

Verification Continuum™

FPGA Synthesis

Command Reference Manual

March 2021

SYNOPSYS®

<http://solvnet.synopsys.com>

Synopsys Confidential Information

Copyright Notice and Proprietary Information

© 2021 Synopsys, Inc. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Free and Open-Source Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at

<http://www.synopsys.com/Company/Pages/Trademarks.aspx>.

All other product or company names may be trademarks of their respective owners.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
690 East Middlefield Road
Mountain View, CA 94043
www.synopsys.com

March 2021

Synopsys Statement on Inclusivity and Diversity

Synopsys is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

Contents

Chapter 1: Overview of the Synthesis Commands

About Tcl Commands	16
About the GUI Commands	18

Chapter 2: Tcl Synthesis Commands

add_file	25
add_folder	30
add_vivado_ip	31
analyst	36
assign_to_region	39
check_fdc_query	40
command_history	44
constraint_file	45
create_fdc_template	46
create_region	48
cdpl_queue	49
define_link	52
design	53
dh_module_sources	63
dm_root	64
dump_metrics	65
encryptIP	68
encryptP1735	71
IEEE 1735 Encryption Use Models	74
export_project	81
generate_context	83
generate_instance_constraints	84
generate_timing_budgets	86
get_env	87
get_option	87
hdl_define	88
hdl_param	89

help	91
history	92
impl	93
ise2syn	95
job	96
launch_vivado	97
log_filter	98
log_report	100
message_override	101
open_design	103
open_file	104
partdata	105
post_par_resynthesis	107
process_bd_ip	107
program_terminate	109
program_version	110
project	111
project_data	120
project_file	121
project_folder	123
qsf2sdc	124
qsf2syn	125
query_available_metrics	126
query_metric	129
query_metric_details	131
recording	133
report_clocks	134
report_external_tool_versions	135
report_messages	136
report_message_summary	138
run_config	139
run_tcl	141
sch_sim_disp	142
sch_sim_load	143
sch_sim_watch	145
select	146
sdc2fdc	147
set_option	149
slice_primitive	195
sopc2syn	197
spy_glass	201
status_report	202

sub_impl	206
syn_connect	207
syn_create_err_net	208
synplify, synplify_pro, synplify_premier, synplify_premier_dp	210
sxml2pxml	213
vcs_launch	215
Tcl Command Categories	216

Chapter 3: Tcl Find, Expand, and Collection Commands

find	221
Tcl Find Syntax	222
Tcl Find Syntax Examples	225
find -filter	229
expand	237
Collection Commands	240
c_diff	241
c_info	242
c_intersect	242
c_list	243
c_print	243
c_symdiff	245
c_union	245
define_collection	246
define_scope_collection	247
get_prop	247
set	247
Query Commands	249
all_clocks	251
all_fanin	252
all_fanout	253
all_inputs	255
all_outputs	256
all_registers	257
dot2slash	258
get_cells	259
get_clocks	261
get_clock_source	263
get_flat_cells	264
get_flat_nets	266
get_flat_pins	268

get_nets	269
get_pins	271
get_ports	274
get_registers	276
object_list	277
report_timing	278
slash2dot	281
Synopsys Standard Collection Commands	282
add_to_collection	282
append_to_collection	284
copy_collection	285
foreach_in_collection	286
get_object_name	288
index_collection	288
remove_from_collection	290
sizeof_collection	291

Chapter 4: Constraint Commands

SCOPE Constraints Editor	294
SCOPE Tabs	295
Clocks	296
Generated Clocks	301
Collections	304
Inputs/Outputs	306
Registers	309
Delay Paths	310
Attributes	313
I/O Standards	314
Compile Points	316
TCL View	319
Industry I/O Standards	321
Industry I/O Standards	322
Delay Path Timing Exceptions	325
Multicycle Paths	325
False Paths	328
Specifying From, To, and Through Points	330
Timing Exceptions Object Types	330
From/To Points	330
Through Points	332
Product of Sums Interface	333

Clocks as From/To Points	336
Conflict Resolution for Timing Exceptions	338
 Timing Constraints	342
create_clock	343
create_generated_clock	345
reset_path	349
set_case_analysis	351
set_clock_groups	353
set_clock_latency	360
set_clock_route_delay	362
set_clock_uncertainty	363
set_datapathonly_delay	365
set_false_path	368
set_input_delay	371
set_max_delay	374
set_min_delay	377
set_multicycle_path	380
set_output_delay	384
set_reg_input_delay	387
set_reg_output_delay	388
Naming Rule Syntax Commands	388
 Design Constraints	391
define_compile_point	392
define_current_design	393
define io standard	394

Chapter 5: User Interface Commands

File Menu	396
New Command	397
Create Image Command	399
Build Project Command	401
Open Project Command	401
Edit Menu	402
Find Command (Text)	404
Find Command (In Project)	405
Find Command (HDL Analyst)	407
Find in Files Command	411
Replace Command	413
Goto Command	414
View Menu	415

Toolbar Command	418
View Sheets Command	419
View Log File Command	420
 Project Menu	423
Design Intent	425
Add Source File Command	426
Remove Implementation	428
Change File Command	429
Set VHDL Library Command	429
Add Implementation Command	430
Convert Vendor Constraints Command	430
Archive Project Command	433
Un-Archive Project Command	435
Copy Project Command	438
Hierarchical Project Options Command	441
Insert Subproject Command	442
 Implementation Options Command	444
Device Panel	445
Options Panel	447
Constraints Panel	450
Implementation Results Panel	453
Timing Report Panel	455
High Reliability Panel	456
VHDL Panel	458
Verilog Panel	462
Compiler Directives and Design Parameters	470
GCC Panel	480
GCC & Prototyping Tools Panel	481
Place and Route Panel	483
 Import Menu	485
Add Vivado IP Command	486
Import Xilinx EDK/ISE Project Command	489
Import Intel SOPC Project Command	492
Import Intel QSF Project Command	495
Import IP Package Command	496
Launch Vivado Command	497
 Run Menu	498
Run Tcl Script Command	502
Run Implementations Setup Command	502
Translate Vendor IO Command	504

Post Place & Route Resynthesis Command	504
Job Status Command	505
Identify Instrumentor Command	506
Launch Identify Debugger Command	507
Launch SYNCORE Command	508
Configure and Launch VCS Simulator Command	508
Analysis Menu	519
Timing Report View	520
Timing Report Generation Parameters	530
Generate Congestion Analysis	541
HDL Analyst Menu	542
HDL Analyst Menu: RTL and Technology View Submenus	543
HDL Analyst Menu: Hierarchical and Current Level Submenus	544
HDL Analyst Menu: Filtering and Flattening Commands	546
HDL Analyst Menu: Timing Commands	549
HDL Analyst Menu: Analysis Commands	549
HDL Analyst Menu: Selection Commands	553
HDL Analyst Menu: FSM Commands	553
HDL Analyst Menu: VCD Commands	554
Options Menu	568
Configure Parallel or Compile Point Process Command	570
Project View Options Command	573
Editor Options Command	580
Place and Route Environment Options Command	583
Configure 3rd Party Tools Options Command	584
Project Status Page Location	585
HDL Analyst Options Command	587
Standard HDL Analyst Options Command	589
Configure External Programs Command	597
Tech-Support Menu	598
Web Support Command	599
Web Menu	601
Help Menu	602
Preferred License Selection Command	603
Tip of the Day Command	604

Chapter 6: GUI Popup Menu Commands

Popup Menus	606
Watch Window Popup Menu	607

Tcl Window Popup Menu	607
Text Editor Popup Menu	607
Log File Popup Menu	608
FSM Viewer Popup Menu	610
 Project View Popup Menus	613
Project Management View Popup Folder Commands	619
Vendor Tool Invocation Popup Menu Command	620
File Options Popup Menu Command	622
Copy File Popup Menu Command	624
Change Implementation Popup Menu Commands	624
Show Compile Points Popup Menu Command	625
Project Options Popup Menu Command	626
Add P&R Implementation Popup Menu Command	626
Options for Place & Route Jobs Popup Menu Command	630
Create Subproject Popup Menu Commands	631
Design Block/Instance Properties Popup Menu Command	634
Insert Subproject Command	636
Subproject Parameter Sync	636
Insert & Link Subproject to Module Command	637
Allocate Timing and Resource Budgets	639
 RTL and Technology Views Popup Menus	641

Chapter 7: Design Planner Commands

Design Planner Menu (Generation 2)	656
Design Planner Preferences	656
 Design Planner UI Commands	658
File Menu: New Command	658
Edit Menu: Design Planner Commands	661
View Menu: Design Planner Commands	662
Implementation Options: Design Planner	665
Run Menu: Design Planner Commands	666
 Design Planner Tools Menu	667
Tools Menu Commands	667
Show Replicated Assignments Command	668
Preferences Command	668
 Design Planner Popup Menus	673
Design Planner View Commands	681
Physical Analyst User Interface	683

Physical Analyst UI Commands	686
Edit Menu: Physical Analyst Commands	686
View Menu: Physical Analyst Commands	686
HDL Analyst Menu: Physical Analyst Command	691
Options Menu: Physical Analyst Commands	691
Physical Analyst View Popup Menus	692
Global Commands	692
Physical Analyst Popup Commands with Instances Selected	695
Physical Analyst Popup Commands with Nets Selected	696
Physical Analyst Popup Menu Property Commands	696
Physical Analyst Find Command	699
Resolve Selection Dialog Box	703

Chapter 8: Unified Power Format Commands

associate_supply_set	707
corrupt_pd	708
create_power_domain	711
create_power_switch	713
create_supply_set	716
load_upf	718
map_retention_cell	719
name_format	720
set_domain_supply_net	721
set_equivalent	723
set_isolation	726
set_isolation_control	730
set_retention	733
set_retention_control	735
set_scope	736

Chapter 9: Netlist Editing Commands

connect_net	740
copy_view	742
create_cell	743
create_instance	744
create_net	746
create_port	747
create_process_hierarchy	748
define_current_view	749
disconnect_connector	750
disconnect_net	751
edit_netlist	753

get_net	754
insert_buffer	755
load_database	757
load_library	759
optimize_netlist	760
pop_feedthroughs	761
propagate_constants	762
remove_buffer	764
remove_instance	765
remove_net	766
remove_port	767
set_property	768
swap_instance	769
tie_net	770
tie_pin	772

CHAPTER 1

Overview of the Synthesis Commands

This document is part of a set that includes reference and procedural information for the Synopsys® FPGA tools. This document describes the commands available for the synthesis tools, which usually includes a graphical user interface (GUI) as well as command line access. Commands may vary with the capabilities of the synthesis tool.

The following sections provide an overview of the commands in the tool:

- [About Tcl Commands](#), on page 16
- [About the GUI Commands](#), on page 18

About Tcl Commands

Tcl (Tool Command Language) is a popular scripting language for controlling software applications. Synopsys has extended the Tcl command set with additional commands that you can use to run the Synopsys FPGA programs. These commands are not intended for use in controlling interactive debugging, but you can use them to run synthesis multiple times with alternate options to try different technologies, timing goals, or constraints on a design.

Tcl scripts are text files that have a .tcl file extension and contain a set of Tcl commands designed to complete a task or set of tasks. You can also run Tcl scripts through the Tcl window (see [Tcl Script Window, on page 58](#)).

The Synopsys FPGA Tcl commands are described here. For information on the standard Tcl commands, syntax, language, and conventions, refer to the Tcl online help (Help->TCL).

Tcl Conventions

Here is a list of conventions to respect when entering Tcl commands and/or creating Tcl scripts.

- Tcl is case sensitive.
- Comments begin with a hash mark or pound sign (#).
- Enclose all path names and filenames in double quotes (").
- Use a forward slash (/) as the separator between directory and path names (even on the Microsoft® Windows® operating system). For example:

```
designs/big_design/test.v
```

Tcl Commands

You can enter the Tcl commands directly in the Tcl window, or include them in Tcl scripts that you can run in batch mode. For more information about Tcl commands, see [Tcl Synthesis Commands, on page 23](#).

Tcl Scripts and Batch Mode

For procedures for creating Tcl scripts and using batch mode, see [Working with Tcl Scripts and Commands, on page 781](#) in the *User Guide*:

- [Running Batch Mode on a Project File, on page 774](#)
- [Running Batch Mode with a Tcl Script, on page 775](#)
- [Generating a Job Script, on page 782](#)
- [Creating a Tcl Synthesis Script, on page 783](#)
- [Using Tcl Variables to Try Different Clock Frequencies, on page 785](#)
- [Running Bottom-up Synthesis with a Script, on page 787](#)

About the GUI Commands

The GUI commands are accessed from the software graphical interface. Most commands open dialog boxes where you can specify parameters for the command.

The GUI provides a few ways to access commands:

- [Menus](#), on page 18
- [Context-sensitive Popup Menus](#), on page 19
- [Toolbars](#), on page 19
- [Keyboard Shortcuts](#), on page 19
- [Buttons and Options](#), on page 19
- [Tcl Commands](#), on page 16

Most commands have GUI and command line versions, so you can use either method to specify commands.

Menus

The set of commands on the pull-down menus in the menu bar varies depending on the view, design status, task to perform, and selected object(s). For example, the [File](#) menu commands in the Project view differ slightly from those in the RTL view. Menu commands that are not available for the current context are dimmed out. The menu bar in the Project view is shown below:



The individual menus, their commands, and the associated dialog boxes are described in the following sections:

- [File Menu](#), on page 396
- [Edit Menu](#), on page 402
- [View Menu](#), on page 415
- [Project Menu](#), on page 423
- [Import Menu](#), on page 485

- [Run Menu](#), on page 498
- [Analysis Menu](#), on page 519
- [HDL Analyst Menu](#), on page 542
- [Options Menu](#), on page 568
- [Tech-Support Menu](#), on page 598
- [Web Menu](#), on page 601
- [Help Menu](#), on page 602

Context-sensitive Popup Menus

Popup menus, available by right-clicking, offer access to commonly used commands that are specific to the current context. See [Popup Menus, on page 606](#), [Project View Popup Menus, on page 613](#), and [RTL and Technology Views Popup Menus, on page 641](#) for information on individual popup menus.

Toolbars

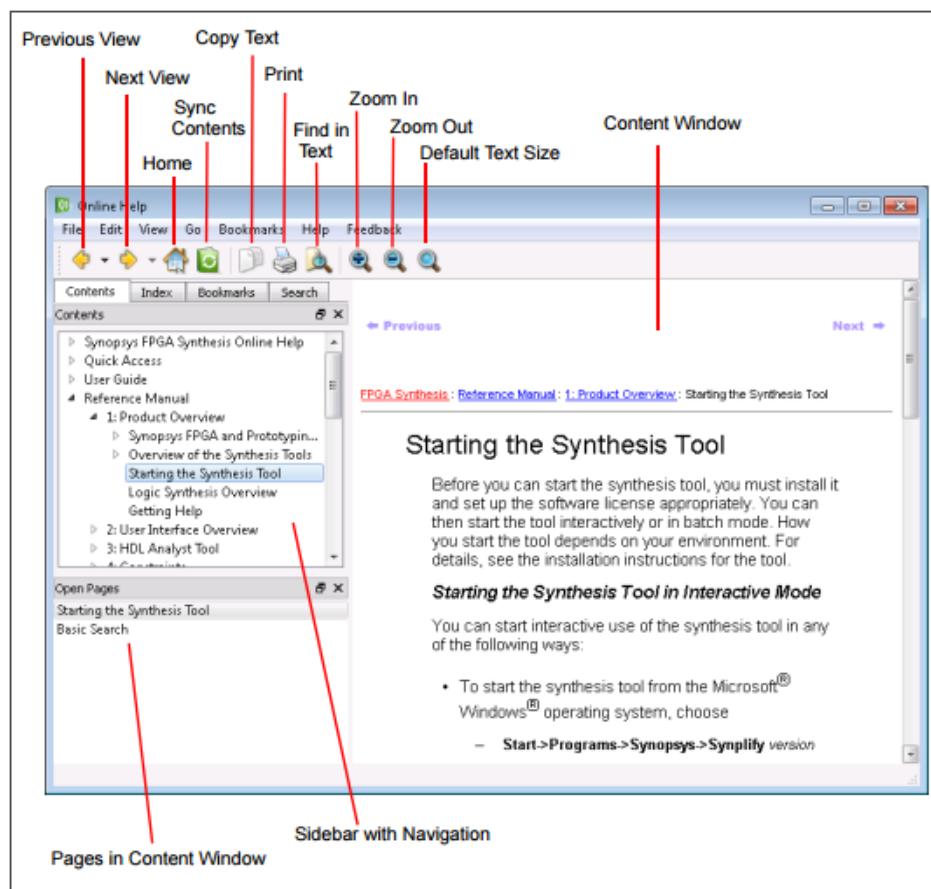
Toolbars contain icons associated with commonly used commands. For more information about toolbars, see [Toolbars, on page 76](#).

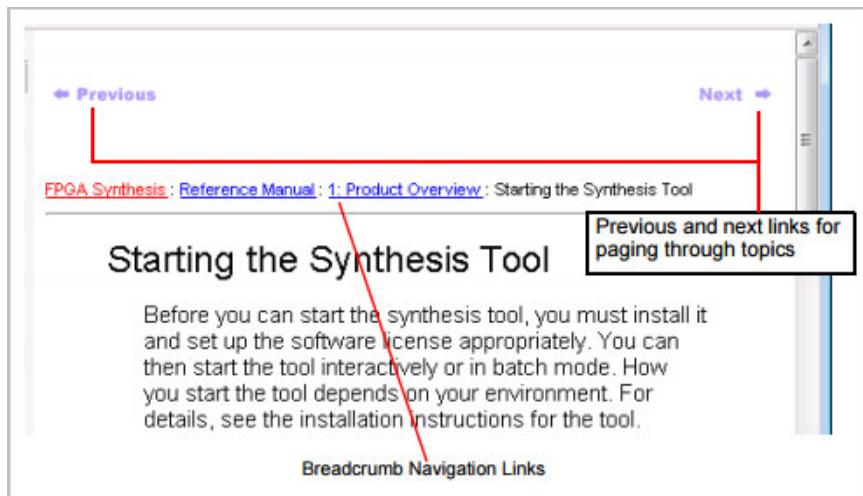
Keyboard Shortcuts

Keyboard shortcuts are available for commonly used commands. The shortcut appears next to the command in the menu. See [Keyboard Shortcuts, on page 84](#) for details.

Buttons and Options

The Project view has buttons for quick access to commonly used commands and options. See [Buttons and Options, on page 92](#) for details.





CHAPTER 2

Tcl Synthesis Commands

This chapter describes supported Tcl synthesis commands. The commands are listed in alphabetical order.

add_file	add_folder	add_vivado_ip
analyst	assign_to_region	check_fdc_query
command_history	constraint_file	create_fdc_template
create_region	cdpl_queue	define_link
design	dh_module_sources	dm_root
dump_metrics	encryptIP	encryptP1735
export_project	generate_context	generate_instance_constraints
generate_timing_budgets	get_env	get_option
help	hdl_define	hdl_param
history	impl	ise2syn
job	launch_vivado	log_filter
log_report	message_override	open_design
open_file	partdata	post_par_resynthesis
process_bd_ip	program_terminate	program_version
project	project_data	project_file
project_folder	qsf2sdc	qsf2syn

query_available_metrics	query_metric	query_metric_details
recording	report_clocks	report_external_tool_versions
report_messages	report_message_summary	run_config
run_tcl	sch_sim_disp	sch_sim_load
sch_sim_watch	sdc2fdc	select
set_option	slice_primitive	sopc2syn
spy_glass	status_report	sub_impl
syn_connect	syn_create_err_net	synplify
synplify_pro	synplify_premier	synplify_premier_dp
sxml2pxml	vcs_launch	

See also:

- For specific categories of synthesis commands (for example, log file or high reliability commands), see [Tcl Command Categories](#), on page 216.
- For a description of the find, expand, and collection commands, see [Tcl Find, Expand, and Collection Commands](#), on page 219.
- For the TCL timing and design constraints syntax and their descriptions in SCOPE, see [Constraint Commands](#), on page 293.
- For the IEEE Unified Power Format (UPF) commands syntax, see [Unified Power Format Commands](#), on page 705.
- For the netlist editing commands syntax, see [Netlist Editing Commands](#), on page 737.

add_file

The `add_file` command adds one or more files to a project.

Syntax

```
add_file [-filetype] fileName [ fileName [ ... ] ]
add_file -verilog [-lib fileName [ fileName [ ... ]]] [-folder folderName]
add_file -vhdl [-lib libName[ libName] ] fileName [ fileName [ ... ] ] [-folder folderName]
add_file -include fileName [ fileName [ ... ] ]
add_file [-filetype] -job_owner par | simulation [ fileName [ ... ] ]
add_file -structver [fileName [ ... ] ]
add_file -tooltag tooltagName -toolargs [toolArguments] fileName
add_file -vlog_std standard fileName [ fileName [ ... ] ]
add_file -clearbox_verilog fileName [ fileName [ ... ] ]
add_file -clearbox_vhdl fileName [ fileName [ ... ] ]
add_file -ucf fileName [ fileName [ ... ] ]
```

-filetype	Specifies the type of file being added to the project (files are placed in folders according to their file types; including this argument overrides automatic filename-extension placement). See Filename Extensions , on page 27 for a list of the recognized file types.
fileName	Specifies the name of the file being added to the project. Files are added to the individual project folders according to their filename extensions (View Project Files in Folders must be set in the Project View Options dialog box). You can add multiple files by separating individual filenames with a space, and you can specify different file types (extensions) within the same command.
-verilog or -vhdl	Adds HDL files with non-standard extensions to the Verilog or VHDL directory, so that they can be compiled with the project. For example, the following command adds the file alu.v.new to the project's verilog directory: <pre>% add_file -verilog /designs/megachip/alu.v.new</pre> If you do not specify -verilog, the file is added to the Other directory (new is not a recognized Verilog extension), and the file would not be compiled with the files in the Verilog directory.

[-lib <i>libName</i>]	<p>Specifies the library associated with Verilog or VHDL files. The default library is <i>work</i>. The <i>-lib</i> option sets the VHDL library to <i>libName</i>.</p>
	<p>Note: You can also specify multiple libraries for Verilog or VHDL files.</p>
	<p>Verilog Example:</p>
	<pre>add_file -verilog -lib top -vlog_std sysv "top.v"</pre>
	<p>VHDL Example:</p>
	<pre>add_file -vhdl -lib {mylib,work} "ff.vhd"</pre>
	<p>Both the logical and physical libraries must be specified in the Project file (if you only specify the logical library associated with the Verilog or VHDL files, the compiler treats the module as a black box).</p>
[-folder <i>folderName</i>]	<p>Creates logical folders with custom files in various hierarchy groupings within your Project view. For example:</p> <pre>add_file -verilog -folder memory "ram_1.v" add_file -verilog -folder memory "C:/examples/verilog/common_rtl/memory/ram_1.v"</pre>
-clearbox_verilog	<p><i>Intel FPGA</i></p>
-clearbox_vhdl	<p>Designates the specified file as a Clearbox model file and copies it to the <i>implementationName</i>/par_1 directory that is created after synthesis. When you run Quartus from the synthesis interface with the Run P&R option, it requires that the Clearbox files be in this directory.</p>
-include	<p>Indicates that the specified file is to be added to the project as an include file (include files are added to the <i>Include</i> directory regardless of their extension). Include files are not passed to the compiler, but are assumed to be referenced from within the HDL source code. Adding an include file to a project, although not required, allows it to be accessed in the user interface where it can be viewed, edited, or cross-probed.</p>
-job_owner	<p>Allows you to determine how files are used; you can specify these options from the File Options dialog box. For example, you can automatically decide to pass files to the back-end place-and-route tool (Use for Place and Route) or use them for test benches containing HDL constructs for simulation (Use for Simulation only).</p>

-structver <i>fileName</i>	Adds structural Verilog files as input for your design project. The software performs fast compilation of the structural Verilog files, providing runtime improvements for the design. For example: <code>add_file -structver <i>fileName</i>.vm</code> For more information, see Using the Structural Verilog Flow, on page 88 .
-tooltag <i>tooltagName</i>	Creates a tool tag name for the application tool you want to invoke from within the Synopsys FPGA synthesis tools. For example: <code>add_file -tooltag {Coregen} "ram.v"</code>
-toolargs <i>tool</i>	Specifies any argument options to use with the application tool you want to invoke from within the Synopsys FPGA synthesis tools. For example: <code>add_file -tooltag {EDK} -toolargs {\$SynCode} "ram.v"</code>
-vlog_std <i>standard</i>	Overrides the global Verilog standard for an individual file. The accepted values for <i>standard</i> are v95 (Verilog 95), v2001 (Verilog 2001), and sysv (SystemVerilog). The file (<i>fileName</i>) is added to the Verilog folder in the project; the specified standard is listed after the filename in the project view and is enclosed in angle brackets (for example, commchip.v <sysv>). Note that when you add a SystemVerilog file (a file with an sv extension) to a project, the add_file entry in the project file includes the -vlog_std <i>standard</i> string. The default standard for new projects is SystemVerilog. For Verilog 2005 extensions, use sysv (SystemVerilog).

Filename Extensions

Files with the following extensions are automatically added to their corresponding project directories; files with any other extension are added to the Other directory. The **-filetype** argument overrides automatic filename extension placement.

Extension	-Filetype	Project Folder
.adc	-analysis_constraint	Analysis Design Constraint
.cdc ¹	-compiler_constraint	Compiler Directive
.dpf ¹	-dpf	Design Plan File

Extension	-Filetype	Project Folder
.edf, .edn	-edif	EDIF
.fdc	-fpga_constraint/-constraint	Logic Constraints (FDC)
.ngc, .ngo	-xilinx	Xilinx File
.nrf ¹	-nrf	Netlist Restructure
.opt	-xflow	P&R Options
.prt	-partition	Partition File
.ptf ²	-ptf	Prototyping Constraints
.sdc	-constraint	Logic Constraints (SDC)
.sfp ¹	-floorplan	Design Plan
.sv ³	-verilog	Verilog
.tcl	-tcl	Tcl Script
.ucf	-xilinx	Xilinx Constraint File
.v	-verilog	Verilog
.vhd, .vhdl	-vhdl	VHDL
.vm, .vqm	-structver	Structural Verilog File
any	-include	Include
-	-clearbox_verilog	Clearbox Verilog
-	-clearbox_vhdl	Clearbox VHDL

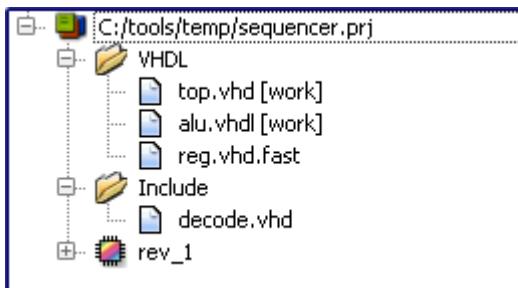
1. Available in Synplify Premier tools only.
2. Available in the Certify tool only
3. Use the sv format for SystemVerilog keyword support. Both Verilog and SystemVerilog formats are added to the Verilog folder.

Example: Add Files

Add a series of VHDL files to the VHDL directory and add an include file to the project:

```
% add_file /designs/sequencer/top.vhd  
% add_file /designs/sequencer/alu.vhdl  
% add_file -vhdl /designs/sequencer/reg.vhd.fast  
% add_file -include /designs/std/decode.vhd
```

The corresponding directory structure in the Project view is shown in the following figure:



Example: Designate Clearbox Files

Designate some files as Clearbox model files for Intel Quartus:

```
add_file -clearbox_verilog "dsp/my_dsp_syn.v"  
add_file -clearbox_vhdl "dsp/my_dsp_syn.vhd"
```

Example: File Options Designation

Designate some IP core wrappers as well as their associated instantiated component files that must be passed on to the place-and-route tool, since they are not written to the final netlist:

```
add_file -verilog -job_owner par "my_ip_core.v"  
add_file -verilog -job_owner par "my_ip_core_enc.v"
```

add_folder

The **add_folder** command adds a custom folder to a project.

Syntax

add_folder *folderName*

Creates logical folders with files in various custom hierarchy groupings within your Project view. These custom folders can be specified with any name or hierarchy level.

```
add_folder verilog  
add_folder verilog/common_rtl  
add_folder verilog/common_rtl/prep
```

For more information about custom folders, see [Managing Project File Hierarchy, on page 113](#) in the *User Guide*.

add_vivado_ip

Xilinx

Adds Vivado IP to the current project with the specified options.

Syntax

add_vivado_ip *IPsource* *IPimplMode* *IP format additionalOptions*

Where the arguments include the following options:

- *IPsource*
 - dir** *IPdirectory* | **-list** *IPlistFile* | **-xci** *fileName.xci* | **-xcix** *filename.xcix* | **-dcp** *fileName.dcp*
- *IPimplMode*
 - mode** [**absorb** | **white_box**]
- *IP format*
 - edif** | **-vm**
- *additionalOptions*
 - ip_output_location** *directory*

For example, you can specify the arguments for the `add_vivado_ip` command with the options shown below:

```
add_vivado_ip
  -xci-filename.xci {-netlist }
  [-verilog wrapper_filename]
  -mode {absorb | white_box}
  -edif|-vm
  [-log logfilename.log]
```

The following table describes the complete list of command arguments and options.

Option	Description
IPsource Options	
-dir IPdirectory	Imports all IPs in the directory within a single run. Specify IPs for xci, xcix, or dcp files in different directory hierarchies. If XCI/XCIX and DCP is available for same IP, XCI/XCIX will be picked up.
-list IPlistFile	Imports all IPs listed in the list file within a single run. Each xci, xcix, or dcp file must be added in the list file as one file per line.
-xci -xcix -dcp filename	Adds the IP source project or database file located in the IP directory. An xci, xcix, or dcp file is required if the -dir and -list options are not specified. To add Vivado IP successfully, you either need xci, xcix, or dcp file. If the input is xci, xci is added to the place-and-route options file. And if the input is DCP, DCP is added to the place-and-route options file to fill in the IP functionality. If the file is available, the Vivado IP is added successfully.
-vm -edif	Adds the Vivado IP to the Synplify project in Verilog netlist or EDIF netlist format. <ul style="list-style-type: none"> • edif- Adds the Vivado IP to the synplify project as EDIF netlist. • vm(Default) - Adds the Vivado IP to the Synplify project as Verilog netlist. In this format, the default result format for the synthesis output will be VM.
-netlist	Adds a Vivado-synthesized netlist file to the project.
-verilog filename	Includes the specified Verilog file as part of IP. The -verilog option is used only with the -netlist option which is specified once followed by the list of netlist files with each file preceded by the file type (-verilog).
IPimplMode Options	

Option	Description
-mode absorb white_box	Specify one of the following modes: <ul style="list-style-type: none"> white_box (Default)- Treats the IP block as a macro and uses the netlist for timing estimation, but does not include it in the final netlist. The xci, xcix, or dcp file provides the content of the IP block for place and route. absorb - Allows synthesis to optimize the content of the IP block and include it in the final netlist.
<i>additionalOptions</i>	
-ip_output_location directory	Specifies the directory where all required IP files are to be placed. By default, adds IP files under the ipcores/ <i>ipName</i> directory within the current project directory.
-log logfilename.log	Writes messages to the specified log file. If this option is omitted, writes messages to the default log file add_ip_name.log file in the logs directory under the current project directory.
-help	Displays the help for the add_vivado_ip command. For additional details about the command syntax, specify: add_vivado_ip -help -more

Description

The `add_vivado_ip` command incorporates Vivado IP into the design by letting you either import the RTL or a synthesized netlist for the IP. For details about usage, see [Incorporating Vivado IP, on page 740](#) in the *User Guide*, and the examples below.

Use `add_vivado_ip` instead of the `xdc2fdc` command, which was available in previous releases.

Example 1: Automatically Synthesized Netlist

The following examples runs Vivado synthesis for the IP and generates a synthesized Vivado netlist. It translates the XDC constraints to FDC, and includes the netlist and translated constraints in a project, which it adds to the top-level design. It adds the original XDC constraints to the

place-and-route directory and writes out the results to the default log file in the project directory. The remainder of the design is synthesized using timing information from the IP netlist.

- With XCI input, White Box mode, and EDIF IP format.

```
add_vivado_ip -edif -xci
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci} -mode white_box -ip_output_location ./my_ip_import_loc
```

- With XCI input, White Box mode, and VM IP format.

```
add_vivado_ip -vm -xci
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci} -mode white_box -ip_output_location ./my_ip_import_loc
```

- With XCI input, Absorb mode, and EDIF IP format

```
add_vivado_ip -edif -xci
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci} -mode absorb -ip_output_location ./my_ip_import_loc
```

- With XCI input, Absorb mode and VM IP format.

```
add_vivado_ip -vm -xci
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci} -mode absorb -ip_output_location ./my_ip_import_loc
```

- With a file which consists a list of XCI file names, White Box mode, and VM IP format.

```
add_vivado_ip -vm -list {iplist.txt} -mode white_box -
ip_output_location ./my_ip_import_loc
```

The contents of the list file, iplist.txt, which contains one entry per line:

```
multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci
```

```
adder_ip/adder_ip.srcs/sources_1/ip/c_addsub_0/c_addsub_0.xci
```

- With specified input directory, White Box mode, and VM IP format

```
add_vivado_ip -vm -dir
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/ } -mode white_box
-ip_output_location ./my_ip_import_loc
```

Example 2: Pre-Synthesized Vivado Netlist

The following example uses the basic options to add a Vivado-generated synthesized wrapper and netlist file to a synthesis project. It translates the XDC constraints to FDC, and includes the wrapper, netlist and translated constraints in a subproject, which it adds to the top-level design. It adds the original XDC constraints to the place-and-route directory and writes out the results to the default log file in the project directory. The remainder of the design is synthesized using timing information from the IP netlist.

```
add_vivado_ip -vm -xci  
{multiplier_ip/multiplier_ip.srcs/sources_1/ip/mult_gen_0/mult_gen_0.xci} -mode white_box -netlist -verilog  
{multiplier_ip/mult_gen_0_netlist.v} -ip_output_location  
.my_ip_import_loc
```

analyst

Changes and manipulates the schematic views of the netlist. Note that the following HDL Analyst commands can be used without the analyst prefix as well.

analyst clone_view	analyst critical_path	analyst dissolve
analyst filter	analyst flatten	analyst get_selected
analyst group	analyst pop	analyst push
analyst select	analyst unfilter	analyst view

analyst clone_view

Opens a copy of the current view.

analyst clone_view *designID*

designID

The design ID to clone. If not specified, clones the current view.

analyst critical_path

Filters the view to show the instances that are part of the critical path, if available.

analyst critical_path

analyst dissolve

Removes targeted hierarchies from the design. Contents of the hierarchy are put into the level that originally contained the hierarchy.

analyst dissolve *collection*

collection

Collection of instances to dissolve.

analyst filter

Filters the view by selected instances and ports.

analyst filter

analyst flatten

Flattens the current view.

analyst flatten

analyst get_selected

Returns the names of currently selected objects.

analyst get_selected [-inst] [-net] [-port] [-pin]

-inst

Returns the names of selected instances. If no *-type* option is set, the names of all selected objects are returned.

-net

Returns the names of selected nets. If no *-type* option is set, the names of all selected objects are returned.

-port

Returns the names of selected ports. If no *-type* option is set, the names of all selected objects are returned.

-pin

Returns the names of selected pins. If no *-type* option is set, the names of all selected objects are returned.

analyst group

Creates a graphical group of instances.

analyst group [collection] [-name groupName]

collection

Instances to group. All instances must be on the same level of the hierarchy.

-name groupName

The name of the graphical group to be created.

analyst pop

Pops up the hierarchy.

analyst pop

analyst push

Pushes down the hierarchy.

analyst push *hierarchyName*

hierarchyName

The name of the group or instance to push the hierarchy down into.

analyst select

Selects specified objects.

analyst select [*collection*] [-append] [-clear] [-instances] [-primitives]

collection

The ID of the collection to select.

-append

Appends objects to the selection list.

-clear

Clears the selection list.

-instances

Selects all instances in the current view.

-primitives

Selects all primitives (leaf) instances in the current view.

analyst unfilter

Unfilters the view.

analyst unfilter

analyst view

Opens a schematic view.

analyst view *designID*

designID

The design ID to view.

assign_to_region

Synplify Premier

The `assign_to_region` command assigns start and end points of the critical paths to a region created in the Design Planner. You need to determine which critical paths are collected for `$sel` constraints that are assigned to the region.

Syntax

assign_to_region *regionName \$sel*

The following table describes the command arguments.

Option	Description
<i>regionName</i>	Name of the region created for the floorplan, such as <code>rgn1</code> .
<code>\$sel</code>	Determine which critical path start and end points can be collected and are placed in <code>\$sel</code> and which are then assigned to the floorplan region you created.

Example

The following example contains the Tcl expand commands to find the HDL instances for the start and end points of the critical path and create a region for these assignments.

```
set start [define_coll [find -inst {mem_add[9:0]} -hier]
    [find -inst {mau_yu[1:0]} -hier]]
set end [find -inst {yu_add[9:0]} -hier]
set sel [expand -from $start -to $end -hier]
select $assignments1
create_region region1 21 36 32 31
assign_to_region region1 $assignments1
```

If the collection of start and end points is placed in `assignments` and the region name is `rgn1`, then use the following command to assign them to the region:

```
assign_to_region rgn1 $assignments
```

check_fdc_query

Runs the constraint checker for constraints using the `get_*` and/or `all_*` query commands specified in the timing constraint file for the project.

Syntax

```
check_fdc_query [-full_check]
```

Arguments and Options

-full_check

Runs the full constraint checker before checking the query commands. The default is to run the `check_fdc_query` command without this option.

When the `-full_check` option is *not* specified, the command only runs the constraint syntax checker, which reduces runtime significantly, since most objects being searched are found in pre-mapping and do not require full mapping to be run. However, this option does not find bit-blasted registers and objects using the advanced `-filter @property =~` commands, where the property is created or applied during mapping because it requires optimizations such as register replication.

For example, if a 4-bit RAM output is targeted with the `get_cell` command, the differences in the results are shown below:

Command	Run Stage	Results
Default (without <code>-full_check</code>)	Pre-mapping	<code>ram_out [3:0]</code>
With <code>-full_check</code>	Mapping	<code>ram_out [3]</code> <code>ram_out [2]</code> <code>ram_out [1]</code> <code>ram_out [0]</code>

Description

The `check_fdc_query` command reads the `.fdc` constraint file of the current project file. It runs the constraint checker for the following object query commands that are used with FDC constraints:

all_* Commands	get_* Commands
<code>all_clocks</code>	<code>get_cells</code>
<code>all_fanin</code>	<code>get_clocks</code>
<code>all_fanout</code>	<code>get_nets</code>
<code>all_inputs</code>	<code>get_pins</code>
<code>all_outputs</code>	<code>get_ports</code>
<code>all_registers</code>	

The report provides feedback on how these query commands are applied and ensures that the commands are used properly with constraints in the constraint file.

Collections created with `define_scope_collection`, `find`, and `expand` are not covered by this Tcl command. You can check these SCOPE collections in the HDL Analyst and the SCOPE interface. The report does not cover the `define_io_standard` constraint either.

Example

Invoke `check_fdc_query` from the Tcl command line for the project. You can also invoke it from a shell window.

The command writes out the results of the object query commands to the `projectName_cck_fdc.rpt` file that opens in the GUI. You may need to run the constraint checker (Run->Constraint Check) to find additional issues with constraints.

The following example shows the results of running the constraint checker in the *projectName_cck_fdc.rpt* file.

```
FDC query commands results
*****
#####
# 1019 : set_multicycle_path 2 -from [get_cells -hier {*[4]}]
# line 175 in :
C:/check_fdc_query/all_clocks/test1_basic/top_translated.fdc
Results of query command: get_cells -hier {*[4]}
(none)
#####
# 1027 : set_multicycle_path 3 -to [all_clocks]
# line 196 in :
C:/check_fdc_query/all_clocks/test1_basic/top_translated.fdc
Results of query command: all_clocks
    clka
    clkb
    dcm|CLK0_BUF_clock_CLKIN1
    dcm|clk0_i_clock_CLKIN1
    dcm|CLK0_BUF_1_clock_CLKIN1
```

The syntax checker reports the object query commands and any issues it found and writes them to the *projectName_scck.rpt* file.

```
# Synopsys Constraint Checker (syntax only), version map610dev,
Build 1085R
# Copyright (C) 1994-2016, Synopsys, Inc.
# Written on Tue Apr 30 15:39:07 2013
##### DESIGN INFO
#####
Top View:          "top"
Constraint File(s):
"C:\check_fdc_query\all_clocks\test1_basic\top_translated.fdc"
"C:\builds\syn201309_063R\lib\fdc_query.fdc"

# Run constraint checker to find more issues with constraints.
#####
#####
```

No issues found in constraint syntax.

Clock Summary

Start

Requested

Requested

Clock

Clock

Clock Frequency Period Type Group

Clock	Frequency	Period	Type	Group
clka	100.0 MHz	10.000	declare	default_clkgroup
clkb	50.0 MHz	20.000	declared	default_clkgroup
dcm CLK0_BUFB	clock_CLKIN1	200.0 MHz	5.000	derived (from clka) default_clkgroup
dcm CLK0_BUFB	clock_CLKIN1	50.0 MHz	20.000	derived (from clka) default_clkgroup

See Also

- [Constraint Checking, on page 176](#)
- [Constraint Checking Report, on page 218](#)

command_history

Displays a list of the Tcl commands executed during the current session.

Syntax

```
command_history [-save filename]
```

Arguments and Options

-save

Writes the list of Tcl commands to the specified *filename*.

Description

The command_history command displays a list of the Tcl commands executed during the current session. Including the -save option, saves the commands to the specified file to create Tcl scripts.

Examples

```
command_history -save C:/DesignsII/tut/proto/myTclScript.tcl
```

See Also

- [recording, on page 133](#)

constraint_file

The **constraint_file** command manipulates the constraint files used by the active implementation.

Syntax

```
constraint_file
  -enable constraintFileName
  -disable constraintFileName
  -list
  -all
  -clear
```

The following table describes the command arguments.

Option	Description
-enable	Selects the specified constraint file to use for the active implementation.
-disable	Excludes the specified constraint file from being used for the active implementation
-list	Lists the constraint files used by the active implementation
-all	Selects (includes) all the project constraint files for the active implementation.
-clear	Clears (excludes) all the constraint files for the active implementation

Examples

List all constraint files added to a project, then disable one of these files for the next synthesis run.

```
% constraint_file -list
attributes.fdc clocks1.fdc clocks2.fdc eight_bit_uc.fdc
% constraint_file -disable eight_bit_uc.fdc
```

Disable all constraint files previously enabled for the project, then enable only one of them for the next synthesis run.

```
% constraint_file -clear
% constraint_file -enable clocks2.fdc
```

create_fdc_template

Lets you create an initial constraint file (.fdc) for your specific design.

Syntax

```
create_fdc_template [-period float] [-in_delay float] [-out_delay float]
```

The following table describes the `create_fdc_template` command options.

Option	Description
-period float	Specifies the default values for port clocks.
-in_delay float	Specifies the default values for the input delay ports.
-out_delay float	Specifies the default values for the output delay ports.

Examples

Each port clock includes a `set_clock_groups` header with details shown below, which can help you determine whether clocks have been optimized away or if there are any derived clocks.

```
#####
### Individual "set_clock_groups" commands for all "clka" derived clocks
### appear at the end of this file. Enabling a given command will make the
### given clock asynchronous to all other clocks. If a given clock (below)
### does not appear in the final Performance Summary (in the *.srr
### file after synthesis), the clock may have been optimized away due to
### Gated/Generated Clock Conversion.
### See the "CLOCK OPTIMIZATION REPORT" in the *.srr file.
### Below is a list of any clocks derived from "clka":
###   clka DERIVED CLOCKS:
###     dcm|CLK0_BUFA_1_derived_clock_CLKIN Clock Object: {t:dcm_inst.CLK_BUFA_0}
###     dcm|CLK0_BUFB_1_derived_clock_CLKIN1 Clock Object: {t:dcm_inst.CLK_BUFB_0}
#####

set_clock_groups -disable -asynchronous -name {clka_group}
    -group {clka} -comment {Source clock clka group}

set_clock_groups -disable -asynchronous
    -name {dcm|CLK0_BUFA_1_derived_clock_CLKIN1_group}
    -group [get_clocks
        -of_objects [get_pins {t:dcm_inst.CLK_BUFA_0}]]
    -comment {Derived clock dcm|CLK0_BUFA_1_derived_clock_CLKIN1
        from source clock clka}
```

```
set_clock_groups -disable -asynchronous  
-name {dcm|CLK0_BUFB_derived_clock_CLKIN1_group}  
-group [get_clocks  
-of_objects [get_pins {t:dcm inst.CLK_BUFO.O}]]  
-comment {Derived clock dcm|CLK0_BUFB_derived_clock_CLKIN1  
from source clock clka}  
  
set_clock_groups -disable -asynchronous -name {clkb_group}  
-group {clkb} -comment {Source clock clkb group}
```

create_region

Synplify Premier

The `create_region` command is used to create a floorplan region.

Syntax

`create_region regionName coordinates`

The following table describes the command arguments.

Option	Description
<code>regionName</code>	Name of the region you create on the floorplan.
<code>coordinates</code>	Location of the floorplan region you create. Specify the lower-left and upper-right CLB corners of the region.

Example

The following example creates a region called `rgn1` and specifies its CLB locations on the floorplan.

```
create_region rgn1 21 36 32 31
```

cdpl_queue

Synplify Premier, Synplify Pro

Use the `cdpl_queue` command to set a CDPL queue, get the configuration setting of the CDPL queue, or clear the CDPL queue settings. These settings are valid only for the current session, and not saved in the .ini file or any other script. Create a Tcl file to save the settings and use it for subsequent sessions of the Synplify tool. This command does not have a GUI equivalent.

Syntax

`cdpl_queue -set | -get | -clear [queueType] [queueConfig]`

The `cdpl_queue` command includes the following options:

-set [queueType][queueConfig]

Sets a CDPL queue configuration.

- `queueType` is lowmem, highmem, mediummem, default or vendor. The queue type vendor is for all place and route jobs.
- `queueConfig` is the queue host file. See CDPL help documentation on how to set up a queue host file.

Example:

```
cdpl_queue -set highmem /remote/sbg_ui/myqueue/myhighmem.sge
```

-get [queueType]

Returns the configuration setting of the `queueType` specified. If `queueType` is not given, the command returns all the queue types set in the current session.

-clear

Clears all the queue settings.

Examples

To use this command, set up the following string in your project file:

```
cdpl_queue -set default $PWD/hosts_default
```

```
set_option cdpl 1
cdpl_queue -set default $PWD/hosts_default
cdpl_queue -set vendor $PWD/hosts_vendor
cdpl_queue -set lowmem $PWD/hosts_lowmem
cdpl_queue -set mediummem $PWD/hosts_mediummem
cdpl_queue -set highmem $PWD/hosts_highmem
```

- hosts_default - To set up host files.

cdpllogs - From the CDPL_LOG directory, access the log files from a typical CDPL setup. In the cdpllogs directory, check the files master.XXX.XXX.XXX.log and worker.XX.XX.XXX.log to verify how the cdpl_queue command executes during the synthesis flow.

For more information, see the CDPL (Common Distributed Processing Library) documentation.

CDPL Configuration Queue Type

cdpl_queue command supports five queue types. Specific queue types are used for specific primary jobs. For memory requirements for specific jobs, see [Memory Requirement Recommendations](#), on page 51.

- vendor - Used for running place and route jobs.
- highmem - Used for high memory consumption jobs such as the partition, global optimization, pre-partition, or the GCC flow in distributed synthesis mode.
- mediummem - Used for medium memory consumption jobs such as distributed synthesis running in QOR mode.
- lowmem - Used for low memory consumption jobs such as area estimation or distributed compiler/fast distributed synthesis.
- default - Used if queueType is not given and the current session is used to run the job. This queue type is used for all other jobs.

Memory Requirement Recommendations

vendor	64GB + 8core for Xilinx devices
highmem	Synthesis flow, GCC/advanced synthesis in distributed synthesis - 32GB or more depending on the design.
mediummem	Distributed synthesis in QOR mode - 16GB or more depending on the design
lowmem	Area estimation, distributed compiler/fast distributed synthesis - 8 GB or more depending on the design.
default	Synthesis flow - 64GB - if place and route job is combined 32GB - if place and route is not combined

define_link

Use the `define_link` command to add or remove an instance or design block to/from the subproject mapping.

Syntax

```
define_link [{i | v}: {instanceName | designBlockName}]
[-name linkSubProjectName] [-run_type value] [-remove] [-list] [-parameter]
```

The following table describes the `define_link` command options.

Option	Description
i: <i>instanceName</i> or v: <i>designBlockName</i>	Specifies the name of the instance or design block to be linked for the subproject.
-name	Specifies the name of the subproject implementation that contains the design block.
-run_type <i>value</i>	Defines how the subproject is run. You can specify: <ul style="list-style-type: none">• top-down - Runs the top-down flow from HDL or SRS.• bottom-up - Runs the bottom-up flow from EDIF.
-remove	Removes the link to an instance or design block from the subproject database. When the <code>-remove</code> option is not used, the instance or design block is removed from the top-level project and added as a subproject for that project.
-list	Lists all existing links for the project.
-parameter	Maps the subproject base on the parameterized design block.

Examples

Links the instance or design block to the subproject implementation that contains the block definition. For example:

```
"define_link "i:inst1" -name proj1_1|rev_1"
```

or

```
"define_link "v:designblockA" -name proj2|rev_3"
```

design

Returns netlist data representing information about the design. Commands are available in both batch and GUI mode. Note that the following HDL Analyst find commands can be used without the design prefix as well.

design c_diff	design c_filter	design c_info
design c_intersect	design c_list	design c_print
design c_symdiff	design c_union	design close
design expand	design find	design get_prop
design get	design list	design open
design set	design top_level	

design c_diff

Returns a new find collection containing the differences between two existing find collections.

Syntax

design c_diff *collection1 collection2*

collection1

The first collection to compare.

collection2

The second collection to compare.

design c_filter

Filters a find collection based on set properties.

Syntax

design c_filter *collection pattern [-inst] [-net] [-port] [-pin] [-view]*

collection

The collection ID to filter.

pattern

Statement used to filter.

-inst

Returns matching instances. If no -type option (**-inst**, **-net**, **-port**, or **-pin**) is set, all types will be returned.

-net

Returns matching nets. If no -type option (**-inst**, **-net**, **-port**, or **-pin**) is set, all types will be returned.

-port

Returns matching ports. If no -type option (**-inst**, **-net**, **-port**, or **-pin**) is set, all types will be returned.

-pin

Returns matching pins. If no -type option (**-inst**, **-net**, **-port**, or **-pin**) is set, all types will be returned.

-view

Returns matching views. If no -type option (**-inst**, **-net**, **-port**, or **-pin**) is set, all types will be returned.

design c_info

Returns information about the contents of a find collection.

Syntax

design c_info [collection] [-array value]

collection

Find collection information to display.

-array value

Specify an array to store collection information in.

design c_intersect

Defines common objects that are included in each of the collections being compared.

Syntax

design c_intersect collectionList

collectionList

List of collections separated by spaces.

design c_list

Converts a collection to a Tcl list of objects.

Syntax

design c_list *collection*

collection

Collection to convert.

design c_print

Displays collections or properties in column format.

Syntax

design c_print *collection* [-prop *propertyName*] [-file *filename*] [-append]

collection

The collection to print as a table.

-prop

Writes a column in the table for properties of type *propname*.

-file

Writes the collection to *filename*.

-append

Appends to the file specified in **-file** rather than overwriting it.

design c_symdiff

Returns a new find collection containing the difference between two existing find collections.

Syntax

design c_symdiff *collection1* *collection2*

collection1 The first collection.

collection2 The second collection.

design c_union

Combines multiple collections into a single collection.

Syntax

design c_union *collectionList*

collectionList

Space-separated list of collections.

design close

Closes the specified design ID. If no design ID is provided, this command closes the current active design.

Syntax

design close *designID*

designID

The design ID to close.

design expand

The design expand command identifies objects based on their connectivity, by expanding forward from a given starting point. Returns a collection.

Syntax

design expand [-*objectType*] [-from *object*] [-thru *object*] [-to *object*] [-level *integer*]
[-hier] [-leaf] [-seq] [-print]

-objectType

Optionally specifies the type of object to be returned by the expansion. If you do not specify *objectType*, all objects are returned. The object type is one of the following:

-inst – returns all instances between the expansion points. This is the default.

-pin – returns all instance pins between the expansion points.

-net – returns all nets between the expansion points.

-port – returns all top-level ports between the expansion points.

-from object

Specifies a list or collection of ports, instances, pins, or nets for expansion forward from all listed pins. Instances and input pins are automatically expanded to all output pins of the instances. Nets are expanded to all output pins connected to the net. If you do not specify this argument, backward propagation stops at a sequential element.

-thru object

Specifies a list or collection of instances, pins, or nets for expansion forward or backward from all listed output pins and input pins respectively. Instances are automatically expanded to all input/output pins of the instances. Nets are expanded to all input/output pins connected to the net. You can have multiple -thru lists for product of sum (POS) operations.

-to object

Specifies a list or collection of ports, instances, pins, or nets for expansion backward from all the pins listed. Instances and output pins are automatically expanded to all input pins of the instances. Nets are expanded to all input pins connected to the net. If you do not specify this argument, forward propagation stops at a sequential element.

-level integer

Limits the expansion to N logic levels of propagation. You cannot specify more than one -from, -thru, or -to point when using this option.

-hier

Modifies the range of any expansion to any level below the current view. The default for the current view is the top level and is defined with the `define_current_design` command as in the compile-point flow.

-leaf

Returns only non-hierarchical instances.

-seq

Modifies the range of any expansion to include only sequential elements. By default, the `expand` command returns all object types. If you want just sequential instances, make sure to define the `object_type` with the `-inst` argument, so that you limit the command to just instances.

-print

Evaluates the `expand` function and prints the first 20 results. If you use this command from HDL Analyst, results are printed to the Tcl window; for constraint-file commands, the results are printed to the log file at the

start of the Mapper section. For a full list of objects found, you must use `c_print` or `c_list`. Reported object names have prefixes that identify the object type. There are curly braces around each name to allow for spaces in the names. For example:

```
{i:reg1}
{i:reg2}
{i:\weird_name[foo$]}
{i:reg3}
<<found 233 objects. Displaying first 20 objects. Use
c_print or c_list for all. >>
```

design find

Identifies design objects based on specified criteria.

Syntax

```
design find
[-objectType] pattern
[-seq]
[-inst instance]
[-net net]
[-port port]
[-pin pin]
[-view view]
[-depth viewNumber]
[-flat]
[-print]
[-filter expression]
-in value
-below value
-nocase
```

-objectType *pattern*

Specifies the type of object to be found. Object types are view, inst, port, pin, or net. The *pattern* argument is required and specifies the search pattern to be matched. The pattern can include the * and ? wildcard characters (see [Wildcards and Special Characters, on page 224](#)).

-seq

Finds sequential (clocked) instances (the -inst object type is not required). This argument is equivalent to `-filter @is_sequential`.

-hier

Extends the search downward through each level of the local hierarchy, instead of limiting the search to the current view. The default hierarchy separator for the search is the period (.).

-inst *instance*

Finds instances. If no -type option is set, find defaults to finding instances, nets, and ports.

-net *net*

Finds nets. If no -type option is set, find defaults to finding instances, nets, and ports.

-port *port*

Finds ports. If no -type option is set, find defaults to finding instances, nets, and ports.

-pin *pin*

Finds pins. If no -type option is set, find defaults to finding instances, nets, and ports.

-view *view*

Finds views. If no -type option is set, find defaults to finding instances, nets, and ports.

-depth *depth*

Sets the start depth for the search. *depth* may be a single hierarchy depth or a range. Using **-depth** with a range will cause **-hier** and **-flat** arguments to be ignored. Setting **-depth** to **0** will start the search at the top level.

-flat

Extends the search to all levels, but with **-flat**, the * wildcard character matches hierarchy separators as well as characters. This means that the following example finds instance a1_fft at the current level as well as the hierarchical instance a1.fft:

```
find -seq -flat a1*fft
```

-print

Prints the first 20 search results. For a full list of objects found, use **c_print** or **c_list**. If you use **find** from the shell, the results are printed to the Tcl window; if you **find** in the constraint file, the results are printed to the log file at the beginning of the Mapper section. Reported object names have prefixes that identify the object type and curly braces around each name to allow for spaces in the names as shown below:

```
{i:reg1}
{i:\weird_name[foo$]}
{i:reg2}
<<found 233 objects. Displaying first 20 objects. Use c_print
or c_list for all. >>
```

-filter expression

Further refines the results of find by filtering the results using the specified object property. For syntax details, refer to [find -filter, on page 229](#).

-in value

Searchs a collection to find a subset of the collection.

-below value

Sets the start point of the search to the specified instance path. Only search for objects below that point.

-nocase

Ignores the case when matching object names.

design get_prop

Returns a list of property values for an object or collection.

Syntax

design get_prop [objectName | collection] [-prop value] [-all] [-array value]

objectName | collection

The object or collection to use.

-prop

The property value to return.

-all

Prints all available properties.

-array

Specifies array where properties are stored. Only use with -all and the find collection must be limited to one object.

design get

Returns the design ID for the current active design.

Syntax**design get****design list**

Returns a list of available design IDs.

Syntax**design list****design open**

View schematic of the design in its current state.

Syntax**design open [netlist]***netlist*

The netlist to view.

design set

Sets specified design ID as the active design.

Syntax**design set *designID****designID*

The design ID to set as active.

design top_level

Returns a Tcl list of top-level information in the following order:
lib topModule topView.

Syntax

design top_level

For example:

```
design top_level  
work eight_bit_uc verilog
```

dh_module_sources

Synplify Premier

The `dh_module_sources` command returns a list of the source files defining a module hierarchy. You can also use this command to identify files that are not required for the specified module hierarchy.

Syntax

`dh_module_sources [-of] [-noself] moduleName`

moduleName

Specifies the module (view) name of the module hierarchy.

-of

Includes list of sources referencing the module.

-noself

Excludes sources in which the module definition occurs.

Examples

```
dh_module_sources work.sub2.verilog
```

```
dh_module_sources -of -noself work.sub2.vhdl
```

See Also

- [*dm_root*, on page 64](#)

dm_root

Synplify Premier

The dm_root command displays the name of the top-level module for the current design in batch mode.

Syntax

```
dm_root
```

Example

```
% dm_root
```

For example, this is the result of this command:

```
work.test.verilog
```

See Also

- [*dh_module_sources*, on page 63](#)

dump_metrics

Shows metrics and values available for the current implementation of a design. By default, only primary metrics are shown.

Syntax

dump_metrics [-show_queries] [-all]

-all

Shows detailed metrics as well as primary metrics for the design.

-show_queries

Shows available metrics in the form of a Tcl command that can be used to retrieve each metric.

The default output format is

table.[object|global]: metric = value [units] from job [/ description]

Examples

```
% dump_metrics -all

*clock_conversion.global: icg_removed = 0    from premap
    //Number of ICG latches removed
*clock_conversion.global: icg_retained = 0    from premap
    //Number of ICG latches not removed
*clock_conversion.global: clean_clock_trees = 1  from fpga_mapper
    //Number of non-gated/non-generated clock trees
*clock_conversion.global: clean_clock_pins = 270  from fpga_mapper
    //Number of clock pins driven by non-gated/non-generated clock trees
*clock_conversion.global: gated_clock_trees = 0   from fpga_mapper
    //Number of gated/generated clock trees
*clock_conversion.global: gated_clock_pins = 0   from fpga_mapper
    //Number of clock pins driven by gated/generated clock trees
*clock_conversion.global: instancesConverted = 0  from fpga_mapper
    //Number of sequential instances converted
*clock_conversion.global: instancesNotconverted = 0 from fpga_mapper
    //Number of sequential instances left unconverted
*hdl_compile.global: modified_files = 28    from compiler
    //Total number of HDL input files compiled
*hdl_compile.global: modified_modules = 11    from compiler
    //Total number of modules compiled
*hdl_compile.global: total_modules = 11    from compiler
    //Total number of modules
*hdl_compile.global: total_files = 28    from compiler
    //Total number of HDL input files
*misc.global: Part = xc7vx485tffg1157-1  from fpga_mapper
*runtime.global: realtime = 3.154000 seconds from compiler
*runtime.global: cputime = 1.809612 seconds from compiler
```

```
*runtime.global: realtime = 1.452000 seconds from premap
*runtime.global: cputime = 1.622410 seconds from premap
*runtime.global: realtime = 9.881000 seconds from fpga_mapper
*runtime.global: cputime = 9.828063 seconds from fpga_mapper
*timing.global: "Worst Slack" = -0.445800 ns from fpga_mapper
utilization.global: LUT1 = 31 from fpga_mapper
utilization.global: LUT2 = 64 from fpga_mapper
utilization.global: LUT3 = 45 from fpga_mapper
utilization.global: LUT4 = 84 from fpga_mapper
utilization.global: LUT5 = 57 from fpga_mapper
utilization.global: LUT6 = 160 from fpga_mapper
utilization.global: IBUF = 1 from fpga_mapper
utilization.global: IBUFG = 1 from fpga_mapper
utilization.global: IOBUF = 24 from fpga_mapper
*utilization.global: "I/O primitives" = 26 from fpga_mapper
utilization.global: BUFG = 1 from fpga_mapper
*utilization.global: "I/O Register bits" = 0 from fpga_mapper
*utilization.global: "Total Luts" = 411 from fpga_mapper
```

Note: The * denotes a primary metric.

```
% dump_metrics -show_queries

query_metric clock_conversion.icg_removed -jobname premap
query_metric clock_conversion.icg_retained -jobname premap
query_metric clock_conversion.clean_clock_trees -jobname fpga_mapper
query_metric clock_conversion.clean_clock_pins -jobname fpga_mapper
query_metric clock_conversion.gated_clock_trees -jobname fpga_mapper
query_metric clock_conversion.gated_clock_pins -jobname fpga_mapper
query_metric clock_conversion.instancesConverted -jobname fpga_mapper
query_metric clock_conversion.instancesNotconverted -jobname fpga_mapper
query_metric hdl_compile.modified_files -jobname compiler
query_metric hdl_compile.modified_modules -jobname compiler
query_metric hdl_compile.total_modules -jobname compiler
query_metric hdl_compile.total_files -jobname compiler
query_metric misc.Part -jobname fpga_mapper
query_metric runtime.realtime -jobname compiler
query_metric runtime.cputime -jobname compiler
query_metric runtime.realtime -jobname premap
query_metric runtime.cputime -jobname premap
query_metric runtime.realtime -jobname fpga_mapper
query_metric runtime.cputime -jobname fpga_mapper
query_metric {timing.Worst Slack} -jobname fpga_mapper
query_metric {utilization.I/O primitives} -jobname fpga_mapper
query_metric {utilization.I/O Register bits} -jobname fpga_mapper
query_metric {utilization.Total Luts} -jobname fpga_mapper
```

Naming Conventions for Metrics

The naming convention used for metrics consists of the following:

- Table – Represents a group of related metrics, such as, timing, runtime, or clock conversion.
- Metric Name – Descriptive string used to query metrics. This name usually consists of lower case letters with underscores between words.
- Units – Values associated with the metric, such as ns or percent are only shown if details are specified.
- Object – Some metrics are associated with an object, while others are global. Objects can be a clock net name, view name, or an instance path.
- Description – Brief description of the metric.

For example, clock conversion metrics can be specified as follows:

Table	Metric Name	Description
clock_conversion	clean_clock_trees	Number of non-gated/non-generated clock trees
clock_conversion	clean_clock_pins	Number of clock pins driven by non-gated/non-generated clock trees
clock_conversion	gated_clock_trees	Number of gated/generated clock trees
clock_conversion	instancesConverted	Number of sequential instances converted
clock_conversion	instancesNotconverted	Number of sequential instances left unconverted

See Also

To query metrics for a design, see the following commands:

- [query_available_metrics](#), on page 126
- [query_metric](#), on page 129
- [query_metric_details](#), on page 131

encryptIP

The encryptIP script lets you encrypt data with the OpenIP scheme.

Download the script

(<https://solvnet-plus.synopsys.com/s/article/encryptIP-Perl-Script-1576173366118>) and run it directly from Perl. The Perl command line syntax for running the script is as follows:

Perl encryptIP Script Syntax

```
encryptIP
  -in | input inputFile
  -out | output outputFileName
  -c | cipher "{des-cbc | 3des-cbc | aes128-cbc}"
  -k | key symmetricEncryptionKeyInTextFormat
  -kx | keyx symmetricEncryptionKeyInHexadecimalFormat
  -bd | build_date ddmmmyyyy
  -om | outputmethod "{plaintext | blackbox | persistent_key}"
  -incv | includevendor vendorKeyBlock
  -dkn | datakeyname sessionKeyName
  -dko | datakeyowner sessionKeyOwner
  -a | author dataAuthor
  -v | verbose
```

You must specify all required parameters.

-in input	Names the input HDL file to be encrypted.
-out output	Names the output file generated after encryption.
-c cipher	<p>Specifies the symmetric encryption cipher. The key length must match the algorithm being used, with each character using 8 bits.</p> <ul style="list-style-type: none"> • des-cbc specifies the Data Encryption Standard (DES); uses a 64-bit key. • 3des-cbc specifies the Triple Data Encryption Standard (Triple DES); uses a 192-bit key. • aes128-cbc specifies the Advanced Encryption Standard (AES Rijndael); uses a 128-bit key. <p>See Encryption and Decryption, on page 651 in the <i>FPGA Synthesis User Guide</i> for an overview.</p>

-k key	Specifies the symmetric data decryption key used to encode your HDL data block. The key is in text format, and can be any string (e.g. ABCDEFG). The exact length of the key depends on the data method you use.
-kx keyx*	Optional parameter. Specifies the symmetric encryption key in hexadecimal format.
-bd build_date	Specifies a date (ddmmmyyyy). The IP only works in Synopsys software released after the specified date. This option lets you force users to use newer Synopsys FPGA releases that contain more security features. Contact Synopsys if you need help in deciding what build date to use.
-om outputmethod	Determines how the IP is treated in the output after synthesis: <ul style="list-style-type: none"> • plaintext specifies that the IP is unencrypted in the synthesis netlist. • blackbox specifies that the IP is treated as a black box, and only interface information is in the output. • persistent_key is the default setting. It is used for Lattice designs, and re-encrypts the output after synthesis in accordance with the OpenIP standard. <p>See Specifying the Script Output Method for OpenIP Encryption, on page 668 for more information.</p>
-incv includevendor	Optional parameter that specifies a key block for an EDA vendor, so that IP can be read by the vendor tools. C
-dkn datakeyname	Specifies a string that denotes your session key, that was used to encrypt your IP.
-dko datakeyowner	Optional parameter that names the owner of the session key. The value can be any string.
-a author	Optional parameter that names the author of the session key. The value can be any string.
-v verbose	Specifies that the script run in verbose mode.

Example of encryptIP (OpenIP) Script Output

The following is an example of the script output:

The diagram illustrates the structure of the encryptIP script output. A light blue rectangular area contains the script code. Red annotations with arrows point to specific parts of the code:

- A red bracket labeled "Key block with session key" spans the first two lines of the script.
- A red bracket labeled "Key block header with key information" spans the entire section from "%%% protect begin_protected" to the end of the first data block.
- A red bracket labeled "Data block" points to the base of the second data block.
- A red bracket labeled "Data block header with encryption method" spans the entire section from "%%% protect end_protected" to the end of the script.

```
%%% protect protected_file 1.0
<optional unencrypted HDL>

%%% protect begin_protected
%%% protect key_keyowner=Synopsys
%%% protect key_keyname=SYNP05_001
%%% protect key_block
U9n263KwF7RWb8GSz7C+700tKshqQgTmb8UdRxISekIJDfon/
...
<other key blocks>

%%% protect data_method=aes128-cbc
%%% protect data_block
UWhcm3CPmGz27DXAMQZF8rY7hSsvIwedXiP59HYZHJfoMIM/
...
%%% protect end_protected
<optional unencrypted HDL>
```

For brief descriptions of the pragmas used in the output of the encryptIP script, see [Pragmas Used by Encryption Scripts, on page 73](#).

encryptP1735

Run the encryptP1735 script directly from Perl.

The script supports different models for encrypting HDL files and accessing the encrypted information (see [IEEE 1735 Encryption Use Models, on page 74](#)). The use model is determined by how blocks are marked for encryption in the HDL, combined with the information in the public keys file, which is described in [Public Keys File, on page 72](#).

Perl encryptP1735 Script Syntax

```
encryptP1735
  [-l | list listofFiles]
  [-pk | public_keys keyFileName]
  [-sk | showkey]
  [-verbose]
  [-verilog]
  [-vhdl]
  [-log logFileName]
  [-h | -help]
```

The following table describes the command-line arguments.

-l list	Specifies a list of the files to be encrypted; <i>listofFiles</i> is a list of the non-encrypted HDL input files with each filename entry on a separate line.
-pk public_keys	Specifies the public keys repository file. This file contains public keys for various tools. If the encryption envelope contains a key block with a particular keyowner and keyname, the script searches the public keys file to find a corresponding public key to use during key-block generation. See Public Keys File , on page 72 for information about this file.
-sk showkey	When used, the encryption script displays the session key in use. This is useful when random keys are used and you want to know which key is being used.
-verbose	Prints more detailed messages to the screen or log file.

-verilog	Specifies Verilog HDL file format when filename does not include a .v or .sv extension.
-vhdl	Specifies VHDL HDL file format when filename does not include a .vhdl or .vhd extension.
-log	Prints messages to the specified log file.

Public Keys File

The encryptP1735.pl encryption script requires public key information, which is specified in a designated file (-public_keys or -pk) option). This file includes public keys for each of the tools that are allowed access to the envelope with the encrypted data. The default keys file is called keys.txt and is located with the encryption script in the lib directory of the tool installation.

```
// Use verilog pragma syntax in this file
`pragma protect version=1
`pragma protect author="default"
`pragma protect author_info="default"

`pragma protect key_keyowner="Synopsys", key_keyname="SYNP15_1",
key_method="rsa"
`pragma protect key_public_key
<public_key_block>

// Add additional public keys below this line
// Add additional public keys above this line
`pragma protect data_keyowner="default-ip-author"
`pragma protect data_keyname="default-ip-key"
`pragma protect data_method="aes128-cbc"

// End of file
```

For the partial file with all pragmas use model, the following pragma attribute values must match the corresponding values in the key-block section of the encryption envelope:

```
`pragma protect key_keyowner="Synopsys", key_keyname="SYNP15_1",
key_method="rsa"
```

For information on the pragmas supported, see *Pragmas Used by Encryption Scripts, on page 73*.

Pragmas Used by Encryption Scripts

Both the encryptIP1735 and encryptIP (OpenIP) schemes use the pragmas described in the following tables. Note the following:

- The %%% protect directive must be placed at the exact beginning of a line.
- Exactly one white-space character must separate the %%% sequence from the command that follows.

The following table lists the pragmas used. In Verilog, the pragma must be preceded by the word `pragma`; this is not required in VHDL.

