from} *from_list*] [-through *through_list*]
[-rise_through *through_list*] [-fall_through *through_list*]
[{-to | -rise_to | -fall_to} *to_list*] [-group *group*] [-max_paths *count*]

Arguments

- {-from | -rise_from | -fall_from} *from_list*

  Reports timing for paths that start at any given object in the list. The <from_list> argument is a list of clocks, pins, or instances. These paths can start at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from). Zero or one of -from, -rise_from, or -fall_from arguments can be used at once. If an instance is specified, the source can be any output of that instance.

- -through *through_list*

  Reports timing for paths through any given object in the list. The <through_list> argument is a list of pins, nets, or instances. Multiple -through, -rise_through, or -fall_through arguments can be specified, where the report applies to any path that goes through at least one object in each <through_list> in order that they are specified.

- -rise_through *through_list*

  Reports timing for paths through rising edge paths in the list. The <through_list> argument is a list of pins, nets, or instances. Multiple -through, -rise_through, or -fall_through arguments can be specified, where the report applies to any path that goes through at least one object in each <through_list> in order that they are specified.

- -fall_through *through_list*

  Reports timing for paths through falling edge paths in the list. The <through_list> argument is a list of pins, nets, or instances. Multiple -through, -rise_through, or -fall_through arguments can be specified, where the report applies to any path that goes through at least one object in each <through_list> in order that they are specified.

- {-to | -rise_to | -fall_to} *to_list*

  Reports timing for paths that end at any given object in the list. The <to_list> argument is a list of clocks, pins, or instances. Zero or one of arguments -to, -rise_to, or -fall_to can be used at once.

- -group *group*

  Specifies that the report only contains timing paths of the specified <group>. Groups can be defined using the group_path command.

- -max_paths *count*

  Specifies the maximum number of paths to report. The default is 1.

Examples

Get the timing paths from A1 through (B1 or B2) to C1 and report their attributes.

```
[oasys-RTL]$ report_attribute [get_timing_paths -from A1 \
   -through {B1 B2} -to C1]
```

Related Topics

Timing Commands

report_timing

get_timing

report_attribute

get_timing_precision

Returns the value of the internal timing scale set using the set_timing_precision command.

Usage

get_timing_precision

Arguments

None.

Description

Returns the integer value set for internal timing precision set using the set_timing_precision command. The default value is 1.

Examples

The following example returns the timing scale, in this case the default value:

```
% get_timing_precision
1
```

Related Topics

set_timing_precision

read_db

write_db

remove_case_analysis

Removes a previously defined case analysis for timing.

Usage

remove_case_analysis {**-all** | ***pin_list***} [-comment "*comment*"]

Arguments

- { **-all** | ***pin_list*** }

  Specifies whether to remove all previously defined timing case analyses, or only those associated with pins on the provided *pin_list*. The *pin_list* must be a Tcl list of pin names or pin objects.

- -comment "*comment*"

  Specifies a string used for commenting the operation.

Description

Removes a timing case analysis. Specify either -all to remove all timing cases or a list of pins to remove cases for a specified group of pins.

Examples

```
% remove_case_analysis -all
```

Related Topics

Timing Commands

report_case_analysis

set_case_analysis

remove_clock_uncertainty

Removes previously set clock skew characteristics.

Usage

remove_clock_uncertainty [-setup *clock_list*] [-all] [{-from | -rise_from | -fall_from} *from_list*] [{-to | -rise_to | -fall_to} *to_list*] [{-rise | -fall} *clock_list*]

Arguments

- -setup

  Removes uncertainty only related to setup checks. This is the default.

- -all

  Removes uncertainty for all properties on all clocks.

- {-from | -rise_from | -fall_from} *from_list*

  Removes uncertainty associated with the source clocks specified in *from_list*. You can remove the uncertainty for rise time (-rise_from), fall time (-fall_from), or both (-from).

- {-to | -rise_to | -fall_to} *to_list*

  Removes uncertainty associated with the destination clocks specified in *to_list*. You can remove the uncertainty for rise time (-rise_to), fall time (-fall_to), or both (-to).

- {-rise | -fall} *clock_list*

  Removes uncertainty for the rising or falling edge of the specified destination clock(s).

Examples

```
% remove_clock_uncertainty -from PHI1 -to PHI1
```

Related Topics

Timing Commands

set_clock_uncertainty

remove_driving_cell

Removes a previously specified driving cell for the specified pins.

Usage

remove_driving_cell ***pin_list*** [-min | -max] [-clock *clock_name*] [-clock_fall] [-rise | -fall]

Arguments

- ***pin_list***

  Specifies a list of pin names or pin objects in the current design. The driving cell is removed from each object in the *pin_list*. To specify more than one object, enclose the objects in quotes ("") or braces ({}).

- -min | -max

  Removes driving cell information for analysis at the minimum or maximum operating condition only. Default: max.

- -clock *clock_name*

  Specifies the clock from which the specified driving cell is removed.

- -clock_fall

  Specifies to remove the driving cell for the falling edge of the reference clock. The default is the positive edge.

- -rise | -fall

  Specifies to remove the driving cell for the rising or falling transition on specified ports.

Examples

Remove a previously specified driving cell from pin B:

```
% remove_driving_cell B
```

Related Topics

Timing Commands

set_driving_cell

remove_input_delay

Removes the input delay for the specified ports.

Usage

remove_input_delay ***port_pin_list*** [-min | -max] [-clock *clock_name*] [-clock_fall] [-level_sensitive] [-rise | -fall]

Arguments

- ***port_pin_list***

  Specifies a list of ports, pin names or pin objects in the current design. The input delay is removed from each object in the *port_pin_list*. To specify more than one object, enclose the objects in quotes ("") or braces ({}).

- -min | -max

  Unsupported and ignored.

- -clock *clock_name*

  Unsupported and ignored.

- -clock_fall

  Unsupported and ignored.

- -level_sensitive

  Unsupported and ignored.

- -rise | -fall

  Unsupported and ignored.

Examples

```
% remove_input_delay [get_ports "IN1"]
```

Related Topics

Timing Commands

remove_output_delay

remove_input_transition

Removes the specified input transition on the specified pin list.

Usage

remove_input_transition ***pin_list***

Arguments

- ***pin_list***

  Specifies a list of pin names or objects in the current design. The input transition is removed from each object in the <pin_list>.

Examples

Remove the input_transition on pin A1.

```
% remove_input_transition A1
```

Related Topics

Timing Commands

set_input_transition

remove_load

Removes loads on the specified pin list.

Usage

remove_load ***pin_list*** [-comment *string*]

Arguments

- ***pin_list***

  Specifies a list of pin names in the current design. The load is removed from each object in the <pin_list>.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

Remove the load on pin OUT2:

```
% remove_load [get_pins "OUT2"]
```

Related Topics

Timing Commands

set_load

remove_output_delay

Removes output delay ports.

Usage

remove_output_delay **<*port_pin_list*>** [-min | -max] [-clock *clock_name*] [-clock_fall] [-level_sensitive] [-rise | -fall]

Arguments

- <port_pin_list>

  Specifies a list of port or pin names in the current design. The output delay is removed from each object in the *port_pin_list*. To specify more than one object, enclose the objects in quotes ("") or braces ({}).

- -min | -max

  Unsupported and ignored.

- -clock *clock_name*

  Unsupported and ignored.

- -clock_fall

  Unsupported and ignored.

- -level_sensitive

  Unsupported and ignored.

- -rise | -fall

  Unsupported and ignored.

Examples

```
% remove_output_delay [get_ports "OUT1"]
```

Related Topics

Timing Commands

set_output_delay

remove_input_delay

remove_timing_exceptions

Removes timing exceptions.

Usage

remove_timing_exceptions [-type *string*] [{-setup | -hold}] [-modes *mode_list*] [-from *pin_list/src_clock_list*] [-through *pin_list*] [-to *pin_list/target_clock_list*]

Arguments

- -type *string*

  Specifies the type of exception to remove. (Default value: all)

  Legal values: {all, false_path, multicycle_path, min_max_delay, group_path, reset_path}

- -setup | -hold

  Remove only setup or hold exceptions. If you do not specify this argument, both setup and timing exceptions are removed.

- -modes *mode_list*

  Remove timing exceptions associated with the specified timing modes. If you do not specify this option, timing exceptions from all modes are removed.

- -from *pin_list/src_clock_list*

  Specifies a list of “from” pins or a list of source clocks.

- -through *pin_list*

  Specifies a list of “through” pins.

- -to *pin_list/target_clock_list*

  Specifies a list of “to” pins or a list of target clocks.

Description

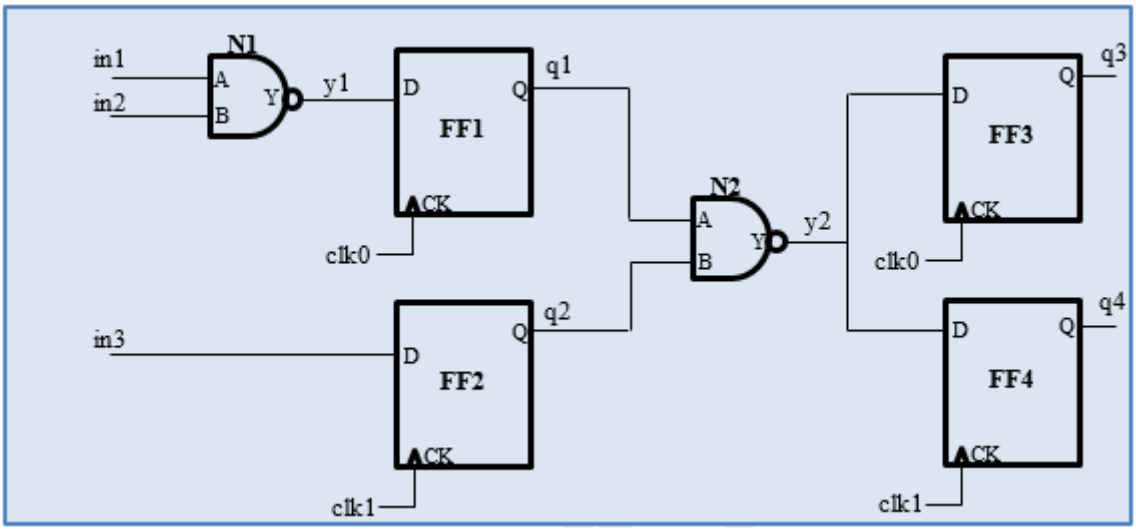
Removes timing exceptions. Timing exceptions can include false paths, multicycle paths, min/max delays, path groups, and reset_path.

Examples

Example 1

Consider the following design netlist and SDC file.

Figure 11-1. Design Netlist 1




```
## SDC File 1 ##
create_clock -period 6 -waveform {0 3} -name clk0 [get_ports clk0]
create_clock -period 10 -waveform {0 5} -name clk1 [get_ports clk1]
set_input_delay 0.5 -clock clk0 [get_ports {in1 in2}]
set_input_delay 0.5 -clock clk1 [get_ports {in3}]
----
----
##Exceptions##
set_false_path -from [get_clocks clk1] -to [get_clocks clk0]
(#Exc-Id: 1)
set_multicycle_path -setup 3 -from [get_pins FF2/CK] -to [get_pins FF4/D]
(#Exc-Id: 2)
set_multicycle_path -hold 2 -from [get_pins FF2/CK] -to [get_pins FF4/D]
(#Exc-Id: 3)
set_max_delay -setup 2 -to [get_pins FF1/D]
(#Exc-Id: 4)
```

The following scenarios demonstrate the behavior of the remove_timing_exceptions command:

Scenario 1

Remove all exceptions which are passing through pin N2/Y.

```
[oasys-RTL]$ remove_timing_exceptions -through [get_pins N2/Y]
  'Three' exceptions (Id: #1, #2 & #3) will be deleted.
```

Scenario 2

Remove all exceptions ending at pin FF3/D.

```
[oasys-RTL]$ remove_timing_exceptions -to [get_pins FF3/D]
  'Two' exceptions (Id: #2 & #3) will be deleted.
```

Scenario 3

Remove all exceptions targeted for clk1.

```
[oasys-RTL]$ remove_timing_exceptions -to [get_clocks clk1]
  'Two' exceptions (Id: #2 & #3) will be deleted.
```

Scenario 4

Remove exceptions starting from in1 port.

```
[oasys-RTL]$ remove_timing_exceptions -from [get_ports in1]
  'Single' exception (Id: #4) will be deleted.
```

Scenario 5

Remove all setup exceptions.

```
[oasys-RTL]$ remove_timing_exceptions -setup
  'Three' exceptions (Id: #1, #2 & #4) will be deleted.
```

Related Topics

Timing Commands

report_timing_exceptions

reset_timing

Resets the internal timing calculation created in the database.

Usage

reset_timing

Arguments

None.

Examples

```
[oasys-RTL]% reset_timing
```

Related Topics

Timing Commands

set_domain_operating_conditions

Sets the operating conditions for the specified power domain.

Usage

set_domain_operating_conditions **-domain** ***power_domain*** **-library** ***library_name*** ***operating_condition***

Arguments

- **-domain** ***power_domain***

  Specifies the name of an existing power domain.

- **-library** ***library_name***

  Specifies the library attached to the domain.

- ***operating_condition***

  Specifies the name of the operating condition defined in the library.

Related Topics

Timing Commands

set_dont_retime

Specifies register instances that should not be retimed.

Usage

set_dont_retime ***register_instance_list*** [true | false]

Arguments

- ***register_instance_list***

  Specifies the register instances that are not to be retimed.

- true | false

  Specifies not to retime the registers in *register_instance_list*. Default: true.

- false

  Resets the default value of the set_dont_retime command to false. This means that the objects in the <register_instance_list> are not modified by default.

Description

Specifies register instances in the argument should not be retimed. This command is used in conjunction with the set_retime_module and retime commands. This command should be issued after the synthesize command and before the optimize command so the optimizer and timer can handle retiming the modules accordingly. If the instances specified in the <register_instance_list> are not found, they are ignored.

Examples

Specify that all registers in mod1, are retimed, except for the reg_a register.

```
% synthesize

% set_retime_module mod1
% set_dont_retime mod1Instance/reg_a

% read_sdc file.sdc

% optimize

% retime
```

Related Topics

Timing Commands

retime

set_retime_module

set_retime_module

Identifies the modules that can be retimed.

Usage

set_retime_module ***module_name*** [true | false]

Arguments

- ***module_name***

  Specifies the modules to retime.

- true | false

  Specifies whether to retime the modules. Default: true.

Description

Identifies the modules that can be retimed. All registers within the boundary of the module are retimed, unless specified by the set_dont_retime command. Both the set_retime_module and set_dont_retime commands should be issued after the synthesize command and before the optimize command so that the optimizer and timer can handle retiming the modules accordingly.

Examples

Specifies that all registers in the mod1 module are retimed, except for the reg_a register.

```
% synthesize

% set_retime_module mod1

% set_dont_retime mod1Instance/reg_a

% read_sdc file.sdc
% optimize
% retime
```

Related Topics

Timing Commands

retime

set_dont_retime

set_timing_mode

Sets the timing mode of the design.

Usage

set_timing_mode ***timing_mode***

Arguments

- ***timing_mode***

  Sets the timing mode of the design to the specified <timing_mode>.

Description

Creates or sets the timing mode of the design to the specified *timing_mode*. Creates the timing mode if it doesn't already exist. You can apply further timing constraints to this mode by using the source or read_sdc commands until you specify another mode. The read_sdc -mode option also sets the timing mode.

Examples

This example sets the timing mode of the design to fast.

```
% set_timing_mode fast
% read_sdc fast1.sdc
% read_sdc fast2.sdc
```

This example sets the timing mode of the design to standby.

```
% set_timing_mode standby
% read_sdc standby.sdc
```

This example sets the timing mode of the design to fast using the read_sdc -mode option.

```
% read_sdc -mode fast fast1.sdc
% read_sdc -mode fast fast2.sdc
% read_sdc -mode standby standby.sdc
```

Related Topics

Timing Commands

read_sdc

write_sdc

set_timing_precision

Sets the internal timing scale.

Usage

set_timing_precision ***timing_scale***

Arguments

- ***timing_scale***

  Sets the internal timing scale. (default value: 1)

  Legal values: {1,10, 100, 1000}.

Description

The value specified by the *timing_scale* parameter modifies the internal timing scale and enables you to provide larger values to clock periods, cell delays, and transitions. This is primarily to support all types of designs ranging from slow to fast. This value must be set before loading the libraries, and is not permitted to change the precision value midway through the flow. The value is also stored in the Oasys-RTL database exported using the write_db command. The Oasys-RTL tool will report an error when there is a conflict of precision values between the library and design databases during reloading.

Note

The default precision in the Oasys-RTL tool is 0.1 picoseconds (ps). This is the smallest delay number that can be represented and reported. The value specified by the set_timing_precision command is used to scale this precision. For example, specifying a value of 10 means the smallest delay number that can be represented will be 1 picosecond. As the smallest delay is increased, the allowable maximum delay is also increased.

Examples

The following example sets the precision values to 10, 100 and 1000:

```
% set_timing_precision 10
info:    precision scale is changed to '10'  [TA-146]

% set_timing_precision 100
info:    precision scale is changed to '100'  [TA-146]

% set_timing_precision 100
info:    precision scale is changed to '1000'  [TA-146]
```

Related Topics

get_timing_precision

read_db

write_db

total_negative_slack

Returns the sum of negative slack values.

Usage

total_negative_slack [-group *group*] [-mode *name*]

Arguments

- -group *group*

  Specifies that the command calculates the total negative slack of the specified *group*. You can define groups with the group_path command.

- -mode *name*

  Specifies to report total negative slack for the given mode. If you do not specify a mode, returns the total negative slack of all violating endpoints in the design.

Description

Returns the absolute value of the sum of all negative slack values. If there are no paths with negative slack, 0 is reported. You can configure time units with the time_units_for_reports parameter.

Examples

```
% total_negative_slack
29317
```

Related Topics

Timing Commands

group_path

time_units_for_reports

worst_slack

Returns the worst slack value in the current design.

Usage

worst_slack [-group *group*] [-mode *name*]

Arguments

- -group *group*

  Specifies that the worst slack is output for the specified *group*.

- -mode *name*

  Specifies to report worst slack for the given mode. If you do not specify a mode, returns the worst slack in the design.

Examples

```
% worst_slack
-50ps

% worst_slack -group grp1
18.5ps
```

Related Topics

Timing Commands

Chapter 12
DFT Commands

The DFT commands control design for test (DFT) operations.

Table 12-1. DFT Commands

Command	Description
compress_scan_chains	Compresses any uncompressed scan chains by partitioning the long chains into smaller chains and connecting them to compression logic via muxes in the scan path.
config_tessent	Configures Tessent settings, including library, database, design ID, and executable path.
config_tessent_scan	Configures Tessent scan chain insertion.
config_tessent_tpi	Configures Tessent test point insertion.
connect_clock_gating_test_pin	Connects the clock gating test pin.
connect_scan_chains	Connects all scan-mapped flip-flops that pass DFT rules into scan chains.
current_dft_partition	Sets the current partition to the specified DFT partition or returns the current partition.
define_dft_partition	Divides the design into multiple partitions for localizing scan connection related operations.
define_scan_chain	Defines the scan chain for a DFT design. Internal lockups are inserted automatically when necessary.
define_scan_model	Defines a scan model for a DFT design. The model defines a mini scan chain that is included in a larger scan chain.
define_tessent_scan_mode	Defines a Tessent scan mode for scan chains
define_test_clock	Defines a pin or port as the test clock for a DFT-enabled design.
define_test_pin	Defines the test pins for a DFT design that must be controlled to a specific value during test.
delete_dft	Removes all DFT information from the database.
fix_dft_violations	Fixes DFT DRC violations on clocks and asynchronous signals.
get_scan_cells_of_chain	Returns a list of the names of the cells on the scan chain.

Table 12-1. DFT Commands (cont.)

Command	Description
get_scan_chains	Returns the number of scan chains in the design.
get_test_clocks	Returns the names of the ports or pins that have been defined as test clocks with the define_test_clock command.
get_test_pins	Returns the names of the ports or pins that have been specified by the define_test_pin command.
import_tsdb	Imports DFT data from TSDB directory of Tessent RTL run.
infer_shift_registers	Identifies shift registers for scan-stitching.
insert_dft_wrapper	Inserts DFT wrapper cells to control and observe specified pins in test mode.
remove_dft_partition	Removes an existing DFT partition.
remove_tessent_config	Removes all common configurations for Tessent TPI and Scan specified with config_tessent command.
remove_scan_chain	Removes an existing scan chain.
remove_tessent_scan_mode	Removes the specified tessent scan mode
remove_tessent_scan_config	Removes all Tessent Scan-related configurations specified with the config_tessent_scan command.
remove_tessent_tpi_config	Removes all Tessent TPI related configurations which were specified with the config_tessent_tpi command.
remove_test_clock	Removes an existing test clock.
remove_test_pin	Removes an existing test pin.
report_tessent_scan_modes	Lists currently defined Tessent scan modes
reset_dft_partition	Resets the current DFT partition to the default partition.
run_tessent_scan	Run Tessent gate-level DFT scan insertion.
run_tessent_tpi	Run Tessent gate-level DFT test point insertion.
set_dont_scan	Sets the dont_scan attribute on the specified registers.
set_equivalent_test_clocks	Specifies that test clocks are considered as identical and mutually skew-free.
set_exclude_scan_instance	Specifies instances not to connect to a scan chain.
trace_scan_chains	Traces and verifies the data connectivity of the scan chains.

compress_scan_chains

Compresses any uncompressed scan chains by partitioning the long chains into smaller chains and connecting them to compression logic via muxes in the scan path.

Usage

compress_scan_chains [-scan_chains <scan_chains>] [-compression_enable <pin>]
{[-scan_in <scan_in_port_pin>] [-scan_out <scan_out_port_pin>]}
| {[-max_length <length> | -compression_ratio <compression_ratio>
| -chain_count <number>} [-input <pin_port>] [-output <pin_port>] [-inside <instance>]

Arguments

- -scan_chains <scan_chains>

  Compresses the specified list of scan chains. If you do not specify a list of scan_chains, the tool selects all chains.

- -compression_enable <pin>

  Specifies the signal that enables compression. The <pin> must have been defined using the define_test_pin command with an active compression_mode value.

- -scan_in <scan_in_port_pin>

  Specifies the scan-in port pin to drive the compressed chains. Drive pins on pre-instantiated decompressor.

- -scan_out <scan_out_port_pin>

  Specifies the scan-out port pins for the compressed chains. Loads pins on pre-instantiated compressor.

- -max_length <length>

  Specifies the maximum length of the compressed chain.

- -compression_ratio <compression_ratio>

  Specifies the ratio between the longest uncompressed and compressed chains.

- -chain_count <number>

  Specifies the number of compressed chains.

- -input <pin_port>

  Specifies input pins to drive the decompressor. By default, the scan-ins of the chains under compression are used.

- -output <pin_port>

  Specifies output pins to get data from the compressor. By default, the pins are multiplexed with the scan-out of the chains under compression.

- -inside <instance>

  Specifies to insert the compression logic in the specified instance. The default is the top module.

Description

Compresses any uncompressed scan chains by partitioning the long chains into smaller chains and connecting them to compression logic using muxes in scan path. You can use a pre-instantiated compressor from a third party ATPG vendor (use Model 1), or let the Oasys-RTL tool insert an XOR based compression (use Model 2).

Mode 1 - Third-party compression

When hooking up to a third party pre-instantiated compression structure use the following options:

```
compress_scan_chains -scan_in -scan_out …
```

Mode 2 - Third-party compression

When creating Oasys-RTL compression, use the following options:

```
compress_scan_chains -max_length -compression_ratio -chain_count -input
   -output -inside …
```

To specify a compression configuration when inserting compression logic, use only one of the following three options:

- -max_length
- -compression_ratio
- -chain_count

An error is issued if you specify options for both model 1 and model 2.

To compress scan chains, first connect chains in non-compression mode and then compress them into smaller chains.

The general command flow for compressing scan chains is listed as follows:

```
define_test_pin
define_test_clock
[connect_clock_gating_test_pin]
check_dft
define_scan_chains
connect_scan_chains (uncompressed mode)
compress_scan_chains (compressed mode)
```

You can compress a chain multiple times and hook it up to different compression logic in different compression modes. For example, to connect chains to two different compression

blocks (compA and compB) in two different modes (controlled by signal ce1 and ce2), enter the following commands:

```
compress_scan_chains -scan_in compA/si -scan_out compA/so \
   -compression_enable ce1

compress_scan_chains -scan_in compB/si -scan_out compB/so
   -compression_enable ce2
```

The Oasys-RTL tool automatically splits the uncompressed chains into two different compression configurations and inserts the necessary muxes controlled by the ce1 and ce2 signals.

Examples

Example 1

The following commands show examples of scan chain compression using third party pre-instantiated compression structures.

```
% compress_scan_chains -scan_in {decomp/isi} -scan_out {comp/iso} \
   -compression_enable se

% compress_scan_chains -scan_in {u1/p0 u1/p1 u1/p2} \
   -scan_out {u1/s0 u1/s1 us/s2} -chains {chain1 chain2} \
   -compression_enable se
```

Example 2

The following commands demonstrate Oasys-RTL compression.

```
% compress_scan_chains -ratio 20 -compression_enable se

% compress_scan_chains -max_length 100 -compression_enable se

% compress_scan_chains -inside u1 -compression_enable se \
   -chain_count 1000 -chains {chain1 chain2} -input {i1 i2 i3} \
   -output {o1 o2 o3}
```

Related Topics

DFT Commands

define_test_pin

define_scan_chain

config_tessent

Configures Tessent settings, including library, database, design ID, and executable path.

Usage

config_tessent [-library *name*] [-rtl_tsdb *path*] [-design_id *id*] [-ignore_clock_gating {on|off}] [-ignore_dft_violations {on|off}] [-exec_path *path*] [-report]

Arguments

- -library *name*

  Specifies the Tessent cell library to load in Tessent for scan or test point insertion. This option must be specified prior to issuing run_tessent_tpi or run_tessent_scan.

- -rtl_tsdb *path*

  Specifies path to an existing Tessent Shell DataBase (TSDB) that is read by the OasysRTL-generated Tessent script.

- -design_id *id*

  Specifies the Design ID to load from the TSDB directory. The id value must be specified if the RTL-TSDB path was specified by the user.

- -ignore_clock_gating {on|off}

  Overrides DRC checks for blocking clock-gates as part of run_tessent_tpi/run_tessent_scan by evaluating the following conditions:

  - If a clock-gater has a test-enable pin tied to 0/1/x/u or disconnected, then the clock-gate is not reported as a blocker. However, a clock-gate with a test enable pin connected to an uncontrollable internal pin or primary is reported as a blocker.
  - If a clock-gater has no test enable pin and it's enable pin is not controllable, then it is reported as a blocker.

  By default, Oasys-RTL reports clock gates as blocking clock gates.

- -ignore_dft_violations {on|off}

  Overrides DRC checks performed as part of run_tessent_tpi/run_tessent_scan. By default, if check_dft is not clean, then Oasys-RTL will not proceed with these commands. Specify this argument only when you are aware of all violations being ignored.

- -exec_path *path*

  Specifies path of Tessent executable. This option must be specified prior to executing the run_tessent_tpi/run_tessent_scan commands.

- -report

  Specifies to report common configurations for Tessent scan and test point insertion.

Description

Configures Tessent path and settings, along with TSDB settings.

Examples

The following example displays the common Tessent configuration:

```
% config_tessent -report
Report Tessent Common Configuration:
-----+---------------------+-------------------------------------------
     |Option               |Value
-----+---------------------+-------------------------------------------
1    |library              |LIBRARY/Tessent/final.tcelllib
2    |rtl_tsdb             |DATA/tsdb_outdir
3    |design_id            |Scan_0
4    |exec_path            |TESSENT_HOME/bin/tessent
5    |ignore_clock_gating  |on
6    |ignore_dft_violations|off
-----+---------------------+-------------------------------------------
```

Related Topics

config_tessent_scan

config_tessent_tpi

import_tsdb

config_tessent_scan

Configures Tessent scan chain insertion.

Usage

config_tessent_scan
[-pre_analysis_mode_config_file *path_to_config_file*]
[-pre_analyze_config_file *path_to_config_file*]
[-pre_analyze_xbounding_config_file *path_to_config_file*]
[-pre_analyze_wrapper_cells_config_file *path_to_config_file*]
[-pre_insert_test_logic_config_file *path_to_config_file*]
[-xbounding_enable *pin_name*]
[-xbounding_connect_to {existing_scan_cell|new_scan_cell|constant_value}]
[-xbounding_scan_drive_limit *integer*]
[-xbounding_mcp_bounding_enable *pin_name*]
[-xbounding_exclude_sdc_cross_domain_paths {off|on}]
[-xbounding_test_clock {off|on}]
[-xbounding_exclude *pin/port_list*]
[-xbounding_use_scan_cells {off|on}]
[-xbounding_bound_clock_gater_enables {off|on}]
[-do_xbounding {off|on}]
[-xbounding_force_analyze {off|on}]
[-scanin_port_format *string*]
[-scanout_port_format *string*]
[-scan_chain_count *integer*]
[-scan_chain_length *integer*]
[-mix_clock_edges {off|on}]
[-report]
[-do_wrapper_analysis {off|on}]
[-wrapper_cell_force_analyze {on|off}]
[-dedicated_wrapper_cells_enable {on|off|auto}]
[-dedicated_wrapper_cells_ports <port_list>]
[-dedicated_wrapper_cells_input_module <library_model>]
[-dedicated_wrapper_cells_output_module <library_model>]
[-dedicated_wrapper_clock_source <pin/port_list>]
[-dedicated_wrapper_cells_location {inside_power_isolation | outside_power_isolation}]
[-input_fanout_flop_threshold <*integer*>]
[-input_fanout_libcell_levels_threshold <*integer*>]
[-output_fanin_flop_threshold <*integer*>]
[-output_fanin_libcell_levels_threshold <*integer*>]
[-non_scan_depth_threshold <*integer*>]
[-register_ports_exceeding_non_scan_depth {on|off}]
[-register_undriven_output {on|off}]
[-allow_internal_segments_as_wrapper {on|off}]
[-pipeline_start_of_wrapper_chains {on|off}]

[-exclude_ports <port_list>]
[-clear_excluded_ports]
[-register_ports_reaching_blackboxes {bbox_instance_list|all|off}]

Arguments

- -pre_analysis_mode_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before switching to Analysis Mode in Tessent. If specified, this file is sourced by the Oasys-generated Tessent script.

- -pre_analyze_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before issuing the analyze_scan_chains command. If specified, this file is sourced by the Oasys-generated Tessent script.

- -pre_insert_test_logic_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before issuing the insert_test_logic command. If specified, this file is sourced by the Oasys-generated Tessent script.

- -pre_analyze_xbounding_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before issuing the analyze_xbounding command. If specified, this file is sourced by the Oasys-generated Tessent script.

- -pre_analyze_wrapper_cells_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before issuing the analyze_wrapper_cells command. If specified, this file is sourced by the Oasys-generated Tessent script.

- -xbounding_enable *pin_name*

  Optionally specifies the name of the xbounding enable signal. This argument value maps to the Tessent command:

  ```
  set_xbounding_options -xbounding_enable <pin_name>
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_connect_to {existing_scan_cell|new_scan_cell|constant_value}

  Maps to the Tessent command:

  ```
  set_xbounding_options -connect_to
  {existing_scan_cell|new_scan_cell|constant_value}
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_scan_drive_limit *integer*

  Maps to the Tessent command:

  ```
  set_xbounding_options -scan_drive_limit <integer>
  ```

  A value of 0 indicates no limit. This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_mcp_bounding_enable *pin_name*

  Maps to the Tessent command:

  ```
  set_xbounding_options -mcp_bounding_enable <pin_name>
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_exclude_sdc_cross_domain_paths {off|on}

  Maps to the Tessent command:

  ```
  set_xbounding_options -exclude_sdc_cross_domain_paths {off|on}
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_test_clock {off|on}

  Maps to the Tessent command:

  ```
  set_xbounding_options -test_clock {off|on}
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_exclude *pin/port_list*

  Maps to the Tessent command:

  ```
  set_xbounding_options -exclude <pin/port_list>
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_use_scan_cells {off|on}

  Maps to the Tessent command:

  ```
  set_xbounding_options -use_scan_cells {off|on}
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -xbounding_bound_clock_gater_enables {off|on}

  Maps to the Tessent command:

  ```
  set_xbounding_options -bound_clock_gater_enables {off|on}
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -do_xbounding {off|on}

  Maps to the Tessent command:

  ```
  analyze_xbounding
  ```

- -xbounding_force_analyze {off|on}

  Modifies the analyze_xbounding command with the following argument:

  ```
  -force
  ```

  This argument only takes effect when specified with the argument:

  ```
  -do_xbounding on
  ```

- -scanin_port_format *string*

  Maps to the Tessent command:

  ```
  set_scan_insertion_options -si_port_format <string>
  ```

- -scanout_port_format *string*

  Maps to the Tessent command:

  ```
  set_scan_insertion_options -so_port_format <string>
  ```

- -scan_chain_count *integer*

  Maps to the Tessent command:

  ```
  set_scan_insertion_options -chain_count <integer>
  ```

- -scan_chain_length *integer*

  Maps to the Tessent command:

  ```
  set_scan_insertion_options -chain_length <integer>
  ```

- -mix_clock_edges {off|on}

  Maps to the Tessent command:

  ```
  set_scan_insertion_options -single_clock_edge_chains {on|off}
  ```

  Note

  Specifying '-mix_clock_edges off' translates to '-single_clock_edge_chains on' and vice versa.