General Pragmas

<code>%%% protect protected_file 1.0</code>	Line 1 of file with encrypted data
<code>%%% protect begin_protected</code>	Marks the beginning of data to be encrypted
<code>%%% protect end_protected</code>	Marks the end of data to be encrypted
<code>%%% protect comment comment</code>	Single-line plain-text comment
<code>%%% protect begin_comment</code>	Marks the beginning of plain-text comment block
<code>%%% protect end_comment</code>	Marks the end of plain-text comment block

Data Block Pragmas (IP Author Data Encryption Information)

<code>%%% protect author=string</code>	Lists name of IP author
<code>%%% protect version=1</code>	Specifies encryption version; required only for IEEE 1735 Partial File with Standard Pragmas encryption use model
<code>%%% protect data_method=des-cbc 3des-cbc aes128-cbc</code>	Specifies the DES encryption method used: <ul style="list-style-type: none"> • des-cbc: Data Encryption Standard (DES) • 3des-cbc: Triple DES • aes128-cbc: Advanced Encryption Standard (AES)
<code>%%% protect data_block</code>	Immediately precedes the encrypted data block

Key Block Pragmas (IP Consumer Public Key Information)

<code>%%% protect key_keyowner=string</code>	Lists the owner of the key
--	----------------------------

<code>%%% protect key_keyname=string</code>	Name recognized by the Synopsys software to select the key block
<code>%%% protect key_method=string</code>	Encryption algorithm (RSA currently supported)
<code>w %%% protect key_block</code>	Immediately precedes encrypted key block

IEEE 1735 Encryption Use Models

Encryption models determine the scope of what gets encrypted and who can access the files. The encryptP1735 script lets you use these use models to encrypt HDL files:

Encryption Model	Details
Full file	Full-File Use Model , on page 74
Partial file with minimal pragmas	Partial File with Minimal Pragmas Use Model , on page 75
Partial file with standard pragmas (Recommended)	Partial File with Standard Pragmas Use Model , on page 76
Partial file with IEEE pragmas	Partial File with IEEE Pragmas Use Model , on page 78

Full-File Use Model

Use this model to encrypt the entire file. The entire HDL file is included in the decryption envelope. This model uses the `keys.txt` file to define which consumers have access to the encrypted data.

HDL	<ul style="list-style-type: none"> Contains no encryption pragmas (entire file is encrypted)
keys.txt	<ul style="list-style-type: none"> Contains public key information for multiple downstream tools Owners of all public keys listed have access to the entire file

This Verilog example encrypts the whole file (`tb_encrypt.v`), including the module named `secret` that it contains.

```
module secret (a, b, clk);
  input a, clk;
  output b;
  reg b=0;
```

```
always @(posedge clk) begin
    b = a;
end
endmodule
```

Run the script to encrypt the file:

```
perl encryptP1735.pl -list mylist -log encryptP1735.log
```

This command runs the script on a file (*mylist*), which lists the single Verilog file *tb_encrypt.v*. The command uses the default *keys.txt* file from the *lib* directory, and creates the decryption envelope file *tb_encrypt.vp*. Messages from the run are written to the *encryptP1735.log* file.

Partial File with Minimal Pragmas Use Model

With this encryption model, pragma protect begin and pragma protect end pragmas are used to indicate the start and end points of encryption regions. This model is best suited for cases where every encryption region must be encrypted for every key in the key file. When using this model, you must encrypt the entire module. The *encryptP1735.pl* script checks all the begin and end pragmas and generates the decryption envelope for each tool specified in the *keys* file.

- | | |
|----------|---|
| HDL | <ul style="list-style-type: none"> • Contains individual blocks marked for encryption (partial file) • Contains no public key information |
| keys.txt | <ul style="list-style-type: none"> • Contains public key information for multiple downstream tools • Owners of all public keys listed have access to all encrypted HDL blocks |

To illustrate this use model, consider a single, Verilog file (*tb_encrypt.v*) to be encrypted with only begin and end pragmas. This file contains a single module named *secret*.

```
`pragma protect begin
module secret (a, b, clk);
    input a, clk;
    output b;
    reg b=0;

    always @(posedge clk) begin
        b = a;
    end
endmodule
`pragma protect end
```

When you run the script with the following command, it uses the begin and end pragmas specified in the HDL file to encrypt the file:

```
perl encryptP1735.pl -list mylist -pk keys.txt
```

Here, the list file (*mylist*) names the Verilog file *tb_encrypt.v*. The command encrypts the data between the begin and end pragmas and creates the decryption envelope file *tb_encrypt.vp* for all tools listed in the key file. No log file (-log option) is specified, so messages are not written to a log file.

Partial File with Standard Pragmas Use Model

This is the recommended encryption use model. It is the most flexible because you can choose to encrypt individual blocks instead of the entire file and specify which tools can access each encrypted block on a per-block basis. When using this model, you must encrypt the entire module. This model requires a side file (*keys.txt*) that contains public key information for IP consumers.

- | | |
|----------|---|
| HDL | <ul style="list-style-type: none">• Marks individual blocks for encryption (partial file)• Includes public key encryption pragmas for each tool that is allowed access to an encrypted block, except the key itself (<i>key_public_key</i>)• Key information must match the information in the <i>keys.txt</i> file• Can allow different keys access to different blocks |
| keys.txt | <ul style="list-style-type: none">• Contains public key information for multiple downstream tools• Must include all public key encryption pragmas, as well as the public key itself• Only those owners of public keys listed in the HDL before the encrypted block have access to that block; all public keys listed in <i>keys.txt</i> need not be used in the HDL |

If there are conflicting pragmas defined in the HDL and the *keys.txt* file, the HDL pragma takes precedence over the corresponding pragma in the *keys.txt* file. For example, if *data_method* in the HDL is defined as *des-cbc* but the same pragma in the *keys.txt* file defines it as *aes128-cbc*, the HDL definition is used and copied to the decryption envelope:

```
data_method="des-cbc"
```

Verilog Example

This example encrypts a single Verilog file (tb_encrypt.v). The file contains a module named secret and includes all the encryption-related pragmas in the HDL, with the exception of key_public_key.

```

`pragma protect version=1
`pragma protect encoding=(enctype="base64")
`pragma protect author="author-a", author_info="author-a-details"
`pragma protect encrypt_agent="encryptP1735.pl",
  encrypt_agent_info="Synplify encryption scripts"
`pragma protect key_keyowner="Synopsys",key_keyname="SYNP15_1",
  key_method="rsa", key_block
`pragma protect
data_keyowner="ip-vendor-a",data_keyname="fpga-ip",
data_method="des-cbc"
`pragma protect begin

module secret (a, b, clk);
  input a, clk;
  output b;
  reg b=0;
  always @ (posedge clk) begin
    b = a;
  end
endmodule

`pragma protect end

```

The script is then run with the following command, where the list file (mylist) names a single file, tb_encrypt.v. The command uses the default keys.txt file from the *installLocation/lib* directory as the public keys file to create the decryption envelope file tb_encrypt.vp. No log file is specified, so messages from the run are not sent to a log file.

```
perl encryptP1735.pl -list mylist -pk keys.txt
```

VHDL Example

This example partially encrypts a VHDL file (tb_encrypt.vhd) where all encryption pragmas are specified in the file, except for key_public_key. The file contains a single entity/architecture pair named secret. For VHDL pragmas, just use the keyword protect.

```

library IEEE;
use IEEE.std_logic_1164.all;

```

```

entity secret is
    port (clk : in std_logic;
          a : in std_logic;
          b : out std_logic);
end entity;

`protect version=1
`protect author="author-a", author_info="author-a-details"
`protect encrypt_agent="encryptP1735.pl",
encrypt_agent_info="Synplify encryption scripts"
`protect encoding=(enctype="base64")
`protect key_keyowner="Synopsys", key_keyname="SYNP15_1",
key_method="rsa", key_block
`protect data_keyowner="ip-vendor-a", data_keyname="fpga-ip",
data_method="des-cbc"
`protect begin

architecture rtl of secret is
signal b_reg: std_logic;
begin
    process (clk) is
    begin
        if rising_edge(clk) then
            b_reg <= a;
        end if;
    end process;
    b <= b_reg;
end architecture;

`protect end

```

Encrypt the file with the following command, where the list file (mylist) names a single VHDL file, tb_encrypt.vhd. The command uses the default keys.txt file from the directory *installLocation/lib* as the public keys file to create the decryption envelope file tb_encrypt.vhdp. Messages are not captured in a log file.

```
perl encryptP1735.pl -list mylist -pk keys.txt
```

Partial File with IEEE Pragmas Use Model

Like the partial file with standard pragmas model, this use model is flexible, but it does not require a side file with the key block information. This makes it the most portable model, because all the information, including the key block information for each IP consumer, is included in the source code. When using this model, you must encrypt the entire module.

- HDL
- Marks individual blocks for encryption (partial file)
 - Includes public key encryption pragmas for each tool that is allowed access to an encrypted block, including the key itself (`key_public_key`)
 - Can allow different keys access to different blocks
-
- keys.txt
- Not required

```

module top (qa, qb, a, b, clk);
  input a, b, clk;
  output qa, qb;
  enc_and iand (qa, a, b, clk);
  enc_or ior (qb, a, b, clk);
endmodule

`pragma protect version=1
`pragma protect author="author-a", author_info="author-a-details"

`pragma protect key_keyowner="Synplicity", key_keyname="SYNP15_1",
key_method="rsa"

`pragma protect key_public_key
<public_key_block>

`pragma protect key_keyowner = "XYZ"
`pragma protect key_method = "rsa"
`pragma protect key_keyname = "XYZ8_001"

`pragma protect key_public_key
<public_key_block>

`pragma protect data_method="aes128-cbc"
`pragma protect begin

module enc_and (q, a, b, clk);
  input a, b, clk;
  output q;
  reg q=0;
  always @(posedge clk) begin
    q = a & b;
  end
endmodule

`pragma protect end

```

```
`pragma protect version=1
`pragma protect author="author-a", author_info="author-a-details"
`pragma protect key_keyowner="Synplicity", key_keyname="SYNP15_1",
key_method="rsa"
`pragma protect key_public_key
<public_key_block>

`pragma protect key_keyowner = "XYZ"
`pragma protect key_method = "rsa"
`pragma protect key_keyname = "XYZ8_001"
`pragma protect key_public_key
<public_key_block>

`pragma protect data_method="aes128-cbc"
`pragma protect begin

module enc_or (q, a, b, clk);
input a, b, clk;
output q;
reg q=0;
always @ (posedge clk) begin
    q = a | b;
end
endmodule

`pragma protect end
```

export_project

Creates a new module- or instance-based subproject that you can export and insert into the current project. By default, HDL-dependent files are included in the subproject. Use the various options for this command to help you create the subproject easily in batch mode.

Syntax

```
export_project -module moduleName | -instance instanceName  
    [-add_file fileName] [-filelist fileListName] [-no_default_hdl] [-run_type configType]  
    [-project projectName]
```

Arguments and Options

-module *moduleName*

Specifies the target HDL module for performing module-base subproject export.

-instance *instanceName*

Specifies the target instance for performing instance-based subproject export. You can specify the instance using either FDC notation {*i*:*insta*} or a hierarchy level such as top.b1.a2. If you do not specify an instance, a module-based subproject and all instances of the module are linked to the exported subproject by default.

-add_file *fileName*

Adds HDL source files to the project. Use either a relative or absolute path, for the source files to be included in your project. These are additional files used in conjunction with the default HDL-dependent files.

-filelist *fileListName*

Specifies a file that contains a list of HDL source files to be included in the subproject. Add one entry per line for each HDL file, specifying either a relative or absolute path to the source files in *fileListName*. When you use this option, the files listed in *fileListName* replace the default files listed in the parent project (prj).

-no_default_hdl

Prevents the automatic adding of HDL-dependent files.

-run_type configType

Specifies how to run the subproject for each implementation of the parent project. Choose one of the following configuration modes:

top_down - Compiles the subproject, linking to the top-level project before mapping to a netlist.

bottom_up - Maps the subproject to a netlist, before linking to the top-level project.

-project projectName

Specifies the project file name. Use either a relative or absolute path for the export subproject.

Examples

```
export_project -instance b1
export_project -module bblock
export_project -instance c1.b2.a1 -add defines.h
export_project -instance a1 -filelist test_source
    -run_type bottom_up
export_project -module bblock -project ./bblock_inst_b1/bblock.prj
```

generate_context

The `generate_context` command takes the port context information that provides connectivity at the instance boundaries and maps them to the output constraint file for subproject(s) of a hierarchical design. For details about using this feature from the synthesis tool, see [Generating Port Information for Instance-Based Subprojects, on page 176](#).

Syntax

`generate_context [instance constraintFilePath]`

instance constraintFilePath

Maps instances to the output constraint file.

generate_instance_constraints

The `generate_instance_constraints` command generates the timing and resource constraints for the instance-based subproject(s) of a hierarchical design. For details about using this feature from the synthesis tool, see [Allocating Resources for Instance-Based Subprojects, on page 172](#) and [Setting Initial Timing Budgets for Instance-Based Subprojects, on page 174](#).

Syntax

```
generate_instance_constraints
[-timing_budgets [instanceName]] | [-resource [instanceName]] |
[-port_context [instanceName]] | [-all [instanceList]]
```

The following table describes the `generate_instance_constraints` command options.

Option	Description
-timing_budgets	Specifies the timing constraints (for example, the clock definitions, I/O delays, and timing exceptions) for the instance-based subproject.
-resource	Specifies the resource constraints (for example, RAM and DSP usage) for the instance-based subproject.
-port_context	Generates port context information, such as ports tied to a fixed value or unused ports for the instance-based subproject with the bottom-up flow.
-all	Specifies both the timing and resource constraints for the selected instances of subprojects in the design hierarchy.
<i>instanceName</i>	Specifies the instance name of the subproject.
<i>instanceList</i>	Specifies the names for selected instances of the subprojects in the design hierarchy.

Examples

This example generates timing and resource constraints for all instance-based subprojects.

```
generate_instance_constraints -all
```

This example generates timing constraints for all instance-based subprojects.

```
generate_instance_constraints -timing_budgets
```

This example generates timing and resource constraints for instances I1, I2, and I3 of the subprojects.

```
generate_instance_constraints -all I1 I2 I3
```

This example generates port context data for instance B1 of the subproject.

```
generate_instance_constraints -port_context B1
```

generate_timing_budgets

The `generate_timing_budgets` command takes the timing and resources specified for the instances and maps them to the output constraint file for subproject(s) of a hierarchical design. For details about using this feature from the synthesis tool, see [Setting Initial Timing Budgets for Instance-Based Subprojects, on page 174](#).

Syntax

`generate_timing_budgets [instance constraintFilePath]`

instance constraintFilePath

Maps instances to the output constraint file.

get_env

The `get_env` command reports the value of a predefined system variable.

Syntax

```
get_env systemVariable
```

Use this command to view system variable values. The following example shows you how to use the `get_env` command to see the value of the previously created `MY_PROJECT` environment variable. The `MY_PROJECT` variable contains the path to an HDL file directory, so `get_env` reports this path.

```
get_env MY_PROJECT  
d:\project\hdl_files
```

In the project file or a Tcl script, you can define a Tcl variable that contains the environment variable. In this example, `my_project_dir` contains the `MY_PROJECT` variable, which points to an HDL file directory.

```
set my_project_dir [get_env MY_PROJECT]
```

Then, use the `$systemVariable` syntax to access the variable value. This is useful for specifying paths in your scripts, as in the following example which adds the file `myfile1.v` to the project.

```
add_file $my_project_dir/myfile1.v
```

get_option

The `get_option` command reports the settings of predefined project and device options. The options are the same as those for `set_option`. See [set_option, on page 149](#) for details.

Syntax

```
get_option -optionName
```

hdl_define

For Verilog designs, this command specifies values for Verilog text macros. You can specify text macro values that you would normally enter using the Verilog `define statement in a Verilog file included at the top of the synthesis project. The parameter value is valid for the current implementation only.

This command is equivalent to the set_option -hdl_define command.

Syntax

```
hdl_define
  -set "directive=value [directive=value ...]"
  -clear
  -list
```

Examples

```
hdl_define -set "SIZE=32"
```

This statement specifies the value 32 for the SIZE directive; the following statement is written to the project file:

```
set_option -hdl_define -set "SIZE=32"
```

To define multiple directive values using hdl_define, enclose the directives in quotes and use a space delimiter. For example:

```
hdl_define -set "SIZE=32 WIDTH=8"
```

The software writes the following statement to the prj file:

```
set_option -hdl_define -set "size=32 width=8"
```

See Also

[Compiler Directives and Design Parameters](#), on page 470 for information on specifying compiler directives in the GUI.

hdl_param

The `hdl_param` command shows or sets HDL parameter overrides. For the GUI equivalent of this command, select Project->Implementation Options->Verilog/VHDL.

Syntax

```
hdl_param
  -add {paramName}
  -list | -set paramName {paramValue}
  -clear
  -overrides
```

The following table describes the command arguments.

Option	Description
-add	Adds a parameter override to the project.
-list	Shows parameters for the top-level module only and lists values for parameters if there is a parameter override.
-set	Sets a parameter override and its value for the active implementation. Only the parameter value is enclosed within curly braces.
-clear	Clears all parameter overrides of the active implementation.
-overrides	Lists all the parameter override values used in this project.

Examples

In batch mode, to set generic values using the `set_option` command in a project file, specify the `hdl_param` generic with quotes and enclose it within `{}`. For example:

```
set_option -hdl_param -set ram_file {"init.mem"}
set_option -hdl_param -set simulation {"false"}
```

Suppose the following parameter is set for the top-level module.

```
set_option -hdl_param -set {"width=8"}
```

Add a parameter override and its value, then list the parameter override.

```
hdl_param -add {"size=32"}  
hdl_param -list "size=32"
```

You can specify `hdl_param` generics with different types, such as, an integer, `std_logic_vector`, or string value for VHDL. Here are some examples that show how to define these generics:

- With an integer value

```
set_option -hdl_param -set DATA_WIDTH 4
```

- With a `std_logic_vector` value

```
set_option -hdl_param -set MY_SLV {"0011"}
```

- Using a string value

```
set_option -hdl_param -set initialization_file {"table2"}
```

help

The help command displays the usage syntax and description for the specified command in the Tcl window.

Syntax

```
help commandName | wildcardTerm
```

Examples

```
help set_option  
usage:set_option -<optionName> <optionValue> -- set option  
          on active implementation  
  
get_option -<optionName> -- return option value on  
          active implementation  
  
help c_*  
c_diff  
c_filter  
c_info  
c_intersect  
c_list  
c_member  
c_print  
c_symdiff  
c_union
```

history

Returns a numbered list of executed Tcl commands.

Syntax

history [event *number*] | clear | info [*number*] | keep [*number*] | nextid | redo [*number*]]

event *number*

Returns command *number* from the history list.

clear

Clears the history list.

info [*number*]

Returns the last *number* of commands. If no *number* is included, returns all.

keep [*number*]

Sets the number of commands to save in history. Also returns the current setting.

nextid

Returns the index number that the next command will be assigned to in the history list.

redo [*number*]

Executes the *number* command. If no *number* is given, executes the latest command.

Examples

```
history event 12
```

```
history redo 4
```

impl

Synplify Pro, Synplify Premier

The `impl` command adds, removes, or modifies an implementation.

Syntax

```
impl
  -add [implName] [model]
  -name implName
  -remove implName
  -active [implName]
  -list
  -type implType
  -result_file
  -dir
```

The following table describes the command arguments.

Option	Description
-add	Adds a new device implementation. If: <ul style="list-style-type: none"> <code>implName</code> is not specified, creates a unique implementation name by incrementing the name of the active implementation. you want to add a new implementation copied from implementation <code>model</code>.
-name	Changes the name of the active implementation.
-remove	Removes the specified implementation.
-active	Reports the active implementation. If you specify an implementation name, changes the specified name to the active implementation.
-list	Lists all the implementations used in this project.
-type	Specifies the type of implementation to add. For example, the: <ul style="list-style-type: none"> -type fpga option creates an FPGA implementation. -type identify option creates an Identify implementation.
-result_file	Displays the implementation results file.
-dir	Displays the implementation directory.

Examples

The following command sequence lists all implementations, reports the active implementation, and then activates a different implementation.

```
% impl -list  
design_worst design_typical design_best  
  
% impl -active  
design_best  
  
%impl -active design_typical  
  
% impl -active  
design_typical  
  
% impl -add rev_1_identify mixed -type identify
```

ise2syn

Xilinx

Converts a Xilinx project to a Synplify Pro or Synplify Premier project. For information about using the utility to convert Xilinx projects, see [Converting Xilinx Projects with ise2syn, on page 761](#) in the *User Guide*.

Syntax

ise2syn [-ise project] [-synproject project] [-force_wbox cores]

-ise project	Specifies the Xilinx project (*.ise, *.xise, or *.xmp) to import.
-synproject project	Specifies a name for the synthesis project to be created.
-force_wbox cores	Forces listed cores to be treated as white boxes. The tool uses the ngc files for these cores instead of the HDL files. Separate multiple core names with commas. See Force White Box Cores Option (EDK), on page 491 for details.

This table describes the input and output files for the ise2syn utility:

Input files	The ise2syn utility can use any of the following as input. It extracts the information needed to construct a synthesis project. <ul style="list-style-type: none">• ise file• xise file• xmp EDK project file
Output files	The utility generates the following output files: <ul style="list-style-type: none">• Synthesis project file: prj• Synthesis constraint file: _conv.sdc, with converted ucf, ncf, or xcf constraints• Xilinx User Constraint file: _unapplied.ucf, that contains unconverted constraints

job

The job command, for place and route job support, creates, removes, identifies, runs, cancels, and sets/gets options for named P&R jobs.

Syntax

```
job jobName [-add jobType] | -remove | -type | -run [mode] | -cancel |
    -option optionName [optionValue] ]
```

job -list

The following table describes the command options.

Option	Description
-run	Runs the P&R job, according to the specified options:
-add <i>jobType</i>	Creates a new P&R job for the active implementation.
-cancel	Cancels a P&R job in progress.
-remove	Removes a P&R job from an active implementation
-list	Returns a list of the P&R jobs in the active implementation.
-remove	Removes a P&R job from the active implementation
-option <i>optionName</i> [<i>optionValue</i>]	Get/set options for <i>jobName</i> .
-type	Returns the P&R job type.

Examples

```
% job pr_2 -add par
% job pr_2 -run
% job pr_2 -option enable_run 1
% job pr_2 -option run_backannotation 1
```

launch_vivado

The `launch_vivado` command launches the Vivado place-and-route tool with the specified options. The Vivado executable is `$XILINX_VIVADO/bin/vivado`.

Syntax

```
launch_vivado
  -project projectFile | -project pathToProject
  [-gui | -batch]
  [-tcl tcl/ScriptName]
  [-device deviceString]
  [-log logFileName]
```

The following table describes the command arguments and options.

Option	Description
-project <i>projectFile</i>	Load specified Vivado project file
-project <i>pathToProject</i>	Create new Vivado project in the specified directory
-gui -batch	Launch Vivado in GUI or batch mode; the default is GUI
-tcl <i>tcl/Script</i>	Run the specified TCL script during Vivado startup
-device <i>deviceString</i>	Setup Vivado project with this device setting; if this option is omitted, the device specified in the project file is used by default
-log <i>logFileName</i>	Write output to specified log file; if this option is omitted, writes messages to the default log file <code>add_ip_name.log</code> file in the <code>logs</code> directory under the current project directory

Example

```
launch_vivado -gui -project my_project.prj
               -device XC7V2000T -log my_project.log
```

The above command launches Vivado in GUI mode (`-gui` option) and loads the project `my_project.prj` (`-project` option). The project uses a Xilinx XC7V2000T device (`-device` option); the results of command execution are written to the `my_project.log` file (`-log` option).

log_filter

Synplify Premier, Synplify Pro

This command lets you filter errors, notes, and warning messages. The GUI equivalent of this command is the Warning Filter dialog box, which you access by selecting the Warnings tab in the Tcl window and then clicking Filter. For information about using this command, see [Filtering Messages in the Message Viewer, on page 308](#) in the *User Guide*.

Syntax

```
log_filter -field fieldName==value  
log_filter -show_matches  
log_filter -hide_matches  
log_filter -enable  
log_filter -disable  
log_filter -clear
```

The following table shows valid *fieldName* and *value* values for the -field option:

Fieldname	Value
type	Error Warning Note
id	The message ID number. For example, MF138
message	The text of the message. You can use wildcards.
source_loc	The name of the HDL file that generated the message.
log_loc	The corresponding srr file (log).
time	The time the message was generated.
report	The log file section. For example, Compiler or Mapper.

Example

```
log_filter -hide_matches
log_filter -field type==Warning -field message==*Una*
    -field source_loc==sendpacket.v -field log_loc==usbHostSlave.srr
    -field report=="Compiler Report"
log_filter -field type==Note
log_filter -field id==BN132
log_filter -field id==CL169
log_filter -field message=="Input *"
log_filter -field report=="Compiler Report"
```

log_report

This command lets you write out the results of the `log_filter` command to a file. For information about using this command, see [Filtering Messages in the Message Viewer, on page 308](#) in the *User Guide*.

Syntax

You specify this command after the `log_filter` commands.

```
log_report -print fileName
```

Example

```
log_report -print output.txt
```

message_override

Allows you to suppress or override the log file message ID specifications with another type or limit.

Use -limit and -count, to limit the number of occurrences for all messages or specific messages in each log file. Messages that exceed the limit still show up in the Report Summary page and can be retrieved later from the message database. Suppressing messages is the same as -limit *ID* -count 0; errors cannot be suppressed or limited.

Syntax

```
message_override [-suppress value] [-read_file value] [-error value]
                  [-warning value] [-note value] [-remove value] [-global] [-clear]
                  [-limit value] [-count value]
```

The following table describes the command arguments and options.

Option	Description
-suppress <i>value</i>	Lists message IDs to suppress in the log file.
-read_file <i>value</i>	Reads the specified message override file.
-error <i>value</i>	Lists the message ID type as an error.
-warning <i>value</i>	Lists the message ID type as a warning.
-note <i>value</i>	Lists the message ID type as a note.
-remove <i>value</i>	Removes the override and resets the message type to its original value.
-global	Allows message operations to be applied globally. Otherwise, the override operation is only applied on messages for the current project.

Option	Description
-clear	Removes all overrides and resets messages to their original types.
-limit value	<p>Lists message IDs for a specific log file to the specified limit. Use with the -count argument. Also, use</p> <ul style="list-style-type: none"> • <code>message_override -limit default -count 1000 7</code> - Changes the default limits for all messages • <code>message_override -limit default -count unlimited</code> - Changes the default limits for all messages to unlimited
-count value	Counts the message IDs specified with the -limit argument.

Examples

It is recommended that you set the default limit to something other than unlimited, if possible. To do this, you can specify the following:

```
message_override -default_limit 100
```

More examples are shown below:

1. Upgrade messages with ID MF446 to be treated as an error.

```
message_override -error MF446
```

2. Suppress messages with ID BN101 (cannot be done for errors):

```
message_override -suppress BN101
```

3. Limit the number of occurrences of messages with IDs MF580 and MF581 to 1000 each in each log file:

```
message_override -limit {MF580 MF581} -count 1000
```

4. Unlimit the number of occurrences of messages with ID CL118 in logs and reports:

```
message_override -limit CL118 -count unlimited
```

5. Clear existing message overrides:

```
message_override -clear
```

open_design

The `open_design` command specifies a netlist file (.srs, .srp, or .srm) that can be used to search the database with the Tcl `find` command in batch mode. With `open_design`, you can use `find` without having to open an RTL, partitioned RTL, or Technology view. Use `open_design` to read in the .srs, .srp, or .srm file before issuing the `find` command. See the example below.

Syntax

`open_design filename`

- Where: `filename` is the RTL (.srs), partitioned RTL (.srp), or Technology (.srm) file that can be used to search the database. The specified netlist is loaded on-demand to minimize memory resources.

Example

```
project -load ../examples/vhdl/prep2_2.prj
open_design prep2_2.srs
set a [find -inst *]
c_print $a -file a.txt
open_design prep2_2.srm
set b [find -net *]
c_print $b -file b.txt
```

In the example above, `prep2_2` is loaded and the information from the RTL view file is read in. Then, the `find` command searches for all instances in the design and prints them to file `a`. Next, the technology view file is read in, then `find` searches for all nets in the design and prints them to file `b`.

See Also

- [find, on page 221](#).

open_file

The `open_file` command opens views within the tool. The command accepts two arguments: `-rtl_view` and `-technology_view`.

Syntax

```
open_file -rtl_view | -technology_view
```

The `-rtl_view` option displays the RTL view for the current implementation, and the `-technology_view` option displays the technology view for the current implementation. Views remain displayed until overwritten and multiple views can be displayed.

partdata

The partdata command loads part files and returns information regarding a part such as available families, family parts, vendors, attributes, grades, packages.

Syntax

```
partdata
  -load filename
  -family
  -part family
  -vendor family
  -attribute attribute family
  -grade [family:]part
  -package [family:]part
  -oem [family:]part
```

Option	Description
-load <i>filename</i>	Loads part file.
-family	Lists available technology families.
-part <i>family</i>	Lists all parts in specified family.
-vendor <i>family</i>	Returns vendor name for the specified family.
-attribute <i>attribute family</i>	Returns the value of the job attribute for the specified family.
-grade [<i>family:</i>] <i>part</i>	Lists the speed grades available for the specified part.
-package [<i>family:</i>] <i>part</i>	Lists the packages available for the specified part.
-oem [<i>family:</i>] <i>part</i>	Returns true if the part entered is an OEM part.

Example

The following example prints out the available vendors, their supported families, and the parts for each family.

```
% foreach vendor [partdata -vendorlist]
% puts VENDOR:$vendor;
% foreach family [partdata -family $vendor]
% puts \tFAMILY:$family;
% puts \t\tPARTS:;
% foreach part [partdata -part $family]
% puts \t\t$part;
```

post_par_resynthesis

Synplify Premier
Intel FPGA, Xilinx

Runs congestion alleviation and timing optimizations on the active implementation to improve routability and timing QoR of the design. Run post place and route resynthesis on the current implementation with the specified QPF/DCP database file. For details, see [Using Post Place and Route Resynthesis, on page 1151](#).

Syntax

post_par_resynthesis -dcp fileName | -qpf quartusProjectFile [-ncd ndcInputFile] [-rundir directoryPath]

[bg] Arguments and Options

Option	Description
-bg	Runs the job in background.
-qpf <i>quartusProjectFile</i> / -dcp <i>fileName</i>	Specifies the path to the Quartus QPF/Vivado DCP database file name.
-ncd <i>ndcInputFile</i>	Specifies the NCD input file name.
-rundir <i>directoryPath</i>	Specifies the directory path location to write the new version of the netlist generated after congestion alleviation and timing optimizations.

Examples

```
post_par_resynthesis -qpf \designDirectory\post_place.qpf  
-rundir c:\designName\postpar_resynthesis  
  
post_par_resynthesis -dcp \designDirectory\post_place.dcp  
-rundir c:\designName\postpar_resynthesis
```

process_bd_ip

Synplify Premier
Xilinx

Use a Tcl procedure to automatically process the block design (BD) file and convert it to a DCP file in the synthesis tool using batch mode.

Syntax

```
process_bd_ip -bdfile bdFileName | -cboard cboardName | -cboardpath cboardPath
-repopath repositoryPath -xboard xboardName
```

Arguments and Options

Option	Description
bdfile <i>bdFileName</i>	BD input file to be processed
xboard <i>xboardName</i>	Specifies the Xilinx board
cboard <i>cboardName</i>	Specifies the custom board
cboardpath <i>cboardPath</i>	Specifies the path where custom board file is located
repopath <i>repositoryPath</i>	Specifies the path to the IP repository

When the BD file is created using any Xilinx technology:

```
process_bd_ip -bdfile bdFileName
```

When the BD file is created using the Xilinx board:

```
process_bd_ip -bdfile bdFileName --xboard xboardName
```

When the BD file is created using a custom board:

```
process_bd_ip -bdfile bdFileName -cboard cboardName -cboardpath cboardPath
```

When the BD file consists of customized IPs along with the Xilinx IPs:

```
process_bd_ip -bdfile bdFileName -repopath repositoryPath
```

Multiple IP repositories can be used by using space between the paths:

```
process_bd_ip -bdfile bdFileName -repopath {repositoryPath1 repositoryPath2
repositoryPath3}
process_bd_ip -bdfile zusys.bd -repopath
{ /Reference_Design/StarterKit/ip_lib/axis_live_audio_1.0/
/Reference_Design/StarterKit/ip_lib/rgpio/ /Reference_Design/StarterKit
/ip_lib/SC0808BF/ }
```

program_terminate

Immediately terminates the tool session without prompting or saving any data.

Syntax

```
program_terminate
```

Arguments and Options

None

Description

The program_terminate command terminates a tool session without prompting or saving data. Use this command with caution as any unsaved data is lost and cannot be recovered.

Examples

```
program_terminate
```

program_version

Returns the product and software release version.

Syntax

```
program_version
```

Arguments and Options

None

Description

The program_version command returns the software product version number.

Examples

```
% program_version  
Synplify Pro L-2017.09
```

project

The project command runs job flows to create, load, save, and close projects, to change and examine project status, and to archive projects.

Syntax

```
project -run [-all] [-bg] implementationList [-impl implementationName]
[-clean] implementationList [-impl implementationName]
[-parallel] implementationList [-impl implementationName]
[-from processName] [-to processName]

project {-new [projectPath] | -load projectPath | -close [projectPath]}
[-save [projectPath] | -insert projectPath} |

project {-active [projectName] | -dir | -file | -name | -list | -filelist |
-fileorder filepath1 filepath2 [... filepathN] | -addfile filepath |
-movefile filepath1 [filepath2] | -removefile filepath}

project {-result_file resultFilePath | -log_file [logfileName]}

project -propagate_params

project -copy [-project filename] [-implement implementationName]
[-dest_dir pathname] [-copy_type {full | local | customize}]
[-add_srs [fileList] -no_input]

project -unarchive [-archive_file pathname/filename] [-dest_dir pathname]
```

run option

The run option lets you synthesize selected implementations of a Project file. You can choose to use the arguments for the run option independently or in any combination. The arguments available are described in the table below.

You can also use the Batch Run Setup dialog box to set the arguments to use with the run option. For details, see [Run Implementations Setup Command, on page 502](#).

Option	Description
-run [-all] [-bg] [-clean] [-from] [-parallel] [-to] [processName]	Synthesizes the project, according to the specified options:

Option	Description
	<p>You can use <i>run</i> with any of the following arguments:</p> <ul style="list-style-type: none">• -all - Runs all implementations of the active project.• -bg - Runs specified implementations in non-blocking background mode. This is the default.• -clean - Runs specified implementations, while ignoring up-to-date checking. This option cleans all previous results and forces a complete rerun.• -parallel - Runs specified implementations concurrently. Additional licenses are required for each job.• -quartus_project quartusFile - Runs Quartus backannotation.• -from - Runs from and including the specified process name.• -to - Runs up to and including the specified process name.• processName - Specifies the process name.

Option	Description
	<p>The <i>mode</i> can be one of the following keywords:</p> <ul style="list-style-type: none">• backannotation - (<i>Synplify Premier</i>) Runs backannotation only without impacting the synthesis or P&R results. To run backannotation only in batch mode, see Example: Running Backannotation in Batch Mode , on page 118.• compile - Compiles the active project, but does not map it.• constraint_check - Validates the syntax and applicability of constraints defined in one or more constraint files.• estimator - (<i>Synplify Premier</i>) Runs area estimation on the design• fsm_explorer - Selects optimum FSM-encoding style for finite-state machines.• netlist_optimizer - Runs netlist optimization.• syntax_check - Verifies that the HDL is syntactically correct; errors are reported in the log file.• synthesis - Default mode if no mode is specified. Compiles (if necessary) and synthesizes the currently active project. If followed by the -clean option (project -run synthesis -clean), resynthesizes the entire project, including the top level and <i>all compile points</i>, whether or not their constraints, implementation options or source code changed since the last synthesis. If not followed by -clean, only compile points that have been modified are resynthesized.• synthesis_check - Verifies that the design is functionally correct; errors are reported in the log file.• timing - Runs the Timing Analyst. This is equivalent to clicking the Generate Timing button in the Timing Report Generation dialog box with user-specified values.• write_netlist - Writes the mapped output netlist to structural Verilog (vm) or VHDL (vhm) format. You can also use this command in an incremental timing analysis flow. For details, see Run Menu , on page 498 and Generating Custom Timing Reports with STA, on page 482.

The following table describes the rest of the Project file command options.

Option	Description
-new [projectPath]	Creates a new project in the current working directory. If <i>projectPath</i> is specified, creates the project in the specified directory. For the Hierarchical Project Management flow, you can create a new subproject for the top-level project.
-load projectPath	Opens and loads the project file specified by <i>projectPath</i> .
-close [projectPath]	Closes the currently active project. If <i>projectPath</i> is specified, closes the specified project.
-save [projectPath]	Saves the currently active project. If <i>projectPath</i> is specified, saves the specified project.
-insert projectPath	Adds the specified project to the workspace project.
-insert projectFile	For the Hierarchical Project Management flow, adds the specified subproject for the top-level project. See <i>Example: Inserting Subproject for a Top-level Project</i> , on page 117.
-active [projectName]	Shows the active project. If <i>projectName</i> is specified, makes the specified project the active project.
-dir	Shows the project directory for the active project.
-file	Returns the path to the active project.
-name	Returns the filename (prj) of the active project.
-list	Returns a list of the loaded projects.
-filelist	Returns the pathnames of the files in the active project.
-fileorder filepath1 filepath2 [... filepathN]	Reorders files by adding the specified files to the end of the project file list.
-addfile filepath	Adds the specified file to the project.
-movefile filepath1 [filepath2]	Moves <i>filepath1</i> to follow <i>filepath2</i> in project file list. If <i>filepath2</i> is not specified, moves <i>filepath1</i> to top of list.
-removefile filepath	Removes the specified file from the project.

Option	Description
-result_file <i>resultFilePath</i>	Changes the name of the synthesis result file to the path specified.
-log_file [<i>logfileName</i>]	Reports the name of the project log file. If <i>logfileName</i> is specified, changes the base name of the log file.
-propagate_params	Synchronizes parameters for a subproject, when you make changes to these parameters in the design. This ensures that the parameters are propagated properly from the top-level design to lower-level blocks of the subproject. GUI equivalent: Run Subproject Parameter Sync. For an example, see Example: Using Subproject Parameter Sync , on page 117 .
-archive -project <i>filename</i> -root_dir <i>pathname</i> -archive_file <i>filename.sar</i> -archive_type {full local customize} -add_srs [<i>fileList</i>] -no_input	<ul style="list-style-type: none"> • project filename - copies a project other than the active project. If you do not use this option, by default the active project is copied. • root_dir pathname - specifies the top-level directory containing the project files. • archive_file filename - is the name of the archived project file. • archive_type - specifies the type of archive: <ul style="list-style-type: none"> • full - performs a complete archive; all input and result files are contained in the archive file. • customize - performs a partial archive; only the project files that you select are included in the archive. • local - includes only project input files in the archive; does not include result files. • add_srs - adds the listed srs files to the archived project. Use the -no_input option with this command. If <i>fileList</i> is omitted, adds all srs files for the project/implementations. The srs files are the RTL schematic views that are output when the design is compiled (Run->Compile Only). <p>For more information about, and examples of the project -archive command, see Archive Utility , on page 118.</p>

Option	Description
-copy <ul style="list-style-type: none"> -project filename -implement implementationName -dest_dir pathname -copy_type {full local customize} -add_srs [fileList] -no_input 	<ul style="list-style-type: none"> • project filename - copies a project other than the active project. If you do not use this option, by default the active project is copied. • implement implementation_name - archives all files in the specified implementation. • dest_dir directory_pathname - specifies the directory in which to copy the project files. • copy_type - specifies the type of file/project copy: <ul style="list-style-type: none"> • full - performs a complete copy; all input and result files are contained in the archive file. • customize - performs a partial copy; only the project files that you select are included in the archive. • local - includes only project input files in the copy; does not include result files. • add_srs - adds the listed srs files to the archived project. Use the -no_input option with this command. If fileList is omitted, adds all srs files for the project/implementations. The srs files are the RTL schematic views that are output when the design is compiled (Run->Compile Only). <p>For more information about, and examples of the project -copy command, see Archive Utility , on page 118.</p>
-unarchive <ul style="list-style-type: none"> -archive_file pathname/filename -dest_dir pathname 	<ul style="list-style-type: none"> • archive_file pathname/filename - is the name of the archived project file. • dest_dir pathname - specifies the directory in which to write the project files. <p>For more information about, and examples of the project -unarchive command, see Archive Utility , on page 118.</p>

project Command Examples

Load the project top.prj and compile the design without mapping it. Compiling makes it possible to create a constraint file with the SCOPE spreadsheet and display an RTL schematic representation of the design.

```
% project -load top.prj
% project -run compile
```

Load a project and synthesize the design.

```
% project -load top.prj  
% project -run synthesis
```

In the example above, you can also use the command `project -run`, since the default is `synthesis`.

Example: Inserting Subproject for a Top-level Project

You can add a project within a project using the following syntax:

```
project -insert [projectFile]
```

This command inserts a subproject within the top-level design, which becomes the active project.

```
% project -insert "./block2/block2.prj"  
% project -insert "./block3/block3.prj"  
% project -insert "./block1/block1.prj"  
% project -insert "./control/control.prj"
```

Example: Using Subproject Parameter Sync

If parameters have been changed in a design, you can make sure they are properly propagated to lower-level blocks of a subproject using the following Tcl command syntax:

```
project -propagate_params
```

The output generated for this command specifies the top-level parameters that have changed. For example:

```
set_option -hdl_param -set DATA_WIDTH 32  
set_option -hdl_param -set FIFO_DEPTH 128
```

Note: This command is the same as using the Subproject Parameter Sync option from the GUI.

Example: Running Backannotation in Batch Mode

Synplify Premier

To run backannotation only on the placed and routed data to get accurate timing and placement information back to the synthesis tool, enter the following commands in a Tcl file:

```
project -load projectName.prj
project -run backannotation
project -close projectName.prj
```

Then source this Tcl file in batch mode. For example:

```
synplify_premier_dp -batch -tcl ba.tcl
```

Archive Utility

The archive utility provides a way to archive, extract, or copy your design projects. An archive file is in Synopsys proprietary format and is saved to a file name using the .sar extension. You can also use this utility to submit your design along with a request for technical support.

The archive utility is available through the Project menu in the GUI or through the project Tcl command. See the following for details:

For information about ...	See ...
Archiving, un-archiving, or copying projects	Archiving Files and Projects, on page 187 in the <i>User Guide</i>
Archiving a project for technical support	Tech-Support Menu, on page 598

Project Archive Examples

The following example archives all files in the project and stores the files in the specified .sar file:

```
project -archive -project c:/proj1.prj
        -archive_file c:/archive/proj1.sar
```

The next example archives the project file (.prj) and all local input files into the specified .sar file.

```
project -archive -project c:/proj1.prj -archive_type local
        -archive_file c:/archive/proj1.sar
```

The following example archives the project file (.prj) only for selected .srs files into the specified sar file. Any input source files that are in the project are not included.

```
project -archive -project c:/proj1.prj -archive_type customize  
-add_srs -no_input -archive_file c:/archive/proj1.sar
```

Project Unarchive Example

The following example extracts the project files from c:/archive/proj1.sar to directory c:/proj1. All directories and sub-directories are created if they do not already exist.

```
project -unarchive -archive_file c:/archive/proj1.sar  
-dest_dir c:/proj1
```

Project Copy Examples

The following example copies only selected .srs files for the project to the destination project file directory.

```
project -copy -project d:/test/proj_2.prj -copy_type customize  
-add_srs -no_input -dest_dir d:/test_1
```

The next example copies all input source files and .srs files selected for the project to the destination project file directory.

```
project -copy -project d:/test/proj_2.prj -copy_type customize  
-dest_dir d:/test_1
```

project_data

The project_data command shows or sets properties of a project.

Syntax

```
project_data {-active [projectName] | -dir | -file}
```

The following table describes the command options.

Option	Description
-active	Set/show active project. With no argument, shows the active project. If <i>projectName</i> is specified, changes the active project to <i>projectName</i> .
-dir	Show directory of active project.
-file	Show the project file for the active project. The full path is included with the file name.

project_file

The `project_file` command manipulates and examines project files.

Syntax

```
project_file {-lib fileName [libName] | -name fileName [newPath] |
              -time fileName [format] | -date fileName | -type fileName |
              -savetype fileName [relative | absolute] -move fileName1 [fileName2] |
              -remove fileName | -top topModule |
              -tooltag applicationTagName | -toolargs [arguments] fileName }
```

The following table describes the command options.

Option	Description
-lib	Shows the project file library associated with <i>fileName</i> . If <i>libName</i> is specified, changes the project file library for the specified file to <i>libName</i> .
-name	Shows the project file path for the specified file. If <i>newPath</i> is specified, changes the location of the specified project file to the directory path specified by <i>newPath</i> .
-time	Shows the file time stamp. If a <i>format</i> is specified, changes the composition of the time stamp according to the combination of the following time formatting codes: %H (hour 00-23) %M (minute 00-59) %S (second 00-59) %d (day 01-31) %b (abbreviated month) %Y (year with century)
-date	Shows the file date.
-type	Shows the file type.
-savetype	Sets or shows whether a file is saved relative to the project or its absolute path.
-move	Positions <i>fileName1</i> after <i>fileName2</i> in HDL file list. If <i>fileName2</i> is not specified, moves <i>fileName1</i> to the top of the list.
-remove	Removes the specified file from the project file list.

Option	Description
-top	Sets or shows the top-level module of the specified file for the active implementation.
-tooltag	Sets or shows the third-party tool tag for the specified file.
-toolargs	Sets or shows the third-party tool tag arguments for the specified file.

Examples

List the files added to a project. Remove a file.

```
% project -filelist path_name1/cpu.v path_name1/cpu_cntrl.v  
path_name2/cpu_cntrl.vhd  
% project_file -remove path_name2/cpu_cntrl.vhd
```

project_folder

The project_folder command manipulates and examines attributes for project folders.

Syntax

```
project_folder [folderName] [-folderlist] [-filelist] [-printout] [-add] [-remove] [-r]
                [-tooltag] [-toolargs]
```

The following table describes the command options.

Option	Description
<i>folderName</i>	Specifies the name of the folder for which attributes are examined.
-folderlist	Lists folders contained in the specified project folder.
-filelist	Lists files contained in the specified project folder.
-printout	Prints the specified project folder hierarchy including its files.
-add	Adds a new project folder.
-remove	Removes the specified project folder.
-r	Removes the specified project folder and all its containing sub-folders. Files are removed from the project folder, but are not deleted.
-tooltag	Sets or shows the third-party tool tag name.
-toolargs	Sets or shows the additional arguments for the third-party tool tag.

Examples

Add a folder and list the files added to a project folder.

```
% project_folder -add newfolder
% project_folder -filelist newfolder
```

qsf2sdc

Intel FPGA

Converts a Quartus settings file (qsf) into a Synopsys FPGA sdc constraints file. Run the utility from any shell window.

Syntax

```
install_directory/bin/qsf2sdc -iqsf input_qsf_file -osdc output_sdc_file  
[-oqsf output_residue_qsf_file] [-all] [-silent]
```

install_directory Defines the path of the directory containing the Synplicity software

-iqsf Defines the input Quartus settings file (qsf)

-osdc Defines the output constraint file (sdc)

-oqsf Defines the residual constraints that cannot be converted and must remain in a Quartus settings file (qsf). This file is optional.

-all Converts any instances with location assignments. By default, only pin location assignments and IO standards are automatically converted.

-silent Suppresses all the successfully translated (#Supported) and unsuccessfully translated (#Unsupported) messages in the sdc file.

Currently, only two Quartus constraints are translated by this utility: `set_location_assignment` and `set_instance_assignment`. For a detailed discussion of the qsf2sdc utility and the constraints it translates, refer to [qsf2sdc Conversion, on page 160](#) in the *Reference Manual*.

qsf2syn

Intel FPGA

Takes a Quartus project and converts it into a synthesis project. The equivalent GUI command is Import->Import Intel FPGA QSF Project.

Syntax

qsf2syn [-qsf value] [-synproject value] [-black_box] [-constraints_only]

-qsf value	The Intel qsf file to import.
-synproject value	The name and path to the synthesis project to be created.
-black_box	Implements encrypted cores as black boxes.
-constraints_only	Lets you continue the conversion of the project from the previous point when you have to debug and fix errors. See Importing Quartus Projects, on page 718 for information on how to use it.

For detailed information about using the qsf2syn utility, see [Importing Projects from Quartus, on page 717](#).

query_available_metrics

Shows metrics that can be queried for the design. If specified, only metrics matching the required values are shown. Otherwise, shows all metrics for all tables. You should use the `query_available_metrics` command primarily for scripting, since it returns a Tcl list. For a more readable format, use the `dump_metrics` command.

Syntax

```
query_available_metrics [[table.]name]
```

`[table.]name`

Name of the metric to query, optionally preceded by '`table.`'.

For details about specifying metrics, see the [Naming Conventions for Metrics, on page 67](#).

Examples

This command returns values in a list of the form as follows:

```
{ {name1 object1 jobname1} {name2 object2 jobname2} ... }
```

1. Show a Tcl list of metrics that can be queried for the current implementation:

```
% query_available_metrics
```

Format is {{name1 object1 jobname1} {name2 object2 jobname2} ...}:

```
{clock_conversion.icg_removed {} premap} {clock_conversion.icg_retained {} premap} {clock_conversion.clean_clock_trees {} fpga_mapper}
{clock_conversion.clean_clock_pins {} fpga_mapper}
{clock_conversion.gated_clock_trees {} fpga_mapper}
{clock_conversion.gated_clock_pins {} fpga_mapper}
{clock_conversion.instancesConverted {} fpga_mapper}
{clock_conversion.instancesNotconverted {} fpga_mapper}
{hdl_compile.modified_files {} compiler} {hdl_compile.modified_modules {} compiler}
{hdl_compile.total_modules {} compiler} {hdl_compile.total_files {} compiler}
{misc.Part {} fpga_mapper} {runtime.realtime {} compiler}
{runtime.cputime {} compiler} {runtime.realtime {} premap}
{runtime.cputime {} premap} {runtime.realtime {} fpga_mapper}
{runtime.cputime {} fpga_mapper} {timing.Worst Slack} {} fpga_mapper}
{utilization.LUT1 {} fpga_mapper} {utilization.LUT2 {} fpga_mapper}
{utilization.LUT3 {} fpga_mapper} {utilization.LUT4 {} fpga_mapper}
```

```
{utilization.LUT5 {} fpga_mapper} {utilization.LUT6 {} fpga_mapper}
{utilization.IBUF {} fpga_mapper} {utilization.IBUFG {} fpga_mapper}
{utilization.IOBUF {} fpga_mapper} {{utilization.I/O primitives} {} fpga_mapper}
{utilization.BUFG {} fpga_mapper} {{utilization.I/O Register bits} {} fpga_mapper}
{{utilization.Total Luts} {} fpga_mapper}
```

2. Use a simple loop to show the values of all available metrics:

```
% foreach amt [query_available_metrics] {set metric
    [lindex $amt 0]; set object [lindex $amt 1];
    set job [lindex $amt 2]; puts "$metric $object:
[query_metric $metric -object $object -jobname $job]"}
```

Format is {{name1 object1 jobname1} {name2 object2 jobname2} ...}:

```
clock_conversion.icg_removed : 0
clock_conversion.icg_retained : 0
clock_conversion.clean_clock_trees : 1
clock_conversion.clean_clock_pins : 270
clock_conversion.gated_clock_trees : 0
clock_conversion.gated_clock_pins : 0
clock_conversion.instancesConverted : 0
clock_conversion.instancesNotconverted : 0
hdl_compile.modified_files : 28
hdl_compile.modified_modules : 11
hdl_compile.total_modules : 11
hdl_compile.total_files : 28
misc.Part : xc7vx485tffg1157-1
runtime.realtime : 9.881000
runtime.cputime : 9.828063
timing.Worst Slack : -0.445800
utilization.LUT1 : 31
utilization.LUT2 : 64
utilization.LUT3 : 45
utilization.LUT4 : 84
utilization.LUT5 : 57
utilization.LUT6 : 160
utilization.IBUF : 1
utilization.IBUFG : 1
utilization.IOBUF : 24
utilization.I/O primitives : 26
utilization.BUFG : 1
utilization.I/O Register bits : 0
utilization.Total Luts : 411
```

3. Show a Tcl list of metrics for the specified metric name:

```
% query_available_metrics cputime
```

Format is {{name1 object1 jobname1} {name2 object2 jobname2} ...}:

```
{runtime.cputime {} compiler} {runtime.cputime {} premap} {runtime.cputime
{} fpga_mapper}
```

-
4. Optionally, show a Tcl list of metrics for the specified table value:

```
% query_available_metrics runtime.cputime  
Format is {{name1 object1 jobname1} {name2 object2 jobname2} ...}:  
{runtime.cputime {} compiler} {runtime.cputime {} premap} {runtime.cputime  
{} fpga_mapper}
```

5. Enclose within curly braces {} whenever metrics contain spaces.

```
% query_available_metrics {timing.worst slack}  
Format is {{name1 object1 jobname1} {name2 object2 jobname2} ...}:  
{ {timing.Worst Slack} {} fpga_mapper}
```

See Also

See the related query commands below:

- [dump_metrics](#), on page 65
- [query_metric](#), on page 129
- [query_metric_details](#), on page 131

query_metric

Queries specific QoR metrics for the current implementation of a design.

Syntax

```
query_metric table.name [-object value] [-jobname value]
```

table.name

Name of the metric to query, preceded by '*table.*'.

-object *value*

Queries metrics associated with a specific object or global metrics if not specified.

-jobName *value*

Queries metrics associated with a specific job name or any job name if not specified.

For details about specifying metrics, see the *Naming Conventions for Metrics*, [on page 67](#).

Examples

Here are examples of how to query metrics for a design:

```
% query_metric clock_conversion.icg_removed  
0  
% query_metric clock_conversion.instancesConverted -jobname  
fpga_mapper  
1  
% query_metric runtime.realtime -jobname compiler  
3.15400  
% query_metric {timing.worst slack} -jobname fpga_mapper  
-0.445800  
% query_metric {utilization.total lut}s  
411
```

Suppose you have a design with compile points.

```
dump_metrics -show_queries  
...  
query_metric {timing.Worst Slack} -object r2p_cordic -jobname fpga_mapper  
...
```

Then you can query worst slack for one of the compile points.

```
% query_metric {timing.Worst Slack} -object r2p_cordic  
-jobname fpga_mapper  
-1.339500 ns {Estimated slack during compile point synthesis}
```

See Also

See the related query commands below:

- [dump_metrics](#), on page 65
- [query_available_metrics](#), on page 126
- [query_metric_details](#), on page 131

query_metric_details

Queries information about a QoR metric from the current implementation. Exactly one metric must match.

Syntax

```
query_metric table.name [-object value] [-jobname value]
```

table.name

Name of the metric to query, preceded by '*table.*'.

-object *value*

Queries metrics associated with a specific object or a global metric if not specified.

-jobName *value*

Queries metrics associated with a specific job name or any job name if not specified.

For details about specifying metrics, see the *Naming Conventions for Metrics*, [on page 67](#).

Examples

The Tcl command returns a list of three values: <*value*> <*units*> <*comment*>.

```
% query_metric_details clock_conversion.clean_clock_pins  
270 {} {Number of clock pins driven by non-gated/non-generated clock  
trees}  
  
% query_metric_details clock_conversion.icg_removed  
0 {} {Number of ICG latches removed}  
  
% query_metric_details runtime.realtime -jobname compiler  
3.154000 seconds {}  
  
% query_metric_details runtime.cputime -jobname fpga_mapper  
9.828063 seconds {}  
  
% query_metric_details {timing.worst slack} -jobname fpga_mapper  
-0.445800 ns {}  
  
% query_metric_details utilization.lut1
```

```
31 {} {}
```

Suppose you have a design with compile points.

```
dump_metrics -show_queries

...
query_metric {timing.Worst Slack} -object r2p_cordic -jobname fpga_mapper
...
```

Then you can query worst slack for one of the compile points.

```
% query_metric_details {timing.Worst Slack} -object r2p_cordic
    -jobname fpga_mapper
-1.339500 ns {Estimated slack during compile point synthesis}
```

See Also

See the related query commands below:

- [dump_metrics](#), on page 65
- [query_available_metrics](#), on page 126
- [query_metric](#), on page 129

recording

Allows you to record and store the Tcl commands generated when you work on your projects in the GUI. You can use this command for creating job scripts. The complete syntax for the recording command is:

recording
-on|-off
-file [*historyLogFile*]
-save [*historyLogFile*]
-state
-edit [*filename*]

In the command line:

- **-on|off-** turns Tcl command recording on or off. Recording mode is off by default.
 - **-file** - if you specify a history log file name, this option uses the specified file in which to store the recorded Tcl commands for the current session. If you do not specify a history log name, reports the name of the current history log file.
 - **-save** - if you do not specify a file name, updates the current history log. If you specify a history log file name, saves Tcl command history to the specified file.
 - **-state** - returns the Boolean value of recording mode.
 - **-edit** - displays the Tcl command log file in a text editor.

Examples

Turn on recording mode and save the Tcl commands in the `cpu_tcl_log` file created.

```
% recording -on  
% recording -file cpu tcl log
```

report_clocks

Reports the clocks in the design database.

Syntax

```
report_clocks -netlist [srsNetlistFile] [-csv_format] [-out fileName]
```

Arguments and Options

srsNetlistFile

The name of the srs netlist file. If this optional argument is not specified, the netlist file is taken from the active project implementation.

-csv_format

Displays the report in spread-sheet format.

-out

Specifies the name of the output report file (default name is *design-Name_clk.rpt*).

Description

The report_clocks command generates a report of the clocks found in the design database. The report includes a listing of the clock domain, parent clock, and clock type for each clock. If the -csv_format option is included, the report is output in spread-sheet format.

Examples

```
report_clocks c:/designs/mem_ctrl/mem_ctrl.srs -csv_format
```

report_external_tool_versions

Displays third-party external tool information, including installation environment variable settings and supporting tool versions.

Syntax

report_external_tool_versions [tool] -list -run_requirements

tool

Shows the version data for the tool specified. By default, the data is displayed for all external tools.

-list

Lists the names of all recognized external tools specified with this command.

-run_requirements

Shows the version data for all tools required to run the active/current job flow.

Examples

1. Show version information for all external tools.

```
report_external_tool_version
```

2. Show names of all recognized external tools.

```
report_external_tool_version -list
```

3. Show version information for Vivado tool.

```
report_external_tool_version vivado
```

report_messages

Queries messages from jobs based on ID or severity. This can be used to show duplicate messages that were suppressed in the log files.

Syntax

report_messages *logFile* [**-id** *value*] [**-severity** *value*] [**-out** *value*] [**-outa** *value*]

logFile

Specifies one or more log files to query for message details.

-id* *value

Restricts report to messages matching this id. Use the * or % wildcard to match any string.

-severity* *value

Limits messages to a specific severity. Can specify one or more messages as an error, warning, note, or advice. Multiple severities can be specified. If none is specified, all types are shown.

-out* *value

Name of the output file to be written rather than writing to the Tcl window.

-outa* *value

Name of the output file to be appended rather than writing to the Tcl window.

Output Format

Running the report_messages command produces a list of messages. Messages are displayed in the following format with each message beginning on a new line:

ID {messageText}

Examples

1. Query BN132 messages from the test.srr log file.

```
report_messages test.srr -id BN132
```

2. Query error and warning messages from the test.srr log file.

```
report_messages test.srr -severity error warning
```

3. Query critical warnings (CW) or downgradable errors (DE). You can search for these messages using wildcards. For example:

```
report_messages test.srr -id DE*
```

4. Query error and warning messages from the test.srr log file. and writes output to a file called messages in the current working directory.

```
report_messages test.srr -severity not warning -out messages
```

report_message_summary

Retrieves a summary of the messages in the log file. This summary contains a list of the message IDs and the number of occurrences for each type, along with their message descriptions.

Syntax

report_message_summary *logFile*

logFile

Specifies a log file to query for the message summary.

Examples

```
report_message_summary test.srr
```

run_config

The `run_config` command lets you set up the subproject implementations and configure how to run these implementations with the top-level project. You can specify:

- Which implementations to run for each subproject.
- Whether to run the subproject top-down or bottom-up.
- Which options to synchronize for all the subprojects.

Syntax

```
run_config
  -add implementationList [-impl implementationName]
  -set implementationList [-impl implementationName]
  -run_type runType [-subproject projectName] [-impl implementationName]
  -list
  -clear
  -reset_default
```

The following table describes the `run_config` command options.

Option	Description
-add	Lets you add subproject implementations to run with the top-level project for this configuration. Specify <i>implementationName</i> as { <i>projectName</i> <i>implName</i> }.
-set	Lets you select the subproject implementations to run with the top-level project for this configuration. Specify <i>implementationName</i> as { <i>projectName</i> <i>implName</i> }.
-impl	Specifies the parent implementation to apply for this configuration run. The default is the active implementation.
-subproject	Specifies the path for the subproject.
-run_type	Specifies how to run the subprojects. You can choose: <ul style="list-style-type: none"> • <code>top_down</code> • <code>bottom_up</code>

Option	Description
-list	Lists the current subproject implementations to run with the top-level project for this configuration.
-clear	For this configuration, removes all the subprojects for the top-level project.
-reset_default	Resets the subprojects back to its original settings to run with the top-level project for this configuration.

Examples

Set the subproject implementations for the top-level project to run with the current configuration.

```
run_config -set -impl rev_1  
{block2/block2.prj|rev_1 block3/block3.prj|rev_1  
block1/block1.prj|rev_1 control/control.prj|rev_1}
```

Specify whether to run the `top_down` or `bottom_up` flow for the specified subproject.

```
run_config -run_type top_down -impl rev_1 -subproject  
control/control.prj
```

run_tcl

The `run_tcl` command lets you synthesize your project using a Tcl script file from the Tcl Script window of the synthesis tool.

Syntax

`run_tcl [-fg] tclFile`

You can also use the following command:

`source tclFile`

These commands are equivalent.

The following table describes the `run_tcl` command options.

Option	Description
<code>-fg</code>	Synthesizes the project in foreground mode.
<code>TclFile</code>	Specifies the name of the Tcl file used to synthesize the project. To create a Tcl Script file, see Creating a Tcl Synthesis Script, on page 783 .

sch_sim_disp

Synplify Premier

The **sch_sim_disp** command displays simulation values for nets in the Control Panel signal watch list.

Syntax

```
sch_sim_disp
  -out [filename]
  -append filename
  -at [value] [netNames]
  -to value [netNames]
  -bracket
  -csv
  -table
```

The following table describes the **sch_sim_disp** command options.

Option	Description
-out	Writes display output to a file. The default is to write to the Tcl window.
-append	Appends data to the end of the specified output file.
-at <i>value</i>	Displays value at a specified time. The default is the current time.
-to <i>value</i>	Displays range of values from -at to -end.
-bracket	Displays values for times before and after the time range.
-csv	Specifies the CSV format for importing simulation values to a spreadsheet. CSV is a comma separated value text file.
-table	Displays a table format for the simulation values.
<i>netNames</i>	List of net names for which simulation values are displayed. The default is to display all watched nets.

sch_sim_load

Synplify Premier

The sch_sim_load command loads a VCD simulation file for use in the HDL Analyst view.

Syntax

```
sch_sim_load modulePath |VCDPath [-top_scope value] [-validate] [-log logFile]  
[-timeunits value] [-scopes value] [-top_module value] [-start value] [-end value]  
[-identify] [-simulator]
```

The following table describes the sch_sim_load command options.

Option	Description
<i>modulePath</i>	Specifies the module path through the test bench to the root netlist.
<i>VCDPath</i>	Specifies the path to the VCD file.
-top_scope <i>value</i>	Scope path through the test bench to the top-level module. This option is only used in simulator mode.
-validate	Validates the VCD file with the root netlist after being loaded.
-log <i>value</i>	Allows you to specify a log file to record diagnostic messages.
-timeunits <i>units</i>	Specifies the start/end time units, such as ps; for example: 100ps. The default time unit is pico seconds (ps).
-scopes <i>value</i>	Specifies a list of delimited VCD scope paths to load. The default is to load all scope paths.
-top_module <i>value</i>	Specifies the name of the top-level module in the VCD file. The default is the netlist root.
-start <i>value</i>	Specifies the start time for loading the VCD file.
-end <i>value</i>	Specifies the end time for loading the VCD file.
-identify	Loads the VCD file in Identify mode.
-simulator	Loads the VCD file in simulator mode.

Examples

Example 1: Opens a VCD file generated by VCS in the SRM View. The test bench path to the top-level module is `test.x`:

```
sch_sim_load -top_scope test.x single1.vcd
```

Example 2: Opens a VCD file generated by Identify in the SRS View:

```
sch_sim_load iice_1.vcd
```

sch_sim_watch

Synplify Premier

The `sch_sim_watch` command allows you to manage which nets to watch for simulation values.

Syntax

```
sch_sim_watch
  -add [-coll] [-select] [names]
  -rem [-coll] [names]
  -list
  -clear
```

The following table describes the sch_sim_watch command options.

Option	Option
-add	You can use the find -net command to select nets and then specify the -add option to watch this collection of nets. Otherwise, identify specific nets to watch.
-rem	Specifies a collection of nets or else specific nets to remove from the VCD Control Panel watch list.
-coll	List of net name expressions or net collection handles to watch. Specifies the find (net) collection <i>names</i> .
-select	Select the nets.
-list	Specifies the list of nets that are being watched for simulation values.
-clear	Removes all the nets from the VCD Control Panel watch list.

select

Selects specified objects.

Syntax

select *collection*

- append
- clear
- instances

The following table describes the select command options.

Option	Option
-append	Appends objects to the existing selection list.
-clear	Clears the selection list.
-instances	Selects all instances in the current view.

sdc2fdc

Translates legacy FPGA timing constraints to Synopsys FPGA timing constraints.

Syntax

sdc2fdc

Run it from the Tcl window in the synthesis tool.

See also

- [Converting SDC to FDC, on page 247](#) in the *User Guide*
- [sdc2fdc Conversion, on page 154](#) in the *Reference Manual*

Examples of sdc2fdc Translation

The following are examples of feedback after running the command. For information about the translated FDC file and handling the error messages, see [sdc2fdc Conversion, on page 154](#) in the *Reference Manual*.

```
% sdc2fdc

INFO: Translation successful.
See:"D:/bugs/timing_88/clk_prior/scratch/FDC_constraints/rev_2
      /top_translated.fdc"
Replace your current *.sdc files with this one.

INFO: Automatically updating your project to reflect the new
constraint file(s)
Do "Ctrl+S" to save the new settings.

% sdc2fdc

ERROR: Bad -from list for define_false_path: {my_inst}
Missing qualifier(s) (i: p: n: ...)
ERROR: Translation problems were found.
See:"D:/bugs/timing_88/clk_prior/scratch/FDC_constraints/rev_2
      /top_translate.log" for details.
      _translate.log

ERROR: Bad -from list for define_false_path {my_inst}
Missing qualifier(s) (i: p: n: ...)
```

```
"define_false_path -from {my_inst} -to i:abc.def.g_reg  
-through {n:bar}"  
Synplicity SDC source file: D:/bugs/timing_88/clk_prior/scratch  
/top.sdc.  
Line number: 79
```

set_option

The **set_option** command sets options for the technology (device) as well as for the design project.

Syntax

set_option -optionName optionValue

For syntax and descriptions of the options and related values, see one of the following tables:

- [Device Options for set_option/get_option](#)
- [Project Options for set_option/get_option](#)

Device Options for set_option/get_option

The following table lists *generic* device arguments for the technology, part, and speed grade. These are the options on the Implementation Options-> Device tab.

Information on all other Implementation Options tabs are listed in the next section, [Project Options for set_option/get_option, on page 150](#).

Option Name	Description
-technology parameter	Sets the target technology for the implementation. <i>parameter</i> is the string for the vendor architecture. Check the Device panel in the GUI or see Device Panel, on page 445 , for a list of supported families.
-part part_name	Specifies a part for the implementation. Check the Device panel of the Implementation Options dialog box (see Device Panel, on page 445) for available choices.
-speed_grade -value	Sets the speed grade for the implementation. Check the Device panel of the Implementation Options dialog box (see Device Panel, on page 445) for available choices.

Option Name	Description
-package value	Sets the package for the implementation. This option is not available for certain vendor families, because it is set in the place-and-route software. Check the Device panel of the Implementation Options dialog box (see <i>Device Panel</i> , on page 445) for available choices.
-grade -value	Same as <code>-speed_grade</code> . Included for backwards compatibility.

In general, device options are technology-specific, or have technology-specific defaults or limitations. For vendor-specific details, see *synhooks File Syntax*, on page 796.