- -report

  Displays all scan chain configurations set with the config_tessent_scan command.

- -do_wrapper_analysis {off|on}

  Specifies whether wrapper analysis needs to be done before scan chain is inserted by run_tessent_scan. If set to on, OasysRTL adds the following Tessent command to the generated script::.

  ```
  analyze_wrapper_cells
  ```

- -wrapper_cell_force_analyze on|off

  Optional. Directs Tessent to forcefully issue the 'analyze_wrapper_cells' command. In order to avoid using invalidated data, Tessent will delete all the scan modes and scan chain families. In the generated Tessent script, OasysRTL will recreate the scan modes and scan chain families as needed after the 'analyze_wrapper_cells -force' command. By default, this switch is set to false.

  ```
  analyze_wrapper_cells -force
  ```

- -dedicated_wrapper_cells_enable on|off|auto

  Specify on/off/auto to infer dedicated wrapper cells on all or specified primary IO of current design. (defaults to Tessent's default value "auto")

  ```
  set_dedicated_wrapper_cell_options  {on|off|auto}
  ```

- -dedicated_wrapper_cells_ports <port_list>

  Specify ports on which dedicated wrapper cells should be inferred. (ports specification)

  ```
  set_dedicated_wrapper_cell_options on -ports <port_list>
  ```

- -dedicated_wrapper_cells_input_module <library_model>

  Specifies a library model to be used to instantiate a dedicated Input wrapper cell.

  ```
  set_dedicated_wrapper_cell_options on -input_module_name
  <library_model>
  ```

- -dedicated_wrapper_cells_output_module <library_model>

  specifies a library model to be used to instantiate a dedicated output wrapper cell.

  ```
  set_dedicated_wrapper_cell_options on -output_module_name
  <library_model>
  ```

- -dedicated_wrapper_clock_source <*pin_port_list*>

  specifies an internal pin or top level port to be used as the clock source for the inserted dedicated wrapper cells

  ```
  set_dedicated_wrapper_cell_options on -clock_source <port_pin_list>
  ```

- -dedicated_wrapper_cells_location {inside_power_isolation | outside_power_isolation}

  specifies where to insert the Input and Output dedicated wrapper cells with relation to the encountered power isolation cells.

  - inside_power_isolation - insert Input dedicated wrapper cells after the encountered power isolation cells and the Output dedicated wrapper cells before the encountered power isolation cells. This is the default
  - outside_power_isolation - insert Input wrapper cells immediately after the PI ports being registered and insert the dedicated Output wrapper cells immediately before the PO ports being registered.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -dedicated_wrapper_cells_location
  {inside_power_isolation|outside_power_isolation}
  ```

- -input_fanout_flop_threshold <*integer*>

  specifies the maximum number of sequential elements allowed to be reached during the forward tracing from a PI. The default is 32.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -input_fanout_flop_threshold <integer>
  ```

- -input_fanout_libcell_levels_threshold <*integer*>

  specifies the maximum number of levels of library cells to be traversed during the backward tracing from a PI until the first sequential element is reached. The default is 32.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -input_fanout_libcell_levels_threshold
  <integer>
  ```

- -output_fanin_flop_threshold <*integer*>

  specifies the maximum number of sequential elements allowed to be reached during the backward tracing from a PO. The default is 32.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -output_fanin_flop_threshold <integer>
  ```

- -output_fanin_libcell_levels_threshold <*integer*>

  specifies the maximum number of levels of library cells to be traversed during the backward tracing from a PO until the first sequential element is reached. The default is 32.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -output_fanin_libcell_levels_threshold
  <integer>
  ```

- -non_scan_depth_threshold <*integer*>

  controls the depth of non_scan sequential elements traced by the tool. When the threshold is exceeded, the behavior depends on the -register_ports_exceeding_non_scan_depth switch. The default is 2.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -non_scan_depth_threshold <integer>
  ```

- -register_ports_exceeding_non_scan_depth {on|off}

  controls the behavior of the tool when the non-scan depth threshold is exceeded. If it is set to on, a dedicated wrapper cell will be added to the port that is being traced. If it is set to off, the branch that exceeds the threshold will be ignored and the tracing will continue to search for shared wrapper cell candidates.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option
      -register_ports_exceeding_non_scan_depth {<on|off}
  ```

- -register_undriven_output {on|off}

  specifies whether an output port having no functional source should receive a dedicated wrapper cell.

  - on - undriven outputs will receive dedicated wrapper cells except for specified ports for whom dedicated wrapper cell inference is turned off.
  - off - undriven output ports will not receive any wrapper cells except for specified ports for whom dedicated wrapper cell inference is turned on.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -register_undriven_output {on|off}
  ```

- -allow_internal_segments_as_wrapper {on|off}

  specifies whether to identify the defined sub-chains as Input or Output wrapper elements during the forward and backward tracing from the Primary IOs.

  - on - identify encountered sub-chains as Input or Output wrapper elements. This is the default.
  - off - do not identify encountered sub-chains as wrapper elements. Ports which reach sub-chains will have a dedicated wrapper cell inferred unless these ports are specified to not allow inferring dedicated wrapper cells for them.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option
     -allow_internal_segments_as_wrapper {on|off}
  ```

- -pipeline_start_of_wrapper_chains {on|off}

  specifies whether to automatically insert an at-speed test flop at the beginning of each wrapper chain.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option
     -pipeline_start_of_wrapper_chains {on|off}
  ```

- -exclude_ports port_list

  specifies that all ports in port_spec will be excluded from wrapper analysis without regard to their previous state (off/on/auto). Ports can only be removed from exclusion with -clear_excluded_ports switch.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -exclude_ports <port_list>
  ```

- -clear_excluded_ports {on|off}

  clears the list of ports that have been excluded from wrapper analysis. These ports are now included in wrapper analysis and will receive shared, dedicated, or no wrapper cells based on future and/or global dedicated wrapper cell configurations

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option -clear_excluded_ports
  ```

- -register_ports_reaching_blackboxes {bbox_instance_list|all|off}

  directs the tool to add a dedicated wapper cell to ports connected to the specified blackbox instances. The argument choices are as follows:

  - blackbox_instances_list - Adds dedicated wrapper cells to ports connected to any blackbox instance in the list.
  - all - Adds dedicated wrapper cells to ports connected to any blackbox instance.
  - off - A literal that disables this feature. This is the default.

  Maps to the following Tessent command:

  ```
  set_wrapper_analysis_option
   -register_ports_reaching_blackboxes {<bbox_instance_list>|all|off}
  ```

Description

The config_tessent_scan command configures scan chain insertion for the Tessent scan run. The run_tessent_scan command writes out the corresponding Tessent commands to the Tessent dofile for execution. You can find full descriptions of the Tessent commands and arguments in the *Tessent® Shell Reference Manual*.

Note

Any Tessent scan settings not explicitly set with config_tessent_scan assume the default value set by Tessent.

Examples

Example 1

The following example displays the Tessent Scan configuration:

```
% config_tessent_scan -report
Report Tessent Scan Configuration:
-----+----------------------------------------+-------
     |Option                                  |Value
-----+----------------------------------------+-------
1    |pre_analysis_mode_config_file           |
2    |pre_analyze_scan_config_file            |
3    |pre_insert_test_logic_config_file       |
4    |scanin_port_format                      |SI_%d
5    |scanout_port_format                     |SO_%d
6    |scan_chain_count                        |     20
7    |scan_chain_length                       |   5000
8    |mix_clock_edges                         |off
9    |pre_analyze_xbounding_config_file       |
10   |do_xbounding                            |on
11   |xbounding_force_analyze                 |
12   |xbounding_enable                        |
13   |xbounding_connect_to                    |
14   |xbounding_scan_drive_limit              |
15   |xbounding_mcp_bounding_enable           |
16   |xbounding_exclude_sdc_cross_domain_paths|
17   |xbounding_test_clock                    |
18   |xbounding_exclude                       |
19   |xbounding_use_scan_cells                |
20   |xbounding_bound_clock_gater_enables     |
21   |dedicated_wrapper_cells_enable          |
22   |dedicated_wrapper_cells_ports           |
23   |dedicated_wrapper_cells_input_module    |
24   |dedicated_wrapper_cells_output_module   |
25   |dedicated_wrapper_cells_clock_source    |
26   |pre_analyze_wrapper_cells_config_file   |
27   |wrapper_cell_force_analyze              |off
28   |do_wrapper_analysis                     |on
29   |output_fanin_flop_threshold             |
30   |input_fanout_flop_threshold             |
31   |output_fanin_libcell_levels_threshold   |
32   |input_fanout_libcell_levels_threshold   |
33   |non_scan_depth_threshold                |
34   |register_ports_exceeding_non_scan_depth |
35   |register_undriven_output                |
36   |allow_internal_segments_as_wrapper      |
37   |pipeline_start_of_wrapper_chains        |
38   |dedicated_wrapper_cells_location        |
39   |exclude_ports                           |
40   |clear_excluded_ports                    |
41   |register_ports_reaching_blackboxes      |
-----+----------------------------------------+-------
```

Related Topics

config_tessent

config_tessent_tpi

import_tsdb

config_tessent_tpi

Configures Tessent test point insertion.

Usage

config_tessent_tpi [-pre_analysis_mode_config_file *path_to_config_file*]
[-pre_analyze_config_file *path_to_config_file*]
[-pre_insert_test_logic_config_file *path_to_config_file*]
[-notest_points *list*]
[-notest_points_control_only {off|on}]
[-notest_points_observe_only {off|on}]
[-notest_points_path_file *path_to_critical_path_info_file*]
[-test_point_type {lbist_test_coverage|edt_pattern_count|lbist_test_coverage edt_pattern_count}]
[-total_number *integer*|integer%]
[-num_control_points *integer*|integer%]
[-num_observe_points *integer*|integer%]
[-control_points *integer*|integer%]
[-observe_points *integer*|integer%]
[-test_coverage_target *percentage*]
[-pattern_count_target *integer*]
[-max_control_points_per_path *integer*]
[-exclude_cross_domain_paths {on|off}]
[-observe_primary_outputs {on|off}]
[-exclude_clock_domains {off|*object_spec*}]
[-allow_in_xbounding_regions {on|off}]
[-assume_clock_controllability {ideal|default}]
[-control_point_enable *pin_name*]
[-observe_point_enable *pin_name*]
[-observe_point_share *integer*]
[-test_point_clock {system_clocks|always_tclk|tclk_as_needed}]
[-prefer_scan_cells {on|off}]
[-control_points *pin/port_spec*]
[-control_points_type AND|OR}]
[-control_points_clock_pin *pin/port_spec*]
[-observe_points *pin/port_spec*]
[-observe_points_clock_pin *pin/port_spec*]
[-report]

Arguments

- -pre_analysis_mode_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before switching to Analysis Mode in Tessent. This file is executed from the Oasys-generated Tessent script.

- -pre_analyze_config_file *path_to_config_file*

  Specifies a custom script file containing Tessent commands to execute immediately before analyzing test points. This file is executed from the Oasys-generated Tessent script.

- -pre_insert_test_logic_config_file *path_to_config_file*

  Specifies a file containing Tessent commands to execute immediately before inserting test logic. This file is executed from the Oasys-generated Tessent script.

- -notest_points *list*

  Specifies a list of paths to pins and/or instances to be excluded from test logic insertion. These objects could be excluded to control points, observe points, fixing bi-directionals, set/reset, xbounding, and clock lines. Maps to the Tessent command:

  ```
  add_notest_points <list>
  ```

- -notest_points_control_only {off|on}

  Specifies that no control points will be added in the specified regions. If you specify the ‘on’ parameter, this switch modifies the Tessent command add_notest_points with the argument:

  ```
  -control_only
  ```

- -notest_points_observe_only {off|on}

  Specifies that no observation points will be added in the specified regions. If you specify the ‘on’ parameter, modifies the Tessent command add_notest_points with the argument:

  ```
  -observe_only
  ```

- -notest_points_path_file *path_to_critical_path_info_file*

  Specifies the pathname to a “Path Definition” file that contains the critical path information. If this switch is defined, the OasysRTL-generated script modifies the Tessent command add_notest_points with the following argument:

  ```
  -path <path_to_critical_path_info_file>
  ```

- -test_point_type {lbist_test_coverage|edt_pattern_count|{lbist_test_coverage edt_pattern_count} }

  Maps to the Tessent command:

  ```
  set_test_point_types
  {lbist_test_coverage|edt_pattern_count|{lbist_test_coverage
  edt_pattern_count} }
  ```

- -total_number *integer*|integer%

  Specifies the maximum number of test points to be inserted as an integer or integer percentage of the total number of registers in the design. By default, there is no maximum value. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -total_number <integer>|<integer%>
  ```

- -num_control_points *integer*|integer%

  Specifies the maximum number of control points to be inserted as an integer or integer percentage of the total number of registers in the design. By default, there is no maximum value. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -control_points_number
  <integer>|<integer%>
  ```

- -num_observe_points *integer*|integer%

  Specifies the maximum number of observe points to be inserted as an integer or integer percentage of the total number of registers in the design. By default, there is no maximum value. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -observe_points_number
  <integer>|<integer%>
  ```

- -test_coverage_target *percentage*

  Specifies the target fault coverage to be achieved by the number of random patterns. Tessent will insert test points until either the maximum number of test points is reached or the desired fault coverage with the target number of random patterns is achieved. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -test_coverage_target <percentage>
  ```

- -pattern_count_target *integer*

  Specifies the number of random patterns to be applied to the design. Fault coverage estimations will be done using this parameter as reference. Default is 100,000. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -pattern_count_target <integer>
  ```

- -max_control_points_per_path *integer*

  Specifies the maximum number of control points to be inserted on a single path. The default maximum number of control points per path is 5. If this switch is defined, the OasysRTL-generated script adds the following command to the Tessent script:

  ```
  set_test_point_analysis_options -max_control_points_per_path
  <integer>
  ```

- -exclude_cross_domain_paths {on|off}

  Maps to the Tessent command:

  ```
  set_test_point_analysis_options -exclude_cross_domain_paths
  {on|off}
  ```

- -observe_primary_outputs {on|off}

  Maps to the Tessent command:

  ```
  set_test_point_analysis_options -observe_primary_outputs {on|off}
  ```

- -exclude_clock_domains {off|*object_spec*}

  Maps to the Tessent command:

  ```
  set_test_point_analysis_options -exclude_clock_domains
  {off|<object_spec>}
  ```

- -allow_in_xbounding_regions {on|off}

  Maps to the Tessent command:

  ```
  set_test_point_analysis_options -allow_in_xbounding_regions
  {on|off}
  ```

- -assume_clock_controllability {ideal|default}

  Maps to the Tessent command:

  ```
  set_test_point_analysis_options -assume_clock_controllability
  {ideal|default}
  ```

- -control_point_enable *pin_name*

  Maps to the Tessent command:

  ```
  set_test_point_insertion_options -control_point_enable <path>
  ```

- -observe_point_enable *pin_name*

  Maps to the Tessent command:

  ```
  set_test_point_insertion_options -observe_point_enable <path>
  ```

- -observe_point_share *integer*

  Maps to the Tessent command:

  ```
  set_test_point_insertion_options -control_point_share <integer>
  ```

- -test_point_clock {system_clocks|always_tclk|tclk_as_needed}

  Maps to the Tessent command:

  ```
  set_test_point_insertion_options -test_point_clock
  {system_clocks|always_tclk|tclk_as_needed}
  ```

- -prefer_scan_cells {on|off}

  Maps to the Tessent command:

  ```
  set_test_point_insertion_options -prefer_scan_cells {on|off}
  ```

- -control_points *pin/port_spec*

  Maps to the Tessent command:

  ```
  add_control_points -location <pin/port_spec>
  ```

- -control_points_type {AND|OR}

  Modifies the add_control_points command with the following argument:

  ```
  -type {AND|OR}
  ```

- -control_points_clock_pin *pin/port_spec*

  Modifies the add_control_points command with the following argument:

  ```
  -clock <pin/port_spec>
  ```

- -observe_points *pin/port_spec*

  Maps to the Tessent command:

  ```
  add_observe_points -location <pin/port_spec>
  ```

- -observe_points_clock_pin *pin/port_spec*

  Modifies the add_observe_points command with the following argument:

  ```
  -clock <pin/port_spec>
  ```

- -report

  Displays all test point configurations set with the config_tessent_tpi command.

Description

The config_tessent_tpi command configures test-point options for the Tessent TPI run. The run_tessent_tpi command writes out the corresponding Tessent commands to the Tessent dofile for execution. You can find full descriptions of the Tessent commands and arguments in the *Tessent® Shell Reference Manual*.

Examples

Example 1

The following example displays the Tessent TPI configuration:

```
% config_tessent_tpi -report
Report Tessent TPI Configuration:
-----+---------------------------------+-------------------
     |Option                           |Value
-----+---------------------------------+-------------------
1    |pre_analysis_mode_config_file    |
2    |pre_analyze_config_file          |
3    |pre_insert_test_logic_config_file|
4    |test_point_type                  |lbist_test_coverage
5    |notest_points                    |
6    |notest_points_control_only       |
7    |notest_points_observe_only       |
8    |notest_points_path_file          |
9    |total_number                     |5%                 }
10   |control_points_number            |
11   |observe_points_number            |
12   |test_coverage_target             |
13   |pattern_count_target             |
14   |maximum_control_points_per_path  |
15   |exclude_cross_domain_paths       |
16   |observe_primary_outputs          |
17   |exclude_clock_domains            |
18   |allow_in_xbounding_regions       |
19   |assume_clock_controllability     |
20   |control_point_enable             |cpe
21   |observe_point_enable             |ope
22   |observe_point_share              |
23   |test_point_clock                 |
24   |prefer_scan_cells                |
25   |control_point_type               |
26   |control_point_clock_pin          |
27   |control_points                   |
28   |observe_point_clock_pin          |
29   |observe_points                   |
-----+---------------------------------+-------------------
```

Related Topics

config_tessent

config_tessent_scan

import_tsdb

connect_clock_gating_test_pin

Connects the clock gating test pin.

Usage

connect_clock_gating_test_pin [-test_pin <port_pin>] [-skip_pre_connected]
[-instance <list_of_instances>] [-exclude <list_of_instances>]

Arguments

- -test_pin <port_pin>

  Specifies a design port or pin. It can be a top-level port or a lower level instance pin.

- -skip_pre_connected

  Skips any clock gating cell that are already connected to a test pin.

- -instance <list_of_instances>

  Specifies to only connect the clock gating <list_of_instances>. It can be a hierarchical instance, in which case the command only connects the test pin of clock gating instances included in the specified hierarchical instance.

- -exclude <list_of_instances>

  Excludes the specified instances from any connect operations.

Examples

Connect the top-level clock gating test port named T1.

```
% define_test_pin -pin T1 -scan_mode 1

% connect_clock_gating_test_pin -test_pin T1
```

Related Topics

DFT Commands

define_scan_chain

define_test_pin

connect_scan_chains

Connects all scan-mapped flip-flops that pass DFT rules into scan chains.

Usage

connect_scan_chains [-skip_scan_enable] [-mix_clock_edges] [-physical]

Arguments

- -skip_scan_enable

  Skips connecting the scan enable for all the scan chains.

- -mix_clock_edges

  Specifies to mix clock edges in a scan chain.

- -physical

  Connects scan chains based on the physical location of the flip-flops. This option minimizes the scan wire length.

Description

Connects all scan-mapped flip-flops, that pass DFT rules, into scan chains. The chains are automatically balanced with internal lockup-insertion wherever necessary. This command must be entered after the optimize command.

Supports scan-mapping and scan-stitching for multi-bit registers Handles multi-bit registers by treating them as scan segments.

Examples

Connect all scan-mapped flops.

```
% read_library library.lib
% read_lef tech.lef
% read_verilog top.v

% set_dont_scan */*DScn*
% set_clock_gating_options -control_port CGT

% synthesize -map_to_scan -gate_clock
% read_sdc file.sdc

% define_test_pin -pin CGT -scan_mode 1
% define_test_pin -pin SE -scan_mode 1 -default_scan_enable -create_port
% define_test_clock -pin CK

% check_dft -verbose -auto_test_clocks -auto_test_pins

% optimize
% for {set i 0} {$i < 100} {incr i} {
% define_scan_chain -scan_in EDT/si[$i] -scan_out EDT/so[$i]
% }
% connect_scan_chains

% write_scandef top.scandef
% write_verilog top.v
```

Related Topics

DFT Commands

define_scan_model

define_test_pin

set_dont_scan

define_test_clock

define_scan_chain

synthesize

optimize

current_dft_partition

Sets the current partition to the specified DFT partition or returns the current partition.

Usage

current_dft_partition [<partition_name>]

Arguments

- <partition_name>

  Specifies the name of the DFT partition.

Description

Sets the current partition to the specified DFT partition to which subsequent commands are applied. By setting the current partition, you can define scan chains without the -partition option.

If no argument is specified, the command returns the name of the current partition. By default, the current partition is set to the default partition.

A DFT partition is a group of design instances in which you can create scan chains. Creating or removing a DFT partition does not modify the netlist.

Examples

This example returns the default partition because the current partition was not previously set to a specific partition.

```
% current_dft_partition
default
```

This example sets the current partition to the *part1* partition.

```
% current_dft_partition part1
part1
```

Related Topics

DFT Commands

define_scan_chain

define_dft_partition

remove_dft_partition

report_dft_partitions

reset_dft_partition

define_dft_partition

Divides the design into multiple partitions for localizing scan connection related operations.

Usage

define_dft_partition <partition_name> [-instances <inst_list>]

Arguments

- <partition_name>

  Specifies the name of the partition.

- -instances <inst_list>

  Specifies a list of instances that are included in the specified partition.

Description

Sections the design into multiple partitions for scan connection. Each DFT partition can be associated with its own set of scan chains by using the -partition option of the define_scan_chain command.

A DFT partition is a group of instances for scan chain purposes. Creating or removing a DFT partition does not modify the netlist.

The instances specified in the <inst_list> can be register instances or a hierarchical instances (in which case all scan registers under that hierarchical instance are included in the partition).

Examples

```
% define_dft_partition -instances {inst1 inst2 inst3} part1
```

Related Topics

DFT Commands

remove_dft_partition

define_scan_chain

report_dft_partitions

define_scan_chain

Defines the scan chain for a DFT design. Internal lockups are inserted automatically when necessary.

Usage

define_scan_chain [-name <name>] [-scan_in <scan_in_port_pin>]
[-scan_in_hookup <hookup_pin>] [-scan_out <scan_out_port_pin>]
[-scan_out_hookup <hookup_pin>] [-shared_out] [-out_enable <enable_pin>]
[-out_enable_active {0 | 1]} [-scan_enable <scan_enable_port_pin>]
[-max_length <length>] [-elements <list>] [-fixed_order] [-head_lockup [latch | flop]
-tail_lockup {latch | flop | auto} [-test_domain <clock_domain>]
[-partition <partition_name>] [-create_port]

Arguments

- -name <name>

  Specifies a name for the scan chain.

- -scan_in <scan_in_port_pin>

  A design scan-in port or pin. It can be a top-level port or a lower level instance pin.

- -scan_in_hookup <hookup_pin>

  Specifies the hookup point for connecting to scan_in.

- -scan_out <scan_out_port_pin>

  Specifies a design scan-out port or pin. It can be a top-level port or a lower-level instance pin.

- -scan_out_hookup <hookup_pin>

  Specifies the hookup point for connecting to scan-out.

- -shared_out

  Specifies that the scan_out pin is to be multiplexed or shared with the other connection.

- -out_enable <enable_pin>

  Specifies that the shared mux is controlled by the specified <enable_pin> and its indicated active polarity (out_enable_active).

- -out_enable_active {0 | 1}

  Specifies the polarity for out_enable.

- -scan_enable <scan_enable_port_pin>

  Specifies a design scan-enable port or pin. The object can be a top-level port or a lower-level instance pin. If scan_enable is not set, the tool uses the default_scan_enable object defined using define_test_pin.

- -max_length <length>

  Specifies the maximum length of a scan chain.

- -elements <list>

  Specifies a list of hierarchy names of flip-flops that are assigned to this chain. If this is omitted, the tool automatically assign flops to the chain.

- -fixed_order

  Specifies that the order of instances in the -elements list is not to be altered.

- -head_lockup [latch | flop]

  Specifies to insert a head lockup at the beginning of the chain. You can choose to specify a latch or flip-flop using the latch and flop keywords, respectively.

- -tail_lockup {latch | flop | auto}

  Specifies to insert a tail lockup at the end of the chain. You can choose to specify a latch or flip-flop using the latch and flop keywords, respectively. In case of auto, the tool automatically determines if a lockup is necessary and what type is inserted based on the last element of this chain, and the first element of the next chain.

- -test_domain <clock_domain>

  Specifies the test clock domain. This domain must be defined with the define_test_clock command.

- -partition <partition_name>

  Specifies the partition. The define_dft_partition command defines partitions. This option is used to section the design into DFT partitions.

- -create_port

  Creates scan-in and scan-out pins if they do not exist as top-level ports.

Examples

```
% define_scan_chain -name c_1 -scan_in si_1 -scan_out so_1 -create_port
```

Related Topics

DFT Commands

check_dft

define_dft_partition

set_dont_scan

connect_scan_chains

synthesize

define_scan_model

define_test_clock

define_test_pin

report_scan_chains

define_scan_model

Defines a scan model for a DFT design. The model defines a mini scan chain that is included in a larger scan chain.

Usage

define_scan_model [-name <name>] [-lib_cell <cell_name>] [-instance <instance>] ...
[-scan_in <scan_in_pin>] [-scan_out <scan_out_pin>] [-scan_enable <scan_enable_pin>]
[-scan_enable_active {0 | 1]} [-capture_clock <clock>] [-capture_edge {rise | fall}]
[-launch_clock <clock>] [-launch_edge {rise | fall}] [-length <length>] [-tail_lockup_latch]
[-tail_lockup_flop]

Arguments

- -name <name>

  Specifies an arbitrary name for the scan model.

- -lib_cell <cell_name>

  Specifies a library cell name. This value is applied to each <instance>.

- -instance <instance> ...

  Specifies instance names.

- -scan_in <scan_in_pin>

  Specifies the name of the scan-in pin for the model.

- -scan_out <scan_out_pin>

  Specifies the name of the scan-out pin for the model.

- -scan_enable <scan_enable_pin>

  Specifies the name of the scan-enable pin for the model.

- -scan_enable_active {0 | 1}

  Specifies the active state of the scan enable pin.

- -capture_clock <clock>

  Specifies the name of the capture clock. This applies to the first element of the scan chain.

- -capture_edge {rise | fall}

  Specifies the capture edge. This applies to the first element of the scan chain.

- -length <length>

  Specifies the length of the scan chain in the scan model.

- -launch_clock <clock>

  Specifies the name of the launch clock. This applies to the last element of the scan chain.

- -launch_edge {rise | fall}

  Specifies the launch edge. This applies to the last element of the scan chain.

- -tail_lockup_latch

  Specifies the synchronization element used in the scan chain is a latch.

- -tail_lockup_flop

  Specifies the synchronization element used in the scan chain is a flip-flop.

Examples

```
% define_scan_model -name chn7 -instance inst2 -scan_in si1 \
   -scan_out so1 -scan_enable SE1 -scan_enable_active 1 \
   -capture_clock clk1 -capture_edge 0 -launch_clock clk1 -launch_edge 0 \
   -length 22
```

Related Topics

DFT Commands

define_tessent_scan_mode

Defines a Tessent scan mode for scan chains

Usage

define_tessent_scan_mode **-name <string> -enable_pin <string>** [-max_length <integer>] [-edt_instances <list>] [-lockup_cell_type <flop | latch>] [-head_lockup_type <flop | latch>] [-tail_lockup_type <flop | latch>] [-head_cell_type <flop | latch>] [-tail_cell_type <flop | latch>] [-elements <list>] [-default_scan_mode <true | false>]

Arguments

- -name <string>

  Specifies the mode name. This argument is required.

- -enable_pin <string>

  Specifies the mode-enable pin. This switch is required. The mode-enable pin is expected to be previously declared as a test pin using the ‘define_test_pin’ command.

- -max_length <integer>

  Optional. Specifies the maximum length of scan chains under this defined mode.

- -edt_instances <list>

  Specifies the list of edt_instances to be associated with this mode. This switch is required for internal edt modes.

- -lockup_cell_type <flop | latch>

  Optional. Specifies the type of lockup to use between time-constrained scan elements of new scan chains.

- -head_lockup_type <flop | latch>

  Optional. Specifies the type of lockup to use between the scan-in pin and the first scan cell.

- -tail_lockup_type <flop | latch>

  Optional. Specifies the type of lockup to use between the last scan cell and scan-out pin.

- -elements <list>

  Optional. Specifies the list of scan elements to be associated with this mode.

- -default_scan_mode <true|false>

  Optional. Specifies this mode as the default scan mode.

Description

Defines a Tessent scan mode for scan chains.

Examples

The following example defines two scan modes (int_mode and ext_mode) and runs Tessent on the post-optimize design:

```
define_tessent_scan_mode -name int_mode \
              -enable_pin cpu_rtl1_tessent_tdr_sri_ctrl_inst/int_mode \
              -edt_instances {cpu_rtl1_tessent_edt_c1_inst}
define_tessent_scan_mode -name ext_mode -
              -enable_pin cpu_rtl1_tessent_tdr_sri_ctrl_inst/ext_mode \
              -max_length 20
define_scan_chain -scan_in si1 \
                  -scan_out so1 \
                  -mode ext_mode \
                  -create_port
define_scan_chain -scan_in si2 \
                  -scan_out so2 \
                  -mode ext_mode \
                  -create_port
report_tessent_scan_modes
config_tessent_scan -do_wrapper_analysis on \
                    -exclude_ports {si1 si2 so1 so2} \
                    -do_xbounding on
run_tessent_scan
```

Related Topics

remove_tessent_scan_mode

report_tessent_scan_modes

define_test_clock

Defines a pin or port as the test clock for a DFT-enabled design.

Usage

define_test_clock [-derived] [-hookup <hookup_pin>] [-inverted] [-name <name>] [-pin <port_pin>] [-test_domain <domain>]

Arguments

- -derived

  Optional. Derives the specific test clock domains automatically.

- -hookup <hookup_pin>

  Optional. Specifies the pin to use to make any connections to the specified clock.

- -inverted

  Optional. Specifies that the test clock has an inverse waveform. The waveform starts with a high value, then falls, and then rises in a clock period (1-0-1 ...). This is typically the case in test mode. This information is used when mixing scan flops with different clock edges into chains, and lockup latch insertion.

- -name <name>

  Optional. Specifies the name of the test clock.

- -pin <port_pin>

  Optional. Specifies a test clock port or pin. It can be a top-level port or a lower-level instance pin.

- -test_domain <domain>

  Optional. Specifies that the test clock is for this specific <domain>. If this option is not used, each test clock has its own test clock domain, or they are derived if -derived option is specified.

Description

Defines a test clock for a DFT design. The test clock is assumed to be controllable from a tester during a manufacturing test. It can be set on top-level ports or lower-level instance pins.

To make a particular flip-flop in the design scannable, it may be necessary to provide a different clock to the design during the test mode. This clock is referred to as the test clock and is used for scan loading and unloading as well as data capture. The original clock is muxed with the test clock.

Examples

Defines the test clock port to be TCK.

```
% define_test_clock -pin TCK
```

Related Topics

DFT Commands

remove_test_clock

define_scan_chain

set_dont_scan

define_test_pin

connect_scan_chains

report_test_clocks

write_scandef

synthesize

define_test_pin

Defines the test pins for a DFT design that must be controlled to a specific value during test.

Usage

define_test_pin [-name <string>] [-pin <port_pin>] [-hookup <pin>] [-scan_mode {0 | 1}] [-compression_mode {0 | 1}] [-mask_mode {0 | 1}] [-create_port] [-default_scan_enable]

Arguments

- -name <string>

  Optional. Specifies the name of the test pin.

- -pin <port_pin>

  Specifies a design test port or pin. It can be a top-level port or a lower-level instance pin.

- -hookup <pin>

  Specifies the connection point for a pin. As an example, you can define a test port as a test pin and an internal pin (such as a core-side of a pad cell) as its hookup point.

- -scan_mode {0 | 1}

  Specifies the value to which the pin is set to during scan operations.

- -compression_mode {0 | 1}

  Defines the active value of signal in compression mode.

- -mask_mode {0 | 1}

  Defines the active value of signal to mask off outputs during compression.

- -create_port

  Creates the port at the top level. Use this option if the pin does not exist and should be added to the top module.

- -default_scan_enable

  Indicates that the pin is a shift-enable pin for the chains that do not have a specific scan enable. The scan enable can be defined in the define_scan_chain command.

Examples

Example

Define a test pin CGT that is active high.