Project Options for `set_option/get_option`

Below is a list of options for the `set_option` and `get_option` commands. Click the option below for the corresponding description and GUI equivalents. Options set through the Device tab are listed in *Device Options for `set_option/get_option`*, on page 149.

advanced_uram_features_on	allow_duplicate_modules	analysis_constraint
areadelay	area_delay_percent	auto constrain io
auto_infer_blackbox	automatic_compile_point	autosm
beta_vfeatures	bindandforce	block
cdpl	clique/cliquing	compile_strategy
compiler_compatible	compiler_constraint	constant_prop
constprop	constraint	continue_on_error
createhierarchy	dc_root	default_enum_encoding
design_intent	disable_io_insertion	distributed_compile
distributed_synthesis	dup	dw_foundation
dw_minpower	dw_library	dw_stop_on_nolic
enable64bit	enable_nfilter	enable_prepacking
enable_upf_1_0_applies_to_default_output	enable_upf_map_retention_cell	fanin_limit

fanout_limit	feedthrough	fix_gated_and_generated_clocks
floorplan/sfp	force_async_genclk_conv	force_gsr
frequency auto	frequency	globalthreshold
hamming3	hamming3_ded	hamming3_ded_recover_y
hdl_param	hdl_define	help
identify_debug_mode	ignore_undefined_libs	include_path
incremental	Incremental_synthesis	job (BA)
job (PC)	job (PR)	job (SP)
job (xtclsh)	level_hierarchy	libext
library_path	local_tmr_rename	log_file
looplimit	map_logic	maxfan
maxfanin	max_par_explorer	max_parallel_jobs
maxterms	max_terms_per_macrocel	mif_files_dirs
multi_file_compilation_unit	no_sequential_opt	num_critical_paths
num_startend_points	opcond	optimize_ngc
par_explorer	par_use_xflow	par_use_xilinx_partition
pipe	popfeed	project_relative_includes
preserve_registers	report_path	reporting_reportType
resolve_multiple_driver	resource_sharing	result_file
retiming	run_prop_extract	rw_check_on_ram
safe_case	soft/soft_buffers	support_implicit_init_nelist
support_multidim_vqm	supporttypedfltm	symbolic_fsm_compiler
syn_altera_model	synthesis_onoffPragma	synthesis_strategy
timequest	top_module	update_models_cp

<code>upf_iso_filter_element</code>	<code>upf_verbose</code>	<code>use_fsm_explorer</code>
<code>s_with_applies_to</code>		
<code>use_new_altera_map_per</code>	<code>use_vivado</code>	<code>user_physical_separation</code>
<code>validate_mif_files</code>	<code>vlog_std</code>	<code>voltage</code>
<code>write_apr_constraint</code>	<code>write_declared_clocks_only</code>	<code>write_verilog</code>
<code>write_vhdl</code>		

Option	Description	GUI Equivalent
<code>-advanced_uram_features_on</code>	Enables advanced URAM packing. By default, this option is enabled.	
<code>-allow_duplicate_modules</code>	For Verilog designs, allows the use of duplicate module names. When true, the last definition of the module is used by the software and any previous definitions are ignored. You should not use duplicate module names in your Verilog design, therefore, this option is disabled by default. However, if you need to, you can allow for duplicate modules by setting this option to 1.	Allow Duplicate Modules,
<code>-analysis_constraint path/filename.adc</code>	Specifies the analysis design constraint file (.adc) you can use to modify constraints for the stand-alone Timing Analyst only.	Constraint File section on the Timing Report Generation Parameters dialog box
<code>-areadelay percentValue</code>	Sets the percentage of paths you want optimized. This option is available only in certain device technologies.	Percent of design to optimize for timing, Device Panel
<code>-area_delay_percent percentValue</code>		

Option	Description	GUI Equivalent
-auto constrain io 1 0	Determines whether default constraints are used for I/O ports that do not have user-defined constraints. When disabled, only <code>define_input_delay</code> or <code>define_output_delay</code> constraints are considered during synthesis or forward-annotated after synthesis. When enabled, the software considers any explicit <code>define_input_delay</code> or <code>define_output_delay</code> constraints, as before.	Use clock period for unconstrained IO check box, Constraints Panel
-auto infer blackbox 0 1 2	<i>Synplify Premier</i> Determines action taken when an undefined Verilog module is encountered. <ul style="list-style-type: none"> • When 0 (None, the default) is specified, an undefined module causes the compiler to error out. • When 1 (With BIDR ports) is specified, a black box with bi-directional ports is created for the undefined module and a warning is issued. • When 2 (With BIDIR/IN ports) is specified, a black box with either input or bi-directional ports is created for the undefined module and a warning is issued. 	Auto Infer Blackbox field,
-automatic_compile_point 1 0	Enables/disables the automatic compile point flow, which can analyze a design and identify modules that can automatically be defined as compile points and mapped in parallel using Multiprocessing.	Automatic compile point check box, Options Panel
-autosm 1 0	Enables/disables the FSM compiler.	FSM Compiler check box, Options Panel
-symbolic_fsm_compiler 1 0		

Option	Description	GUI Equivalent
-beta_vfeatures 1 0	Enables/disables the use of Verilog compiler beta features.	Beta Features for Verilog,
-bindandforce 1 0	<i>Synplify Premier</i> Enables/disables the bind and force functions used in Verilog procedural assignments.	Support Bind & Force,
-block 1 0	Enables/disables I/O insertion in some technologies.	Disable I/O Insertion check box, Device Panel
-disable_io_insertion 1 0		
-cdpl 1 0	<i>Linux only</i> Runs distributed processing across multiple machines. For details, see Using Distributed Processing, on page 808	
-cliique 1 0	<i>Intel FPGA</i>	Perform Cliuqing check box, Device Panel
-cliuing 1 0	Enables/disables the grouping of logic functions in Intel FPGA technologies.	
-compile_strategy base fast	Sets the strategy for compile stage. base: It is default mode. This mode performs additional optimizations during compile stage to provide better QoR, but with longer runtimes. fast: This mode will turn off some optimizations to speed up the synthesis time, useful when it is necessary to reduce synthesis time or when the final result does not need to be optimal in speed/area.	Compile Strategy, Device Panel
-compiler_compatible 1 0	Disables pushing of tristates across process/block boundaries.	<i>Complement of the Push Tristates Across Process/Block Boundaries</i> check box, VHDL Panel and

Option	Description	GUI Equivalent
-compiler_constraint <i>constraintFile</i>	When multiple constraint files are defined, specify which constraint files are to be used from the Constraints tab of the Implementation Options panel.	Constraints Files, Constraints Panel
-constant_prop 1 0 -constprop 1 0	<i>Synplify Premier</i> Enables/disables constant propagation for netlist restructuring when the Enable prototyping tools option is enabled.	Constant Propagation check box, GCC & Prototyping Tools Panel
-constraint -option	Manipulates constraint files in the project: -enable/disable <i>filename</i> - adds or removes constraint file from active implementation -list - lists all enabled constraint files in active implementation -all - enables all constraint files in active implementation -clear - disables all constraint files in active implementation	Constraint Files, Constraints Panel

Option	Description	GUI Equivalent
-continue_on_error 1 0	The continue_on_error option serves two related, but separate functions. Compiler (<i>Synplify Premier</i>) - when enabled during compilation, allows the compiler to continue, if possible, after encountering a non-syntax error within a design unit and to resume compilation with the next design unit without stopping. Continuing on a compilation error is only available with the Synplify Premier products. Mapper - (<i>Synplify Pro</i> , <i>Synplify Premier</i>) When enabled during compile-point synthesis, allows the mapping operation to continue on error and synthesize the remaining compile points. The default for this option (0) is to stop on any compilation or synthesis error.	Continue on Error, Project View checkbox or Options Panel, or Configure Parallel or Compile Point Process Command
-createhierarchy 1 0	<i>Synplify Premier DP</i> Creates an additional level of hierarchy for always or process blocks in a design plan region.	Create Always/Process Level Hierarchy check box, GCC & Prototyping Tools Panel
-level_hierarchy 1 0		
-dc_root installPath	<i>Synplify Premier</i> Specifies path to Design Compiler installation for access to Synopsys foundation and minPower DesignWare libraries. The default is \$SYNOPSYS.	Design Compiler Installation Location field, VHDL Panel and
-default_enum_encoding default onehot gray sequential	(VHDL only) Sets the default for enumerated types.	Default Enum Encoding, VHDL panel (see VHDL Panel and)
-design_intent fast_turn_around timing_qor custom	<i>Synplify Premier</i> Selects pre-defined option configurations for the intended operation.	Design Intent dialog box (see Design Intent)

Option	Description	GUI Equivalent
-disable_io_insertion 1 0	Enables/disables I/O insertion in some technologies.	Disable I/O Insertion, Device Panel
-block 1 0	For more information about the impact of using this command, see syn_insert_pad, on page 351 .	
-distributed_compile 1 0	<p><i>Synplify Premier</i></p> <p>Enables/disables distributed compilation, which splits the design into smaller netlists (srs) so they can be run in parallel to improve runtime.</p> <p>For more information, see Running Distributed Compilation, on page 811.</p>	Distributed Compilation, Options Panel , Project view panel
-distributed_synthesis 1 0	<p><i>Synplify Premier</i></p> <p>When enabled, distributed synthesis automatically splits large designs into smaller sub-designs for parallel processing on separate machines (or on the same machine) to improve runtime. The tool uses the Synopsys CDPL (Common Distributed Processing Library) scheme for distributed processing.</p> <p>For more information, see Running Distributed Synthesis, on page 815</p>	Distributed Synthesis, Options Panel , Project view panel
-dup	<p>For Verilog designs, allows the use of duplicate module names. When true, the last definition of the module is used by the software and any previous definitions are ignored.</p> <p>You should not use duplicate module names in your Verilog design, therefore, this option is disabled by default. However, if you need to, you can allow for duplicate modules by setting this option to 1.</p> <p>Recommended to use -allow_duplicate_modules option instead of -dup option.</p>	Allow Duplicate Modules,

Option	Description	GUI Equivalent
-dw.foundation 1 0	<i>Synplify Premier</i> Specifies the source of the DesignWare models. When set to 1, uses standard DesignWare building blocks from the Synopsys foundation library. If dw_minpower is also 1, minimum power DesignWare building blocks from the Synopsys minPower library overlay the corresponding models from the DesignWare foundation library.	Use DesignWare Foundation Library checkbox, VHDL Panel and
-dw_library {dw.foundation [dw_minpower]}	<i>Synplify Premier</i> Specifies the Synopsys foundation library as the source of the DesignWare building blocks (same as -dw.foundation 1). Including the dw_minpower argument overlays the Synopsys DesignWare minPower library over the DesignWare foundation library.	Use DesignWare Foundation Library/ Use DesignWare MinPower Library checkbox, VHDL Panel and
-dw_minpower 1 0	<i>Synplify Premier</i> Specifies the source of the DesignWare building blocks. When set to 1, overlays minimum power building blocks from the Synopsys minPower library over the corresponding DesignWare foundation library building blocks.	Use DesignWare MinPower Library checkbox, VHDL Panel and
-dw_stop_on_nolic 1 0	<i>Synplify Premier</i> Specifies action taken when a Synopsys DesignWare building block is encountered and no license is found; 0 (default) black boxes DesignWare building block and continues, 1 stops with error message.	Stop synthesis if no DesignWare license found checkbox, VHDL Panel and
-enable64bit 1 0	Enables/disables the 64-bit mapping switch. When enabled, this switch allows you to run client programs in 64-bit mode, if available on your system.	Enable 64-bit Synthesis, Options Panel

Option	Description	GUI Equivalent
-enable_nfilter 1 0	<i>Synplify Premier</i> Enables the following netlist filtering options: -constprop (or -constant_prop), -createhierarchy (or -level_hierarchy), -create_mac_hier, -feedthrough (or -popfeed)	Enable Netlist Optimization, GCC & Prototyping Tools Panel
-enable_prepacking 1 0	<i>Xilinx Virtex-5 and Spartan-6 or newer technologies</i> Enable this option to prepare the synthesis netlist for advanced LUT combining, so that Xilinx ISE P&R can then pack into the LUT6_2 depending on available resources. Use in ISE version 10.1 service pack 1 or later. The default is on.	Enable Advanced LUT Combining (use only with Xilinx version 10.1 sp1 or later), see Virtex and Spartan-II and Later set_option Tcl Command .
-enable_upf_1_0_applies_to_default_output true false	<i>Synplify Premier</i> Use the enable_upf_1_0_applies_to_default_output Tcl variable with the set_isolation command to control how the power domain is instrumented when the -applies_to option is not specified. You can specify: <ul style="list-style-type: none">• false – Specifies both as the default value for the -applies_to option with UPF 1.0 and UPF 2.0 constructs. This is the default.• true – Specifies outputs as the default value for the -applies_to option with UPF 1.0 constructs and both as the default value for UPF 2.0 constructs.	Enter UPF command from the Tcl window

Option	Description	GUI Equivalent
-enable_upf_map_retention_cell 1 0	<i>Synplify Premier</i> Use the enable_upf_map_retention_cell option to ensure that the tool processes the map_retention_cell command properly: <ul style="list-style-type: none"> • 0 – Parses and ignores the map_retention_cell command. This is the default. • 1 – Processes the map_retention_cell command. The tool looks for user-defined retention models and issues an error when they cannot be found. 	Enter UPF command from the Tcl window.
-fanin_limit value	<i>Intel FPGA Max and Lattice ispMach technologies</i> Specifies the maximum fan-in.	Maximum Cell Fanin, Device Panel
-fanout_limit value	Sets the fanout limit guideline for the current project.	Fanout Guide, Device Panel
-maxfan value		
-feedthrough 1 0	<i>Synplify Premier</i> Enables/disables feedthrough optimization.	Feedthrough Optimization, GCC & Prototyping Tools Panel
-popfeed 1 0		
-fix_gated_and_generated_clocks 1 0	Performs gated and generated clock optimization when enabled. See Working with Gated Clocks, on page 828 and Optimizing Generated Clocks, on page 881 of the <i>User Guide</i> for details.	Gated Clocks, GCC & Prototyping Tools Panel
-floorplan filename	<i>Synplify Premier Design Planner</i> Specifies the Design Plan file (sfp) for the implementation.	Select sfp file, Design Planning Panel

Option	Description	GUI Equivalent
-force_async_genclk_conv [1 0]	<i>Synplify Premier</i> When enabled (1), gated-clock conversion occurs regardless of the presence of asynchronous set/reset signals in generated-clock logic or datapath latches. When disabled (the default), conversion is inhibited when asynchronous set/reset signals on generated-clock logic have signals different from the asynchronous signals on the driven datapath latches.	Force Generated Clock Conversion with Asynchronous Signals, GCC & Prototyping Tools Panel
-force_gsr yes no auto	<i>Lattice</i> The default value is no. Enables (yes) or disables forced use of the global set/reset routing resources. When the value is auto, the synthesis tool decides whether to use the global set/reset resources. The default value is auto.	Force GSR Usage, Device Panel
-frequency value	Sets the global frequency.	Frequency, Constraints Panel
-frequency auto	Enables/disables auto constraints.	Auto Constrain, Constraints Panel
-globalthreshold value	<i>Microchip</i> This option applies only to the following <i>Microchip technologies</i> : <ul style="list-style-type: none">• FUSION• IGLOO/IGLOOE/IGLOO+• ProASIC3/3E/3L Sets the minimum number of fanout loads. Signals that exceed the load value are promoted to global signals. Global buffers are assigned by the synthesis tool to drive the global signals.	Device Panel

Option	Description	GUI Equivalent
-hamming3 1 0	<i>Synplify Premier</i> When enabled, a Hamming3 encoding is used to build single-bit error correction logic in the FSM state registers.	High Reliability Panel
-hamming3_ded 1 0	When enabled in conjunction with the error monitoring TCL commands (<i>syn_create_error_net -double_bit</i> / <i>syn_connect</i>), a Hamming3 encoding is used to build single-bit error correction logic along with double-bit error detection in the FSM state registers.	High Reliability Panel
-hamming3_ded_recovery 1 0	When enabled, a Hamming3 encoding is used to build single-bit error correction logic and default state recovery for double-bit error detection in the FSM state registers.	High Reliability Panel
-hdl_define	For Verilog designs; used for extracting design parameters and entering compiler directives.	Compiler Directives and Design Parameters,
-hdl_param	Shows or sets HDL parameter overrides. See hdl_param , on page 89 for command syntax.	Use this command in the Tcl window of the UI.
-help	This option is useful for getting syntax help on the various implementation options used for compiling and mapping a design. For examples, see help for set_option , on page 181 .	Use this command in the Tcl window of the UI.

Option	Description	GUI Equivalent
-identify_debug_mode 1 0	When set option to 1, creates an Identify implementation in the Project view. Then, you can launch the Identify Instrumentor or Debugger from within the FPGA synthesis tools.	Select the Identify implementation, then launch: • Launch Identify Instrumentor or • Launch Identify Debugger
-ignore_undefined_libs 1 0	(VHDL only) When enabled (default), the compiler will ignore any declared library files not included with the source file. In previous releases, the missing library file would cause the synthesis tool to error out. To set this option to error out when a library file is missing (as in previous releases), use 0 for the command value.	Not available in the UI

Option	Description	GUI Equivalent
-include_path path	(Verilog only) Defines the search path used by the ‘include’ commands in Verilog design files. Argument <i>path</i> is a string that is a semicolon-delimited list of directories where the included design files can be found. The software searches for include files in the following order: <ul style="list-style-type: none"> • First, the source file directory. • Then, looks in the included path directory order and stops at the first occurrence of the included file it finds. • Finally, the project directory. The include paths are relative. Use the <i>project_relative_includes</i> option to update older project files.	Include Path Order, Verilog panel (see)
./extra_input/	The archive utility allows you to add the <i>extra_input</i> directory path for all include files and copies them to your project. Use the Add extra input path to project option on the Un-Archive Utility dialog box.	
-incremental 1 0	<i>Synplify Premier</i> If enabled, runs the incremental compiler. This can improve compiler runtime for large designs. For details, see Using the Incremental Compiler, on page 86 .	Incremental Compile, Verilog or VHDL panel; see and VHDL Panel .
-incremental_synthesis 1 0	<i>Synplify Premier</i> If enabled, runs incremental distributed synthesis to improve runtime on subsequent runs, when only part of a design has changed. For details, see Running Incremental Distributed Synthesis, on page 819 .	Incremental Distributed Synthesis; see Options Panel

Option	Description	GUI Equivalent
-job PR_job_name	If enabled, runs the specified place-and-route job with the appropriate vendor-specific place-and-route tool after synthesis.	Specify the place-and-route job you want to run for the specified implementation. See Place and Route Panel .
-job PR_job_name -option run_backannotation 1 0	<i>Synplify Premier</i> If enabled, synthesis backannotates placement and timing data after the appropriate vendor-specific place-and-route job is run.	Backannotate placement and timing data following Place & Route option; see Running P&R Automatically after Synthesis .
-job PR_job_name -option use_placement_constraints 1 0	<i>Synplify Premier</i> <i>Intel Stratix III, Stratix II GX, and Stratix II technologies</i> <ul style="list-style-type: none"> • By default, the Quartus II place-and-route tool uses the placement constraints forward annotated from the synthesis tool. • When this option is disabled (not checked), the Quartus II place-and-route tool determines how it will handle physical constraints. 	Use placement constraints from physical synthesis option; see Running P&R Automatically after Synthesis .
-job PR_job_name -option xtclsh_option_file options_filename.tcl	<i>Xilinx</i> This option allows you to specify the place-and-route options file to use when running the PAR job. You can use this command in the various Xilinx synthesis flows, including the incremental flows.	Add Place & Route Options File; see Add P&R Implementation Popup Menu Command . You can choose an Existing Options File or Create New Options File.

Option	Description	GUI Equivalent
-level_hierarchy 1 0	<i>Synplify Premier</i>	Create
-createhierarchy 1 0	Creates an additional level of hierarchy for always or process blocks for Synplify Premier netlist restructuring when the Enable prototyping tools option is enabled.	Always/Process Level Hierarchy check box, GCC & Prototyping Tools Panel .
-libext .libextName1 .libextName2 ...	Adds library extensions to Verilog library files included in your design for the project and searches the directory paths you specified that contain these Verilog library files. To use library extensions, see Using Library Extensions for Verilog Library Files, on page 78 in the <i>User Guide</i> .	Library Extensions (space separated) for each unique file extension, .
-library_path <i>directory_pathname</i>	For Verilog designs, specifies the paths to the directories which contain the library files to be included in your design for the project. Defines the search path used by the tool to include all the Verilog design files for your project. The argument <i>directory_pathname</i> is a string that specifies the directories where these included library files can be found. The software searches for all included Verilog files and the tool determines the top-level module. The names of files read from the library path must match module names. Mismatches result in error messages.	Library Directories on .
-local_tmr_rename 1 0	<i>Synplify Premier</i> When enabled in conjunction with <i>syn_radhardlevel=tmr</i> , the name of the original register is suffixed with _tmr1 to maintain consistency.	Not available in the UI

Option	Description	GUI Equivalent
-log_file <i>logFileName</i>	Allows you to change the name for the default log file. For example: <pre>set_option -log_file test generates the synlog\test_fpga_mapper.srr file in the Implementation Directory after synthesis is run.</pre>	Enter command from the Tcl window
looplimit <i>loopLimitValue</i>	Allows you to override the default compiler loop limit value of 2000 in the HDL. You can also apply loop limits using the Verilog <code>loop_limit</code> or the VHDL <code>syn_looplimit</code> directive. For details about these directives, see loop_limit, on page 79 and syn_looplimit, on page 373 in the <i>Attribute Reference</i> .	Loop Limit, and VHDL Panel .
-map_logic 1 0	<i>Intel FPGA and Lattice</i> Turns on direct mapping to technology-specific devices during synthesis.	Map Logic to ATOMs LCells Macrocells, Device Panel . Macrocells applies to Lattice; ATOMs and LCells apply to Intel FPGA.
-maxfan <i>value</i>	Sets the fanout limit for the current project. The limit is value is guideline for the tool rather than a hard limit.	Fanout Guide , Device Panel
-fanout_guide <i>value</i>		
-maxfanin 1 0	<i>Intel FPGA Max and Lattice ispMach</i>	Maximum Cell Fanin, Device Panel
-fanin_limit <i>value</i>	Specifies the maximum fan-in.	
max_par_explorer <i>parallelJobs</i>	<i>Synplify Premier</i> When exploratory place and route is enabled, this option lets you specify the maximum number of CPUs you can use to run place-and-route jobs in parallel. See Running Exploratory Place and Route, on page 822 .	Maximum par_explorer jobs, Device Panel

Option	Description	GUI Equivalent
max_parallel_jobs <i>n</i>	Lets you run multiprocessing with compile points. This allows the synthesis software to run multiple, independent compile point jobs simultaneously, providing additional runtime improvements for the compile point synthesis flow. For information on setting the maximum number of parallel synthesis jobs, see Setting Number of Parallel Jobs, on page 782 in the <i>User Guide</i> .	Maximum number of parallel mapper jobs, on the Configure Compile Point Process dialog box.
-maxterms <i>value</i>	<i>Lattice</i> Sets the maximum number of terminals per macrocell. This option is available only if -map_logic is set to true.	Maximum Terms/Macrocell, Device Panel
-max_terms_per_macrocel <i>value</i>		
-mif_files_dirs <i>mifDirectoryPath</i>	<i>Intel FPGA</i> Specifies the absolute directory path location for the missing MIF files. For example, specify: "U:/mif_tests" Note: Separate multiple directory entries with a semicolon.	MIF File Directories Device Panel
-multi_file_compilation_unit <i>1 0</i>	When you enable the Multiple File Compilation Unit switch, the Verilog compiler uses the compilation unit for modules defined in multiple files.	
-no_sequential_opt <i>1 0</i>	Enables or disables the sequential optimizations for the design. (Note that unused registers will still be removed from the design.) The default value is false (sequential optimizations are performed). When true, delay and area size might increase. Value can be 1 or true, 0 or false. With this option enabled, the FSM Compiler and FSM Explorer options are effectively disabled.	Device Panel

Option	Description	GUI Equivalent
-num_critical_paths <i>value</i>	Specifies the number of critical paths to report in the timing report.	Number of Critical Paths, Timing Report Panel
-num_startend_points <i>value</i>	Specifies the number of start and end points to include when reporting paths with the worst slack in the timing report.	Number of Start/End Points, Timing Report Panel . Number of Start/End Points, Timing Report Generation dialog box.
-opcond <i>value</i>	<i>Microchip Axcelerator, Fusion, IGLOO, and ProASIC technologies</i> Sets the operating condition for device performance in the areas of optimization, timing analysis, and timing reports. Values are Default, MIL-WC, IND-WC, COM-WC, and Automotive-WC. See Operating Condition Device Option, on page 811 for more information.	Device Panel
-optimize_ngc 1 0	<i>Synplify Premier</i> Enables NGC/EDIF optimization during fast synthesis strategy.	Optimize NGC/EDIF, Options Panel
-par_explorer 1 0	<i>Synplify Premier</i> When enabled, lets you run the current implementation in parallel with different place-and-route configurations for the design. Overall runtime can be reduced, as well, using the Synopsys CDPL (Common Distributed Processing Library) scheme for distributed processing. See Running Exploratory Place and Route, on page 822 .	Exploratory Place and Route, Device Panel
-par_use_xflow 1 0	<i>Xilinx</i> When enabled, you can use the previous xflow function to run the Xilinx place-and-route tool. This option is turned off by default.	Use Xilinx xflow, Device Panel

Option	Description	GUI Equivalent
-par_use_xilinx_partition 1 0	<p><i>Xilinx</i></p> <p>When enabled, you can use compile points with the Xilinx Partition flow to compare partitions or modules in the previous and current implementations. Partitions that have not changed are preserved from the previous implementation. This increases overall design stability by freezing the implementation for blocks that have met timing. This feature also saves significant time in that the implementation for the entire design does not need to be rerun. This option is turned off by default.</p> <p>You must also enable the Use Xilinx Xflow option (-par_use_xflow 1) as well.</p>	Use Xilinx Partition Flow, Device Panel
-pipe 1 0	Runs designs at a faster frequency by moving registers into the multiplier, creating pipeline stages.	Pipelining, Device Panel
-popfeed 1 0	<i>Synplify Premier</i>	Feedthrough Optimization, GCC & Prototyping Tools Panel
-feedthrough 1 0	<p>When enabled, the software eliminates unused routes through a module after it has been compiled. These optimizations are available when you floorplan the design.</p>	Feedthrough Optimization, GCC & Prototyping Tools Panel
-preserve_registers 1 0	<p><i>Microchip</i></p> <p>When enabled, the software uses less restrictive register optimizations during synthesis if area is not as great a concern for your device. The default for this option is disabled (0).</p>	Conservative Register Optimization switch on the Device Panel

Option	Description	GUI Equivalent
-project_relative_includes 1 0	Enables/disables the Verilog include statement to be relative to the project, rather than a verilog file. For projects built with software after 8.0, the include statement is no longer relative to the files but is relative to the project: project_relative (1). See Updating Verilog Include Paths in Older Project Files, on page 111 in the <i>User Guide</i> for information about updating older project files.	Include Path Order,
-report_path integer	<i>Microchip 500K, Fusion, IGLOO, and ProASIC technologies</i> Sets the maximum number of critical paths in a forward-annotated SDF constraint file	Max Number of Critical Paths in SDF, Device Panel
-reporting_reportType	Sets parameters for the stand-alone Timing Analyst report. See Timing Report Parameters for set_option , on page 178 for details.	Analysis->Timing Analyst command: Timing Report Generation Parameters
-resolve_multiple_driver 1 0	When a net is driven by a VCC or GND and active drivers, enable this option to connect the net to the VCC or GND driver. The default for this option is disabled (0). See Resolve Mixed Drivers Option , on page 182 for details.	Resolve Multiple Drivers, Device Panel
-resource_sharing 1 0	Enables or disables resource sharing globally. This is a compiler-specific optimization, and does not affect resource sharing in the mapper. To enable or disable individual modules, use the syn_sharing directive.	Resource Sharing, Device Panel

Option	Description	GUI Equivalent
-result_file <i>filename</i>	Specifies the name of the results file.	Result File Name and Result Format, Implementation Results Panel
-retiming 1 0	When enabled (1), registers may be moved into combinational logic to improve performance. The default value is 0 (disabled).	Retiming, Device Panel
-run_prop_extract 1 0	Enables/disables the annotation of certain generated properties relating to clocks and expansion onto the RTL view. This enables the Tcl expand and find commands to work correctly with clock properties.	Options Panel
-rw_check_on_ram 1 0	Enabling this option automatically inserts bypass logic when required to prevent simulation mismatch in read-during-write scenarios. For asynchronous clocks, the tool will not generate bypass logic which can cause unintended CDC paths between the clocks. For more information about using this option in conjunction with the <code>syn_ramstyle</code> attribute, see syn_ramstyle, on page 509 .	Automatic Read/Write Check Insertion for RAM, Device Panel
-safe_case 1 0	When enabled, the high reliability safe case option turns off sequential optimizations for counters, FSM, and sequential logic to increase the circuit's reliability.	Preserve and Decode Unreachable States (FSM, Counters, Sequential Logic), High Reliability Panel
-soft 1 0	<i>Intel FPGA</i>	Soft LCELLS, Device Panel
-soft_buffers 1 0	Makes non-critical Lcells soft in Intel FPGA MAX designs.	

Option	Description	GUI Equivalent
-support_implicit_init_netlist 1 0	<i>Synplify Premier</i> When enabled, implicit initial values for the technology primitives are automatically loaded and used during optimization (the advanced synthesis strategy must also be enabled). See Using Implicit Initial Values, on page 536 in the <i>User Guide</i> .	Implicit Initial value support for Instantiated primitives, Device Panel
support_multidim_vqm	<i>Intel FPGA</i> The FPGA mapper now supports multi-dimensional arrays and preserves the multi-dimensional array port structures in the synthesis output netlist (.vqm).	
-supporttypedfit 1 0	When enabled (1), the compiler passes init values through a syn_init property to the mapper. For more information, see VHDL Implicit Data-type Defaults, on page 334 .	Implicit Initial Value Support, VHDL Panel

Option	Description	GUI Equivalent
-symbolic_fsm_compiler 1 0	Enables/disables the FSM compiler. Controls the use of FSM synthesis for state machines. The default is false (FSM Compiler disabled). Value can be 1 or true, 0 or false.	FSM Compiler check box, Device Panel
-autosm 1 0	<p>When this option is true, the FSM Compiler automatically recognizes and optimizes state machines in the design. The FSM Compiler extracts the state machines as symbolic graphs, and then optimizes them by re-encoding the state representations and generating a better logic optimization starting point for the state machines.</p> <p>However, if you turn off sequential optimizations for the design, FSM Compiler and/or the <code>syn_state_machine</code> directive and <code>syn_encoding</code> attribute are effectively disabled.</p> <p>See -no_sequential_opt 1 0 for more information on turning off sequential optimizations.</p>	
-syn_altera_model on off clearbox_only	<p><i>Intel FPGA</i></p> <p>Sets Clearbox options, which determine how instantiated and inferred Intel Megafunctions are implemented for synthesis.</p> <p>See Intel FPGA Models Parameters for set_option , on page 180 for details.</p>	Intel FPGA Models option, Device Panel
-synthesis_onoff_pragma 1 0	Determines whether code between synthesis on/off directives is ignored. When enabled, the software ignores any VHDL code between <code>synthesis_on</code> and <code>synthesis_off</code> directives. It treats these third-party directives like <code>translate_on/off</code> directives (see translate_off/translate_on, on page 749 for details).	Synthesis on/off Implemented as Translate on/Off, VHDL Panel

Option	Description	GUI Equivalent
-synthesis_strategy advanced base fast routability	<p><i>Synplify Premier</i></p> <p>Specifies a synthesis strategy to use for the design. Synthesis modes include:</p> <ul style="list-style-type: none"> • advanced – Performs additional optimizations during logic synthesis to provide an output netlist with better QoR, but with longer runtimes compared to other modes. This is the default. • base – Generates Synplify Pro results. This is the same as turning off advanced and fast modes. • fast – Reduces the number of optimizations performed to improve the synthesis runtime. • routability – Performs placement-aware congestion reduction techniques to produce a routable netlist. There is a trade-off in timing QoR to improve synthesis runtime. 	Synthesis Strategy, Device Panel
-timequest 1 0	<p><i>Intel FPGA</i></p> <p>Determines whether to use the Intel Quartus II TimeQuest Timing Analyzer (1) or the Classic Timing Analyzer (0). This option is available only for Intel Stratix II and Stratix II GX devices.</p> <p>Note: The Quartus II TimeQuest Timing Analyzer is the default for Intel Stratix III, Stratix IV, Stratix IV GT, Arria GX, and Arria II GX devices.</p>	Use TimeQuest Timing Analyzer, Device Panel

Option	Description	GUI Equivalent
-top_module name	Specifies the top-level module. If the top-level entity does not use the default work library to compile the VHDL/Verilog files, you must specify the library file where the top-level entity can be found. To do this, the top-level entity name must be preceded by the VHDL/Verilog library followed by the dot (.).	Top-level Entity/Module, VHDL Panel or
-update_models_cp 1 0	Determines whether (1) or not (0) changes inside a compile point can cause the compile point (or top-level) containing it to change accordingly.	Update Compile Point Timing Data, Device Panel
-upf_iso_filter_elements_with_applies_to enable disable error	<p><i>Synplify Premier</i></p> <p>Use the <code>upf_iso_filter_elements_with_applies_to</code> Tcl variable with the <code>set_isolation</code> command to control filtering of isolation elements when the <code>-applies_to</code> option is specified with the <code>-elements</code> option for the power domain. You can specify:</p> <ul style="list-style-type: none"> • disable – Ignores the <code>-applies_to</code> option and applies the isolation strategy to all the elements specified with the <code>-elements</code> option. • enable – Filters the isolation elements (ports/pins) based on the value specified with the <code>-applies_to</code> option. This is the default. • error – Generates an error message. 	Enter UPF command from the Tcl window
-upf_verbose 1 0	<p><i>Synplify Premier</i></p> <p>Generates a detailed message for every command that is parsed and instrumented in the UPF file.</p> <p>By default, this option is disabled (<code>set_option -upf_verbose 0</code>).</p>	Enter UPF command from the Tcl window

Option	Description	GUI Equivalent
-use_fsm_explorer 1 0	Enables/disables the FSM Explorer.	FSM Explorer, Device Panel
-use_new_altera_mapper 1 0	<i>Intel Arria 10, Arria V GZ, Stratix V</i> New version of the Intel FPGA mapper supports the latest Intel FPGA devices and provides performance and runtime improvements for the design. Both versions of the Intel FPGA mappers will coexist.	Intel FPGA New Mapper, Options Panel
-use_vivado 1 0	<i>Xilinx UltraScale and 7 Series (Virtex-7, Kintex-7, and Artix-7)</i> It is recommended that you use the Vivado Integrated Design environment (IDE) tool to run place and route for these devices. You must enable this option. Disable this option, to run Xilinx ISE place and route.	Use Vivado for Place and Route Implementation Results Panel
-user_physical_separation 1 0	Enable or disable physical separation of the TMR modules using the Synplify Premier Design Planner.	
validate_mif_files 1 0	<i>Intel FPGA</i> Checks that MIF files exist for associated Intel FPGA MegaWizard generated ROM or RAM instances. If MIF files are missing, the synthesis tool generates an error.	Validate MIF Files Options Panel
-vlog_std v2001 v95 sysv	The default Verilog standard for new projects is SystemVerilog. Turning off both options in the Verilog panel defaults to v95.	Verilog 2001, SystemVerilog,
-voltage none 0.9V 1.1V	<i>Intel FPGA</i> For Stratix III devices that support speed grade 4, you can specify a core voltage value.	Core Voltage, Device Panel

Option	Description	GUI Equivalent
-write_apr_constraint 1 0	Writes vendor-specific constraint files.	Write Vendor Constraint File, Implementation Results Panel
-write_declared_clocks_only 1 0	<i>Lattice</i> Forward-annotates declared clocks only to the LPF file. By default, this option is enabled. See Write Declared Clocks Only, on page 713 , for details.	Write Declared Clocks Only, Device Panel
-write_verilog 1 0	Writes Verilog or VHDL mapped netlists.	Write Mapped Verilog/VHDL Netlist,
-write_vhdl 1 0		Implementation Results Panel

Timing Report Parameters for `set_option`

The following lists the parameters for the stand-alone timing report (.ta file).

<code>async_clock</code>	<code>margin</code>
<code>ctd</code>	<code>netlist</code>
<code>filename</code>	<code>number_paths</code>
<code>filter</code>	<code>output_srm</code>
<code>gen_output_srm</code>	

Reporting Option	Description
-reporting_async_clock	Generates a report for paths that cross between clock groups using the stand-alone Timing Analyst.
-reporting_ctd slack end_point off	<p>Controls how the <i>design_ctd.txt</i> (correlation timing dump) file is generated when the Timing Analyst is run. You can specify one of the following values:</p> <ul style="list-style-type: none"> • slack - The timing information in the ctd file is sorted by slack. This is the default. • end_point - The timing information in the ctd file is sorted by end points. • off - Turns off generating the ctd file. <p>The ctd file contains a timing summary of the design that is used by the Timing Report View to display and analyze the synthesis timing for the design and correlate this synthesis timing with the P&R timing in the GUI.</p>
-reporting_filename <i>filename.ta</i>	Specifies the standard timing report file (ta) generated from the stand-alone Timing Analyst.
-reporting_filter <i>filter options</i>	<p>Generates the standard timing report based on the filter options you specify for paths, such as:</p> <ul style="list-style-type: none"> • From points • Through points • To points <p>For more information, see:</p> <ul style="list-style-type: none"> • Timing Report Generation Parameters , on page 530. • Combining Path Filters for the Timing Analyzer , on page 534 • Timing Analyzer Through Points , on page 533. • Specifying From, To, and Through Points , on page 330.

Reporting Option	Description
-reporting_gen_output_srm 1 0	Specifies the new name of the output SRM File when you change the default name. If this option is set to 1, this new name is used for the output <code>srm</code> file after you run the stand-alone Timing Analyst.
-reporting_margin slackValue	You can specify a slack margin to obtain a range of paths within the worst slack time for the design after you run the stand-alone Timing Analyst.
-reporting_netlist filename.srm	Specifies the associated gate-level netlist file (<code>srm</code>) generated from the stand-alone Timing Analyst.
-reporting_number_path numberOfPaths	You can specify the number of critical paths to report after you run the stand-alone Timing Analyst.
-reporting_output_srm 1 0	Allows you to change the name of the output <code>srm</code> file. If you enable the output SRM File option, you can change this default name.

For GUI equivalent switches for these parameters, see [Timing Report Generation Parameters, on page 530](#).

Intel FPGA Models Parameters for `set_option`

Sets Clearbox options, which determines how you want instantiated and inferred Intel FPGA Megafunctions implemented for synthesis.

Syntax

```
set_option -syn_altera_model [-on|-off] [-clearbox_only]
```

-on	Specifies that the synthesis software calls Clearbox functions for instantiated and inferred Intel FPGA Megafunctions. For example, the <code>altsyncram</code> megafunction is replaced with <code>ram_block</code> primitives.
-off	Specifies that the synthesis software does not call Clearbox functions for instantiated and inferred Intel FPGA Megafunctions. Therefore, the <code>altsyncram</code> megafunction is <i>not</i> replaced with <code>ram_block</code> primitives.
-clearbox_only	Specifies that the synthesis software only calls the Clearbox function for inferred Intel FPGA Megafunctions.

help for set_option

This option is useful for getting syntax help on the various implementation options used for compiling and mapping a design, especially since this list of options keeps growing.

Syntax

```
% set_option -help
```

Usage:

```
set_option optionName optionValue [-help [value]]
```

Where:

- *optionName*—specifies the option name.
- *optionValue*—specifies the option value.
- `-help [value]`—to get help on options. Use:
 - `-help *` for the list of options
 - `-help optionName` for a description of the option

Examples

To list all option commands in the Tcl window:

```
set_option -help *
```

To list all option commands beginning with the letters fi in the Tcl window:

```
% set_option -help fi*
fixgatedclocks
fixgeneratedclocks
fixsmult
```

To get help on a specific option in the Tcl window:

```
% set_option -help fixgatedclocks
0: Don't fix; 1: fix, no report; 2: fix, report exception
registers; 3: fix, report all registers
```

Use the following Tcl commands to print a description of the options:

```
% set_option -help c*
% set hl [set_option -help c*]
% puts $hl
% foreach option $hl { puts "$option:\t [set_option -help
$option]"; }
```

This example will print a list of set_option options that begin with the letter c.

Resolve Mixed Drivers Option

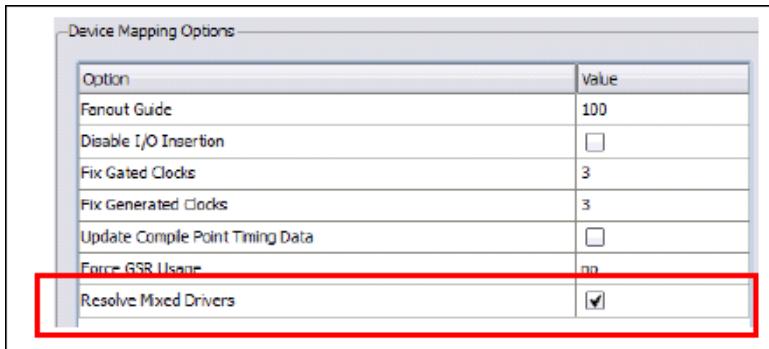
Use the Resolve Mixed Drivers option when mapping errors are generated for input nets with mixed drivers. You might encounter the following messages in the log file:

```
@A:BN313 | Found mixed driver on pin pin:data_out inst:dpram_lut3
of work.dpram(verilog), use option "Resolve Mixed Drivers" in
"Device" tab of "Implementation Options" to automatically resolve
this
@E:BN314 | Net "GND" in work.test(verilog) has mixed drivers

@A:BN313 | Found mixed driver on pin pin:Q[0] inst:dff1.q of
PrimLib.sdffr(prim), use option "Resolve Mixed Drivers" in
"Device" tab of "Implementation Options" to automatically resolve
this
@E:BN314 | Net "VCC" in work.test(rtl) has mixed drivers
```

Whenever a constant net (GND or VCC) and an active net are driving the same output net, enable the Resolve Mixed Drivers option so that synthesis can proceed. To set this switch:

- Check **Resolve Mixed Drivers** on the Device tab of the Implementation Options panel.



- Use the Tcl command, `set_option -resolve_multiple_driver 1`.

By default this option is disabled and set to:

```
set_option -resolve_multiple_driver 0.
```

When you rerun synthesis, you should now see messages like the following in the log file:

```
@W:BN312 | Resolving mixed driver on net GND, connecting output
pin:data_out inst:dpram_lut3 of work.dpram(verilog) to GND
@N:BN116 | Removing sequential instance dpram_lut3.dout of
view:PrimLib.dffe(prim) because there are no references to its
outputs
@N:BN116 | Removing sequential instance dpram_lut3.mem of
view:PrimLib.raml(prim) because there are no references to its
outputs

@W:BN312 | Resolving mixed driver on net VCC, connecting output
pin:Q[0] inst:dff1.q of PrimLib.sdffr(prim) to VCC
@N:BN116 | Removing sequential instance dff1.q of
view:PrimLib.sdffr(prim) because there are no references to its
outputs
```

Example - Active Net and Constant GND Driving Output Net (Verilog)

```
module test(clk,data_in,data_out,radd,wradd,wr,rd);
    input clk,wr,rd;
    input data_in;
```

```
input [5:0]radd,wradd;
output data_out;
// component instantiation for shift register module
shrl srl_lut0 (
    .clk(clk),
    .sren(wr),
    .srin(data_in),
    .srout(data_out)
);
// Instantiation for ram
dpram dpram_lut3 (
    .clk(clk),
    .data_in(data_in),
    .data_out(data_out),
    .radd(radd),
    .wradd(wradd),
    .wr(wr),
    .rd(rd)
);

endmodule

module shrl (clk,sren,srin,srout);
input clk;
input sren;
input srin;
output srout;
```

```
parameter width = 32;
reg [width-1:0] sr;

always@ (posedge clk)
begin
    if (sren == 1)
        begin
            sr <= {sr[width-2:0], srin};
        end
    end
// Constant net driving

// the output net
assign srou = 1'b0;
endmodule

module dpram(clk,data_in,data_out,radd,wradd,wr,rd);
input clk,wr,rd;
input data_in;
input [5:0]radd,wradd;
output data_out;

reg dout;
reg [0:0]mem[63 :0];

always @ (posedge clk)
```

```
begin
    if(wr)
        mem[wradd] <= data_in;
    end

    always @ (posedge clk)
    begin
        if(rd)
            dout <= mem[radd];
        end

        assign data_out = dout;

    endmodule

module test(clk,data_in,data_out,radd,wradd,wr,rd);
    input clk,wr,rd;
    input data_in;
    input [5:0]radd,wradd;
    output data_out;
    // component instantiation for shift register module
    shrl_srl_lut0 (
        .clk(clk),
        .sren(wr),
        .srin(data_in),
        .srout(data_out)
    );
    // Instantiation for ram
    dpram dpram_lut3 (
        .clk(clk),
        .data_in(data_in),
        .data_out(data_out),
        .radd(radd),
        .wradd(wradd),
        .wr(wr),
        .rd(rd)
    );

```

```
endmodule

module shrl (clk,sren,srin,srout);
    input clk;
    input sren;
    input srin;
    output srout;

    parameter width = 32;
    reg [width-1:0] sr;

    always@(posedge clk)
    begin
        if (sren == 1)
        begin
            sr <= {sr[width-2:0], srin};
        end
    end
    // Constant net driving

    // the output net
    assign srout = 1'b0;
endmodule

module dpram(clk,data_in,data_out,radd,wradd,wr,rd);
    input clk,wr,rd;
    input data_in;
    input [5:0]radd,wradd;
    output data_out;

    reg dout;
    reg [0:0]mem[63 :0];

    always @ (posedge clk)
    begin
        if(wr)
            mem[wradd] <= data_in;
    end

    always @ (posedge clk)
    begin
        if(rd)
            dout <= mem[radd];
```

```

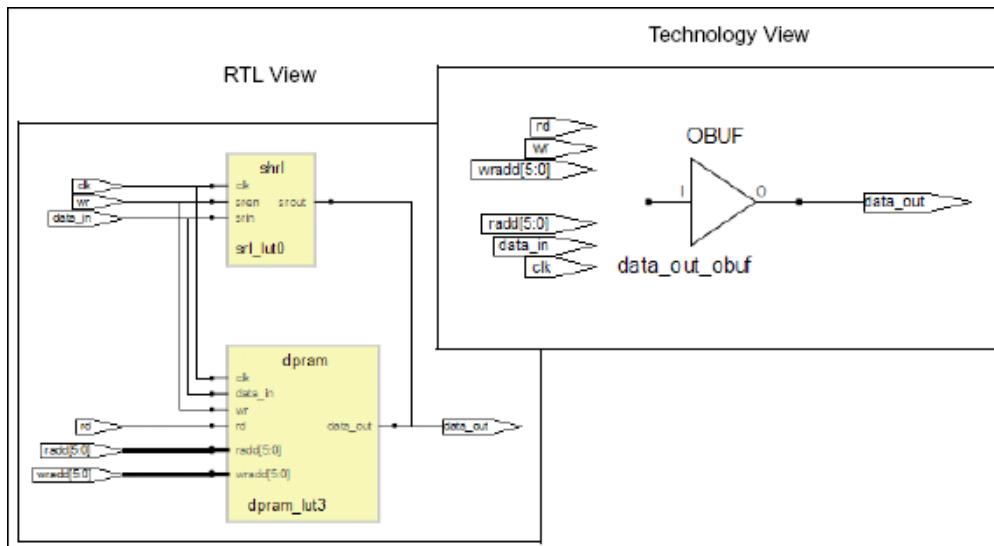
    end

    assign  data_out = dout;

endmodule

```

See the following RTL and Technology views; the Technology view shows the constant net tied to the output.



Example - Active Net and Constant VCC Driving Output Net (VHDL)

```

library ieee;
use ieee.std_logic_1164.all;

entity test is
port (
    clk,rst      : in std_logic;
    sr_en        : in std_logic;
    data          : in std_logic;
    data_op       : out std_logic
);

```

```
);  
end entity test;  
  
architecture rtl of test is  
component shrl  
generic ( sr_length : natural );  
port (  
    clk : in std_logic;  
    sr_en : in std_logic;  
    sr_ip : in std_logic;  
    sr_op : out std_logic  
);  
end component shrl;  
  
component d_ff  
port (  
    data, clk, rst : in std_logic;  
    q : out std_logic  
);  
end component d_ff;  
  
begin  
-- instantiation of shift register  
shift_register : shrl  
generic map( sr_length => 64 )  
port map (  
    clk => clk,  
    ...  
);  
end;
```

```
        sr_en      => sr_en,
        sr_ip      => data,
        sr_op      => data_op  );

-- instantiation of flipflop
dff1 : d_ff
port map
    data => data,
    clk => clk,
    rst => rst,
    q => data_op);

end rtl;

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity shr1 is
generic (  sr_length      : natural      );
port (
    clk          : in  std_logic;
    sr_en       : in  std_logic;
    sr_ip       : in  std_logic;
    sr_op       : out std_logic
);
end entity shr1;
```

```
architecture rtl of shrl is
    signal sr_reg      : std_logic_vector(sr_length-1 downto 0) ;
begin
    shreg_lut: process (clk)
begin
    if rising_edge(clk) then
        if sr_en = '1' then
            sr_reg <= sr_reg(sr_length-2 downto 0) & sr_ip;
        end if;
    end if;
end process shreg_lut;

-- Constant net driving output net
sr_op <= '1';

end architecture rtl;
library IEEE;
use IEEE.std_logic_1164.all;

entity d_ff is
    port
        data, clk, rst : in std_logic;
        q : out std_logic);
    end d_ff;

architecture behav of d_ff is
begin
```

```
FF1:process (clk) begin
    if (clk'event and clk = '1') then
        if (rst = '1') then
            q <= '0';
        else q <= data;
        end if;
    end if;
end process FF1;

end behav;

library ieee;
use ieee.std_logic_1164.all;

entity test is
port (clk,rst : in std_logic;
      sr_en : in std_logic;
      data : in std_logic;
      data_op : out std_logic );
end entity test;

architecture rtl of test is
component shrl
generic (sr_length : natural);
port (clk : in std_logic;
      sr_en : in std_logic;
      sr_ip : in std_logic;
      sr_op : out std_logic );
end component shrl;

component d_ff
port (data, clk, rst : in std_logic;
      q : out std_logic );
end component d_ff;

begin
-- instantiation of shift register
shift_register : shrl
generic map (sr_length => 64)
port map (clk => clk,
          sr_en => sr_en,
```

```
        sr_ip => data,
        sr_op => data_op );
-- instantiation of flipflop
dff1 : d_ff
port map (data => data,
           clk => clk,
           rst => rst,
           q => data_op );
end rtl;

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity shrl is
generic (sr_length : natural);
port (clk : in std_logic;
      sr_en : in std_logic;
      sr_ip : in std_logic;
      sr_op : out std_logic );
end entity shrl;

architecture rtl of shrl is
signal sr_reg : std_logic_vector(sr_length-1 downto 0);
begin
    shreg_lut: process (clk)
begin
    if rising_edge(clk) then
        if sr_en = '1' then
            sr_reg <= sr_reg(sr_length-2 downto 0) & sr_ip;
        end if;
    end if;
    end process shreg_lut;
-- Constant net driving output net
sr_op <= '1';
end architecture rtl;

library IEEE;
use IEEE.std_logic_1164.all;

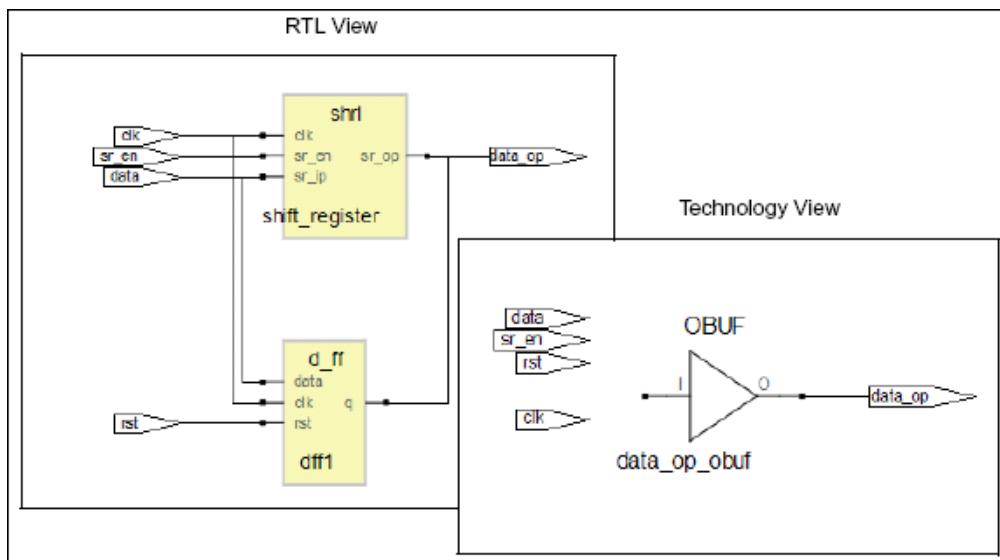
entity d_ff is
port (data, clk, rst : in std_logic;
      q : out std_logic );
end d_ff;

architecture behav of d_ff is
begin
    FF1:process (clk) begin

```

```
if (clk'event and clk = '1') then
  if (rst = '1') then
    q <= '0';
  else q <= data;
  end if;
end if;
end process FF1;
end behav;
```

See the following RTL and Technology views; the Technology view shows the constant net tied to the output.



slice_primitive

Synplify Premier

The slice_primitive command defines where a primitive is logically divided by identifying groups of output signals by number, explicitly by name, or by the number of instances. You can place this command in an .nrf or a .tcl file.

Syntax

```
slice_primitive [ -push ] [ -nl ] [ -numslices ] { primitiveInstanceName }
{ { number } | { bitGroup1 } { bitGroup2 } ... { bitGroupN } }
```

The following table describes the command options.

Option	Description
-push	(Optional) Creates a new level of hierarchy (primitive division is shown at the next level down in the hierarchy). This argument is equivalent to enabling Preserve the Hierarchical View on the Bit Slices panel.
-nl	(Optional) Allows multiple instances of the sliced element to be used in certain specific cases. This argument is equivalent to enabling Slice all instances of this type on the Bit Slices panel.
-numslices	(Optional) Divides the primitive into the specified number of instances.
<i>primitiveInstanceName</i>	Name of the output signal to slice into subelements. The HDL bus port name syntax is used for bus signals (for example, big_bus[63:0]). The output signal name is enclosed in curly braces ({}).
<i>number</i>	The number of bits to assign to each primitive, or when the -numslices option is included, the number of sliced instances. Bits are assigned sequentially beginning with bit 0; any remaining bits are assigned to the next primitive. The <i>number</i> entry is enclosed in curly braces ({}).
<i>bitGroup1</i> <i>bitGroup2</i> ... <i>bitGroupN</i>	Explicitly specifies which bits are assigned to individual primitives. This format requires that you specify every bit. Each bit group entry is enclosed in curly braces ({}) and is separated from the next group by a space.

When manually creating entries in an .nrf or a .tcl file, if the length of the slice_primitive command exceeds one line, the command can be continued on subsequent lines by including a line continuation character (\) as the last character of each line to be continued.

sopc2syn

Intel FPGA

Lets you include subsystems created with the SOPC Builder in your design. It is the command line equivalent of Import IP->Import Intel FPGA SOPC Project.

The sopc2syn utility reads an SOPC Builder project file (.ptf) and script file (scr), and uses information extracted from them as well as other SOPC Builder files to generate a synthesis project file and add definition and constraint files to it for the SOPC subsystem.

For additional information about this utility, see [sopc2syn Conversion](#), on page 158 in the *Reference Manual*.

Syntax

```
sopc2syn  
-ptf sopcProjectFileName  
-clearbox 1|0  
-hdl_lang vhdl|verilog  
-force_wbox fileName|string  
-wrap 1|0  
-log fileName  
-quiet  
-noexit  
-verbose 0|1|2|3  
-help
```

-ptf <i>sopcProjectFileName</i>	Specifies the name of the SOPC Builder project (ptf) file.
-clearbox 1 0	<p>This is an advanced option that uses the Synplify Clearbox flow for encrypted cores.</p> <ul style="list-style-type: none"> • 0 turns the Clearbox flow off. This adds black box wrapper files for encrypted cores to the project and add the core files to PAR directory. • 1 turns the Clearbox flow on. This adds black box wrapper files for encrypted cores to the project. It also adds the core HDL files to the project, but only uses them for P&R. This is the recommended setting and the default. <p>See SOPC2syn Black Boxes , on page 200, Encrypted HDL Files for SOPC2syn Black Boxes , on page 199, and SOPC2syn White Boxes , on page 200. SOPC2syn White Boxes , on page 200</p>
-hdl_lang vhdl verilog	This basic option sets the preferred HDL language in case of duplicate files. Verilog is the default.
-force_wbox <i>fileName string</i>	<p>This is an advanced option that forces listed cores to be treated as white boxes. You can specifies the cores in a file (<i>fileName</i>), or in a comma-separated string of core names (<i>string</i>).</p> <p>The -clearbox value determines how the core is processed.</p> <ul style="list-style-type: none"> • If -clearbox is 1, the white box goes through the Clearbox flow. • If -clearbox is 0, the white box does not go through the Clearbox flow, and is treated as a black box. <p>See SOPC2syn Black Boxes , on page 200 and SOPC2syn White Boxes , on page 200.</p>
-wrap 1 0	<p>This is an advanced option that enables or disables the generation of missing core wrappers.</p> <ul style="list-style-type: none"> • 1 generates missing core wrappers. • 0 does not generate missing core wrappers. This is the recommended setting and the default.

-log <i>logfileName</i>	This is an advanced option that converts the PTF project file that writes out all the messages to the logfile. If you specify -log <i>logfileName</i> , the utility writes out all the messages to the specified file. STDOUT prints important messages such as the version, command options, error messages, and summary information on STDOUT and redirects all STDOUT messages to the log file. If you specify the -quiet option, the utility does not write any messages to STDOUT.
-quiet	This is an advanced option that suppresses messages from being written to the STDOUT.
-noexit	This is a debug option that writes out the project file even if there are errors in translation. See SOPC2syn Debug Mode , on page 200 .
-verbose 0 1 2 3	This is a debug option that controls the messages reported in the log file during debug or normal runs. <ul style="list-style-type: none"> • 0 writes out the important messages. This is the default. • 1 reports all the messages reported by -verbose 0, and also reports all the files read during translation. • 2 reports all messages reported by -verbose 1, as well as the lines processed from each of the files read during translation and the number of lines processed. • 3 reports all messages reported by -verbose 2, as well as more details on the state of run. See SOPC2syn Debug Mode , on page 200 .
-help	This basic option prints the help.

Encrypted HDL Files for SOPC2syn Black Boxes

If the ptf file points to an encrypted IP core definition file, one of the following happens:

- If you specified **-clearbox 1**, the utility adds black box wrapper files for encrypted cores to the projects. It also adds the HDL files for the encrypted cores to the project, but only uses them for P&R. It issues a warning message for encrypted cores with a black box.
- If you specified **-clearbox 0**, the utility adds black box wrapper files for encrypted cores to the project and copies the encrypted core files to the P&R directory.

SOPC2syn Black Boxes

If you want to treat a core as a black box during synthesis, specify the `-force_wbox` option with a list of cores to be black-boxed, and set `-clearbox 0`.

With these settings, the tool copies the core wrapper file to the Synplify folder and edits it to add the black box attribute. The component is treated as a black box during synthesis. If the tool does not find the named component, it issues a warning message.

SOPC2syn White Boxes

If you want a core to be a white box during synthesis, specify the `-force_wbox` option with list of cores to be white-boxed, and set `-clearbox 1`.

With these settings, the tool goes through the Clearbox flow, and treats the core as a white box during synthesis. If it does not find the named component or the Clearbox file, it issues a warning message.

SOPC2syn Debug Mode

You can use the following options to debug translation issues:

- Specify the `-noexit` option if you want the utility to write out the project file, even if there is an error.
- Use the `-verbose` option to better control the reporting of messages. All messages reported by these options are written to the `sopc2syn.log` file.

Example

```
installDirectory/bin/sopc2syn -ptf sopc_builder_prj_dir/system.ptf  
-log sopc_builder_prj_dir/system.log -verbose 3
```

spy_glass

Launches the SpyGlass® lint tool from the Synplify Premier products. This tool only runs on Linux.

Syntax

```
spy_glass [-generate] [-goals {value}] [-install {value}] [-methodology {value}] [-run]
           [-run_gui] [-run_dir {value}]
```

-generate

Generates the SpyGlass project scripts.

-goals {value}

Specifies the list of goals or analysis types.

-install {value}

Specifies the SpyGlass installation directory.

-methodology {value}

Specifies the methodology, which can contain a sub-methodology or collection of goals.

-run

Runs the SpyGlass tool in batch mode, which is the default.

-run_gui

Runs the SpyGlass GUI.

-run_dir {value}

Specifies the path to the generated SpyGlass project and scripts, and generate SpyGlass reports. <implementationDir>/spyglass is the default directory.

Description

SpyGlass linting lets you run static analysis to identify functional, structural, and coding consistency issues for the top-level RTL early in the design cycle. For details, see [Using the SpyGlass Tool for RTL Analysis, on page 93](#).

status_report

Writes out the results of reports displayed in the Project Status view after synthesizing a design.

Syntax

```
status_report -name reportName [-parameter reportSectionName]
  [-csv] [-output_file fileName] [-msgtype msgStatus] [-status] [-help]
```

Examples

```
status_report -name area_report

status_report -name timing_report -csv -output_file reports

status_report -name area_report -parameter io_port

status_report -name run_status -msgtype warnings

status_report -name timing_report -help
```

Option	Description
-name reportName	The type of report to access. Use any of the following keywords for <i>reportName</i> : <ul style="list-style-type: none"> • area_report • timing_report • opt_report • cp_report • hier_area_report • run_status
-parameter reportSectionName	Specifies a specific section of the area, timing, or message reports to access. See Parameters , on page 203 for details of the appropriate keywords to use for the section names.
-csv	Generates the report as a comma-separated list.
-output_file fileName	Specifies the name of the file that writes out the report. If you do not specify an output file, the report is displayed in the Tcl window.

Option	Description
-msgtype msgStatus	Generates the number of messages found for the following types of messages: <ul style="list-style-type: none">• Errors• Warnings• Notes
-status	Generates the status for a job. The results of the job can be specified with one of the following conditions: <ul style="list-style-type: none">• Completed• Failed
-help	Allows you to get help on a parameter list. Use <code>-help *</code> for a list of parameters.

Parameters

For the area, timing, and message reports you can report results for specific sections by specifying the appropriate keywords for the -parameter argument.

Area Report section keywords for -parameter	io_port non_io_reg total_io_reg v_ram dsp_used total_luts
Timing Report section keywords for -parameter	clock_name req_freq est_freq slack
Run Status section keywords for -parameter	compiler premap fpga_mapper

For example:

```
% status_report -name area_report
I/O ports(io_port) 26
Non I/O Register bits(non_io_reg) 242 (0%)
I/O Register bits(total_io_reg) 24
Block Rams(v_ram) 0 (1030)
DSP48s(dsp_used) 1 (2800)
LUTs(total_luts) 310 (0%)
```

Additional Reporting Commands

There are other commands available from the command line to report commonly-required information: `report_timing_summary`, `report_area`, and `report_opt`.

- Report Timing Summary

```
% report_timing_summary
Timing Summary
Clock Name Req Freq Est Freq Slack
eight_bit_uc|clock 198.9 MHz 169.1 MHz-0.887
```

- Report Area

```
% report_area
LUTs for combinational functions 0
Non I/O Registers 0
I/O Pins 66
I/O registers 0
DSP Blocks 0 (256)
Memory Bits 32768
```

- Report Optimizations

```
% report_opt
Combined Clock Conversion 1 / 0
```

Messages Reporting Commands

Here are examples of commands available from the command line to report message information using the option: run_status.

```
%status_report -name run_status
{compiler {notes "8"}{warnings "0"}{errors "0"}
 {job_status "Completed"}}
{fpga_mapper {notes "46"}{warnings "1"}{errors "0"}
 {job_status "Completed"}}
{premap {notes "3"}{warnings "2"}{errors "0"}
 {job_status "Completed"}}

% status_report - name run_status -msgtype warnings
{compiler {warnings "0"}}
{fpga_mapper {warnings "10"}}
{premap {warnings "0"}}

% status_report - name run_status -parameter {compiler premap}
-msgtype warnings
{compiler {warnings "0"}}
{premap {warnings "0"}}

% status_report -name run_status -parameter compiler -status
{compiler {job_status "Completed"}}

% status_report -name run_status -parameter premap -status
```

sub_impl

Sub-implementation editing command.

Syntax

```
sub_impl
  implName -add jobType
  implName -remove
  implName -type
  implName -run mode
  implName -cancel
  implName -option optionName [optionValue]
  -list
```

Arguments and Options

Option	Description
<i>implName -add jobType</i>	Creates a new sub-implementation that belongs to an active implementation.
<i>implName -remove</i>	Removes an active sub-implementation.
<i>implName -type</i>	Lists a sub-implementation type.
<i>implName -run mode</i>	Runs sub-implementation.
<i>implName -cancel</i>	Cancels a running sub-implementation.
<i>implName -option optionName [optionValue]</i>	Sets option for sub-implementation.
-list	Lists all sub-implementations for an active implementation.

syn_connect

Intel FPGA, Lattice, Microchip, Xilinx

Specifies the connectivity between the module/instance being monitored for the error with the error monitoring IP port or top-level port for the error monitoring module. It is used with the `syn_create_err_net` command and the `syn_radhardlevel`, `syn_ramstyle` or `syn_fsm_correction` attributes to probe the error flag in high-reliability designs.

Syntax

syn_connect

-from {n:netName}
-to {n:existingNetpath | t:inputPinEMIPpath | p:topErrorport}

-from {i:netName}

Specifies the new net name for the error flag connection used with the `syn_create_err_net` command. See [syn_create_err_net, on page 208](#).

-to
{n:existingNetpath |
t:inputPinEMIPpath |
p:topErrorport}

Specifies the path to an existing net, the input pin for the Error Monitoring IP (EMIP), or the top-level error port.

Example

```
syn_connect {-from {n:dwc_err0_net} -to {n:emp[0]}}
```

For Microchip examples, see [RAM Inference in ECC Mode, on page 786](#).

Limitation

Currently, when using the `-to` argument with the `t:` | `p:` option, an existing net must be present that feeds into the ports.

syn_create_err_net

Intel FPGA, Lattice, Microchip, Xilinx

Specifies the connectivity between the module/instance being monitored for the error with the error monitoring IP port or top-level port for the error monitoring module. It is used with the syn_connect command, and the syn_raddr, syn_ramstyle or syn_fsm_correction attributes to probe the error flag in high-reliability designs.

This is the syntax:

```
sync_create_err_net -name {netName}
  -inst {i:highRelInst}
  -err_pipe_num {numPipelineStages}
  -err_clk {n:clkSourceForPipelineStages}
  -err_reset {n:resetSourceForPipelineStages}
  -err_set {n:setSourceForPipelineStages}
  -err_enable {n:enableSourceForPipelineStages}
  -err_synch {synchValue}
  -single_bit | -double_bit
```

<i>netName</i>	Specifies the new net name for the error flag connection.
-inst <i>highRelInst</i>	Specifies the high reliability instance being monitored for the error.
-err_pipe_num <i>{numPipelineStages}</i>	Specifies the number of pipeline stages using the retiming property. The default value is 0. Note: If err_pipe_num=N, then the property syn_allow_retimimg=0 is set on the Nth stage of the pipeline register and syn_allow_retimimg=1 is set on the first (N-1)th stage of the pipeline registers by the synthesis tool. Where N>0 and err_clk is specified.
-err_clk <i>{n:clkSourceforPipelineStages}</i>	Specifies the hierarchical path to the input pin for the clock. Note: You must define the -err_clk for the pipeline stages if you specify -err_pipe_num. Otherwise, the pipeline stages to stabilize the error signals are not added.
-err_reset <i>{n:resetSourceforPipelineStages}</i>	Specifies the hierarchical path to the input pin for the reset signal.

-err_set {n:setSourceforPipelineStages}	Specifies the hierarchical path to the input pin for the set signal.
-err_enable {n:enableSourceforPipelineStages}	Specifies the hierarchical path to the input pin for the enable signal.
-err_synch {synchValue}	Specifies the boolean value for synchronous or asynchronous set/reset. The default is TRUE.
-single_bit -double_bit	Specifies monitoring of single-bit or double-bit error flags. This argument is used only with hamming3_ded or hamming3_ded_recovery values of syn_fsm_correction attribute.

Example

```
syn_create_err_net {-name {dwc_err0_net} -inst {i:inst1}
    -err_pipe_num {2} -err_clk {n:inst1.clk}
    -err_reset {n:inst1.rst} -err_synch {TRUE}}
```

For Microchip examples, see [RAM Inference in ECC Mode, on page 786](#).

synplify, synplify_pro, synplify_premier, synplify_premier_dp

Starts the FPGA synthesis tool and runs synthesis from the command line. Use the appropriate command for the tool you are using. The command to start the synthesis tool from the command line includes a number of command line options.

Syntax

```
synplify | synplify_pro | synplify_premier | synplify_premier_dp
[options ...]
[projectFile]
```

projectFile Specifies the project (prj) file to use. If no file is specified, the tool defaults to the last project file opened.

options Any of the command line options described in the next table. These options control tool action on startup and, in many cases, can be combined on the same command line. See the next table for a description of the *options* you can specify.

The following table describes the *options* you can specify:

Option	Description
-batch	<i>Synplify Pro (except node-locked), Synplify Premier</i> Starts the synthesis tool in batch mode from the specified project or Tcl file without opening the Project window.
-compile	Compiles the project, but does not map it.
-evalhostid	Reports host ID for node-locked and floating licenses.
-help	Lists available command line options and descriptions.
-history <i>filename</i>	Records all Tcl commands and writes them to the specified history log file when the command exits.
-Identify_dir <i>dir</i>	Specifies the location of the Identify installation directory for launching the Identify tool set. The installation path specified appears in the Configure Identify Launch dialog box (Options->Configure Identify Launch).
-impl <i>impName</i>	Runs only the specified implementation. You can use this option in conjunction with the -batch keyword. The Synplify tool supports only a single implementation.

Option	Description
-ip_license_wait <i>waitTime</i>	<p>Specifies how long to wait for a Synopsys DesignWare IP license when one is not immediately available. If you do not specify the -ip_license_wait option, license queuing is not enabled.</p> <p>If all requested licenses are checked out or if the specified wait time elapses, the tool excludes the IP and continues to process the rest of the design. Any IP block without a license is treated either as an error or a black box.</p> <p>License queuing allows you to wait until a license becomes available or specify a wait time in seconds. You can use this option in conjunction with the -batch keyword. For details, see Queuing Licenses, on page 777 in the <i>User Guide</i>.</p> <p>The <i>waitTime</i> value determines license queuing and sets a maximum wait time:</p> <ul style="list-style-type: none"> • Undefined or 0 = Queuing off • 1 = Queuing enabled, indefinite wait time • >1 = Queuing enabled for the specified time
-license_release	<p>Releases FPGA synthesis licenses for a session after the place-and-route job is launched. The software allows place and route to continue running even after exiting the synthesis tool so that it does not consume an FPGA license. This command option must be run in batch mode. Specify the following command:</p> <pre data-bbox="470 976 940 1010">toolName -batch -license_release</pre> <p>For details, see Releasing the Synthesis License During Place and Route, on page 1104.</p>
-licensetype <i>featureName</i>	<p>Specifies a license if you work in an environment with multiple Synopsys FPGA licenses. You can use this option in conjunction with the -batch keyword.</p> <p>If you have licenses for multiple products, separate each feature license by a colon so that licenses can be searched in the order they are read until an available license is found.</p> <p>For example:</p> <p>Suppose all your Synplify Pro licenses have been consumed, then you can automatically invoke a Synplify Premier license to run the Synplify Pro product if you specify the following command:</p> <pre data-bbox="470 1416 1209 1451">synplify_pro -licensetype synplifypro:synplifypremier</pre> <p>However, the Synplify Premier license does not automatically support the Synplify Pro product.</p>

Option	Description
-license_wait <i>waitTime</i>	<p>Specifies how long to wait for a Synopsys FPGA license. If you do not specify the -license_wait option, license queuing is not enabled.</p> <p>License queuing allows you to wait until a license becomes available or specify a wait time in seconds. You can use this option in conjunction with the -batch keyword. For details, see Queuing Licenses, on page 777 in the <i>User Guide</i>.</p> <p>The <i>waitTime</i> value determines license queuing and sets a maximum wait time in seconds:</p> <ul style="list-style-type: none"> • Undefined or 0 = Queuing off • 1 = Queuing enabled, indefinite wait time • >1 = Queuing enabled for the specified wait time
-log <i>filename</i>	Writes all output to the specified log file.
-loga <i>filename</i>	Writes all output to be appended to the specified log file.
-max_parallel_jobs <i>value</i>	Specifies the maximum number of concurrent processes used for synthesis.
-nopopup	Suppresses popup dialog boxes.
-run <i>implName</i>	Runs the specified implementation in the project file.
-runall	<p>Runs all the implementations in the project file.</p> <p>The Synplify tool supports only a single implementation.</p>
-shell	<p>Starts synthesis tool in shell mode.</p> <p>Note: The FPGA synthesis tools only support the -shell option on UNIX and Linux platforms.</p>
-tcl <i>prjFile Tclscript</i>	Starts the synthesis tool in the graphical user interface using the specified project or Tcl file.
-tclcmd <i>command</i>	Specifies Tcl command to be executed on startup.
-verbose_log	Writes messages to stdout.log in verbose mode.
-version	Reports version of specified synthesis tool.

sxml2pxml

Xilinx

If you choose to run the Xilinx P&R tool from outside the synthesis tool, you must first run the `sxml2pxml` Tcl command for the standalone XML converter. This Tcl command is used in the compile point synthesis and Xilinx Partition flow to generate a Xilinx `xpartition.pxml` file wh compile point information from the `xpartition.sxml` file for the specified implementation directory and design.

The syntax for the Tcl command is as follows:

```
sxml2pxml -design_name moduleName [-idir directoryName] [-odir directoryName]
```

-design_name <i>moduleName</i>	The name of the top-level module in the design.
-idir <i>directoryName</i>	The synthesis implementation directory containing the xpartition.sxml file. This input is optional; if missing, the current synthesis directory is assumed.
-odir <i>directoryName</i>	The P&R directory to which the xpartition.pxml file is written. This input is optional; if missing the current P&R directory is assumed.

Running XML Converter from the GUI

You can run this command from the Tcl Script window within the synthesis tool. For example:

- Run with all arguments:

```
sxml2pxml -design_name prep2_1 -idir /project/syn/rev_1  
-odir /project/pnr/par 1
```

- Run from the P&R directory:

```
sxml2pxml -design name prep2 1 -idir /project/syn/rev 1
```

- Run from the P&R directory with edif/ucf/sxml files copied to this directory:

```
sxml2pxml -design name prep2 1
```

Running XML Converter in Batch Mode

You can run this command in batch mode using the following methods:

- Run the command in batch .mode using a Tcl script file.
- The `sxml2pxml` Tcl command is contained in the script file. For example:

```
synplify_pro -batch script.tcl
```



- Run the command in batch mode with the TCL command using the `-tclcmd` option. For example:

```
synplify_pro -batch -tclcmd "sxml2pxml -design_name value  
[-idir value -odir value]"
```

vcs_launch

The vcs_launch command launches VCS with the specified options. For details about launching VCS from the synthesis tool, see [Simulating with the VCS Tool, on page 1174](#).