```
% define_test_pin -pin CGT -scan_mode 1
```

Example

Define a test pin called se that is active high, is a default scan enable, and create a pin at the top level.

```
% define_test_pin -pin SE -scan_mode 1 -default_scan_enable -create_port
```

Example

Define the test pin se and specify the hookup connection point pad_SE/z.

```
% define_test_pin -pin se -hookup pad_SE/z
```

Related Topics

DFT Commands

report_test_pins

remove_test_pin

set_dont_scan

define_scan_chain

define_test_clock

connect_scan_chains

synthesize

write_scandef

delete_dft

Removes all DFT information from the database.

Usage

delete_dft

Arguments

None.

Description

Removes all DFT information from the database.

Examples

```
delete_dft
```

fix_dft_violations

Fixes DFT DRC violations on clocks and asynchronous signals.

Usage

fix_dft_violations -test_control <test_enable> [-test_clock <clock_name>]
[-type {async | clock | all}]

Arguments

- -test_control <test_enable>

  Required. Specifies the test mode signal that controls the bypass logic during scan.

- -test_clock <clock_name>

  Required when you use the "-type clock" or "-type all" options. Specifies that the tool fixes violations on the specified test clock, <clock_name>.

- -type {async | clock | all}

  Optional. Specifies the type of DFT DRC violations to fix. The default is all.

Description

This command fixes two types of DFT violations: uncontrollable clocks and uncontrollable asynchronous set and reset signals. In the case that the clock, set, and reset pins of registers are uncontrollable at the primary inputs, the registers are excluded from scan chains. Exclusion from the scan chains results in decreased test coverage. Some examples of these violations include internally gated clocks or divider clocks that are generated from internal logic. The check_dft command reports these violations if they occur in your design.

An output message from the command indicates any new cells the tool added to fix DFT violations.

For information about using this command in the design flow, refer to the section "Auto-Fixing of DFT Design Rule Checking (DRC) Violations" in the *Oasys-RTL User's Guide*.

Examples

Example 1

This example fixes the DFT DRC violations of clock type using the *test_mode* signal.

```
% fix_dft_violations -type clock -test_clock core_clk \
   -test_control test_mode
Created 78 muxes to fix clock violation(s)
```

Example 2

This example fixes the DFT DRC violations of asynchronous type using the *test_mode* signal.

```
% fix_dft_violations -type async -test_control test_mode
Created 3 gates to fix Async violation(s)
```

To fix asynchronous signal violations, the tool uses a test mode signal (as opposed to the scan_enable) for the test_control pin. It uses the test mode signal since the asynchronous pin of the scan registers must be set to the inactive value during both the scan cycle and capture cycle.

Example 3

This example fixes all DFT DRC violations using the *test_mode* signal.

```
% fix_dft_violations -type all -test_clock core_clk \
   -test_control test_mode
Created 3 gates to fix Async violation(s)
Created 78 muxes to fix clock violation(s)
```

Related Topics

DFT Commands

check_dft

get_scan_cells_of_chain

Returns a list of the names of the cells on the scan chain.

Usage

get_scan_cells_of_chain -chain <scan_chain>

Arguments

- -chain <scan_chain>

  Specifies the name of the scan chain.

Examples

```
% get_scan_cells_of_chain chain1
```

Related Topics

DFT Commands

get_scan_chains

get_scan_chains

Returns the number of scan chains in the design.

Usage

get_scan_chains

Arguments

None.

Examples

```
% get_scan_chains
```

Related Topics

DFT Commands

get_scan_cells_of_chain

get_test_clocks

Returns the names of the ports or pins that have been defined as test clocks with the define_test_clock command.

Usage

get_test_clocks

Arguments

None.

Examples

The following example shows a test clock being defined on pin, gclk, using define_test_clock. Then, get_test_clocks returns the name of test clock pin.

```
% define_test_clock -pin gclk -test_domain A
   gclk

% get_test_clocks
   gclk
```

Related Topics

DFT Commands

get_test_pins

get_test_pins

Returns the names of the ports or pins that have been specified by the define_test_pin command.

Usage

get_test_pins

Arguments

None.

Examples

The following example defines test signals using the define_test_pin command. The get_test_pins command returns the names of the ports or pins of the test signals:

```
% define_test_pin -pin TEST_MODE -scan 1 -default_scan_enable -create_port
% define_test_pin -pin [get_pins spc_hdr/I0/sync_cluster_slave/so] \
   -scan_mode 1

% get_test_pins
   TEST_MODE
```

Related Topics

DFT Commands

define_test_pin

import_tsdb

Imports DFT data from TSDB directory of Tessent RTL run.

Usage

import_tsdb [-verbose]

Arguments

- -verbose

  Displays detailed information about the import.

Related Topics

config_tessent

config_tessent_scan

config_tessent_tpi

infer_shift_registers

Identifies shift registers for scan-stitching.

Usage

infer_shift_registers [-map [true | false]}]

Arguments

- -map [true | false]

  Specifies that only the head of the shift register be mapped to a scan flop. The rest of the registers in the shift register are to be unmapped to non-scan flops. If set to false, shift registers are identified but corresponding scan models are not created. The default is true if the -map option is not specified.

Description

When this command is applied, shift registers that are identified can be used as scan segments for scan-stitching purposes. After the identification, only the first register in the shift register, or head, is mapped to a scan flop and the remaining registers are mapped to non-scan flops. This way, the area overhead for the design due to DFT insertion is reduced. Potential congestion and wire length is improved by removing the external scan connections within the shift register segment. For DFT purposes, these shift register segments are modeled as a scan model and used as an entity during the scan chain connection.

The recommended place in the flow to apply this command is immediately after the “optimize -place_only” command. However, the command can be applied anywhere in the flow before the connect_scan_chains command.

Multi-bit mapping does not consider inferred shift registers. If you use infer_shift_registers prior to multi-bit mapping, the tool excludes the identified shift register registers from multi-bit mapping.

Related Topics

DFT Commands

connect_scan_chains

check_dft

optimize

insert_dft_wrapper

Inserts DFT wrapper cells to control and observe specified pins in test mode.

Usage

insert_dft_wrapper [-pins <pin_list>] [-exclude <pin_list>] [-clock <pin>] [-rising | -falling] [-enable <string>] [-active_high | -active_low] [-naming_style <string>] [-reuse_threshold <value>] [-depth_threshold <value>] [-shared]

Arguments

- -pins <pin_list>

  Specifies a list of pins where wrapper cells are to be inserted. Supports wild card matching in the pin names.

- -exclude <pin_list>

  Excludes the specified pins from the operation.

- -clock <pin>

  Specifies that the clock pin is used to clock the wrapper cell.

- -rising

  Specifies that a rising edge is used to clock the wrapper cell. The -rising and -falling options cannot be used together.

- -falling

  Specifies that a falling edge is used to clock the wrapper cell. The -falling and -rising options cannot be used together.

- -enable <string>

  Specifies the enable signal that controls the wrapper cell.

- -active_high

  Specifies that the enable signal is active high. The -active_high and -active_low options cannot be used together.

- -active_low

  Specifies that the enable signal is active low. The -active_low and -active_high options cannot be used together.

- -naming_style <string>

  Specifies a naming convention for the wrapper cells.

- -reuse_threshold <value>

  Specifies a threshold number of registers to determine the assignment of wrapper cells. If the number of registers of a port fanout is above the threshold value, a dedicated wrapper

cell is inserted. If the number of registers of a port fanout is below the threshold value, all registers in the fanout are converted to shared-wrapper cells. The default value is 1.

- -depth_threshold <value>

  Specifies the number of logic levels allowed between a port and the register it drives. This number includes single-input gates. The default value is 1.

- -shared

  Specifies that wrapper cells share the available flip-flops surrounding the logic that is being wrapped. By default, the tool creates dedicated wrapper cells.

Examples

This example defines a list of instances inserted for the wrapper. You can create a DFT partition on the instances and define separate scan chains for the partition to create input and output wrapper chains.

```
% set oWrapper [insert_dft_wrapper -pins [all_outputs] \
   -exclude {*scan_out*} -clock wck -rising -enable ob_isolate \
   -active_high -naming_style DFT_OB_%s_wrap_reg]

% define_dft_partition ow -instances $oWrapper

% define_scan_chain -name owChain -scan_in scan_in[1] \
   -scan_out scan_out[1] -shared -out_enable ce1 -out_enable_active 0 \
   -scan_enable se -partition ow
```

Related Topics

DFT Commands

define_scan_chain

define_dft_partition

remove_dft_partition

Removes an existing DFT partition.

Usage

remove_dft_partition <dft_partition_name>

Arguments

- <dft_partition_name>

  Specifies the name of a DFT partition.

Description

Removes the <dft_partition_name> created by define_dft_partition command. A DFT partition is a group of instances for DFT scan chain purposes. Creating or removing a DFT partition does not modify the netlist.

Examples

```
% remove_dft_partition part1
```

Related Topics

DFT Commands

define_dft_partition

remove_tessent_config

Removes all common configurations for Tessent TPI and Scan specified with config_tessent command.

Usage

remove_tessent_config

Arguments

None

Related Topics

DFT Commands

config_tessent

remove_scan_chain

Removes an existing scan chain.

Usage

remove_scan_chain <chain_name>

Arguments

- <chain_name>

  Specifies the name of a scan chain.

Examples

```
% remove_scan_chain chain1
```

Related Topics

DFT Commands

define_scan_chain

remove_tessent_scan_mode

Removes the specified tessent scan mode

Usage

remove_tessent_scan_mode <mode_name>

Arguments

- <mode_name>

  Specifies the name of the mode to remove.

Description

This command removes a previously defined scan mode. You can use the report_tessent_scan_modes command to list the currently defined scan modes.

Examples

The following examples lists the currently available scan modes and removes the ext3 mode. The report in the example is truncated for easier viewing.

```
report_tessent_scan_modes
# Report Tessent Scan Modes (:
----+----------+------------+----+--------------+ ...
#   | Name     |Chain Length|Type|Mode Enable   | ...
#---+----------+------------+----+--------------+ ...
# 1 |edt_mode  |default     |    |cpu/edt_mode  | ...
# 2 |multi_mode|default     |    |cpu/multi_mode| ...
# 3 |ext3      |default     |    |cpu/test_en   | ...
remove_tessent_scan_mode ext3
```

Related Topics

DFT Commands

define_tessent_scan_mode

report_tessent_scan_modes

remove_tessent_scan_config

Removes all Tessent Scan-related configurations specified with the config_tessent_scan command.

Usage

remove_tessent_scan_config

Arguments

None

Related Topics

DFT Commands

config_tessent_scan

remove_tessent_tpi_config

Removes all Tessent TPI related configurations which were specified with the config_tessent_tpi command.

Usage

remove_tessent_tpi_config

Arguments

None

Related Topics

DFT Commands

config_tessent_tpi

remove_test_clock

Removes an existing test clock.

Usage

remove_test_clock <test_clock_name>

Arguments

- <test_clock_name>

  Specifies the name of a test clock.

Examples

```
% remove_test_clock tclk1
```

Related Topics

DFT Commands

report_test_clocks

define_test_clock

remove_test_pin

Removes an existing test pin.

Usage

remove_test_pin <test_pin_name>

Arguments

- <test_pin_name>

  Specifies the name of a test pin.

Examples

```
% remove_test_pin SE
```

Related Topics

DFT Commands

report_test_pins

define_test_pin

report_tessent_scan_modes

Lists currently defined Tessent scan modes

Usage

report_tessent_scan_modes [-detail]

Arguments

- -detail

  Provide detailed information on the various scan modes

Description

Lists currently defined Tessent scan modes

Examples

The following example shows the information generated by this command. The last two columns in the report were truncated from the example due to page width.

```
report_tessent_scan_modes
Report Tessent Scan Modes:
--+----------+-------+----+------------------+--------+--------+------+
  |Name      |Chain  |Type|Mode              |EDT     |Scan    |Lockup
  |          |       |    |Enable            |instance|Elements| Type
--+----------+-------+----+------------------+--------+--------+------+
1 |edt_mode  |default|    |cpu_ctl/edt_mode  |cpu_edt |       0|      |
2 |multi_mode|default|    |cpu_inst/multi_mode|        |       0|      |
--+----------+-------+----+------------------+--------+--------+------+
```

reset_dft_partition

Resets the current DFT partition to the default partition.

Usage

reset_dft_partition

Arguments

None.

Description

A DFT partition represents a group of design instances within which you can create scan chains. In partition-based DFT flows, resets the current partition to the default partition so that you can implement a new set of DFT settings.

Examples

```
%  reset_dft_partition
info:    Current DFT partition is set to 'default'  [DFT-337]
```

Related Topics

DFT Commands

define_scan_chain

current_dft_partition

remove_dft_partition

report_dft_partitions

define_dft_partition

run_tessent_scan

Run Tessent gate-level DFT scan insertion.

Usage

run_tessent_scan [-base_dir *directory*]

Arguments

- -base_dir *directory*

  Specifies the base directory for writing the Tessent script.

Related Topics

DFT Commands

config_tessent

config_tessent_scan

import_tsdb

run_tessent_tpi

run_tessent_tpi

Run Tessent gate-level DFT test point insertion.

Usage

run_tessent_tpi [-base_dir *directory*]

Arguments

- -base_dir *directory*

  Specifies the base directory for writing the Tessent script.

Related Topics

DFT Commands

config_tessent

config_tessent_tpi

import_tsdb

run_tessent_scan

set_dont_scan

Sets the dont_scan attribute on the specified registers.

Usage

set_dont_scan <register_list> [true | false] [-verbose]

Arguments

- <register_list>

  Specifies a list of registers for which the dont_scan attribute will be applied or removed. The list is a space-separated Tcl list. This command supports glob-style pattern matching (*, ?).

- true

  Sets the dont_scan attribute on the specified objects. This is the default.

- false

  Removes the dont_scan attribute on the specified objects.

- -verbose

  Specifies that detailed information be returned when the command is applied.

Description

Sets the dont_scan attribute on the specified registers. Oasys-RTL maps these registers to non-scan registers during synthesis and optimization. This command applies only to registers that are inferred as flip-flops.

Examples

Mark registers with the string DnS in their names (in any module) as not to be mapped to scan.

```
% set_dont_scan {*/*DnS*}
```

The next example displays detailed information when the command is applied.

```
% set_dont_scan -verbose [get_cells  u1_a_reg6]
info: applied set_dont_scan = true to 1 instance(s)
```

The next example synthesizes with the -map_to_scan option, which maps all registers to be scanned. It then sets the dont_scan attribute on all registers matching dout*[1]. Oasys-RTL maps these registers to non-scan registers during optimization.

```
read_verilog test.v
synthesize -map_to_scan
set_dont_scan [get_cell dout*[1]]
optimize
```

Related Topics

DFT Commands

synthesize

set_equivalent_test_clocks

Specifies that test clocks are considered as identical and mutually skew-free.

Usage

set_equivalent_test_clocks <test_clocks>

Arguments

- <test_clocks>

  Specifies a list of test clocks that are considered to be mutually skew-free.

Description

Specifies that the given set of test clocks are considered as mutually skew-free and are mixed in a chain without any lockup latch between the scan flops clocked by any of the clocks in the set.

Examples

```
% set_equivalent_test_clock { tclk1 tclk2 }
```

Related Topics

DFT Commands

set_dont_scan

connect_scan_chains

set_exclude_scan_instance

Specifies instances not to connect to a scan chain.

Usage

set_exclude_scan_instance <instances> [true | false]

Arguments

- <instances>

  Specifies a list of instances that are not connected to a scan chain. This list does not support wildcards.

- true

  Sets the set_exclude_scan_instance attribute on the specified objects. This is the default.

- false

  Removes the set_exclude_scan_instance attribute on the specified objects.

Description

Specifies instances not to connect to a scan chain. The instance can be a leaf scan flop or a hierarchical instance, in which case all scan flops in the hierarchical instance and its lower child instances are excluded from the scan chain.

Examples

Exclude all flop instances within instance A1 from a scan chain:

```
% set_exclude_scan_instance A1
```

Related Topics

DFT Commands

connect_scan_chains

set_dont_scan

trace_scan_chains

Traces and verifies the data connectivity of the scan chains.

Usage

trace_scan_chains [-scan_in <scan_in_port_pin>] [-scan_out <scan_out_port_pin>] [-set_high <scan_enable> | -set_low <scan_enable>] [-verbose]

Arguments

- -scan_in <scan_in_port_pin>

  Specifies a design scan-in port or pin. It can be a top-level port or a lower-level instance pin.

- -scan_out <scan_out_port_pin>

  Specifies a design scan-out port or pin. It can be a top-level port or a lower-level instance pin.

- -set_high <scan_enable>

  Defines the <scan_enable> as active high.

- -set_low <scan_enable>

  Defines the <scan_enable> as active low.

- -verbose

  Outputs the elements from scan in to scan out.

Description

This command reestablishes the scan chain status in an Oasys-RTL session where the design has been loaded from a post scan-stitched netlist. Based on the scan input, scan output, and scan enable information, it traces and verifies the connectivity of the scan chain.

The DFT ports definition (test clocks, test pins, and so forth) must be provided prior to issuing this command.

Tracing is also supported for scan chains with multi-bit registers, which will be abstracted as scan segments.

Examples

Example

This example traces and verifies the connectivity of the scan chain from si_0 to so_0.

```
% trace_scan_chains -scan_in si_0 -scan_out so_0 -set_high se

Info: Successfully traced scan chain (scanIn: si_0, scanOut: so_0) with 11
elements)
```

Example

The following example traces and verifies the connectivity of the scan chain from si_0 to s0_0, showing each of the elements:

```
% trace_scan_chains -scan_in si_0 -scan_out so_0 -set_high se -verbose

Info: tracing chain from si_0 to so_0 1 tracing through alu/result_reg_1 2
tracing through alu/result_reg_0 3 tracing through rc/lockup_3 4 tracing
through rc/q_reg_1 5 tracing through rc/q_reg_0 6 tracing through rb/
lockup_2 7 tracing through rb/q_reg_1 8 tracing through rb/q_reg_0 9
tracing through ra/lockup_1 10 tracing through ra/q_reg_1 11 tracing
through ra/q_reg_0

Info: Successfully traced scan chain (scanIn: si_0, scanOut: so_0) with 11
elements)
```

Related Topics

DFT Commands

define_scan_chain

connect_scan_chains

Chapter 13
Parallel Equivalence Checking Commands

This verify command uses an external logic equivalence checking tool to verify that the synthesized netlist is logically equivalent to the original RTL.

Table 13-1. Parallel Equivalence Checking Commands

Command	Description
verify	Verifies the formal equivalence of the synthesized netlist to the original RTL, using a supported equivalence checker.

verify

Verifies the formal equivalence of the synthesized netlist to the original RTL, using a supported equivalence checker.

Usage

verify <netlist_file> [-base_directory <path>] [-dont_clean]
[-ignore_pg_pins <list>] [-library_directory <cell_library_path>]
[-time_limit <minutes>] [-suffix <verification_directory_suffix>]
[-write_scripts] [-execute_scripts <script_directory_path>] [-fpopt <formalpro_options>]

Arguments

- <netlist_file>

  Required. Specifies the gate-level description to be verified with respect to the RTL.

- -base_directory <path>

  Optional. Specifies the parent directory where the verification directory is created. If -base_directory is not specified, the default directory is /tmp/oasys.<pid>.

- -dont_clean

  Optional. Prevents the cleaning up of intermediate files generated in the verification cache directory.

- -ignore_pg_pins <list>

  Optional. A list of power and ground pins to ignore during verification. This option is useful if the Liberty simulation models contain power pins in the cell description and those pins are not used in the RTL.

- -library_directory <cell_library_path>

  Optional. Specifies the path to the simulation or functional models of the library cells. If not specified, by default the Liberty files are translated into a library directory inside the verification directory. This option may be used when the Liberty files do not change across verification runs, saving translation run time.

- -time_limit <minutes>

  Optional. Time limit, in minutes, for verification. The default limit is 240 minutes (4 hours).

- -suffix <verification_directory_suffix>

  Optional. Specifies the suffix to be used in the verification cache directory name. The default is the top-level module name of the design.

- -write_scripts

  Optional. Directs the tool to just create the verification setup directory without performing the actual verification. The verify command sequence to be executed for verification is also indicated at the end of generation.

- -execute_scripts <script_directory_path>

  Optional. Performs the verification from a verification setup generated with the -write_scripts option.

- -fpopt <formalpro_options>

  Specifies options to pass to the FormalPro command line.

Description

The verify command verifies the formal equivalence of the synthesized netlist to the original RTL, using a supported equivalence checker. It performs in-line verification of the user-supplied netlist against RTL used for synthesis or, optionally, generates the setup for launching the off-line verification in a different session. It is an integrated verification solution that is fully automated with little or no setup costs.

Caution

If you edit the netlist using the "Design Edit Commands" on page 266, the verify command can potentially report false differences.

Examples

Example 1: Verify a Gate-Level Netlist Against an RTL Design

The following example logically verifies the gate-level *netlist.v* netlist against the RTL design. It creates a verification setup directory in the *testdir* directory. If the top-level module name is *top*, the directory is named *testdir/verify_dir.top* in the current working directory.

```
[oasys-RTL]$ verify netlist.v -base_directory testdir
```

Example 2: Create a Setup Directory and Run Verification Scripts

This example creates a verification directory named verify_dir.opt in the current working directory before running the verification scripts. This directory is where the command logically verifies the gate level netlist "netlist.v" against RTL design. The functional definitions of the library cells are expected to be in the cell_lib directory; this includes a <cellname>.v file per library cell.

The maximum verification runtime is set to 60 minutes. The power and ground pins in the library cells, VDD and VSS respectively, are ignored during verification. None of the intermediate files in the directory are removed after verification because the -dont_clean switch is specified.

```
[oasys-RTL]$ verify netlist.v -write_scripts -base_directory [pwd]
   -suffix opt -library_directory cell_lib -time_limit 60 \
   -ignore_pg_pins {VDD VSS}

[oasys-RTL]$ verify -dont_clean -execute_scripts [pwd]/verify_dir.opt
```

Chapter 14
Design Space Exploration Commands

Design space exploration (DSE) commands allow you to vary different design metrics during synthesis and report the variations.

Caution

DSE commands are only supported when the Oasys-RTL tool is invoked with a DSE script in batch mode. The commands do not function if you attempt to apply them or source a DSE script when the tool is already running.

Table 14-1. Design Space Exploration Commands

Command	Description
explore	Specifies explore variables and corresponding settings to iterate when performing design space exploration.
scale_clock	Scales the clock period by a positive or negative percentage factor.
scale_utilization	Scales the utilization.
set_explore	Configures the design parameters for DSE.
set_explore_setup	Specifies the load sharing software command to launch DSE batch jobs, the CPU allocation, and the working directory for Oasys-RTL DSE batch jobs.

explore

Specifies explore variables and corresponding settings to iterate when performing design space exploration.

Usage

explore <explore_variable> '{' '{'<opt_1> '{'<cmd_1>'}' ['{'<opt_2> '{'<cmd_2>'}'] ... '}' '}'

Arguments

- <explore_variable>

  Specifies the name of an explore variable that was previously defined using the set_explore command.

- '{' '{'<opt_1> '{'<cmd_1>'}' ['{'<opt_2> '{'<cmd_2>'}'] ... '}' '}'

  Specifies a list of option-command pairs. The option value must be one of the values defined in the <list_of_options> argument of the set_explore command. The cmd_<*n*> is any Oasys-RTL input command. Each option and command pair must be enclosed in {} brackets. The full list of option and command pairs must also be enclosed in brackets.

Description

Specifies explore variables to iterate when performing design space exploration.

You first must define explore variables using the set_explore command, which specifies a variable name and a list of possible values. Next, these variables can be iterated using the explore command to create multiple Oasys-RTL runs.

As an example, you may want to create multiple Oasys-RTL runs that vary the voltage and the clock period using several combinations. You can compare metrics such as area, power, and worst case negative slack using the report_explore command.

Examples

The following example creates three explore variables named lib_vt, voltage, and clock_period. These variables are used to create 16 Oasys-RTL runs, varying the libraries (2x), voltages (2x), and clock period (4x).

```
set_explore -variable clock_period -options { 0.9ns 1.0ns 1.1ns 1.2ns }
set_explore -variable lib_vt -options { hvt lvt }
set_explore -variable voltage -options { 0.95v 0.85v }
…

# varying libraries - high vs low voltage threshold
explore lib_vt {
   hvt { set_target_library high_vt }
   lvt { set_target_library low_vt }
}
…

# varying voltages - .95 volt vs .85 volt UPF files
explore voltage {
   0.95v { load_upf $demo/constraints/cpu.95.upf }
   0.85v { load_upf $demo/constraints/cpu.85.upf }
}
…
# varying clock frequencies -
explore clock_period {
   0.9ns { scale_clock -all -10 }
   1.1ns { scale_clock -all 10 }
   1.2ns { scale_clock -all 20 }
}
…
```

Note

Oasys automatically generates results for the default clock period of 1.0ns.

Related Topics

Design Space Exploration Commands

set_explore

report_explore

scale_clock

scale_clock

Scales the clock period by a positive or negative percentage factor.

Usage

scale_clock {-clocks <clock_list> | -all} <scale_percentage>

Arguments

- -clocks <clock_list>

  Specifies a list of clocks to scale.

- -all

  Specifies that all clocks are scaled.

- <scale_percentage>

  Specifies the percentage that the clocks are scaled.

Examples

Create multiple runs while varying clock periods. The original clock period was 1.0 ns.

```
set_explore -variable clock_period -options {0.9ns 1.1ns 1.2ns }

create_clock -period 1.0 -name GCLK gclk

explore clock_period {
   0.9ns { scale_clock -all -10}
   1.1ns { scale_clock -all 10 }
   1.2ns { scale_clock -all 20 }
}
```

Related Topics

Design Space Exploration Commands

scale_utilization

explore

report_explore

set_explore

scale_utilization

Scales the utilization.

Usage

scale_utilization <scale_percentage>

Arguments

- <scale_percentage>

  Specifies the percentage as an integer that scales the utilization.

Examples

Create multiple runs while varying placement utilization. Assume that the original utilization was 60%.

```
set_explore -variable utilization -options { 50% 70% }

explore utilization {
   50% { scale_utilization -10 }
   70% { scale_utilization 10 }
}
```

Related Topics

Design Space Exploration Commands

scale_clock

report_explore

explore

set_explore

set_explore

Configures the design parameters for DSE.

Usage

set_explore -floorplan | {-variable <explore_variable> -options <list_of_labels>}

Arguments

- -floorplan

  Required if the -variable option is not used. Specifies the floorplan directive that enables automatic exploration of different floorplan strategies. This option can only be set once per script. This option cannot be used with the -variable option.

- -variable <explore_variable> -options <list_of_labels>

  Required if the -floorplan option is not used. Specifies the <explore_variable> name for the set of design scenario names listed in <list_of_labels>. You can specify the scenarios using the explore command. Only one variable can be defined in a single “set_explore -variable” command. Multiple “set_explore -variable” commands can be used in a single script. This option cannot be used with the -floorplan option.

  <explore_variable>

  Specifies the variable to define, for example, clock period, voltage threshold library, or operating voltage.

  -options <list_of_labels>

  Specifies the possible labels for the explore_variable. Each label value defines the set of specifications to use for the explore_variable option. Each specification is defined with a corresponding explore command.

Description

The set_explore command determines how DSE performs automatic exploration. Exploration approaches include choosing different preconfigured macro placement recipes, or changing the design parameter values (user-specified).

The set_explore command must be specified after the set_explore_setup command. The -floorplan and -variable arguments are mutually exclusive.

Each set_explore -variable command specifies one variable and its values only. For multiple explore variables, multiple “set_explore -variable” commands should be used.

Examples

Example 1: Configure DSE to Explore Clock Periods, Libraries, and Voltages

This example creates three explore variables named lib_vt, voltage, and clock_period. These variables are used to create 16 runs with varying libraries (2), voltages (2), and clock periods (4).

```
# specify explore_variables and their labels
set_explore -variable clock_period -options { 0.9ns 1.0ns 1.1ns 1.2ns }
set_explore -variable lib_vt -options { hvt lvt }
set_explore -variable voltage -options { 0.95v 0.85v }
…

# varying libraries - high vs low voltage threshold

explore lib_vt {
   hvt { set_target_library high_vt }
   lvt { set_target_library low_vt }
}
…

# varying voltages - .95 volt vs .85 volt UPF files
explore voltage {
   0.95v { load_upf $demo/constraints/cpu.95.upf }
   0.85v { load_upf $demo/constraints/cpu.85.upf } }
…

# varying clock frequencies
explore clock_period {
   0.9ns { scale_clock -all -10 }
   1.0ns { scale_clock -all 0 }
   1.1ns { scale_clock -all 10 }
   1.2ns { scale_clock -all 20 } }
…
```

Example 2: Scale the Clock Period by Different Percentages

This example creates a clock_period variable that scales the clock period by 10%, -10%, and -20%.

```
# specify explore_variables and their labels
set_explore -variable clock_period -options { 10% -10% -20% }

# varying clock frequencies
explore clockPeriod {
 10% scale_clock -all 10
-10% scale_clock -all -10
-20% scale_clock -all -20 }
```

Related Topics

Design Space Exploration Commands

explore

scale_clock

scale_utilization

report_explore

set_explore_setup

Specifies the load sharing software command to launch DSE batch jobs, the CPU allocation, and the working directory for Oasys-RTL DSE batch jobs.

Usage

set_explore_setup [-directory <directory>] [-job_limit <number>]
[-launch_command <command>] [-save_gui_views [true | false]]

Arguments

- -directory <directory>

  Optional. Specifies the root directory from which jobs are launched.

- -job_limit <number>

  Optional. Specifies the maximum number of CPUs to use in parallel.

- -launch_command <command>

  Optional. Specifies the command used to launch parallel Oasys-RTL runs when performing design space exploration.

- -save_gui_views [true | false]

  Optional. Specifies whether the tool saves the physical GUI views after a DSE run.

  true - The tool saves the GUI views after a DSE run. If the display cannot be opened, DSE exits with an error.

  false - The tool does not save the GUI views after a DSE run. No error occurs even if the display cannot be opened.

  The default is true.

Description

This command must be the first command in a DSE script.

With the -launch_command option you can specify the bsub command for the IBM® Platform LSF® or the qsub command for the Univa® Grid Engine®. By default, the tool runs parallel DSE jobs on the same machine using the rsh or ssh commands.

Note

set_explore_setup must be the first command in the DSE script; otherwise, it is ignored.

The tool saves physical GUI views after each DSE run by default. Set the -save_gui_views option false to not save GUI views.

Examples

Example 1: Configure DSE to Use the IBM Platform LSF

This example sets up DSE to use the bsub command.

```
% set_explore_setup -launch_command "/bin/bsub -E '//bin/linux-pre-exec' \
   -q linux -R 'select\[type==LINUX64\] rusage\[mem=7500\]'"
```

Example 2: Configure DSE to use the my.exploreS directory, Four CPUs, and the Univa Grid Engine

This example sets up DSE to use the */project/cpu/Me/my.explore* exploration directory and four CPUs using the qsub command for load sharing.

```
set_explore_setup -directory /project/cpu/Me/my.explore -job_limit 4 \
   -launch_command {qsub -V -l hostclass='n*|s*' -l \
   'mem_total=20G,mem_free=20G,pmem=20' -l h_rt=5:0:0 -P CpuTeam -w n}
```

Example 3: Configure DSE to Use the IBM Platform LSF Without Saving GUI Views

This example sets up DSE to use the IBM Platform LSF but not save the physical GUI views.