Syntax

```
vcs_launch [-config_file path] [-simtype rtl|postsyn] [-vlogan options]
            [-vhdlan options] [-vcs options] [-simv options] [-rundir path]
            [-lib path] [-testbench path] [-script_only]
```

The following table describes the options you can specify:

Option	Description
-config_file path	Specify a path to the VCS configuration file. This file is saved by VCS launch dialog.
-simtype rtl postsyn	Select either rtl postsyn for RTL or post-synthesis simulation, respectively.
-vlogan options	Specify the -vlogan options.
-vhdlan options	Specify the -vhdlan options.
-vcs options	Specify the -vcs options.
-simv options	Specify the -simv options.
-rundir path	Specify the path to directory to run simulation.
-lib path	Specify the path to vendor-specific library. Multiple -lib arguments can be used.
-testbench path	Specify the path to testbench file. Multiple -testbench arguments can be used.
-script_only	Generate VCS script only; don't run the script.

Tcl Command Categories

The following tables group Tcl commands together by type or functionality.

[Log File Commands](#), on page 216

[High Reliability Commands](#), on page 216

[Technology-Specific Tcl Commands](#), on page 217

Log File Commands

Synplify Pro, Synplify Premier

These Tcl commands let you filter messages in the log file.

log_filter	Lets you filter errors, notes, and warning messages.
log_report	Lets you write out the results of the <code>log_filter</code> command to a file.
message_override	Allows you to suppress or override the log file message ID specification.

High Reliability Commands

Intel FPGA, Lattice, Microchip, Xilinx

High-reliability commands specify I/O redundancy in high-reliability designs.

syn_connect	Specifies the connectivity between the module/instance being monitored for the error with the error monitoring IP port or top-level port for the error monitoring module. It is used with the <code>syn_connect</code> command.
syn_create_err_net	Specifies the connectivity between the module/instance being monitored for the error with the error monitoring IP port or top-level port for the error monitoring module. It is used with the <code>syn_connect</code> command.

Technology-Specific Tcl Commands

You can find vendor-specific Tcl commands in the appropriate vendor chapter.

Vendor/Family	Tcl Command Lists
Intel Stratix, Cyclone, Arria, HardCopy	Stratix, Arria, and Cyclone Families, on page 621
Intel FPGA MAX	MAX Family, on page 628
Lattice	ispGAL, ispGDX, and ispLSI Devices, on page 723 ispMACH and MACH Devices, on page 728 ispXPGA Devices, on page 725 ispXPLD5000MX and ispLSI5000VE Devices, on page 726 ORCA Devices, on page 715 LatticeECP/EC and Later Devices, on page 717 LatticeSC/SCM and LatticeXP2/XP Devices, on page 719 MachXO and Platform Devices, on page 721 Lattice iCE40/iCE5LP Devices, on page 730
Microchip	Microchip Tcl set_option Command Options, on page 814
Xilinx Virtex, Spartan-3	Virtex and Spartan-II and Later Technologies, on page 927
Xilinx CoolRunner, CoolRunner-II	Xilinx CPLDs, on page 937

CHAPTER 3

Tcl Find, Expand, and Collection Commands

The FPGA synthesis software includes powerful search functionality in the Tcl find and expand commands. Objects located by these commands can be grouped into collections and manipulated. The following sections describe the commands and collections in detail:

- [find](#), on page 221
- [find -filter](#), on page 229
- [expand](#), on page 237
- [Collection Commands](#), on page 240
- [Query Commands](#), on page 249
- [Synopsys Standard Collection Commands](#), on page 282

The find, expand, and collection commands are listed alphabetically in the following table.

add_to_collection	all_clocks	all_fanin
all_fanout	all_inputs	all_outputs
all_registers	append_to_collection	c_diff
c_info	c_intersect	c_list
c_print	c_symdiff	c_union
copy_collection	define_collection	define_scope_collection

dot2slash	expand	find (Tcl find)
foreach_in_collection	get_cells	get_clocks
get_clock_source	get_nets	get_object_name
get_pins	get_ports	get_prop
get_registers	index_collection	object_list
remove_from_collection	report_timing	set
sizeof_collection	slash2dot	

find

The Tcl find command identifies design objects based on specified criteria. Use this command to locate multiple objects with a common characteristic. If you want to locate objects that share connectivity, use the expand command instead of the find command ([expand, on page 237](#)).

You can specify the find command from the SCOPE environment or enter it as a Tcl command. This command operates on the RTL database.

You can define objects identified by find as a group or *collection*, and operate on all the objects in the collection at the same time. To do this, you embed the find command as part of a collection creation or manipulation command to do this in a single step. The combination of find and collection commands provides you with very powerful functionality to operate on and manipulate multiple design objects simultaneously.

The table summarizes where to find detailed information:

For ...	See ...
Command syntax	Tcl Find Syntax , on page 222
Syntax details: object types, expressions, case sensitivity, and special characters	Tcl Find Command Object Types , on page 223 Wildcards and Special Characters , on page 224 Tcl Find Command Case Sensitivity , on page 224
Examples of find syntax	Demos and Examples button, accessible from the tool UI Tcl Find Syntax Examples , on page 225
Filtering find searches by property	find -filter , on page 229 Find Filter Properties , on page 230 Refining Tcl Find Results with -filter, on page 229 in the <i>User Guide</i> .
Using find search patterns and using find in collections	Finding Objects with Tcl find and expand, on page 227 in the <i>User Guide</i> .

Tcl Find Syntax

Finds design objects based on specified criteria.

Find is available as part of the HDL Analyst tool.

Syntax

```
find
  [-flat]
  [-inst]
  [-net]
  [-port]
  [-pin]
  [-view]
  [-nocase]
  [-print]
  [-depth value]
  [-filter expression]
  [-seq]
  [pattern]
```

-flat

Extends the search to all levels. The * wildcard character matches hierarchy separators as well as characters. See [Wildcards and Special Characters, on page 224](#) for additional information.

-inst

Finds matching instances. If no -type (-inst, -net, -port, -pin, or -view) option is set, results include instances, nets, and ports.

-net

Finds matching nets. If no -type (-inst, -net, -port, -pin, or -view) option is set, results include instances, nets, and ports.

-port

Finds matching ports. If no -type (-inst, -net, -port, -pin, or -view) option is set, results include instances, nets, and ports.

-pin

Finds matching pins. If no -type (-inst, -net, -port, -pin, or -view) option is set, results include instances, nets, and ports.

-view

Finds matching views. If no -type (-inst, -net, -port, -pin, or -view) option is set, results include instances, nets, and ports.

-nocase

The **-nocase** option makes the search case-insensitive.

-print

Prints the first 20 search results. For a full list of objects found, use **c_print** or **c_list**. If you use **find** from the shell, the results are printed to the Tcl window; if you **find** in the constraint file, the results are printed to the log file at the beginning of the Mapper section. Reported object names have prefixes that identify the object type and are contained in curly braces ({}).

-depth value

Sets the starting depth for the search. Value may be a single number or a range. When **-depth** with a range is used, for example **-depth 4-7**, **-hier** and **-flat** arguments are ignored.

-filter expression

Further refines the results of **find** by filtering the results using the specified object property. For syntax details, refer to [find -filter, on page 229](#).

-seq

Finds sequential (clocked) instances (the **-inst** object type is not required). This argument is equivalent to **-filter @is_sequential**.

pattern

The value to search for.

Tcl Find Command Object Types

You can specify the following types of objects:

Object	Prefix	Example	Synopsis
view (Design)	v:	work.cpu.rt1 p:rst is the hierarchical rst port in the cpu view which points to all instances of cpu .	lib_cell
inst (Instance)	i:	Default object type. i:core.i_cpu.reg1 points to the reg1 instance inside i_cpu .	cell
port	p:	p:data_in[3] points to bit 3 of the primary data_in port. work.cpu.rt1 p:rst is the hierarchical rst port in the cpu view which eventually points to all instances of cpu .	port

Object	Prefix	Example	Synopsis
pin	t:	t:core.i_cpu.rst points to the hierarchical rst pin of instance i_cpu.	pin
net	n:	n:core.i_cpu.rst points to the rst net driven in i_cpu.	net
seq (Sequential instance)	i:	i:core.i_cpu.reg[7:0]	cell

Wildcards and Special Characters

The Tcl find command significantly differs from a simple Tcl search. A simple Tcl search does not treat any character, except for the backslash (\), as a special character, so * matches everything in a string. The Tcl find command uses various special characters, as shown in the following table.

Use curly brackets {} or double quotes to prevent the interpretation of special characters within a pattern, and the backslash to escape a single character.

Syntax Matches ...

- | | |
|---|---|
| * | A sequence of 0 or more matches
If you do not specify -hier, the search is restricted to the current view only.
To traverse downward through the hierarchy, either use the -hier argument or specify the hierarchical levels to be searched by adding the hierarchical delimiter to the pattern. For example, *.* matches objects one level below the current view. |
| ? | A sequence of 0 or 1 matches |

Tcl Find Command Case Sensitivity

Case sensitivity depends on the rules of the language used to specify the object. In mixed-language designs, the case-sensitivity rules for the parent object prevail, even when another language is used to define the lower-level object.

Tcl Find Syntax Examples

The following are examples of find syntax:

Example	Description
find {a*}	Finds any object in the current view that starts with a
find {a*} -hier -nocase	Finds any object that starts with a or A
find -net {*synp*} -hier	Finds any net that contains synp
find -seq * -filter {@clock==myclk}	Finds any register in the current view that is clocked by myclk
find -flat -seq {U1.*}	Finds all sequential elements at any hierarchical level under U1 (* matches hierarchy separator)
find -hier -flat -inst {i:A.B.C.*} -filter @view==ram*	Finds all RAM instances starting from a submodule and all lower hierarchical levels from A downwards
find -hier-seq {*} -filter @clock_enable==ena	Finds all registers enabled by the ena signal
find -hier-seq {*} -filter @slack <{-0.0}	Finds all sequential elements with negative slack
find -hier-seq {*} -filter {@clock ==clk1}	Finds all sequential elements within the clk1 clock domain
find -hier-net {*} -filter {@fanout >20}	Finds high fanout nets that drive more than 20 destinations
find -hier-seq * -in \$all_inst_coll	Finds sequential elements inside the all_inst_coll collection
find -net -regexp {[a-b].*}	Finds all nets in hierarchy a and b. This means {n:a.*} and {n:b.*} ; regular expressions are only supported in the earlier standard version of the HDL Analyst and are not supported in the current version.

Use the {} characters to protect patterns that contain [] from Tcl evaluation. For example, use the following command to find instance reg[4]:

```
find -inst {reg[4]}
```

Find Equivalents for UCF Group Commands

The following table shows UCF commands (simple Tcl) and their equivalent find commands:

UCF	Constraint Using find Command
INST “*ctrlfifo*” TNM = “FIFO_GRP”;_	set FIFO [find -hier -inst {i:*ctrlfifo*}]
INST “*ctrlfifo” TNM = “FIFO_GRP”;_	set FIFO [find -hier -inst {i:*ctrlfifo}]
INST “ctrlfifo*” TNM = “FIFO_GRP”;_	set FIFO [find -hier -inst * -filter @hier_rtl_name == ctrlfifo*]
INST “ctrlfifo*hier_inst” TNM = “FIFO_GRP”;_	set FIFO [find -hier -inst * -filter @hier_rtl_name == ctrlfifo*hier_inst]

Example: Custom Report Showing Paths with Negative Slack

Use the following commands:

```
open_design implementation_a/top.srm
set find_negslack[find -hier -seq -inst {*} -filter @slack <
    {-0.0}]
c_print -prop slack -prop view $find_negslack -file negslack.txt
```

The result of running these commands is a report called negslack.txt:

Object Name	slack	view
{i:CPU_A_SOC.CPU.DATAPATH.GBR[0]}	-3.264	"FDE"
{i:CPU_A_SOC.CPU.DATAPATH.GBR[1]}	-3.158	"FDE"
{i:CPU_A_SOC.CPU.DATAPATH.GBR[2]}	-3.091	"FDE"

Example: Custom Report for Negative Slack FFs in a Clock Domain

The following procedure steps through the commands used to find all negative slack flip-flops with a given clock domain:

1. Create a collection that contains all sequential elements with negative slack:

```
set negFF [find -tech -hier -seq {*} -filter @slack < {-0.0}]
```

2. Create a collection of all sequential elements within the clk clock domain

```
set clk1FF find -hier -seq * -filter {@clock==clk1}
```

3. Isolate the common elements in the two collections:

```
set clk1Slack [c_intersect $negFF $clk1FF]
```

4. Generate a report using the `c_print` command:

```
c_print [find -hier -net * -filter @fanout>=2]
{n:ack1_tmp}
{n:ack2_tmp}
...
{n:blk_xfer_cntrl_inst.lfsr_data[20:14]}
{n:blk_xfer_cntrl_inst.lfsr_inst.blk_size[6:0]}
{n:blk_xfer_cntrl_inst.lfsr_inst.clk_c}
...
```

Custom Fanout Report Example

The following command generates a fanout report:

```
% c_print -prop fanout [find -hier -net * -filter @fanout>=2]
```

This is an example of the report generated by the command:

Object Name	fanout
{n:ack1_tmp}	3
{n:ack2_tmp}	4
...	
{n:blk_xfer_cntrl_inst.lfsr_data[14]}	3
{n:blk_xfer_cntrl_inst.lfsr_data[15]}	3
{n:blk_xfer_cntrl_inst.lfsr_data[16]}	2
...	

You can add additional information to the report, by specifying more properties. For example:

```
% c_print -prop fanout [find -hier -net * -filter @fanout>=2] -prop pins
```

This command generates a report like the one shown below:

Object Name	Fanout	Pins
{n:ack1_tmp}	3	"t:word_xfer_cntrl_inst.ack1_tmp t:word_xfer_inst.ack1_tmp"
{n:ack2_tmp}	4	"t:blk_xfer_cntrl_inst.ack2_tmp t:blk_xfer_inst.ack2_tmp"
{n:adr_o_axb_1}	2	"t:blk_xfer_inst.adr_o_axb_1

```
{n:adr_o_axb_2} 2          t:adr_o_cry_1_0.S t:adr_o_s_1.LI"  
                            "t:blk_xfer_inst.adr_o_axb_2  
{n:adr_o_axb_3} 2          t:adr_o_cry_2_0.S t:adr_o_s_2.LI"  
                            "t:blk_xfer_inst.adr_o_axb_3  
{n:adr_o_axb_4} 2          t:adr_o_cry_3_0.S t:adr_o_s_3.LI"  
                            "t:blk_xfer_inst.adr_o_axb_4  
{n:adr_o_axb_5} 2          t:adr_o_cry_4_0.S t:adr_o_s_4.LI"  
                            "t:blk_xfer_inst.adr_o_axb_5  
{n:adr_o_axb_6} 2          t:adr_o_cry_5_0.S t:adr_o_s_5.LI"  
                            "t:blk_xfer_inst.adr_o_axb_6  
...                          t:adr_o_cry_6_0.S t:adr_o_s_6.LI"
```

To save the report as a file, use a command like this one:

```
c_print -prop fanout [find -hier -net * -filter @fanout>=2]  
      -prop pins -file prop.txt
```

find -filter

The Tcl find command includes the optional `-filter` option, which provides a powerful way to further refine the results of the find command and filter objects based on properties. See the following for details about the `find -filter` command:

- [Find -filter Syntax](#), on page 229
- [Find Filter Properties](#), on page 230
- [Find Filter Examples](#), on page 235

For the Tcl find command syntax, see

Find -filter Syntax

`find pattern otherOptions -filter {[!}@propertyName operator value}`

!	Optional character to specify the negative. Include the ! character if you are checking for the absence of a property; leave it out if you are checking for the presence of a property.
<code>@propertyName</code>	Property name to use for filtering. The name must be prefixed with the @ character. For example, if clock is the property name, specify {@clock==myclk}.
<code>operator</code>	Evaluates and determines the property value used for the filter expression. For the operators you can use in the expressions, see Filter Operators , on page 230.
<code>value</code>	Property value for the property in the filter expression, when the property has a value. The value can either be an object name such as myclk in {@clock==myclk}, or a value, such as 60 in {@fanout>=60}.

When specified, the `-filter` option must be the last option specified for the `find` command.

Filter Operators

You can use the following relational operators with the `-filter` option:

<code>==</code>	Equal
<code>!=</code>	Not equal
<code>></code>	Greater than
<code><</code>	Less than
<code>>=</code>	Greater than or equal to
<code><=</code>	Less than or equal to
<code>=~</code>	Matches pattern
<code>!~</code>	Does not match pattern

You can use the following logical operators with the `-filter` option:

<code>&&</code>	And
<code> </code>	Or
<code>!</code>	Not

Find Filter Properties

The object properties are based on the design or constraint, and are used to qualify searches and build collections. To generate these properties, open Project->Implementation Options->Device and enable the Annotated Properties for Analyst check box. The properties display in the Tcl window when the RTL or Technology view is active. Some properties are only available in a certain view. The tool creates .sap and .tap files (design and timing properties, respectively) in the project folder.

The table below lists the common filter object properties. It does not include some vendor-specific properties. Use the table as a guide to filter the properties you want. Here is how to read the columns:

Property Name	Property Value	HDL View	Description
Common Properties			
type	view port net instance pin	All	Specifies the type of object to be filtered from the netlist find * -filter -object -print
View Properties			
compile_point	locked	Tech	Filters the view based on compile point properties find * -view -filter @compile_point==locked -print
is_black_box	1	All	Filters the black box view find * -view -filter @is_black_box==1 -print
is_verilog	0 1	All	Filters the Verilog based view find * -view -filter @is_verilog=={0 1} -print
is_vhdl	0 1	All	Filters the VHDL based view find * -view -filter @is_vhdl=={0 1} -print
orig_inst_of	<i>viewName</i>	RTL and Tech	Filters the view based on original instance find * -view -filter @orig_inst_of==viewName -print
syn_hier	remove flatten soft firm hard	Tech	Filters the view based on the syn_hier attribute value find * -view -filter @syn_hier=={remove flatten soft firm hard} -print

Property Name	Property Value	HDL View	Description
Port Properties			
direction	input output inout	All	Filters the port based on port direction find * -port -filter @direction=={input output inout} -print
fanout	<i>value</i>	All	Filters the port based on fanout value find * -port -filter @fanout== <i>value</i> -print
package_pin	<i>pinPackageName</i>	All	Forward annotates the package_pin property for the port to the Xilinx XDC file. ^a
side	bottom top left right	Tech	Xilinx only
Instance Properties			
area	<i>areaValue</i>	Tech	
arrival_time	<i>value</i>	Tech	Corresponds to worst slack
async_reg	true false	Tech	Xilinx only
async_reset	n: <i>netName</i>	All	
async_set	n: <i>netName</i>	All	
clock	<i>clockName</i>	All	Could be a list if there are multiple clocks
clock_edge	rise fall high low	All	Could be a list if there are multiple clocks
clock_enable	n: <i>netName</i>	All	Highest branch name in the hierarchy, and closest to the driver
compile_point	locked	Tech	Automatically inherited from its view
hier_rtl_name	<i>hierInstanceName</i>	All	

Property Name	Property Value	HDL View	Description
inout_pin_count	<i>value</i>	All	
input_pin_count	<i>value</i>	All	
inst_of	<i>viewName</i>	All	
is_black_box	1 (Property added)	All	Automatically inherited from its view
is_hierarchical	1 (Property added)	All	
is_sequential	1 (Property added)	All	
is_combinatorial	1 (Property added)	All	
is_pad	1 (Property added)	All	
is_tristate	1 (Property added)	All	
is_keepbuf	1 (Property added)	All	
is_clock_gating	1 (Property added)	All	
is_vhdl	0 1	All	Automatically inherited from its view
is_verilog	0 1	All	Automatically inherited from its view
kind	<i>primitive</i> For example: inv and dff mux statemachine ...)	All	Tech view contains vendor-specific primitives
location	(<i>x</i> , <i>y</i>)	Tech	Format can differ
name	<i>instanceName</i>	All	
orientation	N S E W	Tech	
output_pin_count	<i>value</i>	All	
pin_count	<i>value</i>	All	
placement_type	unplaced placed	All	
rtl_name	<i>nonhierInstanceName</i>	All	
slack	<i>value</i>	Tech	Worst slack of all arcs
slow	1	Tech	

Property Name	Property Value	HDL View	Description
sync_reset	n:netName	All	
sync_set	n:netName	All	
syn_hier	remove flatten soft firm hard	Tech	Automatically inherited from its view
view	<i>viewName</i>	All	

Pin Properties

arrival_time	<i>timingValue</i>	Tech	
clock	<i>clockName</i>	All	Could be a list if there are multiple clocks
clock_edge	rise fall high low	All	Could be a list if there are multiple clocks
direction	input output inout	All	
fanout	<i>value</i>	All	Total fanout (integer)
is_clock	0 1	All	
is_const	0 1	All	
is_gated_clock	0 1	All	Set in addition to <i>is_clock</i>
slack	<i>value</i>	Tech	

Net Properties

clock	<i>clockName</i>	All	Could be a list if there are multiple clocks
is_clock	0 1	All	
is_gated_clock	0 1	All	Set in addition to <i>is_clock</i>
fanout	<i>value</i>	All	Total fanout (integer)

- a. It is recommended that you use the package_pin port property to assign pins in the Vivado tool, instead of the loc property used for cells.

Find Filter Examples

The following examples show how `find -filter` is used to check for the presence or absence of a property, with the `!` character indicating a negative check:

<code>c_print [find -hier -view {*} -filter {@is_black_box}]</code>	Finds all objects that are black boxes.
---	---

<code>c_print [find -hier -view {*} -filter {!@is_black_box}]</code>	Finds all objects that are not black boxes
--	--

Positive Check Examples

Finds all ports, pins, and nets from the top level with a fanout greater than 8:

```
find * -filter {@fanout>8}
```

Finds all instances other than `andv` and `orv` in the design:

```
find * -hier -filter {@view!=andv||@view!=orv}
```

Finds all instances of `statemachine` throughout the hierarchy:

```
find -hier -inst * -filter {@inst_of==statemachine}
```

```
find -hier -inst * -filter {@kind==statemachine}
```

Finds all instances throughout the hierarchy that include the string `reg` and are clocked by `CLK`:

```
find -hier -inst {*reg*} -filter {@clock==CLK}
```

Finds all nets throughout the hierarchy that have a fanout greater than 4:

```
find -hier -net {*} -filter {@fanout>4}
```

Negative Check Example

Finds all instances from the top level that have the include string `big`, that are not black boxes, and that have more than 10 pins:

```
find -inst *big* -filter {@is_black_box&&{@pin_count>10}}
```

Example of Boolean Expression Specified on Multiple Properties

Finds all instances from the top level that have more than eight pins and negative slack:

```
find * -filter {(@pin_count>8) && (@slack<0) }
```

Example of Pin Property Specified for Constants

Finds pins driven by constant 0 or constant 1:

```
find -pin *.* -filter @const==value -print
```

Where value of 1 lists all the pins that are tied to a constant.

expand

The expand command identifies objects based on their connectivity, by expanding forward from a given starting point. For more information, see [Using the Tcl expand Command to Define Collections, on page 232](#) of the *User Guide*.

Tcl expand Syntax

The syntax for the expand command is as follows:

```
expand [-objectType] [-from object] [-thru object] [-to object] [-level integer]
[-hier] [-leaf] [-seq] [-print]
```

Argument	Description
-from object	Specifies a list or collection of ports, instances, pins, or nets for expansion forward from all the pins listed. Instances and input pins are automatically expanded to all output pins of the instances. Nets are expanded to all output pins connected to the net. If you do not specify this argument, backward propagation stops at all sequential elements.
-hier	Searches for the pattern from every level of hierarchy, instead of just the top level and identifies objects to be expanded based on their connectivity. The default for the current view is the top level and is defined with the <code>define_current_design</code> command as in the compile-point flow.
-leaf	Returns only non-hierarchical instances.
-level integer	Limits the expansion to N logic levels of propagation. You cannot specify more than one -from, -thru, or -to point when using this option.
-objectType	Optionally specifies the type of object to be returned by the expansion. If you do not specify an <i>objectType</i> , all objects are returned. The object type is one of the following: <ul style="list-style-type: none"> • -instance returns all instances between the expansion points. This is the default. • -pin returns all instance pins between the expansion points. • -net returns all nets between the expansion points. • -port returns all top-level ports between the expansion points.

Argument	Description
-print	Evaluates the <code>expand</code> function and prints the first 20 results. If you use this command from HDL Analyst, these results are printed to the Tcl window; for constraint file commands, the results are printed to the log file at the start of the Mapper section. For a full list of objects found, you must use <code>c_print</code> or <code>c_list</code> . Reported object names have prefixes that identify the object type. There are double quotes around each name to allow for spaces in the names. For example: <pre>"i:reg1" "i:reg2" "i:\weird_name[foo\$]" "i:reg3" <<found 233 objects. Displaying first 20 objects. Use c_print or c_list for all. >></pre>
-seq	Modifies the range of any expansion to include only sequential elements. By default, the <code>expand</code> command returns all object types. If you want just sequential instances, make sure to define the <i>object_type</i> with the <code>-inst</code> argument, so that you limit the command to just instances.
-thru object	Specifies a list or collection of instances, pins, or nets for expansion forward or backward from all listed output pins and input pins respectively. Instances are automatically expanded to all input/output pins of the instances. Nets are expanded to all input/output pins connected to the net. You can have multiple <code>-thru</code> lists for product of sum (POS) operations.
-to object	Specifies a list or collection of ports, instances, pins, or nets for expansion backward from all the pins listed. Instances and output pins are automatically expanded to all input pins of the instances. Nets are expanded to all input pins connected to the net. If you do not specify this argument, forward propagation stops at all sequential elements.

Tcl expand Syntax Examples

Example	Description
expand -hier -from {i:reg1} -to {i:reg2}	Expands the cone of logic between two registers. Includes hierarchical instances below the current view.
expand -inst -from {i:reg1}	Expands the cone of logic from one register. Does not include instances below the current view.
expand -inst -hier -to {i:reg1}	Expands the cone of logic to one register. Includes hierarchical instances below the current view.
expand -pin -from {t:i_and2.z} -level 1	Finds all pins driven by the specified pin. Does not include pins below the current view.
expand -hier -to {t:i_and2.a} -level 1	Finds all instances driving an instance. Includes hierarchical instances below the current view.
expand -hier -from {n:cen}	Finds all elements in the transitive fanout of a clock enable net, across hierarchy.
expand -hier -from {n:cen} -level 1	Finds all elements directly connected to a clock enable net, across hierarchy.
expand -hier -thru {n:cen}	Finds all elements in the transitive fanout and transitive fanin of a clock enable net, across hierarchy.

Collection Commands

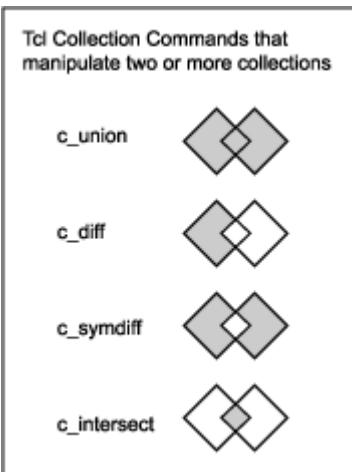
Synplify Pro, Synplify Premier

A collection is a group of objects. Grouping objects lets you operate on multiple group members at once; for example you can apply the same constraint to all the objects in a collection. You can do this from both the SCOPE editor (see [Collections, on page 304](#)) or in a Tcl file.

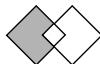
The following table lists the commands for creating, copying, evaluating, traversing, and filtering collections, and subsequent sections describe the collections, except for find and expand, in alphabetical order. For information on using collections, see [Using Collections, on page 236](#) in the *User Guide*.

Command	Description
Creation	
<code>define_collection</code>	Creates a collection from a list
<code>set modules</code>	Creates a collection
<code>set modules_copy \$modules</code>	Copies a collection
Creation from Objects Identified by Embedded Commands	
<code>find</code>	Does a targeted search and finds objects. Embedding the <code>find</code> command in a collection creation command first finds the objects, and then creates a collection out of the identified group of objects.
<code>expand</code>	Identifies related objects by expanding from a selected point. Embedding the <code>expand</code> command in a collection creation command first finds the objects, and then creates a collection out of the identified group of objects.
Operators for Comparison and Analysis	
<code>c_diff</code>	Identifies differences between lists or collections
<code>c_intersect</code>	Identifies objects common to a list and a collection
<code>c_symdiff</code>	Identifies objects that belong exclusively to only one list or collection
<code>c_union</code>	Concatenates a list to a collection

Command	Description
Operators for Evaluation and Statistics	
c_info	Prints statistics for a collection
c_list	Converts a collection to a Tcl list for evaluation
c_print	Displays collections or properties for evaluation



c_diff



Identifies differences by comparing collections, or a list and a collection. For this command to work, the design must be open in the GUI.

Syntax

c_diff {\$collection1 \$collection2 | \$collection {list}} [-print]

This command also includes a -print option to display the result.

Examples

The following examples combine the `set` with the `c_diff` command to create a new collection that contains the results of the `c_diff` command. The first example compares two collections and puts the results in `diffCollection`:

```
set diffCollection [c_diff $collection1 $collection2]
```

The next example creates `collection1` consisting of objects `i:reg1` and `i:reg2`, compares this collection to a Tcl list containing object `i:reg1`, puts the results in the collection `diffCollection` and prints the result (`i:reg2`).

```
%set collection1 {i:reg1 i:reg2}
%set diffCollection [c_diff $collection1 {i:reg1}]
%c_print $diffCollection
{i:reg2}
```

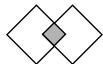
c_info

Returns specifics of a collection, including database name, number of objects per type, and total number of objects. You can save the results to a Tcl variable (array) using the `-array name` option.

Syntax

```
c_info $mycollection [-array name]
```

c_intersect



Defines common objects that are included in each of the collections or lists being compared.

Syntax

```
c_intersect $collection1 $collection2 | list [-print]
```

This command also includes a `-print` option to display the result.

Example

The following example uses the set command to create a new collection that contains the results of the c_intersect command. The example compares a list to a collection (myCollection) and puts the common elements in a new collection called commonCollection:

```
%set myCollection {i:reg1 i:reg2}
%set commonCollection [c_intersect $myCollection {i:reg1 i:reg3}]
%c_print $commonCollection
{i:reg1}
```

c_list

Converts a collection to a Tcl list of objects. You can evaluate any collection with this command. If you assign the collection to a variable, you can then manipulate the list using standard Tcl list commands like lappend and lsort.

Syntax

c_list *\$collection|list*

Example

```
%set myModules [find -view *]
%c_list $myModules
{v:top}{v:block_a}{v:block_b}
```

c_print

Displays collections or properties in column format. Object properties are printed using one or more -prop *propertyName* options.

Syntax

c_print *{\$collection | {list}} [-prop *propertyName*]* [-file *filename*] [-append] [-foot footer] [-head *header*]*

To print to a file, use the `-file` option. Use `-append` to append to the specified file instead of overwriting it. Use the `-head` and the `-foot` options to add the header and footer respectively, before printing the file. The following command in a constraint file prints the whole collection to a file:

```
c_print -file foo.txt $col
```

The following example adds a header and a footer to the file before printing:

```
% c_print [find -inst *] -head "Search for all..." -foot "Ends here."
```

```
Total finding took: 0 seconds
Search for all...
{:inst1}
{:inst2}
<<Collection has 2 objects>>
Ends here.
```

Note that the command prints the file to the current working directory. If you have multiple projects loaded, check that the file is written to the correct location. You can use the `pwd` command in the Tcl window to echo the current directory and then use `cd directoryName` to change the directory as needed.

Example

```
%set modules [find -view *]
%c_print $modules
{v:top}
{v:block_a}
{v:block_b}

%c_print -prop is_vhdl -prop is_verilog $modules
Name is_vhdl is_verilog
{v:top}1 0
{v:block_a}1 0
{v:block_b}1 0
```

c_symdiff



Compares a collection to another collection or Tcl list and finds the objects that are unique, not shared between the collections or Tcl lists being compared. It is the complement of the `c_intersect` command ([c_intersect, on page 242](#)).

Syntax

```
c_symdiff {$collection1 $collection2 | $collection {list}} [-print]
```

This command also includes a `-print` option to display the result.

Examples

The following example uses the `set` command together with the `c_symdiff` command to compare two collections and create a new collection (`symDiffCollection`) that contains the results of the `c_symdiff` command.

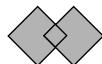
```
set symDiff_collection [c_symdiff $collection1 $collection2]
```

The next example is more detailed. It compares a list to a collection (`collection1`) and creates a new collection called `symDiffCollection` from the objects that are different. In this case, `reg1` is excluded from the new collection because it is common to both the list and `collection1`.

```
set collection1 {i:reg1 i:reg2}
set symDiffCollection [c_symdiff $collection1 {i:reg1 i:reg3}]
c_list $symDiffCollection
    {"i:reg2" "i:reg3"}
```

You can also use the command to compare two collections:

c_union



Adds a collection, or a list to a collection, and removes any redundant instances. For this command to work, the design must be open in the GUI.

Syntax

```
c_union {$collection1 $collection2 | $collection {list}} [-print]
```

The **c_union** command automatically removes redundant elements. This command also includes a **-print** option to display the result.

Examples

You can concatenate two collections into a new collection using the **c_union** and **set** commands, as shown in the following example where **collection1** and **collection2** are concatenated into **combined_collection**:

```
set combined_collection [c_union $collection1 $collection2]
```

The following example creates a new collection called **sumCollection**, which is generated by adding a Tcl list with one object (**reg3**) to **collection1**, which consists of **reg1** and **reg2**. The new collection created by **c_union** contains **reg1**, **reg2**, and **reg3**.

```
%set collection1 [find -instance {reg?} -print]
{i:reg1}
{i:reg2}
%set sumcollection [c_union $collection1 {i:reg3}]
%c_list $sumcollection
{i:reg1} {i:reg2} {i:reg3}
```

If instead you added **reg2** and **reg3** to **collection1** with the **c_union** command, the command removes redundant instances (**reg2**), so that the new collection still consists of **reg1**, **reg2**, and **reg3**.

```
%set collection1 {i:reg1 i:reg2}
%set sumcollection [c_union $collection1 {i:reg2 i:reg3}]
%c_list $sumcollection
{i:reg1} {i:reg2} {i:reg3}
```

define_collection

Creates a collection from any combination of single elements, Tcl lists, and collections. You get a warning message about empty collections if you define a collection with a leading asterisk and then define an attribute for it, as shown here:

```
set noretimesh [define_collection [find -hier -seq *uc_alu]]
define_attribute {$noretimesh} {syn_allow_retiming} {0}
```

To avoid the error message, remove the leading asterisk and change `*uc_alu` to `uc_alu`.

Example

```
set modules [define_collection {v:top} {v:cpu} $mycoll $mylist]
```

define_scope_collection

The `define_scope_collection` command combines `set` and `define_collection` to create a collection and assigns it to a variable.

```
define_scope_collection my_regs {find -hier -seq *my*}
```

get_prop

Returns a single property value for each member of the collection in a Tcl list.

Examples

```
get_prop -prop clock [find -seq *]
get_prop -array arr [find A1] -all
get_prop $listExpandedInst -prop rtl_name LOROM32X1inst
get_prop $listExpandedInst -prop location SLICE_X1y36
get_prop $listExpandedInst -prop bel C6LUT
get_prop $listExpandedInst -prop slack 0.678
```

If this command is used in a Tcl script and the results need to be printed, use a `puts` command.

```
foreach cel [c_list $all_hier] {puts [get_prop -prop view $cel];}
```

set

Copies a collection to create a new collection. This command copies the collection but not the name, so the two are independent. Changes to the original collection do not affect the copied collection.

Syntax

```
set collectionName collectionCriteria
```

```
set copyName $collectionName
```

collectionName The name of the new collection.

collectionCriteria Criteria for defining the elements to be included in the collection. Use this argument to embed other commands, like Tcl *find* and *expand*, as shown in the examples below, or other collection commands like *define_collection*, *c_intersect*, *c_diff*, *c_union*, and *c_symdiff*. Refer to these commands for examples.

copyName The name assigned to the copied collection.

\$collectionName Name of an existing collection to copy.

Examples

The following syntax examples illustrate how to use the *set* command:

- Use the *set* command to copy a collection:

```
set my_mod_copy $my_module
```

- Use the *set* command with a variable name and an embedded *find* command to create a collection from the *find* command results:

```
set my_module [find -view *]
```

- Use the *set* command with *define_collection* to create a collection:

```
set my_module [define_collection {v:top} {v:cpu} $col_1 $mylist]
```

For more examples of the *set* command used with embedded Tcl collection commands, see the examples in [c_diff, on page 241](#), [c_intersect, on page 242](#), [c_symdiff, on page 245](#), [c_union, on page 245](#), and [define_collection, on page 246](#).

Query Commands

The query commands are Synopsys SDC commands from the Design Compiler® tool for creating collections of specific object types. Functionally, they are equivalent to the Tcl *find* and *expand* commands ([find, on page 221](#) and [expand, on page 237](#)).

These query commands are intended to be used in the FDC file or the HDL Analyst view (see [Query Commands in HDL Analyst Tool, on page 249](#)) to create collections of objects for constraints. This section describes the syntax for the query commands supported in the FPGA synthesis tools. For complete documentation on these commands, see the Design Compiler documentation.

all_clocks	dot2slash	get_nets
all_fanin	get_cells	get_pins
all_fanout	get_clocks	get_ports
all_inputs	get_clock_source	get_registers
all_outputs	get_flat_cells	object_list
get_flat_pins	get_flat_nets	slash2dot

Note: Since all the query commands above are used to create Tcl collections of objects for constraints, they must be enclosed in [] to be applied. For example:

```
set_input_delay 0.5 [all_inputs] -clock clk
```

Query Commands in HDL Analyst Tool

Most of the query commands can be used in both the FDC file and the HDL Analyst view to create collections of objects for constraints. However, the [all_clocks](#) command cannot be implemented in the HDL Analyst view.

Note that the [get_clock_source](#), [slash2dot](#), and [dot2slash](#) commands are only used in the HDL Analyst view. You can test the query commands in the HDL Analyst tool, before implementing them in the FDC file.

To use the query commands in the HDL Analyst RTL view:

1. Enable the option: Implementation Options->Device->Annotate Properties for Analyst.

2. If results are not as expected, check that this option is turned on during compile and before you open the SRS view.

Annotated Properties for Analyst	<input checked="" type="checkbox"/>
Verification Mode	<input type="checkbox"/>
Resolve Mixed Drivers	<input type="checkbox"/>
Read Write Check on RAM	<input checked="" type="checkbox"/>

Query Commands and Tcl find and expand Commands

The Synopsys get* commands and all* commands are functionally similar to the Tcl find and expand commands. The get* commands and all* commands are better suited to use with constraints and the fdc file, because they handle properties like @clock better than the Tcl find and expand commands. In certain cases, the .fdc file does not support the find and expand commands, although you can still enter them in the Tcl window. See *Query Commands and Tcl find and expand Commands*, on page 250 for examples.

Query and Tcl find/expand Examples

The following table lists parallel examples that compare how to use either the Tcl find/expand or the get/all commands to query design objects and set constraints.

Return the output pins of top-level registers clocked by clk_b (e.g. inst1.inst2.my_reg.Q)

all_registers	FDC Constraint:
	<pre>set_multicycle_path {4} -from [all_registers -no_hierarchy -output_pins -clock [get_clocks {clk_b}]]] set_multicycle_path {4} -from [get_pins -of_objects [get_cells * -filter {@clock == clk_b}] -filter {@name == Q}]]</pre>

find	Tcl Window:
	<pre>% define_collection [regsub -all {i:([^s]+)} [join [c_list [find -inst * -filter @clock == clkfx]]] {t:\1.Q}]</pre>

Return all registers in the design clocked by the rising edge of clock clk_fx

all_registers	FDC Constraint:
	<pre>set_multicycle_path {3} -to [all_registers -cells -rise_clock [get_clocks {clk_fx}]] set_multicycle_path {3} -to [get_cells -hier * -filter {@clock == clk_fx && @clock_edge == rise}]</pre>

find Tcl Window:
`find -hier -inst * -filter {@clock == clkfx && @clock_edge == rise}`

Return clock pins of all registers clocked by the falling edge of clkfx

all_registers FDC Constraint:
`set_multicycle_path {2} -from [all_registers -clock_pins -fall_clock [get_clocks {clkfx}]]`
`set_multicycle_path {2} -from [get_pins -of_objects [get_cells -hier * -filter {@clock == clkfx && @clock_edge == fall}] -filter {@name == C}]`

find Tcl Window:
`% find -hier -inst * -filter {@clock == clkfx && @clock_edge == fall}`

Return the E pins of all instances of dffre cells (e.g. inst1.inst2.my_reg.E)

get_pins FDC Constraint:
`set_multicycle_path -to [get_pins -filter {@name == E} -of_objects [get_cells -hier * -filter {@inst_of == dffre}]]`

find Tcl Window and FDC Constraint:
`% regsub -all {\:([^\s]+)} [join [c_list [find -hier -inst * -filter @inst_of == dffre]] {t:\1.E}]`

all_clocks

Use this command in the .fdc constraint file to return a collection of objects. This command is not supported in the HDL Analyst view.

Returns a collection of clocks in the current design.

Syntax

This is the supported syntax for the `all_clocks` command:

all_clocks

This command has no arguments. All clocks must be defined in the design before using this command. To create clocks, you can use the `create_clock` command.

Example

The following constraint sets a multicycle path from all the starting points.

```
set_multicycle_path 3 -from [all_clocks]
```

all_fanin

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Reports pins, ports, or cells for the fanin of the specified sinks in the to list.

Syntax

This is the supported syntax for the all_fanin command:

```
all_fanin
[-break_on_bboxes]
[-endpoints_only]
[-exclude_bboxes]
[-flat]
[-levels integer]
[-only_cells]
[-startpoints_only]
-to listC
[-trace_arcs all | timing]
```

Arguments

-break_on_bboxes	Stops timing fanin from traversing on black boxes.
-endpoints_only	Returns only timing end points.
-exclude_bboxes	Excludes black boxes from the final result.
-flat	The fanin function operates in flat mode. This function can be specified in hierarchical (default) or flat mode. For hierarchical mode, only objects in the same hierarchy level as the current sink are returned. The pins within a level of hierarchy below the sink are traversed, but are not reported.
-levels <i>integer</i>	Stops traversal when the perimeter of the search <i>integer</i> hops is reached. For example, a level 2 hop traverses through two levels of combinational logic and stops, instead of hopping through all levels and stopping at the first sequential or port object. Counting is performed for the layers of cells that are equidistant from the sink.

-only_cells	Results include a set of all cells from the timing fanin for <i>listC</i> .
-startpoints_only	Returns only timing start points.
-to <i>listC</i>	<i>Required.</i> Reports a list of sink pins, ports, or nets in the design and the timing fanin of each sink in the <i>listC</i> Tcl list or collection specified. When you specify a net, effectively all drivers on the net are listed.
-trace_arcs all timing	Specifies the type of combinational arcs to trace while traversing the fanin. You can specify either: <ul style="list-style-type: none"> • all - Permits tracing of all combinational arcs. This is the default. • timing - Permits tracing of valid timing arcs only.

Examples

The following examples show the timing fanin of a port in the design.

```
all_fanin -to [get_ports y*]
  {t:yobuf[4].O t:yobuf[3].O t:yobuf[0].O t:yobuf[1].O
   t:yobuf[2].O t:yobuf[5].O t:yobuf[6].O t:yobuf[7].O t:GND.G
   t:moduley_inst.y_c[0] t:moduley_inst.y_c[1]
   t:moduley_inst.y_c[2]}

all_fanin -to [get_ports y*] -startpoints_only -flat
  {t:moduley_inst.q[2].Q t:moduley_inst.q[1].Q
   t:moduley_inst.q[0].Q}

all_fanin -to [get_ports y*] -startpoints_only -flat -only_cells
  {i:moduley_inst.q[0] i:moduley_inst.q[1] i:moduley_inst.q[2]}
```

all_fanout

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Returns a set of pins, ports, or cells for the fanout of the specified sources in the from list.

Syntax

This is the supported syntax for the `all_fanout` command:

```
all_fanout
[-break_on_bboxes]
[-clock_tree | -from listC
[-endpoints_only]
[-exclude_bboxes]
[-flat]
[-levels integer]
[-only_cells]
[-trace_arcs all |timing]
```

Arguments

-break_on_bboxes	Stops timing fanout from traversing on black boxes.
-clock_tree	Uses all clock source pins and/or ports in the design as its list of sources. Clock sources are specified with the <code>create_clock</code> command. If there are no clocks or clocks have no sources, then the report is empty. The <code>-clock_tree</code> option generates a report displaying the clock trees or networks in the design. The <code>-clock_tree</code> and <code>-from</code> options are mutually exclusive.
-endpoints_only	Returns only timing end points.
-exclude_bboxes	Excludes black boxes from the final result.
-flat	The fanout function operates in flat mode. This function can be specified in hierarchical (default) or flat mode. For hierarchical mode, only objects in the same hierarchy level as the current source are returned. The pins within a level of hierarchy below the source are traversed, but are not reported.
-from <i>listC</i>	Specifies a list of source pins, ports, or nets in the design. The timing fanout for each source of the <i>listC</i> Tcl list or collection is reported. When you specify a net, effectively all load pins on the net are listed. The <code>-clock_tree</code> and <code>-from</code> options are mutually exclusive.

-levels <i>integer</i>	Stops traversal when the perimeter of the search <i>integer</i> hops is reached. For example, a level 2 hop traverses through two levels of combinational logic and stops, instead of hopping through all levels and stopping at the first sequential or port object. Counting is performed for the layers of cells that are equidistant from the source.
-only_cells	Results include a set of all cells from the timing fanout for the <i>listC</i> .
-trace_arcs all timing	Specifies the type of combinational arcs to trace while traversing the fanout. You can specify either: <ul style="list-style-type: none"> • all - Permits tracing of all combinational arcs. This is the default. • timing - Permits tracing of valid timing arcs only.

Examples

The following examples show the timing fanout of a port in the design.

```
all_fanout -from [get_ports {a*}]
  {t:a_ibuf[0].I t:a_ibuf[1].I t:a_ibuf[2].I t:hold_a.D
   t:modulex_inst.a_c[0] t:modulex_inst.a_c[1]
   t:modulex_inst.a_c[2]}

all_fanout -from [get_ports {a*}] -level 1
  {t:a_ibuf[0].I t:a_ibuf[1].I t:a_ibuf[2].I}

all_fanout -from [get_ports {a*}] -flat -endpoints_only
  {t:hold_a.D t:modulex_inst.qa[0].D t:modulex_inst.qa[1].D
   t:modulex_inst.qa[2].D t:modulex_inst.qa_fast[0].D}
```

all_inputs

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Returns a collection of input or inout ports in the current design.

Syntax

This is the supported syntax for the `all_inputs` command:

```
all_inputs
  [-clock clockName]
  [-exclude_clock_port]
```

Arguments

-clock <i>clockName</i>	Limits the search to ports that have input delay relative to <i>clockName</i> .
-exclude_clock_port	Excludes clock ports from the search.

Examples

The following constraints set a default input delay.

```
set_input_delay 3 [all_inputs]
set_input_delay 3 -clock {clk} [all_inputs]
```

all_outputs

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Returns a collection of output or inout ports in the current design.

Syntax

This is the supported syntax for the `all_outputs` command:

```
all_outputs
  [-clock clockName]
```

Arguments

-clock <i>clockName</i>	Limits the search to ports that have output delay relative to <i>clockName</i> .
--------------------------------	--

Examples

The following constraints set a default output delay.

```
set_output_delay 2 [all_outputs]  
set_output_delay 2 -clock {clk} [all_outputs]
```

all_registers

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Returns a collection of sequential cells or pins in the current design.

Syntax

This is the supported syntax for the all_registers command:

```
all_registers  
[-clock clockName]  
[-rise_clock clockName]  
[-fall_clock clockName]  
[-cells]  
[-data_pins]  
[-clock_pins]  
[-output_pins]  
[-no_hierarchy]
```

Arguments

-clock *clockName* Searches only sequential cells that are clocked by the specified clock.

By default, all sequential cells in the current design are searched.

-rise_clock *clockName* Searches only sequential cells triggered by the rising edge of the specified clock.

By default, all sequential cells in the current design are searched.

-fall_clock <i>clockName</i>	Searches only sequential cells triggered by the falling edge of the specified clock. By default, all sequential cells in the current design are searched.
-cells	Returns a collection of sequential cells that meet the search criteria. If you do not specify any of the object types, the command returns a collection of sequential cells.
-data_pins	Returns a collection of data pins for the sequential cells that meet the search criteria.
-clock_pins	Returns a collection of clock pins for the sequential cells that meet the search criteria.
-output_pins	Returns a collection of output pins for the sequential cells that meet the search criteria.
-no_hierarchy	Limits the search to only the current level of hierarchy. Sub-designs are not searched. By default, the entire hierarchy is searched.

Examples

The following constraint sets a max delay target for timing paths leading to all registers.

```
set_max_delay 10.0 -to [all_registers]
```

The following constraint sets a max delay target for timing paths leading to all registers clocked by PHI2.

```
set_max_delay 10.0 -to [all_registers -clock [get_clocks PHI2]]
```

dot2slash

*Synplify Pro, Synplify Premier
Intel, Xilinx*

Translates the synthesis tool instance strings to the place-and-route instance strings. This command is helpful for debugging your design in the HDL Analyst view.

Syntax

This is the supported syntax for the dot2slash command:

dot2slash *dotHierInstStrg*

The Synplify Pro or Synplify Premier instance string is replaced in the place-and-route tool instance string, where the:

- Dot is converted to a slash.
- Square brackets "[]" can be escaped if they occur in the hierarchy string, but not when they are included in the leaf portion of the string.

Arguments

<i>dotHierInstStrg</i>	Converts the synthesis instance string to a format that the place-and-route tools understand.
------------------------	---

Examples

In the following example, the dot is replaced by the slash in the instance string.

```
% dot2slash {framer.go_dual\.\[9\]\.data.foo_bar[0]}
```

```
framer/go_dual.[9].data/foo_bar[0]
```

get_cells

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Creates a collection of cells from the current design that is relative to the current instance.

Syntax

This is the supported syntax for the `get_cells` command:

```
get_cells
[-hierarchical]
[-nocase]
[-regexp]
[-filter expression]
[pattern]
```

Arguments

-hierarchical	Searches each level of hierarchy for cells in the design relative to the current instance. The object name at a particular level must match the patterns. For the cell <code>block1/adder</code> , a hierarchical search uses "adder" to find this cell name. By default, the search is <i>not</i> hierarchical.
-nocase	Ensures that matches are case-insensitive. This applies for both the patterns argument and the filter operators (<code>=~</code> and <code>!~</code>).
-regexp	Views the patterns argument as a regular expression rather than a simple wildcard pattern. The behavior of the filter operators (<code>=~</code> and <code>!~</code>) have also been modified to use regular expression rather than simple wildcard patterns. When using the <code>-regexp</code> option, be careful how you quote the patterns argument and filter expression. Rigidly quoting with curly braces around regular expressions is recommended. Regular expressions are always anchored; that is, the expression assumes matching begins at the beginning of the object name and ends matching at the end of an object name. You can expand the search by adding <code>"./*"</code> to the beginning or end of the expressions, as needed.
-filter expressions	Filters the collection with the specified expression. For each cell in the collection, the expression is evaluated based on the cell's attributes. If the expression evaluates to true, the cell is included in the result.
<i>pattern</i>	Creates a collection of cells whose names match the specified patterns. Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Pattern matching is case sensitive unless you use the <code>-nocase</code> option.

Examples

The following example creates a collection of cells that begin with o and reference an FD2 library cell.

```
get_cells "o*" -filter "@ref_name =~ FD2"
```

The following example creates a collection of cells connected to a collection of pins.

```
set pinsel [get_pins o*/cp]
get_cells -of_objects $pinsel
```

The following example creates a collection of cells connected to a collection of nets.

```
set netsel [get_nets tmp]
get_cells -of_objects $netsel
```

This example creates a collection of cells with the string BSCAN in its name. Make sure to use the " \sim " operator when performing wildcard matching.

```
get_cells -hier * -filter {@hier_rtl_name =~ *BSCAN*}
```

get_clocks

Use this command in the .fdc constraint file and in the HDL Analyst view (on a limited basis) to return a collection of objects.

Creates a collection of clocks from the current design.

Syntax

This is the supported syntax for the get_clocks command:

```
get_clocks
  [-nocase]
  [-regexp]
  [-filter expression]
  [pattern | -of_objects objects]
  [-include_generated_clocks]
```

Arguments

-nocase	Ensures that matches are case-insensitive. This applies for both the patterns argument and the filter operators ($=\sim$ and $!\sim$).
-regexp	<p>Views the patterns argument as a regular expression rather than a simple wildcard pattern. The behavior of the filter operators ($=\sim$ and $!\sim$) have also been modified to use regular expression rather than simple wildcard patterns.</p> <p>When using the -regexp option, be careful how you quote the patterns argument and filter expression. Rigidly quoting with curly braces around regular expressions is recommended. Regular expressions are always anchored; that is, the expression assumes matching begins at the beginning of the object name and ends matching at the end of an object name. You can expand the search by adding ". * " to the beginning or end of the expressions, as needed.</p>
-filter expressions	<p>Filters the collection with the specified expression.</p> <p>For each clock in the collection, the expression is evaluated based on the clock's attributes. If the expression evaluates to true, the clock is included in the result. This option is not supported in the HDL Analyst view.</p>
<i>pattern</i>	Creates a collection of clocks whose names match the specified patterns. Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Pattern matching is case sensitive unless you use the -nocase option.
-of_objects <i>objects</i>	Creates a collection of clocks that are defined for the given net or pin objects.
-include_generated_clocks	<p>Creates a collection of clocks matching the search criteria and includes any clocks derived or generated from the source clocks found. Use this option to propagate exception constraints through MMCM/PLL or to generated/derived clocks.</p> <p>This option is not supported in the HDL Analyst view.</p>

Examples

The following example creates a collection of clocks that match the wildcard pattern.

```
get_clocks {*BUF_1*derived_clock*}
```

The following example creates a collection of clocks that match the given regular expression.

```
get_clocks -regexp {.*derived_clock}
```

The following example creates a collection that includes `clk` and any generated or derived clocks of `clk`.

```
get clocks -include generated clocks {clka}
```

The following example propagates exception constraints on generated/derived clocks:

```
set_false_path -from [get_clocks -include_generated_clocks GCLK2] -to  
[get_clocks -include_generated_clocks GCLK1]
```

get_clock_source

Intel, Xilinx

Use this command to identify derived clock sources when debugging your design in the HDL Analyst view. However, this command is not as effective if clocks in the design are heavily gated, because it can return all sources including registers of the gating logic.

Syntax

This is the supported syntax for the get_clock_source command:

get_clock_source *clockAlias*

Arguments

clockAlias Specifies the clock alias name for the synthesis clock.

Examples

```
get_clock_source {dcm|CLK0_BUF_derived_clock_CLKIN1}  
t:dcm inst.CLK_BUFO.O
```

For:

- Xilinx - The search stops at BUFG objects.

- Intel - The search stops at PLL objects.

get_flat_cells

Creates a collection of leaf cells that match certain criteria in the current design.

Syntax

```
get_flat_cells
  [-regexp | -exact]
  [-nocase]
  [-filter exper]
  [patterns | -of_objects objects]
  [-print]
```

Arguments

-regexp

Views the *patterns* argument as a regular expression rather than a simple wildcard pattern. This option also modifies the behavior of the `=~` and `!~` filter operators to use regular expressions rather than simple wildcard patterns.

When using the **-regexp** option, be careful how you quote the patterns argument and filter expression. Using rigid quoting with curly braces around regular expressions is recommended. Note that regular expressions are always anchored; that is, the expression is assumed to begin matching at the beginning of an object name and end matching at the end of an object name. You can widen the search by adding `".*"` at the beginning or end of the expressions, as needed.

The **-regexp** and **-exact** options are mutually exclusive; you can use only one.

-exact

Considers wildcards to be plain characters, and does not interpret their meaning as wildcards.

The **-regexp** and **-exact** options are mutually exclusive; you can use only one.

-nocase	Makes matches case-insensitive, both for the patterns argument and for the ==, =~, and !~ filter operators.
-filter expression	Filters the collection with the specified expression. For each cell in the collection, the expression is evaluated based on the cell's attributes. If the expression evaluates to true, the cell is included in the result.
<i>patterns</i>	Creates a collection of cells whose full names match the specified <i>patterns</i> . Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Pattern matching is case sensitive unless you use the -nocase option. When using the <i>patterns</i> argument, the command searches all leaf cells to match the <i>patterns</i> argument on their full names regardless of their hierarchical level. The <i>patterns</i> and -of_objects arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.
-of_objects objects	Creates a collection of cells connected to the specified objects. The <i>patterns</i> and -of_objects arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.
-print	Prints the contents of the collection.

get_flat_nets

Creates a collection of top-level nets of hierarchical net groups in the current design that match the specified criteria.

Syntax

```
get_flat_pins
[-regexp | -exact]
[-nocase]
[-filter expr]
[patterns | -of_objects objects]
[-print]
```

Arguments

-regexp	Views the <i>patterns</i> argument as a regular expression rather than a simple wildcard pattern. This option also modifies the behavior of the <code>=~</code> and <code>!~</code> filter operators to use regular expressions rather than simple wildcard patterns. When using the -regexp option, be careful how you quote the <i>patterns</i> argument and filter expression. Using rigid quoting with curly braces around regular expressions is recommended. Note that regular expressions are always anchored; that is, the expression is assumed to begin matching at the beginning of an object name and end matching at the end of an object name. You can widen the search by adding <code>".*"</code> at the beginning or end of the expressions, as needed. The -regexp and -exact options are mutually exclusive; you can use only one.
-exact	Considers wildcards to be plain characters, and does not interpret their meaning as wildcards. The -regexp and -exact options are mutually exclusive; you can use only one.
-nocase	Makes matches case-insensitive, both for the <i>patterns</i> argument and for the <code>==</code> , <code>=~</code> , and <code>!~</code> filter operators.

-filter expression	Filters the collection with the specified expression. For each net in the collection, the expression is evaluated based on the net's attributes. If the expression evaluates to true, the net is included in the result.
patterns	Creates a collection of top-level nets of hierarchical net groups whose full names match the specified patterns. Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Patterns can include the asterisk (*) and question mark (?) wildcard characters. Pattern matching is case sensitive unless you use the -nocase option. When using the patterns argument, the command searches all top-level nets of hierarchical net groups to match the patterns argument on their full name. The patterns and -of_objects arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.
-of_objects objects	Creates a collection of top-level nets of hierarchical net groups that are connected to the specified objects. Each object is either a named pin, port, net, cell, or a collection of these objects. The patterns and -of_objects arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.
-print	Prints the contents of the collection.

get_flat_pins

Creates a collection of leaf-cell pins that match specified criteria in the current design.

Syntax

```
get_flat_pins
[-regexp | -exact]
[-nocase]
[-filter expr]
[patterns | -of_objects objects]
[-print]
```

Arguments

-regexp	Views the <i>patterns</i> argument as a regular expression rather than a simple wildcard pattern. This option also modifies the behavior of the <code>=~</code> and <code>!~</code> filter operators to use regular expressions rather than simple wildcard patterns. When using the -regexp option, be careful how you quote the <i>patterns</i> argument and filter expression. Using rigid quoting with curly braces around regular expressions is recommended. Note that regular expressions are always anchored; that is, the expression is assumed to begin matching at the beginning of an object name and end matching at the end of an object name. You can widen the search by adding <code>".*"</code> at the beginning or end of the expressions, as needed. The -regexp and -exact options are mutually exclusive; you can use only one.
-exact	Considers wildcards to be plain characters, and does not interpret their meaning as wildcards. The -regexp and -exact options are mutually exclusive; you can use only one.
-nocase	Makes matches case-insensitive, both for the <i>patterns</i> argument and for the <code>==</code> , <code>=~</code> , and <code>!~</code> filter operators.

-filter expression Filters the collection with the specified expression. For each pin in the collection, the expression is evaluated based on the pin's attributes. If the expression evaluates to true, the pin is included in the result.

patterns Creates a collection of leaf-cell pins whose full names match the specified patterns. Patterns can include the asterisk (*) and question mark (?) wildcard characters. Pattern matching is case sensitive unless you use the **-nocase** option.

When using the patterns argument, the command searches all pins of leaf cells to match the patterns argument on their full names regardless of their hierarchical level.

The patterns and **-of_objects** arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.

-of_objects objects Creates a collection of leaf-cell pins connected to the specified objects. Each object is a named leaf cell, a net, or a collection of these objects. The **patterns** and **-of_objects** arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses the * (asterisk) as the default pattern.

-print Prints the contents of the collection.

get_nets

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Creates a collection of nets from the current design.

Syntax

This is the supported syntax for the `get_nets` command:

```
get_nets
  [-hierarchical]
  [-nocase]
  [-regexp | -exact]
  [-filter expression]
  [pattern | -of_objects objects]
```

Arguments

-hierarchical	Searches each level of hierarchy for nets in the design relative to the current instance. The object name at a particular level must match the patterns. For the net <code>block1/muxsel</code> a hierarchical search uses <code>muxsel</code> to find this net name. By default, the search is <i>not</i> hierarchical.
-nocase	Ensures that matches are case-insensitive. This applies for both the patterns argument and the filter operators (<code>=~</code> and <code>!~</code>).
-regexp	<p>Views the patterns argument as a regular expression rather than a simple wildcard pattern. The behavior of the filter operators (<code>=~</code> and <code>!~</code>) have also been modified to use regular expression rather than simple wildcard patterns.</p> <p>When using the <code>-regexp</code> option, be careful how you quote the patterns argument and filter expression. Rigidly quoting with curly braces around regular expressions is recommended. Regular expressions are always anchored; that is, the expression assumes matching begins at the beginning of the object name and ends matching at the end of an object name. You can expand the search by adding <code>".*"</code> to the beginning or end of the expressions, as needed.</p>
-filter expressions	<p>Filters the collection with the specified expression.</p> <p>For any nets in the collection, the expression is evaluated based on the net's attributes. If the expression evaluates to true, the net is included in the result.</p>

<i>pattern</i>	Creates a collection of nets whose names match the specified patterns. Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Pattern matching is case sensitive unless you use the -nocase option. The patterns and -of_objects arguments are mutually exclusive; you can specify only one. If you do not specify any of these arguments, the command uses * (asterisk) as the default pattern.
-of_objects objects	Creates a collection of nets connected to the specified objects. Each object can be a pin, port, or cell.

Examples

The following example creates a collection of nets connected to a collection of pins.

```
set pinsel [get_pins {o_reg1.Q o_reg2.Q}]\nget_nets -of_objects $pinsel
```

The following example creates a collection of nets connected to the E pin of any cell in the modulex_inst hierarchy.

```
get_nets {*.*} -filter {@pins =~ modulex_inst.*.E}
```

get_pins

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects. Creates a collection of pins from the current design that match the specified criteria.

When used without -hierarchical, include a dot (.) as a pin separator between the name of the instance and the pin name. Not including the hierarchy separator results in a warning message.

Syntax

This is the supported syntax for the `get_pins` command:

```
get_pins
  [-hierarchical]
  [-nocase]
  [-regexp | -exact]
  [-filter expression]
  [pattern |-of_objects objects] [-leaf]
```

Arguments

-hierarchical

Searches each level of hierarchy for pins, relative to the current instance, and reports all instances with that pin name. By default, searches are not hierarchical.

You can use wildcards with the `-hier` argument. The object name at a particular level must match the pattern. For the cell `block1/adder/D[0]`, a hierarchical search uses `adder/D[0]` to find pin names.

The pin separator is not required with `-hier`, although it is required if you use `get_pins` without `-hier` (see [Examples of get_pins, on page 273](#)).

However, when narrowing searches by specifying instance names as well as pin names, make sure to include the hierarchy separator. Otherwise, you will not get any search results:

```
% get_pins -hier {*reset_pipe*Q}
{ }

% get_pins -hier {*reset_pipe*.Q}
{t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[0].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[1].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[2].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[3].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[4].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[5].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe[6].Q
 t:sysip_inst.I_haps80_core\I_umr_clk_gen\reset_pipe_0[7].Q}
```

You cannot use the `-hierarchical` option with the `-of_objects` option.

-nocase

Ensures that matches are case-insensitive. This applies for both the *pattern* argument and the filter operators (== and !=).

-regexp|-exact

Views the patterns argument as a regular expression rather than a simple wildcard pattern. The behavior of the filter operators (== and !=) have also been modified to use regular expressions rather than simple wildcard patterns. When using the **-regexp** option, be careful how you quote the patterns argument and filter expression. Rigidly quoting with curly braces around regular expressions is recommended. Regular expressions are always anchored; that is, the expression assumes matching begins at the beginning of the object name and ends matching at the end of an object name. You can expand the search by adding ".*" to the beginning or end of the expressions, as needed. The **-exact** option treats wildcards as plain characters, so the meanings of these wildcard are not interpreted.

-filter expressions

Filters the collection with the specified expression. For each pin in the collection, the expression is evaluated based on the pin's attributes. If the expression evaluates to true, the pin is included in the result.

***pattern* | **-of_objects** *objects* [**-leaf**]**

Creates a collection of pins whose names match the specified patterns. Patterns can include the * (asterisk) and ? (question mark) wildcard characters. Pattern matching is case sensitive unless you use the **-nocase** option. The **-of_objects** option creates a collection of pins connected to the specified objects. Each object can be a cell or net. By default, the command considers only pins connected to the specified nets at the same hierarchical level. To consider only pins connected to leaf cells on the specified nets, use the **-leaf** option (the tool can cross hierarchical boundaries to find pins on leaf cells). You cannot use the **-hierarchical** option with the **-of_objects** option.

Examples of `get_pins`

This example creates a collection of all pins in the design.

```
get_pins -hier *.*
```

This example shows that without a separator, the command returns no results and generates a warning message:

```
% get_pins {Q}
```

Warning: No pin separator ('.') specified. Pattern must include a pin separator.

The following example creates a collection of pins from the top-level hierarchy that match the regular expression.

```
get_pins -regexp {.*\.ena}
```

The following example creates a collection of pins throughout the hierarchy that match the regular expression.

```
get_pins -hier - regexp {.*\.ena}
```

This example illustrates that you do not need the pin separator when you specify the **-hier** argument:

```
% get_pins -hier {Q}
{t:haps_system_capim.capi_di[0].Q t:haps_system_capim.capi_di[1].Q
t:haps_system_capim.capi_di[2].Q t:haps_system_capim.capi_di[3].Q
t:haps_system_capim.capi_di[4].Q t:haps_system_capim.capi_di[5].Q
t:haps_system_capim.capi_di[6].Q t:haps_system_capim.capi_di[7].Q
t:haps_system_capim.capi_di[8].Q t:haps_system_capim.capi_di[9].Q
t:haps_system_capim.capi_di[10].Q
t:haps_system_capim.capi_di[11].Q
...}
```

The next example creates a collection of hierarchical pin names for the library cell pin DQSFOUND, and for each instantiation of a library cell named PHASER_IN_PHY.

```
get_pins -filter {@name == DQSFOUND} -of_objects [get_cells -hier
* -filter {@inst_of == PHASER_IN_PHY}]
```

The following example creates a collection of library cell pins with the string DRCK in its name, for each instantiation of a library cell with the string BSCAN in its name. Whenever you use wildcards to match names, make sure to specify the "**=~**" operator instead of the "**==**" operator.

```
[get_pins -filter {@name=~*DRCK} -of_objects [get_cells -hier *
-filter {@hier_rtl_name =~ *BSCAN*}]]
```

get_ports

Use this command in the .fdc constraint file to return a collection of objects and/or in the HDL Analyst view to return a Tcl list of objects.

Creates a collection of top-level ports from that match the specified criteria.

Syntax

This is the supported syntax for the `get_ports` command:

```
get_ports
  [-nocase]
  [-regexp]
  [-filter expression]
  [pattern]
```

Arguments

-nocase	Ensures that matches are case-insensitive. This applies for both the patterns argument and the filter operators (<code>=~</code> and <code>!~</code>).
-regexp	Views the patterns argument as a regular expression rather than a simple wildcard pattern. The behavior of the filter operators (<code>=~</code> and <code>!~</code>) have also been modified to use regular expression rather than simple wildcard patterns. When using the <code>-regexp</code> option, be careful how you quote the patterns argument and filter expression. Rigidly quoting with curly braces around regular expressions is recommended. Regular expressions are always anchored; that is, the expression assumes matching begins at the beginning of the object name and ends matching at the end of an object name. You can expand the search by adding <code>". *"</code> to the beginning or end of the expressions, as needed.
-filter expressions	Filters the collection with the specified expression. For each port in the collection, the expression is evaluated based on the port's attributes. If the expression evaluates to true, the port is included in the result.
pattern	Creates a collection of ports whose names match the specified patterns. Patterns can include the <code>*</code> (asterisk) and <code>?</code> (question mark) wildcard characters. Pattern matching is case sensitive unless you use the <code>-nocase</code> option. The patterns and <code>-of_objects</code> arguments are mutually exclusive, so only specify one of them. If you do not specify either argument, the command uses <code>*</code> (asterisk) as the default pattern.

Examples

The following example queries all input ports beginning with mode.

```
get_ports mode* -filter {@direction =~ input}
```

get_registers

Intel, *Xilinx*

Use this command in the .fdc constraint file to return a collection of registers and/or in the HDL Analyst view to return a Tcl list of registers.

Syntax

This is the supported syntax for the get_registers command:

```
get_registers  
[-nocase]
```

Arguments

-nocase	Ensures that matches are case-insensitive.
----------------	--

Examples

This command is typically used for Intel technologies. However, the software honors the command and forward annotates it as get_cells to the XDC file when used with Xilinx technologies.

FDC File

```
set_max_delay -from [get_registers a*] -to [get_registers q*] 5
```

Intel SCF File

```
# Point-to-point Delay Constraints  
  
set_max_delay 5.000 -from [get_registers {a_reg[0] a_reg[3]  
a_reg[2] a_reg[1]}] -to [get_registers {q_c[0] q_c[1] q_c[2]  
q_c[3] q_c[4] q_c[5] q_c[6] q_c[7]}]
```

Xilinx XDC File

```
# 1002 : set_max_delay -from [get_registers a*]
          -to [get_registers q*] {5}
# line 4 in ../test.fdc

set_max_delay -from [get_cells {a_reg[*]}] -to [get_cells
{q_2_I_21 q_2_I_22 q_2_I_23 q_2_I_24 q_2_I_25 q_2_I_26 q_2_I_28
q_2_I_29 q_2_I_30 q_2_I_31 q_2_I_32 q_2_I_37}] {5.000}
```

object_list

Translates object strings returned by query commands in the HDL Analyst tool to proper Tcl lists. This allows you to process the results using Tcl commands.

Syntax

This is the supported syntax for the **object_list** command:

object_list *objectString*

Arguments

<i>objectString</i>	Converts the object string returned by an FDC query command to proper Tcl lists.
---------------------	--

Examples

For example:

```
% foreach x [object_list [get_cells -hier {q[*]}]]
  {puts "Match: $x"}
Match: i:modulex_inst.q[7:0]
Match: i:moduley_inst\[4\].q[7:0]
```

report_timing

Alternatively, use this command to generate timing reports for a design from the Tcl window, which follows the standards set for the Design Compiler or PrimeTime® commands. One advantage this command has over the Timing Analyst GUI in the synthesis tool is that the filter options (for example, -from/-to/-through) support the FDC query commands.

Syntax

This is the supported syntax for the report_timing command:

```
report_timing
  [-delay_type max]
  [-fall_from clock]
  [-fall_to clock]
  [-file string]
  [-from list]
  [-max_paths int]
  [-nworst 1]
  [-path_type full]
  [-rise_from clock]
  [-rise_to clock]
  [-slack_margin float]
  [-through instance]
  [-to list]
```

Arguments

-delay_type max	Specifies the path type for the end points. This default value can be specified as max; the maximum delay. All other values are ignored.
-fall_from <i>clock</i>	Reports paths from the falling edge of the specified clock. For a given clock, selects the starting points for paths clocked on the falling edge of the clock source.
-fall_to <i>clock</i>	Reports paths to the falling edge of the specified clock. For a given clock, selects the ending points for paths clocked on the falling edge of the clock source.
-file <i>string</i>	Allows the report to be re-directed to the specified file.

-from <i>list</i>	Reports paths from the specified port, register, register pin, or clock.
-max_paths <i>int</i>	Reports the number of paths to report for the path group. The default integer value is 1.
-nworst <i>1</i>	Reports the maximum number of paths to report for each timing end point. The default is 1, which reports the single worst path at a given end point. All other values are ignored.
-path_type <i>full</i>	Specifies how to display the timing path. This default value can be specified as full. A complete timing report is displayed, for example, containing timing start and end points, required time, and slack. All other values are ignored.
-rise_from <i>clock</i>	Reports paths from the rising edge of the specified clock. For a given clock, selects the starting points for paths clocked on the rising edge of the clock source.
-rise_to <i>clock</i>	Reports paths to the rising edge of the specified clock. For a given clock, selects the ending points for paths clocked on the rising edge of the clock source.
-slack_margin <i>float</i>	Reports paths for the specified slack margin, which allows you to specify a floating point value for the range of worst slack times.
-through <i>instance</i>	Reports paths that pass through the specified pins or nets.
-to <i>list</i>	Reports paths to the specified ports, register, register pin, or clock.

Examples

The following example reports timing to all registers clocked by clkb.

```
%report_timing -to [all_registers -clock {clkb}]

##### START OF TIMING REPORT #####
# Timing Report written on Mon Dec 16 10:35:02 2013|
#
Top view:          top
Requested Frequency: 50.0 MHz
Wire load mode:    top
Paths requested:   1
to:                moduley_inst.qa[7:0] moduley_inst qb[7:0]

Worst From-To Path Information
*****
```

Path information for path number 1:

Requested Period:	10.000
- Setup time:	-1.000
+ Clock delay at ending point:	0.909
= Required time:	11.909
- Propagation time:	0.000
= Slack :	11.909
Number of logic level(s) :	1
Starting point:	c[7:0] / c[0]
Ending point:	moduley_inst.qa[0] / D
The start point is clocked by	clkb [rising]
The end point is clocked by	clkb [rising] on pin C

Instance/Net Name	Type	Pin Name	Pin Dir	Delay	Arrival Time	No. of Fan Out(s)
c[7:0]	Port	c[0]	In	0.000	0.000	-
c[0]	Net	-	-	0.000	-	1
c_ibuf[0]	IBUF	I	In	-	0.000	-
c_ibuf[0]	IBUF	O	Out	0.000	0.000	-
c_c[0]	Net	-	-	0.000	-	1
moduley_inst.qa[0]	FDE	D	In	-	0.000	-

Path delay compensated for clock skew. Clock skew is added to clock-to-out value, and is subtracted from setup time value

End clock path:

Instance/Net Name	Type	Pin Name	Pin Dir	Delay	Arrival Time	No. of Fan Out(s)
clkb	Port	c[0]	In	0.000	0.000	-
clkb	Net	-	-	0.000	-	1
clkb_IBUFG	IBUF	I	In	-	0.000	-
clkb_IBUFG	IBUF	O	Out	0.000	0.000	-
clkb_i	Net	-	-	0.000	-	1
moduley_inst.qa[0]	FDE	D	In	-	0.000	-

END OF TIMING REPORT #####]

slash2dot

*Synplify Pro, Synplify Premier
Intel, Xilinx*

Translates place-and-route instance strings to the synthesis tool instance strings. This command is helpful for debugging your design in the HDL Analyst view.

Syntax

This is the supported syntax for the **slash2dot** command:

slash2dot *slashHierInstStrg*

The place-and-route instance string is replaced in the synthesis tool instance string, where the:

- Slash is converted to a dot.
- Square brackets "[]" can be escaped if they occur in the hierarchy string, but not when they are included in the leaf portion of the string.

Arguments

slashHierInstStrg Converts the place-and-route instance string to a format that the synthesis tools understand.

Examples

In the following example, the slash is replaced by the dot in the instance string.

```
% slash2dot {framer/go_dual.[9].data/foo_bar[0]}\n      framer.go_dual\[9\].data.foo_bar[0]
```

Synopsys Standard Collection Commands

There are a number of Synopsys standard SDC collection commands that can be included in the .fdc file. These commands are not compatible with the `define_scope_collection` command.

The collection commands let you manipulate or operate on multiple design objects simultaneously by creating, copying, evaluating, iterating, and filtering collections. This section describes the syntax for the following collection commands supported in the FPGA synthesis tools; for the complete syntax for these commands, refer to the Design Compiler documentation.

add_to_collection	append_to_collection
copy_collection	foreach_in_collection
get_object_name	index_collection
remove_from_collection	sizeof_collection

Use these commands in the FDC constraint file to facilitate the shared scripting of constraint specifications between the FPGA synthesis and Design Compiler tools.

add_to_collection

Adds objects to a collection that results in a new collection. The base collection remains unchanged. Any duplicate objects in the resulting collection are automatically removed from the collection.

Syntax

This is the supported syntax for the `add_to_collection` command:

```
add_to_collection
  [collection1]
  [objectSpec]
```

Arguments

<i>collection1</i>	Specifies the base collection to which objects are to be added. This collection is copied to a resulting collection, where objects matching <i>objectSpec</i> are added to this results collection.
<i>objectSpec</i>	Specifies a list of named objects or collections to add. Depending on the base collection type (heterogeneous or homogeneous), the searches and resulting collection may differ. For more information, see <i>Heterogeneous Base Collection</i> , on page 283 and <i>Homogeneous Base Collection</i> , on page 283.

Description

The `add_to_collection` command allows you to add elements to a collection. The result is a new collection representing the objects added from the *objectSpec* list to the base collection. Objects are duplicated in the resulting collection, unless they are removed using the `-unique` option. If *objectSpec* is empty, then the new collection is a copy of the base collection. Depending on the base collection type (heterogeneous or homogeneous), the searches and resulting collection may differ.

Heterogeneous Base Collection

If the base collection is heterogeneous, then only collections are added to the resulting collection. All implicit elements of the *objectSpec* list are ignored.

Homogeneous Base Collection

If the base collection is homogeneous and any elements of *objectSpec* are not collections, then the command searches the design using the object class of the base collection.

When *collection1* is an empty collection, special rules apply to *objectSpec*. If *objectSpec* is not empty, at least one homogeneous collection must be in the *objectSpec* list (can be any position in the list). The first homogeneous collection in the *objectSpec* list becomes the base collection and sets the object class for the function.

Examples

```
set result [get_cells{u*}]
get_object_name $result
```

```

==> {u:u1} {i:u2} {i:u3}

set result_1 [add_to_collection $result {get_cells {i:clkb_IBUFG}}
get_object_name $result_1

==> {i:u1} {i:u2} {i:u3} {i:clkb_IBUFG}

```

See Also

- [append_to_collection](#)

append_to_collection

Adds objects to the collection specified by a variable, modifying its value. Objects must be unique, since duplicate objects are not supported.

Syntax

This is the supported syntax for the `append_to_collection` command:

```

append_to_collection
  [variableName]
  [objectSpec]

```

Arguments

<i>variableName</i>	Specifies a variable name. The objects matching <i>objectSpec</i> are added to the collection referenced by this variable.
<i>objectSpec</i>	Specifies a list of named objects or collections to add to the resulting collection.

Description

The `append_to_collection` command allows you to add elements to a collection. This command treats the *variableName* option as a collection, and appends all the elements of *objectSpec* to that collection. If the variable does not exist, it creates a collection with elements from the *objectSpec* as its value. So, a collection is created that was referenced initially by *variableName* or automatically if the *variableName* was not provided. However, if the variable exists but does not contain a collection, then an error is generated.

The `append_to_collection` command can be more efficient than the `add_to_collection` command ([add_to_collection, on page 282](#)) when you are building a collection in a loop.

Examples

```
set result [get_cells{u*}]
get_object_name $result
==> {u:u1} {i:u2} {i:u3}

append_to_collection result {get_cells {i:clkb_IBUFG}}
get_object_name $result
==> {i:u1} {i:u2} {i:u3} {i:clkb_IBUFG}
```

See Also

- [add_to_collection](#)

copy_collection

Duplicates the contents of a collection that results a new collection. The base collection remains unchanged.

Syntax

This is the supported syntax for the `copy_collection` command:

```
copy_collection
[collection1]
```

Arguments

<i>collection1</i>	Specifies the collection to be copied.
--------------------	--

Description

The `copy_collection` command is an efficient mechanism to create a duplicate of an existing collection. It is sometimes more efficient and usually sufficient to simply have more than one variable referencing the same collection. However, whenever you want to copy the collection instead of reference it, use the `copy_collection` command.

Be aware that if an empty string is used for the `collection1` argument, the command returns an empty string. This means that a copy of the empty collection is an empty collection.

Examples

```
set insts [define_collection {u1 u2 u3 u4}]
set result_copy [copy_collection $insts]
get_object_name $result_copy
==> {u1} {u2} {u3} {u4}
```

foreach_in_collection

Iterates on the elements of a collection.

Syntax

This is the supported syntax for the `foreach_in_collection` command:

```
foreach_in_collection
[iterationVariable]
[collections]
[body]
```

Arguments

<i>iterationVariable</i>	Specifies the name of the iteration variable. It is set to a collection of one object. Any argument that accepts collections as an argument can also accept the <i>iterationVariable</i> , as they are the same data type.
--------------------------	--

<i>collections</i>	Specifies a list of collections on which to iterate.
--------------------	--

body Specifies a script to execute for the iteration. If the body of the iteration is modifying the netlist, all or part of the collection involved in the iteration can be deleted. The `foreach_in_collection` command is safe for such operations. A message is generated that indicates the iteration ended prematurely.

Description

The `foreach_in_collection` command is a Design Compiler and PrimeTime command used to iterate on each element of a collection. This command requires the following arguments: an iteration variable (do not specify a list), the collection on which to iterate, and the script to apply for each iteration.

You can nest this command within other control structures, including another `foreach_in_collection` command.

You can include the command in an FDC file, but if you are using the Tcl window and the HDL Analyst tool, you must use the standard Tcl `foreach` command instead of `foreach_in_collection`.

Examples

The following examples show valid methods to reference a collection for this command:

```
set seqs[all_registers]
set ports[all_inputs]

foreach_in_collection x [all_registers] {body}
foreach_in_collection x $ports {body}
foreach_in_collection x [list $seqs $ports] {body}
foreach_in_collection x {$seqs} {body}
foreach_in_collection x {$seqs $ports} {body}
```

get_object_name

Returns a list of names for objects in a collection.

Syntax

This is the supported syntax for the `get_object_name` command:

```
get_object_name  
[collection1]
```

Arguments

collection1 Specifies the name of the collection that contains the requested objects.

Examples

```
set c1[define_collection {u1 u2}]  
get_object_name $c1  
==> {u1} {u2}
```

index_collection

Creates a new collection that contains only the single object for the index specified in the base collection. You must provide an index to the collection.

Syntax

This is the supported syntax for the `index_collection` command:

```
index_collection  
[collection1]  
[index]
```

Arguments

collection1 Specifies the collection to be searched.

index Specifies an index to the collection. Allowed values are integers from 0 to sizeof collection - 1.

Description

You can use the `index_collection` command to extract a single object from a collection. The result is a new collection that contains only this object. The range of indices can be from 0 to one less than the size of the collection. If the specified index is outside that range, an error message is generated.

Commands that create a collection of objects do not impose a specific order on the collection, but they do generate the objects in the same, predictable order each time. Applications that support the sorting of collections allow you to impose a specific order on a collection.

If you use an empty string for the `collection1` argument, then any index to the empty collection is not valid. This results in an empty collection and generates an error message.

Be aware that all collections cannot be indexed.

Examples

```
set c1[get_cells {u1 u2}]
get_object_name [index_collection $c1 0]
==> {u1}
```

See Also

- `sizeof_collection`

remove_from_collection

Removes objects from a collection that results in a new collection. The base collection remains unchanged.

Syntax

This is the supported syntax for the remove_from_collection command:

```
remove_from_collection
[-intersect]
[collection1]
[objectSpec]
```

Arguments

-intersect	Removes objects from the base collection that are <i>not</i> found in <i>objectSpec</i> . By default, when this option is not specified, objects are removed from the base collection that are found in the <i>objectSpec</i> .
collection1	Specifies the base collection that is copied to a resulting collection, where objects matching <i>objectSpec</i> are removed from this results collection.
objectSpec	Specifies a list of named objects or collections to remove. The object class for each element in this list must be the same in the base collection. If the name matches an existing collection, that collection is used. Otherwise, objects are searched in the design using the object class for the base collection.

Description

The remove_from_collection command removes elements from a collection and creates a new collection.

When the -intersect option is not specified and there are no matches for *objectSpec*, the resulting collection is just a copy of the base collection. If everything in the *collection1* option matches the *objectSpec*, this results in an empty collection. When using the -intersect option, nothing is removed from the resulting collection.

Heterogeneous Base Collection

If the base collection is heterogeneous, then any elements of *objectSpec* that are not collections are ignored.

Homogeneous Base Collection

If the base collection is homogeneous and any elements of *objectSpec* are not collections, then the command searches the design using the object class of the base collection.

Examples

```
set c1[define_collection {u1 u2 u3}]
set c2[define_collection {u2 u3 u4}]
get_object_name [remove_from_collection $c1 $c2]
=> {u1}

get_object_name [remove_from_collection $c2 $c1]
=> {u4}

get_object_name [remove_from_collection -intersect $c1 $c2]
=> {u2} {u3}
```

See Also

- add to collection

sizeof collection

Returns the number of objects in a collection.

Syntax

This is the supported syntax for the `sizeof` collection command:

sizeof_collection
[*collection1*]

Arguments

<i>collection1</i>	Specifies the name of the collection for which the number of objects is requested. If no collection argument is specified, then the command returns 0.
--------------------	---

Examples

```
set c1[define_collection {u1 u2 u3}]\nsizeof_collection $c1\n==> 3
```

CHAPTER 4

Constraint Commands

The SCOPE (Synthesis Constraints OPTimization Environment®) editor automatically generates syntax for synthesis constraints. Enter information in the SCOPE tabs, panels, columns, and pull-downs to define constraints and parameter values. You can also drag and drop objects from the HDL Analyst UI to populate values in the constraint fields.

This interface creates Tcl-format *Synopsys Standard timing constraints* and *Synplify-style design constraints* and saves the syntax to an FPGA design constraints (FDC) file that is automatically added to your synthesis project. See [Constraint Types](#), on page 144 for definitions of synthesis constraints.

Topics in this section include:

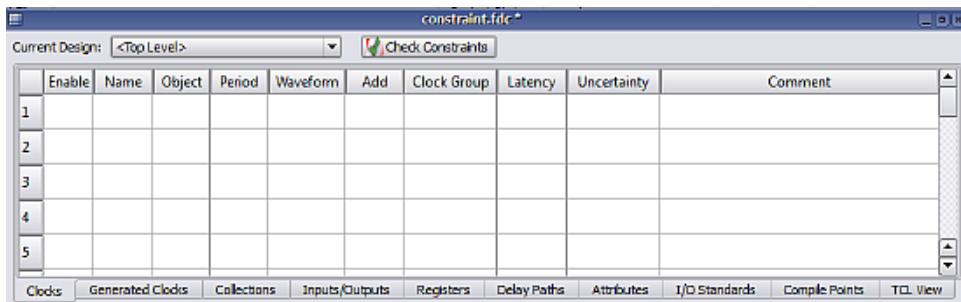
- [SCOPE Constraints Editor](#), on page 294
 - [SCOPE Tabs](#), on page 295
 - [Industry I/O Standards](#), on page 321
 - [Delay Path Timing Exceptions](#), on page 325
 - [Specifying From, To, and Through Points](#), on page 330
 - [Conflict Resolution for Timing Exceptions](#), on page 338

You can also specify Tcl equivalents for the timing and design constraints that are included in the SCOPE editor or a constraint file. For the constraint command syntax, see:

- [Timing Constraints](#), on page 342
 - [Design Constraints](#), on page 391

SCOPE Constraints Editor

The SCOPE editor contains a number of panels for creating and managing timing constraints and design attributes. This GUI offers the easiest way to create constraint files for your project. The syntax is saved to a file using an FDC extension and can be included in your design project.



From this editor, you specify timing constraints for clocks, ports, and nets as well as design constraints such as attributes, collections, and compile points. However, you cannot set black-box constraints from the SCOPE window.

To bring up the editor, use one of the following methods from the Project view:

- For a new file (the project file is open and the design is compiled):
 - Choose File->New->FPGA Design Constraints; select FPGA Constraint File (SCOPE).
 - Click the SCOPE icon in the toolbar; select FPGA Constraint File (SCOPE).
- You can also open the editor using an existing constraint file. Double-click the constraint file (FDC), or use File->Open, specifying the file type as FPGA Design Constraints File (*.fdc).

For more information about using FPGA timing constraints with your project, see [Using the SCOPE Editor, on page 204](#) in the *User Guide*.

For general information on the Design Constraints Format, see the *Using the Synopsys Design Constraints Format Application Note* on SolvNetPlus.

SCOPE Tabs

Here is a summary of the constraints created through the SCOPE editor:

SCOPE Panel	See ...
Clocks	Clocks , on page 296
Generated Clocks	Generated Clocks , on page 301
Collections	Collections , on page 304
Inputs/Outputs	Inputs/Outputs , on page 306
Registers	Registers , on page 309
Delay Paths	Delay Paths , on page 310
Attributes	Attributes , on page 313
I/O Standards	I/O Standards , on page 314
Compile Points	Compile Points , on page 316
TCL View	TCL View , on page 319

If you choose an object from a SCOPE pull-down menu, it has the appropriate prefix appended automatically. If you drag and drop an object from an RTL view, for example, make sure to add the prefix appropriate to the language used for the module. See [Naming Rule Syntax Commands](#), on page 388 for details.

Clocks

You use the Clocks panel of the SCOPE spreadsheet to define a signal as a clock.

	Enable	Name	Object	Period	Waveform	Add	Clock Group	Latency	Uncertainty	Comment
1										
2										
3										
4										
Clocks										

The Clocks panel includes the following options:

Field	Description
Name	Specifies the clock object name. Clocks can be defined on the following objects: <ul style="list-style-type: none"> • Pins • Ports • Nets For virtual clocks, the field must contain a unique name not associated with any port, pin, or net in the design.
Period	Specifies the clock period in nanoseconds. This is the minimum time over which the clock waveform repeats. The period must be greater than zero.
Waveform	Specifies the rise and fall edge times for the clock waveforms of the clock in nanoseconds, over an entire clock period. The first time in the list is a rising transition, typically the first rising transition after time zero. There must be two edges, and they are assumed to be rise and then fall. The edges must be monotonically increasing. If you do not specify this option, a default waveform is assumed, which has a rise edge of 0.0 and a fall edge of period/2.
Add Delay	Specifies whether to add this delay to the existing clock or to overwrite it. Use this option when multiple clocks must be specified on the same source for simultaneous analysis with different clock waveforms. When you use this option, you must also specify the clock, and clocks with the same source must have different names.

Field	Description
Clock Group	Assigns clocks to asynchronous clock groups. The clock grouping is inclusionary (for example, clk2 and clk3 can each be related to clk1 without being related to each other). For details, see Clock Groups , on page 297.
Latency	Specifies the clock latency applied to clock ports and clock aliases. Applying the latency constraint on a port can be used to model the off-chip clock delays in a multichip environment. Clock latency can only: <ul style="list-style-type: none"> • Apply to clocks defined on input ports. • Be used for source latency. • Apply to port clock objects.
Uncertainty	Specifies the clock uncertainty (skew characteristics) of the specified clock networks. You can only apply latency to clock objects.

Clock Groups

Clock grouping is associative; two clocks can be asynchronous to each other but both can be synchronous with a third clock.

The SCOPE GUI prompts you for a clock group for each clock that you define. By default, the tool assigns all clocks to the default clock group. When you add a name that differs from the default clock group name, the clock is assigned its own clock group and is asynchronous to the default clock group as well as all other named clock groups.

This section presents scenarios for defining clocks and includes the following examples:

- [Example 1 - SCOPE Definition](#)
- [Example 2 - Equivalent Tcl Syntax](#)
- [Example 3 - Establish Clock Relationships](#)
- [Example 4 - Using a Single Group Option](#)
- [Example 5 - Legacy Clock Grouping](#)
- [Example 6 - UCF Forward Annotation](#)

Example 1 - SCOPE Definition

A design has three clocks, clk1, clk2, clk3. You want clk1 and clk2 to be in the same clock group—synchronous to each other but asynchronous to clk3. You can apply this clock definition by adding a name in the Clock Group column, as shown below:

1	Enable	Name	Object	Period	Waveform	Add	Clock Group	Latency	U
1	<input checked="" type="checkbox"/>	clk1	clk1	7		<input type="checkbox"/>	group1		
2	<input checked="" type="checkbox"/>	clk2	nsclk2	10		<input type="checkbox"/>	group1		
3	<input checked="" type="checkbox"/>	clk3	clk3	12		<input type="checkbox"/>	<default>		
4									
5									
6									

This definition assigns clk1 and clk2 to clock group group1, synchronous to each other and asynchronous to clk3. The equivalent Tcl command for this appears in the text editor window as follows:

```
set_clock_groups -derive -asynchronous -name {group1}
                  -group {{c:clk1} {c:clk2}}
```

Example 2 - Equivalent Tcl Syntax

A design has three clocks: clk1, clk2, clk3. Use the following commands to set clk2 synchronous to clk3, but asynchronous to clk1:

```
set_clock_groups -asynchronous -group [get_clocks {clk3 clk2}]
set_clock_groups -asynchronous -group [get_clocks {clk1}]
```

Example 3 - Establish Clock Relationships

A design has the following clocks defined:

```
create_clock -name {clka} {p:clka} -period 10 -waveform {0 5.0}
create_clock -name {clkb} {p:clkb} -period 20 -waveform {0 10.0}
create_clock -name {my_sys} {p:sys_clk} -period 200 -waveform {0
100.0}
```

You want to define clka and clkb as asynchronous to each other and clka and clkb as synchronous to my_sys.

For the tool to establish these relationships, multiple -group options are needed in a single `set_clock_groups` command. Clocks defined by the first -group option are asynchronous to clocks in the subsequent -group option. Therefore, you can use the following syntax to establish the relationships described above:

```
set_clock_groups -asynchronous -group [get_clocks {clka}]
                  -group [get_clocks {clkb}]
```

Example 4 - Using a Single Group Option

`set_clock_groups` has a unique behavior when a single -group option is specified in the command. For this example, the following constraint specifications are applied:

```
set_clock_groups -asynchronous -name {default_clkgroup_0} -group
                  [get_clocks {clka my_sys}]
set_clock_groups -asynchronous -name {default_clkgroup_1} -group
                  [get_clocks {clkb my_sys}]
```

The first statement assigns clka AND my_sys as asynchronous to clkb, and the second statement assigns clkb AND my_sys as asynchronous to clka. Therefore, with this specification, all three clocks are established as asynchronous to each other.

Example 5 - Legacy Clock Grouping

This section shows how the legacy clock group definitions (Synplify-style timing constraints) are converted to the Synopsys standard timing syntax (FDC). Legacy clock grouping can be represented through Synopsys standard constraints, but the multi-grouping in the Synopsys standard constraints cannot be represented in legacy constraints.

For example, the following table shows legacy clock definitions and their translated FDC equivalents:

Legacy Definition	define_clock -name{clk_a}{p:clk_a}-period 10 -clockgroup default_clkgroup_0 define_clock -name {clk_b}{p:clk_b} -freq 150 -clockgroup default_clkgroup_1 define_clock -name {clk_c}{p:clk_c} -freq 200 -clockgroup default_clkgroup_1
FDC Definition	##### BEGIN Clocks - (Populated from SCOPE tab, do not edit) create_clock -name {clk_a} {p:clk_a} -period 10 -waveform {0 5.0} create_clock -name {clk_b} {p:clk_b} -period 6.667 -waveform {0 3.3335} create_clock -name {clk_c} {p:clk_c} -period 5.0 -waveform {0 2.5} set_clock_groups -derive -name default_clkgroup_0 -asynchronous -group {c:clk_a} set_clock_groups -derive -name default_clkgroup_1 -asynchronous -group {c:clk_b c:clk_c} ##### END Clocks

The `create_generated_clock` constraints used in legacy SDC are preserved in FDC. The `-derive` option directs the `create_generated_clock` command to inherit the `-source` clock group. This behavior is unique to FDC and is an extension of the Synopsys SDC standard functionality.

Example 6 - UCF Forward Annotation

To forward-annotate clock groups to a Xilinx technology UCF for the Xilinx ISE place and route tool, each clock must belong to only one clock group. Clock relationships in UCF must be represented in synthesis using the *single group* option method for one or more `set_clock_groups` constraints, but any given clock may be present in only one of the constraints. This is due to the linear nature of clock groups in UCF in that, if two clocks in a design are defined asynchronous to each other, a third clock in the design can only be defined as synchronous to one of the first two clocks, or asynchronous to both. The third clock can never be synchronous to both of the first two clocks.

The SCOPE method of defining clock groups as show in [Example 1 - SCOPE Definition](#), is specifically designed to make this linear type grouping the default and avoid issues with clock groups when going from the Synopsys standard clock grouping to Xilinx UCF clock grouping. The Synplify Xilinx mappers detect when the target forward annotation is UCF for Xilinx designs and issues an error if clock grouping in synthesis is not compatible with UCF. It is recommended that you use SCOPE to define your clocks and establish clock relationships when targeting Xilinx technologies for ISE place and route.

An example of the SCOPE generated SDC Standard syntax for clock grouping for a design with three clocks—clk_a, clk_b, and my_sys—is shown below:

```
set_clock_groups -asynchronous -group [get_clocks {clk_a my_sys}]
set_clock_groups -asynchronous -group [get_clocks {clk_b}]
```

OR

```
set_clock_groups -asynchronous -group [get_clocks {clk_b my_sys}]
set_clock_groups -asynchronous -group [get_clocks {clk_a}]
```

See Also

For equivalent Tcl syntax, see the following sections:

- [create_clock](#), on page 343
- [create_generated_clock](#), on page 345
- [set_clock_latency](#), on page 360
- [set_clock_uncertainty](#), on page 363

For information about other SCOPE panels, see [SCOPE Tabs](#), on page 295.

Generated Clocks

Use the Generated Clocks panel of the SCOPE spreadsheet to define a signal as a generated clock. The equivalent Tcl constraint is `create_generated_clock`; its syntax is described in [create_generated_clock](#), on page 345.

	Enable	Name	Source	Object	Master Clock	Generate Type	Generate Parameters	Generate Modifier	Modifier Parameters	Invert	Add	Comment
1	<input checked="" type="checkbox"/>											
2												
3												
4												

Generated Clocks

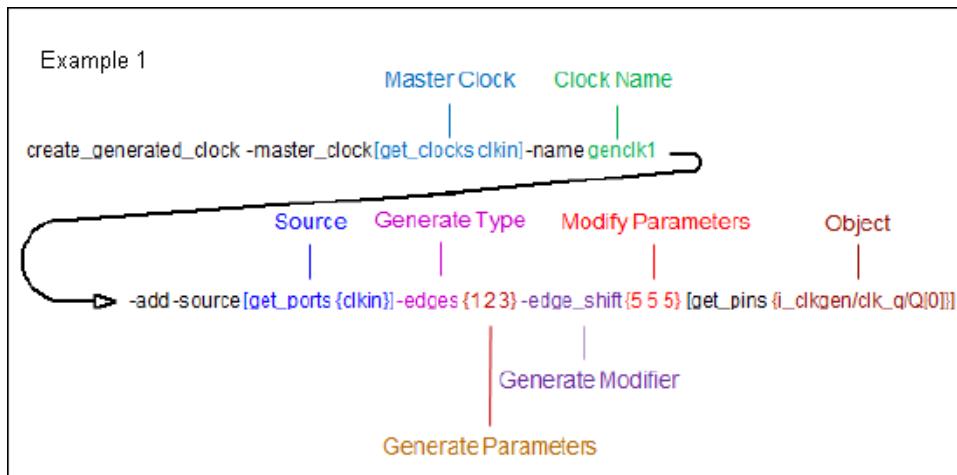
The Generated Clocks panel includes the following options:

Field	Description
Name	Specifies the name of the generated clock. If this option is not used, the clock gets the name of the first clock source specified in the source.
Source	Specifies the master clock pin, which is either a master clock source pin or a fanout pin of the master clock driving the generated clock definition pin. The clock waveform at the master pin is used for deriving the generated clock waveform.
Object	Generated clocks can be defined on the following objects: <ul style="list-style-type: none"> • Pins • Ports • Nets • Instances - Where instances have only one output (for example, BUFGs)
Master Clock	Specifies the master clock to be used for this generated clock, when multiple clocks fan into the master pin.
Generate Type	Specifies any of the following: <p>edges - Specifies a list of integers that represents edges from the source clock that are to form the edges of the generated clock. The edges are interpreted as alternating rising and falling edges and each edge must not be less than its previous edge. The number of edges must be an odd number and not less than 3 to make one full clock cycle of the generated clock waveform. For example, 1 represents the first source edge, 2 represents the second source edge, and so on.</p> <p>divide_by - Specifies the frequency division factor. If the divide factor value is 2, the generated clock period is twice as long as the master clock period.</p> <p>multiply_by - Specifies the frequency multiplication factor. If the multiply factor value is 3, the generated clock period is one-third as long as the master clock period.</p>
Generate Parameters	Specifies integers that define the type of generated clock.
Generate Modifier	Defines the secondary characteristics of the generated clock.

Field	Description
Modify Parameters	Defines modifier values of the generated clock.
Invert	Specifies whether to use invert - Inverts the generated clock signal (in the case of frequency multiplication and division).
Add	Either add this clock to the existing clock or overwrite it.
	Use this option when multiple generated clocks must be specified on the same source, because multiple clocks fan into the master pin. Ideally, one generated clock must be specified for each clock that fans into the master pin. If you specify this option, you must also specify the clock and master clock. The clocks with the same source must have different names.

Examples

In the following example, the generated clock `genclk1` is created with the same frequency as the source clock `clkin`, but its phase is shifted by 180. Each of the edges of the generated clock shifts by 5 ns, which is specified by the `-edges` and `-edge_shift` options.



For this example, a generated clock is created with half the frequency of the source clock.

Example 2

```

        Generate Type
create_generated_clock-name divclk-source [get_ports {clkin}]-divide_by 2 [get_pins {un1_divclk.OUT[0]}]
        |Generate Parameters

```

For more information about other SCOPE options, see [SCOPE Tabs, on page 295](#).

Collections

The Collections tab allows you to set constraints for a group of objects you have defined as a collection with the Tcl command. For details, see [Creating and Using SCOPE Collections, on page 237](#) of the *User Guide*.

	Enable	Name	Command	Comment
1				
2				
3				

Collections

Field	Description
Enable	Enables the row.
Name	Enter the collection name.
Command	Select a collection creation command from the drop-down menu. See Collection Commands , on page 305 for descriptions of the commands.
Comment	Enter comments that are included in the constraints file.

You can crossprobe the collection results to an HDL Analyst view. To do this, right-click in the SCOPE cell and select the option Select in Analyst.

Collection Commands

You can use the collection commands on collections or Tcl lists. Tcl lists can be just a single element long.

To ...	Use this command ...
Create a collection	<p><code>set modules</code></p> <p>To create and save a collection, assign it to a variable. You can also use this command to create a collection from any combination of single elements, TCL lists and collections:</p> <pre>set modules [define_collection {v:top} {v:cpu} \$mycoll \$mylist]</pre> <p>Once you have created a collection, you can assign constraints to it in the SCOPE interface.</p>
Copy a collection	<p><code>set modules_copy \$modules</code></p> <p>This copies the collection, so that any change to \$modules does not affect \$modules_copy.</p>
Evaluate a collection	<p><code>c_print</code></p> <p>This command returns all objects in a column format. Use this for visual inspection.</p> <p><code>c_list</code></p> <p>This command returns a Tcl list of objects. Use this to convert a collection to a list. You can manipulate a Tcl list with standard Tcl list commands.</p>
Concatenate a list to a collection	<code>c_union</code>
Identify differences between lists or collections	<p><code>c_diff</code></p> <p>Identifies differences between a list and a collection or between two or more collections. Use the -print option to display the results.</p>
Identify objects common to a list and a collection	<p><code>c_intersect</code></p> <p>Use the -print option to display the results.</p>
Identify objects common to two or more collections	<p><code>c_sub</code></p> <p>Use the -print option to display the results.</p>
Identify objects that belong exclusively to only one list or collection	<p><code>c_symdiff</code></p> <p>Use this to identify unique objects in a list and a collection, or two or more collections. Use the -print option to display the results.</p>

For information about all SCOPE panels, see [SCOPE Tabs, on page 295](#).

Inputs/Outputs

The Inputs/Outputs panel models the interface of the FPGA with the outside environment. You use it to specify delays outside the device.

	Enable	Delay Type	Port	Rise	Fall	Max	Min	Clock	Clock Fall	Add Delay	Value	Comment
1												
2												
3												

Inputs/Outputs

The Inputs/Outputs panel includes the following options:

Field	Description
Delay Type	Specifies whether the delay is an input or output delay.
Port	Specifies the name of the port.
Rise	Specifies that the delay is relative to the rising transition on specified port. Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the <code>-rise</code> option is preserved and forward annotated to the place-and-route tool.
Fall	Specifies that the delay is relative to the falling transition on specified port Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the <code>-fall</code> option is preserved and forward annotated to the place-and-route tool.
Max	Specifies that the delay value is relative to the longest path. Note: The <code>-max</code> delay values are reported in the top-level log file and are forward annotated to the place-and-route tool.

Field	Description
Min	Specifies that the delay value is relative to the shortest path. Note: The synthesis tool does not optimize for hold time violations and only reports -min delay values in the <code>synlog/topLevel_fpga_mapper.srr_Min</code> timing report section of the log file. The -min delay values are forward annotated to the place-and-route tool.
Clock	Specifies the name of a clock for which the specified delay is applied. If you specify the clock fall, you must also specify the name of the clock.
Clock Fall	Specifies that the delay relative to the falling edge of the clock. For examples, see Input Delays , on page 307 and Output Delays , on page 308 .
Add Delay	Specifies whether to add delay information to the existing input delay or overwrite the input delay. For examples, see Input Delays , on page 307 and Output Delays , on page 308 .
Value	Specifies the delay path value.

Input Delays

Here is how this constraint applies for input delays:

- Clock Fall - The default is the rising edge or rising transition of a reference pin. If you specify clock fall, you must also specify the name of the clock.
- Add Delay - Use this option to capture information about multiple paths leading to an input port relative to different clocks or clock edges.

For example, `set_input_delay 5.0 -max -rise -clock phi1 {A}` removes all maximum rise input delay from A, because the -add_delay option is not specified. Other input delays with different clocks or with -clock_fall are removed.

In this example, the -add_delay option is specified as `set_input_delay 5.0 -max -rise -clock phi1 -add_delay {A}`. If there is an input maximum rise delay for A relative to clock phi1 rising edge, the larger value is used. The smaller value does not result in critical timing for maximum delay. For minimum delay, the smaller value is used. If there is maximum rise input delay relative to a different clock or different edge of the same clock, it remains with the new delay.

Output Delays

Here is how this constraint applies for output delays:

- Clock Fall - If you specify clock fall, you must also specify the name of the clock.
- Add Delay - By using this option, you can capture information about multiple paths leading from an output port relative to different clocks or clock edges.

For example, the `set_output_delay 5.0 -max -rise -clock phi1 {OUT1}` command removes all maximum rise output delays from OUT1, because the `-add_delay` option is not specified. Other output delays with a different clock or with the `-clock_fall` option are removed.

In this example, the `-add_delay` option is specified: `set_output_delay 5.0 -max -rise -clock phi1 -add_delay {Z}`. If there is an output maximum rise delay for Z relative to the clock phi1 rising edge, the larger value is used. The smaller value does not result in critical timing for maximum delay. For minimum delay, the smaller value is used. If there is a maximum rise output delay relative to a different clock or different edge of the same clock, it remains with the new delay.

Priority of Multiple I/O Constraints

You can specify multiple input and output delays constraints for the same I/O port. This is useful for cases where a port is driven by or feeds multiple clocks. The priority of a constraint and its use in your design is determined by a few factors:

- The software applies the tightest constraint for a given clock edge, and ignores all others. All applicable constraints are reported in the timing report.
- You can apply I/O constraints on three levels, with the most specific overriding the more global:
 - Global (top-level netlist), for all inputs and outputs
 - Port-level, for the whole bus
 - Bit-level, for single bits

If there are two bit constraints and two port constraints, the two bit constraints override the two port constraints for that bit. The other bits get the two port constraints. For example, take the following constraints:

```
a[3:0]3 clk1:r
a[3:0]3 clk2:r
a[0]2 clk1:r
```

In this case, port a[0] only gets one constraint of 2 ns. Ports a[1], a[2], and a[3] get two constraints of 3 ns each.

- If at any given level (bit, port, global) there is a constraint with a reference clock specified, then any constraint without a reference clock is ignored. In this example, the 1 ns constraint on port a[0] is ignored.

```
a[0]2 clk1:r
a[0]1
```

See Also

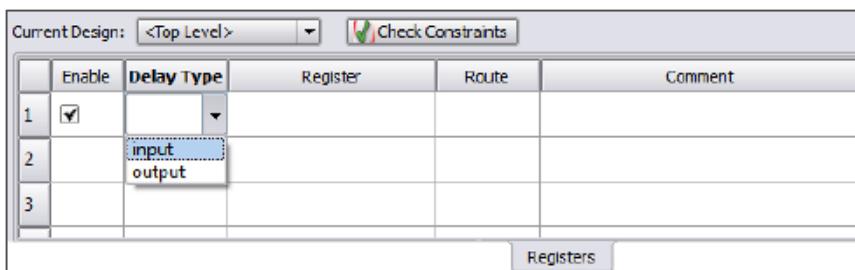
For equivalent Tcl syntax, see:

- [set_input_delay](#), on page 371
- [set_output_delay](#), on page 384

For information about all SCOPE panels, see [SCOPE Tabs](#), on page 295.

Registers

This panel lets the advanced user add delays to paths feeding into/out of registers, in order to further constrain critical paths. You use this constraint to speed up the paths feeding a register. See [set_reg_input_delay](#), on page 387, and [set_reg_output_delay](#), on page 388 for the equivalent Tcl commands.

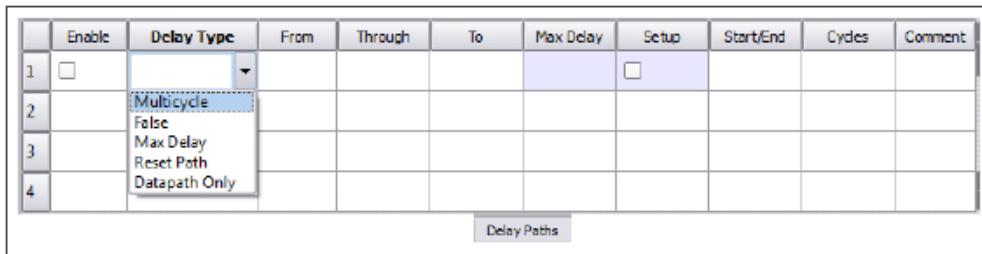


The Registers SCOPE panel includes the following fields:

Field	Description
Enabled	(Required) Turn this on to enable the constraint.
Delay Type	(Required) Specifies whether the delay is an input or output delay.
Register	(Required) Specifies the name of the register. If you have initialized a compiled design, you can choose from the pull-down list.
Route	(Required) Improves the speed of the paths to or from the register by the given number of nanoseconds. The value shrinks the effective period for the constrained registers without affecting the clock period that is forward-annotated to the place-and-route tool.
Comment	Lets you enter comments that are included in the constraints file.

Delay Paths

Use the Delay Paths panel to define the timing exceptions.



The Path Delay panel includes the following options:

Field	Description
Delay Type	<p>Specifies the type of delay path you want the synthesis tool to analyze. Choose one of the following types:</p> <ul style="list-style-type: none"> • Multicycle • False • Max Delay • Reset Path • Datapath Only
From	<p>Starting point for the path. From points define timing start points and can be defined for clocks (c:), registers (i:), top-level input or bi-directional ports (p:), black box output pins (i:) or sequential cell clock pins. For details, see the following:</p> <ul style="list-style-type: none"> • Defining From/To/Through Points for Timing Exceptions • Naming Rule Syntax Commands, on page 388
Through	<p>Specifies the intermediate points for the timing exception. Intermediate points can be combinational nets (n:), hierarchical ports (t:), or instantiated cell pins (t:). If you click the arrow in a column cell, you open the Product of Sums (POS) interface where you can set through constraints. For details, see the following:</p> <ul style="list-style-type: none"> • Product of Sums Interface • Defining From/To/Through Points for Timing Exceptions • Naming Rule Syntax Commands, on page 388
To	<p>Ending point of the path. To points must be timing end points and can be defined for clocks (c:), registers (i:), top-level output or bi-directional ports (p:), or black box input pins (i:). For details, see the following:</p> <ul style="list-style-type: none"> • Defining From/To/Through Points for Timing Exceptions • Naming Rule Syntax Commands, on page 388
Max Delay	Specifies the maximum delay value for the specified path in nanoseconds.

Field	Description
Setup	Specifies the setup (maximum delay) calculations used for specified path.
Start/End	Used for multicycle paths with different start and end clocks. This option determines the clock period to use for the multiplicand in the calculation for clock distance. If you do not specify a start or end clock, the end clock is the default.
Cycles	Specifies the number of cycles required for the multicycle path.

See Also

- For equivalent Tcl syntax, see:
 - [set_multicycle_path](#), on page 380
 - [set_false_path](#), on page 368
 - [set_max_delay](#), on page 374
 - [reset_path](#), on page 349
 - [set_datapathonly_delay](#), on page 365
- For more information on timing exception constraints and how the tool resolves conflicts, see:
 - [Delay Path Timing Exceptions](#), on page 325
 - [Conflict Resolution for Timing Exceptions](#), on page 338
- For information about all SCOPE panels, see [SCOPE Tabs](#), on page 295.

Attributes

You can assign attributes directly in the editor.

	Enabled	Object Type	Object	Attribute	Value	Val Type
1	<input checked="" type="checkbox"/>	output_port	<global>	syn_noclockbuf		
2	<input checked="" type="checkbox"/>			syn_clean_reset		
3	<input checked="" type="checkbox"/>			syn_dspstyle		
4	<input checked="" type="checkbox"/>			syn_edif_bit_format		
5	<input checked="" type="checkbox"/>			syn_edif_scalar_format		
				syn_forwar...onstraints		
				syn_multstyle		
				syn_netlist_hierarchy		
				syn_noarrayports		
				syn_noclockbuf		
				syn_ramstyle		

Here are descriptions for the Attributes columns:

Column	Description
Enabled	(Required) Turn this on to enable the constraint.
Object Type	Specifies the type of object to which the attribute is assigned. Choose from the pull-down list, to filter the available choices in the Object field.
Object	(Required) Specifies the object to which the attribute is attached. This field is synchronized with the Attribute field, so selecting an object here filters the available choices in the Attribute field.
Attribute	(Required) Specifies the attribute name. You can choose from a pull-down list that includes all available attributes for the specified technology. This field is synchronized with the Object field. If you select an object first, the attribute list is filtered. If you select an attribute first, the Synopsys FPGA synthesis tool filters the available choices in the Object field. You must select an attribute before entering a value. If a valid attribute does not appear in the pull-down list, simply type it in this field and then apply appropriate values.
Value	(Required) Specifies the attribute value. You must specify the attribute first. Clicking in the column displays the default value; a drop-down arrow lists available values where appropriate.

Val Type	Specifies the kind of value for the attribute. For example, string or boolean.
Description	Contains a one-line description of the attribute.
Comment	Lets you enter comments about the attributes.

Enter the appropriate attributes and their values, by clicking in a cell and choosing from the pull-down menu.

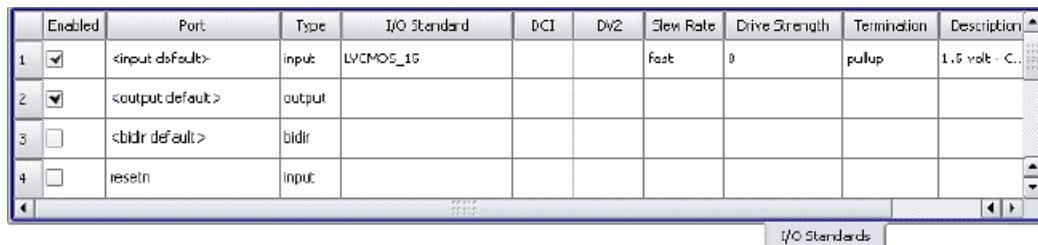
To specify an object to which you want to assign an attribute, you may also drag-and-drop it from the RTL or Technology view into a cell in the Object column. After you have entered the attributes, save the constraint file and add it to your project.

See Also

- For more information on specifying attributes, see [How Attributes and Directives are Specified, on page 10](#).
- For information about all SCOPE panels, see [SCOPE Tabs, on page 295](#).

I/O Standards

You can specify a standard I/O pad type to use in the design. Define an I/O standard for any port appearing in the I/O Standards panel.



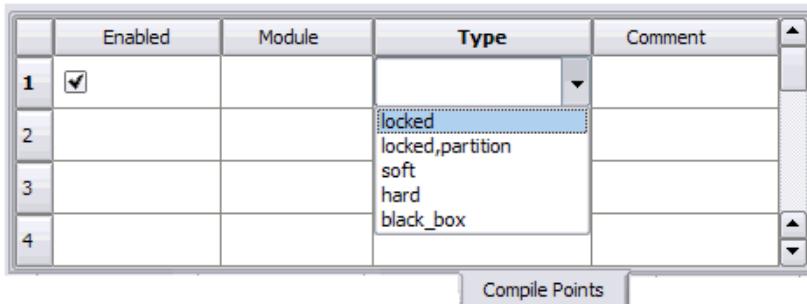
Field	Description
Enabled	(Required) Turn this on to enable the constraint, or off to disable a previous constraint.
Port	(Required) Specifies the name of the port. If you have initialized a compiled design, you can select a port name from the pull-down list. The first two entries let you specify global input and output delays, which you can then override with additional constraints on individual ports.
Type	(Required) Specifies whether the delay is an input or output delay.
I/O Standard	Supported I/O standards by Synopsys FPGA products. See Industry I/O Standards , on page 321 for a description of the standards.
DCI (Xilinx)	The values for these parameters are based on the selected I/O standard.
DV2 (Xilinx)	
Slew Rate	
Drive Strength	
Termination	
Power	
Schmitt	
Description	Describes the selected I/O Standard.
Comment	Enter comments about an I/O standard.

See Also

- The Tcl equivalent of this constraint is [define_io_standard](#).
- For information about all SCOPE panels, see [SCOPE Tabs, on page 295](#).

Compile Points

Use the Compile Points panel to specify compile points in your design, and to enable/disable them. This panel, available only if the device technology supports compile points, is used to define a top-level constraint file.

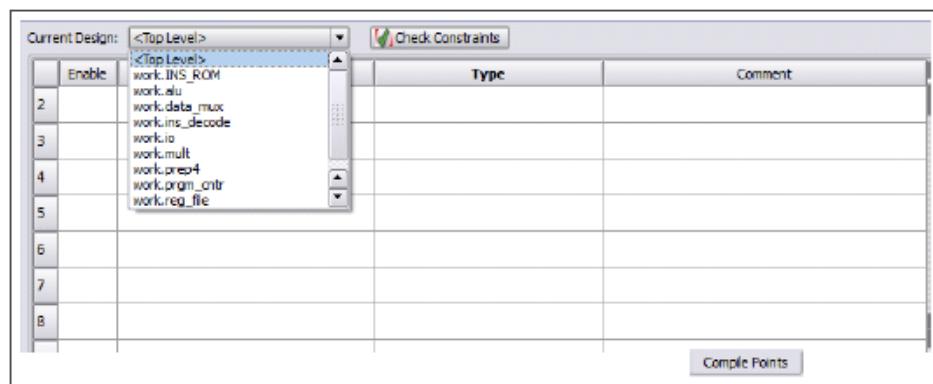


Here are the descriptions of the fields in the Compile Points panel.

Field	Description
Enabled	(Required) Turn this on to enable the constraint.
Module	(Required) Specifies the name of the compile-point module. You must specify a view module, with a v: prefix to identify the module as a view. For example: v:alu.
Type	<p>(Required) Specifies the type of compile point:</p> <ul style="list-style-type: none"> • locked (default) - no timing reoptimization is done on the compile point. The hierarchical interface is unchanged and an interface logic model is constructed for the compile point. • locked, partition - locked compile point, for which compile point information is forward annotated to the place and route tool. This mode provides place and route runtime advantages and allows for obtaining stable results for a completed design. • soft - compile point is included in the top-level synthesis, boundary optimizations can occur. • hard - compile point is included in the top-level synthesis, boundary optimizations can occur, however, the boundary remains unchanged. Although, the boundary is not modified, instances on both sides of the boundary can be modified using top-level constraints. • black_box - the compile point is treated as a black box. The software ignores the contents of the compile point that includes all instances in the module; only its ports exist for synthesis. Black box compile point modules only write port definitions to the netlist files. <p>For details, see Compile Point Types, on page 610 in the <i>User Guide</i>.</p>
Comment	Lets you enter a comment about the compile point.

Constraints for Compile Points

You can set constraints at the top-level or for modules to be used as the compile points from the Current Design pull-down menu shown below. Use the Compile Points tab to select compile points and specify their types.

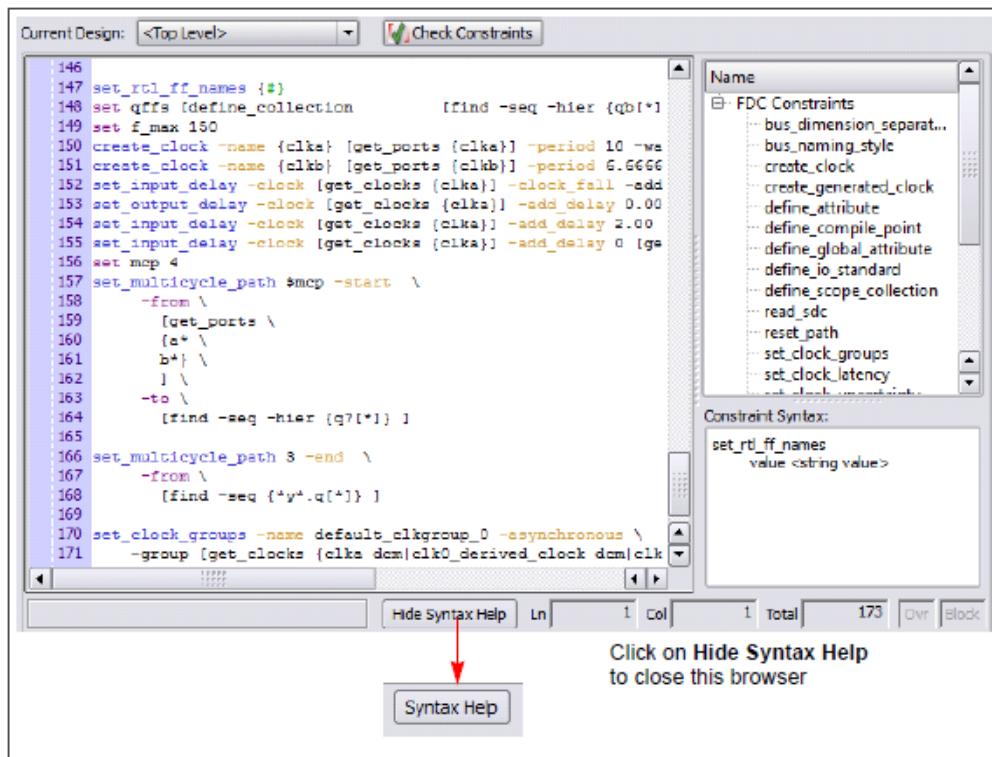


See Also

- The Tcl equivalent is [define_compile_point](#).
- For more information on compile points and using the Compile Points panel, see [Synthesizing Compile Points, on page 626](#) in the *User Guide*.
- For information about all SCOPE panels, see [SCOPE Tabs, on page 295](#).

TCL View

The **TCL View** is an advanced text file editor for defining FPGA timing and design constraints.



This text editor provides the following capabilities:

- Uses dynamic keyword expansion and tool tips for commands that
 - Automatically completes the command from a popup list
 - Displays complete command syntax as a tool tip
 - Displays parameter options for the command from a popup list
 - Includes a keyword command syntax help
- Checks command syntax and uses color indicators that
 - Validate commands and command syntax
 - Identifies FPGA design constraints and SCOPE legacy constraints

- Allows for standard editor commands, such as copy, paste, comment/un-comment a group of lines, and highlighting of keywords

For information on how to use this Tcl text editor, see [Using the TCL View of SCOPE GUI, on page 216](#).

See Also

- For Tcl timing constraint syntax, see [Timing Constraints, on page 342](#).
- For Tcl design constraint syntax, see [Design Constraints, on page 391](#).
- You can also use the SCOPE editor to set attributes. See [How Attributes and Directives are Specified, on page 10](#) for details.

Industry I/O Standards

The synthesis tool lets you specify a standard I/O pad type to use in your design. You can define an I/O standard for any port supported from the industry standard and proprietary I/O standards.

For industry I/O standards, see [Industry I/O Standards](#), on page 322.

For vendor-specific I/O standards, see:

- [Intel FPGA I/O Standards](#), on page 599
- [Lattice I/O Standards](#), on page 706
- [Microchip I/O Standards](#), on page 802
- [Xilinx I/O Standards](#), on page 875

Industry I/O Standards

The following table lists industry I/O standards.

I/O Standard	Description
AGP1X	Intel Corporation Accelerated Graphics Port
AGP2X	Intel Corporation Accelerated Graphics Port
BLVDS_25	Bus Differential Transceiver
CTT	Center Tap Terminated - EIA/JEDEC Standard JESD8-4
DIFF_HSTL_15_Class_I	1.5 volt - Differential High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
DIFF_HSTL_15_Class_II	1.5 volt - Differential High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
DIFF_HSTL_18_Class_I	1.8 volt - Differential High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-9A
DIFF_HSTL_18_Class_II	1.8 volt - Differential High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-9A
DIFF_SSTL_18_Class_II	1.8 volt - Differential Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-6
DIFF_SSTL_2_Class_I	2.5 volt - Pseudo Differential Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-9A
DIFF_SSTL_2_Class_II	2.5 volt - Pseudo Differential Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-9A
GTL	Gunning Transceiver Logic - EIA/JEDEC Standard JESD8-3
GTL+	Gunning Transceiver Logic Plus
GTL25	Gunning Transceiver Logic - EIA/JEDEC Standard JESD8-3
GTL+25	Gunning Transceiver Logic Plus
GTL33	Gunning Transceiver Logic - EIA/JEDEC Standard JESD8-3
GTL+33	Gunning Transceiver Logic Plus

I/O Standard	Description
HSTL_12	1.2 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_15_Class_II	1.5 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_18_Class_I	1.8 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_18_Class_II	1.8 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_18_Class_III	1.8 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_18_Class_IV	1.8 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_Class_I	1.5 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_Class_II	1.5 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_Class_III	1.5 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HSTL_Class_IV	1.5 volt - High Speed Transceiver Logic - EIA/JEDEC Standard JESD8-6
HyperTransport	2.5 volt - Hypertransport - HyperTransport Consortium

I/O Standard	Description
LVCMOS_12	1.2 volt - EIA/JEDEC Standard JESD8-16
LVCMOS_15	1.5 volt - EIA/JEDEC Standard JESD8-7
LVCMOS_18	1.8 volt - EIA/JEDEC Standard JESD8-7
LVCMOS_25	2.5 volt - EIA/JEDEC Standard JESD8-5
LVCMOS_33	3.3 volt CMOS - EIA/JEDEC Standard JESD8-B
LVCMOS_5	5.0 volt CMOS
LVDS	Differential Transceiver - ANSI/TIA/EIA-644-95
LVDSEXT_25	Differential Transceiver
LVPECL	Differential Transceiver - EIA/JEDEC Standard JESD8-2
LVTTL	3.3 volt TTL - EIA/JEDEC Standard JESD8-B
MINI_LVDS	Mini Differential Transceiver
PCI33	3.3 volt PCI 33MHz - PCI Local Bus Spec. Rev. 3.0 (PCI Special Interest Group)
PCI66	3.3 volt PCI 66MHz - PCI Local Bus Spec. Rev. 3.0 (PCI Special Interest Group)
PCI-X_133	3.3 volt PCI-X - PCI Local Bus Spec. Rev. 3.0 (PCI Special Interest Group)
PCML	3.3 volt - PCML
PCML_12	1.2 volt - PCML
PCML_14	1.4 volt - PCML
PCML_15	1.5 volt - PCML
PCML_25	2.5 volt - PCML
RSDS	Reduced Swing Differential Signalling
SSTL_18_Class_I	1.8 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-15
SSTL_18_Class_II	1.8 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-15
SSTL_2_Class_I	2.5 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-9B
SSTL_2_Class_II	2.5 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-9B
SSTL_3_Class_I	3.3 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-8
SSTL_3_Class_II	3.3 volt - Stub Series Terminated Logic - EIA/JEDEC Standard JESD8-8
ULVDS_25	Differential Transceiver

Delay Path Timing Exceptions

For details about the following path types, see:

- [Multicycle Paths](#), on page 325
- [False Paths](#), on page 328

Multicycle Paths

Multicycle paths lets you specify paths with multiple clock cycles. The following table defines the parameters for this constraint. For the equivalent Tcl constraints, see [set_multicycle_path](#), on page 380. This section describes the following:

- [Multi-cycle Path with Different Start and End Clocks](#), on page 325
- [Multicycle Path Examples](#), on page 326

Multi-cycle Path with Different Start and End Clocks

The start/end option determines the clock period to use for the multiplicand in the calculation for required time. The following table describes the behavior of the multi-cycle path constraint using different start and end clocks. In all equations, n is number of clock cycles, and *clock_distance* is the default, single-cycle relationship between clocks that is calculated by the tool.

Basic required time for a multi-cycle path	$\text{clock_distance} + [(n-1) * \text{end_clock_period}]$
Required time with no end clock defined	$\text{clock_distance} + [(n-1) * \text{global_period}]$
Required time with -start option defined	$\text{clock_distance} + [(n-1) * \text{start_clock_period}]$
Required time with no start clock defined	$\text{clock_distance} + [(n-1) * \text{global_period}]$

If you do not specify a start or end option, by default the end clock is used for the constraint. Here is an example:

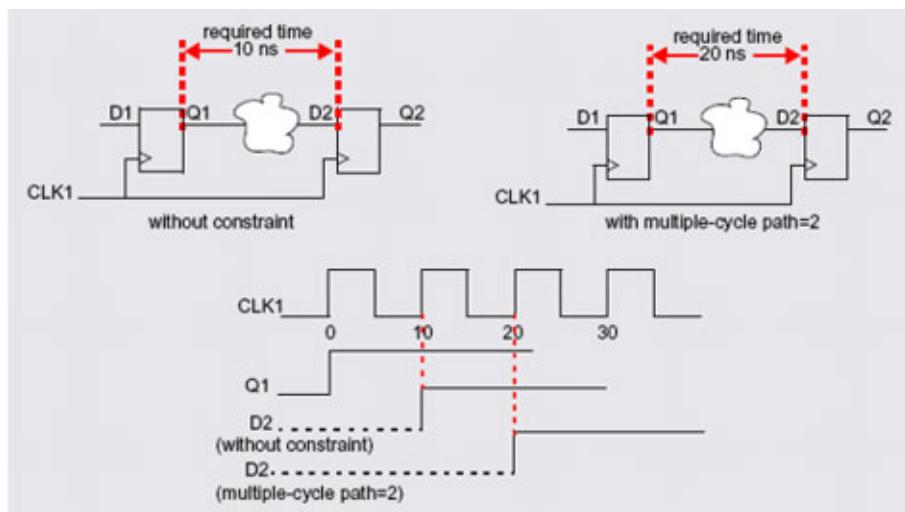
	Enabled	Delay Type	From	To	Through	Start/End	Cycles	Max Delay(ns)	Comment
1	<input checked="" type="checkbox"/>	Multicycle				End			
2						Start			
3						End			
4									

Delay Paths

Multicycle Path Examples

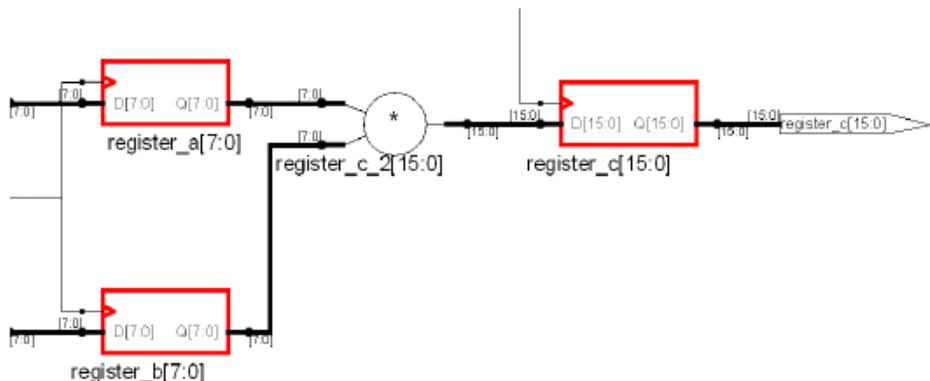
Multicycle Path Example 1

If you apply a multicycle path constraint from D1 to D2, the allowed time is #cycles x normal time between D1 and D2. In the following figure, CLK1 has a period of 10 ns. The data in this path has only one clock cycle before it must reach D2. To allow more time for the signal to complete this path, add a multiple-cycle constraint that specifies two clock cycles (10 x 2 or 20 ns) for the data to reach D2.



Multicycle Path Example 2

The design has a multiplier that multiplies signal_a with signal_b and puts the result into signal_c. Assume that signal_a and signal_b are outputs of registers register_a and register_b, respectively. The RTL view for this example is shown below. On clock cycle 1, a state machine enables an input enable signal to load signal_a into register_a and signal_b into register_b. At the beginning of clock cycle 2, the multiply begins. After two clock cycles, the state machine enables an output_enable signal on clock cycle 3 to load the result of the multiplication (signal_c) into an output register (register_c).



The design frequency goal is 50 MHz (20 ns) and the multiply function takes 35 ns, but it is given 2 clock cycles. After optimization, this 35 ns path is normally reported as a timing violation because it is more than the 20 ns clock-cycle timing goal. To avoid reporting the paths as timing violations, use the SCOPE window to set 2-cycle constraints (From column) on register_a and register_b, or include the following in the timing constraint file:

```

# Paths from register_a use 2 clock cycles
set_multicycle_path -from register_a 2

# Paths from register_b use 2 clock cycles
set_multicycle_path -from register_b 2

```

Alternatively, you can specify a 2-cycle SCOPE constraint (To column) on register_c, or add the following to the constraint file:

```

# Paths to register_c use 2 clock cycles
set_multicycle_path -to register_c 2

```

False Paths

You use the Delay Paths constraint to specify clock paths that you want the synthesis tool to ignore during timing analysis and assign low (or no) priority during optimization. The equivalent Tcl constraint is described in [set_false_path](#), on page 368.

This section describes the following:

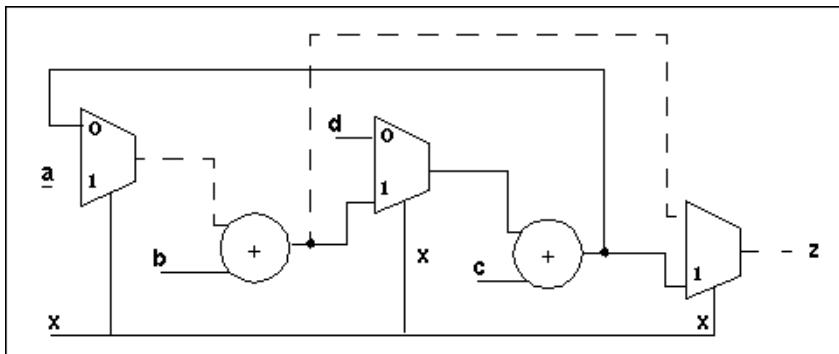
- [Types of False Paths](#), on page 328
- [False Path Constraint Examples](#), on page 329

Types of False Paths

A false path is a path that is not important for timing analysis. There are two types of false paths:

- Architectural false paths

These are false paths that the designer is aware of, like an external reset signal that feeds internal registers but which is synchronized with the clock. The following example shows an architectural false path where the primary input *x* is always 1, but which is not optimized because the software does not optimize away primary inputs.



- Code-introduced false paths

These are false paths that you identify after analyzing the schematic.

False Path Constraint Examples

In this example, the design frequency goal is 50 MHz (20ns) and the path from register_a to register_c is a false path with a large delay of 35 ns. After optimization, this 35 ns path is normally reported as a timing violation because it is more than the 20 ns clock-cycle timing goal. To lower the priority of this path during optimization, define it as a false path. You can do this in many ways:

- If all paths from register_a to any register or output pins are not timing-critical, then add a false path constraint to register_a in the SCOPE interface (From), or put the following line in the timing constraint file:

```
#Paths from register_a are ignored  
set_false_path -from {i:register_a}
```

- If all paths to register_c are not timing-critical, then add a false path constraint to register_c in the SCOPE interface (To), or include the following line in the timing constraint file:

```
#Paths to register_c are ignored  
set_false_path -to {i:register_c}
```

- If only the paths between register_a and register_c are not timing-critical, add a From/To constraint to the registers in the SCOPE interface (From and To), or include the following line in the timing constraint file:

```
#Paths to register_c are ignored  
set_false_path -from {i:register_a} -to {i:register_c}
```

Specifying From, To, and Through Points

The following section describes from, to, and through points for timing exceptions specified by the multicycle paths, false paths, and max delay paths constraints.

- [Timing Exceptions Object Types](#), on page 330
- [From/To Points](#), on page 330
- [Through Points](#), on page 332
- [Product of Sums Interface](#), on page 333
- [Clocks as From/To Points](#), on page 336

Timing Exceptions Object Types

Note the following timing exceptions for object types:

- Timing exceptions must contain the type of object in the constraint specification. You must explicitly specify an object type, n: for a net, or i: for an instance, in the instance name parameter of all timing exceptions. For example:

```
set_multicycle_path -from {i:inst2.lowreg_output[7] }  
                      -to {i:inst1.DATA0[7]} 2
```

If you use the SCOPE GUI to specify timing exceptions, it automatically attaches the object type qualifier to the object name.

- When defining constraints for the fdc file from the Tcl View window in SCOPE, it is recommended you use get_* as the object type qualifier. For example, use get_ports, get_nets, get_pins, or get_cells instead of p:, n:, t:, or i:.

From/To Points

From specifies the starting point for the timing exception. To specifies the ending point for the timing exception. When you specify an object, use the appropriate prefix (see [syn_black_box](#), on page 149) to avoid confusion. The following table lists the objects that can serve as starting and ending points:

From Points	To Points
Clocks. See Clocks as From/To Points , on page 336 for more information.	Clocks. See Clocks as From/To Points , on page 336 for more information.
Registers	Registers
Top-level input or bi-directional ports	Top-level output or bi-directional ports
Instantiated library primitive cells (gate cells)	Instantiated library primitive cells (gate cells)
Black box outputs	Black box inputs

You can specify multiple from points in a single exception. This is most common when specifying exceptions that apply to all the bits of a bus. For example, you can specify constraints From A[0:15] to B - in this case, there is an exception, starting at any of the bits of A and ending on B.

Similarly, you can specify multiple to points in a single exception. If you specify both multiple starting points and multiple ending points such as From A[0:15] to B[0:15], there is actually an exception from any start point to any end point. In this case, the exception applies to all $16 * 16 = 256$ combinations of start/end points.

Through Points

Through points are limited to nets, hierarchical ports, and pins of instantiated cells. There are many ways to specify these constraints.

- Single Point
- Single List of Points
- Multiple Through Points
- Multiple Through Lists

You define these constraints in the appropriate SCOPE panels, or in the POS GUI (see [Product of Sums Interface, on page 333](#)). When a port and net have the same name, preface the name of the through point with n: for nets or t: for hierarchical ports. For example, you can specify n:regs_mem[2] or t:dmux.bdpol. The n: prefix must be specified to identify nets; otherwise, the associated timing constraint will not be applied for valid nets.

Single Point

You can specify a single through point. In this case, the constraint is applied to any path that passes through net regs_mem[2] as follows:

```
set_false_path -through n:regs_mem[2]
set_false_path -through [get_nets {regs_mem[2]}]
```

Single List of Points

If you specify a list of through points, the through option behaves as an OR function and applies to any path that passes through any of the points in the list. In the following example, the constraint is applied to any path through regs_mem[2] OR prgcntr.pc[7] OR dmux.alub[0] with a maximum delay value of 5 ns (-max 5):

```
set_max_delay
-through {t:regs_mem[2] t:prgcntr.pc[7] t:dmux.alub[0]} 5
```

Multiple Through Points

You can specify multiple points for the same constraint by preceding each point with the -through option. In the following example, the constraint operates as an AND function and applies to paths through `regs_mem[2]` AND `prgcntr.pc[7]` AND `dmux.alub[0]:`

```
set_max_delay
-through t:regs_mem[2]
-through t:prgcntr.pc[7]
-through t:dmux.alub[0] 5
```

Multiple Through Lists

If you specify multiple -through lists, the constraint is applied as an AND/OR function and is applied to the paths through all points in the lists. The following constraint applies to all paths that pass through nets { A_1 or A_2 or... A_n } AND nets { B_1 or B_2 or B_3 }:

```
set_false_path -through {n:A1 n:A2...n:An} -through {n:B1 n:B2 n:B3}
```

In this example,

```
set_multicycle_path
-through {n:net1 n:net2}
-through {n:net3 n:net4} 2
```

all paths that pass through the following nets are constrained at 2 clock cycles:

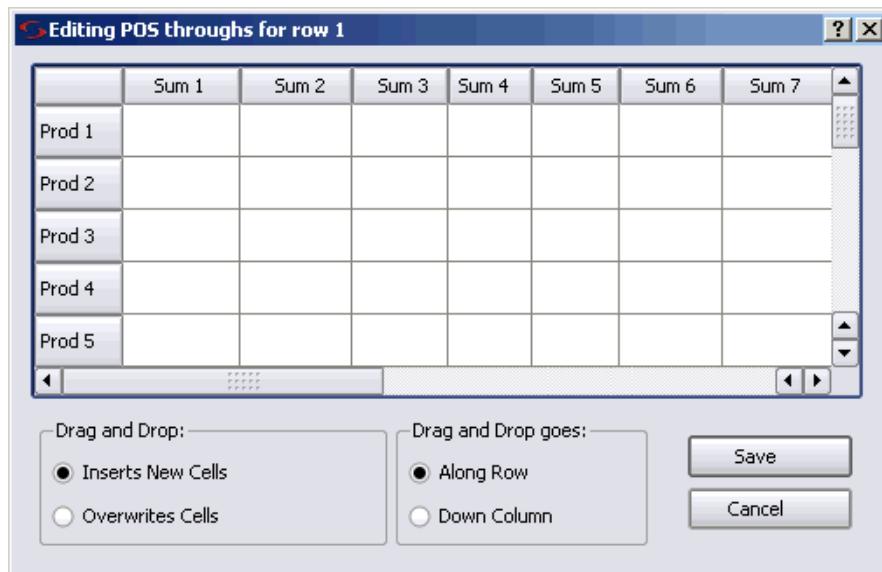
```
net1 AND net3
OR net1 AND net4
OR net2 AND net3
OR net2 AND net4
```

Product of Sums Interface

You can use the SCOPE GUI to format -through points for nets with multicycle path, false path, and max delay path constraints in the Product of Sums (POS) interface of the SCOPE editor. You can also manually specify constraints that use the -through option. For more information, see [Defining From/To/Through Points for Timing Exceptions, on page 221](#) in the *User Guide*.

The POS interface is accessible by clicking the arrow in a Through column cell in the following SCOPE panels:

- Multi-Cycle Paths
- False Paths
- Delay Paths



Field	Description
Prod 1, 2, etc.	Type the first net name in a cell in a Prod row, or drag the net from a HDL Analyst view into the cell. Repeat this step along the same row, adding other nets in the Sum columns. The nets in each row form an OR list.
Sum 1, 2, etc.	Type the first net name in the first cell in a Sum column, or drag the net from a HDL Analyst view into the cell. Repeat this step down the same Sum column. The nets in each column form an AND list.