```
% set_explore_setup -launch_command "/bin/bsub -E '//bin/linux-pre-exec' \
   -q linux -R 'select\[type==LINUX64\] rusage\[mem=7500\]'" \
   -save_gui_views false
```

Related Topics

Design Space Exploration Commands

explore

set_explore

scale_clock

report_explore

scale_utilization

Chapter 15
SDC Commands

Oasys-RTL uses a set of SDC commands for defining constraints on the design. The SDC command set supported by the tool is listed below.

Table 15-1. SDC Commands

Command	Description
all_clocks	Returns a list of all defined clocks in the current design.
all_inputs	Returns a list of all input or bidirectional and output ports defined in the current design.
all_outputs	Returns a list of all output or bidirectional and output ports in the current design.
all_registers	Returns the sequential cells in the current design.
create_clock	Creates a clock and defines its waveform.
create_generated_clock	Creates a clock derived from another clock.
current_design	Shows or sets the current design.
current_instance	Sets the current instance.
group_path	Groups critical timing paths for optimization.
remove_path_group	Removes timing path groups from the current design.
report_sdc_status	Prints a summary table showing the pass/fail/ignore status of SDC/timing constraints read in. Warning columns indicate that a given constraint has partially passed. The table also indicates filenames for constraints read in.
reset_path	Resets a specified exception path to a normal single cycle path.
set_case_analysis	Specifies that a port or pin is at a constant logic value for timing analysis and optimization.
set_clock_groups	Specifies clock groups that are mutually exclusive or asynchronous with each other.
set_clock_latency	Specifies the clock network latency.
set_clock_sense	Specifies the clock sense. This command is used to specify the unateness of the clock network through a specific pin.
set_clock_transition	Sets clock transition times.

Table 15-1. SDC Commands (cont.)

Command	Description
set_clock_uncertainty	Specifies the clock skew characteristics.
set_disable_timing	Disables timing arcs in the design.
set_driving_cell	Specifies that a library cell or pin drives specified ports.
set_false_path	Specifies the paths to be ignored for timing analysis and optimization.
set_input_delay	Sets input delay on pins or input ports relative to a clock.
set_input_transition	Sets a fixed transition time on input or bidirectional ports.
set_load	Sets the capacitance load to a specified value on specified ports and nets.
set_logic_one	Sets a logic value of 1 to a list of ports.
set_logic_zero	Sets a logic value of 0 to a list of ports.
set_max_delay	Sets maximum delay for specified timing paths.
set_multicycle_path	Allows more than one clock cycle for designated paths.
set_operating_conditions	Sets the operating conditions.
set_output_delay	Sets output delay on pins or output ports relative to a clock.
set_sense	Specifies the signal sense (with respect to the clock source) propagating forward through a non-unate network from the specified pins.
set_timing_derate	Sets the derating factors for the timing paths.
set_units	Specifies the unit values for capacitance, resistance, time, voltage, current, and power, for the SDC commands.
set_wire_load_min_block_size	Specifies the minimum block area that is used for wire load selection on the current design.
set_wire_load_mode	Specifies how Wire Load Models are applied to calculate delay values on nets in hierarchical designs.
set_wire_load_model	Specifies the name of the wire load model to apply to the object.
set_wire_load_selection_group	Specify a selection group defined in the library to use in wire load estimation for delays and loading on nets.

Note

Min/Hold Timing Analysis Support: The timer of the Oasys-RTL tool does not perform early timing or hold time analysis; therefore, the -hold and -min options are ignored by the timer. However, the -hold and -min options in any SDC command are parsed, retained in the database and can be passed through to the output SDC with the write_sdc command.

all_clocks

Returns a list of all defined clocks in the current design.

Usage

all_clocks

Arguments

None.

Examples

```
% all_clocks
clk1 clk2
```

Related Topics

SDC Commands

create_clock

all_inputs

Returns a list of all input or bidirectional and output ports defined in the current design.

Usage

all_inputs [-level_sensitive] [-edge_triggered] [-clock *clocks*] [-comment *string*]

Arguments

- -level_sensitive

  This option is ignored.
- -edge_triggered

  This option is ignored.
- -clock *clocks*

  This option is ignored.
- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

```
% all_inputs
{DRAM0_DQ[127]} {DRAM0_DQ[126]} {DRAM0_DQ[125]} …
```

Related Topics

SDC Commands

all_outputs

all_outputs

Returns a list of all output or bidirectional and output ports in the current design.

Usage

all_outputs [-level_sensitive] [-edge_triggered] [-clock *clocks*] [-comment *string*

Arguments

- -level_sensitive

  This option is ignored.
- -edge_triggered

  This option is ignored.
- -clock *clocks*

  This option is ignored.
- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

Example

```
% all_outputs
DRAM0_RAS_L DRAM0_CAS_L DRAM0_WE_L {DRAM0_CS_L[3]} ...:1
```

Example

```
% set_false_path -to [all_outputs -clock TCLK]
```

Related Topics

SDC Commands

all_inputs

all_registers

Returns the sequential cells in the current design.

Usage

all_registers [-no_hierarchy] [-clock *clocks*] [-cells | -data_pins | -clock_pins | -output_pins] [-level_sensitive | -edge_triggered] [-rise_clock *rise_clock_name* | -fall_clock *fall_clocks*] [-slave_clock_pins] [-async_pins] [-master_slave] [-comment *string*]

Arguments

- -no_hierarchy

  Limits the output to only the current level of hierarchy. By default, this option is off.

- -clock *clocks*

  Considers only sequential cells clocked by <clocks> in the search. By default, this option is off.

- -cells | -data_pins | -clock_pins | -output_pins

  Specifies the type of objects to return.

  - -cells - Return a list of sequential cells that meet the search criteria. (default)
  - -data_pins - Returns a list of data pins of the sequential cells that meet the search criteria.
  - -clock_pins - Returns a list of clock pins of the sequential cells that meet the search criteria.
  - -output_pins - Returns a list of output pins of the sequential cells that meet the search criteria.

- -level_sensitive | -edge_triggered

  Specifies to return level-sensitive or edge-triggered cells.

- -rise_clock *rise_clock_name* | -fall_clock *fall_clocks*

  Outputs only registers driven by the rising or falling edge of the specified clocks. By default, this option is off.

- -slave_clock_pins

  This option is ignored.

- -async_pins

  Returns asynchronous pins such as ‘set’ and ‘reset’.

- -master_slave

  This option is ignored.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Description

The all_registers command returns a collection of sequential cells or pins in the current design, filtered as specified by the options. By default, the command returns a collection of all sequential cells in the design. If you specify the -clock argument, it considers only the sequential cells in the transitive fanout of the sources of the clock.

Examples

This example lists all edge triggered cells:

```
% all_registers -edge_triggered
```

Related Topics

SDC Commands

create_clock

Creates a clock and defines its waveform.

Usage

create_clock [-add] {***port_pin_list*** | **-name** ***clock_name***}
{ -period *period* | -waveform '{'*edge_list*'}' } [-comment *string*]

Arguments

- -add

  Adds a clock. Does not replace an existing clock.

- {*port_pin_list* | -name *clock_name*}

  Specifies the name

  Optional list of pins and ports to apply this clock. If no port or pins are specified, the -name option is required.

- -name *clock_name*

  The name of the clock. Without this option, the clock name defaults to the first pin in <port_pin_list>. If no port or pins are specified, this option is required.

- -period *period*

  The clock period, in library time units.

- -waveform '{'*edge_list*'}'

  The time of the rising and falling edges in library time units. If -waveform is not specified, but -period is, a default waveform with a rising edge of 0.0 and a falling edge of <period>/2 is assumed. The <edge_list> must be enclosed in {} brackets.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Description

Creates a clock with the specified period and waveform and applies it to the specified pins or ports. If no <clock_name> is given, the clock is named after the first pin from the <port_pin_list>.

Examples

The following examples demonstrate how a clock is created:

```
% create_clock -period 100 CK
% create_clock -period 100 -waveform {0 50} [get_ports {clk}]
% create_clock -period 10 -name CK1
```

Related Topics

all_clocks

create_liberty_clocks

SDC Commands

create_generated_clock

Creates a clock derived from another clock.

Usage

create_generated_clock [-add] [-name *clock_name*] -source *source_pin*
[-edges '{'*e1 e2 e3*'}'] [-combinational] [-divide_by *divider*]
[-multiply_by *multiplier*] [-edge_shift '{'*s1 s2 s3*'}'] [-duty_cycle *duty_cycle*]
[-invert] [-master_clock *master_clock*] [-comment *string*] *port_pin_list*

Arguments

- -add

  Adds the clock. Does not replace an existing clock.

- -name *clock_name*

  Specifies the name of the generated clock. If -name is not specified, the clock uses the same name as the first clock in <source_pin>. If you specify -add, you must use the -name option; clocks with the same source must have different names.

- -source *source_pin*

  Required argument that specifies the master clock pin from which the generated clock is derived.

- -edges '{'*e1 e2 e3*'}'

  Specifies a list of positive integers that represent the edges of the master clock that are used to create the generated clock edges. The edges are interpreted as alternating rising and falling edges. Each edge must not be less than the previous edge. Specify an odd number of edges, not less than 3, to make one full clock cycle of the generated clock. Only one of the following is allowed: -edges, -divide_by, -multiply_by, -duty_cycle.

- -combinational

  Indicates that the source clock latency includes only the delay through combinational paths. It does not include sequential elements such as latches and flop pins or other clocks.

- -divide_by *divider*

  Indicates that the source clock frequency gets divided by the divider argument. If the argument is 2, the period of the generated clock will be two times the source clock. Only one of the following is allowed: -edges, -divide_by, -multiply_by, -duty_cycle.

- -multiply_by *multiplier*

  Specifies the frequency multiplication factor. If the <multiplier> is 3, the period is one third as long as the master clock period. Exactly one instance of -multiply_by can be specified. Only one of the following is allowed: -edges, -divide_by, -multiply_by, -duty_cycle.

- -edge_shift '{'*s1 s2 s3*'}'

  Specifies a list of floating point numbers that represents the amount of shift, in library time units, that the specified edges have when generating the clock waveform. You must specify the same number of edge shifts as the number of edges specified with -edges. The shift value can be positive or negative, indicating a shift later or earlier in time, respectively.

- -duty_cycle *duty_cycle*

  Specifies the duty cycle as a percentage. Only one of the following is allowed: -edges, -divide_by, -multiply_by, -duty_cycle.

- -invert

  Specifies to invert the generated clock.

- -master_clock *master_clock*

  Specifies the master clock to be used for this generated clock, if multiple clocks fan into the master pin.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

- *port_pin_list*

  Specifies a list of pins, ports, or nets to receive this clock application.

Description

Creates a clock derived from a master clock at specified *source_pin* and applies it to specified pins. If no *clock_name* is specified, the clock is named after the first pin from the *port_pin_list*.

Examples

The following command creates a generated clock whose source is ff/CK and is half the frequency and applies it to pin ff/Q:

```
% create_generated_clock -source ff/CK -divide_by 2 ff/Q
```

Related Topics

create_clock

SDC Commands

current_design

Shows or sets the current design.

Usage

current_design [*design*]

Arguments

- *design*

  Sets the design that you are currently working on. The default is the top of the design.

Description

Shows or sets the current design. If *design* is specified, this becomes the current working design. The default is the top level of the design. The commands, such as, get_cells, get_nets, get_pins, get_ports, create_net, create_cell, create_port, ungroup, connect_net, all operate relative to current_design.

Examples

The following command shows the current design:

```
% current_design
```

Related Topics

get_designs

SDC Commands

current_instance

Sets the current instance.

Usage

current_instance [*instance*] [-comment *string*]

Arguments

- *instance*

  Sets the instance that you are currently working on.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Return Values

The command returns the current design.

Description

Also sets the current_design to the module corresponding to the given instance. If you do not specify the *instance* argument, the command resets the current instance and the current design to the top design.

Examples

```
% current_design
TOP
% current_instance i_modA/i_modB
MODB
% current_instance ..
MODA
% current_instance
TOP
```

Related Topics

SDC Commands

current_design

group_path

Groups critical timing paths for optimization.

Usage

group_path {**-name *group_name*** | **-default**} [-critical_range *range*] [-priority *integer*]
[-slack_shift *delay*] [-weight *value*]
[{-from | -rise_from | -fall_from} *from_list*]
[{{-through | -rise_through | -fall_through} *through_list*}...]
[{-to | -rise_to | -fall_to} *to_list*] [-comment *string*]

Arguments

- **-name *group_name***

  Required if the -default option is not used. Specifies the name of the group.

- **-default**

  Required if the -name option is not used. Specifies that endpoints or paths be moved to the default group and removed from the current group.

- -critical_range *range*

  Optional. Specifies a margin of delay for the <group_name> during optimization. The <range> value must be positive or 0.0. If this command is not specified, the default range is 0.0.

- -priority *integer*

  Optional. Assigns a positive integer priority level to the group. The tool uses the priority level in cases where a path can belong to multiple groups. The tool assigns the path to the group with the highest user priority level. This option overrides the default SDC priority.

- -slack_shift *delay*

  Optional. Specifies that the timing constraint on the path group be adjusted by the specified delay value to relax or tighten a path. A positive value relaxes the path by increasing the available delay by the specified value, and a negative value tightens the path by reducing the available time. The number specified should be a floating value.

- -weight *value*

  Optional. Assigns an optimization weight to the group. Weight defines how the tool trades off optimizations between different paths. To assign a higher weight to any path relative to another, use the -weight option with a non-negative floating point number. The higher the number, the higher the priority that the tool assigns to the path during optimization. A zero weight makes the tool completely ignore this group during optimization. The default weight is 1.0.

- {-from | -rise_from | -fall_from} *from_list*

  <from_list> is a list of clocks, pins, or instances. Paths which start at any given object in the list are grouped. These paths can start at the rising edge (-rise_from), falling edge

(-fall_from), or both edges (-from). None or one of these arguments (-from, -rise_from, or -fall_from) can be used at once. If an instance is specified, this means the source can be any output of that instance.

- {{-through | -rise_through | -fall_through} *through_list*}...

  The <through_list> is a list of pins, nets, or instances. Points through any given object in the list are grouped.

  Multiple -through, -rise_through, or -fall_through arguments can be specified. In such a case, any path that goes through at least one object in each <through_list>, in the specified order, is disabled.

  These paths can go through the rising edge (-rise_through), falling edge (-fall_through), or both edges (-through).

- {-to | -rise_to | -fall_to} *to_list*

  <object_list> is a list of clocks, pins or instances. Paths which end at any given object in the list are grouped. None or one of these arguments (-to, -rise_to, or -fall_to) can be used at once.

- -comment *string*

  Optional. Specifies a string to be used for commenting for tracking purposes.

Examples

Example 1: Create a Group of Paths

This example specifies the *group1* group of paths that start at the *data1* port and end at the *out1* port. It also sets the optimization weight of the group to 2.5.

```
[oasys-RTL]$ group_path -name group1 -from [get_ports data1] \
   -to [get_ports out1] -weight 2.5
```

Example 2: Create Two Groups of Paths

This example creates the *PG1* group of paths that are launched by the *CK* clock and the *PG2* group of paths that are captured by the *CK* clock. Any path that gets launched and captured by the *CK* clock can belong to either group. Default SDC rules dictate that a “from clock” exception has higher priority than a “to clock” exception. Thus, the tool assigns any path that is launched and captured the *CK* clock to the *PG1* group.

```
[oasys-RTL]$ group_path -name PG1 -from [get_clock CK]
[oasys-RTL]$ group_path -name PG2 -to [get_clock CK]
```

Example 3: Set a Priority on a Group

This example creates two groups and sets a priority for one of them. The *PG1* group consists of paths that are launched by the *CK* clock. The *PG2* group consists of paths that are captured by the *CK* clock. Although by default paths that are launched and captured by the *CK* clock belong to the *PG1* group, the priority level of the *PG2* group (set to 1) overrides the default SDC priority. As a result, the tool assigns any path that is launched and captured by the *CK* clock to the *PG2* group.

```
[oasys-RTL]$ group_path -name PG1 -from [get_clock CK]
[oasys-RTL]$ group_path -name PG2 -to [get_clock CK] -priority 1
```

Example 4: Tighten Constraints on a Bank of Registers

This example tightens the timing constraints on the D pins of the *alu_reg* bank by 25 ps.

```
[oasys-RTL]$ group_path -name alu_regs -to [get_pins -hier alu_reg*/D] \
   -slack_shift -25.0
```

Example 5: Relax an Over-Constrained Bus

This example relaxes the over-constrained *Results* bus IO path by 100 ps.

```
[oasys-RTL]$ group_path -name overConIO -to [get_ports Results*] \
   -slack_shift 100.0
```

Related Topics

SDC Commands

remove_path_group

report_path_groups

remove_path_group

Removes timing path groups from the current design.

Usage

remove_path_group [-all *bool*] [-groups *path_group_list*] [-modes *string_list*]

Arguments

- -all *bool*

  Removes all path groups in the current design. The default value is false, where only specified path groups are removed.

- -groups *path_group_list*

  Specifies the list of path groups to remove.

- -modes *string_list*

  Specifies the timing mode list. The default value is to remove path groups for all timing modes.

Examples

Remove all path groups in the design:

```
remove_path_group -all
```

Remove the path group “reg2reg” in the design:

```
remove_path_group -groups reg2reg
```

Remove the test mode path group in2reg in the design:

```
remove_path_group -groups in2reg -modes test
```

Related Topics

SDC Commands

group_path

report_sdc_status

Prints a summary table showing the pass/fail/ignore status of SDC/timing constraints read in. Warning columns indicate that a given constraint has partially passed. The table also indicates filenames for constraints read in.

Usage

report_sdc_status [-verbose] [-comment *string*]

Arguments

- -verbose

  Optional. Dumps commands that have failed, been ignored, or have passed with warnings, with line number and file name, at the end of every file read through the read_sdc command or by sourcing a Tcl file. Note that the line number and filename is displayed only when the file is read using the read_sdc command.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

Examples

```
% report_sdc_status

Constraints Files:
-------------------------------------------------------------------------------
-<test-case-path>/scripts/constraints1.tcl
-<test-case-path>/scripts/constraints1.tcl
-------------------------------------------------------------------------------

Constraints Status Summary Table:
-------------------------------------------------------------------------------
Constraint                    Total      Failed     Ignored    Warnings   Message
-------------------------------------------------------------------------------
set_operating_conditions       1           0           0           0      Success
create_clock                   2           0           0           0      Success
set_input_delay                1           0           0           0      Success
set_output_delay               1           0           0           0      Success
set_false_path                 2           2           0           0      Failure
group_path                     1           0           0           0      Success
set_min_delay                  1           0           1           0      Ignored
-------------------------------------------------------------------------------
```

reset_path

Resets a specified exception path to a normal single cycle path.

Usage

reset_path [{-from | -rise_from | -fall_from} *from_list*]
[{{-through | -rise_through | -fall_through} *through_list*} ...]
[{-to | -rise_to | -fall_to} *to_list*] [-hold] [-setup] [-rise] [-fall] [-comment *string*]

Arguments

- {-from | -rise_from | -fall_from} *from_list*

  Specifies that paths starting at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from) of the listed clocks, pins, or instances be reset.

- {{-through | -rise_through | -fall_through} *through_list*} ...

  Specifies that paths passing through the list of pins, nets, or instances be reset.

- {-to | -rise_to | -fall_to} *to_list*

  Specifies that paths leading to the list of pins, nets, or instances be reset.

- -hold

  Pass-through option. See Min/Hold Timing Analysis Support.

- -setup

  Specifies that any exceptions with the -setup option is reset. This is the default.

- -rise

  Not supported. Use -rise_from, -rise_to, or -rise_through as needed to reset specific paths.

- -fall

  Not supported. Use -fall_from, -fall_to, or -fall_through as needed to reset specific paths.

- -comment *string*

  Specifies a string to be used for commenting the reset operation for tracking purposes.

Description

Resets exception path that has been set with the set_false_path, set_multicycle_path, or set_max_delay commands with the different source and capture options. Multiple -through, -rise_through, or -fall_through arguments can be specified. In these cases, the exception on any path that goes through at least one object in each *through_list*, in the specified order, is reset.

Examples

This example sets the path to all specified objects to a false path. Next, the original single cycle path to the object is reset.

```
# set the false_path to all objects
% set_false_path -to [get_pins u3/delay_*_reg/D ]
#  restore the original single cycle path to a specific object
% reset_path -to [get_pins u3/delay_0_reg/D ]
```

Related Topics

SDC Commands

set_case_analysis

Specifies that a port or pin is at a constant logic value for timing analysis and optimization.

Usage

set_case_analysis {**0** | **1** | **rise** | **fall**} [-comment *string*] *port_pin_list*

Arguments

- {**0** | **1** | **rise** | **fall**}

  Specifies the logic value assigned to the specified ports or pins.

- -comment *string*

  Specifies a string to be used for commenting the operation for tracking purposes.

- *port_pin_list*

  Lists ports or pins to which the value is assigned.

Description

Case analysis is a way to specify a particular design mode without modifying the netlist. The logical constant is propagated forward from the ports or pins listed and applies to the current timing analysis session.

Examples

```
% set_case_analysis 0 my_port
```

Related Topics

SDC Commands

remove_case_analysis

report_case_analysis

set_clock_groups

Specifies clock groups that are mutually exclusive or asynchronous with each other.

Usage

set_clock_groups -group *clocks_list ...*
{**-physically_exclusive** | **-logically_exclusive** | **-asynchronous**}
[-name *name*] [-allow_paths]

Arguments

- -group *clocks_list ...*

  Specifies the list of clocks. The -group option can be used multiple times.

- **-physically_exclusive**

  Required argument if -logically_exclusive or -asynchronous are not specified. This argument specifies clock groups that are physically exclusive with each other. Physically exclusive clocks cannot coexist in the design at the same time.

- **-logically_exclusive**

  Required argument if -physically_exclusive or -asynchronous are not specified. This argument specifies clock groups that are logically exclusive with each other. Logically exclusive clocks do not have any functional paths between them, but may have coupling interactions with each other.

- **-asynchronous**

  Required argument if -physically_exclusive or -logically_exclusive are not specified. This argument specifies clock groups that are asynchronous with respect to each other. Asynchronously exclusive clocks have no phase relationship at all.

- -name *name*

  Specifies the name of the clock grouping.

- -allow_paths

  This option is ignored.

Description

Specifies clock groups that are mutually exclusive or asynchronous with each other so that the paths between these clocks are not considered during timing analysis. This is similar to using set_false_paths, but often can replace many set_false_path commands. For all three types of relationships, timing paths between clocks are suppressed.

Examples

Example

This example command sets an asynchronous group:

```
% set_clock_groups -asynchronous -name g1 -group TESTCLK
```

Example

This example command sets a physically exclusive group:

```
% set_clock_groups -physically_exclusive -group {CLK1 CLK3} \
   -group {CLK2 CLK4}
```

Related Topics

SDC Commands

set_false_path

create_clock

set_clock_latency

Specifies the clock network latency.

Usage

set_clock_latency [-rise | -fall] [-min] [-max] [-source] [-early] [-late] [-clock *clocks*] ***delay*** ***object_list***

Arguments

- -rise

  Indicates that <delay> is applied only to rise clock network latency. By default he <delay> is applied to both rise and fall clock network latency.

- -fall

  Indicates that <delay> is to apply only to fall clock network latency. By default the <delay> is applied to both rise and fall clock network latency.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Indicates that <delay> is applied only to maximum clock network latency. This is the default.

- -source

  Indicates that <delay> is applied to clock source latency instead of the network latency. See description below.

- -early

  Not applicable, the argument is ignored.

- -late

  Indicates that <delay> is applied only to late clock source latency. This is the default.

- -clock *clocks*

  This option is ignored.

- ***delay***

  Clock latency value.

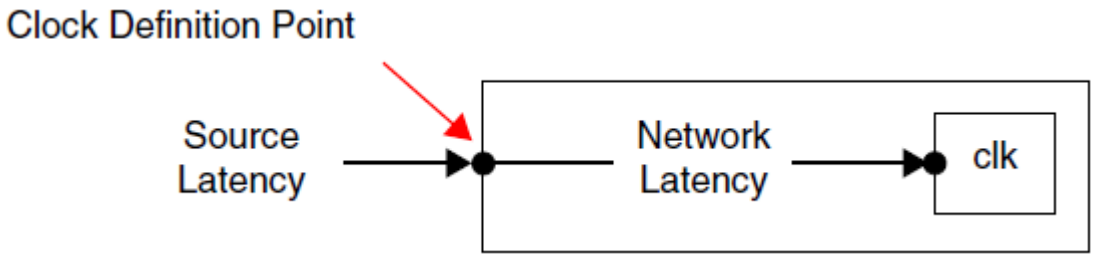
- ***object_list***

  A list of clocks, ports, or pins for which the clock latency is to be set.

Description

Specifies the clock network latency. Two types of latency can be specified:

1. **Network latency** - This is the time it takes the clock signal to propagate from the clock definition point to the internal clock pin.
2. **Source latency** - (also known as insertion delay) This is the time it takes for a clock signal to propagate from its actual ideal waveform origin point to the clock definition point in the design. It can be used to model off-chip clock latency when the clock generation circuit is not part of the current design. For generated clocks, clock source latency can be used to model the delay from master-clock to the generated clock definition point.



Examples

```
% set_clock_latency -rise -max -late 2 portA
```

Related Topics

SDC Commands

create_clock

set_clock_transition

set_clock_uncertainty

set_input_delay

set_clock_sense

Specifies the clock sense. This command is used to specify the unateness of the clock network through a specific pin.

Usage

set_clock_sense ***pin_object_list*** [-clocks *clock_list*] [-negative | -positive] [-pulse *pulse_type*] [-stop_propagation]

Arguments

- ***pin_object_list***

  Required. Specifies a list of pins for which the clock sense is to be set.

- -clocks *clock_list*

  Optional. Specifies a list of clocks to which the clock sense is applied.

- -negative

  Optional. Indicates a negative clock sense.

- -positive

  Optional. Indicates a positive clock sense.

- -pulse *pulse_type*

  Optional. Specifies the type of pulse. The pulse_type can be any of the following keywords:

  rise_triggered_high_pulse

  rise_triggered_low_pulse

  fall_triggered_high_pulse

  fall_triggered_low_pulse

- -stop_propagation

  Optional. Prevents further propagation of the clock from the specified downstream pin.

Examples

Example 1: Stopping Propagation

In the following example, two clocks (CK1 and CK2) pass from a MUX to flip-flops FF1 and FF2. Before propagation is stopped, the tool reports timing through Q of of FF1; after propagation is stopped, no timing is reported, because the clock has been stopped from reaching the flip-flop.

```
% report_timing -through [get_pin htop/part/ff1/Q]
-------------------------------------------------------------------------------------------
Startpoint: htop/part/ff1/Q
    (Clocked by CK2 R)
Endpoint: htop/part/ff2/D
    (Clocked by CK1 F)
Data required time: 446.5
    (Clock period: 500.0, minus Uncertainty: 0.0, plus Latency 0.0, minus Setup time: 53.5)
Data arrival time: 122.2
Slack: 324.3
Logic depth: 0
-------------------------------------------------------------------------------------------
                                                                  Arrival          Total
Path                     Module/Cell              Slew     Delay     Time  Edge     Load
                                                  (ps)      (ps)      (ps)           (ff)
-------------------------------------------------------------------------------------------
CK2                      {create_clock}                      0.0       0.0     r      0.0
htop/part/xr1/B->Y       XOR2X1                    9.0       0.0       0.0    rr      0.0
htop/part/mx0/B->Y       MX2X1                     9.0       0.0       0.0    rr      0.0
htop/part/ff1/CK->Q      DFFQX1                    9.0     121.3     121.3    rf      3.1
htop/part/ff2/D          DFFQX1                   47.6       0.9     122.2     f
-------------------------------------------------------------------------------------------

% set_clock_sense -clocks CK2 -stop_propagation [get_pin htop/part/mx0/Y]
% set_clock_sense -clocks CK1 -stop_propagation [get_pin htop/part/mx0/Y]

# After the previous commands, no timing is reported.
% report_timing -through [get_pin htop/part/ff1/Q]
```

Example 2: Setting the Clock Sense to Negative

This example sets the clock sense on the Y pin of a MUX to negative. Observe the highlighted differences in the edge colum that indicate changes in clock polarity.

```
% report_timing -through Q
-------------------------------------------------------------------------------------------
Startpoint: htop/gen/ff2/Q
    (Clocked by CK1 R)
Endpoint: Q
    (Clocked by: CK2 R)
Data required time: 1000.0
    (Clock period: 1000.0, minus Uncertainty: 0.0, plus Latency 0.0, minus Outdelay: 0.0)
Data arrival time: 116.9
Slack: 883.1
Logic depth: 0
-------------------------------------------------------------------------------------------
                                                                 Arrival          Total
Path                     Module/Cell              Slew     Delay     Time  Edge     Load
                                                  (ps)      (ps)      (ps)           (ff)
-------------------------------------------------------------------------------------------
CK1                      {create_clock}                      0.0       0.0     r      0.0
htop/gen/xr0/B->Y        XOR2X1                    9.0       0.0       0.0    rr      0.0
htop/gen/mx0/A->Y        MX2X1                     9.0       0.0       0.0    rr      0.0
htop/gen/ff2/CK->Q       DFFQX1                    9.0     116.6     116.6    rf      2.3
Q                                                 42.6       0.3     116.9     f
-------------------------------------------------------------------------------------------
% set_clock_sense -clocks CK1  -negative [get_pin htop/gen/mx0/Y]

## After applying set_clock_sense -negative
% report_timing -through Q
-------------------------------------------------------------------------------------------
Startpoint: htop/gen/ff2/Q
    (Clocked by CK1 F)
Endpoint: Q
    (Clocked by: CK2 R)
Data required time: 500.0
    (Clock period: 500.0, minus Uncertainty: 0.0, plus Latency 0.0, minus Outdelay: 0.0)
Data arrival time: 116.9
Slack: 383.1
Logic depth: 0
-------------------------------------------------------------------------------------------
                                                                 Arrival          Total
Path                     Module/Cell              Slew     Delay     Time  Edge     Load
                                                  (ps)      (ps)      (ps)           (ff)
-------------------------------------------------------------------------------------------
CK1                      {create_clock}                      0.0       0.0     f      0.0
htop/gen/xr0/B->Y        XOR2X1                    9.0       0.0       0.0    fr      0.0
htop/gen/mx0/A->Y        MX2X1                     9.0       0.0       0.0    rr      0.0
htop/gen/ff2/CK->Q       DFFQX1                    9.0     116.6     116.6    rf      2.3
Q                                                 42.6       0.3     116.9     f
-------------------------------------------------------------------------------------------
```

Example 3: Setting the Clock Sense to Positive

This example sets the clock sense on the Y pin of a MUX to positive. The timing report does not change, because a positive clock sense results in the same polarity.