Drag and Drop Goes	Along Row - places objects in multiple Sum columns, utilizing only one Prod row. Down Column - places objects in multiple Prod rows, utilizing only one Sum column.
Drag and Drop	Inserts New Cells - New cells are created when dragging and dropping nets. Overwrites Cells - Existing cells are overwritten when dragging and dropping nets.
Save/Cancel	Saves or cancels your session.

Clocks as From/To Points

You can specify clocks as from/to points in your timing exception constraints. Here is the syntax:

```
set_timing_exception -from | -to {c:clock_name [:edge]}
```

where

- *timing_exception* is one of the following constraint types: multicycle path, false path, or max delay
- **c:clock_name:edge** is the name of the clock and clock edge (r or f). If you do not specify a clock edge, by default both edges are used.

See the following sections for details and examples on each timing exception.

Multicycle Path Clock Points

When you specify a clock as a from or to point, the multicycle path constraint applies to all registers clocked by the specified clock.

The following constraint allows two clock periods for all paths from the rising edge of the flip-flops clocked by clk1:

```
set_multicycle_path -from {c:clk1:r} 2
```

You cannot specify a clock as a through point. However, you can set a constraint from or to a clock and through an object (net, pin, or hierarchical port). The following constraint allows two clock periods for all paths to the falling edge of the flip-flops clocked by clk1 and through bit 9 of the hierarchical net:

```
set_multicycle_path -to {c:clk1:f} -through (n:MYINST.mybus2[9]} 2
```

False Path Clock Points

When you specify a clock as a from or to point, the false path constraint is set on all registers clocked by the specified clock. False paths are ignored by the timing analyzer. The following constraint disables all paths from the rising edge of the flip-flops clocked by clk1:

```
set_false_path -from {c:clk1:r}
```

You cannot specify a clock as a through point. However, you can set a constraint from or to a clock and through an object (net, pin, or hierarchical port). The following constraint disables all paths to the falling edge of the flip-flops clocked by clk1 and through bit 9 of the hierarchical net.

```
set_false_path -to {c:clk1:f} -through {n:MYINST.mybus2[9]}
```

Path Delay Clock Points

When you specify a clock as a from or to point for the path delay constraint, the constraint is set on all paths of the registers clocked by the specified clock. This constraint sets a max delay of 2 ns on all paths to the falling edge of the flip-flops clocked by clk1:

```
set_max_delay -to {c:clk1:f} 2
```

You cannot specify a clock as a through point, but you can set a constraint from or to a clock and through an object (net, pin, or hierarchical port). The next constraint sets a max delay of 0.2 ns on all paths from the rising edge of the flip-flops clocked by clk1 and through bit 9 of the hierarchical net:

```
set_max_delay -from {c:clk1:r} -through {n:MYINST.mybus2[9]}.2
```

Conflict Resolution for Timing Exceptions

The term *timing exceptions* refers to the false path, max path delay, and multicycle path timing constraints. When the tool encounters conflicts in the way timing exceptions are specified through the constraint file, the software uses a set priority to resolve these conflicts. Conflict resolution is categorized into four levels, meaning that there are four different tiers at which conflicting constraints can occur, with one being the highest. The table below summarizes conflict resolution for constraints. The sections following the table provide more details on how conflicts can occur and examples of how they are resolved.

Conflict Level	Constraint Conflict	Priority	For Details, see ...
1	Different timing exceptions set on the same object.	1 - False Path 2 - Path Delay 3 - Multi-cycle Path	Conflicting Timing Exceptions , on page 339.
2	Timing exceptions of the same constraint type, using different semantics (from/to/through).	1 - From 2 - To 3 - Through	Same Constraint Type with Different Semantics , on page 340.
3	Timing exceptions of the same constraint type using the same semantic, but set on different objects.	1 - Ports/Instances/Pins 2 - Clocks	Same Constraint and Semantics with Different Objects , on page 341.
4	Identical timing constraints, except constraint values differ.	Tightest, or most constricting constraint.	Identical Constraints with Different Values , on page 341.

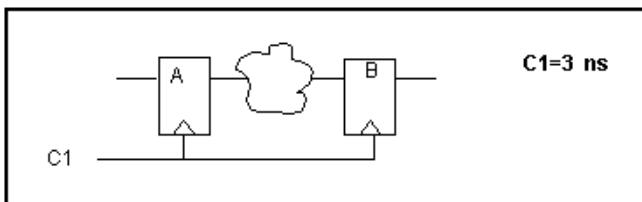
In addition to the four levels of conflict resolution for timing exceptions, there are priorities for the way the tool handles multiple I/O delays set on the same port and implicit and explicit false path constraints. For information on resolving these types of conflicts, see [Priority of Multiple I/O Constraints, on page 308.](#)

Conflicting Timing Exceptions

The first (and highest) level of resolution occurs when timing exceptions—false paths, max path delay, or multicycle path constraints—conflict with each other. The tool follows this priority for applying timing exceptions:

1. False Path
2. Path Delay
3. Multicycle Path

For example:



```
set_false_path -from {c:C1:r}  
set_max_delay -from {i:A} -to {i:B} 10  
set_multicycle_path -from {i:A} -to {i:B} 2
```

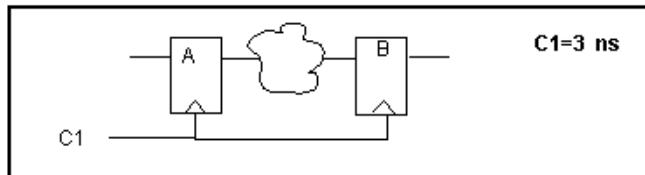
These constraints are conflicting because the path from A to B has three different constraints set on it. When the tool encounters this type of conflict, the false path constraint is honored. Because it has the highest priority of all timing exceptions, `set_false_path` is applied and the other timing exceptions are ignored.

Same Constraint Type with Different Semantics

The second level of resolution occurs when conflicts between timing exceptions that are of the same constraint type, use different semantics (from/to/through). The priority for these constraints is as follows:

1. From
2. To
3. Through

If there are two multicycle constraints set on the same path, one specifying a from point and the other specifying a to point, the constraint using -from takes precedence, as in the following example.



```
set_multicycle_path -from {i:A} 3  
set_multicycle_path -to {i:B} 2
```

In this case, the tool uses:

```
set_multicycle_path -from {i:A} 3
```

The other constraint is ignored even though it sets a tighter constraint.

Same Constraint and Semantics with Different Objects

The third level resolves timing exceptions of the same constraint type that use the same semantic, but are set on different objects. The priority for design objects is as follows:

1. Ports/Instances/Pins
2. Clocks

If the same constraints are set on different objects, the tool ignores the constraint set on the clock for that path.

```
set_multicycle_path -from {i:mac1.datax[0]} -start 4  
set_multicycle_path -from {c:clk1:r} 2
```

In the example above, the tool uses the first constraint set on the instance and ignores the constraint set on the clock from i:mac1.datax[0], even though the clock constraint is tighter.

For details on how the tool prioritizes multiple I/O delays set on the same port or implicit and explicit false path constraints, see [Priority of Multiple I/O Constraints, on page 308](#).

Identical Constraints with Different Values

Where timing constraints are identical except for the constraint value, the tightest or most constricting constraint takes precedence. In the following example, the tool uses the constraint specifying two clock cycles:

```
set_multicycle_path -from {i:special_regs.trisa[7:0]} 2  
set_multicycle_path -from {i:special_regs.trisa[7:0]} 3
```

Timing Constraints

The FPGA synthesis tools support FPGA timing constraints for a subset of the clock definition, I/O delay, and timing exception constraints.

The remainder of this section describes the constraint file syntax for the following FPGA timing constraints in the FPGA synthesis tools.

create_clock	create_generated_clock
reset_path	set_case_analysis
set_clock_groups	set_clock_latency
set_clock_route_delay	set_clock_uncertainty
set_datapathonly_delay	set_false_path
set_input_delay	set_max_delay
set_multicycle_path	set_output_delay
set_reg_input_delay	set_reg_output_delay

Note: When adding comments for constraints, use standard Tcl syntax conventions. Otherwise, invalid specifications can cause the constraint to be ignored. The (#) comment must begin on a new line or needs to be preceded by a (;), if the comment is on the same line as the constraint. For example:

```
create_clock -period 10 [get_ports CLK]; # comment text  
# comment text  
set_clock_groups -asynchronous -group  
MMCM_module|clk100_90_MMCM_derived_clock_CLKIN1
```

create_clock

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

Syntax

The supported syntax for the create_clock constraint is:

```
create_clock  
  -name clockName [-add] {objectList} |  
    -name clockName [-add] [{objectList}]  
      [-name clockName [-add]] {objectList}  
  -period value  
  [-waveform {riseValue fallValue}]  
  [-disable]  
  [-comment commentString]
```

Arguments

-name <i>clockName</i>	Specifies the name for the clock being created, enclosed in quotation marks or curly braces. If this option is not used, the clock gets the name of the first clock source specified in the <i>objectList</i> option. If you do not specify the <i>objectList</i> option, you must use the -name option, which creates a virtual clock not associated with a port, pin, or net. You can use both the -name and <i>objectList</i> options to give the clock a more descriptive name than the first source pin, port, or net. If you specify the -add option, you must use the -name option and the clocks with the same source must have different names.
-add	Specifies whether to add this clock to the existing clock or to overwrite it. Use this option when multiple clocks must be specified on the same source for simultaneous analysis with different clock waveforms. When you specify this option, you must also use the -name option.
-period <i>value</i>	Specifies the clock period in nanoseconds. This is the minimum time over which the clock waveform repeats. The <i>value</i> type must be greater than zero.

-waveform	Specifies the rise and fall edge times for the clock waveforms of the clock in nanoseconds, over an entire clock period. The first time is a rising transition, typically the first rising transition after time zero. There must be two edges, and they are assumed to be rise followed by fall. The edges must be monotonically increasing. If you do not specify this option, a default waveform is assumed, which has a rise edge of 0.0 and a fall edge of <i>periodValue</i> /2.
objectList	Clocks can be defined on the following objects: pins, ports, and nets. The FPGA synthesis tools support nets and instances, where instances have only one output (for example, BUFGs).
-disable	Disables the constraint.
-comment textString	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Examples

Refer to the following examples.

Example 1

A clock named clk_in1 is created for port clk_in1 that uses a period of 10 with rising edge of 0 and falling edge of 5.

```
create_clock -name {clk_in1} -period 10 [get_ports {clk_in1}]
```

Example 2

A clock named clk is created for port clk_in that uses a period of 10.0 with rising edge of 5.0 and falling edge of 9.5.

```
create_clock -name {clk} -period 10 -waveform {5.0 9.5}
[get_ports {clk_in}]
```

Example 3

A virtual clock named CLK is created that uses a period of 12 with a rising edge of 0.0 and falling edge of 6.0.

```
create_clock -name {CLK} -period 12
```

create_generated_clock

Creates a generated clock object.

Syntax

The supported syntax for the create_generated_clock constraint is:

```
create_generated_clock
  -name clockName [-add]] | {clockObject}
  -source masterPinName
  [-master_clock clockName]
  [-divide_by integer | -multiply_by integer [-duty_cycle value]]
  [-invert]
  [-edges {edgeList}]
  [-edge_shift {edgeShiftList}]
  [-combinational]
  [-disable]
  [-comment commentString]
```

Arguments

-name <i>clockName</i>	Specifies the name of the generated clock. If this option is not used, the clock gets the name of the first clock source specified in the -source option (<i>clockObject</i>). If you specify the -add option, you must use the -name option and the clocks with the same source must have different names. <i>Xilinx Virtex-7 devices only</i> When the -name option is used only with an object, you can rename the derived clock on the given object. The object must be an MMCM/PLL clock output pin or the output pin of a register that is used as a clock divider. For example: <code>create_generated_clock -name {clkfx} [get_pins {dcm_inst.PLL_ADV_inst.CLKOUT0}]</code>
-add	Specifies whether to add this clock to the existing clock or to overwrite it. Use this option when multiple generated clocks must be specified on the same source, because multiple clocks fan into the master pin. Ideally, one generated clock must be specified for each clock that fans into the master pin. If you specify this option, you must also use the -name and -master_clock options.

<i>clockObject</i>	The first clock source specified in the -source option in the absence of <i>clockName</i> . Clocks can be defined on pins, ports, and nets. The FPGA synthesis tools support nets and instances, where instances have only one output (for example, BUFGs).
-source <i>masterPinName</i>	Specifies the master clock pin, which is either a master clock source pin or a fanout pin of the master clock driving the generated clock definition pin. The clock waveform at the master pin is used for deriving the generated clock waveform.
-master_clock <i>clockName</i>	Specifies the master clock to be used for this generated clock, when multiple clocks fan into the master pin.
-divide_by <i>integer</i>	Specifies the frequency division factor. If the <i>divideFactor</i> value is 2, the generated clock period is twice as long as the master clock period.
-multiply_by <i>integer</i>	Specifies the frequency multiplication factor. If the <i>multiplyFactor</i> value is 3, the generated clock period is one-third as long as the master clock period.
-duty_cycle <i>percent</i>	Specifies the duty cycle, as a percentage, if frequency multiplication is used. Duty cycle is the high pulse width. Note: This option is valid only when used with the -multiply_by option.
-invert	Inverts the generated clock signal (in the case of frequency multiplication and division).
-edges <i>edgeList</i>	Specifies a list of integers that represents edges from the source clock that are to form the edges of the generated clock. The edges are interpreted as alternating rising and falling edges and each edge must not be less than its previous edge. The number of edges must be set to 3 to make one full clock cycle of the generated clock waveform. For example, 1 represents the first source edge, 2 represents the second source edge, and so on.
-edge_shift <i>edgeShiftList</i>	Specifies a list of floating point numbers that represents the amount of shift, in nanoseconds, that the specified edges are to undergo to yield the final generated clock waveform. The number of edge shifts specified must be equal to the number of edges specified. The values can be positive or negative; positive indicating a shift later in time, while negative indicates a shift earlier in time. For example, 1 indicates that the corresponding edge is to be shifted by one library time unit.

-combinational	The source latency paths for this type of generated clock only includes the logic where the master clock propagates. The source latency paths do not flow through sequential element clock pins, transparent latch data pins, or source pins of other generated clocks.
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Examples

Refer to the following examples.

Example 1

A frequency of -divide_by 2 is used for the generated clock.

```
create_generated_clock -name {gen_clk} -source
[get_pins {DCM0.CLK0}] [get_pins {BUFGMUX_inst.O}] -divide_by 2
```

Example 2

A generated clock is created whose edges are 1, 3, and 5 of the master clock source. If the master clock period is 30 and the master waveform is {24 30}, then the generated clock period becomes 60 with waveform {24 54}.

```
create_generated_clock -name {genclk} -source
[get_ports {clk_in1}] [get_nets {dut.clk_out2}] -edges {1 3 5}
```

Example 3

This example shows the generated clock from the previous example with each derived edge shifted by 1 time unit. If the master clock period is 30 and the master waveform is {24 36}, then the generated clock period becomes 60 with waveform {25 55}.

```
create_generated_clock -name {genclk}
-source [get_ports {clk_in1}] [get_nets {dut.clk_out2}]
-edges {1 3 5} -edge_shift {1 1 1}
```

Example 4

This example shows the generated clock with the same edges as the master clock, where edge 2 is shifted by 0.8 time unit and edge 3 is shifted by -0.4 time unit. If the master clock period is 4 and the master waveform is {0 2}, then the generated clock period becomes 3.6 and the waveform is {0 2.8}.

```
create_generated_clock -name {genclk}
    -source [get_ports {clk_in1}] [get_nets {dut.clk_out2}]
    -edges {1 2 3} -edge_shift {0 0.8 -0.4}
```

reset_path

Resets the specified paths to single-cycle timing.

Syntax

The supported syntax for the reset_path constraint is:

```
reset_path [-setup]
[-from {objectList} |-rise_from riseFromClock | -fall_from fallFromClock]
[-through {objectList} [-through {objectList} ...] ]
[-to {objectList} |-rise_to riseToClock | -fall_to fallToClock]
[-disable]
[-comment commentString]
```

Arguments

-setup	Specifies that setup checking (maximum delay) is reset to single-cycle behavior.
-from	Specifies the names of objects to use to find path start points. The -from <i>objectList</i> includes: <ul style="list-style-type: none">• Clocks• Registers• Top-level input or bi-directional ports)• Black box outputs• Sequential cell clock pins When the specified object is a clock, all flip-flops, latches, and primary inputs related to that clock are used as path start points
-rise_from <i>riseFromClock</i>	Specifies to use the rising edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.
-fall_from <i>fallFromClock</i>	Specifies to use the falling edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-through	Specifies the intermediate points for the timing exception. The -through <i>objectList</i> includes:
	<ul style="list-style-type: none"> • Combinational nets • Hierarchical ports • Pins on instantiated cells • Cell instances
	By default, the through points are treated as an OR list. The constraint is applied if the path crosses any points in <i>objectList</i> . If more than one object is included, the objects must be enclosed either in quotation marks ("") or in braces ({}). If you specify the -through option multiple times, <i>reset_path</i> applies to the paths that pass through a member of each <i>objectList</i> . If you use the -through option in combination with the -from or -to options, <i>reset_path</i> applies only if the -from or -to and the -through conditions are satisfied.
-to	Specifies the names of objects to use to find path end points. The -to <i>objectList</i> includes:
	<ul style="list-style-type: none"> • Clocks • Registers • Top-level output or bi-directional ports • Black box inputs • Sequential cell data input pins <p>If a specified object is a clock, all flip-flops, latches, and primary outputs related to that clock are used as path end points.</p>
-rise_to <i>riseToClock</i>	Specifies to use the rising edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.
-fall_to <i>fallToClock</i>	Specifies to use the falling edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

set_case_analysis

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

Note: Currently `set_case_analysis` is only supported on the select input pin for BUFGMUX, BUFGMUX_1, and BUFGMUX_CTRL.

Syntax

The supported syntax for the `set_case_analysis` constraint is:

`set_case_analysis` *value* *portOrPinList*

For example:

```
set_case_analysis 1 [get_ports {IN1}]
```

Arguments

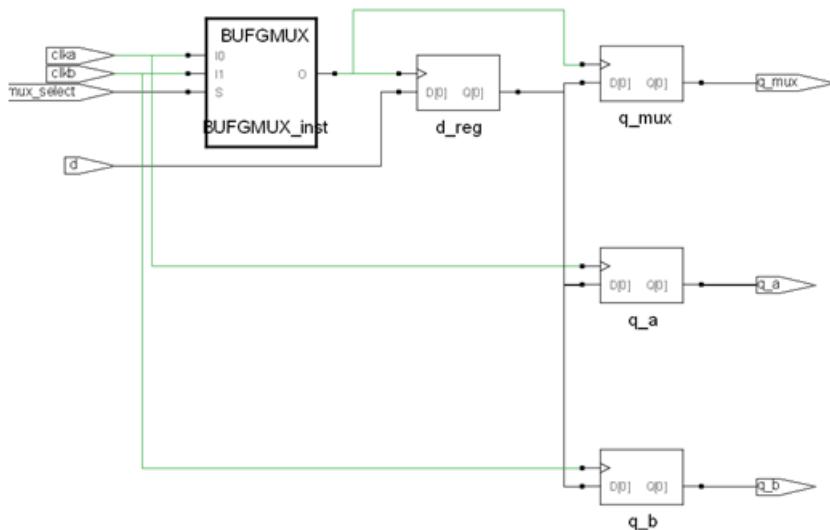
<i>value</i>	Specifies that the port or pin is at a constant 1 or 0 logic value. Valid values are <ul style="list-style-type: none">• 1 or one• 0 or zero
--------------	--

<i>portOrPinList</i>	Specifies the port or pin list.
----------------------	---------------------------------

Examples

The synthesis tool allows multiple clocks to propagate along a single net, either through `unate` logic or a mux for timing all possible clocks. In the following example, incorrect clock reporting can occur when both clocks are propagated through the mux.

Unate logic refers to inverters and buffers in clock trees that recognizes the positive or negative sense of the clock signal arriving at each register clock pin.



If both clocks are propagated, the timing report will contain the following timing paths:

Starting point: d_reg / Q

Ending point: q_mux / D

The start point is clocked by clka [rising] on pin C

The end point is clocked by clkb [rising] on pin C

This path is invalid because clka and clkb cannot exist on the net at the same time. The `set_case_analysis` constraint lets you select the clock to propagate through the mux.

This constraint specified with a query command allows clka to propagate through the mux:

```
set_case_analysis 0 [get_ports {mux_select}]
```

If you want to propagate clkb through the mux, use the following constraint:

```
set_case_analysis 1 [get_ports {mux_select}]
```

set_clock_groups

Specifies clock groups that are mutually exclusive or asynchronous with each other in a design. Clocks created with `create_clock` are considered synchronous as long as no `set_clock_groups` constraints specify otherwise. Paths between asynchronous clocks are not considered for timing analysis.

Clock grouping in the FPGA synthesis environment is inclusionary or exclusionary. For example, `clk2` and `clk3` can each be related to `clk1` without being related to each other.

Syntax

```
set_clock_groups
  -asynchronous | -physically_exclusive | -logically_exclusive
  [-name clockGroupname]
  -group {clockList} [-group {clockList} ... ]
  -derive
  [-disable]
  [-comment commentString]
```

Arguments

-asynchronous	Specifies that the clock groups are asynchronous to each other (the default assumes all clock groups are synchronous). Two clocks are asynchronous with respect to each other if they have no phase relationship at all.
-physically_exclusive	Specifies that the clock groups are physically exclusive to each other. An example is multiple clocks that are defined on the same source pin. Synthesis accepts this option, but treats it as <code>-asynchronous</code> .
-logically_exclusive	Specifies that the clock groups are logically exclusive to each other. An example is multiple clocks that are selected by a multiplexer, but might have coupling with each other in the design. Synthesis accepts this option, but treats it as <code>-asynchronous</code> .

-name <code>{clockGroupName}</code>	<p>Specifies a unique name for a clock grouping. This option allows you to easily identify specified clock groups, which are exclusive or asynchronous with all other clock groups in the design.</p> <p>Note: Use with Xilinx Virtex-7 devices, where the clock group names for this command are forward annotated to Vivado place and route. For Intel, this option is not forward annotated.</p>
-group { <code>clockList</code> }	<p>Specifies a space-separated list of clocks in <code>{clockList}</code> that are asynchronous to all other clocks in the design, or asynchronous to the clocks specified in other -group arguments in the same command.</p> <p>If you specify only one group, the clocks in that group are exclusive or asynchronous with all other clocks in the design. Whenever a new clock is created, it is automatically included in the default “other” group that includes all the other clocks in the design.</p> <p>If you specify -group multiple times in a single command execution, the listed clocks are only asynchronous with the clocks in the other groups specified in the same command. You can include a clock in only one group in a single command execution. To include a clock in multiple groups, use multiple <code>set_clock_groups</code> commands.</p> <p>Do not use commas between clock names in the list. See -group Option , on page 355.</p>
-derive	<p>Specifies that generated and derived clocks inherit the clock group of the parent clock. By default, a generated clock and its master clock are not in the same group when the exclusive or asynchronous clock groups are defined. The -derive option lets you override this behavior and allow generated or derived clocks to inherit the clock group of their parent source clock.</p> <p>For restrictions when using this option, see -derive Option , on page 355.</p>
-disable	Disables the constraint.
-comment <code>textString</code>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Restrictions

Be aware of the restrictions for the following `set_clock_groups` options:

-group Option

Do not insert commas between clock names when you use the `-group` option, because the tool treats the comma as part of the clock name. This is true for all constraints that contain lists. This means that if you specify the following constraint, the tool generates a warning that it cannot find `clk1,:`:

```
set_clock_groups -asynchronous -group {clk1, clk2}
```

-derive Option

Note the following:

- For Xilinx Virtex-6 and older technologies, always use the `-derive` option for source clocks specified with the `-group` option. You cannot explicitly use `set_clock_groups` (with or without `-derive`) for mapper derived or user-generated clocks.
- For Xilinx Virtex-7 and newer technologies, you can use `set_clock_groups` (with or without the `-derive` option) for source and generated/derived clocks. In this case, when you want derived clocks to inherit their source clock group, the `-derive` option is forward annotated as `get_clocks -include_generated_clocks`:

```
set_clock_groups -asynchronous -group [get_clocks sourceClk  
-include_generated_clocks]
```

Examples

The following examples illustrate how to use this constraint.

Example 1

This `set_clock_groups` constraint specifies that `clk4` is asynchronous to all other clocks in the design.

```
set_clock_groups -asynchronous -group {clk4}
```

Example 2

This `set_clock_groups` constraint specifies that clock `clk1`, `clk2`, and `clk3` are asynchronous to all other clocks in the design. If a new clock called `clkx` is added to the design, `clk1`, `clk2`, and `clk3` are asynchronous to it too.

```
set_clock_groups -asynchronous -group {clk1 clk2 clk3}
```

Example 3

The following `set_clock_groups` constraint has multiple `-group` arguments, and specifies that `clk1` and `clk2` are asynchronous to `clk3` and `clk4`.

```
set_clock_groups -asynchronous -group {clk1 clk2}
                  -group {clk3 clk4}
```

Example 4

The following `set_clock_groups` constraint specifies that `clk1` and `clk2` which were synchronous when defined with the `create_clock` command, are now asynchronous.

```
create_clock [get_ports {c1}] -name clk1 -period 10
create_clock [get_ports {c2}] -name clk2 -period 16
create_clock [get_ports {c3}] -name clk3 -period 5
set_clock_groups -asynchronous -group [get_clocks {clk1}]
                  -group [get_clocks {clk2}]
```

The following constructs are equivalent:

```
set_clock_groups -asynchronous -group [get_clocks {clk1}]
set_clock_groups -asynchronous -group {clk1}
```

Example 5

The following constraint specifies that `test|clkout0_derived_clock_CLKIN1` and `test|clkout1_derived_clock_CLKIN1` are asynchronous to all other clocks in the design:

```
set_clock_groups -asynchronous -group [get_clocks {*clkout*}]
```

Example 6

This example defines the clock on the `u1.clkout0` net is asynchronous to all other clocks in the design:

```
set_clock_groups -asynchronous -group [get_clocks -of_objects {n:u1.clkout0}]
```

Examples of Asynchronous Clocks

Example 1: Multiple -group Arguments for Asynchronous Clock Definition

This method uses multiple -group arguments in one constraint:

```
set_clock_groups -asynchronous -group {clk1 clk2} -group {clk3  
bclk4} -group {clk5 cclk6}
```

With this constraint, members of the same group are synchronous, but relationships between clocks from different groups defined in this constraint are asynchronous. This has the following implications:

- `clk1` and `clk2` are synchronous to each other, but asynchronous to clocks in all other groups defined in this constraint
- `clk3` and `clk4` are synchronous to each other, but asynchronous to clocks in all other groups defined in this constraint
- `clk5` and `clk6` are synchronous to each other, but asynchronous to clocks in all other groups defined in this constraint

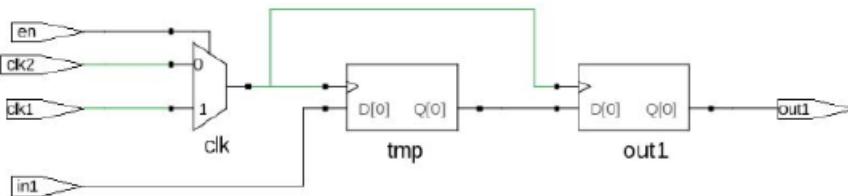
Example 2: Single -group Argument for Asynchronous Clock Definition

Asynchronous clocks defined with a single -group argument in a constraint are asynchronous to all other clocks in the design. You can specify multiple such constraints. In this example, all six clocks are asynchronous, because each individual constraint makes that clock asynchronous to all others.

```
set_clock_groups -asynchronous -group {clk1}  
set_clock_groups -asynchronous -group {clk2}  
set_clock_groups -asynchronous -group {clk3}  
set_clock_groups -asynchronous -group {clk4}  
set_clock_groups -asynchronous -group {clk5}  
set_clock_groups -asynchronous -group {clk6}
```

Examples of Defining Clocks for Clock Muxes

The definition of clocks that are to be muxed together varies slightly, depending on whether the clocks have the same frequency or not. The following procedures use this example as an illustration:



Defining Muxed Clocks with Different Frequencies

If the clocks are asynchronous, separate clock paths must be defined, as described below.

1. Define the clocks with `create_clock` constraints.

For the example, two clocks are defined:

```

create_clock -name {clk1} [get_nets {clk1}] -period 10.0
    -waveform {0 5.0}
create_clock -name {clk2} [get_nets {clk2}] -period 10.0 -waveform {0 5.0}

```

2. Use multiple `set_clock_groups` constraints to mark them as asynchronous to each other:

```

set_clock_groups -derive -asynchronous -name {default_clkgroup_0} -group
    [get_clocks {clk1}]
set_clock_groups -derive -asynchronous -name {default_clkgroup_1} -group
    [get_clocks {clk2}]

```

3. Check the timing report.

For the example, the tool reports two separate clock paths, one for each clock.

Defining Muxed Clocks with the Same Frequency

If the clocks have the same phase and frequency, follow this procedure to define the clocks.

1. Define the clock at the net connected to the output pin of the mux.

For example:

```

create_clock -name {clk} [get_nets {clk}] -period 10.0
    -waveform {0 5.0}

```

2. Define the mux output clock as asynchronous to all other clocks, using a `set_clock_groups` constraint:

```
set_clock_groups -derive -asynchronous -name {default_clkgroup_2} -group  
[get_clocks {clk}]
```

3. Check the timing report.

In this case, there should be a single clock path, instead of separate paths.

set_clock_latency

Specifies clock network latency.

Syntax

The supported syntax for the set_clock_latency constraint is:

```
set_clock_latency
  -source
  [-clock {clockList}]
  delayValue
  {objectList}
  [-disable]
```

Arguments

-source	Indicates that the specified delay is applied to the clock source latency.
-clock <i>clockList</i>	Indicates that the specified delay is applied with respect to the specified clocks. By default, the specified delay is applied to all specified objects.
<i>delayValue</i>	Specifies the clock latency value.
<i>objectList</i>	Specifies the input ports for which clock latency is to be set

Description

In the FPGA synthesis tools, the set_clock_latency constraint accepts both clock objects and clock aliases. Applying a set_clock_latency constraint on a port can be used to model the off-chip clock delays in a multi-chip environment. Clock latency is forward annotated in the top-level constraint file as part of the time budgeting that takes place in the Certify/HAPS flow. The annotated values represent the arrival times for clocks on specific ports of any particular FPGA in a HAPS design.

In the above syntax, *objectList* references either input ports with defined clocks or clock aliases defined on the input ports. When more than one clock is defined for an input port, the -clock option can be used to apply different latency values to each alias.

Restrictions

The following limitations are present in the FPGA synthesis environment:

- Clock latency can only be applied to clocks defined on input ports.
- The `set_clock_latency` constraint is only used for source latency.
- The constraint only applies to port clock objects.
- Latency on clocks defined with `create_generated_clock` is not supported.

set_clock_route_delay

Translates the -route option for the legacy define_clock constraint.

Syntax

The supported syntax for the set_clock_route_delay constraint is:

```
set_clock_route_delay {clockAliasList} {delayValue}
```

Arguments

clockAliasList Lists the clock aliases to include the route delay.

delayValue Specifies the route delay value.

Description

The sdc2fdc translator performs a translation of the -route option for the legacy define_clock constraint and places a set_clock_route_delay constraint in the *_translated.fdc file using the following format:

```
set_clock_route_delay [get_clocks {clk_alias_1 clk_alias_2 ...}]  
{delay_in_ns}
```

set_clock_uncertainty

Specifies the uncertainty (skew) of the specified clock networks.

Syntax

The supported syntax for the set_clock_uncertainty constraint is:

```
set_clock_uncertainty
{objectList}
-from fromClock | -rise_from riseFromClock | -fall_from fallFromClock
-to toClock | -rise_to riseToClock | -fall_to fallToClock
value
```

Arguments

<i>objectList</i>	Specifies the clocks for simple uncertainty. The uncertainty is applied to the capturing latches clocked by one of the specified clocks. You must specify either this argument or a clock pair with the -from/-rise_from/-fall_from and -to/-rise_to/-fall_to options; you cannot specify both an object list and a clock pair.
- from <i>fromClock</i>	Specifies the source clocks for interclock uncertainty. You can use only one of the -from, -rise_from, and -fall_from options and you must specify a destination clock with one of the -to, -rise_to, and -fall_to options.
- rise_from <i>riseFromClock</i>	Specifies that the uncertainty applies only to the rising edge of the source clock. You can use only one of the -from, -rise_from, and -fall_from options and you must specify a destination clock with one of the -to, -rise_to, and -fall_to options.
- fall_from <i>fallFromClock</i>	Specifies that the uncertainty applies only to the falling edge of the source clock. You can use only one of the -from, -rise_from, and -fall_from options and you must specify a destination clock with one of the -to, -rise_to, and -fall_to options.
- to <i>toClock</i>	Specifies the destination clocks for interclock uncertainty. You can use only one of the -to, -rise_to, and -fall_to options and you must specify a source clock with one of the -from, -rise_from, and -fall_from options.
- rise_to <i>riseToClock</i>	Specifies that the uncertainty applies only to the rising edge of the destination clock. You can use only one of the -to, -rise_to, and -fall_to options and you must specify a source clock with one of the -from, -rise_from, and -fall_from options.

-fall_to *fallToClock* Specifies that the uncertainty applies only to the falling edge of the destination clock. You can use only one of the -to, -rise_to, and -fall_to options and you must specify a source clock with one of the -from, -rise_from, and -fall_from options.

value Specifies a floating-point number that indicates the uncertainty value. Only positive uncertainty numbers are acceptable.

Examples

Refer to the following examples.

Example 1

All paths to registers clocked by `clk` are specified with setup uncertainty of 0.4 in the following example:

```
set_clock_uncertainty 0.4 -setup [get_clocks clk]
```

Example 2

For this example, interclock uncertainties are specified between clock `clk` and `clk2`:

```
set_clock_uncertainty -from [get_clocks clk] -to
[get_clocks clk2] 0.2

set_clock_uncertainty -from [get_clocks clk2] -to
[get_clocks clk] 0.1
```

Example 3

For this example, interclock uncertainties are specified between clock `clk` and `clk2` with specific edges:

```
set_clock_uncertainty -rise_from [get_clocks clk2] -to
[get_clocks clk] 0.5

set_clock_uncertainty -rise_from [get_clocks clk2] -rise_to
[get_clocks clk] 0.1

set_clock_uncertainty -from [get_clocks clk2] -fall_to
[get_clocks clk] 0.1
```

set_datapathonly_delay

Specifies a point-to-point path delay that excludes any applicable clock delays at the start and end points.

The `set_datapathonly_delay` constraint specifies a point-to-point path delay that excludes any applicable clock delays at the start and end points. The constraint includes clock to out (T_{CO}) and setup time (T_{SU}) as part of the data path delay.

Use this constraint in Xilinx designs, as the tool only forward annotates the constraint for Xilinx designs. The `set_datapathonly_delay` constraint appears in the XDC file with the `set_max_delay -datapath_only` option. The original .fdc command is forward annotated as a comment. For this example, the `-from` and `-to` options are used to specify the pins for the start and end points of the path delay.

```
#set_datapathonly_delay -from {p:din[*]} -to {i:int[*]} {4}
set_max_delay -datapathonly_delay -from [get_ports {d_in[*]}]
              -to [get_cells {int[*]}] {4}
```

This constraint appears in the UCF file as a Xilinx TIMESPEC constraint with the DATAPATHONLY option. The original .sdc command is also forward-annotated as a comment. For example:

```
# set_datapathonly_delay -from {p:din[*]} -to {i:int[*]} {4}
...
NET "din[7]" TNM = "from_1005_0";
INST "int[*]" TNM = "to_1005_0"; # int[7:0]
TIMESPEC "TS_1005_0" = FROM "from_1005_0" TO "to_1005_0"
          DATAPATHONLY 4.000;
```

When you specify this constraint, the tool ignores any clock skew or phase information. It constrains, analyzes, and forward-annotates just the data path between the specified points.

Syntax

set_datapathonly_delay option {value}

In the above syntax, *option* is one or more of the following:

```
-comment textString
-disable
-from fromClock | -rise_from riseFromClock | -fall_from fallFromClock
-to toClock | -rise_to riseToClock | -fall_to fallToClock
-through {objectList} [-through {objectList} ...]
```

value

Specifies a floating-point number that indicates the delay value

-from {port|inst|clock}

Specifies the start point for the path delay.

-rise_from riseFromClock

Specifies that the start delay applies only to the rising edge of the source clock. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-fall_from fallFromClock

Specifies that the start delay applies only to the falling edge of the source clock. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-to {port|inst|clock}

Specifies the end point for the path delay.

-through {objectList} [-through {objectList} ...]

Specifies the intermediate points for the timing exception. The -through *objectList* includes:

- Combinational nets
- Hierarchical ports
- Pins on instantiated cells
- Cell instances

By default, the through points are treated as an OR list. The constraint is applied if the path crosses any points in *objectList*. If more than one object is included, the objects must be enclosed either in quotation marks ("") or in braces ({}). If you specify the -through option multiple times, set_path applies to the paths that pass through a member of each *objectList*. If you use the -through option in combination with the -from or -to options, set_false_path applies only if the -from or -to and the -through conditions are satisfied.

-rise_to riseToClock

Specifies that the end delay applies only to the rising edge of the source clock. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.

-fall_to fallToClock

Specifies that the end delay applies only to the falling edge of the source clock. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.

-disable

Disables the constraint.

-comment *textString*

Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

In the above options, the variable *clock* is a clock alias.

set_false_path

Removes timing constraints from particular paths.

Syntax

The supported syntax for the set_false_path constraint is:

```
set_false_path
  [-setup]
  [-from {objectList} |-rise_from riseFromClock | -fall_from fallFromClock]
  [-through {objectList} [-through {objectList} ...]]
  [-to {objectList} |-rise_to riseToClock | -fall_to fallToClock]
  [-disable]
  [-comment commentString]
```

Arguments

-setup	Specifies that setup checking (maximum delay) is reset to single-cycle behavior.
-from	Specifies the names of objects to use to find path start points. The -from <i>objectList</i> includes: <ul style="list-style-type: none"> • Clocks • Registers • Top-level input or bi-directional ports • Black box outputs • Sequential cell clock pins • Sequential cell data output pins • When the specified object is a clock, all flip-flops, latches, and primary inputs related to that clock are used as path start points.
-rise_from <i>riseFromClock</i>	Specifies to use the rising edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.
-fall_from <i>fallFromClock</i>	Specifies to use the falling edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-through	<p>Specifies the intermediate points for the timing exception. The -through <i>objectList</i> includes:</p> <ul style="list-style-type: none"> • Combinational nets • Hierarchical ports • Pins on instantiated cells • Cell instances <p>By default, the through points are treated as an OR list. The constraint is applied if the path crosses any points in <i>objectList</i>. If more than one object is included, the objects must be enclosed either in quotation marks ("") or in braces ({}). If you specify the -through option multiple times, set_path applies to the paths that pass through a member of each <i>objectList</i>. If you use the -through option in combination with the -from or -to options, set_false_path applies only if the -from or -to and the -through conditions are satisfied.</p>
-to	<p>Specifies the names of objects to use to find path end points. The -to <i>objectList</i> includes:</p> <ul style="list-style-type: none"> • Clocks • Registers • Top-level output or bi-directional ports • Black box inputs • Sequential cell data input pins <p>If a specified object is a clock, all flip-flops, latches, and primary outputs related to that clock are used as path end points.</p>
-rise_to <i>riseToClock</i>	<p>Specifies to use the rising edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.</p>
-fall_to <i>fallToClock</i>	<p>Specifies to use the falling edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.</p>
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Examples

Refer to the following examples.

Example 1

All the paths from the sequential cell output pins and clock pins in module `gen_sub\[*\].u_sub` with names matching `out*` (i.e. registers `out1` and `out2` in modules `gen_sub\[0\].u_sub`, `gen_sub\[1\].u_sub`, and `gen_sub\[2\].u_sub`) are set as false paths.

```
set_false_path -from [get_pins {gen_sub\[*\]\.u_sub.out*.*}]
```

Note: Only sequential clock pins are valid pins that can be used as start points for a timing path. Note that hierarchical module pins are not valid as starting points for a timing path.

Example 2

All the paths from the sequential cells in module `gen_sub\[*\].u_sub` with names matching `out*` (i.e. registers `out1` and `out2` in modules `gen_sub\[0\].u_sub`, `gen_sub\[1\].u_sub`, and `gen_sub\[2\].u_sub`) are set as false paths.

```
set_false_path -from [get_cells {gen_sub\[*\]\.u_sub.out*}]
```

Example 3

All paths from top-level input ports with names `in*` are set as false paths.

```
set_false_path -from [get_ports {in*}]
```

Note: Only top-level ports are valid port based start points for timing paths. Do not use the `get_ports` command to reference hierarchical module pins.

Example 4

All paths with end points clocked by clock `clka` are set as false paths.

```
set_false_path -to [get_clocks {clka}]
```

set_input_delay

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

Syntax

The supported syntax for the set_input_delay constraint is:

```
set_input_delay
[-clock clockName [-clock_fall]]
[-rise|-fall]
[-min|-max]
[-add_delay]
delayValue
{portPinList}
[-disable]
[-comment commentString]
```

Argument

-clock <i>clockName</i>	Specifies the clock to which the specified delay is related. If -clock_fall is used, -clock <i>clockName</i> must be specified. If -clock is not specified, the delay is relative to time zero for combinational designs. For sequential designs, the delay is considered relative to a new clock with the period determined by considering the sequential cells in the transitive fanout of each port.
-clock_fall	Specifies that the delay is relative to the falling edge of the clock. The default is the rising edge.
-rise	Specifies that <i>delayValue</i> refers to a rising transition on the specified ports of the current design. If neither -rise nor -fall is specified, rising and falling delays are assumed to be equal. Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the -rise option is preserved and forward annotated to the place-and-route tool.

-fall	Specifies that <i>delayValue</i> refers to a falling transition on the specified ports of the current design. If neither -rise nor -fall is specified, rising and falling delays are assumed equal. Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the -fall option is preserved and forward annotated to the place-and-route tool.
-min	Specifies that <i>delayValue</i> refers to the shortest path. If neither -max nor -min is specified, maximum and minimum input delays are assumed equal. Note: The synthesis tool does not optimize for hold time violations and only reports -min delay values in the <code>synlog/topLevel_fpga_mapper.srr_Min</code> timing report section of the log file. The -min delay values are forward annotated to the place-and-route tool.
-max	Specifies that <i>delayValue</i> refers to the longest path. If neither -max nor -min is specified, maximum and minimum input delays are assumed equal. Note: The -max delay values are reported in the top-level log file and are forward annotated to the place-and-route tool.
-add_delay	Specifies if delay information is to be added to the existing input delay or if it is to be overwritten. The -add_delay option enables you to capture information about multiple paths leading to an input port that are relative to different clocks or clock edges.
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.
<i>delayValue</i>	Specifies the path delay. The <i>delayValue</i> must be in units consistent with the technology library used during optimization. The <i>delayValue</i> represents the amount of time before the signal is available after a clock edge. This represents a combinational path delay from the clock pin of a register.
<i>portPinList</i>	Specifies a list of input port names in the current design to which <i>delayValue</i> is assigned. If more than one object is specified, the objects are enclosed in quotes ("") or in braces ({}).

Examples

Refer to the following examples.

Example 1

This example sets an input delay of 1.0 relative to the rising edge of clk.

```
set_input_delay 1.00 -clock clk [get_ports {din1 din2}]
```

Example 2

The following example sets an input delay of 1.0 relative to the rising edge of clk for all inputs in the design.

```
set_input_delay 1.00 -clock clk [all_inputs]
```

Example 3

In this scenario, there are two paths to the input port din1. The input delay for the first path is relative to the rising edge of clk. For the second path, the input delay is relative to the falling edge of clk. The `-add_delay` option indicates that the new input delay information does not cause old information to be removed.

```
set_input_delay 1.00 -clock clk [get_ports {din1}]
set_input_delay 2.00 -clock clk [get_ports {din1}] -add_delay
    -clock_fall
```

set_max_delay

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

Syntax

The supported syntax for the set_max_delay constraint is:

```
set_max_delay
[-from {objectList} |-rise_from riseFromClock | -fall_from fallFromClock]
[-through {objectList} [-through {objectList} ...]]
[-to {objectList} |-rise_to riseToClock | -fall_to fallToClock]
delayValue
[-disable]
[-comment commentString]
```

Arguments

-from Specifies the names of objects to use to find path start points. The -from *objectList* includes:

- Clocks
- Registers
- Top-level input or bi-directional ports
- Black box outputs
- Sequential cell clock pins

When the specified object is a clock, all flip-flops, latches, and primary inputs related to that clock are used as path start points. All paths from these start points to the end points in the -from *objectList* are constrained to *delayValue*. If a -to *objectList* is not specified, all paths from the -from *objectList* are affected. If you include more than one object, you must enclose the objects in quotation marks ("") or braces ({}).

-rise_from
riseFromClock Specifies to use the rising edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-fall_from
fallFromClock Specifies to use the falling edge of the source clock to find path start points. Use only one of the -from, -rise_from, and -fall_from options and specify a destination clock with one of the -to, -rise_to, and -fall_to options.

-through	<p>Specifies the intermediate points for the timing exception. The -through <i>objectList</i> includes:</p> <ul style="list-style-type: none"> • Combinational nets • Hierarchical ports • Pins on instantiated cells • Cell instances <p>By default, the through points are treated as an OR list. The constraint is applied if the path crosses any points in <i>objectList</i>. The max delay value applies only to paths that pass through one of the points in the -through <i>objectList</i>. If more than one object is included, the objects must be enclosed either in quotation marks ("") or in braces ({}). If you specify the -through option multiple times, set_max_delay applies to the paths that pass through a member of each <i>objectList</i>. If you use the -through option in combination with the -from or -to options, set_max_delay applies only if the -from or -to and the -through conditions are satisfied.</p>
-to	<p>Specifies the names of objects to use to find path end points. The -to <i>objectList</i> includes:</p> <ul style="list-style-type: none"> • Clocks • Registers • Top-level output or bi-directional ports • Black box inputs • Sequential cell data input pins <p>If a specified object is a clock, all flip-flops, latches, and primary outputs related to that clock are used as path end points. All paths to the end points in the -to <i>objectList</i> are constrained to <i>delayValue</i>. If a -from <i>objectList</i> is not specified, all paths to the -to <i>objectList</i> are affected. If you include more than one object, you must enclose the objects in quotation marks ("") or braces ({}).</p>
-rise_to <i>riseToClock</i>	<p>Specifies to use the rising edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.</p>
-fall_to <i>fallToClock</i>	<p>Specifies to use the falling edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.</p>
-disable	Disables the constraint.

-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.
delayValue	Specifies the value of the desired maximum delay for paths between start and end points. You must express <i>delayValue</i> in the same units as the technology library used during optimization. If a path start point is on a sequential device, clock skew is included in the computed delay. If a path start point has an input delay specified, that delay value is added to the path delay. If a path end point is on a sequential device, clock skew and library setup time are included in the computed delay. If the end point has an output delay specified, that delay is added into the path delay.

Examples

This example shows how to specify that all paths from cell temp1 to cell temp2 must be less than 4.0 units.

```
set_max_delay -from [get_cells {temp1}] -to [get_cells {temp2}] 4
```

set_min_delay

Specifies a minimum delay target for paths in the current design.

Syntax

```
set_min_delay
  [-from {objectList} |-rise_from riseFromClock | -fall_from fallFromClock]
  [-through {objectList} [-through {objectList} ...]]
  [-to {objectList} |-rise_to riseToClock | -fall_to fallToClock]
  delayValue
  [-disable]
  [-comment commentString]
```

-from {objectList}

Specifies the names of objects to use to find path start points. The **-from objectList** includes:

- Clocks
- Registers
- Top-level input or bidirectional ports
- Black box outputs
- Sequential cell clock pins

When the specified object is a clock, all flip-flops, latches, and primary inputs related to that clock are used as path start points. All paths from these start points to the end points in the **-from objectList** are constrained to *delayValue*. If a **-to objectList** is not specified, all paths from the **-from objectList** are affected. If you include more than one object, enclose the objects in quotation marks ("") or braces ({}).

-rise_from riseFromClock

Specifies to use the rising edge of the source clock to find path start points. Use only one of the **-from**, **-rise_from**, and **-fall_from** options and specify a destination clock with one of the **-to**, **-rise_to**, and **-fall_to** options.

-fall_from fallFromClock

Specifies to use the falling edge of the source clock to find path start points. Use only one of the **-from**, **-rise_from**, and **-fall_from** options and specify a destination clock with one of the **-to**, **-rise_to**, and **-fall_to** options.

-through

Specifies the intermediate points for the timing exception. The -through *objectList* includes:

- Combinational nets
- Hierarchical ports
- Pins on instantiated cells
- Cell instances

By default, the through points are treated as an OR list. The constraint is applied if the path crosses any points in *objectList*. The minimum delay value applies only to paths that pass through one of the points in the -through *objectList*. If more than one object is included, the objects must be enclosed either in quotation marks ("") or in braces ({}). If you specify the -through option multiple times, set_min_delay applies to the paths that pass through a member of each *objectList*. When using the -through option in combination with the -from or -to options, set_min_delay applies only if the -from or -to and the -through conditions are satisfied.

-to

Specifies the names of objects to use to find path end points. The -to *objectList* includes:

- Clocks
- Registers
- Top-level input or bidirectional ports
- Black box outputs

If a specified object is a clock, all flip-flops, latches, and primary outputs related to that clock are used as path end points. All paths to the end points in the -to *objectList* are constrained to *delayValue*. If a -from *objectList* is not specified, all paths to the -to *objectList* are affected. When including more than one object, enclose the objects in quotation marks ("") or braces ({}).

-rise_to *riseToClock*

Specifies to use the rising edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.

-fall_to fallToClock

Specifies to use the falling edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.

delayValue

Specifies the value of the desired minimum delay for paths between start and end points. If a path start point is on a sequential device, clock skew is included in the computed delay. If a path start point has an input delay specified, that delay value is added to the path delay. If a path end point is on a sequential device, clock skew and library setup time are included in the computed delay. If the end point has an output delay specified, that delay is added into the path delay.

-disable

Disables the constraint.

-comment textString

Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Examples

This example shows how to specify that all paths from cell temp1 to cell temp2 must be greater than 4.0 units.

```
set_min_delay -from [get_cells {temp1}] -to [get_cells {temp2}] 4
```

set_multicycle_path

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

Syntax

The supported syntax for the set_multicycle_path constraint is:

```
set_multicycle_path
  [-setup | -hold]
  [-start | -end]
  [-from {objectList} |-rise_from riseFromClock | -fall_from fallFromClock]
  [-through {objectList} [-through {objectList} ...] ]
  [-to {objectList} |-rise_to riseToClock | -fall_to fallToClock]
  pathMultiplier
  [-disable]
  [-comment commentString]
```

Arguments

-setup | -hold The option **-setup** specifies the pathMultiplier to be used for the setup (maximum delay) calculations.

The option **-hold** enables you to over-ride the default hold multiplier— $\{\text{pathMultiplier} - 1\}$ —that is forward annotated to the vendor constraint file. If you use this option, you must specify the hold value for each of the defined multicycle constraints.

If you do not provide **-setup** or **-hold**, the pathMultiplier is used for setup.

-start | -end Specifies if the multi-cycle information is relative to the period of either the start clock or the end clock. These options are only needed for multi-frequency designs; otherwise start and end are equivalent. The start clock is the clock source related to the register or primary input at the path start point. The end clock is the clock source related to the register or primary output at the path endpoint. The default is to move the setup check relative to the end clock, and the hold check relative to the start clock. A setup multiplier of 2 with **-end** moves the relation forward one cycle of the end clock. A setup multiplier of 2 with **-start** moves the relation back one cycle of the start clock. A hold multiplier of 1 with **-start** moves the relation forward one cycle of the start clock. A hold multiplier of 1 with **-end** moves the relation back one cycle of the end clock. If you do not provide **-start** or **-end**, **-end** is assumed.

-from	<p>Specifies the names of objects to use to find path start points. The <i>-from objectList</i> includes:</p> <ul style="list-style-type: none"> • Clocks • Registers • Top-level input or bi-directional ports • Black box outputs • Sequential cell clock pins • Sequential cell data output pins <p>When the specified object is a clock, all flip-flops, latches, and primary inputs related to that clock are used as path start points. If a <i>-to objectList</i> is not specified, all paths from the <i>-from objectList</i> are affected. If you include more than one object, you must enclose the objects in quotation marks ("") or braces ({}).</p>
-rise_from <i>riseFromClock</i>	Specifies to use the rising edge of the source clock to find path start points. Use only one of the <i>-from</i> , <i>-rise_from</i> , and <i>-fall_from</i> options and specify a destination clock with one of the <i>-to</i> , <i>-rise_to</i> , and <i>-fall_to</i> options.
-fall_from <i>fallFromClock</i>	Specifies to use the falling edge of the source clock to find path start points. Use only one of the <i>-from</i> , <i>-rise_from</i> , and <i>-fall_from</i> options and specify a destination clock with one of the <i>-to</i> , <i>-rise_to</i> , and <i>-fall_to</i> options.
-through	<p>Specifies the intermediate points for the timing exception. The <i>-through objectList</i> includes:</p> <ul style="list-style-type: none"> • Combinational nets • Hierarchical ports • Pins on instantiated cells • Cell instances <p>The multi-cycle values apply only to paths that pass through one of the points in the <i>-through objectList</i>. If more than one object is included, the objects must be enclosed either in double quotation marks ("") or in braces ({}). If you specify the <i>-through</i> option multiple times, <i>set_multicycle_delay</i> applies to the paths that pass through a member of each <i>objectList</i>. If the <i>-through</i> option is used in combination with the <i>-from</i> or <i>-to</i> options, the multi-cycle values apply only if the <i>-from</i> or <i>-to</i> conditions and the <i>-through</i> conditions are satisfied.</p>

-to	Specifies the names of objects to use to find path end points. The -to <i>objectList</i> includes:
	<ul style="list-style-type: none"> • Clocks • Registers • Top-level output or bi-directional ports • Black box inputs • Sequential cell data input pins
	If a specified object is a clock, all flip-flops, latches, and primary outputs related to that clock are used as path end points. If a -from <i>objectList</i> is not specified, all paths to the -to <i>objectList</i> are affected. If you include more than one object, you must enclose the objects in quotation marks ("") or braces ({}).
-rise_to <i>riseToClock</i>	Specifies to use the rising edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.
-fall_to <i>fallToClock</i>	Specifies to use the falling edge of the source clock to find path end points. Use only one of the -to, -rise_to, and -fall_to options and specify a source clock with one of the -from, -rise_from, and -fall_from options.
pathMultiplier	Specifies the number of cycles that the data path must have, relative to the start point or end point clock, before data is required at the end point.
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

Examples

Refer to the following examples.

Example 1

All the paths from the sequential cell output pins and clock pins in module gen_sub\[*].u_sub with names matching out* (i.e. registers out1 and out2 in modules gen_sub\[0].u_sub, gen_sub\[1].u_sub, and gen_sub\[2].u_sub) provide 2 timing cycles before the data is required at the end point.

```
set_multicycle_path -from [get_pins{gen_sub\[*\]\.u_sub.out*.*}] 2
```

Note: Only sequential clock pins are pins that can be used as valid start points for a timing path. Note that hierarchical module pins cannot be used as starting points for a timing path.

Example 2

All the paths from the sequential cells in module gen_sub\[*\].u_sub with names matching out* (i.e. registers out1 and out2 in modules gen_sub\[0\].u_sub, gen_sub\[1\].u_sub, and gen_sub\[2\].u_sub) support the timing cycle set to 2.

```
set_multicycle_path -from [get_cells {gen_sub\[*\]\.u_sub.out*}] 2
```

Example 3

All paths from top-level input ports with names in* provide 2 timing cycles before the data is required at the end point.

```
set_multicycle_path -from [get_ports {in*}] 2
```

Note: Only top-level ports are valid port based start points for timing paths. Do not use the get_ports command to reference hierarchical module pins.

Example 4

All paths with end points clocked by clock clka provide 2 timing cycles before the data is required at the end point.

```
set_multicycle_path -to [get_clocks {clka}] 2
```

set_output_delay

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

Syntax

The supported syntax for the set_output_delay constraint is:

```
set_output_delay
  [-clock clockName [-clock_fall]]
  [-rise][-fall]
  [-min|-max]
  [-add_delay]
  delayValue
  {portPinList}
  [-disable]
  [-comment commentString]
```

Arguments

-clock <i>clockName</i>	Specifies the clock to which the specified delay is related. If -clock_fall is used, -clock <i>clockName</i> must be specified. If -clock is not specified, the delay is relative to time zero for combinational designs. For sequential designs, the delay is considered relative to a new clock with the period determined by considering the sequential cells in the transitive fanout of each port.
-clock_fall	Specifies that the delay is relative to the falling edge of the clock. If -clock is specified, the default is the rising edge.
-rise	Specifies that <i>delayValue</i> refers to a rising transition on the specified ports of the current design. If neither -rise nor -fall is specified, rising and falling delays are assumed to be equal. Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the -rise option is preserved and forward annotated to the place-and-route tool.

-fall	Specifies that <i>delayValue</i> refers to a falling transition on the specified ports of the current design. If neither -rise nor -fall is specified, rising and falling delays are assumed equal. Currently, the synthesis tool does not differentiate between the rising and falling edges for the data transition arcs on the specified ports. The worst case path delay is used instead. However, the -fall option is preserved and forward annotated to the place-and-route tool.
-min	Specifies that <i>delayValue</i> refers to the shortest path. If neither -max nor -min is specified, maximum and minimum output delays are assumed equal. Note: The synthesis tool does not optimize for hold time violations and only reports -min delay values in the <i>synlog/topLevel_fpga_mapper.srr_Min</i> timing report section of the log file. The -min delay values are forward annotated to the place-and-route tool.
-max	Specifies that <i>delayValue</i> refers to the longest path. If neither -max nor -min is specified, maximum and minimum output delays are assumed equal. Note: The -max delay values are reported in the top-level log file and are forward annotated to the place-and-route tool.
-add_delay	Specifies whether to add delay information to the existing output delay or to overwrite. The -add_delay option enables you to capture information about multiple paths leading to an output port that are relative to different clocks or clock edges.
-disable	Disables the constraint.
-comment <i>textString</i>	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.
<i>delayValue</i>	Specifies the path delay. The <i>delayValue</i> must be in units consistent with the technology library used during optimization. The <i>delayValue</i> represents the amount of time that the signal is required before a clock edge. For maximum output delay, this usually represents a combinational path delay to a register plus the library setup time of that register. For minimum output delay, this value is usually the shortest path delay to a register minus the library hold time

<i>portPinList</i>	A list of output port names in the current design to which <i>delayValue</i> is assigned. If more than one object is specified, the objects are enclosed in double quotation marks ("") or in braces ({}).
--------------------	--

Examples

Refer to the following examples.

Example 1

This example sets an output delay of 1.00 relative to the rising edge of clk for all the output ports in the design.

```
set_output_delay 1.00 -clock clk [all_outputs]
```

Example 2

The following example sets an output delay of 2.00 relative to the falling edge of clk for the output port dout1. The -add_delay option indicates that this delay value is to be added to any existing output delays defined on this port.

```
set_output_delay 2.0 -clock clk [get_ports {dout1}] -add_delay  
-clock_fall
```

set_reg_input_delay

Speeds up paths feeding a register by a given number of nanoseconds.

Syntax

```
set_reg_input_delay {registerName} [-route ns] [-disable] [-comment textString]
```

Arguments

registerName A single bit, an entire bus, or a slice of a bus.

-route Advanced user option that you use to tighten constraints during resynthesis, when the place-and-route timing report shows the timing goal is not met because of long paths to the register.

-comment Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.

-disable Disables the constraint.

Description

The `set_reg_input_delay` timing constraint speeds up paths feeding a register by a given number of nanoseconds. The Synopsys FPGA synthesis tool attempts to meet the global clock frequency goals for a design as well as the individual clock frequency goals (set with `create_clock`). Use this constraint to speed up the paths feeding a register. For information about the equivalent SCOPE spreadsheet interface, see [Registers, on page 309](#).

Use this constraint instead of the legacy constraint, `define_reg_input_delay`.

set_reg_output_delay

Speeds up paths coming from a register by a given number of nanoseconds.

Syntax

```
set_reg_output_delay {registerName} [-route ns] [-disable] [-comment textString]
```

Arguments

<i>registerName</i>	A single bit, an entire bus, or a slice of a bus.
-route	Advanced user option that you use to tighten constraints during resynthesis, when the place-and-route timing report shows the timing goal is not met because of long paths from the register.
-comment	Allows the command to accept a comment string. The tool honors the annotation and preserves it with the object so that the exact string is written out when the constraint is written out. The comment remains intact through the synthesis, place-and-route, and timing-analysis flows.
-disable	Disables the constraint.

Description

The set_reg_output_delay constraint speeds up paths coming from a register by a given number of nanoseconds. The synthesis tool attempts to meet the global clock frequency goals for a design as well as the individual clock frequency goals (set with create_clock). Use this constraint to speed up the paths coming from a register. For information about the equivalent SCOPE spreadsheet interface, see [Registers, on page 309](#).

Use this constraint instead of the legacy constraint, define_reg_output_delay.

Naming Rule Syntax Commands

The FPGA synthesis environment uses a set of naming conventions for design objects in the RTL when your project contains constraint files. The following naming rule commands are added to the constraint file to change the expected default values. These commands must appear at the beginning of

the constraint file before any other constraints. Similarly, when multiple constraint files are included in the project, the naming rule commands must be in the first constraint file read.

set_hierarchy_separator Command

The `set_hierarchy_separator` command redefines the hierarchy separator character (the default separator character is the period in the FPGA synthesis environment). For example, the following command changes the separator character to a forward slash:

```
set_hierarchy_separator {/}
```

Embedded Tcl commands, such as `get_pins` must be enclosed in brackets `[]` for the software to execute the command. Also, the curly brackets `{ }` are required when object names include the escape `(\)` character or square brackets. For example, the following syntax is honored by the tool:

```
set_hierarchy_separator {/}
create_clock -name {clk1} [get_pins
{pdः_c/ib_phys_c/port_g\1.phy_c/c7_g\gtxe2_common_0_i/GTREFCLK[0]}]
-period {10}
```

set_rtl_ff_names Command

The `set_rtl_ff_names` command controls the stripping of register suffixes in the object strings of delay-path constraints (for example, `set_false_path`, `set_multicycle_path`). Generally, it is only necessary to change this value from its default when constraints that target ASIC designs are being imported from the Design Compiler (in the Design Compiler, inferred registers are given a `_reg` suffix during the elaboration phase; constraints targeting these registers must include this suffix). When importing constraints from the Design Compiler, include the following command to change the value of this naming rule to `{_reg}` to automatically recognize the added suffix.

```
set_rtl_ff_names {_reg}
```

For example, using the above value allows the DC exception

```
set_false_path -to [get_cells {register_bus_reg[0]}]
```

to apply to the following object without having to manually modify the constraint:

```
[get_cells {register_bus[0]}]
```

bus_naming_style Command

The `bus_naming_style` command redefines the format for identifying bits of a bus (by default, individual bits of a bus are identified by the bus name followed by the bus bit enclosed in square brackets). For example, the following command changes the bus-bit identification from the default `busName[busBit]` format to the `busName_busBit` format:

```
bus_naming_style {%-d}
```

bus_dimension_separator_style Command

The `bus_dimension_separator_style` command redefines the format for identifying multi-dimensional arrays (by default, multidimensional arrays such as row 2, bit 3 of array ABC[$n \times m$] are identified as ABC[2][3]). For example, the following command changes the bus-dimension separator from individual square bracket sets to an underscore:

```
bus_dimension_separator_style {_}
```

The resulting format for the above example is:

```
ABC[2_3]
```

read_sdc Command

Reads in a script in Synopsys FPGA constraint format. The supported syntax for the `read_sdc` constraint is:

```
read_sdc fileName
```

Design Constraints

This section describes the constraint file syntax for the following non-timing design constraints:

- [define_compile_point](#), on page 392
- [define_current_design](#), on page 393
- [define_io_standard](#), on page 394

define_compile_point

The `define_compile_point` command defines a compile point in a top-level constraint file. You use one `define_compile_point` command for each compile point you define. For the equivalent SCOPE spreadsheet interface, see [Compile Points, on page 316](#). (Compile points are only available for certain technologies.)

This is the syntax:

```
define_compile_point [-disable] {moduleName}  
    -type {black_box|soft|hard|locked, partition} [-comment textString]
```

-disable Disables a previous compile point definition.

-type Specifies the type of compile point. This can be `black_box`, `soft`, `hard`, `locked`, or `partition`. See [Compile Point Types, on page 610](#) for more information.

Refer to [Guidelines for Entering and Editing Constraints, on page 218](#) for details about the syntax and prefixes for naming objects.

Here is a syntax example:

```
define_compile_point {v:work.prgm_cntr} -type {locked}
```

define_current_design

The `define_current_design` command specifies the module to which the constraints that follow it apply. It must be the first command in a block-level or compile-point constraint file. The specified module becomes the top level for objects defined in this hierarchy and the constraints applied in the respective block-level or compile-point constraint file.

This is the syntax:

```
define_current_design {regionName | libraryName.moduleName}
```

Refer to [Guidelines for Entering and Editing Constraints, on page 218](#) for details about the syntax and prefixes for naming objects.

Here is an example:

```
define_current_design {lib1.prgm_cntr}
```

Objects in all constraints that follow this command relate to `prgm_cntr`.

define_io_standard

Specifies a standard I/O pad type to use for various vendor-specific families. See [I/O Standards, on page 314](#) for details of the SCOPE equivalent.

```
define_io_standard [-disable] {p:portName} -delay_type input|output|bidir
    syn_pad_type {I/O_standard} [parameter {value}...]
```

In the above syntax:

portName

is the name of the input, output, or bidirectional port.

-delay_type

identifies the port direction which must be input, output, or bidir.

syn_pad_type

is the I/O pad type (I/O standard) to be assigned to *portName*.

parameter

is one or more of the parameters defined in the following table. Note that these parameters are device-family dependent.

Parameter	Function
<code>syn_io_termination</code>	The termination type; typical values are pullup and pulldown.
<code>syn_io_drive</code>	The output drive strength; numerical values in mA.
<code>syn_io_dv2</code>	Switch to use a 2x impedance value (DV2).
<code>syn_io_dci</code>	Switch for digitally-controlled impedance (DCI).
<code>syn_io_slew</code>	The slew rate for single-ended output buffers; values include slow and fast or low and high.

Examples:

```
define_io_standard {p:DATA1[7:0]} -delay_type input
    syn_pad_type {LVCMOS_33} syn_io_slew {high}
    syn_io_drive {12} syn_io_termination {pulldown}

define_io_standard {p:en} -delay_type input
    syn_pad_type {LVCMOS_18} syn_io_dci {DCI}
    syn_io_dv2 {DV2}
```

CHAPTER 5

User Interface Commands

The following describe the graphical user interface (GUI) commands available from the menus:

- [File Menu](#), on page 396
- [Edit Menu](#), on page 402
- [View Menu](#), on page 415
- [Project Menu](#), on page 423
- [Implementation Options Command](#), on page 444
- [Import Menu](#), on page 485
- [Run Menu](#), on page 498
- [Analysis Menu](#), on page 519
- [HDL Analyst Menu](#), on page 542
- [Options Menu](#), on page 568
- [Tech-Support Menu](#), on page 598
- [Web Menu](#), on page 601
- [Help Menu](#), on page 602

For information about context-sensitive commands accessed from right-click popup menus, see [Chapter 6, GUI Popup Menu Commands](#).

File Menu

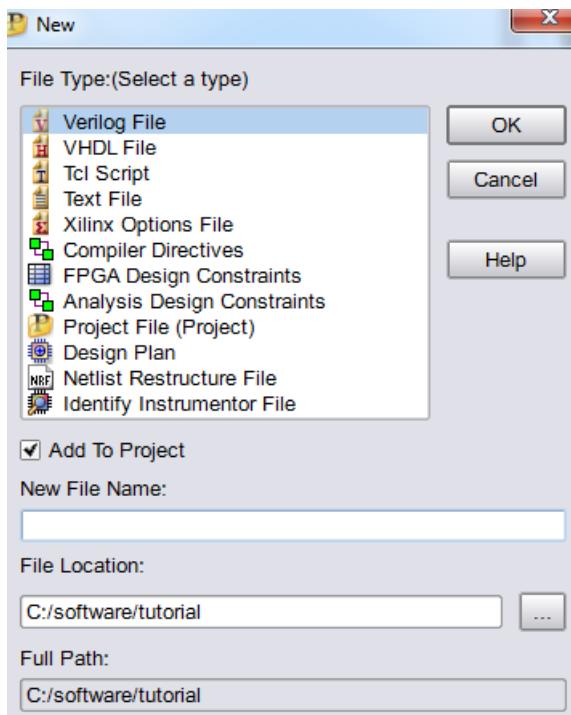
Use the File menu for opening, creating, saving, and closing projects and files. The following table describes the File menu commands.

Command	Description
New	Can create any of the following types of files: Text, Tcl Script, VHDL, Verilog, P&R Options file, Design Constraints, Analysis Design Constraints, or Project file. See New Command , on page 397 . <i>Synplify Premier</i> Also creates the Compiler Directives Constraints, Design Plan, and Netlist Restructure files.
 Open	Opens a project or file.
Close	Closes a project file.
 Save	Saves a project or a file.
Save As	Saves a project or a file to a specified name.
 Save All	Saves all projects or files.
Print	Prints a file. For more information about printing, see the operating system documentation.
Print Setup	Specify print options.
Create Image	This command is available in the following views: <ul style="list-style-type: none"> • HDL Analyst Views • FSM Viewer • Design Planner (<i>Synplify Premier</i>) • Physical Analyst (<i>Synplify Premier</i>) A camera pointer () appears. Drag a selection rectangle around the region for which you want to create an image, then release the mouse button. You can also simply click in the current view, then the Create Image dialog appears. See Create Image Command , on page 399 .
Build Project	Creates a new project based on the file open in the Text Editor (if active), or lets you choose files to add to a new project. See Build Project Command , on page 401 .
 Open Project	Opens a project. See Open Project Command , on page 401 .

Command	Description
New Project	Creates a new project. If a project is already open, it prompts you to save it before creating a new one. If you want to open multiple projects, select Allow multiple projects to be opened in the Project View dialog box. See Project View Options Command , on page 573 .
Close Project	Closes the current project.
Recent Projects	Lists recently accessed projects. Choose a project listed in the submenu to open it.
Recent files (listed as separate menu items)	Lists the last files you recently opened as separate menu items. Choose a file to open it.
Exit	Exits the session.

New Command

Select File->New to display the New dialog box, where you can select a file type to be created (for example, Verilog, VHDL, text, Tcl script, P&R options, design constraints, analysis design constraints, or project). For most file types, a text editor window opens to allow you to define the file contents. You must provide a file name. You can automatically add the new file to your project by enabling the Add To Project checkbox before clicking OK.



File Type	Opens Window	Directory Name	Extension
Verilog	Text Editor	Verilog	.v
VHDL	Text Editor	VHDL	.vhd
Text	Text Editor	Other	.txt
Tcl Script	Text Editor	Tcl Script	.tcl
FPGA Design Constraints	SCOPE	Constraint	.fdc
Analysis Design Constraints	SCOPE	Analysis Design Constraint	.adc
Project	None	None	.prj
Instrumentation Design Constraints	Instrumentor	Identify Design Constraint	.idc
Xilinx Options	Text Editor	P&R Options	.opt
Compiler Directives ¹	Modified Text Editor	Compiler Directive	.cdc

File Type	Opens Window	Directory Name	Extension
Design Plan ¹	Design Planner UI	Design Plan	.dpf/.sfp
Netlist Restructure ¹	Bit Slice UI	Netlist Restructure	.nrf
Prototyping Tool ²	Bit Slice UI	Prototyping Constraint	.ptf

1. Available in the Synplify Premier tools.

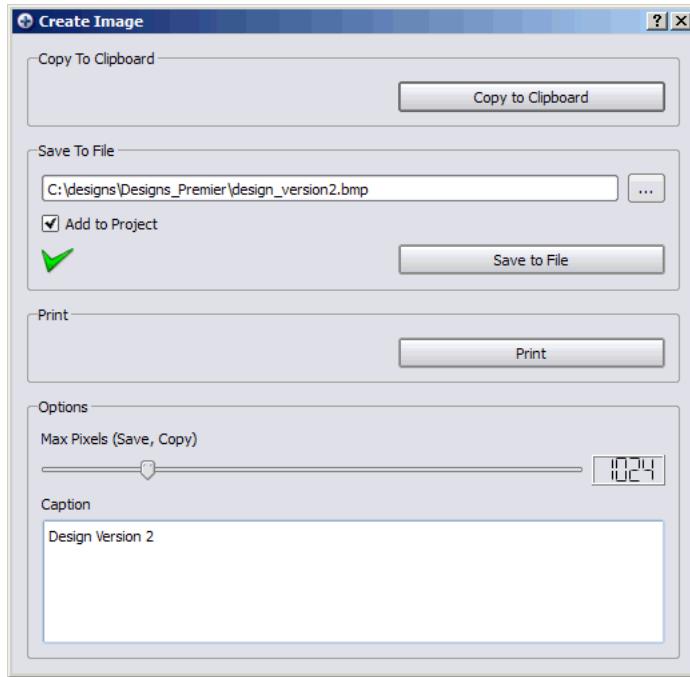
2. Available in the Certify Tool

Create Image Command

Select File->Create Image to create a capture image from any of the following views:

- HDL Analyst Views
- FSM Viewer
- Design Planner (*Synplify Premier*)
- Physical Analyst (*Synplify Premier*)

Drag the camera cursor to define the area for the image. When you release the cursor, the Create Image dialog box appears. Use the dialog box to copy the image, save the image to a file, or to print the image.



Field/Option	Description
Copy to Clipboard	Copies the image to the clipboard so you can paste it into a selected application (for example, a Microsoft Word file). When you copy an image to the clipboard, a green check mark appears in the Copy To Clipboard field.
Save to File	Saves the image to the specified file. You can save the file in a number of formats (platform dependent) including bmp, jpg, png, ppm, tif, xbm, and xpm.
Add to Project	Adds the saved image file to the Images folder in the Project view. This option is enabled by default.
Save to File button	You must click this button to save an image to the specified file. When you save the image, a green check mark appears in the Save To File field.

Field/Option	Description
Print	Prints the image. When you print the image, a green check mark appears in the Print field.
Options	Allows you to select the resolution of the image saved to a file or copied to the clipboard. Use the Max Pixels slider to change the image resolution.
Caption	Allows you to enter a caption for a saved or copied image. The overlay for the caption is at the top-left corner of the image.

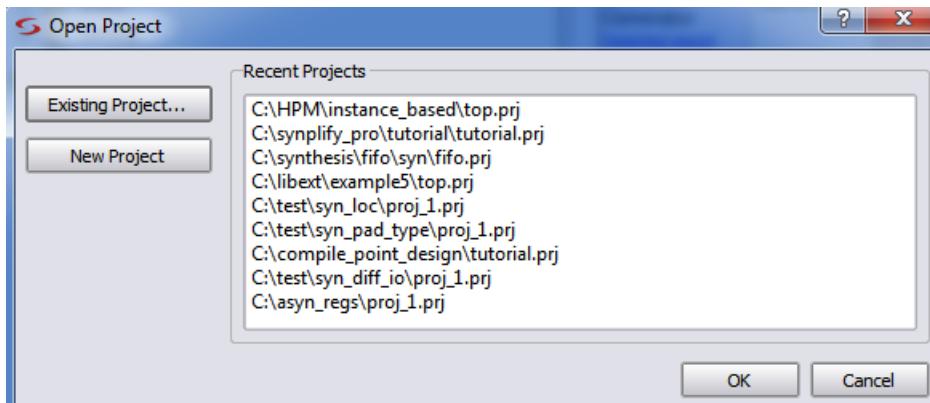
Build Project Command

Select File->Build Project to build a new project. This command behaves differently if an HDL file is open in the Text Editor.

- When an active Text Editor window with an HDL file is open, File->Build Project creates a project with the same name as the open file.
- If no file is open, File->Build Project prompts you to add files to the project using the Select Files to Add to New Project dialog box. The name of the new project is the name of the first HDL file added. See [Add Source File Command, on page 426](#).

Open Project Command

Select File->Open Project to open an existing project or to create a new project.



Field/Option	Description
Existing Project	Displays the Open Project dialog box for opening an existing project.
New Project	Creates a new project and places it in the Project view.

Edit Menu

You use the Edit menu to edit text files (such as HDL source files) in your project. This includes cutting, copying, pasting, finding, and replacing text; manipulating bookmarks; and commenting-out code lines. The Edit menu commands available at any time depend on the active window or view (Project, Text Editor, SCOPE spreadsheet, RTL, or Technology views).

The available Edit menu commands vary, depending on your current view. The following table describes all of the Edit menu commands:

Command	Description
Basic Edit Menu Commands	
 Undo	Cancels the last action.
 Redo	Performs the action undone by Undo.
 Cut	Removes the selected text and makes it available to Paste.
 Copy	Duplicates the selected text and makes it available to Paste.
 Paste	Pastes text that was cut (Cut) or copied (Copy).
Delete	Deletes the selected text.
 Find	Searches the file for text matching a given search string; see Find Command (Text) , on page 404. In the RTL view, opens the Object Query dialog box, which lets you search your design for instances, symbols, nets, and ports, by name; see Find Command (HDL Analyst) , on page 407. In the project view, searches files for text strings; see Find Command (In Project) , on page 405.

Command	Description
Find Next	Continues the search initiated by the last Find.
Find in Files	Performs a string search of the target files. See Find in Files Command , on page 411.

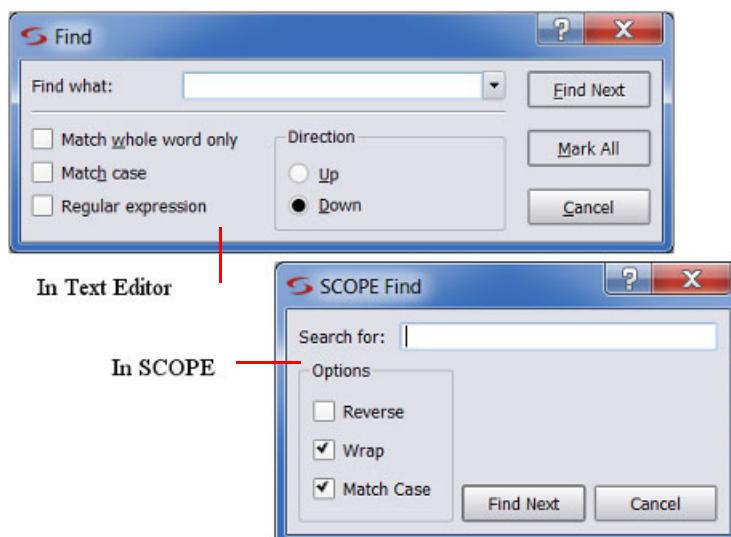
Edit Menu Commands for the Text Editor

Select All	Selects all text in the file.
Replace	Finds and replaces text. See Replace Command , on page 413.
Goto	Goes to a specific line number. See Goto Command , on page 414.
 Toggle bookmark	Toggles between inserting and removing a bookmark on the line that contains the text cursor.
 Next bookmark	Takes you to the next bookmark.
 Previous bookmark	Takes you to the previous bookmark.
 Delete all bookmarks	Removes all bookmarks from the Text Editor window.
Advanced->Comment Code	Inserts the appropriate comment prefix at the current text cursor location.
Advanced-> Uncomment Code	Removes comment prefix at the current text cursor location.
Advanced->Uppercase	Makes the selected string all upper case.
Advanced->Lowercase	Makes the selected string all lower case.
Select->All	Selects all text in the file (same as All).

Find Command (Text)

Select Edit->Find to display the Find dialog box. In the SCOPE window, the FSM Viewer, and the Text Editor window, the command has basic text-based search capabilities. Some search features, like regular expressions and line-number highlighting, are available only in the Text Editor. See [Find Command \(In Project\), on page 405](#), to search for files in the Project.

The HDL Analyst Find command is different; see [Find Command \(HDL Analyst\), on page 407](#) for details.

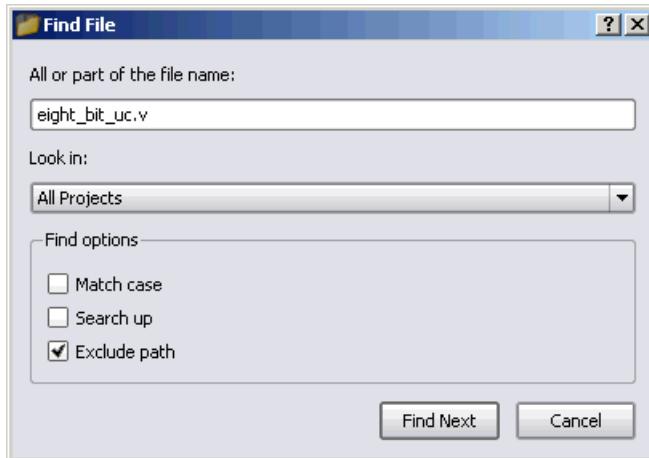


Field/Option	Description
Find What/Search for	Search string matching the text to find. In the text editor, you can use the pull-down list to view and reuse search strings used previously in the current session.
Match whole word only (text editor only)	When enabled, matches the entire word rather than a portion of the word.
Match Case	When enabled, searching is case sensitive.
Regular expression (text editor only)	When enabled, wildcard characters (*) and (?) can be used in the search string: ? matches any single character; * matches any string of characters, including an empty string.

Field/Option	Description
Direction/Reverse	Changes search direction. In the text editor, buttons select the search direction (Up or Down).
Find Next	Initiates a search for the search string (see Find What/Search for). In the text editor, searching starts again after reaching the end (Down) or beginning (Up) of the file.
Wrap (SCOPE only)	When enabled, searching starts again after reaching the end or beginning (Reverse) of the spread sheet.
Mark All (Text editor only)	Highlights the line numbers of the text matching the search string and closes the Find dialog box.

Find Command (In Project)

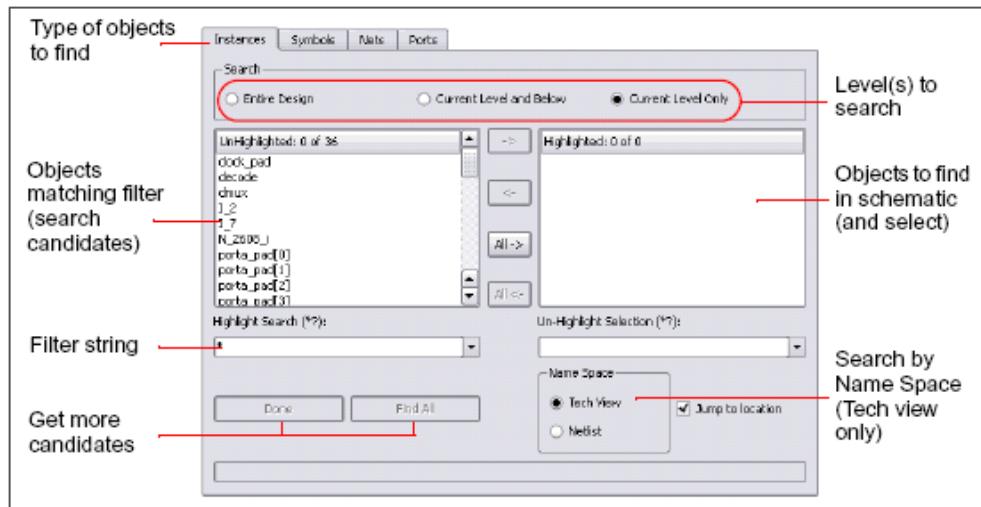
Select Edit->Find to display the Find File dialog box. In the Project view, the command has basic text-based search capabilities to locate files in the project.



Field/Option	Description
All or part of the file name	Search string matching the file to find. You can specify all or part of the file name.
Look in	Search for files in all projects or limit the search to files only in the specified project.
Match Case	When enabled, searching is case sensitive.
Search up	Searches in the up direction (search terminates when an end of tree is reached in either direction).
Exclude path	Excludes the path name during the search.
Find Next	Initiates a search for the file name string.

Find Command (HDL Analyst)

In the RTL or Technology view, use Edit->Find to display the Object Query dialog box. For a detailed procedure about using this command, see [Using Find for Hierarchical and Restricted Searches](#), on page 433 of the *User Guide*.



The available Find menu commands vary, depending on your current view. The following table describes all of the Find menu commands:

Field/Option	Description
Instances, Symbols, Nets, Ports	Tabbed panels for finding different kinds of objects. Choose a panel for a given object type by clicking its tab. In terms of memory consumption, searching for Instances is most efficient, and searching for Nets is least efficient.
Search	Where to search: Entire Design, Current Level & Below, or Current Level Only. See Using Find for Hierarchical and Restricted Searches , on page 433 of the <i>User Guide</i> .

Field/Option Description

UnHighlighted	<p>Names of all objects of the current panel type, in the level(s) chosen to Search, that match the Highlight Search (*?) filter. This list is populated by the Find 200 and Find All buttons.</p> <p>To select an object as a candidate for highlighting, click its name in this list. The complete name of the selected object appears near the bottom of the dialog box. You can select part or all of this complete name, then use the Ctrl-C keyboard shortcut to copy it for pasting.</p> <p>You can select multiple objects by pressing the Ctrl or Shift key while clicking; press Ctrl and click a selection to deselect it. The number of objects selected, and the total number listed, are displayed above the list, after the UnHighlighted: label: # selected of # total.</p> <p>To confirm a selection for highlighting and move the selected objects to the Highlighted list, click the -> button.</p>
Highlight Search (*?)	<p>Determines which object names appear in the UnHighlighted area, based on the case-sensitive filter string that you enter. For tips about using this field, see Using Wildcards with the Find Command, on page 436 of the <i>User Guide</i>.</p> <p>The filter string can contain the following wildcard characters:</p> <ul style="list-style-type: none"> • * (asterisk) - matches any sequence of characters • ? (question mark) - matches any single character • . (period) - does not match any characters, but indicates a change in hierarchical level. <p>Wildcards * and ? only match characters within the current hierarchy level; a*b*, for example, will not cross levels to match alpha.beta (where the period indicates a change in hierarchy).</p> <p>If you must match a period character occurring in a name, use \ (backslash period) in the filter string. The backslash prevents interpreting the period as a wildcard.</p> <p>The filter string is matched at each searched level of the hierarchy (the Search levels are described above). Use filter strings that are as specific as possible to limit the number of unwanted matches.</p> <p>Unnecessarily extensive search can be costly in terms of memory performance.</p>
->	Moves the selected names from the UnHighlighted area to the Highlighted area, and highlights their objects in the RTL and Technology views.
<-	Moves the selected names from the Highlighted area to the UnHighlighted area, and unhighlights their objects in the RTL and Technology views.

Field/Option	Description
All ->	Moves all names from the UnHighlighted to the Highlighted area, and highlights their objects in the RTL and Technology views.
<- All	Moves all names from the Highlighted to the UnHighlighted area, and unhighlights their objects in the RTL and Technology views.
Highlighted	Complementary and analogous to the UnHighlighted area. You select object names here as candidates for moving to the UnHighlighted list. (You move names to the UnHighlighted list by clicking the <- button which unselects and unhighlights the corresponding objects.) When you select a name in the Highlighted list, the view is changed to show the (original, unfiltered) schematic sheet containing the object.
Un-Highlight Selection (*?)	Complementary and analogous to the Highlight Search area: selects names in the Highlighted area, based on the filter string you input here.
Jump to location	When enabled, jumps to another sheet if necessary to show target objects.
Name Space: Tech View	Searches for the specified name using the mapped (.srm) database. For more information, see Using Find for Hierarchical and Restricted Searches , on page 433 of the <i>User Guide</i> .
Name Space: Netlist	Searches for the specified name using the output netlist file. For more information, see Using Find for Hierarchical and Restricted Searches , on page 433 of the <i>User Guide</i> .

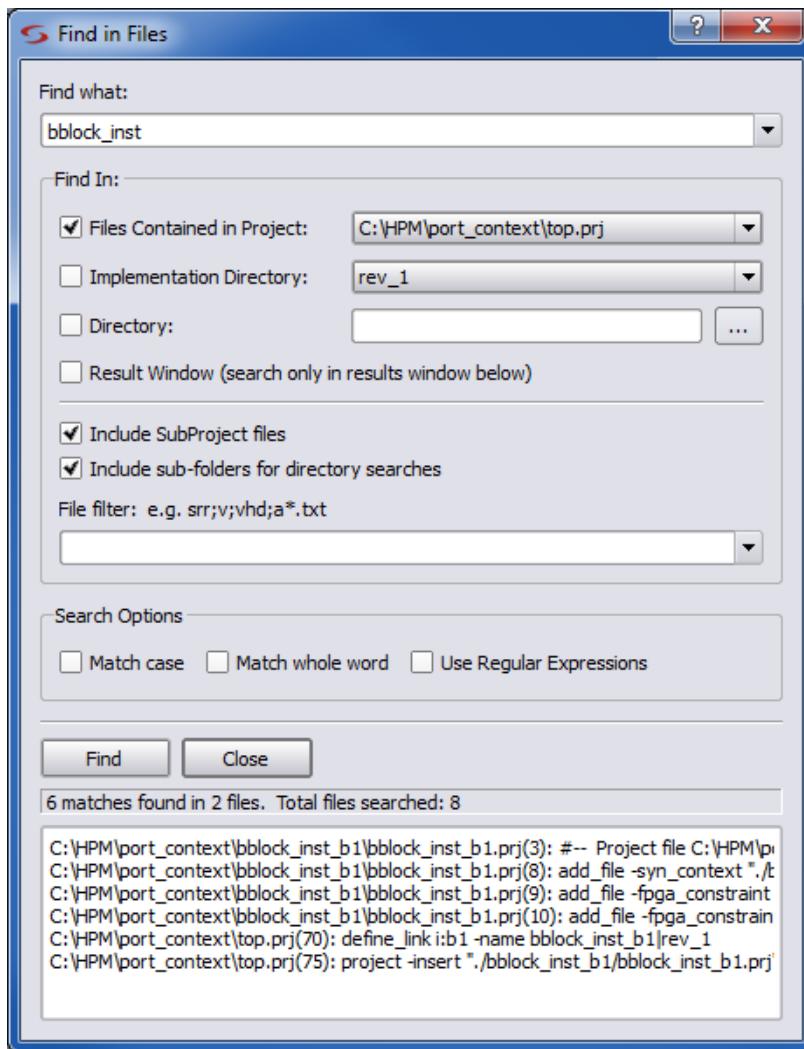
Field/Option Description

Find 200	<p>Adds up to 200 more objects that match the filter string to the UnHighlighted list. This button becomes available after you enter a Highlight Search (*?) filter string. This button does not find objects in HDL Analyst views. It matches names of design objects against the Highlight Search (*?) filter and provides the candidates listed in the UnHighlighted list, from which you select the objects to find.</p> <p>Using the Enter (Return) key when the cursor is in the Highlight Search (*?) field is equivalent to clicking the Find 200 button.</p> <p><i>Usage note:</i></p> <p>Click Find 200 before Find All to prevent unwanted matches in case the Highlight Search (*) string is less selective than you expect.</p>
Find All	<p>Places all objects that match the Highlight Search (*) filter string in the UnHighlighted list. This button does not find objects in HDL Analyst views. It matches names of design objects against the Highlight Search (*) filter and provides the candidates listed in the UnHighlighted list, from which you select the objects to find. (Enter a filter string before clicking this button.) See <i>Usage Note</i> for Find 200, above.</p>

For more information on using the Object Query dialog box, see [Using Find for Hierarchical and Restricted Searches](#), on page 433 of the *User Guide*.

Find in Files Command

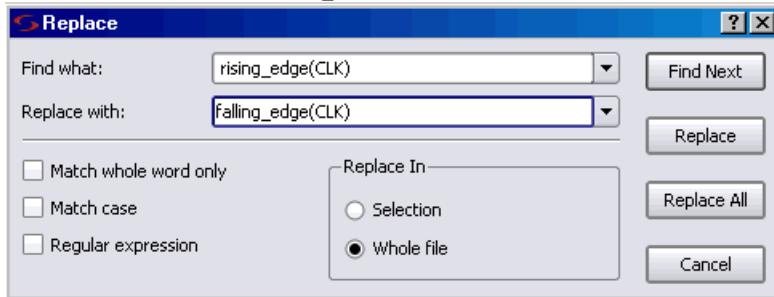
The Find in Files command searches the defined target for the occurrence of a specified search string. The list of files containing the string is reported in the display area at the bottom of the dialog box. For information on using this feature, see [Searching Files, on page 184](#) of the *User Guide*.



Field/Option	Description
Find what	Text string object of search.
Files Contained in Project	Drop-down menu identifying the source project of the files to be searched.
Implementation Directory	Drop-down menu restricting project search to a specific implementation or all implementations.
Directory	Identifies directory for files to be searched.
Result Window	Allows a secondary search string (Find what) to be applied to the targets reported from the initial search.
Include SubProject files	When checked, extends the search to include subproject files of the top-level project file.
Include sub-folders for directory searches	When checked, extends the search to sub-directories of the target directory.
File filter	Excludes files from the search by filename extension.
Search Options	Standard string search options; check to enable.
Find	Initiates search.
Result Display	List of files containing search string. Status line lists the number of matches in each file and the number of files searched.

Replace Command

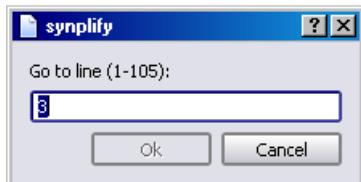
Use Edit->Replace to find and optionally replace text in the Text Editor.



Feature	Description
Find what	Search string matching the text to find. You can use the pull-down list to view and reuse search strings used previously in the current session.
Replace with	The text that replaces the found text. You can use the pull-down list to view and reuse replacement text used previously in the current session.
Match whole word only	Finds only occurrences of the exact string (strings longer than the Find what string are not recognized).
Match case	When enabled, searching is case sensitive.
Regular expression	When enabled, wildcard characters (*) and (?) can be used in the search string: ? matches any single character; * matches any string of characters, including the empty string.
Selection	Replace All replaces only the matched occurrence.
Whole file	Replace All replaces all matching occurrences.
Find Next	Initiates a search for the search string (see Find What).
Replace	Replaces the found text with the replacement text, and locates the next match.
Replace All	Replaces all text that matches the search string.

Goto Command

Use Edit->Goto to go to a specified line number in the Text Editor.



View Menu

Use the View menu to set the display and viewing options, choose toolbars, and display result files. The commands in the View menu vary with the active view. The following tables describe the View menu commands in various views.

- [View Menu Commands: All Views](#), on page 415
- [View Menu: Zoom Commands](#), on page 416
- [View Menu: RTL and Technology Views Commands](#), on page 416
- [View Menu: FSM Viewer Commands](#), on page 417

View Menu Commands: All Views

Command	Description
Font Size	Changes the font size in the Project UI of the synthesis tools. You can select one of the following options: <ul style="list-style-type: none">• Increase Font Size• Decrease Font Size• Reset Font Size (default size)
Toolbars	Displays the Toolbars dialog box, where you specify the toolbars to display. See Toolbar Command , on page 418.
Status Bar	When enabled, displays context-sensitive information in the lower-left corner of the main window as you move the mouse pointer over design elements. This information includes element identification.
Refresh	Updates the UI display of project files and folders.
Output Windows	<i>Synplify Pro, Synplify Premier</i> Displays or removes the Tcl Script/Messages and Watch windows simultaneously in the Project view. Refer to the Tcl Window and Watch Window options for more information.
Tcl Window	<i>Synplify Pro, Synplify Premier</i> When enabled, displays the Tcl Script and Messages windows. All commands you execute in the Project view appear in the Tcl window. You can enter or paste Tcl commands and scripts in the Tcl window. Check for notes, warning, and errors in the Messages window.

Command	Description
Watch Window	<i>Synplify Pro, Synplify Premier</i> When enabled, displays selected information from the log file in the Watch window.
View Log File	Displays a log file report that includes compiler, mapper, and timing information on your design. See View Log File Command , on page 420.
View Result File	Displays a detailed netlist report.

View Menu: Zoom Commands

Command	Description
 Zoom In	Lets you Zoom in or out. When selected, a Z-shaped mouse pointer (\mathbb{Z}) appears. Zoom in or out on the view by clicking or dragging a box around (lassoing) the region. Clicking zooms in or out on the center of the view; lassoing zooms in or out on the lassoed region. Right-click to exit zooming mode.
 Zoom Out	In the SCOPE spreadsheet, selecting these commands increases or decreases the view in small increments.
Pan	Lets you pan (scroll) a schematic or FSM view using the mouse. If your mouse has a wheel feature, use the wheel to pan up and down. To pan left and right, use the Shift key with the wheel.
 Full View	Zooms the active view so that it shows the entire design.
 Normal View	Zooms the active view to normal size and centers it where you click. If the view is already normal size, clicking centers the view.

View Menu: RTL and Technology Views Commands

These commands are available when the RTL view or Technology view is active. These commands are available in addition to the commands described in [View Menu Commands: All Views](#), on page 415 and [View Menu: Zoom Commands](#), on page 416.

Command	Description
 Push/Pop Hierarchy	Traverses design hierarchy using the push/pop mode - see Exploring Design Hierarchy (Standard), on page 423 of the <i>User Guide</i> .
 Previous Sheet	Displays the previous sheet of a multiple-sheet schematic.
 Next Sheet	Displays the next sheet of a multiple-sheet schematic.
View Sheets	Displays the Goto Sheet dialog box where you can select a sheet to display from a list of all sheets. See View Sheets Command , on page 419 .
Visual Properties	Toggles the display of information for nets, instances, pins, and ports in the HDL Analyst view. To customize the information that displays, set the values with Options->HDL Analyst Options->Visual Properties. See Visual Properties Panel , on page 596 .
 Back	Goes backward in the history of displayed sheets for the current HDL Analyst view.
 Forward	Goes forward in the history of displayed sheets for the current HDL Analyst view.
Filter	Filters the RTL/Technology view to display only the selected objects.

View Menu: FSM Viewer Commands

Synplify Pro, Synplify Premier

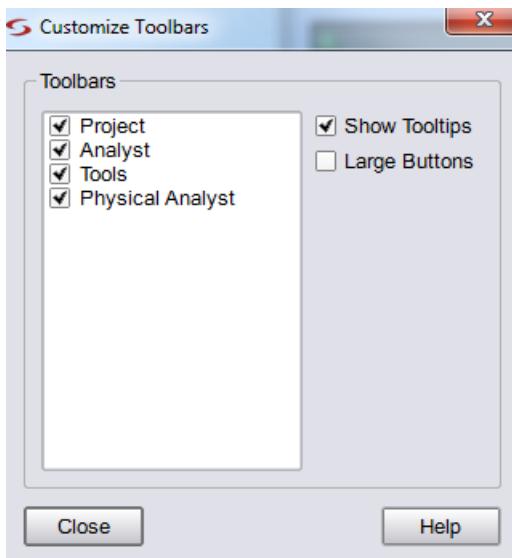
The following commands are available when the FSM viewer is active. These commands are in addition to the common commands described in [View Menu Commands: All Views, on page 415](#) and [View Menu: Zoom Commands, on page 416](#).

Command	Description
 Filter->Selected	Hides all but the selected state(s).
 Filter->By output transitions	Hides all but the selected state(s), their output transitions, and the destination states of those transitions.
Filter->By input transitions	Hides all but the selected state(s), their input transitions, and the origin states of those transitions.
Filter->By any transition	Hides all but the selected state(s), their input and output transitions, and their predecessor and successor states.
 Unfilter	Restores a filtered FSM diagram so that all the states and transitions are showing.
Cross Probing	Enables cross probing between FSM nodes and RTL view schematic.
Select All States	Selects all the states.
 FSM Table	Toggles display of the transition table.
FSM Graph	Toggles FSM state diagram on or off.
Annotate Transitions	Toggles display of state transitions on or off on FSM state diagram.
Selection Transcription	
Tool Tips	Toggles state diagram tool tips on or off.
FSM Properties	Displays FSM Properties dialog box.
Unselect All	Unselects all states and transitions.

Toolbar Command

Select View->Toolbars to display the Toolbars dialog box, where you can:

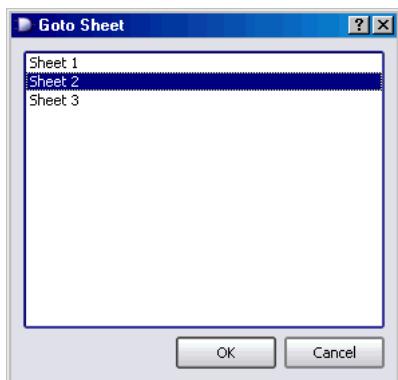
- Choose the toolbars to display
- Customize their appearance



Feature	Description
Toolbars	Lists the available toolbars. Select the toolbars that you want to display.
Show Tooltips	When selected, a descriptive tooltip appears whenever you position the pointer over an icon.
Large Buttons	When selected, large icons are used.

View Sheets Command

Select View->View Sheets to display the Goto Sheet dialog box and select a sheet to display. The Goto Sheet dialog box is only available in an RTL or Technology view, and only when a multiple-sheet design is present.

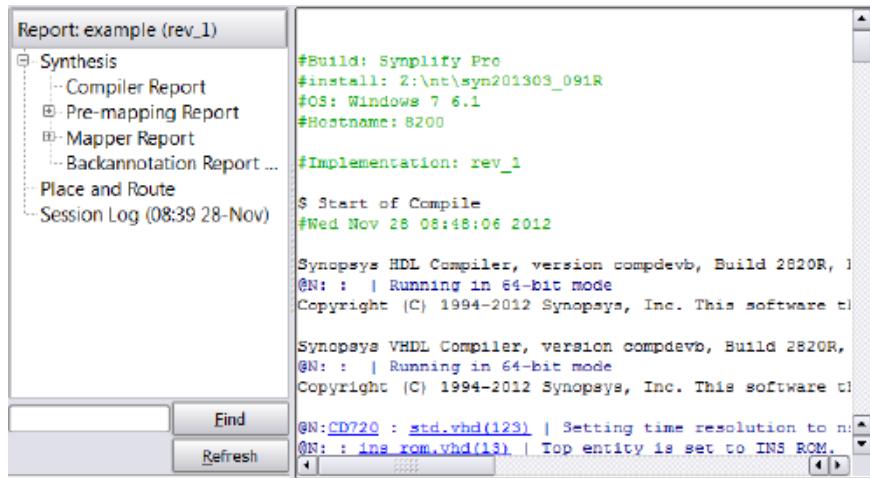


To see if your design has multiple sheets, check the sheet count display at the top of the schematic window.

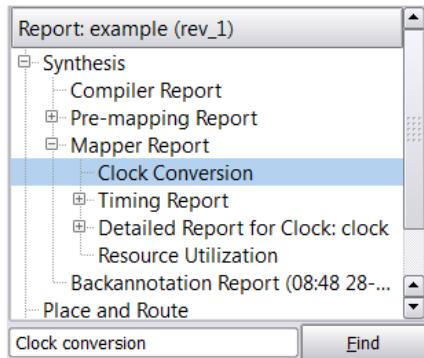
View Log File Command

View->View Log File displays the log file report for your project. The log file is available in either text (*project_name.srr*) or HTML (*project_name_srr.htm*) format. To enable or disable the HTML file format for the log file, select the View log file in HTML option in the Options->Project View Options dialog box.

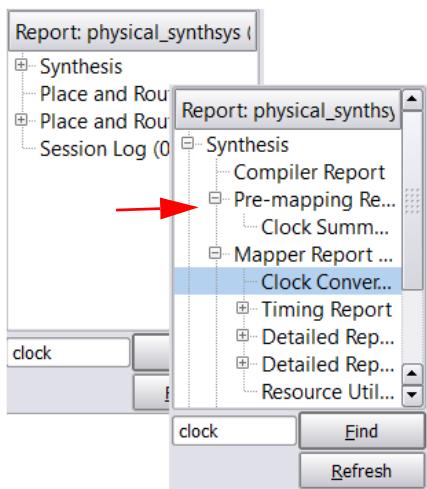
When opening the log file, a table of contents appears. Selecting an item from the table of contents takes you to the corresponding HTML page. To go back, right-click the HTML page and select Back from the menu.



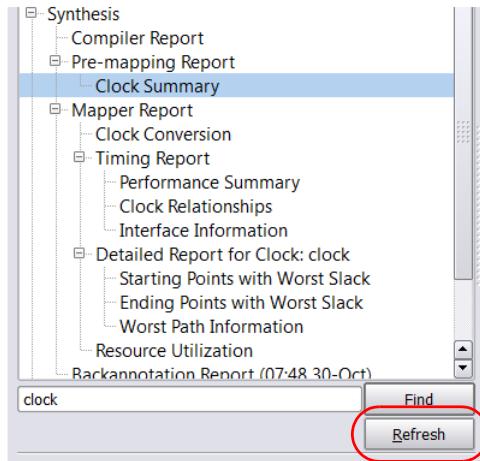
You can use the search field to find an item in the table of contents. Enter all or part of the header name in the search field, then click Find. The log file displays the resulting section.



Find searches within collapsed tables. It expands the tables to show your results.



If the file changes while the search window is open, click the Refresh button to update the table of contents.



Project Menu

You use the Project menu to set implementation options, add or remove files from a project, change project filenames, create new implementations, and archive or copy the project. The Project menu commands change, depending on the view you are in. For example, the HDL Analyst RTL and Technology views only include a subset of the Project menu commands.

Synplify Pro, Synplify Premier

The tool provides a graphical user interface (GUI) with views that help you manage hierarchical designs that can be synthesized independently and imported back to the top-level project in a team design flow called Hierarchical Project Management. For details about using the team design flow, see [Hierarchical Project Management Flows, on page 44](#).

The following table describes the Project menu commands.

Command	Description
Implementation Options	Displays the Implementation Options dialog box, where you set options for implementing your design. See Implementation Options Command , on page 444 .
Design Intent	<i>Synplify Premier</i> Opens a dialog box with pre-defined implementation options intended to configure the synthesis tool for fast turnaround and maximum QoR tasks. See Design Intent , on page 425 .
Add Source File	Displays the Select Files to Add to Project dialog box. See Add Source File Command , on page 426 . Tcl equivalent: add_file -fileType filename
Remove Implementation	Displays the Remove Implementation dialog box that allows you to remove the selected implementation. See Remove Implementation , on page 428 . Tcl equivalent: impl -remove implementationName
Remove File From Project	Removes selected files from your project. Tcl equivalent: project_file -remove filename
Change File	Replaces the selected file in your project with another that you choose. See Change File Command , on page 429 . Tcl equivalent: project_file -name "originalFile" "newFile"

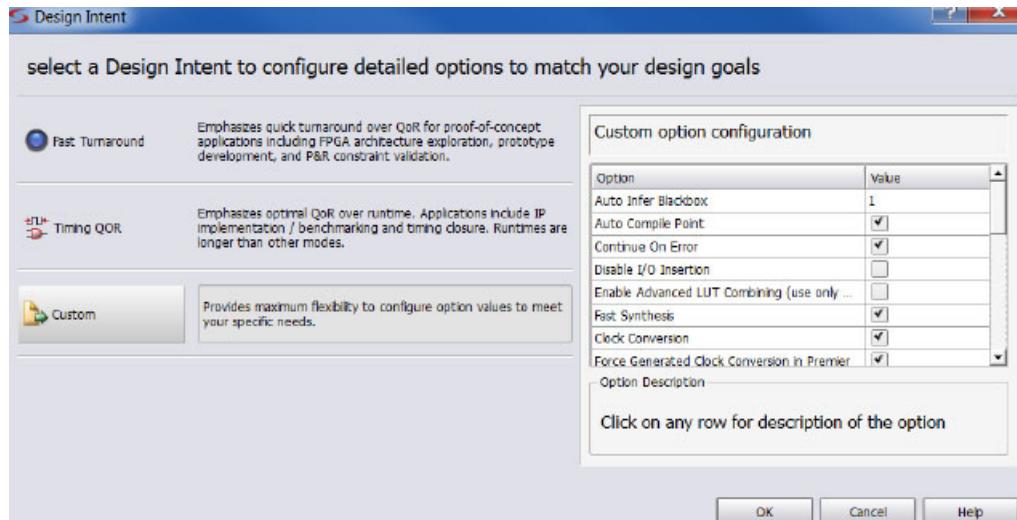
Command	Description
Set VHDL Library	Displays the File Options dialog box, where you choose the library (Library Name) for synthesizing VHDL files. The default library is called work. See <i>Set VHDL Library Command</i> , on page 429.
Add Implementation	Creates a new implementation for a current design. Each implementation pertains to the same design, but it can have different options settings and/or constraints for synthesis runs. See <i>Add Implementation Command</i> , on page 430). Tcl equivalent: <code>impl -add implementation_1 implementation -type implementationType</code>
New Identify Implementation	Creates a new Identify implementation for a current design. To launch the Identify toolset, see the <i>Identify Instrumentor Command</i> , on page 506 and <i>Launch Identify Debugger Command</i> , on page 507. Tcl equivalent: <code>impl -add implementation_1 implementation</code>
Convert Vendor Constraints	Xilinx Use this vendor constraints utility to automatically convert UCF constraints to SDC format. See <i>Convert Vendor Constraints Command</i> , on page 430.
Archive Project	Archives a design project. Use this command to archive a full or partial project, or to add files to or remove files from an archived project. See <i>Archive Project Command</i> , on page 433 for a description of the utility wizard options.
Un-Archive Project	Loads an archived project file to the specified directory. See <i>Un-Archive Project Command</i> , on page 435 for a description of the utility wizard options.

Command	Description
Copy Project	Creates a copy of a full or partial design project. See Copy Project Command , on page 438 for a description of the utility wizard options.
Hierarchical Project Options	<i>Synplify Pro, Synplify Premier</i> Configures how to run synthesis for a subproject or top-level project. See Hierarchical Project Options Command , on page 441 .
Add SubProject Implementation	<i>Synplify Pro, Synplify Premier</i> Adds new implementations to the blocks in the top-level project. This command is only available when the top-level implementation is selected in the Project Files view. See Working with Multiple Implementations in Hierarchical Projects, on page 166 in the <i>User Guide</i> .

Design Intent

Synplify Premier

Selecting Project->Design Intent from the main menu opens a dialog box with pre-defined implementation options intended to configure the synthesis tool for specific tasks such as fast turnaround or maximum timing QoR results.

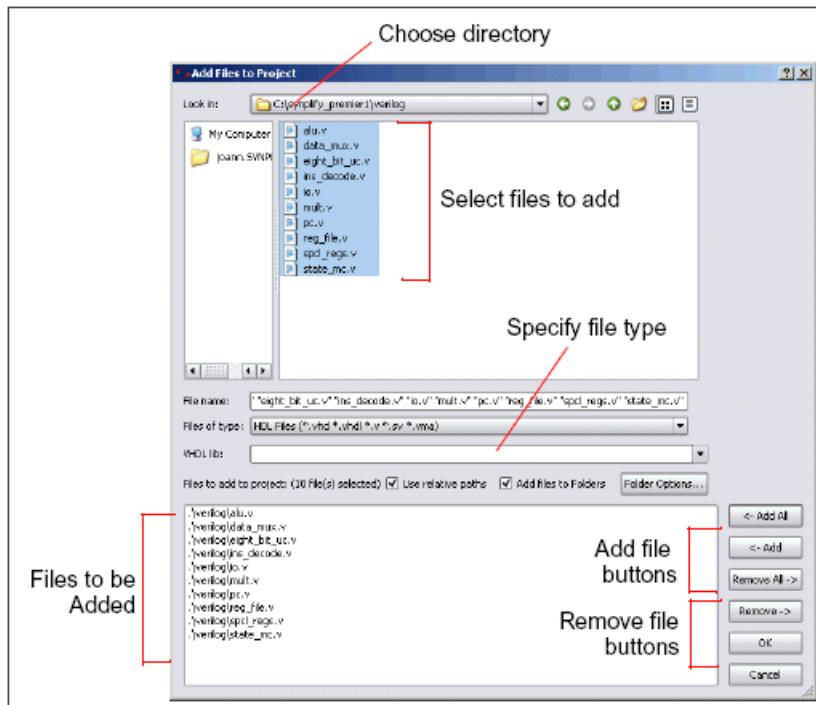


To use the dialog box, select the design intent from the list of icons on the left. The icon selected fills in the corresponding options in the table on the right. You can change any of the recommended settings prior to clicking the OK button. The available design intent selections are outlined in the following table.

Icon Selected	Synthesis Tool Configuration
 Fast Turnaround	Sets the synthesis options for fast turnaround. See Ensuring Fast Turnaround for Prototypes, on page 1217 for a detailed procedure. Tcl equivalent: set_option -design_intent fast_turn_around
 Timing QoR	Sets the synthesis options for maximum timing QoR. See Optimizing QoR for Prototypes, on page 1218 for a detailed procedure. Tcl equivalent: set_option -design_intent timing_qor
 Custom	Allows the user to redefine the set of synthesis options for the active design intent. Tcl equivalent: set_option -design_intent custom

Add Source File Command

Select Project->Add Source File to add files, such as HDL source files, to your project. This selection displays the Select Files to Add to Project dialog box.



Feature	Description
Look in	The directory of the file to add. You can use the pull-down directory list or the Up One Level button to choose the directory.
File name	The name of a file to add to the project. If you enter a name using the keyboard, then you must include the file-type extension.
Files of type	The type (extension) of files to be added to the project. Only files in the active directory that match the file type selected from the drop-down menu are displayed in the list of files. Use All Files to list all files in the directory.

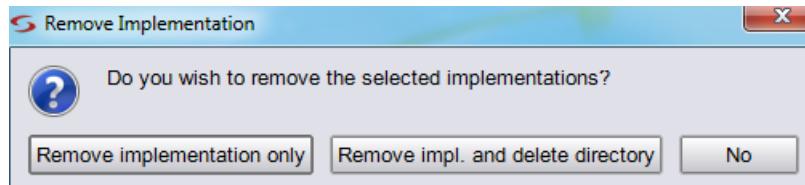
Feature	Description
Files To Add To Project	The files to add to the project. You add files to this list with the <-Add and <-Add All buttons. You remove files from this list with the Remove -> and Remove All -> buttons. For information about adding files to custom folders, see Creating Custom Folders, on page 113 . Tcl equivalent: <code>add_file -type filename</code>
Use relative paths	When you add files to the project, you can specify either to use the relative path or full path names for the files.
Add files to Folders	When you add files to the project, you can specify whether or not to automatically add the files to folders. See the Folder Options described below.
Folder Options	When you add files to folders, you can specify the folder name as either the: <ul style="list-style-type: none"> • Operating System (OS) folder name • Parent path name from a list provided in the display

Remove Implementation

Displays the Remove Implementation dialog box that allows you to remove the selected implementation. You can select any of the following:

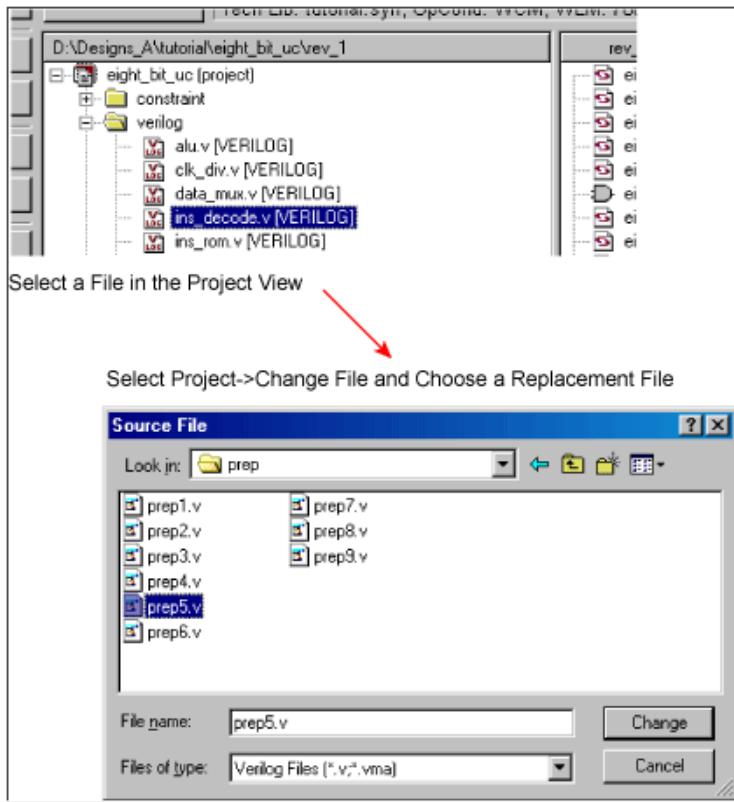
- Remove implementation only - Removes the implementation from the project only.
- Remove impl. and delete directory - Removes the implementation from the project and deletes the directory on the disk.
- No - Do not remove the implementation.

Choose the appropriate option shown in the dialog box below.



Change File Command

Select Project->Change File to replace a file in the project files list with another of the same type. This displays the Source File dialog box, where you specify the replacement file. You must first select the file to replace, in the Project view, before you can use this command.



Set VHDL Library Command

Select Project->Set VHDL Library to display the File Options dialog box, where you view VHDL file properties and specify the VHDL library name. See [File Options Popup Menu Command, on page 622](#). This is the same dialog box as that displayed by right-clicking a VHDL filename in the Project view and choosing File Options.

Add Implementation Command

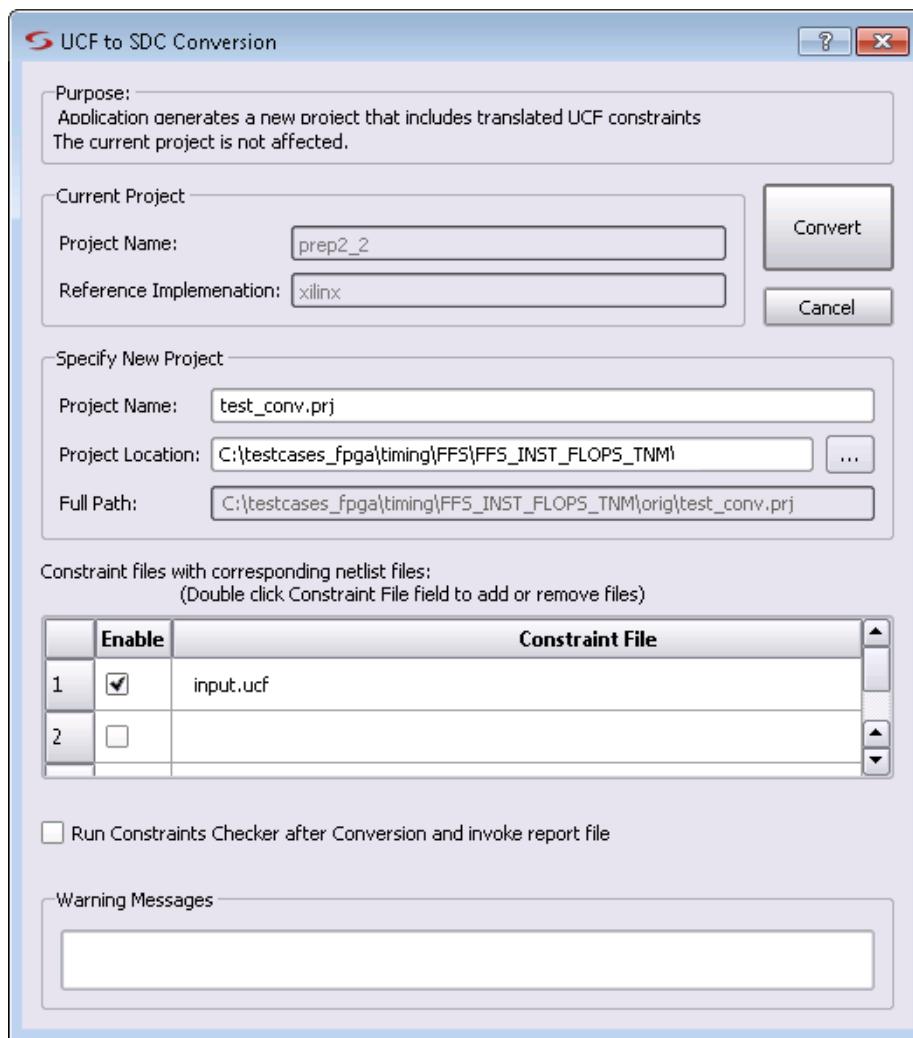
Select Project->Add Implementation to create a new implementation for the selected project. This selection displays the Implementation Options dialog box, where you define the implementation options for the project - see [Implementation Options Command, on page 444](#). This is the same dialog box as that displayed by Project->Implementation Options, except that there is no list of Implementations to the right of the tabbed panels.

Convert Vendor Constraints Command

Xilinx

Use Project->Convert Vendor Constraints to automatically translate Xilinx UCF constraints to SDC format from the information specified in the dialog box. This application creates a new project that contains all supported UCF constraints converted to SDC constraints. However, your original project and project files are not altered.

See [Converting and Using Xilinx UCF Constraints, on page 275](#) of the *User Guide* for details on translating the constraints.



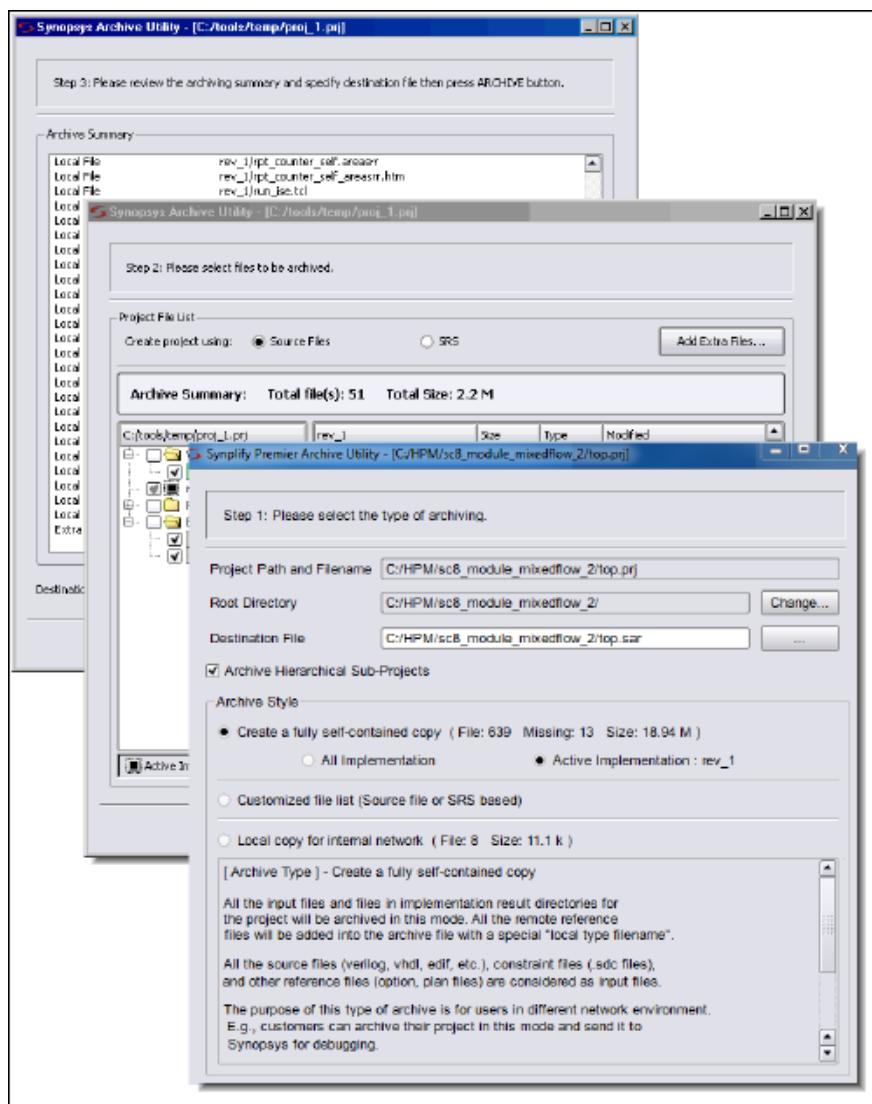
The UCF to SDC Conversion dialog box contains the following options:

Command	Description
Current Project	Automatically displays the current project name and implementation. You cannot edit these fields.
• Project Name	
• Reference Implementation	
Project Name	Specifies a name for the new project to be created.
Project Location	Specifies a location for the project to be created.
Full Path	Displays the full path for the new project.
Constraint files with corresponding netlist files	<p>Lists the Xilinx constraint files added to the current project. Enable the files you want to translate. You can add or remove files by double-clicking the constraint files.</p> <p>This command does not work on ucf files generated by the Synopsys FPGA tools. For details, see Supported Input Files for UCF Conversion, on page 279 in the <i>User Guide</i>.</p>
Run Constraints Checker after Conversion and invoke report file	Determines whether the constraint checker is run after translation, and whether a report file is generated.
Warning Messages	Displays messages about the status of constraint translation or constraint checker messages.
Convert	When clicked, it translates UCF constraints to SDC.

Archive Project Command

Use the Project->Archive Project command to store files for a design project into a single archive file in Synopsys Proprietary Format (.sar). You can archive an entire project or selected files from the project.

The Archive Project command displays the Synopsys Archive Utility wizard consisting of either two (all files archived) or three (custom file selection) tabs.



Option	Description
Project Path and Filename	Path and filename of the .prj file.
Root Directory	Top-level directory that contains the project files.
Destination Directory	Pathname of the directory to store the archive .sar file.

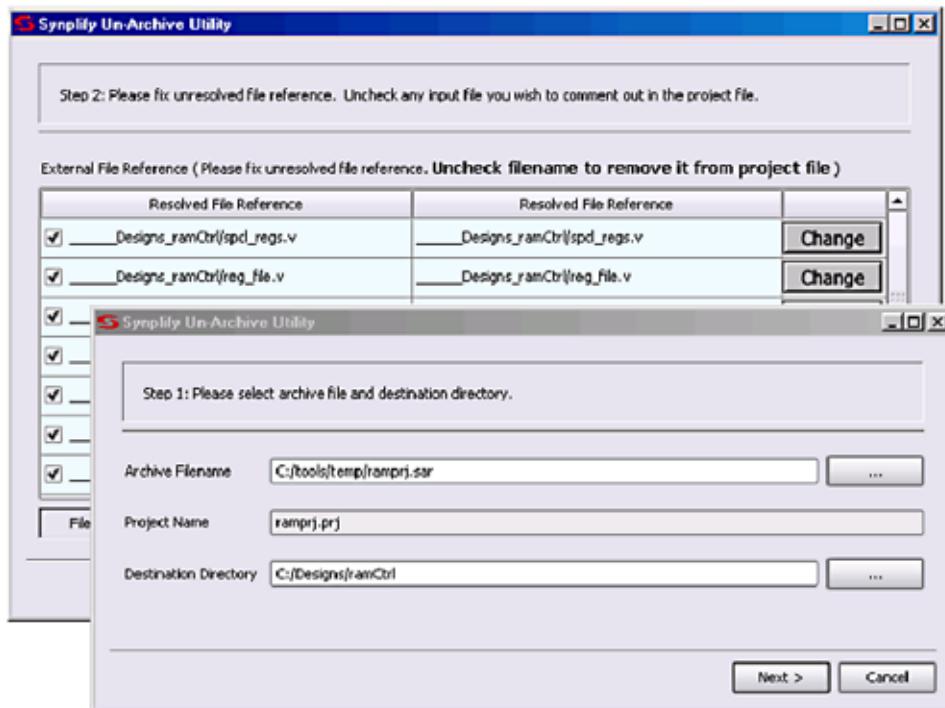
Option	Description
Archive Hierarchical Sub-Projects	<p>If you select the Archive Hierarchical Sub-Projects option in the wizard, you can archive subprojects for the active implementation of a Hierarchical Project Management (HPM) design.</p> <p>When you disable this option, only the top-level project is archived.</p>
Archive Style	<p>The type of archive:</p> <ul style="list-style-type: none"> • Create a fully self-contained copy - all project files are archived; includes project input files and result files. • If the project contains more than one implementation: <ul style="list-style-type: none"> - All Implementation includes all implementations in the project. - Active Implementation includes only the active implementation. • Customized file list - only project files that you select are included in the archive. • Local copy for internal network - only project input files are archived, no result files will be included.
Create Project using	<p>If you select the Customized file list option in the wizard, you can choose one the following options on the second tab:</p> <ul style="list-style-type: none"> • Source Files - Includes all design files in the archive. You cannot enable the SRS option if this option is enabled. • SRS - Includes all .srs files (RTL schematics) in the archive. You cannot enable the Source Files option when this option is enabled.
Add Extra Files	<p>If you select the Customized file list option in the wizard, you can use this button on the second tab to add additional files to the archive.</p>

For step-by-step details on how to use the archive utility, see [Archive Project Command, on page 433](#).

Un-Archive Project Command

Use the Project->Un-Archive Project command to extract the files from an archived design project.

This command displays a Synplify Un-Archive Utility wizard.

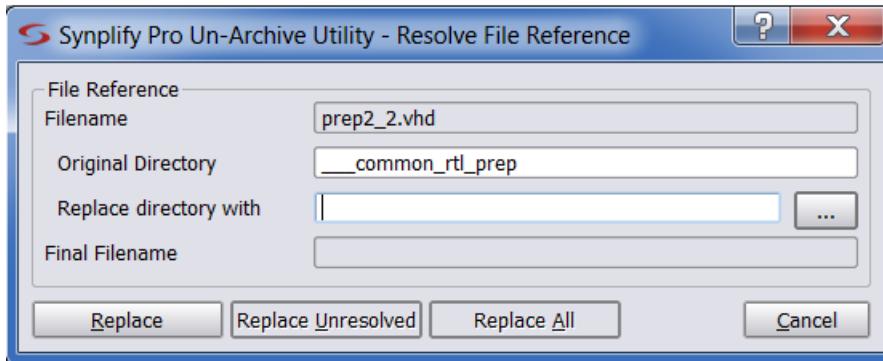


Option	Description
Archive Filename	Path and filename of the .prj file.
Project Name	Top-level directory that contains the project files.
Destination Directory	Pathname of the directory to store the archive .sar file.
Original File Reference/ Resolved File Reference	<p>Displays the files in the archive that will be extracted. You can exclude files from the .sar by unchecking the file in the Original File Reference list. Any unchecked files are commented out in the .prj file.</p> <p>If there are unresolved reference files in the .sar file, you must fix (Resolve button) or uncheck them. Or, if there are files that you want to change when project files are extracted, use the Change button and select files, as appropriate. See Resolve File Reference, next for more details.</p>

For step-by-step details on how to use the un-archive utility, see [Un-Archive Project Command, on page 435](#).

Resolve File Reference

When you use the Un-Archive Utility wizard to extract a project, if there are unresolved file references, use the Resolve button next to the file to point to a new file location. You can also optionally replace project files in the destination directory by clicking the Change button next to the file you want to replace. The Change and Resolve buttons bring up the following dialog box:



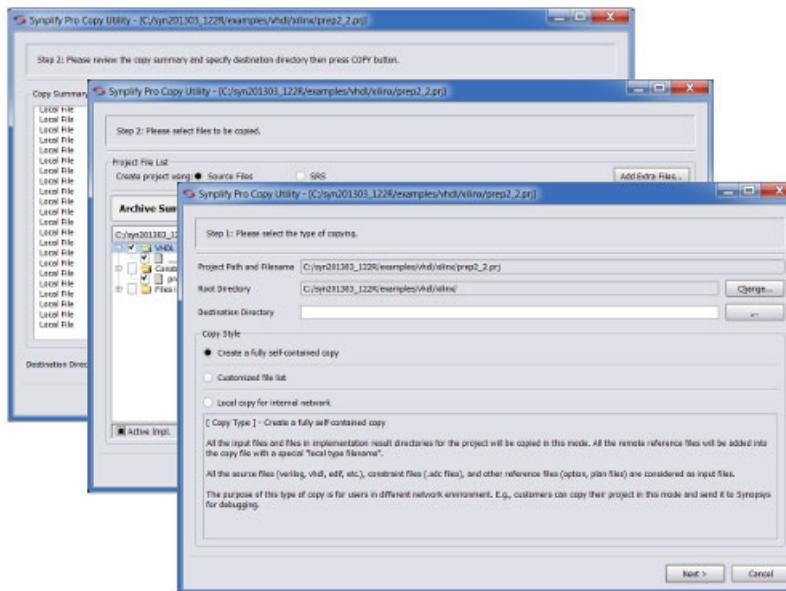
Option	Description
Filename	Specifies the path and name of the file you want to change or resolve.
Original Directory	Specifies the location of the project at the time it was archived.

Option	Description
Replace directory with	Specifies the new location of the project files you want to use to replace files.
Final Filename	Specifies the path name of the directory and the file name of the replace file.
Replace buttons	<ul style="list-style-type: none">• Replace - replaces only the file specified in the Filename field when the project is extracted.• Replace Unresolved - replaces any unresolved files in the project, with files of the same name from the Replace directory.• Replace All - replaces all files in the archived project with files of the same name from the Replace directory.• To undo any replace-file references, clear the Replace directory with field, then click Replace. This causes the utility to point back to the Original Directory and filenames.

Copy Project Command

Use the Project->Copy Project command to create a copy of a design project. You can copy an entire project or selected files from the project.

The Copy Project command displays the Synopsys Copy Utility wizard consisting of either two (all files copied) or three (custom file selection) tabs.



Option	Description
Project Path and Filename	Path and filename of the .prj file.
Root Directory	Top-level directory that contains the project files.
Destination Directory	Pathname of the directory to store the archive .sar file.

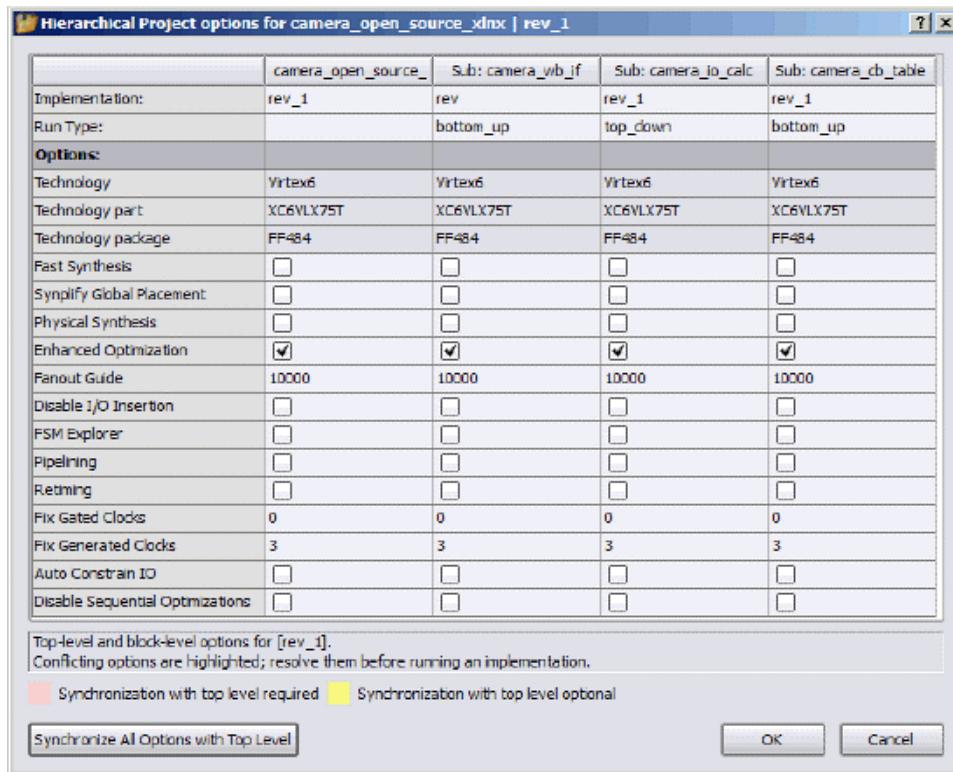
Option	Description
Copy Style	<p>The type of archive:</p> <ul style="list-style-type: none">• Create a fully self-contained copy - all project files are archived; includes project input files and result files.• If the project contains more than one implementation:<ul style="list-style-type: none">- All Implementation includes all implementations in the project.- Active Implementation includes only the active implementation.• Customized file list - only project files that you select are included in the archive.• Local copy for internal network - only project input files are archived, no result files will be included.
Create Project using	If you select the Customized file list option in the wizard, you can choose one the following options on the second tab: <ul style="list-style-type: none">• Source Files - Includes all design files in the archive. You cannot enable the SRS option if this option is enabled.• SRS - Includes all .srs files (RTL schematics) in the archive. You cannot enable the Source Files option if this option is enabled.
Add Extra Files	If you select the Customized file list option in the wizard, you can use this button on the second tab to add additional files to the archive.

For step-by-step details on how to use the copy utility, see [Copy Project Command, on page 438](#).

Hierarchical Project Options Command

Synplify Pro, Synplify Premier

Use this command to configure synthesis runs for a subproject or top-level project when you are working with hierarchical designs. You can determine which subprojects to run, how to run each project (top-down or bottom-up), and synchronize options across all implementations. Most of the default implementation options displayed in this dialog box are automatically filled in for the target device you specify for synthesis. For more information about using this command, see [Configuring Synthesis Runs for Hierarchical Projects, on page 178](#) in the *User Guide*.

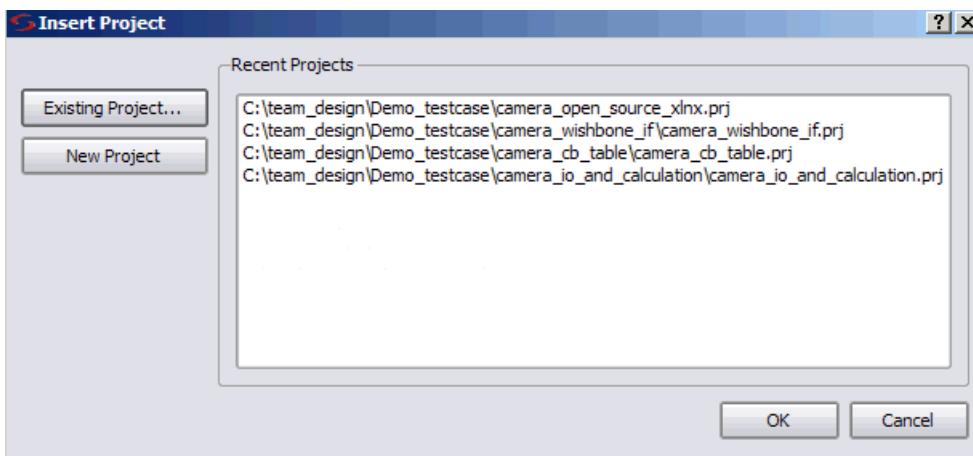


The following table describes the Run Configuration options for the top-level project and its subprojects.

Edit Run Configuration Command	Description
Implementation	Selects the implementation, such as rev_1, to synthesize for the subproject. If you do not want to run synthesis, choose <off> from the pull-down menu.
Run Type	You can choose the following run types: <ul style="list-style-type: none"> • top_down - the subproject .srs files are imported back to the top-level project for final assembly. • bottom_up - design block .srd/.edif files for the subproject are imported back to the top-level project for final assembly. This is the default.
Options	Most default options are automatically filled in with the target device you specify for synthesis from the Device tab of the Implementation Options panel.
Synchronize All Options with Top Level	Synchronizes device options for the top-level project and subprojects so that they all match the top-level project. When you create subprojects, the top-level and block-level device option settings can vary. Running the design with conflicting options can cause errors during synthesis. The software highlights options where there is a mismatch between the top-level and subproject settings. Pink highlighting indicates a mismatch that must be synchronized and yellow highlighting indicates that synchronization with the top level is optional. Resolve conflicting options before synthesizing the design. See Configuring Synthesis Runs for Hierarchical Projects, on page 178 in the <i>User Guide</i> for details.

Insert Subproject Command

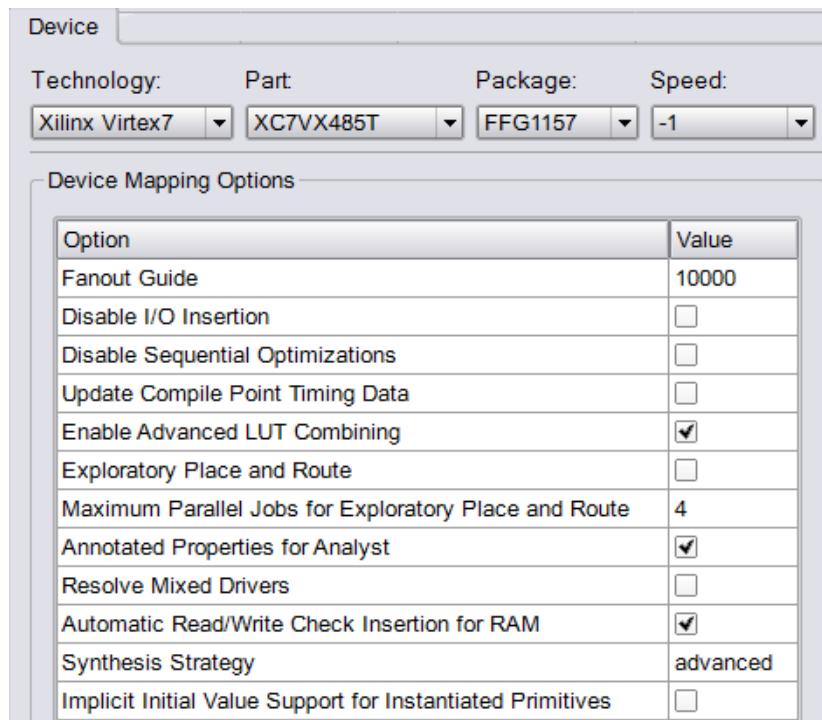
This command lets you nest subprojects within a subproject hierarchy. To do this, right-click on a subproject and select Insert SubProject from the popup menu. You can add an existing project or a new project from the Insert Project dialog box. Then, begin adding design files to your subproject after you have created the new project.



Implementation Options Command

You use the Implementation Options dialog box to define the implementation options for the current project. You can access this dialog box from Project->Implementation Options, by clicking the button in the Project view, or by clicking the text in the Project view that lists the current technology options.

This figure shows the Implementation Options dialog box for the Synplify Premier tool. Some of the tabs are not available in the Synplify Pro and Synplify tools.