```
% report_timing -through Q
-------------------------------------------------------------------------------------------
Startpoint: htop/gen/ff2/Q
    (Clocked by CK1 R)
Endpoint: Q
    (Clocked by: CK2 R)
Data required time: 1000.0
    (Clock period: 1000.0, minus Uncertainty: 0.0, plus Latency 0.0, minus Outdelay: 0.0)
Data arrival time: 116.9
Slack: 883.1
Logic depth: 0
-------------------------------------------------------------------------------------------
                                                                 Arrival          Total
Path                     Module/Cell              Slew     Delay     Time  Edge     Load
                                                  (ps)      (ps)      (ps)           (ff)
-------------------------------------------------------------------------------------------
CK1                      {create_clock}                      0.0       0.0     r      0.0
htop/gen/xr0/B->Y        XOR2X1                    9.0       0.0       0.0    rr      0.0
htop/gen/mx0/A->Y        MX2X1                     9.0       0.0       0.0    rr      0.0
htop/gen/ff2/CK->Q       DFFQX1                    9.0     116.6     116.6    rf      2.3
Q                                                 42.6       0.3     116.9     f
-------------------------------------------------------------------------------------------
% set_clock_sense -clocks CK1  -negative [get_pin htop/gen/mx0/Y]

## After applying set_clock_sense -negative
% report_timing -through Q
-------------------------------------------------------------------------------------------
Startpoint: htop/gen/ff2/Q
    (Clocked by CK1 F)
Endpoint: Q
    (Clocked by: CK2 R)
Data required time: 500.0
    (Clock period: 500.0, minus Uncertainty: 0.0, plus Latency 0.0, minus Outdelay: 0.0)
Data arrival time: 116.9
Slack: 383.1
Logic depth: 0
-------------------------------------------------------------------------------------------
                                                                 Arrival          Total
Path                     Module/Cell              Slew     Delay     Time  Edge     Load
                                                  (ps)      (ps)      (ps)           (ff)
-------------------------------------------------------------------------------------------
CK1                      {create_clock}                      0.0       0.0     r      0.0
htop/gen/xr0/B->Y        XOR2X1                    9.0       0.0       0.0    rr      0.0
htop/gen/mx0/A->Y        MX2X1                     9.0       0.0       0.0    rr      0.0
htop/gen/ff2/CK->Q       DFFQX1                    9.0     116.6     116.6    rf      2.3
Q                                                 42.6       0.3     116.9     f
-------------------------------------------------------------------------------------------
```

Related Topics

SDC Commands

create_clock

set_clock_transition

set_clock_uncertainty

set_input_delay

set_clock_transition

Sets clock transition times.

Usage

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

Arguments

- -rise | -fall

  Specifies whether the value is applicable for rising or falling transition. If neither is specified, both rising and falling clock transitions are set. If you specify -rise or -fall, the other transition value is left as is.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Specifies that the value is applicable for maximum delay analysis. This is the default.

- ***transition***

  Specifies a transition for clock nets directly fanning out to a sequential device. The device is clocked by the specified clock. Specify transition in the same units the technology library uses during optimization.

- ***clock_list***

  A list of virtual clocks in the design.

Description

Set transition times on all pins related to specified virtual clocks; without this, 0.0 is assumed.

Examples

```
% set_clock_transition 0.75 CLK
```

Related Topics

SDC Commands

set_clock_uncertainty

set_clock_latency

set_clock_uncertainty

Specifies the clock skew characteristics.

Usage

set_clock_uncertainty [-rise] [-fall] [-rise_from *rise_from_clock*]
[-fall_from *fall_from_clock*] [-rise_to *rise_to_clock*] [-fall_to *fall_to_clock*]
[-hold] [-setup] [***clock_list*** | **-from *from_clock_list* -to *to_clock_list***]
uncertainty

Arguments

- -rise

  Uncertainty applies to only the rising edge of the destination clock.
- -fall

  Uncertainty applies to only the falling edge of the destination clock.
- -rise_from *rise_from_clock*

  Uncertainty applies to the rising edge of the source clock.
- -fall_from *fall_from_clock*

  Uncertainty applies to the falling edge of source clock.
- -rise_to *rise_to_clock*

  Uncertainty applies to the rising edge of destination clock.
- -fall_to *fall_to_clock*

  Uncertainty applies to the falling edge of destination clock.
- -hold

  Pass-through option. See Min/Hold Timing Analysis Support.
- -setup

  Indicates that uncertainty applies only to setup checks. This is the default.
- ***clock_list***

  This must be specified OR both -from AND -to must be specified.
- **-from *from_clock_list***

  Specifies the source clock for inter-clock uncertainty. If this is specified, also specify -to.
- **-to *to_clock_list***

  Specifies the destination clock for inter-clock uncertainty. If this is specified, also specify -from.
- ***uncertainty***

  Specifies the uncertainty value.

Description

Specifies the skew characteristics of one or more clock networks. You can specify clock uncertainty as simple uncertainty or inter-clock uncertainty.

Simple uncertainty means the setup uncertainty applies to all paths within given clock domains.

Inter-clock uncertainty allows you to specify different skew between various clock domains.

For paths between clock domains which have simple but different uncertainties specified, the average value is used (that is, half of the launching clock domain and half of the receiving clock domain).

Examples

```
% set_clock_uncertainty 0.3 -from PHI1 -to PHI1
```

Related Topics

SDC Commands

set_clock_transition

set_clock_latency

set_disable_timing

Disables timing arcs in the design.

Usage

set_disable_timing [-from *from_pin*] [-to *to_pin*] [-loop] ***object_list***

Arguments

- -from *from_pin*

  Specifies the source pin of the timing arc to disable. Library cell pins are not supported.

- -to *to_pin*

  Specifies the destination pin of the timing arc to disable. Library cell pins are not supported.

- ***object_list***

  Specifies a list of pins, ports, cells, lib cells, or cell pins to disable. Library cell pins are not supported.

- -loop

  Specifies that the objects belong to a combination loop. Library cell pins are not supported.

Description

Disables timing through the specified library cells or instances in the current design. By default, all timing arcs of the specified cells are disabled. When -from or -to pins are specified, only arcs from or to those pins are disabled. Library cell pins are not supported.

Examples

Example

To disable all timing arcs of library cell NAND4:

```
% set_disable_timing [get_lib_cells NAND4]
```

Example

To disable all timing arcs coming from pin A of instance foo/bar/i_3:

```
% set_disable_timing -from A [get_cells foo/bar/i_3]
```

Example

To disable the timing arc from pin A to pin Y of instance foo/bar/i_4:

```
% set_disable_timing -from A -to Y foo/bar/i_4
```

Related Topics

SDC Commands

current_design

set_driving_cell

Specifies that a library cell or pin drives specified ports.

Usage

set_driving_cell [-lib_cell *lib_cell_name*] [-rise] [-fall] [-library *lib*] [-pin *pin_name*] [-from_pin *pin_name*] [-multiply_by *value*] [-dont_scale] [-no_design_rule] [-input_transition_rise *value*] [-input_transition_fall *value*] ***port_list*** [-min] [-max] [-clock *clocks*] [-clock_fall]

Arguments

- -lib_cell *lib_cell_name*

  Specifies the name of the library cell used to drive the ports. If the cell has more than one output pin, you must specify the -pin option.

- -rise

  Indicates that the driving cell specification is being defined related to a rising edge of the port. The -rise and -fall arguments are mutually exclusive. This argument is ignored.

- -fall

  Indicates that the driving cell specification is being defined related to a falling edge of the port. The -rise and -fall arguments are mutually exclusive. This argument is ignored.

- -library *lib*

  Specifies the library for finding *lib_cell_name*. This option is generally not required as library cell names should be unique across all loaded libraries.

- -pin *pin_name*

  Specifies the output pin on the driving cell that is to drive the ports. This is required if the driving cell has more than one output pin.

- -from_pin *pin_name*

  Specifies the from pin name.

- -multiply_by *value*

  Specifies the multiplier used to modify the resulting slew calculation. A value of 1 preserves the slew value, and a value of 0 is equivalent to having no driving cell specification. This argument is ignored.

- -dont_scale

  Prevents scaling of the driving cell output transition times based on voltage scaling that might be applied to the specified library cell. This argument is ignored.

- -no_design_rule

  Specifies that design rules in the driving cell library (such as max_cap or max_transition) are not to be applied to the port to which this constraint is applied. This argument is ignored.

- -input_transition_rise *value*

  Specifies the input transition rise value to apply to the input pin as specified with the -from_pin argument. This argument is ignored.

- -input_transition_fall *value*

  Specifies the input transition fall value to apply to the input pin as specified with the -from_pin argument. This argument is ignored.

- ***port_list***

  List of names of input or input/output ports in the current design on which the driving cell attributes are to be placed.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Sets driving cell information for analysis at the maximum operating condition only. This is the default.

- -clock *clocks*

  Specifies a list of clocks to define the driving cell on a per-clock basis. This argument is ignored.

- -clock_fall

  Configures the driving cell as relative to the falling clock edge. Must be supplied with the -clock argument. This argument is ignored.

Description

Sets attributes on the specified input or input/output ports in the current design to associate an external driving cell with the ports.

Examples

```
% set_driving_cell -lib_cell BUFX8 -library slow -pin Z a_in[3]
```

Related Topics

SDC Commands

remove_driving_cell

set_false_path

Specifies the paths to be ignored for timing analysis and optimization.

Usage

set_false_path [{-from | -rise_from | -fall_from} *from_list*]
[{{-through | -rise_through | -fall_through} *through_list*} ...]
[{-to | -rise_to | -fall_to} *to_list*] [-rise] [-fall] [-hold] [-setup] [-loop] [-reset_path]

Arguments

- {{-from | -rise_from | -fall_from} *from_list*

  from_list is a list of clocks, pins, or instances. Paths which start at any given object in the list is set as a false path. These paths can start at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from). Zero or one of -from, -rise_from, or -fall_from arguments can be used at once. If an instance is specified, this means the source can be any output of that instance.

- -rise

  Not supported. Use -rise_from, -rise_to, or -rise_through as needed.

- -fall

  Not supported. Use -fall_from, -fall_to, or -fall_through as needed.

- {-through | -rise_through | -fall_through} *through_list* ...

  through_list is a list of pins, nets, or instances. Points through any given object in the list are grouped.

  Multiple -through, -rise_through, or -fall_through arguments can be specified. In such a case, any path that goes through at least one object in each *through_list*, in the specified order, is disabled.

  These paths can go through the rising edge (-rise_through), falling edge (-fall_through), or both edges (-through).

- {-to | -rise_to | -fall_to} *to_list*

  <object_list> is a list of clocks, pins or instances. Paths which end at any given object in the list are set as a false path. Zero or one of arguments -to, -rise_to, or -fall_to can be used at once.

- -hold

  Pass-through option. See Min/Hold Timing Analysis Support.

- -setup

  Marks setup checks as false and disables setup checking for specified paths.

- -loop

  Specifies that the path is a loop.

- -reset_path

  Resets any exception existing on the target path and applies the new false path exception. Using this option is equivalent to applying the reset_path command on the specified path and then applying the new false path exception to the same path. Previously existing path exceptions may have been set with the following options:

  - -from
  - -rise_from
  - -fall_from
  - -to
  - -rise_to
  - -fall_to
  - -through
  - -rise_through
  - -fall_through

Description

Removes false paths from timing analysis.

At least one -from, -rise_from, -fall_from, -through, -rise_through, -fall_through, -to, -rise_to, or -fall_to argument must be specified.

Examples

These examples set false path exceptions.

```
% set_false_path -through [get_nets path/to/some/net*]
% set_false_path -from reset
% set_false_path -from clock1 -to clock2
```

This example sets a false path exception and resets the false path before applying a multi-cycle path exception.

```
# original exception
set_false_path -from  [get_pins source_reg*/CLK ]
# remove false path from one of the register paths
set_multicycle_path -reset_path -from source_reg[0]/CLK
```

Related Topics

SDC Commands

set_multicycle_path

set_max_delay

set_input_delay

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

Usage

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

Arguments

- -add_delay

  Adds the specified delay to a pin that already has a delay specified on it. If -add_delay is not specified, the previous delay is overwritten. See example below.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Specifies that *delay_value* refers to the longest path. This is the default.

- -clock *clock_name*

  Specifies the clock to which the specified delay is related.

- -clock_fall

  Specifies to use the falling edge of the reference clock. The default is the positive edge.

- -rise

  Specifies that *delay_value* refers to a rising transition on specified ports.

- -fall

  Specifies that *delay_value* refers to a falling transition on specified ports.

- -network_latency_included

  Specifies that the clock source latency is not added to the input delay value. See network latency definition.

- -source_latency_included

  Specifies that the clock source latency is not added to the input delay value. See source latency definition.

- -level_sensitive

  Must be specified with -clock option. This option is ignored.

- ***delay_value***

  Specifies the necessary arrival time for an input port.

- ***port_pin_list***

  The list of input pins or ports.

Description

Specifies the data arrival times at the specified input ports relative to the specified clock. By default if both -rise and -fall are not specified, the delays are for both.

Examples

Example

The following example specifies the maximum rise delay of input pin In1 to be 3, using the positive edge of the reference clock A:

```
% set_input_delay -clock -rise A 3 In1
```

Example

The following example specifies a max arrival delay on pin In1 of 5 relative to the positive edge of clock A, and 6 relative to the positive edge of B. This is for both the rise and fall delays of pin In1.

```
% set_input_delay -clock A 5 In1
% set_input_delay -add_delay -clock B 6 In1
```

Related Topics

SDC Commands

remove_input_delay

set_output_delay

set_load

all_inputs

set_input_transition

Sets a fixed transition time on input or bidirectional ports.

Usage

set_input_transition [-rise] [-fall] [-min] [-max] ***transition port_list*** [-clock *clock_list*] [-clock_fall]

Arguments

- -rise

  Sets the rise transition time.

- -fall

  Sets the fall transition time only. This is the default.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Indicates that the transition value is to be applied for maximum delay analysis. This is the default.

- ***transition***

  Specifies the port transition value consistent with units in technology library used in the optimization.

- ***port_list***

  Specifies the list of input or input/output ports.

- -clock *clock_list*

  Specifies a list of clocks on which to set the transition. This argument is ignored.

- -clock_fall

  Specifies to use the falling edge of the reference clock. The default is the positive edge. This argument is ignored.

Examples

```
% set_input_transition -fall 7.0 {A, B, C}
```

Related Topics

SDC Commands

remove_input_transition

all_inputs

set_load

set_load

Sets the capacitance load to a specified value on specified ports and nets.

Usage

set_load [-min] [-max] [-subtract_pin_load] [-pin_load] [-wire_load] ***value objects***

Arguments

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Specifies that the load value is to be used for maximum delay analysis.

- -subtract_pin_load

  Subtracts the pin load for the specified objects. This argument is ignored.

- -pin_load

  Indicates that the specified value on the port is treated as a pin load. This is the default. This argument is ignored.

- -wire_load

  Indicates that the specified value on the port is treated as a wire load. This argument is ignored.

- ***value***

  Specifies a capacitance value to apply. Must be consistent with the units in the technology library.

- ***objects***

  Specifies a list of port and net names to which the capacitance load is applied.

Examples

```
% set_load -pin_load 10.0 a_in[0]
```

Related Topics

SDC Commands

remove_load

all_outputs

set_logic_one

Sets a logic value of 1 to a list of ports.

Usage

set_logic_one ***port_list***

Arguments

- ***port_list***

  Specifies a list of ports that get set to a logic value of 1.

Examples

```
% set_logic_one reset
```

Related Topics

SDC Commands

set_logic_zero

set_logic_zero

Sets a logic value of 0 to a list of ports.

Usage

set_logic_zero ***port_list***

Arguments

- ***port_list***

  Specifies a list of ports that get set to a logic value of 0.

Examples

```
%set_logic_zero reset
```

Related Topics

SDC Commands

set_logic_one

set_max_delay

Sets maximum delay for specified timing paths.

Usage

set_max_delay [{-from | -rise_from | -fall_from} *from_list*]
[{{-through | -rise_through | -fall_through} *through_list*} ...]
[{-to | -rise_to | -fall_to} *to_list*] [-rise | -fall] [-reset_path] ***delay_value***

Arguments

- {-from | -rise_from | -fall_from} *from_list*

  <from_list> is a list of clocks, pins, or instances. Apply max delay to paths which start at any given object in the list. These paths can start at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from). Zero or one of -from, -rise_from, or -fall_from arguments can be used at once. If an instance is specified, this means the source can be any output of that instance.

- {-through | -rise_through | -fall_through} *through_list* ...

  <through_list> is a list of pins, nets, or instances. Points through any given object in the list are grouped.

  Multiple -through, -rise_through, or -fall_through arguments can be specified. In such a case, any path that goes through at least one object in each <through_list>, in the specified order, is disabled.

  These paths can go through the rising edge (-rise_through), falling edge (-fall_through), or both edges (-through).

- {-to | -rise_to | -fall_to} *to_list*

  <object_list> is a list of clocks, pins or instances. Apply max delay to paths which end at any given object in the list. Zero or one of arguments -to, -rise_to, or -fall_to can be used at once.

- -rise | -fall

  Not supported. Use -rise_from, -rise_to, -rise_through, -fall_from, -fall_to, or -fall_through as needed.

- -reset_path

  Resets any exception existing on the target path and applies the new maximum delay exception. Using this option is equivalent to applying the reset_path command on the specified path and then applying the new maximum delay exception to the same path. Previously existing path exceptions may have been set with the following options:

  - -from
  - -rise_from
  - -fall_from

- -to
- -rise_to
- -fall_to
- -through
- -rise_through
- -fall_through

- ***delay_value***

  Specifies the maximum delay value.

Description

This command specifies that the maximum path delay for given paths must be less than <delay_value>.

At least one -from, -rise_from, -fall_from, -through, -rise_through, -fall_through, -to, -rise_to, or -fall_to argument must be specified.

Examples

These examples set maximum delay path exceptions:

```
% set_max_delay -from [get_ports data1] -to [get_ports out1] 8
% set_max_delay -from [get_ports data1] -to [get_ports {out1 out2}] 9
```

This example resets a previously set multi-cycle path exception before applying the maximum delay path exception:

```
# original exception
set_multicycle_path 2 -from  source_reg/CLK
# remove multicycle_path and apply max delay
set_max_delay 1.0 -reset_path -from source_reg/CLK
```

Related Topics

SDC Commands

set_input_delay

set_output_delay

set_false_path

set_multicycle_path

Allows more than one clock cycle for designated paths.

Usage

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

Arguments

- {-from | -rise_from | -fall_from} *from_list*

  <from_list> is a list of clocks, pins, or instances. Applies multicycle to paths which start at any specified object in the list. These paths can start at the rising edge (-rise_from), falling edge (-fall_from), or both edges (-from). Zero or one of -from, -rise_from, or -fall_from arguments can be used at once. If an instance is specified, this means the source can be any output of that instance.

- {{-through | -rise_through | -fall_through} *through_list*} ...

  <through_list> is a list of pins, nets, or instances. Points through any given object in the list are grouped.

  Multiple -through, -rise_through, or -fall_through arguments can be specified. In such a case, any path that goes through at least one object in each <through_list>, in the specified order, is disabled.

  These paths can go through the rising edge (-rise_through), falling edge (-fall_through), or both edges (-through).

- {-to | -rise_to | -fall_to} *to_list*

  <object_list> is a list of clocks, pins or instances. Apply multicycle to paths which end at any given object in the list. Zero or one of arguments -to, -rise_to, or -fall_to can be used at once.

- -hold

  Pass-through option. See Min/Hold Timing Analysis Support.

- -setup

  Specifies that <path_multiplier> is used for setup calculations. This is the default.

- -rise | -fall

  Not supported. Use -rise_from, -rise_to, -rise_through, -fall_from, -fall_to, or -fall_through as needed.

- -start | -end

  Specifies whether the multicycle information is relative to the period of the start clock or the end clock. The default is -end.

- -reset_path

  Resets any exception existing on the target path and applies the new multi-cycle path exception. Using this option is equivalent to applying the reset_path command on the specified path and then applying the new multi-cycle path delay exception to the same path. Previously existing path exceptions may have been set with the following options:

  - -from
  - -rise_from
  - -fall_from
  - -to
  - -rise_to
  - -fall_to
  - -through
  - -rise_through
  - -fall_through

- ***path_multiplier***

  Specifies which subsequent edge needs to be used (a value 1 does not change the way the path is timed).

Description

Allows for more than one cycle for designated paths. Normally a signal launched at a start-point of a path is assumed to be captured by the next clock edge at the endpoint. Using this command you can tell the timer that a signal on a particular path needs to be captured by a later clock edge.

At least one -from, -rise_from, -fall_from, -through, -rise_through, -fall_through, -to, -rise_to, or -fall_to argument must be specified.

Examples

This example sets a multi-cycle path exception:

```
% set_multicycle_path 2 -setup -from [get_pins D_reg/Q] \
-to [get_pins B_reg/D]
```

This example resets a previously set false path exception before applying a multi-cycle path exception:

```
# original exception
set_false_path -to u3/delay_reg/D
# remove false_path and apply multi-cycle path
set_multicycle_path 2 -reset_path -to u3/delay_reg/D
```

Related Topics

SDC Commands

set_max_delay

set_false_path

set_operating_conditions

Sets the operating conditions.

Usage

set_operating_conditions [-analysis_type *value*] [-library *lib_name*]
[-max *max_condition*] [-min *min_condition*] [-max_library *library_list*]
[-object_list *objects*] [-min_library *library_list*] [*condition*]

Arguments

- -analysis_type *value*

  Specifies the analysis type. This argument is ignored.

- -library *lib_name*

  Specifies the name of the library.

- -max *max_condition*

  Specifies the operating condition to use for maximum delay analysis.

- -min *min_condition*

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max_library *library_list*

  Specifies a list of libraries for the maximum operating condition. This argument is ignored.

- -object_list *objects*

  Specifies a list of objects for the operating condition. Lists of cells and lists of ports are not supported.

- -min_library *library_list*

  Pass-through option. See Min/Hold Timing Analysis Support.

- *condition*

  Same as -max <max_condition>, since -min is a pass-through option.

Examples

The following example sets the max condition to WC:

```
% set_operating_condition -max WC
```

Related Topics

SDC Commands

set_output_delay

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

Usage

set_output_delay [-add_delay] [-min] [-max] [-network_latency_included] [-source_latency_included] [-clock *clock_name*] [-clock_fall] [-rise] [-fall] [-level_sensitive] ***delay_value port_pin_list***

Arguments

- -add_delay

  Adds the specified delay to a pin that already has a delay specified on it. By default the previous delay is overwritten. See example below.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -max

  Specifies that *delay_value* refers to the longest path. This is the default.

- -network_latency_included

  Specifies that the clock source latency is not added to the output delay value. See network latency definition.

- -source_latency_included

  Specifies that the clock source latency is not added to the output delay value. See source latency definition.

- -clock *clock_name*

  Specifies the clock to which the specified delay is related.

- -clock_fall

  Specifies to use the falling edge of the reference clock. The default is the positive edge.

- -rise

  Specifies that *delay_value* refers to a rising transition on specified ports.

- -fall

  Specifies that *delay_value* refers to a falling transition on specified ports.

- -level_sensitive

  Must be specified with the -clock option. This option is ignored.

- ***delay_value***

  Specifies the data output delay.

- ***port_pin_list***

  Specifies the list of pins or ports associated with the delay.

Description

Specifies the data output delay at the specified output ports relative to the specified clock. By default, if both -rise and -fall are not specified, the delays are for both.

Examples

Example

The following example specifies the output rise delay of pin Out1 to be 3, using the positive edge of the reference clock A:

```
% set_output_delay -clock -rise A 3 Out1
```

Example

The following example specifies a output delay of 5 on pin Out1, relative to the positive edge of clock A, and delay of 6 relative to the positive edge of B. This is for both the rise and fall delays of pin Out1.

```
% set_output_delay -clock A 5 Out1
% set_output_delay -add_delay -clock B 6 Out1
```

Related Topics

SDC Commands

remove_output_delay

set_input_delay

all_outputs

set_load

group_path

set_sense

Specifies the signal sense (with respect to the clock source) propagating forward through a non-unate network from the specified pins.

Usage

set_sense [-type *type*] [-positive] [-negative] [-stop_propagation] [-pulse *pulseType*] [-clocks *clock_list*] [*pin_list*]

Arguments

- -type *type*

  Optional. Specifies the type of network (clock or data) to which the sense is being applied. If you specify "-type data", the -stop_propagation option is mandatory and is the only option required to be supplied in addition to -type.

- -positive

  Optional. Specifies that the unateness being applied to all pins in the pin_list variable is of positive sense with respect to the clock source. This option cannot be used with the -stop_propagation, -negative, or -pulse options.

- -negative

  Optional. Specifies that the unateness being applied to all pins in the pin_list variable is of negative sense with respect to the clock source. This option cannot be used with the -stop_propagation, -positive, or -pulse options.

- -stop_propagation

  Optional. Stops propagation of the clocks specified in the clock_list variable through the pins in the pin_list variable. You can use this option to stop propagation of a clock used as a clock, as a data, or both. To stop propagation of both, specify the set_sense command twice, once using "-type clock" and once using "-type data". This option cannot be used with the -positive, -negative, or -pulse options.

- -pulse *pulseType*

  Optional. Specifies the type of pulse clock that is being generated from the specified pins in the pin_list variable with respect to the clock source. The pulse type can be any of the following:

  rise_triggered_high_pulse

  rise_triggered_low_pulse

  fall_triggered_high_pulse

  fall_triggered_low_pulse

  This option cannot be used with the -stop_propagation, -positive, or -negative options.

- -clocks *clock_list*

  Optional. Specifies the clocks to apply the given unateness to through all pin objects in the pin_list variable. The default (if this is not specified) is to consider all the clocks passing through the given pins.

- *pin_list*

  Optional. Specifies a list of pins.

Description

Use this command to control unateness at a pin (by restricting it to positive or negative) with respect to the clock source. This user-defined sense is propagated forward from the specified pins for the non-unate part of the network.

If the "-type clock" option is specified, the unateness is applied only to the specified clocks. Otherwise, all clocks passing through the given pin are considered.

In addition, you can use "-stop_propagation" to completely stop the propagation of clocks in clock or data networks.

The -stop_propgatation, -positive, -negative, and -pulse options are all mutually exclusive.

Examples

Example 1: Stopping the clock through a specified pin

This example stops the propagation of clocks from the specified register pin CK.

```
% set_sense -stop_propagation [get_pins reg3/CK]
```

Example 2: Stopping the propagation of data

To stop the propagation of data through a gate N2/Y with respect to the clock "clk1":

```
% set_sense -type data -stop_propagation -clocks [get_clocks clk1]\
   [get_pins N2/Y]
```

Example 3: Setting the clock sense to positive for a specific clock

This example propagates the positive unate paths through the *xor/z* pin with respect to the *CLK1* clock:

```
% set_sense -positive -clocks [get_clocks CLK1] [get_pins xor/z]
```

Example 4: Setting pulse type on the output of a pulse generator circuit

This example specifies a pulse type of "rise_triggered_hish_pulse" for the clock passing through the Y pin of the AND gate "and1".

```
set_sense -pulse rise_triggered_high_pulse [get_pins and1/Y]
```

Related Topics

SDC Commands

set_clock_sense

report_timing

set_timing_derate

Sets the derating factors for the timing paths.

Usage

set_timing_derate ***derate_value*** [-cell_check] [-cell_delay] [-clock] [-data] [-early] [-late] [-max] [-min] [-net_delay]

Arguments

- ***derate_value***

  Required. Specifies a number with which the tool multiplies path delays for maximum/minimum delay (setup/hold) checks.

- -cell_check

  Optional. Specifies that the tool applies derating to cell timing checks, such as setup and hold. With this option, all output delays specified using the set_output_delay command are also derated.

- -cell_delay

  Optional. Specifies that the tool applies derating factors only to cell delays.

- -clock

  Pass-through option. This option has no effect on the resulting netlist. However, the tool parses it, retains it in the database, and passes it through to the output SDC with the write_sdc command.

- -data

  Optional. Applies the *derate_value* to elements on the data paths only.

- -early

  Pass-through option. See Min/Hold Timing Analysis Support.

- -late

  Optional. Specifies that the tool applies derating to setup paths. This is the default behavior.

- -max

  Optional. Specifies that the tool uses the *derate_value* for the worst case operating condition.

- -min

  Pass-through option. See Min/Hold Timing Analysis Support.

- -net_delay

  Optional. Applies the *derate_value* only to net delays.

Description

This command scales the computed cell delays, net delays, and timing checks by the specified <derate_value>. Use derating to add timing margin for the design.

Examples

Derate the Design by +10%.

The following example increases the delay shown in timing reports for the setup paths by 10%.

```
set_timing_derate 1.1 -late -cell_delay
```

Related Topics

SDC Commands

report_timing

set_units

Specifies the unit values for capacitance, resistance, time, voltage, current, and power, for the SDC commands.

Usage

set_units [-capacitance *capacitance_units*] [-resistance *resistance_units*]
[-time *time_units*] [-voltage *voltage_units*] [-current *current_units*]
[-power *power_units*]

Arguments

- -capacitance *capacitance_units*

  Specifies the units for capacitance.

- -resistance *resistance_units*

  Specifies the units for resistance.

- -time *time_units*

  Specifies the units for time.

- -voltage *voltage_units*

  Specifies the units for voltage.

- -current *current_units*

  Specifies the units for current.

- -power *power_units*

  Specifies the units for power.

Examples

```
% set_units -time ps
```

Related Topics

SDC Commands

report_units

set_wire_load_min_block_size

Specifies the minimum block area that is used for wire load selection on the current design.

Usage

set_wire_load_min_block_size **<size>**

Arguments

- <size>

  Specifies a positive value in the units of the technology library, that specifies the minimum block area of the design and all its sub-designs, to use when selecting the appropriate wire load.

Description

Specifies the minimum block area that is used for wire load selection. If the area of a block is greater than this specified value, then OasysRTL uses the wire load model based on the area of the block. If the area of a block is less than this specified value, OasysRTL uses the value specified by this command as the area value when selecting a wire load model.

This command can only be used when wire load mode is set to enclosed and using wire load models has been enabled by config_timing.

Examples

The following example sets the minimum block size to 100.

```
config_timing -use_lib_wire_load_model true
set_wire_load_mode enclosed
set_wire_load_min_block_size 100
```

Related Topics

SDC Commands

set_wire_load_mode

set_wire_load_model

set_wire_load_selection_group

set_wire_load_mode

Specifies how Wire Load Models are applied to calculate delay values on nets in hierarchical designs.

Usage

set_wire_load_mode **<mode_name>** [-comment <string>]

Arguments

- <mode_name>

  The mode_name can be one of the following three values.

  - top : Use the Wire Load Model set on the top level design to calculate the delay values on all levels of hierarchy. Any wire load mode setting on a lower level block of hierarchy will be ignored.
  - enclosed : Use the Wire Load Model from the smallest hierarchical block that completely encloses the net. If the block enclosing the net has no Wire Load Model set, then OasysRTL transverses up the hierarchy until a Wire Load Model is found.
  - segmented : Use the Wire Load Model from the module that contains each segment of the net. Nets crossing hierarchy are divided into segments. OasysRTL estimates each net segment based on the Wire Load Model of the hierarchical block enclosing the net net segment. If the block enclosing the net has no Wire Load Model set, then OasysRTL transverses up the hierarchy until a Wire Load Model is found.

- [-comment <string>]

  Specify a comment for reporting

Description

Specifies how Wire Load Models are applied to calculate delay values on nets in hierarchical designs. The Liberty file (.lib) for the technology should contain one or more Wire Load Models (WLM). OasysRTL allows you to specify different WLMs to different levels of hierarchy. In most cases, the WLM is selected based on the approximate size (area) of the hierarchical block and the number of sigmas from the mean when the vendor evaluated the distribution of values.

If you do not specify a wire load mode, then OasysRTL will attempt to used the default wire load mode from the (Liberty) library.

By default, OasysRTL will attempt to calculate the net delay and capacitance loading values for each net using the resistance and capacitance parasitic information in the LEF file and the initial layout of the design. In order to use the utilize the wire load model, you must enable it using the config_timing command prior to setting any wire load-related command.

Examples

The following example shows how to specify the wire load mode:

```
% config_timing -use_lib_wire_load_model true
% set_wire_load_mode enclosed
```

Related Topics

SDC Commands

set_wire_load_min_block_size

set_wire_load_model

set_wire_load_selection_group

set_wire_load_model

Specifies the name of the wire load model to apply to the object.

Usage

set_wire_load_model**-name <model_name>** [-library <library_name>] [-min] [-max] [object_list]

Arguments

- -name <model_name>

  Specifies wire load table name as defined in the Liberty library.

- -library <library_name>

  Specifies one or more libraries to search for the specified Wire Load Model. OasysRTL uses the first library in the list that contains the specified model_name attribute.

- -min

  Specifies that the wire load model that should be used for minimum delay analysis.

- -max

  Specifies that wire load model name should be used for maximum delay analysis.

- object_list

  Applies the specified wire load model to the objects in the list. If no object list is specified, the wire load model is assigned to the current_design.

Description

Specifies the wire load model to use for calculating wire delays and capacitance loading on the specified object(s). If no object is specified, the wire load model is applied to the current design. If the specified object is a design, the wire load model is applied to all nets in and below the specified design.

Examples

The following example sets the “2000” wire load model defined in the liberty library to the top-level design (and all hierarchical blocks below it):

```
config_timing -use_lib_wire_load_model true
set_wire_load_model -name 2000
```

Related Topics

SDC Commands

set_wire_load_min_block_size

set_wire_load_mode

set_wire_load_selection_group

set_wire_load_selection_group

Specify a selection group defined in the library to use in wire load estimation for delays and loading on nets.

Usage

set_wire_load_selection_group **<group_name>** [-library <library_list>] [-min] [-max] [<object_list>]

Arguments

- <group_name>

  Specifies the name of the selection group to use for wire load estimation.

- -library <library_list>

  Specifies one or more libraries to search for the wire_load_selection_group attribute. OasysRTL uses the first library in the list that contains the specified group_name attribute.

- -min

  Specifies that group name should be used for minimum delay analysis.

- -max

  Specifies that group name should be used for maximum delay analysis.

- <object_list>

  Applies the specified group name to the cells in this list.

Description

Applies the wire_group_selection_group attribute to the cell in the object_list. Wire-load selection-group is used only when the auto_wire_load_selection attribute is set to true.Wire load model can also be determined on criteria based on selection groups predefined in the library file.The following example show a wire_load_selection attribute from a library. This attribute segregates wire load models (eg. wm_1, wm_2, wm_3) based on chip area of a block.

```
wire_load_selection(sel_group) {    wire_load_from_area(0,16000,wm_1);
wire_load_from_area(16000,32000,wm_2);
wire_load_from_area(32000,96000,wm_3);
wire_load_from_area(96000,160000,wm_4);
wire_load_from_area(160000,320000,wm_5);}
```

It used wire load model wm_1 for a block area between 0 to 16000, wire load model wm_2 for a block area between 16000 to 32000, and so on. There can be more than one such selection groups defined.

The following list denotes the selection preference of wire load model based on the presence of attributes set in the library:

1. If the library contains a wire_load_selection group attribute, OasysRTL uses the cell area of the current design to select the wire load model
2. If the library doesn't contain a wire_load_selection group attribute, tool uses the default_wire_load attribute from the library.
3. If the library contains neither a wire_load_selection nor a default_wire_load attribute, then OasysRTLdoesn't select any wire load model and no wire load model is used.

Examples

The following example specifies the selection group used in the command description.

```
set_wire_load_selection_group sel_group
```

Related Topics

SDC Commands

set_wire_load_min_block_size

set_wire_load_mode

set_wire_load_model

Chapter 16
General Commands

General commands are used in the Oasys-RTL tool throughout different design areas to control, set, and retrieve design information.

Table 16-1. General Commands

Command	Description
ask_passphrase	Asks for passphrase to read encrypted files.
config_shell	Configures the Oasys-RTL shell properties for the session.
config_tolerance	Controls the severity of RTL parser and elaboration errors allowed for the read_verilog, read_vhdl, and synthesize commands.
config_multi_process	Configures the number of CPUs to work in parallel during area and timing optimization, sizing, RTL partitioning, and detailed placement.
create_menu	Creates a new custom GUI menu for a specified command.
date	Outputs the current date and time.
encrypt	Encrypts a file such that only the Oasys-RTL tool can read it back.
exit	Exits the Oasys-RTL tool.
get_cap_unit	Returns the SDC unit for capacitance.
get_current_unit	Returns the SDC unit for current.
get_leakage_power_unit	Returns the SDC unit for power.
get_liberty_files	Returns the complete filenames of the loaded Liberty libraries.
get_licenses	Returns a list of Oasys-RTL features that are currently checked out.
get_log_file	Returns the full pathname of the log file that is being output for the current session.
get_parameter	Returns the value of a parameter.
get_product	Returns the name of the product currently in use.
get_res_unit	Returns the SDC unit for resistance (R).

Table 16-1. General Commands (cont.)

Command	Description
get_time_unit	Returns the SDC unit for time.
get_voltage_unit	Returns the SDC unit for voltage.
instance_area	Displays the cumulative area of all instances in the design.
instance_count	Returns the number of instances in the design.
is_oasys_shell	Determines if you are in an Oasys-RTL shell.
man	Displays information on usage of the specified Oasys-RTL parameter or command.
message	Controls or counts the output of messages.
mgcdocs	Invokes the Oasys-RTL InfoHub giving access to the tool documentation in HTML or PDF format.
print_cpu	Reports the current memory usage and CPU and wall time.
quit	Quits the Oasys-RTL tool.
redirect	Redirects the output of commands to a file or a variable.
remove_attribute	Removes specified attribute from a design object.
reset_parameter	Resets specified parameter to default value.
save_gui_views	Saves the physical GUI views from the shell for a placed design.
set_attribute	Sets a value for an attribute.
set_design_effort	Sets the effort level of optimizing for leakage or congestion.
set_parameter	Changes the value of a parameter.
set_passphrase	Provides password phrase to read encrypted files.
set_register_merging	Specifies the register merging attribute on cells or designs to control the register merging optimization.
set_selection	Selects the specified instance in the GUI.
set_user_attribute	Sets a value for a user defined attribute.
source	Reads a file and executes the Tcl commands from that file.
start_gui	Starts the GUI when in the command mode.
stop_gui	Stops the GUI and goes to the command mode.

ask_passphrase

Asks for passphrase to read encrypted files.

Usage

ask_passphrase

Arguments

None.

Description

The ask_passphrase command asks you to provide the passphrase to use for decrypting a PGP encrypted input file.

To protect your RTL files or library files, encrypt these files using PGP (Pretty Good Privacy, a utility for the encrypting and decrypting data). This prevents a user from reading these files, unless you provide a passphrase.

PGP encrypted files do not have to be decrypted before you can use them in the Oasys-RTL tool. The tool decrypts these files on the fly, without creating a decrypted copy on the hard disk.

Note, the ask_passphrase command opens a dialog window for entering the passphrase. Typed text is echoed as dots. The command set_passphrase requires a string, which is visible when entered.

Examples

A PGP encrypted file is created as follows:

```
> gpg -c myTopSecretDesign.v
Enter passphrase: <Type pass phrase>
Repeat passphrase: <Re-type pass phrase>
> rm myTopSecretDesign.v
```

This creates the file myTopSecretDesign.v.gpg. A user that does not have the correct passphrase cannot read this file.

To use this file in the Oasys-RTL environment, do the following:

```
% ask_passphrase
```

Enter the passphrase. Then read the encrypted file as follows:

```
% read_verilog myTopSecretDesign.v.gpg
```

Related Topics

General Commands

set_parameter

set_passphrase

config_shell

Configures the Oasys-RTL shell properties for the session.

Usage

config_shell [-echo_commands {true | false}] [-log_commands {true | false}] [-report] [-source_verbose {true | false}] [-auto_exec {true | false}]

Arguments

- -echo_commands {true | false}

  Optional. Specifies whether to echo commands to the standard output. The default is true.

- -log_commands {true | false}

  Optional. Specifies whether to write commands to the log files. The default is true.

- -report

  Optional. Reports the status of option settings.

- -source_verbose {true | false}

  Optional. Specifies that the commands captured in the *.cmd* file should be in verbose format. Scripts are captured with the contents of the sourced files. The contents of nested script files appear as indented commands. The default is false.

- -auto_exec {true | false}

  Optional. Controls the execution of Unix/Linux commands within the Oasys-RTL shell. The default is true, which enables execution of the commands. Set to false, the option disables execution of the commands.

Examples

Example 1: Write All Commands to the Log Files

The following example configures the tool to write all commands to the log files.

```
[oasys-RTL]$ config_shell -log_commands true
```

Example 2: Display the Current Settings of the Shell

The following example is the command to display the current settings of the shell.

```
[oasys-RTL]$ config_shell -report
echo_commands : false
log_commands  : true
source_verbose   : false
auto_exec     : true
```

Example 3: Set the Oasys-RTL Tcl Shell to Auto-Execute Unix commands

The following example configures the Oasys-RTL shell to execute the Unix which command by setting the -auto_exec option to true. When the option is subsequently set to false, the tool no longer recognizes the which command.

```
[oasys-RTL]$ config_shell -auto_exec true
[oasys-RTL]$ which which
   /usr/bin/which
[oasys-RTL]$ config_shell -auto_exec false
[oasys-RTL]$ which which
error:   invalid command name "which"
```

Related Topics

General Commands

config_tolerance

Controls the severity of RTL parser and elaboration errors allowed for the read_verilog, read_vhdl, and synthesize commands.

Caution

Setting config_tolerance to ignore errors can generate a bad design. Do not override errors when synthesizing production designs.

Usage

config_tolerance [-blackbox [true | false]]
[-connection_mismatch [true | false]]
[-continue_on_error [true | false]]
[-missing_physical_library [true | false]]
[-verilog_lrm_violation [true | false]]

Arguments

- -blackbox [true | false]

  If this option is true, the synthesize command only issues a warning when it encounters an unresolved reference (blackbox). Setting this option is equivalent to setting the value of the Oasys-RTL parameter error_on_blackbox to false.

  If this option is false, the synthesize command reports errors when it encounters an unresolved reference in the design. If the -continue_on_error option is true, the synthesize command continues executing even with errors.

  The default is true.

- -connection_mismatch [true | false]

  If this option is true, the RTL parser and synthesize command treat connection mismatch errors as warnings. The tool continues to synthesize poorly connected designs because connection mismatch errors do not affect the contents of the modules.

  If this option is false, the parsing and synthesize commands report errors when they encounter connection mismatch errors. If the -continue_on_error option is true, the RTL parser and synthesize command continue executing even with errors.

  The default is false.

- -continue_on_error [true | false]

  If this option is true and the RTL parser encounters errors, the tool continues to read any remaining RTL files and synthesize the design. The synthesize command also continues if it encounters errors. By allowing the tool to continue, you can collect all of the RTL parser and synthesis errors in a single synthesis attempt.

If this option is false, the RTL parser reports errors for the first file that has errors. It does not read any remaining files in the list. The synthesize command stops executing after the first elaboration error it encounters.

The default is false.

- -missing_physical_library [true | false]

  If this option is true, the synthesize command does not check for Liberty library cells that correspond to the physical library cells prior to synthesis.

  If this option is false, the synthesis engine checks for Liberty library cells that correspond to the physical library cells prior to synthesis.

  The default is false.

- -verilog_lrm_violation [true | false]

  If this option is true, the tool accepts certain Verilog Language Reference Manual (LRM) extensions without generating RTL parser errors. It issues a warning to indicate the non-standard Verilog. The following are a subset of extensions accepted by the Oasys-RTL RTL parser:

  - Angle brackets (< >) for instance arrays

    ```
    sub u1 <$typeof(q)> (.d(<d>), .clk(clk), .q(<q>));
    ```

  - Unpacked array assignment with replication

    ```
    wire [7:0] d; wire [7:0] q[3:0];
    q = {4{d}};
    ```

  - Trailing whitespace after macro expansion

    ```
    'define f(x) x+1   // comment
    ```

  - Assignment patterns without the initial single quote (')

    ```
    wire [7:0] q [3:0]={1, 2, 3, 4}; //LRM requires '{1, 2, 3, 4}
    ```

  - Mismatched quotes

    ```
     y='"z'" //mismatch between the first '" and the second '"
    ```

  - Illegal wire assignment to ANSI-declared output ports

    ```
    module test(input d, output q);
        wire q = d; // considered an illegal redeclaration
    ```

  - 'include directive followed by semicolon

    ```
    'include "defs.h";
    ```

  - Empty event list in a property statement

    ```
    property @( ) ... endproperty
    ```

If this option is false, the RTL parser generates errors when it encounters Verilog LRM extensions. If the -continue_on_error option is true, the parsing and synthesize commands continue executing even with errors.

The default is true.

Description

This command reduces the severity of errors from predefined checks performed by the RTL parser and the elaboration engine. By lowering the severity of errors to warnings, the read_verilog, read_vhdl, and synthesize commands can continue executing when they encounter errors. Use this command to capture the errors in all of the RTL files at one time.

The initial settings for each option are the same as the default values. The option value for each argument is optional. When you change the initial setting of an option, it stays in effect until you explicitly change it again.

All arguments are optional.

Related Topics

error_on_blackbox

read_verilog

read_vhdl

synthesize

config_multi_process

Configures the number of CPUs to work in parallel during area and timing optimization, sizing, RTL partitioning, and detailed placement.

Usage

config_multi_process [-local_host <integer>]

Arguments

- -local_host <integer>

  Specifies the number of CPUs on the local host machine to be used by the current Oasys-RTL session.

Description

The config_multi_process command configures the Oasys-RTL session to perform multi-processing during the different stages. The initial support for multiprocessing is limited to using multiple CPUs on the current host machine. If the job is being launched on a specific host or grid, you are responsible for acquiring the necessary computation resources such as an adequate number of CPUs and sufficient memory needed for the job. The config_multi_process command must be executed as soon as possible after starting the Oasys-RTL tool.

When starting a run from RTL, it is recommended that it is the first command in the script. When a run is started by reading the design or database from an existing ODB, the multiprocess settings are inherited from the Oasys-RTL run that had saved the ODB. If these settings need to be changed, it is recommended to do so immediately after reading the ODB. Once the multiprocess operation has started (for example, by requesting a timing report), the parallelism setting cannot be changed.

Examples

Example 1

This example shows the results of successfully setting the number of CPUs to 2.

```
[oasys-RTL]$ config_multi_process -local_host 2
-local_host 2
```

Example 2

This example shows an attempt to set the number of CPUs greater than the number of available CPUs. A warning message is displayed and the number of CPUs is set to the maximum number of CPUs available.

```
[oasys-RTL]$ config_multi_process -local_host 4
warning: Invalid request to use 4 CPUs on a host that has 2 processors;
this run will use 2 workers.  [MPG-212]
-local_host 2
```

Example 3

A call to the config_multi_process command too late in the flow will be ignored, producing the following warning.

```
[oasys-RTL]$ config_multi_process -local_host 5
warning: The MPG flow has been already activated with 3 CPUs.  Ignoring
the request to reset the local worker count to 5.  [MPG-211]
```

create_menu

Creates a new custom GUI menu for a specified command.

Usage

create_menu [-type command | separator] [-name <menu_name>]
[-text "<menu_text>"] [-command <command_name>]

Arguments

- -type command | separator

  Type of menu operation. Specify command to create a new command operation. Specify separator to create a line between commands in the menu. A type of command is the default.

- -name <menu_name>

  Specifies a handle for the new menu item.

- -text "<menu_text>"

  Specifies text to appear under the User menu in the GUI dropdown for the new command.

- -command <command_name>

  Specifies a command to execute when the menu item is selected.

Examples

This example shows how to create a new GUI menu under **User > Get Product Name**. The new menu executes the get_product command when it is clicked.

```
create_menu -type command -name Product -text "Get Product Name" \
   -command get_product
```

If you want to create a separator between commands, then use the -separator keyword between menu definitions. For example:

```
create_menu -type command -name read_verilog -text "Read Verilog" \
   -command read_verilog design.v
create_menu -type separator -text "Write Commands" \
create_menu -type command -name write_verilog -text "Write Verilog" \
   -command write_verilog out.v
```

Related Topics

General Commands

date

Outputs the current date and time.

Usage

date

Arguments

None.

Examples

To output the current date and time:

```
% date
Fri Feb 10 01:28:31 PM PST 2012
```

Related Topics

General Commands

encrypt

Encrypts a file such that only the Oasys-RTL tool can read it back.

Usage

encrypt <in_file> <out_file>

Arguments

- <in_file>

  The file to be encrypted.

- <out_file>

  The encrypted output file.

Description

Encrypts a file readable by the Oasys-RTL tool. It creates an output file with an .odb extension.

The following commands can be read as an encrypted file: read_def, read_lef, read_library, read_verilog, read_vhdl, and read_ctl.

Examples

This example creates an encrypted file named myMacro.odb:

```
% encrypt myMacro.lib myMacro.odb
```

The Oasys-RTL tool can then read it back as follows:

```
% read_library myMacro.odb
```

Related Topics

General Commands

exit

Exits the Oasys-RTL tool.

Usage

exit

Arguments

None.

Description

Exits the Oasys-RTL tool. Equivalent to the quit command.

Examples

The below command exits the Oasys-RTL tool:

```
% exit
```

Related Topics

quit

General Commands

get_cap_unit

Returns the SDC unit for capacitance.

Usage

get_cap_unit

Arguments

None.

Examples

```
%get_cap_unit
pf
```

Related Topics

report_units

set_units

General Commands

get_current_unit

Returns the SDC unit for current.

Usage

get_current_unit

Arguments

None.

Examples

```
% get_current_unit
ma
```

Related Topics

report_units

set_units

General Commands

get_leakage_power_unit

Returns the SDC unit for power.

Usage

get_leakage_power_unit

Arguments

None.

Examples

```
% get_leakage_power_unit
mw
```

Related Topics

report_units

set_units

General Commands

get_liberty_files

Returns the complete filenames of the loaded Liberty libraries.

Usage

get_liberty_files [-target_lib [<string>]]

Arguments

- -target_lib [<string>]

  Specifies that the Liberty files are to be returned for the specified target library. If a target library is not specified, all loaded Liberty library files are returned.

Examples

```
% get_liberty_files -target_lib std_vt
/libs/0p95/NangateOpenCellLibrary_45nm_SVT_worst_low_conditional_nldm.lib
/libs/0p85/
NangateOpenCellLibrary_45nm_SVT_slow_0p85V_conditional_nldm.lib
```

Related Topics

report_timing

get_licenses

Returns a list of Oasys-RTL features that are currently checked out.

Usage

get_licenses

Arguments

None.

Examples

[oasys-RTL]$ get_licenses

Related Topics

General Commands

get_log_file

Returns the full pathname of the log file that is being output for the current session.

Usage

get_log_file

Arguments

None.

Description

The get_log_file command returns a fully qualified pathname for the current session's log file.

Note

The behavior of this command changed with the 2016.1 release. Previous versions could return relative pathnames or leaf names. To ensure that a script will always return a consistent fully qualified pathname regardless of the tool version, use the Tcl "file join" command as in the following example:

```
set logFile [file join [pwd] [get_log_file]]
```

Examples

The following examples show the behavior of the get_log_file command with different methods of specifying the log file name with the **oasys** command.

Example 1: Default Log Location

Assume that the tool was launched from */home/user1/oasys_output* with the following command:

```
> oasys
```

The get_log_file command returns the following:

```
/home/user1/oasys_output/oasys.log.<xx>
```

where xx is a number that uniquely identifies the current log file.

Example 2: Log File Name Only

Assume that the tool was launched from */home/user1/oasys_output* with the following command:

```
> oasys -log my_logfile
```

The get_log_file command returns the following:

```
/home/user1/oasys_output/my_logfile.log
```

Example 3: Relative Path to Log File

Assume that the tool was launched from */home/user1/oasys_output* with the following command:

```
> oasys -log log_dir/logfile1
```

The get_log_file command returns the following:

```
/home/user1/oasys_output/log_dir/logfile1.log
```

Example 4: Absolute Path to Log File

Assume that the tool was launched from */home/user1/oasys_output* with the following command:

```
> oasys -log /home/user1/oasys_output/logs/session4
```

The get_log_file command returns the following:

```
/home/user1/oasys_output/logs/session4.log
```

Related Topics

General Commands

get_parameter

Returns the value of a parameter.

Usage

get_parameter <name>

Arguments

- <name>

  Specifies the parameter name.

Examples

This example shows how to get the value of a parameter, change that value, and then verify that it has changed:

```
[oasys-RTL]$ get_parameter array_naming_style
[%d]

[oasys-RTL]$ set_parameter array_naming_style "_%d"
info: Parameter 'array_naming_style' set to '_%d' [PARAM-104]

[oasys-RTL]$ get_parameter array_naming_style
_%d
```

Related Topics

General Commands

set_parameter

report_parameters

get_product

Returns the name of the product currently in use.

Usage

get_product

Arguments

None.

Examples

```
% get_product

psyncore
```

Related Topics

General Commands

get_res_unit

Returns the SDC unit for resistance (R).

Usage

get_res_unit

Arguments

None.

Examples

```
[oasys-RTL]$ get_res_unit
kohm
```

Related Topics

report_units

set_units

General Commands

get_time_unit

Returns the SDC unit for time.

Usage

get_time_unit

Arguments

None.

Examples

```
[oasys-RTL]$ get_time_unit
ns
```

Related Topics

report_units

set_units

General Commands

get_voltage_unit

Returns the SDC unit for voltage.

Usage

get_voltage_unit

Arguments

None.

Examples

```
[oasys-RTL]$ get_voltage_unit
mv
```

Related Topics

report_units

set_units

General Commands

instance_area

Displays the cumulative area of all instances in the design.

Usage

instance_area [-no_macros]

Arguments

- -no_macros

  Does not count the macro area.

Examples

```
% instance_area
18786130
```

Related Topics

General Commands

instance_count

Returns the number of instances in the design.

Usage

instance_count [-no_macros]

Arguments

- -no_macros

  Excludes macros in the instance count.

Description

Returns the total count of leaf instances in the entire design hierarchy. By default, macro cells are included in the count. You can exclude macro cells in the count by using the -no_macros option.

Examples

This example returns the number of instances in a design.

```
% instance_count
155254
```

Related Topics

General Commands

instance_area

get_macros

is_oasys_shell

Determines if you are in an Oasys-RTL shell.

Usage

is_oasys_shell

Arguments

None.

Description

Determines if you are in an Oasys-RTL shell. Invoking the command in an Oasys-RTL shell returns a 1. This command is particularly useful in scripts that may contain commands that are executable in different shell environments.

Examples

This code example works for both for Synopsys Design Compiler® and the Oasys-RTL tool in a single script.

```
if {[info commands is_dc_shell] != ""} {
      # put DC specific commands here
   }
if {[info commands is_oasys_shell] != ""} {
      # put Oasys-RTL specific commands here
   }
```

Related Topics

General Commands

man

Displays information on usage of the specified Oasys-RTL parameter or command.

Usage

man {<oasys_cmd> | <oasys_parameter>}

Arguments

- <oasys_cmd>

  Specifies the Oasys-RTL command name.

- <oasys_parameter>

  Specifies the Oasys-RTL parameter name.

Description

Displays text information on usage of the specified Oasys-RTL parameter or command. It functions similarly to the unix man command.

Examples

Example 1

The following command outputs usage information on the read_def command:

```
% man read_def
```

Example 2

The next command outputs usage information on the array_naming_style parameter:

```
% man array_naming_style
```

Related Topics

General Commands

message

Controls or counts the output of messages.

Usage

message [-limit] [-enable] [-count] <message_id> [-severity <info | warning | error>]

Arguments

- -limit [true | false]

  Specifies whether the suppression limit is enforced. A value of true specifies that the message suppression limit is enforced for the specified <message_id>. A value of false deactivates this limit. You normally limit the number of times a message is reported, if it is repeated multiple times while executing a command. The limit is controlled by the message_suppress_limit parameter. The default is true.

- -enable [true | false]

  Specifies whether to print the message. A value of true enables printing of the specified <message_id> message, while a value of false disables it. The default is true.

- -count

  Reports the number of times that the specified <message_id> message has been output.

- <message_id>

  Specifies a message id such as LEF-113. See message tab section to see a graphical view of the error messages.

- -severity <info | warning | error>

  Specifies to change the level of severity for messages. The default error messages in the tool cannot be downgraded.

Description

Controls or counts the output of messages. A detailed summary of the message for the specified id is output except when using the -count option.

Examples

Example

This example reports the number of times the VLOG-520 message occurred, the type of message, whether its output was enabled or limited, and the details the message.

```
% message VLOG-520
--------------------------------------------------------------------------
Message : expression truncated to fit target [VLOG-520]
Type : warning
Enabled : true
Limited : true
Count : 0
Detail : If the width of the right-hand side (RHS) expression is larger
than the width of the left-hand side (LHS) in an assignment, the MSBs of
the RHS expression will always be discarded to match the size of the
LHS. Truncating the sign bit of a signed expression may change the
sign of the result.
--------------------------------------------------------------------------
```

Example

This example disables the PARAM-100 message then enables it:

```
% get_parameter foo
Error: Invalid parameter 'foo' [PARAM-100]
% message -enable false PARAM-100
--------------------------------------------------------------------------
Message : invalid parameter [PARAM-100]
Type : error
Enabled : false
Limited : true
Count : 1
--------------------------------------------------------------------------

% get_parameter foo
% message -enable true PARAM-100
--------------------------------------------------------------------------
Message : invalid parameter [PARAM-100]
Type : error
Enabled : true
Limited : true
Count : 1
--------------------------------------------------------------------------
% get_parameter foo
Error: Invalid parameter 'foo' [PARAM-100]
```

Example

This example activates the message suppression limit on the PARAM-100 message:

```
% message -limit true PARAM-100
--------------------------------------------------------------------------
Message : invalid parameter [PARAM-100]
Type : error
Enabled : true
Limited : true
Count : 1
--------------------------------------------------------------------------
```

Example

The next example reports the number of times the LEF-103 message was output:

```
% message -count LEF-103
11
```

Related Topics

message_suppress_limit

General Commands

mgcdocs

Invokes the Oasys-RTL InfoHub giving access to the tool documentation in HTML or PDF format.

Usage

mgcdocs [-type < all | crm | qsg | rn | ug >]

Arguments

- -type < all | crm | qsg | rn | ug>

  Optional. Specifies which document to open in HTML format. The default is all documentation.

  all — All documents. This is the default.

  crm — Command Reference Manual.

  qsg — Quick Start Guide.

  rn — Release Notes.

  ug — User's Guide.

Description

This command opens the Oasys-RTL InfoHub, a web page that contains links to all locally installed PDF and HTML documentation and other on-line resources like Support Center. The InfoHub stays as a background job in the shell. You must set a valid DISPLAY environment variable.

You can also access the documentation from the Help menu in the GUI.

The search button () performs a search in the scope that you select. You can select the locally installed content or Support Center.

You can customize a document list with the "My Document List" scope.

The recommended method to open PDF manuals from the InfoHub is by setting the browser to open PDF files using a reader directly instead of a plug-in. For example, in Firefox go to the **Edit** > **Preferences > Applications** tab, select the Portable Document Format **Content Type**, and set the **Action** to the path of your preferred PDF reader.

For more details about supported browsers, see *The Mentor Documentation System* manual.

Examples

Example 1: Open the InfoHub

This example opens the InfoHub from the command shell.

```
[oasys-RTL]$ mgcdocs
```

Example 2: Open the User's Guide in a Web Browser

This example displays the HTML Format of the User's Guide in a web browser.

```
[oasys-RTL]$ mgcdocs -type ug
```

print_cpu

Reports the current memory usage and CPU and wall time.

Usage

print_cpu [<operation>]

Arguments

- <operation>

  Optionally reports the CPU time, wall time, and memory for a specified operation.

Examples

```
% print_cpu "load_design"

load_design at 00:00:06(cpu)/7:51:02(wall) 241MB(vsz)
```

Related Topics

General Commands

quit

Quits the Oasys-RTL tool.

Usage

quit

Arguments

None.

Description

Quits the Oasys-RTL tool and is equivalent to the exit command.

Examples

To quit and exit the Oasys-RTL tool, do the following:

```
% quit
```

Related Topics

exit

General Commands

redirect

Redirects the output of commands to a file or a variable.

Usage

redirect [-append] [-file | -variable] <target> <command_string>

Arguments

- -append

  Appends the output to <target>.
- -file

  Redirects the output to a file.
- -variable

  Redirects the output to a variable.
- <target>

  Specifies the target file or variable.
- <command_string>

  Specifies the command to execute.

Examples

The below example outputs the timing and design metric reports to a file called my.rpts:

```
% redirect -file my.rpts {report_timing -net;
report_design_metrics}
```

Related Topics

General Commands

remove_attribute

Removes specified attribute from a design object.

Usage

remove_attribute [-quiet] [-force] {<object> <attribute>}

Arguments

- -quiet

  Suppresses output of warnings.

- -force

  Forces dont_touch directive on a net, pin, or port inside an inferred hierarchy. Hierarchy is inferred during optimization when the Oasys-RTL tool groups logic and creates hierarchy. By default, dont_touch, dont_touch_network, and dont_touch_network_no_propagate directives do not apply to objects inside inferred hierarchy.

- <object> <attribute>

  Specifies the attribute of the object to remove, where the supported object-attribute pairs are described as follows:

  - <object> dont_touch

    Sets the dont_touch attribute to false on a specified object. The <object> argument must be one of the following: net, cell, inst, lib_cell, design, module.

  - <pins> dont_touch_network

    Sets the dont_touch_network attribute to false on the nets and instances in the transitive fanout of the specified pins.

  - <pins> dont_touch_network_no_propagate

    Sets the dont_touch_network_no_propagate attribute to false on the nets and instances in the transitive fanout of the specified pins. The propagation of the command stops at the first combinational cell.

Description

The supported attributes are dont_touch, dont_touch_network, and dont_touch_network_no_propagate.

Examples

Reset the dont_touch property on cellA/inv from true to false.

```
% remove_attribute [get_cell cellA/inv] dont_touch
```

Related Topics

General Commands

set_dont_touch

set_dont_touch_network

get_attribute

set_attribute

reset_parameter

Resets specified parameter to default value.

Usage

reset_parameter <name>

Arguments

- <name>

  Specifies a parameter name.

Examples

```
% reset_parameter parameter_naming_style
info: Parameter 'parameter_naming_style' set to '_%s%d'
[PARAM-104]
```

Related Topics

get_parameter

report_parameters

set_parameter

General Commands

save_gui_views

Saves the physical GUI views from the shell for a placed design.

Usage

save_gui_views [-physical] [-congestion] [-dynamic_power] [-leakage_power] [-hierarchical] [-criticality] [-floorplan] [-format {png | pdf}] [-outdir <output_directory>]

Arguments

- -physical

  Optional. Specifies to save the physical view. One or more of the view options are required.

- -congestion

  Optional. Specifies to save the congestion view. One or more of the view options are required.

- -dynamic_power

  Optional. Specifies to save the dynamic power view. One or more of the view options are required.

- -leakage_power

  Optional. Specifies to save the leakage power view. One or more of the view options are required.

- -hierarchical

  Optional. Specifies to save the hierarchy view. One or more of the view options are required.

- -criticality

  Optional. Specifies to save the criticality view. One or more of the view options are required.

- -floorplan

  Optional. Specifies to save the floorplan view. One or more of the view options are required.

- -format {png | pdf}

  Optional. Specifies the file format for saving GUI snapshots. The default is the *.png* format.

- -outdir <output_directory>

  Optional. Specifies the output directory to which files are written. If the argument is not used, or if the specified directory does not exist, the output files are saved in the current working directory.

Description

The save_gui_views command saves the GUI views to disk from the command line. The design should be placed before using this command. More than one view can be specified. If no views are specified, no GUI snapshot files are generated in the output directory.

Examples

The examples assume you have made an output directory in which to save GUI views:

```
mkdir ./output/gui_views
```

This example saves all types of GUI views in the specified output directory:

```
% save_gui_views -physical -outdir output/gui_views
```

This example saves only the congestion view in the specified output directory:

```
% save_gui_views -congestion -outdir output/gui_views
```

Related Topics

General Commands

start_gui

set_attribute

Sets a value for an attribute.

Usage

set_attribute [-quiet] [-class <class>] {<object> <attribute> <value>}

Arguments

- -quiet

  Suppresses output of warnings.

- -class <class>

  Specifies the type of object. The Oasys-RTL tool supports the following classes: Lib, lib_cell, lib_pin, cell, inst, design, module, net, pin, port, clock, power_domain.

- <object>

  Specifies the object. The Oasys-RTL tool only supports a pin object at the top level.

- <attribute>

  Specifies an attribute. These attributes can only be written and not read. Currently, only location and layer are supported in the Oasys-RTL tool.

- <value>

  This specifies the value of an attribute. The value is dependent on the <attribute>.

Description

This command sets a value for an attribute.

Examples

This example sets a location attribute:

```
% set_attribute -class pin DRAM23_N_REF_RES location "200 1000"
```

Related Topics

report_attribute

get_attribute

General Commands

set_design_effort

Sets the effort level of optimizing for leakage or congestion.

Usage

set_design_effort [-leakage *leakage_effort*] [-congestion *congestion_effort*]

Arguments

- -leakage *leakage_effort*

  Turns on/off the leakage optimization flow and specifies the effort level for minimizing leakage. Specify an integer from 0 to 2.

  - 0: No leakage optimization. (default)
  - 1: Perform leakage optimization without increase in instance count.
  - 2: Attempt additional leakage optimization, which may increase area and runtime.

- -congestion *congestion_effort*

  Specifies the effort level for minimizing congestion. Specify an integer from 1 to 3. The default is 1.

Description

Sets the effort level of optimizing for leakage or congestion. Must be set prior to using the synthesize command. The get_design_effort command reports the current settings of the set_design_effort options.

Examples

Sets the effort level for minimizing leakage to 2:

```
% set_design_effort -leakage 2
% get_design_effort
     leakage effort: 2
```

Related Topics

optimize

max_percentage_lvt

create_threshold_voltage_group

General Commands

get_design_effort

set_parameter

Changes the value of a parameter.

Usage

set_parameter {<name> <value>}

Arguments

- <name>

  Specifies parameter name.

- <value>

  Specifies parameter value.

Description

Changes the value of a parameter. Use get_parameter to get the setting of a specific parameter. Run the report_parameters command to get a list of all parameters and their settings.

Examples

This example shows how to view the value of a parameter, change its value, and verify that the change has been applied:

```
% get_parameter array_naming_style
[%d]

% set_parameter array_naming_style "_%d"
info: Parameter 'array_naming_style' set to '_%d' [PARAM-104]

% get_parameter array_naming_style
_%d
```

Related Topics

get_parameter

report_parameters

reset_parameter

General Commands

set_passphrase

Provides password phrase to read encrypted files.

Usage

set_passphrase <passphrase>

Arguments

- <passphrase>

  Specifies the password phrase.

Description

To protect your RTL files or library files, encrypt these files using PGP (Pretty Good Privacy, a computer program for the encryption and decryption of data). This prevents users from reading these files, unless they have access to the passphrase.

The set_passphrase command allows you to set the passphrase for decrypting PGP encrypted input files in the tool.

PGP encrypted files do not have to be decrypted before you can use them in the Oasys-RTL tool. The tool decrypts these files on the fly, without creating a decrypted copy on the hard disk.

Note

The ask_passphrase command opens a dialog window for entering the passphrase. Typed text is echoed as dots. The command set_passphrase requires a string, which is visible when entered.

Examples

A PGP encrypted file is created as follows:

```
> gpg-c myTopSecretDesign.v
Enter passphrase: <Type pass phrase>
Repeat passphrase: <Re-type pass phrase>
> rm myTopSecretDesign.v
```

This encrypts and creates the file myTopSecretDesign.v.gpg. A user that does not have the correct passphrase cannot read this file.

To use this file in the Oasys-RTL environment, do the following:

```
% set_passphrase <my_pass_phrase>
% read_verilog myTopSecretDesign.v.gpg
```

This reads the encrypted Verilog file.

Related Topics

ask_passphrase

General Commands

set_register_merging

Specifies the register merging attribute on cells or designs to control the register merging optimization.

Usage

set_register_merging [-verbose] [true | false] <object_list>

Arguments

- -verbose

  Specifies to print a message to indicate how many objects receive the register merging directive.

- <object_list>

  Specifies a list of instances or modules.

- [true | false]

  Enables register merging. The default is true (or enabled).

Description

Merges registers of equivalent functionality within a module. It is enabled by default. If set to false, the optimization is disabled. When the optimization is disabled, redundant registers by design intent are not merged.

Examples

Example 1

This example specifies that register merging is not allowed on U1:

```
% set_register_merging U1 FALSE
```

Example 2

This example shows how many objects have the register merging attribute set and prints a message:

```
% set_register_merging -verbose U1 U2 U3 TRUE
```

Related Topics

General Commands

set_selection

Selects the specified instance in the GUI.

Usage

set_selection <instance_name>

Arguments

- <instance_name>

  Specifies an instance in the design to select.

Examples

This example shows how to select the instance sparc6/ffu/ctl in the GUI:

```
% set_selection sparc6/ffu/ctl
```

Related Topics

get_selection

General Commands

set_user_attribute

Sets a value for a user defined attribute.

Usage

set_user_attribute <obj_list> <attribute_name> <attribute_value>
[-class {design | port | pin | cell | net | lib | lib_cell | lib_pin}] [-quiet]

Arguments

- <obj_list>

  Specifies the object list.

- <attribute_name>

  Specifies the name of the attribute to be applied.

- <attribute_value>

  Specifies the value of the attribute. The value must be within the legal range and type specified in the define_user_attribute command.

- -class {design | port | pin | cell | net | lib | lib_cell | lib_pin}

  Specifies the class. Only one value can be provided.

- -quiet

  Suppresses output of warnings.

Examples

The below example sets a design attribute called my_attr2, at the top of the design, with the value of abc:

```
% set_user_attribute [get_design top] my_attr2 abc -class design
```

Related Topics

define_user_attribute

report_attribute

General Commands

source

Reads a file and executes the Tcl commands from that file.

Usage

source [-mode <mode>] [-echo] [-verbose] <file>

Arguments

- -mode <mode>

  The mode of the design. The file is read conditionally based on the <mode>.

- -echo

  Displays the command and the result of every command executed.

- -verbose

  Displays information and execution statistics about the file. Can be used with the -echo option to correlate commands and their output for debugging.

- <file>

  The name of the file to be read. If the Tcl variable search_path has been set, the file is searched for in any of these paths.

Description

This command parses the specified <file> and evaluates it as a Tcl script.

Any SDC file can be sourced (using Tcl native source command). However, in that case, any resulting message is printed immediately after it occurs.

Examples

Example 1

This example executes the contents of the Tcl script called *my_design.tcl*.

```
% source  my_design.tcl
```

Example 2

This example sources the SDC file *my_design.sdc* and sets the mode to fast.

```
% source -mode fast my_design.sdc
```

Example 3

This example shows the usage of the -echo and -verbose options.

```
% source -echo -verbose  constraints.tcl

starting source constraints.tcl  at 00:00:07(cpu)/4:19:56(wall) 64MB(vsz)
info:    File 'constraints.tcl', resolved to path './scripts' using
search_path variable.  [CMD-126]
> create_clock -name clk_push -period 0.7 -waveform { 0 0.4 } clk_push
> create_clock -name clk_pop -period 0.7 -waveform { 0 0.4 } clk_pop
> set_clock_groups -logically_exclusive -name exclusive -group clk_pop -
group clk_push
finished source new.sdc at 00:00:07
```

Example 4

This example prevents the display of commands and results to stdout and the log file.

```
% source -echo false run.tcl > /dev/null
```

Related Topics

read_sdc

General Commands

start_gui

Starts the GUI when in the command mode.

Usage

start_gui [-echo] [-title <name>]

Arguments

- -echo

  Echoes commands while loading files.

- -title <name>

  Sets the window title to <name>.

Examples

The below command starts the GUI from the command mode:

```
% start_gui
```

Related Topics

stop_gui

General Commands

stop_gui

Stops the GUI and goes to the command mode.

Usage

stop_gui

Arguments

None.

Examples

To stop the GUI and go to the command mode, do the following:

```
% stop_gui
```

Related Topics

start_gui

General Commands

Appendix A
Oasys-RTL Parameters

Parameters are used to configure the Oasys-RTL synthesis flow. To use a parameter, apply the set_parameter command: set_parameter <parameter> <value>.

Table A-1. Parameters

Parameter	Description
always_naming_style	Defines the Verilog always block label to be prefixed to an instance name.
architecture_naming_style	Defines a suffix architecture name for an entity if two architectures of a single entity are used.
array_instance_naming_style	Controls the naming style for Verilog arrays of instances.
array_naming_style	Defines the naming style for the bits of a register.
auto_define_derived_test_clocks	Determines if DFT automatically identifies and tags any multi-input logic gate on the test clock path as a separate test clock for scan chain connection.
block_naming_style	Defines a label prefix for instance names of VHDL block statements or Verilog generate begin/end statements.
capacitance_units_for_reports	Defines the capacitance units for reports.
clock_gate_naming_style	Defines the naming style of the integrated clock-gating cells inserted by the Oasys-RTL tool.
create_liberty_clocks	Specifies whether to instantiate clocks from library cells.
current_units_for_reports	Defines the current units for reports
date	Outputs the date of the software build.
detailed_report_header	Causes report_timing output to include information on any cell derating that was applied to the design.
dft_rising_edge_head_lockup	Specifies that rising-edge lockup-flops need to be inserted at the head end of chains and the compression segments if the last flops of a chain are not rising-edge triggered.

Table A-1. Parameters (cont.)

Parameter	Description
dft_rising_edge_tail_lockup	Specifies that rising-edge lockup-flops need to be inserted at the tail end of chains and the compression segments if the last flops of a chain are not rising-edge triggered.
die_aspect_ratio	Defines the aspect ratio of the die.
discrete_clock_gate_naming_style	Defines the naming style of the discrete clock-gating cells inserted by the Oasys-RTL tool. Discrete clock-gating cells are created from discrete cells when integrated clock-gating cells are not available in the library.
dont_synthesize	Specifies modules that should be ignored during synthesis.
energy_units_for_reports	Defines the energy units for reports.
error_on_blackbox	Specifies whether synthesis errors out when it encounters an unresolved reference.
generate_naming_style	Defines the naming style for for-generate statements.
if_generate_naming_style	Specifies how the tool uses an if-generate label to create a prefix for instance names.
interface_naming_style	Controls the naming style for System Verilog interface type ports.
latch_naming_style	Controls the naming style of inferred latches.
library_naming_style	Sets a prefix library name for naming two same-named entities that reside in different libraries.
max_call_depth	Controls the maximum depth of an RTL function or task call chain during elaboration.
max_loop_limit	Controls the maximum number of iterations of a loop in the RTL.
max_percentage_lvt	Sets the maximum percentage of low voltage threshold (LVT) cells when optimizing leakage.
merge_flipflops	Turns off or on the merging of functionally identical flip-flops.
merge_latches	Turns off or on the merging of functionally identical latches.
message_suppress_limit	Specifies the message suppression limit.

Table A-1. Parameters (cont.)

Parameter	Description
multibit_naming_style	Defines the naming style for multi-bit registers used for multi-bit mapping.
parameter_naming_style	Controls the naming style for parameterized modules.
power_ignore_pg_for_leaf_cells	Specifies whether the power and ground nets defined in liberty files are ignored for standard cells and macros.
power_units_for_reports	Defines the power units for reports.
power_upf_naming_style	Specifies a prefix for generated upf names.
power_upf_use_els_for_ls	Determines if enable level shifters is converted to regular level shifters.
preserve_constant_flipflops	Specifies not to remove flip flops that are tied to a constant input.
preserve_constant_latches	Specifies not to remove latches that are tied to a constant input.
preserve_dangling_flipflops	Specifies not to remove unconnected flip flops.
preserve_dangling_latches	Specifies not to remove unconnected latches.
process_naming_style	Specifies the VHDL process label to be prefixed to an instance name.
program	Returns the name of the software product that is currently running (such as oasys or oasysGui). This is a read only parameter.
record_naming_style	Specifies the naming style for VHDL record or SystemVerilog struct fields.
reg_naming_style	Controls the naming style for registers.
resistance_units_for_reports	Defines the resistance units for reports.
retime_register_naming_style	Defines the prefix added to an inserted retime register.
retime_register_prefix	Defines the prefix added to an inserted retime register.
scan_lockup_instance_prefix	Defines the prefix added to the instance name of an inserted scan lockup register.
sdc_continue_on_error	Specifies that the Oasys-RTL tool continues reading in SDC files with the source command even if the file contains errors. This only applies to source command and not the read_sdc command.

Table A-1. Parameters (cont.)

Parameter	Description
time_units_for_reports	Defines the time units for reports.
time_units_for_sdc	Defines the time units for SDC.
timing_report_significant_digits	Defines the number of significant digits to use when reporting timing values
ungroup_small_hierarchies	Specifies the maximum number of leaf cells in a user module that causes the module to automatically dissolve during area optimization.
uniquify_naming_style	Defines the naming style of the designs that have been made unique by the uniquify command.
upf_write_pg_to_netlist	Controls whether write_verilog adds power and ground nets to the netlist.
version	Displays the software version. This is a read only parameter.
vhdl_sort_files	Controls how VHDL files are read.
voltage_units_for_reports	Defines the voltage units for reports
write_implicit_wire_decls	Specifies that write_verilog writes out declarations for implicit wires.
write_sdc_split_line	Specifies that if the SDC line is too long, the command is written on multiple lines.

always_naming_style

Defines the Verilog always block label to be prefixed to an instance name.

Usage

always_naming_style <format>

Arguments

- <format>

  Specifies the always naming style format. The format is any string. The default is no always block label.

Description

Defines a prefix for registers defined in a Verilog labeled always block. A labeled always block is an always block which directly contains a labeled begin-end block. The default is no prefix.

Examples

```
% set_parameter always_naming_style areg_
```

Related Topics

get_parameter

set_parameter

reset_parameter

report_parameters

architecture_naming_style

Defines a suffix architecture name for an entity if two architectures of a single entity are used.

Usage

architecture_naming_style <format>

Arguments

- <format>

  Specifies the architecture naming style format.

Description

Used to uniquify a module when a VHDL entity has multiple architectures. The default is %s_%s.

Examples

```
% set_parameter architecture_naming_style %s__%s
```

Related Topics

Oasys-RTL Parameters

set_design_effort

get_parameter

set_parameter

reset_parameter

report_parameters

array_instance_naming_style

Controls the naming style for Verilog arrays of instances.

Usage

array_instance_naming_style <format>

Arguments

- <format>

  Specifies the array instance naming style with a format of “%s%d”. “%s” is the instance name and “%d” is the array index.

Description

Controls the naming of Verilog instance arrays. Given this array instance:

submod inst[3:0] (.q(q), .d(d));

the default format "%s%d" produces these instances:

inst3, inst2, inst1, inst0

If the array_instance_naming_style is the following:

%s_%d

the instances are the following:

inst_3, inst_2, inst_1, inst_0

Examples

```
% set_parameter array_instance_naming_style %s_%d
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

array_naming_style

Defines the naming style for the bits of a register.

Usage

array_naming_style <format>

Arguments

- <format>

  Specifies the array naming style format.

Description

Specifies a suffix for naming a bit of a multi-bit register. Also for naming one word of a multi-dimensional array. The default is [%d].

Examples

```
% set_parameter array_naming_style <%d>
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

auto_define_derived_test_clocks

Determines if DFT automatically identifies and tags any multi-input logic gate on the test clock path as a separate test clock for scan chain connection.

Usage

auto_define_derived_test_clocks [false | true]

Arguments

- false

  Specifies that DFT does not automatically identify and tag any multi-input logic gate on the test clock path as a separate test clock for scan chain connection. This is the default.

- true

  Specifies that DFT automatically identifies and tags any multi-input logic gate on the test clock path as a separate test clock for scan chain connection. Clock gates are excluded for auto-tagging of derived test clocks.

Examples

```
% set_define_derived_test_clocks true
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

block_naming_style

Defines a label prefix for instance names of VHDL block statements or Verilog generate begin/end statements.

Usage

block_naming_style <format>

Arguments

- <format>

  Specifies the block naming style format.

Description

This style applies to VHDL block statements and labeled Verilog concurrent begin-end blocks that are not immediately within an if-generate or for-generate statement.

A VHDL block statement takes the following form (nested blocks):

```
b1: block
begin
        b2: block
        begin
        u1 : sub port map(...);
        end block;
end block;
```

A Verilog labeled begin/end block takes this form:

```
begin: b1
        begin: b2
          sub u1(...);
        end
end
```

If the block_naming_style is set to "%s_", then both of these examples produce an instance named "b1_b2_u1".

Examples

```
% set_parameter block_naming_style %s_
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

capacitance_units_for_reports

Defines the capacitance units for reports.

Usage

capacitance_units_for_reports {ff | pf}

Arguments

- {ff | pf}

  Specifies the capacitance unit for reports. The default is ff.

Examples

```
% set_parameter capacitance_units_for_reports pf
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

clock_gate_naming_style

Defines the naming style of the integrated clock-gating cells inserted by the Oasys-RTL tool.

Usage

clock_gate_naming_style <naming_style>

Arguments

- <naming_style>

  Specifies the naming style for the clock-gating cells added by the tool. Use %s for the signal name of the register being gated. The default is clk_gate_%s_reg.

Examples

```
% set_parameter clock_gate_naming_style MYCG_%s_reg
```

create_liberty_clocks

Specifies whether to instantiate clocks from library cells.

Usage

create_liberty_clocks [true | false]

Arguments

- true

  Specifies to instantiate clocks from library cells.

- false

  Specifies not to instantiate clocks from library cells.

Description

Specifies whether to instantiate clocks from library cells. Each instantiated clock is assigned a unique name derived from the instance of the library cell.

Before instantiating a clock, you must set the following attributes in the *.lib* file:

- **clock_pin —** must be an output pin defined in the cell
- **master_pin —** must be an input pin defined in the cell
- **divided_by —** must be a positive number

For example, you can configure the attributes in the *.lib* file as follows:

```
generated_clock (gc) {
     clock_pin : "Y";
     master_pin : "A";
     divided_by : 2;
}
```

Examples

```
% set_parameter create_liberty_clocks true
```

Related Topics

create_clock

current_units_for_reports

Defines the current units for reports

Usage

current_units_for_reports { A | mA | uA |nA }

Arguments

- { A | mA | uA |nA }

  Specifies the current units for reports. The default value is “uA”.

Examples

```
set_parameter current_units_for_reports mA
```

Related Topics

Oasys-RTL Parameters

date

Outputs the date of the software build.

Usage

date

Arguments

None.

Examples

```
% get_parameter date
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

detailed_report_header

Causes report_timing output to include information on any cell derating that was applied to the design.

Usage

detailed_report_header [false | true]

Arguments

- false

  Specifies not to include information on any cell derating applied to the design. This is the default.

- true

  Creates a header in the timing report output that includes information on any cell derating applied to the design.

Examples

% set_parameter detailed_report_header true

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

dft_rising_edge_head_lockup

Specifies that rising-edge lockup-flops need to be inserted at the head end of chains and the compression segments if the last flops of a chain are not rising-edge triggered.

Usage

dft_rising_edge_head_lockup [true | false]

Arguments

- true

  Specifies that rising-edge lockup-flops need to be inserted at the head end of chains and the compression segments if the last flops of a chain is not rising-edge triggered.

- false

  Specifies that rising-edge lockup-flops are not inserted.

Examples

```
% set_parameter dft_rising_edge_head_lockup true
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

dft_rising_edge_tail_lockup

Specifies that rising-edge lockup-flops need to be inserted at the tail end of chains and the compression segments if the last flops of a chain are not rising-edge triggered.

Usage

dft_rising_edge_tail_lockup [true | false]

Arguments

- true

  Specifies that rising-edge lockup-flops need to be inserted at the tail end of chains and the compression segments if the last flops of a chain is not rising-edge triggered.

- false

  Specifies that rising-edge lockup-flops are not inserted.

Examples

```
% set_parameter dft_rising_edge_tail_lockup true
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

die_aspect_ratio

Defines the aspect ratio of the die.

Usage

die_aspect_ratio <ratio>

Arguments

- <ratio>

  Specifies the aspect ratio of the die (height/width). The default is 1.0.

Examples

```
% set_parameter die_aspect_ratio 0.8
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

discrete_clock_gate_naming_style

Defines the naming style of the discrete clock-gating cells inserted by the Oasys-RTL tool. Discrete clock-gating cells are created from discrete cells when integrated clock-gating cells are not available in the library.

Usage

discrete_clock_gate_naming_style <string>

Arguments

- <string>

  Specifies the naming style for the discrete clock-gating cells added by the tool. The default is rt_clk_gate_%s_reg. The %s placeholder is for the name of the signal or the register bank which is being gated.

Examples

```
% set_parameter discrete_clock_gate_naming_style RTCG_%s_reg
```

dont_synthesize

Specifies modules that should be ignored during synthesis.

Usage

dont_synthesize <module_list>

Arguments

- <module_list>

  Specifies a list of modules as a string or a tcl list of strings. Currently does not support wildcards.

Examples

```
% set_parameter dont_synthesize mod1
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

energy_units_for_reports

Defines the energy units for reports.

Usage

energy_units_for_reports { nJ | pJ | fJ | aJ }

Arguments

- { nJ | pJ | fJ | aJ }

  Specifies the energy units used in reports. The default value is “pJ”.

Examples

```
set_parameter energy_units_for_reports nJ
```

error_on_blackbox

Specifies whether synthesis errors out when it encounters an unresolved reference.

Usage

error_on_blackbox [false | true]

Arguments

- false

  Specifies that the synthesize command does not error out when it encounters an unresolved reference. This is the default.

- true

  Specifies that the synthesize command errors out when it encounters an unresolved reference.

Examples

```
% set_parameter error_on_blackbox true
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

generate_naming_style

Defines the naming style for for-generate statements.

Usage

generate_naming_style <format>

Arguments

- <format>

  Specifies the generate naming style format. If both "%s" and "%d" are specified, then a prefix is set for instance names. If only "%d" is specified, then a suffix is set for instance names.

Description

Prefix or suffix for instances and registers defined within a for-generate. If "%s" is present, this creates a prefix to the instance name. If "%s" is omitted, this produces a suffix.

For instance:

```
genvar i, j;
for (i = 0; i < 8; i = i + 1)
   begin: foo
   for(j = 2; j < 5; j = j + 1)
      begin: bar
      submod u1(...);
   end
end
```

If generate_naming_style is set to "%s_%d_", then instances of submod are named as follows:

```
foo_0_bar_2_u1, ...,foo_7_bar_5_u1
```

If generate_naming_style is set to "_%d", then instances of submod are named as follows:

```
u1_0_2, ..., u1_7_5
```

For registers within a named process, the generate prefix goes before the process name, while the generate suffix goes after the process name (but before the register name).

Examples

```
% set_parameter generate_naming_style %s_%d
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

if_generate_naming_style

Specifies how the tool uses an if-generate label to create a prefix for instance names.

Usage

if_generate_naming_style <format>

Arguments

- <format>

  Specifies the if-generate naming style format. The *%s* string is a placeholder for the label name.

Description

The default is to not use the if-generate label in the instance name.

Examples

Example 1: Default Prefix for Instance Names

In the following Verilog code, a begin-end block within an if-generate statement has the *b1* label:

```
generate

if (p1 == p2) begin: b1
   sub u1(...);
end

endgenerate
```

By default, the instance name is *u1*. It does not include the *b1* label.

Example 2: Set the Prefix Naming Style to Be the Label Followed by an Underscore

This example sets the prefix for the instance names of if_generate statements to be the label followed by an underscore.

```
[oasys-RTL]$ set_parameter if_generate_naming_style %s_
```

The name of the instance in Example 1 becomes *b1_u1*.

Related Topics

Oasys-RTL Parameters

interface_naming_style

Controls the naming style for System Verilog interface type ports.

Usage

interface_naming_style <format>

Arguments

- <format>

  Specifies the format for generated names of interface type ports. The default naming style is "%s_. The %s in the <format> string represents the name of the interface instance.

Description

Specifies the naming style of generated names for module ports that have an interface type. This parameter may be used to resolve name collisions of generated interface type port names with other identifiers.

Examples

This example uses the interface_naming_style parameter to change the naming style of the interface type ports. Consider the following RTL:

```
interface intf;
   wire x, y, z;
   modport mp(input x, y, output z);

   assign z = x ^ y;
endinterface

module sub(intf ii, input a, output o);
…

endmodule
…
```

The default setting of the interface_naming_style results in the following port names:

```
module sub(ii_x, ii_y, ii_z, a, o);
   input ii_x;
   input ii_y;
   output ii_z;
   input a;
   output o;
...
```

The default naming style can be changed to *%s_IF_* with the interface_naming_style parameter.

```
[oasys-RTL]$ set_parameter interface_naming_style %s_IF_
```

The resulting port names are as follows:

```
module sub(ii_IF_x, ii_IF_y, ii_IF_z, a, o);
   input ii_IF_x;
   input ii_IF_y;
   output ii_IF_z;
   input a;
   output o;...
```

latch_naming_style

Controls the naming style of inferred latches.

Usage

latch_naming_style <format>

Arguments

- <format>

  Specifies the latch naming style format.

Description

Controls the naming style of inferred latches. The default is “”. When this parameter is not specified or set to “”, a latch is named according to the value of the reg_naming_style parameter. The format string, when not empty, must contain exactly one “%s” format.

Examples

```
% set_parameter latch_naming_style %s_latch
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

library_naming_style

Sets a prefix library name for naming two same-named entities that reside in different libraries.

Usage

library_naming_style <format>

Arguments

- <format>

  Specifies the library naming style format. The default is %s_%s.

Description

Designs can have multiple entities or modules that share the same name while residing in different logical libraries. When they are integrated into the design during elaboration, the library_naming_style sets the naming style for generating unique names.

The default is %s_%s where the first %s is the logical library name and the second %s is the name of the module or entity. If an entity “foo” exists in lib1 and lib2, the module names are *lib1_foo* and *lib2_foo*, respectively.

Examples

```
% set_parameter library_naming_style %s__%s
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

max_call_depth

Controls the maximum depth of an RTL function or task call chain during elaboration.

Usage

max_call_depth <depth>

Arguments

- <depth>

  Specifies the maximum depth of an RTL function or task call chain during elaboration. The value may be set to any positive integer value. The default is 1000.

Description

Controls the maximum depth of an RTL function or task call chain during elaboration. Use this command as a sanity check to prevent endless recursion. If the maximum depth is surpassed, an error is reported.

Examples

```
% set_parameter max_call_depth 100
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

max_loop_limit

Controls the maximum number of iterations of a loop in the RTL.

Usage

max_loop_limit <limit>

Arguments

- <limit>

  Specifies the maximum number of iterations of a loop in the RTL code. The value may be set to any positive integer value. The default is 1000.

Description

Controls the maximum number of iterations of a loop in the RTL. It is intended as a sanity check for non-terminating loops. If a loop in the RTL iterates more than this number of times, an error is reported. For instance:

for (i = 0; i < 10000; i = i + 1)

results in the error:

error: loop limit (1000) exceeded (see max_loop_limit parameter) ((for.v:9)[6]) [VLOG-552]

Examples

```
% set_parameter max_loop_limit 100
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

max_percentage_lvt

Sets the maximum percentage of low voltage threshold (LVT) cells when optimizing leakage.

Usage

max_percentage_lvt ***percentage***

Arguments

- ***percentage***

  Specifies the maximum percentage of LVT cells to be used when optimizing leakage. The default is 100 percent.

Description

Sets the maximum percentage of LVT cells when optimizing leakage. This parameter only has an effect if leakage optimization is enabled with the set_design_effort command. The leakage flow attempts to limit LVT cells to the percentage specified; however, this may negatively impact timing. You can run optimize again to attempt fixing timing problems.

Examples

% set_parameter max_percentage_lvt 20

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

merge_flipflops

Turns off or on the merging of functionally identical flip-flops.

Usage

merge_flipflops [true | false]

Arguments

- true

  Specifies that optimization can declone/merge functionally of identical flip-flops. This is the default.

- false

  Specifies that optimization does not declone/merge functionally of identical flip-flops.

Examples

```
% set_parameter merge_flipflops false
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

merge_latches

Turns off or on the merging of functionally identical latches.

Usage

merge_latches [true | false]

Arguments

- true

  Specifies that optimization can declone/merge functionally of identical latches. This is the default.

- false

  Specifies that optimization does not declone/merge functionally of identical latches.

Examples

```
% set_parameter merge_latches false
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

message_suppress_limit

Specifies the message suppression limit.

Usage

message_suppress_limit <integer>

Arguments

- <integer>

  Specifies the message suppression limit. The default is 10.

Description

Specifies the message suppression limit. A value less than 1 does not suppress messages. A value of X that is greater or equal to 1 suppresses messages (per message ID or group) when the count exceeds X.

For example, if X=10 and the total number of messages for a particular message ID or group is more than 10, 10 messages are displayed and the remainder are suppressed.

Examples

```
% set_parameter message_suppress_limit 20
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

multibit_naming_style

Defines the naming style for multi-bit registers used for multi-bit mapping.

Usage

multibit_naming_style <format>

Arguments

- <format>

  Specifies the format of the naming style for multi-bit registers. The format is a string of alphanumeric characters that follow up to two %s placeholders for the group name and the name of the original single-bit registers.

  If the format contains two %s placeholders, the first %s placeholder is for the group name, the following string is the delimiter that separates the group name, and the second %s placeholder is for the original register names that are delimited by its following string.

  ```
  %s<delimiter for goup name>%s<delimiter between register names>
  ```

  If only one %s placeholder appears in the format, it is for the delimiter between the original register names.

  The default format is %s which is the concatenated names of the original registers.

Description

Defines the naming style to specify prefixes and delimiters for multi-bit registers that have been mapped by the map_to_multibit command. The tool accepts up to two %s placeholders in the format for the group name and the names of the original single-bit registers.

Examples

Consider two registers, *A_reg_0_* and *A_reg_1_*, that both belong to the *allA* group.

The multi-bit register name in the default style is:

```
A_reg_0_A_reg_1_
```

This example uses the group name as a prefix, the # sign to delimit the group name, and the _ symbol to delimit the register names.

```
% set_parameter multibit_naming_style %s#%s_
```

The multi-bit register name becomes:

```
allA#A_reg_0__A_reg_1_
```

This example uses no prefix to delimit the group name, and the # symbol to delimit the register names.

```
% set_parameter multibit_naming_style %s#
```

The multi-bit register name becomes:

```
A_reg_0_#A_reg_1_
```

parameter_naming_style

Controls the naming style for parameterized modules.

Usage

parameter_naming_style <format>

Arguments

- <format>

  Specifies the parameter naming style format. The default is _%s%d. Names are mapped into "__parameterizedXxx" suffixes.

Examples

```
% set_parameter parameter_naming_style __%s%d
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

power_ignore_pg_for_leaf_cells

Specifies whether the power and ground nets defined in liberty files are ignored for standard cells and macros.

Usage

power_ignore_pg_for_leaf_cells [true | false]

Arguments

- true

  Specifies that power and ground nets defined in the liberty files (.lib) are ignored for standard cells, including macros. This is the default.

- false

  Specifies that power and ground nets defined in the liberty files (.lib) are read from the libraries.

Description

Specifies whether the power and ground nets defined in liberty files are ignored for standard cells and macros. Without those power/ground nets, cells cannot be connected to supply nets using connect_supply_net in UPF.

Examples

```
% set_parameter power_ignore_pg_for_leaf_cells false
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

power_units_for_reports

Defines the power units for reports.

Usage

power_units_for_reports { mW | uW | nW | pW }

Arguments

- { mW | uW | nW | pW }

  Specifies the power unit for reports. The default value is uW

Examples

```
set_parameter power_units_for_reports nW
```

power_upf_naming_style

Specifies a prefix for generated upf names.

Usage

power_upf_naming_style <format>

Arguments

- <format>

  Specifies a prefix for generated upf names. The default is %s%s%s.

Examples

```
% set_parameter power_upf_naming_style %s_%s_%s
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

power_upf_use_els_for_ls

Determines if enable level shifters is converted to regular level shifters.

Usage

power_upf_use_els_for_ls [false | true]

Arguments

- false

  Instructs commit_upf not to use enable level shifters as regular level shifters. This is the default.

- true

  Instructs commit_upf to use enable type level shifters as regular level shifters by connecting the control line to a constant 1 or 0.

Description

With this parameter set to true, commit_upf uses the enable type level shifter as buffer type level shifters. Depending on the functionality, the enable pins are connected to a constant 1 or 0.

Examples

```
% set_parameter power_upf_use_els_for_ls true
```

Related Topics

Oasys-RTL Parameters

set_parameter

get_parameter

report_parameters

reset_parameter

preserve_constant_flipflops

Specifies not to remove flip flops that are tied to a constant input.

Usage

preserve_constant_flipflops {true | false}

Arguments

- true

  Preserves constant flip flops.

- false

  Removes constant flip flops.

Examples

```
% set_parameter preserve_constant_flipflops true
```

Related Topics

Oasys-RTL Parameters

preserve_constant_latches

preserve_dangling_flipflops

preserve_dangling_latches

preserve_constant_latches

Specifies not to remove latches that are tied to a constant input.

Usage

preserve_constant_latches {true | false}

Arguments

- true

  Preserves constant latches.

- false

  Removes constant latches.

Examples

```
% set_parameter preserve_constant_latches true
```

Related Topics

Oasys-RTL Parameters

preserve_constant_flipflops

preserve_dangling_flipflops

preserve_dangling_latches

preserve_dangling_flipflops

Specifies not to remove unconnected flip flops.

Usage

preserve_dangling_flipflops {true | false}

Arguments

- true

  Preserves dangling flip flops.

- false

  Removes dangling flip flops.

Examples

```
% set_parameter preserve_dangling_flipflops true
```

Related Topics

Oasys-RTL Parameters

preserve_constant_flipflops

preserve_constant_latches

preserve_dangling_latches

preserve_dangling_latches

Specifies not to remove unconnected latches.

Usage

preserve_dangling_latches {true | false}

Arguments

- true

  Preserves dangling latches.

- false

  Removes dangling latches.

Examples

```
% set_parameter preserve_dangling_latches true
```

Related Topics

Oasys-RTL Parameters

preserve_constant_flipflops

preserve_dangling_flipflops

preserve_constant_latches

process_naming_style

Specifies the VHDL process label to be prefixed to an instance name.

Usage

process_naming_style <format>

Arguments

- <format>

  Specifies the process naming style format. The default is no process label.

Examples

```
% set_parameter process_naming_style "%s_"
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

program

Returns the name of the software product that is currently running (such as oasys or oasysGui). This is a read only parameter.

Usage

program

Arguments

None.

Examples

```
% get_parameter program
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

record_naming_style

Specifies the naming style for VHDL record or SystemVerilog struct fields.

Usage

record_naming_style <format>

Arguments

- <format>

  Specifies the record naming style format. The default is _%s. Other common styles are ".%s" or "[%s]". The %s represents the name of the record/struct field.

Examples

```
% set_parameter record_naming_style __%s
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

reg_naming_style

Controls the naming style for registers.

Usage

reg_naming_style <format>

Arguments

- <format>

  Specifies the register naming style format. The default is %s_reg.

Examples

```
% set_parameter reg_naming_style %s_regist
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

resistance_units_for_reports

Defines the resistance units for reports.

Usage

resistance_units_for_reports {ohm | mohm}

Arguments

- {ohm | mohm}

  Specifies the resistance unit for reports. The default is ohm.

Examples

Set resistance units to milliohms.

```
% set_parameter resistance_units_for_reports mohm
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

retime_register_naming_style

Defines the prefix added to an inserted retime register.

Usage

retime_register_naming_style <prefix>

Arguments

- <prefix>

  Specifies the prefix for an inserted retime register. The default is retime_reg_%d_s%d.

Description

Defines the prefix for the instance name of an inserted retime register. The string requires two %d values, one for a unique number for a register on a pipeline stage and another for the number of the pipeline stage.

Examples

```
% set_parameter retime_register_naming_style retime_%d_s%d
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

retime_register_prefix

Defines the prefix added to an inserted retime register.

Usage

retime_register_prefix <prefix>

Arguments

- <prefix>

  Specifies the prefix string for the naming of an inserted retime register. The default prefix string is retime_reg.

Description

Works with retime_register_naming_style parameter to define the complete naming style.

Examples

```
% set_parameter retime_register_prefix myretime_reg
```

scan_lockup_instance_prefix

Defines the prefix added to the instance name of an inserted scan lockup register.

Usage

scan_lockup_instance_prefix <prefix>

Arguments

- <prefix>

  Specifies the prefix for the scan lockup registers. The default is lockup.

Examples

```
% set_parameter scan_lockup_instance_prefix lockup_
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

sdc_continue_on_error

Specifies that the Oasys-RTL tool continues reading in SDC files with the source command even if the file contains errors. This only applies to source command and not the read_sdc command.

Usage

sdc_continue_on_error [true | false]

Arguments

- true

  Specifies to continue reading a file that contains errors. This is the default.

- false

  Specifies not to continue reading a file that contains errors.

Examples

```
% set_parameter sdc_continue_on_error false
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

time_units_for_reports

Defines the time units for reports.

Usage

time_units_for_reports {ps | ns}

Arguments

- {ps | ns}

  Specifies the time unit for reports. The default is ps.

Examples

```
% set_parameter time_units_for_report ns
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

time_units_for_sdc

Defines the time units for SDC.

Usage

time_units_for_sdc {ps | ns}

Arguments

- {ps | ns}

  Specifies the time unit for SDC. The default is ns.

Examples

```
% set_parameter time_units_for_SDC ps
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

timing_report_significant_digits

Defines the number of significant digits to use when reporting timing values

Usage

timing_report_significant_digits <integer>

Arguments

- <integer>

  Specifies the number of significant digits to use when reporting timing values. This value should be a positive integer. The default value is 3.

Examples

```
set_parameter timing_report_significant_digits 5
```

ungroup_small_hierarchies

Specifies the maximum number of leaf cells in a user module that causes the module to automatically dissolve during area optimization.

Usage

ungroup_small_hierarchies <value>

Arguments

- <value>

  Specifies the maximum number of leaf cells in a user module that cause the module to automatically dissolve during area optimization. The default is 30, resulting in all user modules with 30 or less leaf instances being ungrouped. This parameter can be disabled by specifying a value of 0.

Description

Specifies the maximum number of leaf cells in a user module that cause the module to automatically dissolve during area optimization. Any user modules containing more leaf cells than this value is automatically dissolved.

Examples

Specify that all user modules with less than 21 modules are dissolved.

```
% set_parameter ungroup_small_hierarchies 20
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

uniquify_naming_style

Defines the naming style of the designs that have been made unique by the uniquify command.

Usage

uniquify_naming_style <naming_style>

Arguments

- <naming_style>

  Specifies the naming style for uniquified modules. The placeholders, %s and %d, are used for the original name of the module and its unique number, respectively. The default is %s__%d, which creates module names with the module name and a unique number separated by two underscores.

Description

When you use the uniquify command, the tool creates unique copies of any reference module that has multiple instances. Each instance of a reference module refers to a unique reference module. Similarly, during constraining and optimization, the tool may require certain references to be uniquified to provide a unique context to each instance. The uniquify_naming_style parameter specifies the naming scheme for the uniquified modules.

Examples

```
% set_parameter uniquify_naming_style %s#%d
```

upf_write_pg_to_netlist

Controls whether write_verilog adds power and ground nets to the netlist.

Usage

upf_write_pg_to_netlist <integer>

Arguments

- <integer>

  Specifies whether to enable write_verilog to add power and ground nets to the netlist. The default is 0. To enable power and ground writing, specify a non-zero integer.

Examples

Enable power and ground net export when writing the verilog netlist.

```
% set_parameter upf_write_pg_to_netlist 1
```

Related Topics

Oasys-RTL Parameters

write_verilog

get_parameter

set_parameter

reset_parameter

report_parameters

version

Displays the software version. This is a read only parameter.

Usage

version

Arguments

None.

Examples

```
% get_parameter version
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

vhdl_sort_files

Controls how VHDL files are read.

Usage

vhdl_sort_files [true | false]

Arguments

- true

  Specifies that read_vhdl does not directly read the given files, but stores them for later reading during synthesis. The synthesize command then sorts all the files given by all the read_vhdl commands into a valid reading order, and then reads them. Any VHDL syntax errors are reported at the start of synthesize. The advantage of this is that it is not necessary to specify VHDL files on the command line in the correct order. VHDL compilation is order-dependent, and for a large set of VHDL files it can be difficult for the user to determine a valid ordering. This is the default.

- false

  Specifies to behave like read_verilog. Each read_vhdl command reads the given VHDL files, reporting any errors.

Examples

```
% set_parameter vhdl_sort_files false
```

Related Topics

Oasys-RTL Parameters

read_verilog

read_vhdl

synthesize

get_parameter

set_parameter

reset_parameter

report_parameters

voltage_units_for_reports

Defines the voltage units for reports

Usage

voltage_units_for_reports {mV | V }

Arguments

- {mV | V }

  Specifies the voltage units for reports. The default value is “V”

Examples

```
set_parameter voltage_units_for_reports mV
```

write_implicit_wire_decls

Specifies that write_verilog writes out declarations for implicit wires.

Usage

write_implicit_wire_decls [false | true]

Arguments

- false

  Specifies that declarations for implicit wires are not written out by write_verilog. This is the default.

- true

  Specifies that write_verilog writes out declarations for implicit wires of the form n_1, n_2, n_3, and so on.

Examples

```
% write_implicit_wire_decls true
```

Related Topics

Oasys-RTL Parameters

write_verilog

get_parameter

set_parameter

reset_parameter

report_parameters

write_sdc_split_line

Specifies that if the SDC line is too long, the command is written on multiple lines.

Usage

write_sdc_split_line [false | true]

Arguments

- false

  Specifies that SDC commands are not placed on multiple lines. This is the default.

- true

  Specifies that if the SDC line is too long, the command is written on multiple lines.

Examples

```
% write_sdc_split_line true
```

Related Topics

Oasys-RTL Parameters

get_parameter

set_parameter

reset_parameter

report_parameters

Appendix B
UPF Command Support

The Oasys-RTL tool supports all UPF 1.0, standard UPF 2.0 (IEEE 1801-2009) and some UPF 2.1 (IEE 1801-2013) commands.

All UPF commands are parsed when read with the load_upf command, but some may be ignored.

Table B-1 lists the UPF commands and options and whether they are supported by the Oasys-RTL tool. The level of support is defined as follows:

- **yes —** The command is currently fully supported or the option is currently supported.
- **partial —** (command only) The command and some of its options are currently supported, but some of the options are parsed but ignored. Ignored options are marked by an indented “no”, while supported options are marked by an indented “yes”.
- **no —** The command or option is parsed, but it is ignored by the Oasys-RTL tool. Warning messages are issued if ignored commands or options are encountered in files with UPF commands.

Table B-1. UPF Command Support

Command	Option	Supported?
add_domain_elements	<domain_name> -elements <list>	no
add_port_state	<port_name> {-state {<name> [<nom> | <min> <nom> <max> | off]}}*	yes
add_power_state		partial
	<object_name>	yes
	[-supply |	yes
	-domain |	yes
	-group |	yes
	-model	no
	-instance	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
	{-state <state_name> {[-supply_expr {boolean_function [-logic_expr{boolean_function}]}] [-power_expr {power_expression} [-simstate <simstate>][-legal|-illegal]}	yes
	[-simstate <simstate>]	no
	[-legal|-illegal]	no
	[-complete]	yes
add_pst_state	<state_name> -pst <table_name> -state <supply_states>	yes
bind_checker	 <instance_name> -module <checker_name> -elements <list> [-ports {{<port_name> <net_name>}*}]	no
connect_logic_net		partial
	<net_name> -ports <list>	yes
	[-reconnect]	no
connect_supply_net		partial
	<net_name>	yes
	[-ports <list>]	yes
	[-pins <list>]	yes
	[-cells <list> | -domain <domain_name>]	yes
	[-rail_connection <rail_type> |	no
	[-pg_type <pg_type>]* | [-vct <vct_name>]	no
connect_supply_set		no
create_composite_domain		no
create_hdl2upf_vct	<vct_name> -hdl_type {[vhdl | vlog | SV] [<typename>]} -table {{<from_value> <to_value>}*}	no
create_logic_net	<net_name>	yes

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
create_logic_port	<port_name> [-direction <in|out|inout>]	yes
create_power_domain		partial
	[-define_func_type {supply_function pg_type_list}]*	no
	<domain_name>	yes
	[-elements <list>]	yes
	[-exclude_elements exclude_list]	no
	[-include_scope]	yes
	[-scope <instance_name>]	no
	[-supply]	yes
create_power_state_group	<group_name>	yes
create_power_switch	<switch_name> -domain <domain_name> -output_supply_port {<port_name> <supply_net_name>} {-input_supply_port {<port_name> <supply_net_name>}}* {-control_port {<port_name> <net_name>}}* {-on_state {<state_name> <input_supply_port> {<boolean_function>}}}* [-on_partial_state {state_name input_supply_port {boolean_function}}]* [-ack_port {<port_name> <net_name> [{boolean_function}]}]* [-ack_delay {<port_name> <delay>}]* [-off_state {<state_name> {<boolean_function>}}]* [-error_state {<state_name> {<boolean_function>}}]*	yes

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
create_pst	<table_name> -supplies <list>	yes
create_supply_net		partial
	<net_name>	yes
	-domain <domain_name>	yes
	[-reuse]	yes
	[-resolve [unresolved | one_hot | parallel]]	no
create_supply_port	<port_name> [-domain <domain_name>] [-direction [in | out]]	yes
create_supply_set		partial
	<supply_set_name>	yes
	[-function {'<function_name><net_name>'}']	yes
	[-reference_gnd <supply_net_name>]	no
	[-update]	yes
create_upf2hdl	<vct_name> -hdl_type {[vhdl | vlog | SV] [typename]} -table {{<from_value> <to_value>}*}	no
create_voltage_area (MV)	-name <voltage_area_name> -coordinate <float_list> [-guard_band_x <value>] [-guard_band_y <value>] [<cell_list>]	no
describe_state_transition		no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
find_objects		partial
	<scope>	yes
	-pattern <search_pattern>	yes
	[-object_type <inst | port>]	yes
	[-direction <in | out | inout>]	yes
	[-transitive [<TRUE | FALSE>]]	yes
	[-ignore_case]	yes
	[-non-leaf]	yes
	[-leaf_only]	yes
	[-object_type <model | net | process>]	no
	[-regexp | -exact]	no
get_supply_net	<net_name> [-domain <domain_name>] [-scope <scope_name>]	no
load_simstate_behavior		no
load_upf		yes
	<upf_file_name>	yes
	[-scope <instance_name>]	yes
	[-version <string>]	yes
load_upf_protected		no
map_isolation_cell		partial
	<isolation_name>	yes
	-domain <domain_name>	yes
	[-elements <list>]	yes
	[-lib_cells <list>]	yes
	[-lib_cell_type <lib_cell_type>]	no
	[-lib_model_name <lib_model_name> {-port {port_name net_name}}*]	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
map_level_shifter_cell		partial
	<level_shifter_name>	yes
	-domain <domain_name>	yes
	-lib_cells <list>	yes
	[-elements <list>]	yes
	-lib_cell_type	no
	-lib_model_name	no
map_power_switch		no
map_retention_cell		partial
	<retention_name>	yes
	-domain <domain_name>	yes
	[-elements <list>]	yes
	[-exclude_elements exclude_list]	no
	[-lib_cells <list>]	yes
	[-lib_cell_type <lib_cell_type>]	no
	[-lib_model_name <lib_cell_name> {-port <port_name> <net_name>}*]	no
merge_power_domains	<new_domain_name> -power_domains <list> [-scope <instance_name>] [-all_equivalent]	no
name_format		partial
	[- implicit_logic_prefix string]	no
	[- implicit_logic_suffix string]	no
	[- implicit_supply_suffix string]	no
	[-isolation_prefix <string>]	yes
	[-isolation_suffix <string>]	yes
	[-level_shift_prefix <string>]	yes
	[-level_shift_suffix <string>]	yes

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
save_upf	<upf_file_name> [-scope <instance_name>] [-version <string>] [-hierarchical]	yes
set_design_top	<design_top_module>	yes
set_domain_supply_net	<domain_name> -primary_power_net <supply_net_name> -primary_ground_net <supply_net_name>	yes
set_isolation		partial
	[-applies_to [inputs | outputs | both]]	yes
	[-applies_to_clamp <0 | 1 | any | Z | latch | value>]	no
	[-applies_to_sink_off_clamp <0 | 1 | any | Z | latch | value>]	no
	[-applies_to_source_off_clamp <0 | 1 | any | Z | latch | value>]	no
	[-clamp_value [0 | 1 | latch | Z]]	yes
	[-diff_supply_only <TRUE | FALSE>]	no
	-domain <domain_name>	yes
	[-elements <list>]	yes
	[-force_isolation]	no
	[-instance {{instance_name port_name}*}]	no
	<isolation_name>	yes
	[-isolation_power_net <net_name> | -isolation_ground_net <net_name> | -no_isolation]	yes
	[-sink_off_clamp <0 | 1 | any | Z | latch | value> [simstate_list]]	no
	[-source_off_clamp <0 | 1 | any | Z | latch | value> [simstate_list]]	no
	[-transitive <TRUE | FALSE>]	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
set_isolation_control	<isolation_name> -domain <domain_name> -isolation_signal <signal_name> [-isolation_sense [high | low]] [-location [fanout | fanin |automatic]]	yes
set_level_shifter		partial
	<level_shifter_name>	yes
	-domain <domain_name>	yes
	[-elements <list>]	yes
	[-force_shift]	no
	[-input_supply_set supply_set_name]	no
	[-output_supply_set supply_set_name]	no
	[-internal_supply_set supply_set_name]	no
	[-instance {{instance_name port_name}*}]	no
	[-transitive <TRUE | FALSE>]	no
	[-threshold <value>]	no
	[-applies_to [inputs | outputs | both]]	yes
	[-rule [low_to_high | high_to_low | both]]	yes
	[-location [fanout | fanin | automatic>]	yes
	[-no_shift]	yes
set_level_shifter_strategy (MV)	[-rule all | low_to_high | high_to_low] [-comment <String>]	no
set_level_shifter_threshold (MV)	[-voltage <absolute_volt_diff>] [-percent <percent_volt_diff>] [-comment <String>]	no
set_partial_on_translation		no
set_pin_related_supply	<library_cell> -pins <list> -related_power_pin <supply_pin> -related_ground_pin <supply_pin>	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
set_port_attributes		partial
	[-model <name>]	no
	[-elements <element_list>]	no
	[-exclude_elements <element_exclude_list>]	no
	[-ports <port_list>]	yes
	[-exclude_ports <port_exclude_list>]	no
	[-applies_to {inputs | outputs | both}]	yes
	[-attribute <name> <value>]...	no
	[-clamp_value {0 | 1 | any | Z | latch | <value>}]	yes
	[-sink_off_clamp {0 | 1 | any | Z | latch | <value>}]	no
	[-source_off_clamp {0 | 1 | any | Z | latch | <value>}]	no
	[-driver_supply <supply_set_ref>]	yes
	[-receiver_supply <supply_set_ref>]	yes
	[-pg_type <pg_type_value>]	no
	[-related_power_port <supply_port_name>]	yes
	[-related_ground_port <supply_port_name>]	yes
	[-related_bias_ports <supply_port_name_list>]	no
	[-feedthrough]	no
	[-unconnected]	no
	[-is_analog]	yes
	[{-domains <domain_list> [-applies_to	no
	{inputs | outputs | both}]}]	no
	[{-exclude_domains <domain_list>	no
	[-applies_to {inputs | outputs | both}]}]	no
	[-repeater_supply <supply_set_ref>]	no
	[-transitive {TRUE | FALSE}]	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
set_power_switch	<switch_name> -output_supply_port {port_name <supply_net_name>} {-input_supply_port {<port_name> <supply_net_name>}}* {-control_port {<port_name> <net_name>}}* {-on_state {<state_name> <input_supply_port> {boolean_function}}}* [-on_partial_state {<state_name> input_supply_port {<boolean_function>}}]* [-off_state {<state_name> {<boolean_function>}}]* [-error_state {<state_name> {<boolean_function>}}]*	no
set_related_supply_net		yes
set_retention		partial
	<retention_name>	yes
	-domain <domain_name>	yes
	-retention_power_net <net_name>	yes
	-retention_ground_net <net_name>	yes
	[-elements <list>]	yes
set_retention (cont.)	[-no_retention]	no
	[-use_retention_as_primary]	no
	[-parameters <>]	no
	[-instance {{instance_name [signal_name]}*}]	no
	[-transitive <TRUE | FALSE>]	no

Table B-1. UPF Command Support (cont.)

Command	Option	Supported?
set_retention_control		partial
	<retention_name>	yes
	-domain <domain_name>	yes
	-save_signal {{<net_name> <high | low | posedge | negedge>}}	yes
	-restore_signal {{<net_name> <high | low | posedge | negedge>}}	yes
	[-assert_r_mutex {{<net_name> <high | low | posedge | negedge>}}]*	no
	[-assert_s_mutex {{<net_name> <high | low | posedge | negedge>}}]*	no
	[-assert_rs_mutex {{<net_name> <high | low | posedge | negedge>}}]*	no
set_retention_elements		no
set_scope	<instance>	yes
set_simstate_behavior		no
upf_version		yes
	[string]	yes
use_interface_cell		partial
	<interface_implementation_name>	yes
	-strategy <list_of_iso_level_shifter_strategies>	yes
	-domain <domain_name>	yes
	-lib_cells <lib_cell_list>	yes
	[map {{port net_ref}*}]	no
	[-elements <element_list>]	yes
	[-exclude_elements <exclude_list>]	no
	[-applies_to_clamp <0 | 1 | any | Z | latch | value>]	no
	[-update_any <0 | 1 | known | Z | latch | value>]	no
	[-force_function]	no
	[-inverter_supply_set <list>]	no

UPF File Example

This section includes an example UPF file.

```
# top level domain.
create_power_domain PD_TOP -include_scope
create_supply_port vdd_top -domain PD_TOP
create_supply_net vdd_top -domain PD_TOP
connect_supply_net vdd_top -ports vdd_top
create_supply_port vss -domain PD_TOP
create_supply_net vss -domain PD_TOP
connect_supply_net vss -ports vss
set_domain_supply_net PD_TOP \
                      -primary_power_net vdd_top \
                      -primary_ground_net vss

# for U_0_0_0
create_power_domain PD_U000 -elements {U_0_0_0}
create_supply_port vdd_U000 -domain PD_U000
create_supply_net vdd_U000 -domain PD_U000
connect_supply_net vdd_U000 -ports vdd_U000
create_supply_net vss -domain PD_U000 -reuse
set_domain_supply_net PD_U000 \
                      -primary_power_net vdd_U000 \
                      -primary_ground_net vss

# for U_0_0_1
create_power_domain PD_U001 -elements {U_0_0_1}
create_supply_port vdd_U001 -domain PD_U001
create_supply_net vdd_U001 -domain PD_U001
connect_supply_net vdd_U001 -ports vdd_U001
create_supply_net vss -domain PD_U001 -reuse
set_domain_supply_net PD_U001 \
                      -primary_power_net vdd_U001 \
                      -primary_ground_net vss

# power state table
add_port_state vdd_top -state { von 0.810 }
add_port_state vss -state { gnd 0 }
add_port_state vdd_U000 -state { von1 0.810}
add_port_state vdd_U000 -state { von2 0.910}
add_port_state vdd_U000 -state { von3 0.710}
add_port_state vdd_U000 -state { off off}
add_port_state vdd_U001 -state { von1 0.810}
add_port_state vdd_U001 -state { von2 0.910}
add_port_state vdd_U001 -state { von3 0.710}
add_port_state vdd_U001 -state { off off}
create_pst pt -supplies {vdd_top vss vdd_U000 vdd_U001}
add_pst_state state1 -pst pt -state  { von gnd von1 von1 }
add_pst_state state2 -pst pt -state  { von gnd off off }
add_pst_state state3 -pst pt -state  { von gnd von2 von2 }
add_pst_state state4 -pst pt -state  { von gnd von3 von2 }
add_pst_state state5 -pst pt -state  { von gnd von2 von3 }

# isolations
set_isolation  iso_low \
               -domain               PD_U000 \
               -isolation_power_net  vdd_top \
               -isolation_ground_net vss \
               -clamp_value          0
set_isolation_control iso_low \
               -domain           PD_U000 \
```

```
                -isolation_signal RESET \
                -isolation_sense  high \
                -location         self
    set_isolation  iso_high \
                -domain               PD_U001 \
                -isolation_power_net  vdd_top \
                -isolation_ground_net vss \
                -c  lamp_value          1
    set_isolation_control iso_high \
                -domain           PD_U001 \
                -isolation_signal RESET \
                -isolation_sense  high \
                -location         parent
    # retention
    set_retention ret_U000 \
                -domain PD_U000 \
                -retention_power_net vdd_top \
                -retention_ground_net vss \
                -elements "U_0_0_0/Term*"
    set_retention_control ret_U000 \
                -domain         PD_U000 \
                -save_signal    {RESET low} \
                -restore_signal {RESET high}
```

Appendix C
SDC Command Support

Oasys-RTL supports a subset of Synopsys Design Constraint (SDC) commands. Oasys parses all SDC commands, including unsupported ones, checks for correct syntax, and stores them in the database. The write_sdc command writes out all SDC commands. Oasys also supports abbreviated SDC commands.

Unsupported SDC Commands

Some options and commands may be unsupported and/or ignored.

Table C-1 lists the SDC commands and options that are not supported by the Oasys-RTL tool.

Table C-1. Unsupported SDC Commands

Command	Option
remove_disable_clock_gating_check	<object_list>
set_clock_gating_check	[-setup <setup_value>] [-hold <hold_value>] [-rise] [-fall] [-high] [-low] [<object_list >]
set_data_check	[-from <from_object>] [-to <to_object>] [-rise_from <from_object>] [-fall_from <from_object>] [-rise_to <to_object>] [-fall_to <to_object>] [-setup] [-hold] [-clock <clock_object>] <value>

Table C-1. Unsupported SDC Commands (cont.)

Command	Option
set_disable_clock_gating_check	<object list>
set_drive	[-rise] [-fall] [-min] [-max] < resistance> < port_list>
set_fanout_load	<value> <port_list>
set_hierarchy_separator	<separator>
set_ideal_latency	[-rise] [-fall] [-min] [-max] <value> <object_list>
set_ideal_net	<object_list>
set_ideal_transition	[-rise] [-fall] [-min] [-max] <value> < object_list>
set_logic_dc	<port_list>
set_max_area	<area_value>
set_max_capacitance	<value> < object_list>
set_max_dynamic_power	-unit <unit> <power_value>
set_max_fanout	<fanout_number> < object_list>

Table C-1. Unsupported SDC Commands (cont.)

Command	Option
set_max_leakage_power	-unit <unit> <power_value>
set_max_time_borrow	<delay_value> < object_list>
set_max_transition	[-rise] [-fall] [-clock_path] [-data_path] <transition_value > < object_list>
set_min_capacitance	<value> <object_list>
set_min_delay	[-rise] [-fall] [-from <from_list >] [-to <to_list >] [-through <through_list>] [-rise_from <from_object>] [-fall_from <from_object>] [-rise_through <through_object>] [-fall_through <through_object>] [-rise_to <to_object>] [-fall_to <to_object>] < delay_value>
set_min_porosity	<porosity_value> <object_list>
set_port_fanout_number	<value > < object_list>
set_resistance	[-min] [-max] < value> < net_list>

Appendix D
SystemVerilog Command Support

A selection of SystemVerilog features are supported in the Oasys-RTL tool.

All SystemVerilog syntax is parsed when read in with the read_verilog -sv command, but some features may be ignored. Error messages or warnings may be generated.

SystemVerilog Messages

Oasys-RTL messages indicate which SystemVerilog constructs are supported.

The following are examples of SystemVerilog error and warning messages:

- A nested interface declaration is not supported:

  error: nested interface not supported ((test.v:27)) [VLOG-101]

- An assignment in an initial block is ignored with a warning:

  warning: ignoring non-constant assignment in initial block ((test.v:42)[9]) [VLOG-527]

  The reason for a warning in this case is that an assignment in an initial block could result in a simulation/synthesis mismatch, since the simulator honors it.

- Features like SystemVerilog assertions are ignored without a message.

  An assertion does not result in a simulation/synthesis mismatch, so you do not see hundreds of warnings for a feature that you know are not synthesizable.

- Unsupported defparam (not all forms of defparams are supported):

  error: defparam applied to parameter in current module not supported ((test.v:33)[26]) [VLOG-101]

SystemVerilog Features Support

The Language Reference Manual (LRM) version used is SystemVerilog 1800-2009.

Table D-1 on page 734 lists the support for SystemVerilog features. The level of support is defined as follows:

- **yes** — Currently supported.
- **no** — Not supported. The tool parses the input file and issues an error message.
- **partial** — Partially supported. Exceptions are listed in the table. Using any of these unsupported features (exceptions) causes an error message to be output.
- **ignore** — Command is ignored. The input file is parsed and the command is silently ignored. No error message is generated, but sometimes a warning message is issued.

Table D-1. SystemVerilog Feature Support

LRM Section	Supported?	Details
5	yes	Lexical
6		Data types
6.6.2	yes	uwire
6.8	yes	Variable Declarations
6.11	yes	Integer data types
6.11.2	yes	2-state/4-state No distinction made between 2-state and 4-state types.
6.12	no	Real numbers
6.13	yes	Void data type (for functions)
6.14	no	Chandle
6.15	no	Class
6.16	no	String data type String literals are supported, but not the string data type, which is intended to be implemented as a run-time resizable buffer.
6.17	no	event
6.18	partial	User-defined types (typedef) Fully supported except for forward typedefs
6.19		Enum types
6.19.1	yes	Enum type with typedef
6.19.2	yes	Enum type range
6.19.3	partial	Enum type checking (Not all cases are checked)

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
6.19.4	yes	Enum types in numerical expressions
6.19.5	partial	Enum type methods Enum methods are supported only for an enum type with a contiguous ascending range. .name() method not supported.
6.20		Constants
6.20.1	yes	Parameter declarations
6.20.2	yes	Value parameters
6.20.2.1	no	"$" as a parameter value
6.20.3	yes	Type parameters
6.20.4	yes	Localparams
6.20.5	no	Specparams
6.20.6	yes	Const declaration
6.21	partial	Scope and lifetime All subprogram variables are treated as automatic
6.22	partial	Type compatibility Not all type compatibility checks are enforced
6.23	yes	Type operator
6.24	yes	Cast operator
6.24.2	no	$cast dynamic casting
6.24.3	yes	Bit-stream casting
7.2	yes	Structures
7.2.1	yes	Packed structures
7.2.2	yes	Assigning to structures
7.3		Union
7.3.1	yes	Packed unions
7.3.2	no	Tagged unions
7.4	yes	Packed and unpacked arrays
7.4.1	yes	Packed arrays
7.4.2	yes	Unpacked arrays

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
7.4.3	partial	Operations on arrays All operations implemented except for writing to a variable slice of a multidimensional array
7.4.4	yes	Memories
7.4.5	yes	Multidimensional arrays
7.4.6	partial	Indexing and slicing of arrays All operations implemented except for writing to a variable slice of a multidimensional array
7.5	no	Dynamic arrays
7.6	partial	Array assignments All assignments implemented except for writing to a variable slice of a multidimensional array
7.7	yes	Arrays as arguments to subroutines
7.8	no	Associative arrays
7.9	no	Associative array methods
7.10	no	Queues
7.11	yes	Array querying functions
7.12	no	Array manipulation methods
8	no	Classes
9		Processes
9.2.1	ignore	Initial procedures
9.2.2		Always procedures
9.2.2.1	yes	General purpose always
9.2.2.2	yes	always_comb
9.2.2.3	yes	always_latch
9.2.2.4	yes	always_ff
9.2.3	ignore	Final procedure
9.3		Block statements
9.3.1	yes	Sequential block
9.3.2	no	Parallel blocks

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
9.3.4	yes	Block names
9.3.5	yes	Statement labels
9.4		Procedural timing controls
9.4.1	ignore	Delay control
9.4.2	partial	Event control Support one event control at top of always block
9.4.2.1	yes	Event or operator
9.4.2.2	yes	Implicit event_expression $(*)
9.4.2.3	yes	Conditional event controls: always @(posedge clk iff ena)
9.4.2.4	no	Sequence events
9.4.3	no	Level-sensitive event control (wait statement)
9.4.4	no	Level-sensitive sequence control (triggered)
9.4.5	no	Intra-assignment timing controls
9.6.1	no	Wait statement
9.6.2	partial	Disable statement Disable supported only for enclosing subprograms or named procedural blocks
9.6.3	no	Disable fork
9.7	no	Fine-grain process control
10		Assignment statements
10.3	yes	Continuous assignments
10.3.1	yes	Net declaration assignment
10.3.2	yes	Continuous assignment statement
10.3.3	ignore	Continuous assignment delays
10.3.4	ignore	Continuous assignment strengths
10.4	yes	Procedural assignments
10.4.1	yes	Blocking procedural assignments
10.4.2	yes	Non blocking procedural assignments
10.5	ignore	Variable declaration assignment

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
10.6	partial	Procedural continuous assignments Procedural assign is implemented as blocking assignment (all others are unsupported)
10.7	yes	Assignment extension and truncation
10.8	yes	Assignment-like contexts
10.9	yes	Assignment patterns
10.9.1	yes	Assignment patterns
10.9.2	yes	Assignment patterns
10.10	yes	Unpacked array concatenation
10.11	no	Net aliasing (alias x = y;)
11		Operators
		Real operands are supported only in constant expressions
11.3.5	yes	Operator short-circuiting
11.3.6	yes	Assignment within an expression
11.4.1	yes	Assignment operators
11.4.2	yes	Increment/decrement operators
11.4.3	yes	Arithmetic operators A**B is supported if A is a power of 2 or B is a constant
11.4.5	partial	Equality operators Non-constant case equality is implemented as regular equality
11.4.6	partial	Wildcard equality (==?, !=?) Non-constant wildcard equality is implemented as regular equality
11.4.7	partial	Logical operators Implication (->) and equivalence (<->) are not yet supported
11.4.8	yes	Bitwise operators
11.4.9	yes	Reduction operators
11.4.10	yes	Shift operators
11.4.11	partial	Conditional operator (?:) &&& and 'matches' not implemented
11.4.12	yes	Concatenation operators

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
11.4.12.1	yes	Replication operator
11.4.12.2	partial	String concatenation Supported for string literal operands
11.4.13	yes	Set membership operator (inside)
11.4.14	yes	Streaming operators (<<, >>)
11.5.1	yes	Vector bit-select and part-select addressing
11.5.2	partial	Array and memory addressing Except for assignment to a variable slice of a multidimensional array
11.6	yes	Expression bit lengths
11.7	yes	Signed expressions
11.8	yes	Expression evaluation rules
11.9	no	Tagged union expressions
11.10	yes	String literal expressions
11.10.3	yes	Null string literal handling Null string "" is treated as NUL character
11.11	no	Operator overloading
11.12	ignore	min/typ/max expressions
		Middle (typ) expression is used
11.13	ignore	Let construct
12.4	yes	Conditional if-else statement
12.4.1	yes	if-else-if construct
12.4.2	partial	Unique & priority "unique" is treated as "parallel_case" "priority" is treated as "full_case" "unique0" is not supported
12.5	yes	case statement

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
12.5.3	partial	Unique & priority "unique" is treated as "parallel_case" "priority" is treated as "full_case" "unique0" is not supported
12.5.4	yes	Set membership case statement (case-inside)
12.6	no	Pattern matching condition statement
12.7		Loop statements
12.7.1	yes	For loo
12.7.2	yes	Repeat loop
12.7.3	yes	Foreach loop
12.7.4	yes	While loop
12.7.5	yes	Do-while loop
12.7.6	yes	Forever loop
12.8	yes	Jump statements: break, continue, return
13.3	yes	Tasks
13.3.1	partial	Static & automatic tasks All tasks are treated as automatic
13.4	yes	Functions
13.4.1	yes	Void functions
13.4.2	partial	Static & automatic functions All functions are treated as automatic
13.4.3	yes	Constant functions
13.5		Subroutine calls and argument passing
13.5.1	yes	Pass by value
13.5.2	yes	Pass by reference "ref" port treated as inout
13.5.3	yes	Default argument values
13.5.4	yes	Argument binding by name
13.5.5	yes	Optional argument list
13.6	no	Import and export functions

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
14	ignore	Clocking blocks
15	ignore	Interprocess synchronization and communication
16	ignore	Assertions
17	ignore	Checkers
18	ignore	Constrained random value generation
19	ignore	Functional coverage
20	partial	Utility system tasks and functions
		All ignored except for: 20.6.2 and 20.7 (see below)
20.6.2	yes	$bits
20.7	yes	Array querying functions
21	ignore	I/O system tasks and system functions
22	yes	Compiler directives
23		Modules and Hierarchy
23.2		Module definitions
23.2.1		Module header definition
23.2.2		Port declarations
23.2.2.1	partial	Non-ANSI style port declarations All non-ANSI style ports supported except for non-ANSI style interface port
23.2.2.2	yes	ANSI style list of port declarations
23.2.2.4	yes	Default port values
23.2.3	yes	Parameterized modules
23.3		Module instances
23.3.1	yes	$root
23.3.2	yes	Module instantiation syntax
23.3.2.1	yes	Connection by ordered list
23.3.2.2	yes	Connection by name
23.3.2.3	yes	Connection with implicit named ports (.clk, .rst)
23.3.2.4	partial	Connection with wildcard port connection (.*)

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
23.3.3.6	yes	uwire
23.4	no	Nested modules
23.5	yes	Extern modules
23.6	partial	Hierarchical names Hierarchical names supported only for in-module references and for references into interface ports and interface instances.
23.7	partial	Member selects and hierarchical names Member selects are supported. For hierarchical names, see above
23.7.1	partial	Names with package or class scope resolution Only package scope supported, not class scope
23.8	partial	Upwards name referencing Limited to in-module references
23.8.1	partial	Task and function name resolution Limited to tasks/functions in current module in a package, in $unit, or in an interface instance or port
23.10		Overriding module parameters
23.10.1	partial	Defparam statement Defparam may apply only to descendant instance
23.10.2		Module instance parameter value assignment
23.10.2.1	yes	Parameter value assignment by ordered list
23.10.2.2	yes	Parameter value assignment by name
23.11	no	Binding auxiliary code (bind)
24	ignore	Programs
25		Interfaces (see details on interfaces below)
25.3.2	yes	Interface using named bundle
25.3.3	yes	Interface using generic bundle
25.4	yes	Ports in interfaces
25.5	yes	Modports
25.5.4	yes	Modport expressions

Table D-1. SystemVerilog Feature Support (cont.)

LRM Section	Supported?	Details
25.5.5	ignore	Clocking blocks and modports
25.6	ignore	Interfaces and specify blocks
25.7	yes	Tasks and functions in interfaces
25.7.3	yes	Exporting tasks and functions
25.8	yes	Parameterized interfaces
25.9	no	Virtual interfaces
25.10	partial	Access to interface objects
26	yes	Packages
27	yes	Generate constructs
28	partial	Gate-level and switch-level modeling Supported gates: and nand or nor xor xnor buf not bufif0 bufif1 notif0 notif1 tran rtran
29	ignore	User defined primitives
30	ignore	Specify blocks
31	ignore	Timing checks
32	n/a	Back annotation using SDF
33	ignore	Configurations
34	ignore	Protected envelopes