This section describes the following:

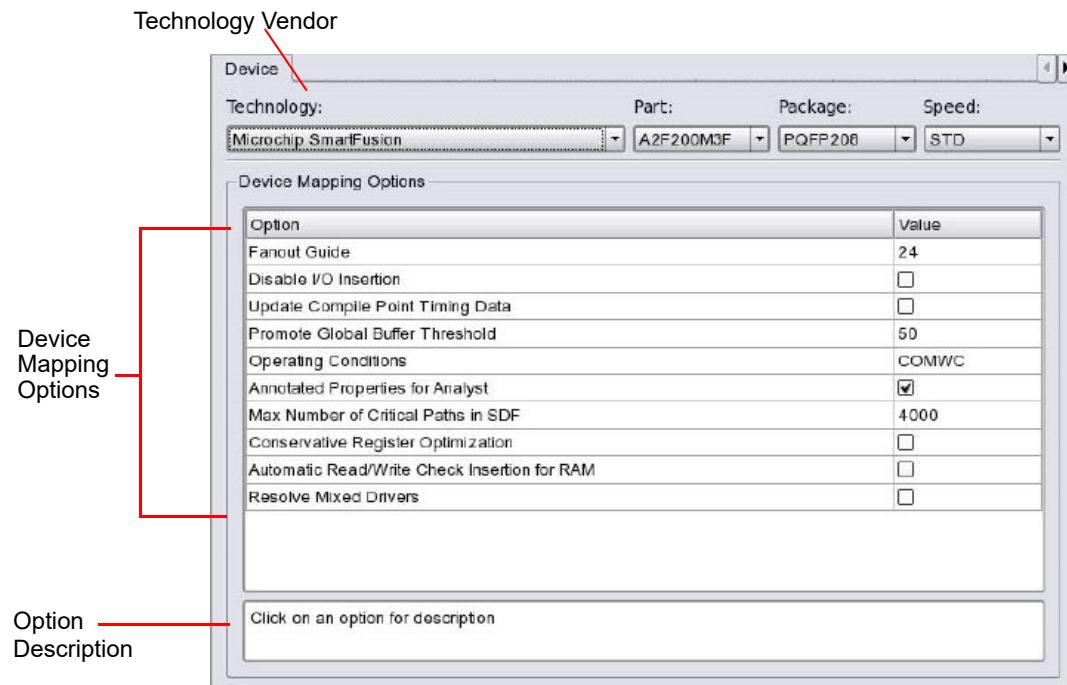
- [Device Panel](#), on page 445. For device-specific details of the options, refer to the appropriate vendor chapter.
- [Options Panel](#), on page 447
- [Constraints Panel](#), on page 450
- [Implementation Results Panel](#), on page 453

- [Timing Report Panel](#), on page 455
- [High Reliability Panel](#), on page 456
- [VHDL Panel](#), on page 458
- [Verilog Panel](#), on page 462
- [GCC Panel](#), on page 480
- [GCC & Prototyping Tools Panel](#), on page 481
- [Place and Route Panel](#), on page 483

For a description of the Design Planning panel, see [Design Planning Panel](#), on page 665.

Device Panel

You use the Device panel to set mapping options for the selected technology.



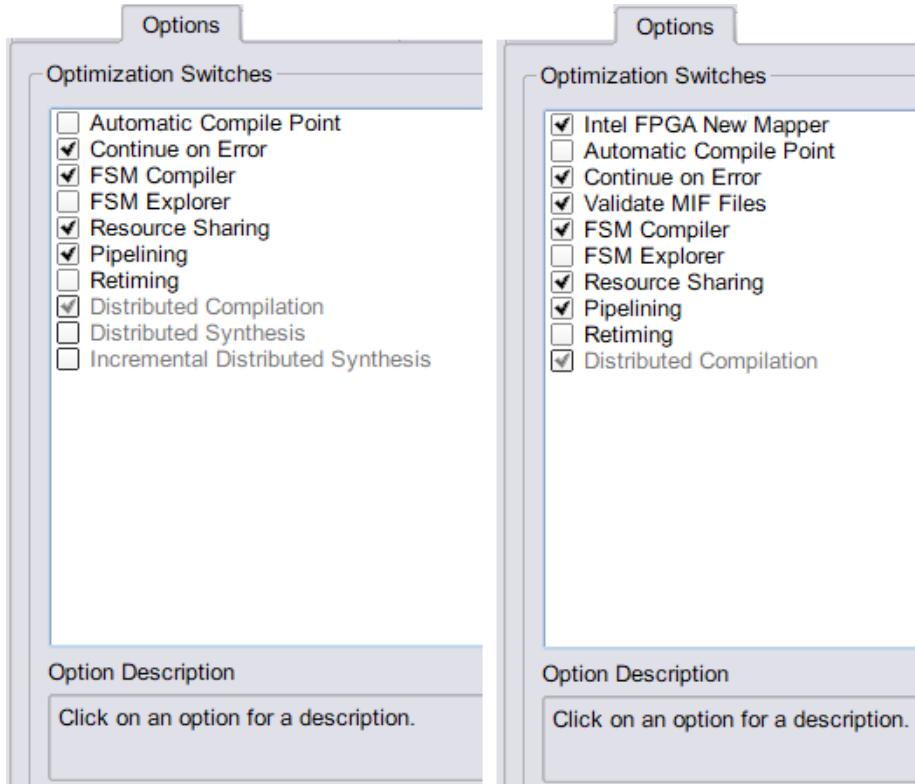
The mapping options vary, depending on the technology. See [Setting Device Options, on page 121](#) in the *User Guide* for a procedure, and the relevant vendor sections in this reference manual for technology-specific descriptions of the options.

The table below lists the following category of options. Not all options are available for all tools and technologies.

Option	Description
Technology Vendor	Specify the device technology you want to synthesize. You can also select the part, package, and speed grade to use. For more information, see the appropriate vendor appendix in the <i>Reference</i> manual.
Device Mapping Options	The device mapping options vary depending on the device technology you select. For more information, see the appropriate vendor appendix in the <i>Reference</i> manual.
Option Description	Click on a device mapping option to display its description in this field. Refer to the relevant vendor sections for technology-specific descriptions of the options.

Options Panel

You use the Options panel of the Implementation Options dialog box to define general options for synthesis optimization. See [Setting Optimization Options, on page 124](#) of the *User Guide* for details.



The following table lists the options alphabetically. Not all options are available for all tools and technologies.

Option	Description
Intel New Mapper	<p><i>Intel Arria 10, Arria V GZ, and Stratix V Technologies</i></p> <p>New version of the Intel mapper supports the latest Intel devices and provides performance and runtime improvements for the design. Both versions of the Intel mappers will coexist.</p> <p>Tcl equivalent: set_option -use_new_altera_mapper 0 1</p>
Auto Compile Point	<p><i>Synplify Pro, Synplify Premier</i></p> <p>Enables the automatic compile point flow, which can analyze a design and identify modules that can automatically be defined as compile points and mapped in parallel using multiprocessing.</p> <p>See The Automatic Compile Point Flow, on page 627 in the <i>User Guide</i>.</p> <p>Tcl equivalent: set_option -automatic_compile_point 0 1</p>
Continue on Error	<p><i>Synplify Pro, Synplify Premier</i></p> <p>When enabled for Synplify Premier logic synthesis, allows the compiler to continue after encountering a non-syntax error within a design unit and to resume compilation with the next design unit without stopping. During compile point synthesis, it allows synthesis to continue without erroring out when it encounters an error.</p> <p>When enabled for Synplify Pro synthesis, it only affects compile-point synthesis, allowing the operation to continue and synthesize other compile points.</p> <p>See Using Continue on Error, on page 320 in the <i>User Guide</i>.</p> <p>Tcl equivalent: set_option -continue_on_error 0 1</p>
Distributed Compilation	<p><i>Synplify Premier</i></p> <p>When enabled, the compiler splits the design into smaller sub-designs generating multiple netlists (srs) so that they can run in parallel to minimize runtime. The tool uses the Synopsys CDPL (Common Distributed Processing Library) scheme for distributed processing.</p> <p>See Using Distributed Processing, on page 808 in the <i>User Guide</i>.</p> <p>Tcl equivalent: set_option -distributed_compile 0 1</p>

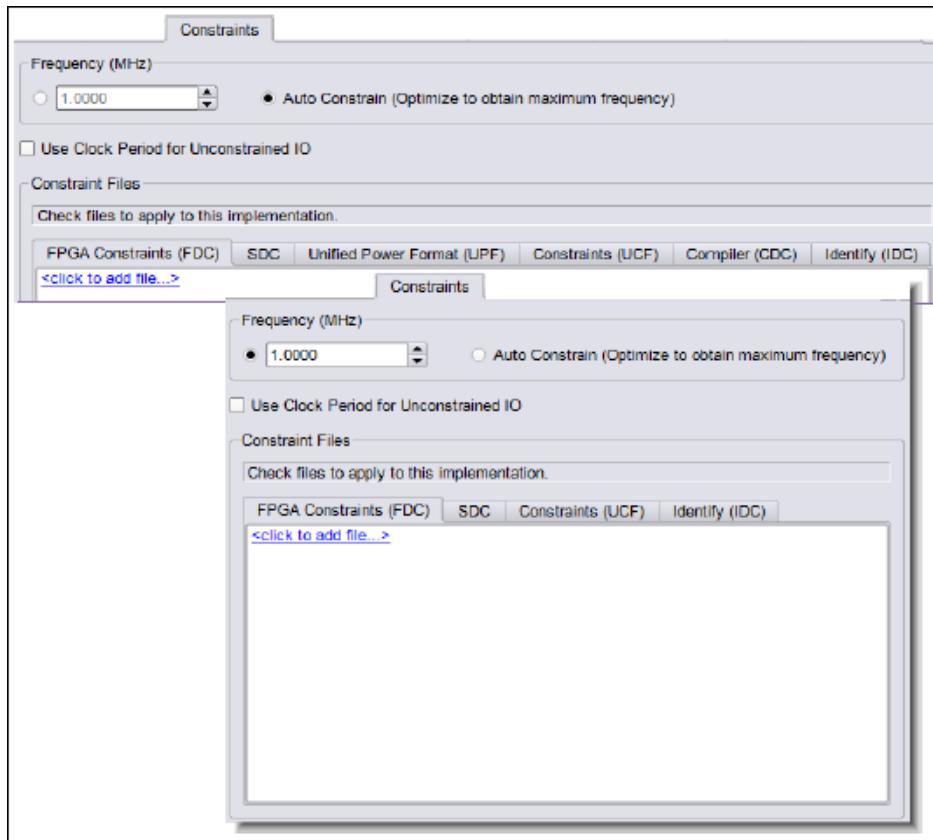
Option	Description
Distributed Synthesis	<p><i>Synplify Premier</i></p> <p>When enabled, distributed synthesis automatically splits large designs into smaller sub-designs for parallel processing on separate machines (or on the same machine) to improve runtime. The tool uses the Synopsys CDPL (Common Distributed Processing Library) scheme for distributed processing.</p> <p>See Using Distributed Processing, on page 808 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -distributed_synthesis 0 1</code></p>
Enable 64-bit Synthesis	<p><i>Synplify Pro, Synplify Premier</i></p> <p>Enables/disables the 64-bit mapping switch. When enabled, this switch allows you to run client programs in 64-bit mode, if available on your system.</p> <p>This option is supported on the Windows and Linux platforms.</p> <p>Tcl equivalent: <code>set_option -enable64bit 0 1</code></p>
FSM Compiler	<p>Determines whether the FSM Compiler is run. See Running the FSM Compiler, on page 585 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -symbolic_fsm_compiler 0 1</code></p>
FSM Explorer	<p><i>Synplify Pro, Synplify Premier</i></p> <p>Determines whether the FSM Explorer is run. See Running the FSM Explorer, on page 589 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -use_fsm_explorer 0 1</code></p>
Incremental Distributed Synthesis	<p><i>Synplify Premier</i></p> <p>When enabled, runs incremental distributed synthesis to improve runtime on subsequent runs, when only part of a design has changed. For details, see Running Incremental Distributed Synthesis, on page 819 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -incremental_synthesis 0 1</code></p>
Optimize NGC/EDIF	<p><i>Synplify Premier</i></p> <p>Enables NGC/EDIF optimization during fast synthesis strategy.</p> <p>Tcl equivalent: <code>set_option -optimize_ngc 0 1</code></p>
Pipelining	<p><i>Synplify Pro, Synplify Premier</i></p> <p>Runs designs at a faster frequency by moving registers after the multiplier or ROM into the multiplier or ROM. See Pipelining, on page 559 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -pipe 0 1</code></p>

Option	Description
Resource Sharing	<p>Determines whether you optimize area by sharing resources. When enabled, this optimization technique runs during the compilation stage of synthesis.</p> <p>Even if it is disabled, the mapper can still flatten the netlist and re-optimize adders, multipliers as needed to improve timing, because this setting does not affect the mapper. See Sharing Resources, on page 581 for information for how to use this option in the <i>User Guide</i>.</p> <p>Enabling this option generates the resource sharing report in the log file (see Resource Usage Report, on page 203).</p> <p>Tcl equivalent: set_option -resource_sharing 0 1</p>
Retiming	<p><i>Synplify Pro, Synplify Premier</i></p> <p>Determines whether the tool moves storage devices across computational elements to improve timing performance in sequential circuits. Note that the tool might retime registers associated with RAMs, DSPs, and generated clocks, regardless of the Retiming setting.</p> <p>See Retiming, on page 563 in the <i>User Guide</i>.</p> <p>Tcl equivalent: set_option -retiming 0 1</p>
Use Quartus Pro	<p><i>Intel Arria 10 Technology</i></p> <p>Specifies whether to use Intel Quartus Prime Pro when launching Quartus place and route.</p> <p>Tcl equivalent: set_option -use_quartus_pro 0 1</p>
Validate MIF Files	<p><i>Intel</i></p> <p>Checks that MIF files exist for associated Intel MegaWizard generated RAM or ROM instances. If MIF files are missing, the synthesis tool generates an error.</p> <p>For more information about handling RAM/ROM MIF files, see Intel FPGA MIF Files, on page 567.</p> <p>Tcl equivalent: set_option -validate_mif_files 0 1</p>

Constraints Panel

You use the Constraints panel of the Implementation Options dialog box to specify target frequency and timing constraint files for design synthesis. Depending on the synthesis tool you are using and the device you specify, the types of constraint files you can apply for the implementation may vary. See the table below for a complete list of option types you can apply.

See [Specifying Global Frequency and Constraint Files, on page 126](#), in the *User Guide* for details.



Option	Description
Frequency	Sets the default global frequency. You can either set the global frequency here or in the Project view. To override the default you set here, set individual clock constraints from the SCOPE interface. Tcl equivalent: <code>set_option -frequency frequency</code>

Option	Description
Auto Constrain	<p><i>Synplify Pro, Synplify Premier</i></p> <p>When enabled and no clocks are defined, the software automatically constrains the design to achieve the best possible timing. It does this by reducing periods of each individual clock and the timing of any timed I/O paths in successive steps. See Using Auto Constraints, on page 492 in the <i>User Guide</i> for information about using this option.</p> <p>You can also set this option in the Project view.</p> <p>Tcl equivalent: set_option -frequency auto</p>
Use clock period for unconstrained IO	<p>Determines whether default constraints are used for I/O ports that do not have user-defined constraints.</p> <p>When disabled, only <code>set_input_delay</code> or <code>set_output_delay</code> constraints are considered during synthesis or forward-annotated after synthesis.</p> <p>When enabled, the software considers any explicit <code>set_input_delay</code> or <code>set_output_delay</code> constraints. In addition, for all ports without explicit constraints, it uses constraints based on the clock period of the attached registers. Both the explicit and implicit constraints are used for synthesis and forward-annotation. The default is off (disabled) for new designs.</p> <p>Tcl equivalent: set_option -auto_constraint_io 0 1</p>
Constraint Files FDC	<p>Specifies which timing constraints (FDC) files to use for the implementation. Select the check box to choose a file.</p> <p><i>Synplify Pro, Synplify Premier</i></p> <p>For block-level files in the compile-point flows, the Module column shows the name of the module or compile point.</p>
Constraint Files SDC	<p>Specifies which timing constraints (SDC) files to use for the implementation. Select the check box to choose a file.</p> <p><i>Synplify Pro, Synplify Premier</i></p> <p>For block-level files in the compile-point flows, the Module column shows the name of the module or compile point.</p>
UPF	<p><i>Synplify Premier</i></p> <p>Specifies the unified power format (UPF) files to use for the implementation. Select the check box to choose a file.</p>
UCF	<p><i>Xilinx</i></p> <p>Specifies which user constraints (UCF) files to use for the implementation. Select the check box to choose a file.</p>

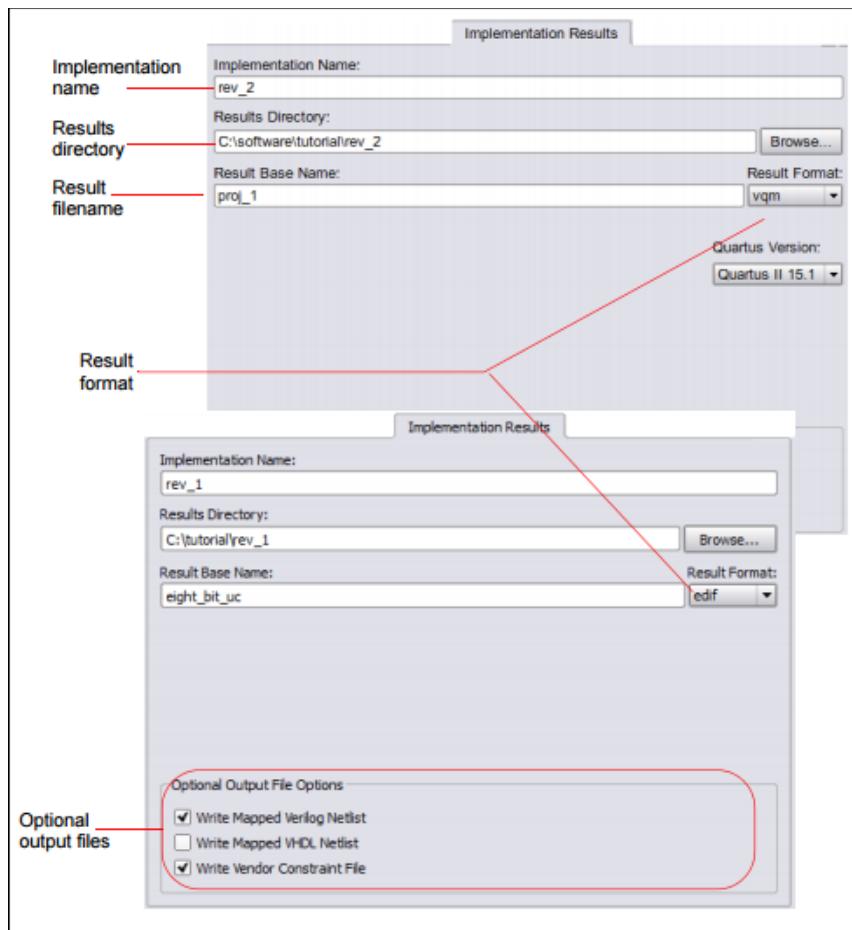
Option	Description
CDC	<i>Synplify Premier</i> Specifies which compiler directives (CDC) files to use for the implementation. Enable the check box to select a file.
Identify (IDC)	Specifies the instrumentation design constraints (IDC) files that add compiler pragmas in these files to the design RTL for the instrumented signals and break points. Enable the check box to select a file.

Implementation Results Panel

You use the Implementation Results panel to specify the implementation name (default: rev_1), the results directory, and the name and format of the top-level output netlist file (Result File). You can also specify output constraint and netlist files. See [Specifying Result Options, on page 129](#) of the *User Guide* for details.

The results directory is a subdirectory of the project file directory. Clicking the Browse button brings up the Select Run Directory dialog box to allow you to browse for the results directory. You can change the location of the results directory, but its name must be identical to the implementation name.

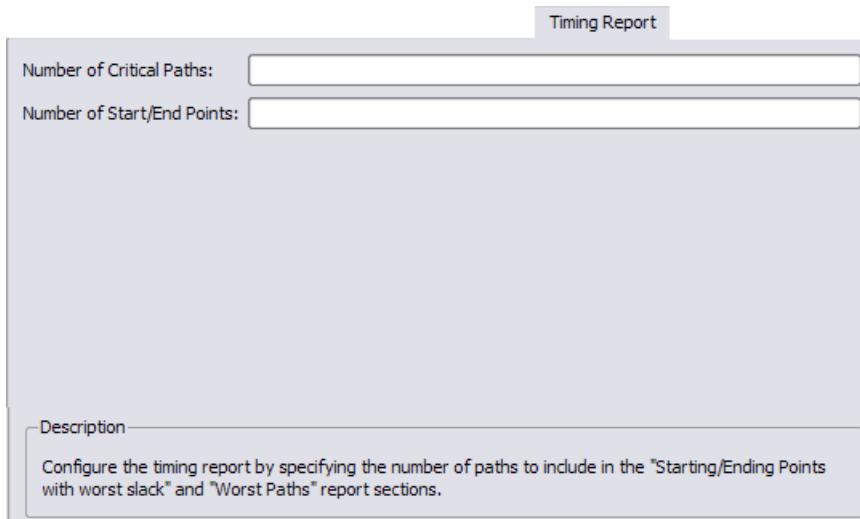
Select optional output file check boxes to generate the corresponding Verilog netlist, VHDL netlist, or vendor constraint files.



Option	Description
Implementation Name	Displays implementation name, directory path for results, and the base name for the result files.
Results Directory	Tcl equivalent: set_option -result_file pathToFile
Result Base Name	
Result Format	Select the output that corresponds to the technology you are using. See the <i>Appendices</i> of the <i>Reference</i> manual for a vendor-specific in the <i>User Guidelist</i> of netlist formats. Tcl equivalent: set_option -result_format format
Quartus Version	For certain Intel technologies, select the version of Quartus place-and-route tool for the Intel library files you are using to compile your design. Tcl equivalent: set_option -quartus_version version
Use Vivado for Place and Route	<i>Xilinx 7 Series (Virtex-7, Kintex-7, and Artix-7) and Newer Technologies</i> It is recommended that you use the Vivado Integrated Design environment (IDE) tool to run place and route for these devices. You must enable this option. For more information, see Running Xilinx Vivado Place and Route, on page 1117 . Tcl equivalent: set_option -use_vivado 0 1
Write Mapped Verilog Netlist	Generates mapped Verilog or VHDL netlist files. Tcl equivalent: set_option -write_verilog 0 1
Write Mapped VHDL Netlist	Tcl equivalent: set_option -write_vhdl 0 1
Write Vendor Constraint File	Generates a vendor-specific constraint file for forward annotation. Tcl equivalent: set_option -write_apr_constraint 0 1

Timing Report Panel

Use the Timing Report panel (Implementation Options dialog box) to set criteria for the (default) output timing report. Specify the number of critical paths and the number of start and end points to appear in the timing report. See [Specifying Timing Report Output, on page 130](#) in the *User Guide* for details. For a description of the report, see [Timing Reports, on page 205](#).



Option	Description
Number of Critical Paths	Set the number of critical paths for the software to report. Tcl equivalent: set_option -num_critical_paths numberOfRowsPaths
Number of Start/End Points	Specify the number of start and end points to see reported in the critical path sections. Tcl equivalent: set_option -num_startend_points numberOfRowsPoints

See also:

- [Timing Reports](#), on page 205, for more information on the default timing report, which is affected by the Timing Report panel settings.
- [Analysis Menu](#), on page 519, information on creating additional custom timing reports for certain device technologies.

High Reliability Panel

Use the High Reliability panel (Implementation Options dialog box) to implement safe logic for the design. For more information about high reliability support, see *Handling High Reliability Designs* in the *User Guide*.

High Reliability

Disable Verification Mode to set High Reliability options

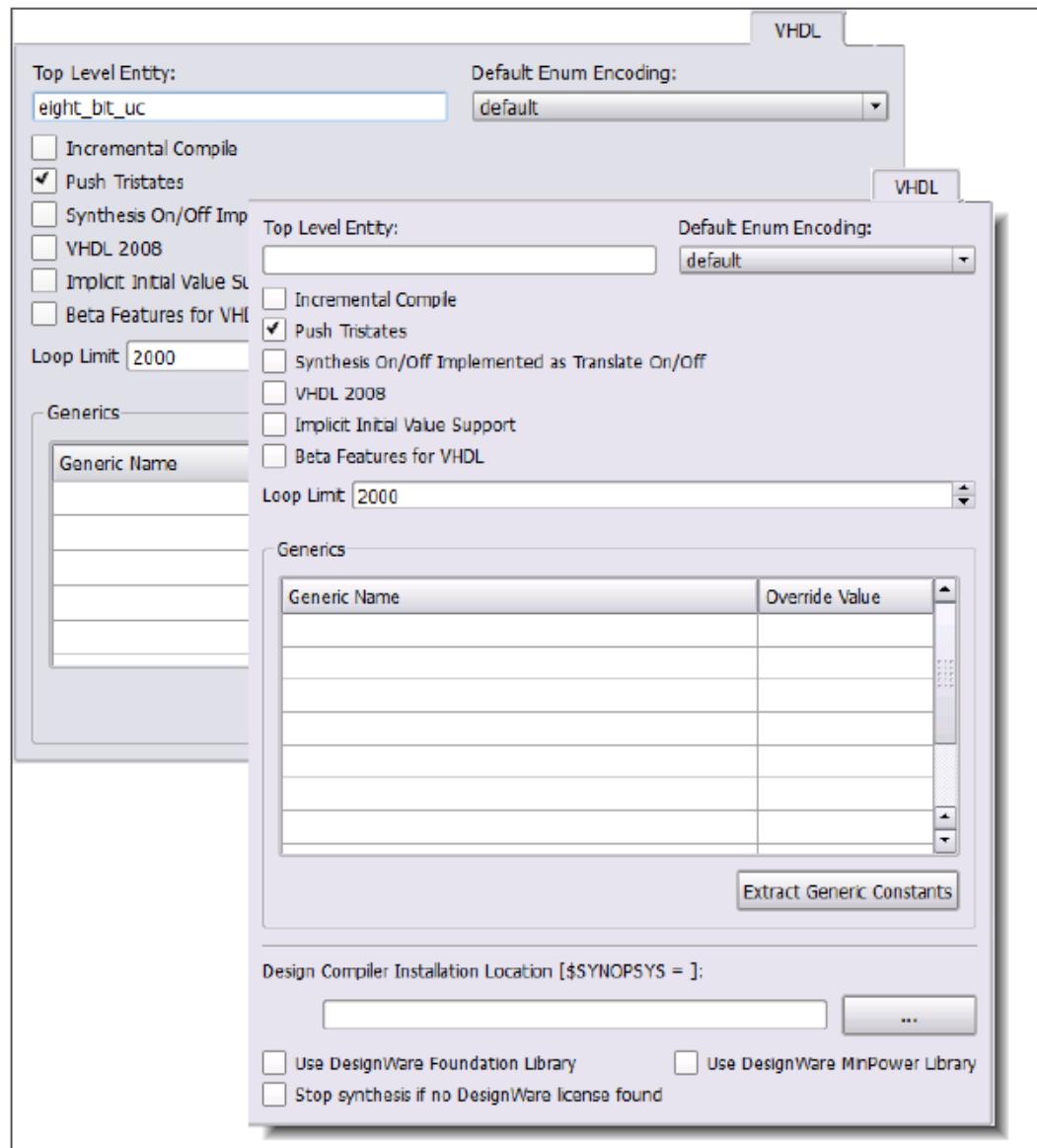
- Preserve and Decode Unreachable States - (FSM, Counters, Sequential Logic)
- FSM Error Correction Using Hamming Distance 3**
- FSM SEC-DED Using Hamming Distance 3
- FSM SEC-DED and Recovery Using Hamming Distance 3
- Gate Level Netlist TMR

Option	Description
Preserve and Decode Unreachable States (FSM, Counters, Sequential Logic)	<p><i>Synplify Pro, Synplify Premier</i></p> <p>When enabled, this option turns off sequential optimizations on all counters, FSMs, and sequential logic, to increase the reliability of the circuit.</p> <p>If you do not want to implement this globally, use the <code>syn_safe_case</code> directive (syn_safe_case, on page 599) on individual FSMs.</p> <p>Tcl equivalent: <code>set_option -safe_case 0 1</code></p>
FSM Error Correction Using Hamming Distance 3	<p><i>Synplify Premier</i></p> <p>When enabled, a Hamming 3 encoding is used into build single-bit error correction logic in the FSM state registers.</p> <p>Tcl equivalent: <code>set_option -hamming3 0 1</code></p>

Option	Description
FSM SEC-DED Using Hamming Distance 3	When enabled in conjunction with the error monitoring TCL commands (<code>syn_create_err_net -double_bit/syn_connect</code>), a Hamming3 encoding is used to build single-bit error correction logic along with double-bit error detection in the FSM state registers. Tcl equivalent: <code>set_option -hamming3_ded 0 1</code>
FSM SEC-DED and Recovery Using Hamming Distance 3	When enabled, a Hamming3 encoding is used to build single-bit error correction logic and default state recovery for double-bit error detection in the FSM state registers. Tcl equivalent: <code>set_option -hamming3_ded_recovery 0 1</code>
Gate Level Netlist TMR	Enable the Gate Level Netlist TMR option if the module specified for Distributed TMR contains technology primitives instantiated either in the RTL or through pre-synthesized EDIF or VM netlists.

VHDL Panel

You use the VHDL panel in the Implementation Options dialog box to specify various language-related options. With mixed HDL designs, the VHDL and Verilog panels are both available. See [Setting Verilog and VHDL Options, on page 131](#), of the *User Guide* for details.



The following table describes the options available.

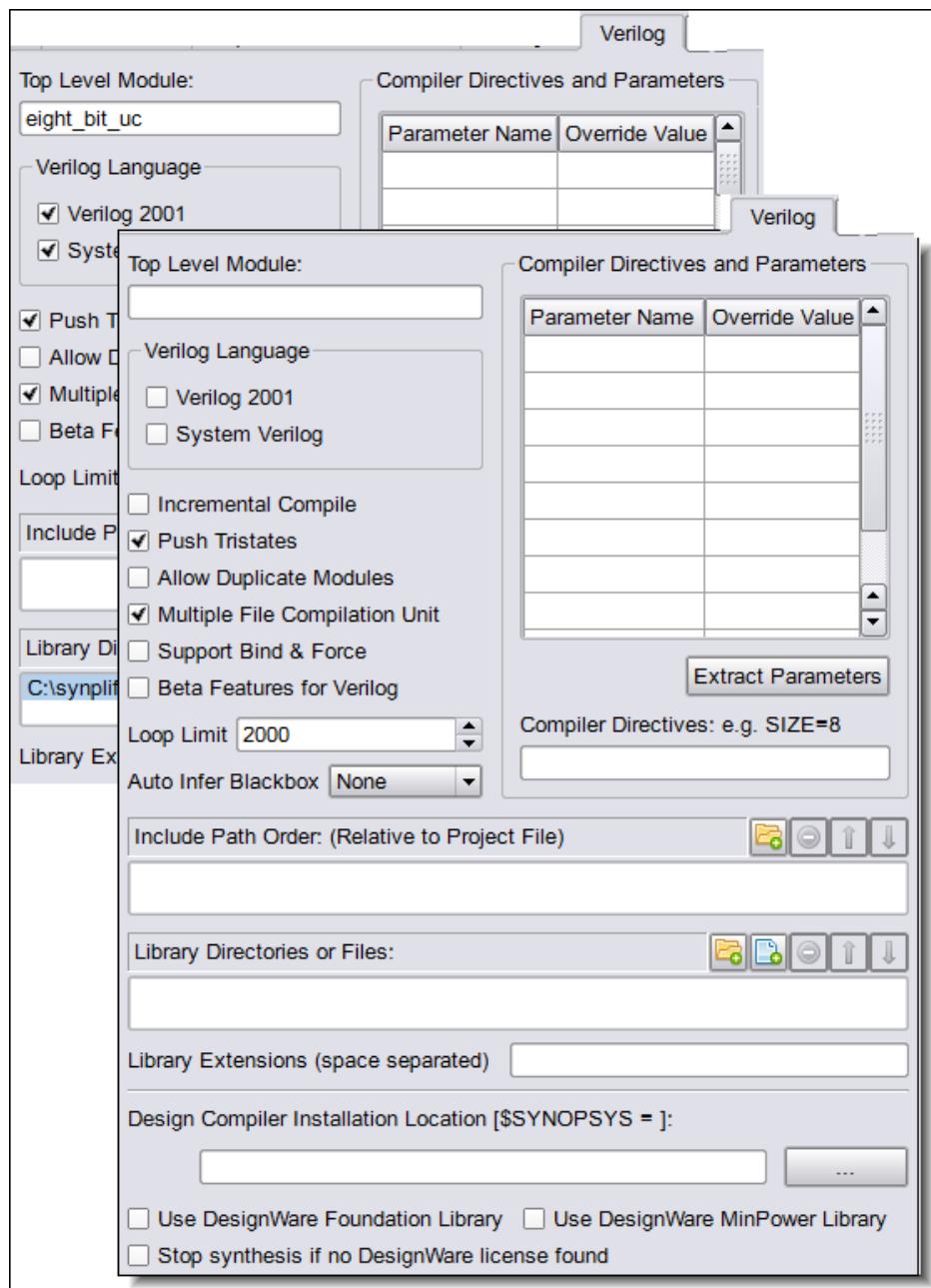
Feature	Description
Top Level Entity	<p>The name of the top-level entity of your design. If the top-level entity does not use the default work library to compile the VHDL files, you must specify the library file where the top-level entity can be found. To do this, the top-level entity name must be preceded by the VHDL library followed by a period (.). To specify VHDL library files, see Project Menu , on page 423 for the Set VHDL Library command, or the File Options Popup Menu Command , on page 622.</p> <p>Tcl equivalent: <code>set_option -top_module topLevelName</code></p>
Default Enum Encoding	<p>The default enumeration encoding to use. This is only for enumerated types; the FSM compiler automatically determines the state-machine encoding, or you can specify the encoding using the syn_encoding attribute.</p> <p>Tcl equivalent: <code>set_option -default_enum_encoding encodingType</code></p> <p>Note: Enable the FSM Compiler before attempting to change the encoding style.</p>
Incremental Compile	<p><i>Synplify Premier</i></p> <p>When enabled, lets you run the incremental compiler to reduce compiler runtime for large designs. The software recompiles only relevant .srs files when a design change is made and reuses the compiler database. For details, see Using the Incremental Compiler, on page 86 in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -incremental 0 1</code></p>
Push Tristates	<p>When enabled (default), tristates are pushed across process/block boundaries. For more information, see Push Tristates Option , on page 468.</p> <p>Tcl equivalent: <code>set_option -compiler_compatible 0 1</code></p>
Synthesis On/Off Implemented as Translate On/Off	<p>When enabled, the software ignores any VHDL code between synthesis_on and synthesis_off directives. It treats these third-party directives like translate_on/translate_off directives (see translate_off/translate_on , on page 749 for details).</p> <p>Tcl equivalent: <code>set_option -synthesis_onoff_pragma 0 1</code></p>

Feature	Description
VHDL 2008	When enabled, allows you to use VHDL 2008 language standards. Tcl equivalent: set_option -vhdl2008 0 1
Implicit Initial Value Support	When enabled, the compiler passes init values through a syn_init property to the mapper. For more information, see VHDL Implicit Data-type Defaults, on page 334 . Tcl equivalent: set_option -supporttypedflt 0 1
Beta Features for VHDL	Enables use of any VHDL beta features included in the release. Enabling this checkbox is equivalent to including a set_option -hdl_define -set _BETA_FEATURES_ON_ directive in the project file. Tcl equivalent: set_option -beta_vhfeatures 0 1
Loop Limit	Overrides the default compiler loop limit value of 2000 in the RTL and sets a new global default. You can apply limits on a per-loop basis using the Verilog loop_limit or the VHDL syn_looplmit directive for individual loops. For details about these directives, see loop_limit, on page 79 and syn_looplmit, on page 373 in the Attribute Reference. Tcl equivalent: set_option -looplmit loopLimitValue
Generics	Shows generics extracted with Extract Generic Constants. You can override the default and set a new value for the generic constant. The value is valid for the current implementation.
Extract Generic Constants	Extracts generics from the top-level entity and displays them in the table.
Design Compiler Installation Location [\$SYNOPSYS=:]	<i>Synplify Premier</i> Specifies the path to the Synopsys DesignWare foundation library building blocks. Do not use this field if the \$SYNOPSYS environment variable is defined. Tcl equivalent: set_option -dc_root pathToDesignWare
Use DesignWare Foundation Library	<i>Synplify Premier</i> Indicates that Synopsys DesignWare foundation library building blocks are to be used in the design. Tcl equivalent: set_option -dw.foundation 0 1

Feature	Description
Use DesignWare MinPower Library	<p><i>Synplify Premier</i></p> <p>Indicates that a corresponding building block from the Synopsys minPower library is to be used in place of its equivalent DesignWare foundation library building block. Selecting this option automatically selects the Use DesignWare Foundation Library checkbox.</p> <p>Tcl equivalent: set_option -dw_minpower 0 1</p>
Stop synthesis if no DesignWare license is found	<p><i>Synplify Premier</i></p> <p>Specifies how to treat a DesignWare building block encountered in a design when no DesignWare license is available (DesignWare support is licensed separately). If checked, synthesis is stopped with an error; if unchecked (the default), the DesignWare building block is treated as a black box and synthesis is allowed to continue.</p> <p>Tcl equivalent: set_option -dw_stop_on_nolic 0 1</p>

Verilog Panel

You use the Verilog panel in the Implementation Options dialog box to specify various language-related options. With mixed HDL designs, the VHDL and Verilog panels are both available. See [Setting Verilog and VHDL Options, on page 131](#) of the *User Guide* for details.



Feature	Description
Top Level Module	The name of the top-level module of your design. Tcl equivalent: set_option -top_module moduleName
Compiler Directives and Parameters	Shows design parameters extracted with Extract Parameters. You can override the default and set a new value for the parameter. The value is valid for the current implementation.
Extract Parameters	Extracts design parameters from the top-level module and displays them in the table. See Compiler Directives and Design Parameters , on page 470 .
Compiler Directives	Provides an interface where you can enter compiler directives that you would normally enter in the code with 'ifdef and 'define statements. See Compiler Directives and Design Parameters , on page 470 .
Verilog Language - Verilog 2001	When enabled, the default Verilog standard for the project is Verilog 2001. When both Verilog 2001 and SystemVerilog are disabled, the default standard is Verilog 95. For information about Verilog 2001, see Verilog 2001 Support, on page 39 . You can override the default project standard on a per file basis by selecting the file, right-clicking, and selecting the File Options command (see File Options Popup Menu Command , on page 622). Tcl equivalent: set_option -vlog_std v2001
Verilog Language - SystemVerilog	When enabled, the default Verilog standard for the project is SystemVerilog which is the default standard for all new projects. Enabling SystemVerilog automatically enables Verilog 2001. Tcl equivalent: set_option -vlog_std sysv
Incremental Compile	<i>Synplify Premier</i> When enabled, lets you run the incremental compiler to reduce compiler runtime for large designs. The software recompiles only relevant srs files when a design change is made and reuses the compiler database. For details, see Using the Incremental Compiler, on page 86 in the <i>User Guide</i> . Tcl equivalent: set_option -incremental 0 1

Feature	Description
Push Tristates	<p>When enabled (default), tristates are pushed across process/block boundaries. For details, see Push Tristates Option , on page 468.</p> <p>Tcl equivalent: set_option -compiler_compatible 0 1</p>
Allow Duplicate Modules CRM 9001291474	<p>Allows the use of duplicate modules in your design. When enabled, the last definition of the module is used by the software and any previous definitions are ignored.</p> <p>You should not use duplicate module names in your Verilog design, therefore this option is disabled by default. However, if you need to, you can allow for duplicate modules by enabling this option.</p> <p>Tcl equivalent: set_option -allow_duplicate_modules 0 1</p>
Multiple File Compilation Unit	<p>When enabled (the default), the Verilog compiler uses the compilation unit for modules defined in multiple files.</p> <p>See SystemVerilog Compilation Units, on page 233 for additional information.</p> <p>Tcl equivalent: set_option -multi_file_compilation_unit 0 1</p>
Support Bind & Force	<p><i>Synplify Premier</i></p> <p>When enabled, bind can be used to insert an instance into a design hierarchy. The force function allows you to override existing drivers of internal signals in the design hierarchy with new drivers. (See Bind Function, on page 31 and Force Function, on page 34.)</p> <p>Tcl equivalent: set_option -bindandforce 0 1</p>
Beta Features for Verilog	<p>Enables use of any Verilog beta features included in the release. Enabling this checkbox is equivalent to including a <code>set_option -hdl_define -set _BETA_FEATURES_ON_</code> directive in the project file.</p> <p>Tcl equivalent: set_option -beta_vfeatures 0 1</p>

Feature	Description
Auto Infer Blackbox	<p><i>Synplify Premier</i></p> <p>Selects how undefined modules are treated in the tool.</p> <ul style="list-style-type: none"> • Selecting None (the default) causes the compiler to error out when an undefined module is encountered. • Selecting With BIDIR ports creates a black box with bi-directional ports for the undefined module and generates a warning message, but allows the operation to continue; this mode matches the behavior of Certify. • Selecting BIDIR/IN ports creates a black box with either input or bi-directional ports for the undefined module, and generates a warning message, but allows the operation to continue. <p>Tcl equivalent: set_option -auto_infer_blackbox 0 1 2</p>
Loop Limit	<p>Overrides the default compiler loop limit value of 2000 in the RTL and sets a new global default. You can apply limits on a per-loop basis using the Verilog <code>loop_limit</code> or the VHDL <code>syn_looplimate</code> directive for individual loops.</p> <p>For details about these directives, see loop_limit, on page 79 and syn_looplimate, on page 373 in the <i>Attribute Reference</i>.</p> <p>Tcl equivalent: set_option -loopleftop limit loopLimitValue</p>
Include Path Order (Relative to Project File)	<p>Specifies the search paths for the include commands in the Verilog design files of your project. Use the buttons in the upper right corner of the box to add, delete, or reorder the paths. The include paths are relative. See Updating Verilog Include Paths in Older Project Files, on page 111 in the <i>User Guide</i> for additional information.</p>
Library Directories or Files	<p>Specifies all the paths to the directories which contain the Verilog library files to be included in your design for the project. You can also add custom library files with module definitions for the design in a single file. See Verilog Single Library File Support , on page 469. The names of files read from the library path must match module names. Mismatches result in error messages.</p> <p>Tcl equivalent:</p> <p style="padding-left: 20px;">set_option -library_path . /libraryPath or libraryFile</p>

Feature	Description
Library Extensions (space separated)	<p>Adds library extensions to Verilog library files included in your design for the project and searches the directory paths you specified that contain these Verilog library files. To use library extensions, see Using Library Extensions for Verilog Library Files, on page 78 in the <i>User Guide</i>.</p> <p>Tcl equivalent: set_option -libext .libextName</p> <p>Enter a space between each unique library extension.</p>
Design Compiler Installation Location [\$SYNOPSYS=:]	<p><i>Synplify Premier</i></p> <p>Specifies the path to the Synopsys DesignWare foundation library building blocks. Do not use this field if the \$SYNOPSYS environment variable is defined.</p> <p>Tcl equivalent: set_option -dc_root pathToDesignWare</p>
Use DesignWare Foundation Library	<p><i>Synplify Premier</i></p> <p>Indicates that Synopsys DesignWare foundation library building blocks are to be used in the design.</p> <p>Tcl equivalent: set_option -dw.foundation 0 1</p>
Use DesignWare MinPower Library	<p><i>Synplify Premier</i></p> <p>Indicates that a corresponding building block from the Synopsys minPower library is to be used in place of its equivalent DesignWare foundation library building block. Selecting this option automatically selects the Use DesignWare Foundation Library checkbox.</p> <p>Tcl equivalent: set_option -dw_minpower 0 1</p>
Stop synthesis if no DesignWare license is found	<p><i>Synplify Premier</i></p> <p>Specifies how to treat a DesignWare building block encountered in a design when no DesignWare license is available (DesignWare support is licensed separately). If checked, synthesis is stopped with an error; if unchecked (the default), the DesignWare building block is treated as a black box and synthesis is allowed to continue.</p> <p>Tcl equivalent: set_option -dw_stop_on_nolic 0 1</p>

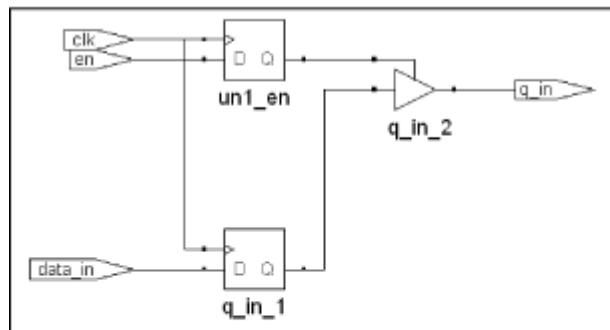
Push Tristates Option

Pushing tristates is a synthesis optimization option you set with Project->Implementation Options->Verilog or VHDL.

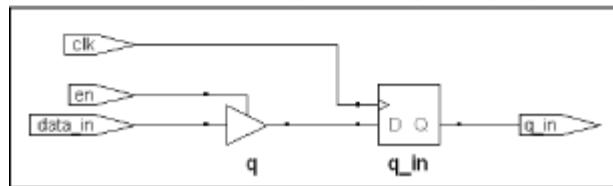
Description

When the Push Tristates option is enabled, the Synopsys FPGA compiler pushes tristates through objects such as muxes, registers, latches, buffers, nets, and tristate buffers, and propagates the high-impedance state. The high-impedance states are not pushed through combinational gates such as ANDs or ORs.

Push Tristates on: tristate is pushed through the flip-flop so that the result matches RTL simulation



Push Tristates off: tristate is not pushed through the flip-flop



If there are multiple tristates, the software muxes them into one tristate and pushes it through. The software pushes tristates through loops and stores the high impedance across multiple cycles in the register.

Advantages and Disadvantages

The advantage to pushing tristates to the periphery of the design is that you get better timing results because the software uses tristate output buffers.

The Synopsys FPGA software approach to tristate inference matches the simulation approach. Simulation languages are defined to store and propagate 0, 1, and Z (high impedance) states. Like the simulation tools, the Synopsys FPGA synthesis tool propagates the high-impedance states instead of producing tristate drivers at the outputs of process (VHDL) or always (Verilog) blocks.

The disadvantage to pushing tristates is that you might use more design resources.

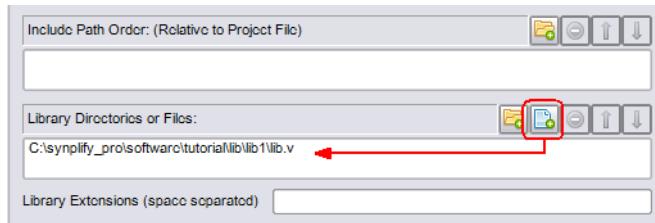
Effect on Other Synthesis Options

Tristate pushing has no effect on the `syn_tristatetomux` attribute. This is because tristate pushing is a compiler directive, while the `syn_tristatetomux` attribute is used during mapping.

Verilog Single Library File Support

You can add a single library file to your project for easier migration from a VCS environment and to ensure their behaviors are consistent for the design. To do this, either:

- Select the Add a file icon from the Verilog tab of the Implementation Options panel. Then, specify the library file to be added to the project from the Library Directories and Files option.



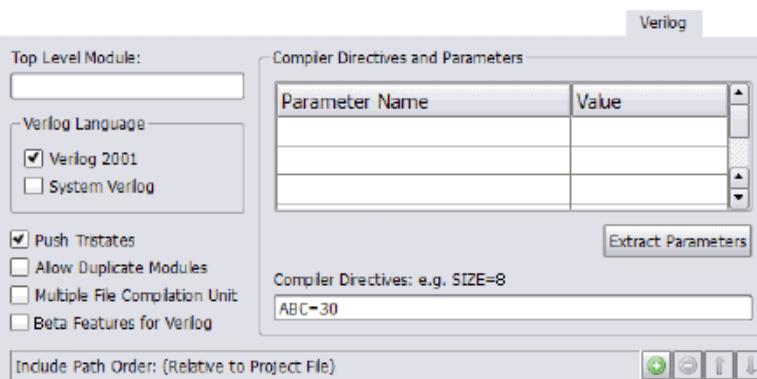
- Add the following Tcl command to your project file:

```
set_option -library_path {./libPath/libFile.v}
```

Compiler Directives and Design Parameters

When you click the Extract Parameters button in the Verilog panel (Implementation Options dialog box), parameter values from the top-level module are displayed in the table. You can also override the default by setting a new value for the parameter. The value is valid for the current implementation only.

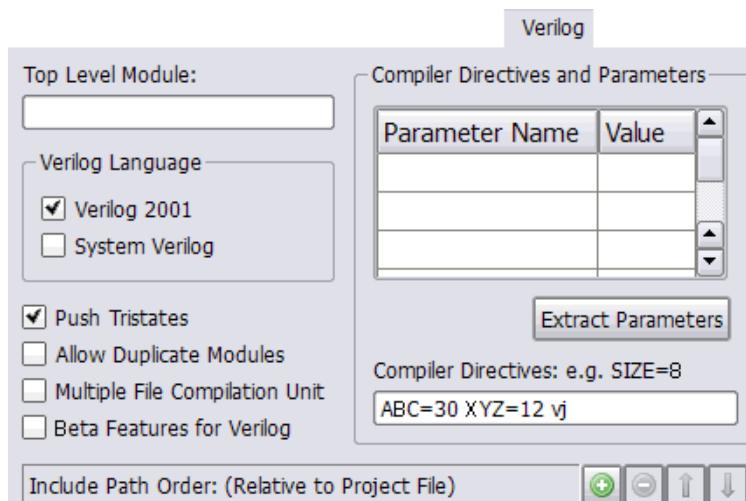
The Compiler Directives field provides an interface where you can enter compiler directives that you would normally enter in the code using 'ifdef' and 'define' statements. Use spaces to separate the statements. The directives you enter are stored in the project file. For example, if you enter the directive shown below to the Compiler Directives field of the Verilog panel:



The software writes the following statement to the project file:

```
set_option -hdl_define -set "ABC=30"
```

To define multiple variables in the GUI, use a space delimiter. For example:



The software writes the following statement to the prj file:

```
set_option hdl_define -set "ABC=30 XYZ=12 vj"
```

More information is provided for the following Verilog compiler directives:

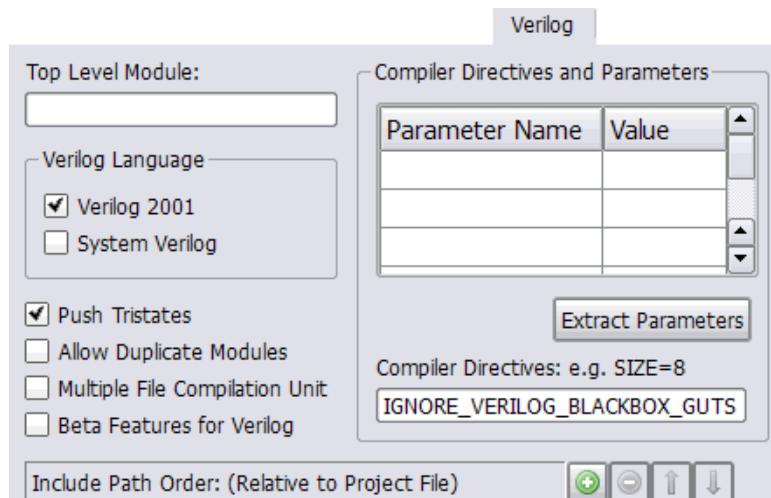
Compiler Directive	Description
<code>_ALLOWNESTEDBLOCKCOMMENTSTART_</code>	Allows for nested comment blocks.
<code>_BETA_FEATURES_ON_</code>	Explicitly enables beta HDL language features.
<code>IGNORE_VERILOG_BLACKBOX_GUTS</code>	Ignores the contents of a black box.
<code>_SEARCHFILENAMEONLY_</code>	Provides workarounds for archive utility limitations.
<code>SYN_COMPATIBLE</code>	Ensure compatibility between different Synopsys tools.
<code>_SYN_COMPATIBLE_INCLUDEPATH_</code>	Specifies that the search path order for includes to be the same as the one used by the simulation tool (VCS).
<code>_SYN_STRICT_MODPORTS_</code>	Requires that modports defined strictly access the associated interface ports specified in the instantiation.

IGNORE_VERILOG_BLACKBOX_GUTS

When you use the `syn_black_box` directive, the compiler parses the contents of the black box and can determine whether illegal syntax or incorrect code is defined within it. Whenever this occurs, an error message is generated. You can specify the `IGNORE_VERILOG_BLACKBOX_GUTS` compiler directive to ignore the contents of the black box. However, make sure that the black box is syntactically correct.

If you want the tool to ignore the contents of your black box, set the:

- Built-in compiler directive `IGNORE_VERILOG_BLACKBOX_GUTS` in the Compiler Directives field of the Verilog panel on the Implementation Options dialog box.



The software writes the following command to the project file:

- ```
set_option -hdl_define -set "IGNORE_VERILOG_BLACKBOX_GUTS"
• `define IGNORE_VERILOG_BLACKBOX_GUTS directive in the Verilog file.
```

This option is implemented globally for the project file.

## Example of the IGNORE\_VERILOG\_BLACKBOX\_GUTS Directive

Note that the IGNORE\_VERILOG\_BLACKBOX\_GUTS directive ignores the contents of the black box. However, whenever you use this directive, you must first define the ports for the black box correctly. Otherwise, the IGNORE\_VERILOG\_BLACKBOX\_GUTS directive generates an error. See the following valid Verilog example:

```
`define IGNORE_VERILOG_BLACKBOX_GUTS
module b1_fpgal (A,B,C,D) /* synthesis syn_black_box */;
 input B;
 output A;
 input [2:0] D;
 output [2:0] C;
 temp;
 assign A = B;
 assign C = D;

endmodule

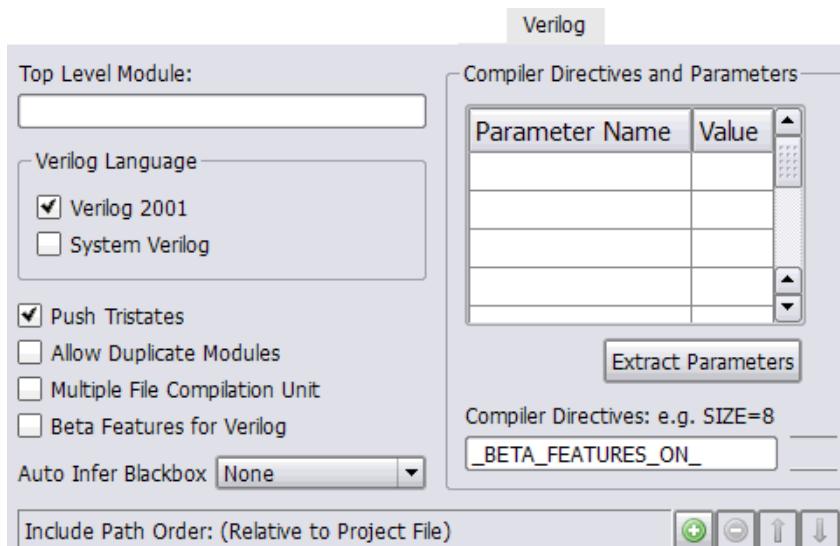
module b1_fpgal_top (inout A, B, inout [2:0] C, D);
 b1_fpgal b1_fpgal_inst(A,B,C,D);
endmodule
```

## BETA FEATURES ON

Beta features for the Verilog, SystemVerilog, or VHDL language must be explicitly enabled. In the UI, a Beta Features checkbox is included on the VHDL or Verilog tab of the Implementations Options dialog box. A \_BETA\_FEATURES\_ON\_ compiler directive is also available. This directive is specified with a set\_option -hdl\_define command added to the project file as shown below:

```
set_option -hdl_define -set _BETA_FEATURES_ON_
```

The directive can also be added to the Compiler Directives field of the Verilog panel.



Current beta features that must be explicitly enabled for the compiler include the following:

#### HDL Language Constructs      Descriptions

- 
- |            |                                                                                                                                               |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Aggregates | <ul style="list-style-type: none"> <li>Multi-assignments for array aggregates</li> <li>Assignments for the union of variable types</li> </ul> |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
- 

### SEARCHFILENAMEONLY

This directive provides a workaround for some known limitations of the archive utility.

If Verilog 'include files belong in any of the following categories, you may encounter problems when compiling a design after un-archiving:

1. The include paths have relative paths to the project file.

```
`include "../../defines.h"
```

2. The include paths have absolute paths to the project file.

```
`include "c:/temp/params.h"
```

```
`include "/remote/sbg_home/user/params.h"
```

3. The include paths have the same file names, but are located in different directories relative to the project file.

```
`include "../myflop.v"
```

```
...
`include "../../myflop.v"
```

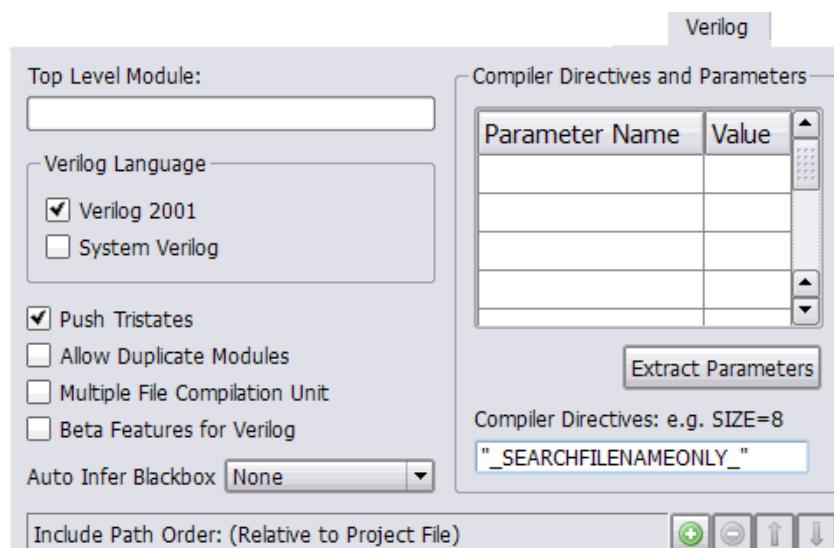
Use the `_SEARCHFILENAMEONLY_` directive to resolve categories 1 and 2 above. Category 3 is a known limitation; in this case it is recommended that you adopt standard coding practices to avoid files with the same name and different content.

When you un-archive a `sar` file that contains relative or absolute include paths for the files in the project, you can add the `_SEARCHFILENAMEONLY_` compiler directive to the unarchived project. This has the compiler remove the relative/absolute paths from the `'include` and search only for the file names.

This directive is specified with a `set_option -hdl_define` command added to an implementation within the project file as shown below:

```
set_option -hdl_define -set "__SEARCHFILENAMEONLY__"
```

The directive can also be added to the Compiler Directives field of the Verilog panel as shown below.



The compiler generates the following warning message whenever it extracts include files using this directive:

```
@W: | Macro _SEARCHFILENAMEONLY_ is set: fileName not found
attempting to search for base file name fileName
```

## \_\_ALLOWNESTEDBLOCKCOMMENTSTART\_\_

Verilog/SystemVerilog comments can be included in RTL code as a

- One-line comment starting with the characters // and ending with a new line
- Block comment starting with /\* and ending with \*/

However, nested block comments are not supported according to the LRM, so most tools generate a warning and ignore these comments in the RTL code. To match this behavior, the synthesis tools must also ignore various nested block comments, such as:

```
/*...../*..../
(An incorrect pair)

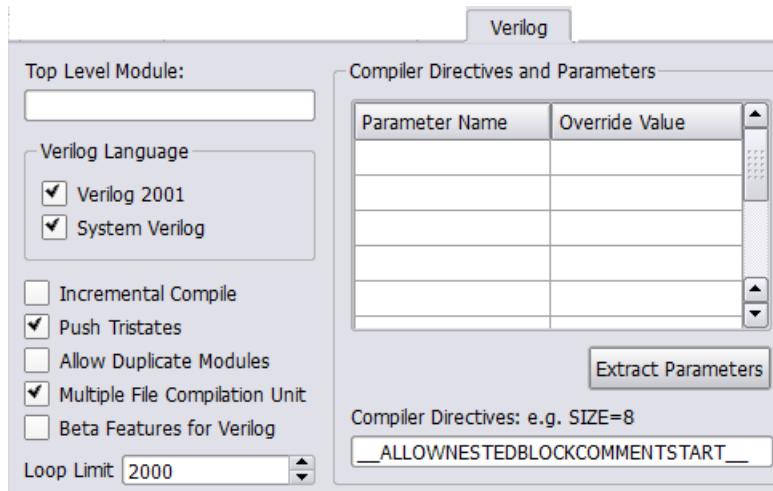
/*...../*...../*.....*/
```

To do this, the compiler uses the `_ALLOWNESTEDBLOCKCOMMENTSTART_` directive and parses the first `/*` until it encounters the associated pair `*/`, then ignores all content between and including any number of these lines and the block comments.

You can specify the `_ALLOWNESTEDBLOCKCOMMENTSTART_` directive with a `set_option -hdl_define` command added to the project file as shown below:

```
set_option -hdl_define -set _ALLOWNESTEDBLOCKCOMMENTSTART_
```

The directive can also be added to the Compiler Directives field of the Verilog panel.



## **\_SYN\_COMPATIBLE\_INCLUDEPATH\_**

Specifies that the search path order for includes to be the same as the one used by the simulation tool (VCS), instead of the following default search order, which searches the current logical library of the file where the module is instantiated first, then the library path for the current logical library, and lastly other logical libraries.

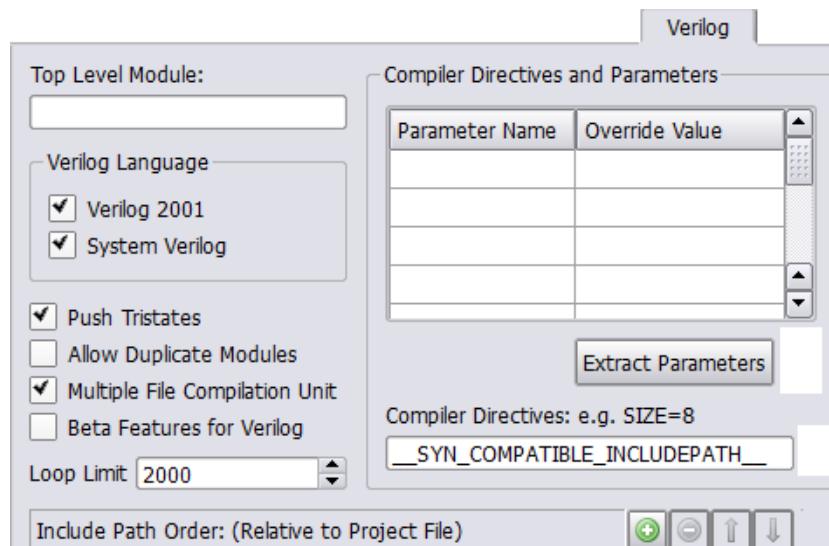
With `_SYN_COMPATIBLE_INCLUDEPATH_`, this is the search path order:

- Current working directory
- Include file directories, with first priority to RTL includes and then include paths

You can specify the `__SYN_COMPATIBLE_INCLUDEPATH__` directive with a `set_option -hdl_define` command added to the project file as shown below:

```
set_option -hdl_define -set __SYN_COMPATIBLE_INCLUDEPATH__
```

The directive can also be added to the Compiler Directives field of the Verilog panel.



## SYN\_COMPATIBLE

Use the `SYN_COMPATIBLE` macro to ensure compatibility between different Synopsys tools. Some Synopsys tools, such as Design Compiler (DC), ignore dynamic initialization assignments, unlike the synthesis tool. In the following example, note the line `logic a = b;` DC leaves the output unconnected, because DC does not handle inline assignments. By contrast, the synthesis tool handles inline assignments, so input `b` drives the output `q`.

```
module test (b, q);
 input b;
 output q;
 logic a = b;
 assign q = a;
endmodule
```

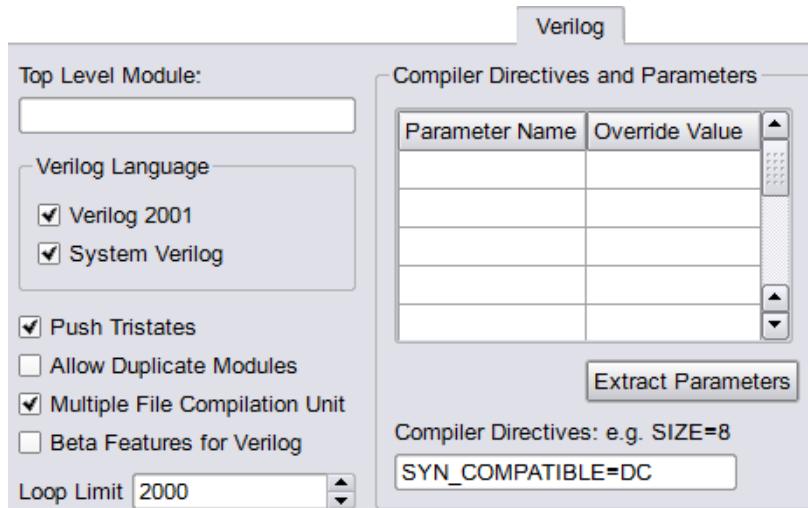
The tool warns you of the difference in handling (warning message @W: CG879), and treats the assignment as a regular assign. If you are using modules from DC and need it to be compatible with the synthesis tool, there are two ways to match the behavior and ignore the non-constant initial values:

1. Specify a macro with a Tcl command.

```
set_option -hdl_define -set SYN_COMPATIBLE=DC
```

2. Specify a macro through the GUI.

- To specify it from the GUI, go to the Verilog tab of the Implementation Options dialog box.
- In the Compiler Directives field, set SYN\_COMPATIBLE=DC.



## **\_\_SYN\_STRICT\_MODPORTS\_\_**

Use the \_\_SYN\_STRICT\_MODPORTS\_\_ macro to require that modports defined strictly access the associated interface ports specified for the instantiation. You might encounter the following error:

@E: CS172 The number of ports in the instantiation does not match the number of ports in the module definition for instance *instanceName*

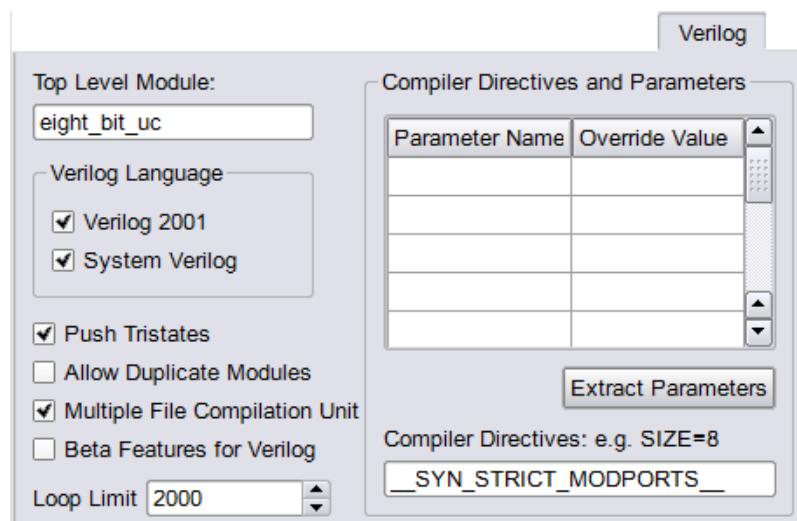
This can occur if you have specified a global interface instantiation for modports using the “.” syntax to access the ports. Set the `_SYN_STRICT_MODPORTS_` macro to ensure mapping completes successfully.

1. To specify the macro with a Tcl command.

```
set_option -hdl_define -set __SYN_STRICT_MODPORTS__
```

2. To specify the macro through the GUI.

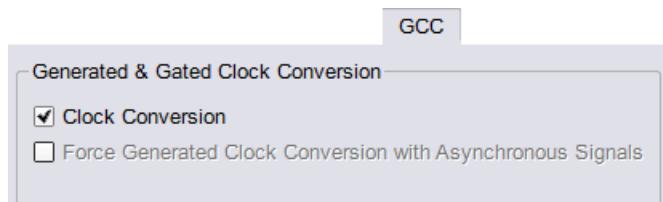
- To specify it from the GUI, go to the Verilog tab of the Implementation Options dialog box.
- In the Compiler Directives field, set `_SYN_STRICT_MODPORTS_`.



## GCC Panel

### *Synplify Pro*

The GCC panel specifies how gated and generated clocks are treated during synthesis. See [GCC & Prototyping Tools Panel, on page 481](#) for the Synplify Premier tool equivalent of this panel.



| Feature                                                    | Description                                                                                                                                                                                                                                                                                                                               |
|------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Clock Conversion                                           | Performs gated and generated clock optimization when enabled. See <a href="#">Working with Gated Clocks, on page 828</a> and <a href="#">Optimizing Generated Clocks, on page 881</a> of the <i>User Guide</i> for details.<br>Tcl equivalent: <b>set_option -fix_gated_and_generated_clocks 0 1</b>                                      |
| Force Generated Clock Conversion with Asynchronous Signals | Used with latches with asynchronous signals to force generated clock optimization when enabled. Note that conversions are only valid when the asynchronous controls are inactive. <sup>1</sup> This option is automatically enabled when a HAPS technology is selected.<br>Tcl equivalent: <b>set_option -force_async_genclk_conv 0 1</b> |

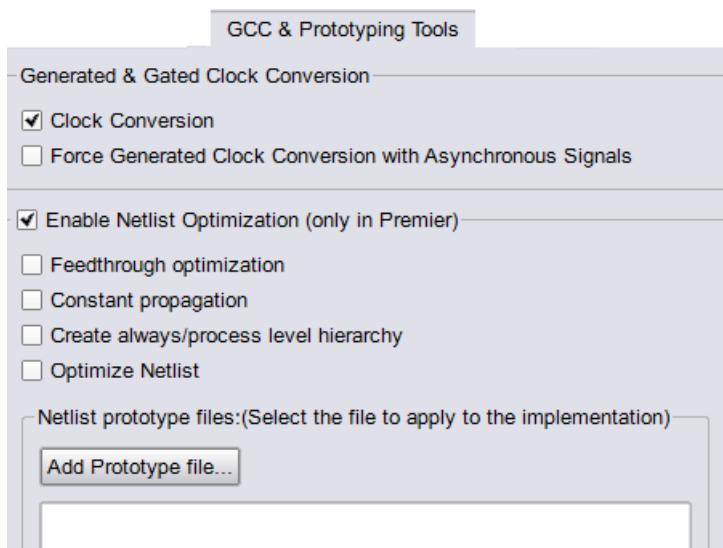
1. Otherwise, the Force Generated Clock Conversion with Asynchronous Signals option can

- Miss edges detected by the original circuit introduced by the asynchronous controls
- Trigger on invalid edges that were masked by the asynchronous controls in the original circuit

## GCC & Prototyping Tools Panel

*Synplify Premier*

The GCC & Prototyping Tools panel of the Implementation Options dialog box is used to specify if gated and generated clocks are to be converted during synthesis. You can also specify which netlist prototyping options are to be applied, and if a netlist restructuring or Tcl file is to be compiled with the design.

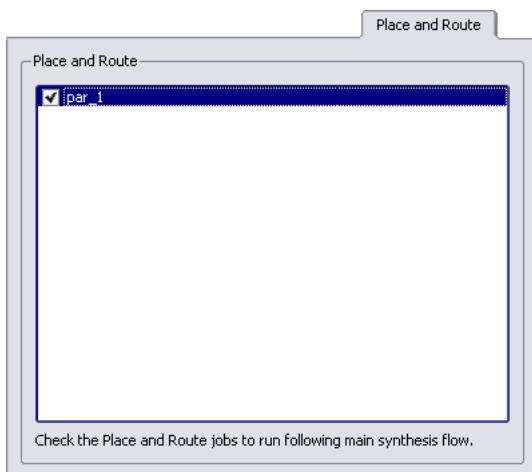


| Feature                                                    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Clock Conversion                                           | <p>Performs gated and generated clock optimization when enabled. See <a href="#">Working with Gated Clocks, on page 828</a> and <a href="#">Optimizing Generated Clocks, on page 881</a> of the <i>User Guide</i> for details.</p> <p>Tcl equivalent: <code>set_option -fix_gated_and_generated_clocks 0 1</code></p>                                                                                                                                                                                                     |
| Force Generated Clock Conversion with Asynchronous Signals | <p>When enabled, the tool converts gated clocks even if there are asynchronous set/reset signals in generated-clock logic or datapath latches. When disabled (the default), the tool does not convert gated clocks if asynchronous set/reset signals on generated-clock logic are different from the asynchronous signals on the driven datapath latches.</p> <p>This option is automatically enabled when a HAPS technology is selected.</p> <p>Tcl equivalent: <code>set_option -force_async_genclk_conv 0 1</code></p> |
| Enable Netlist Optimization                                | <p>Enables the individual prototyping tools options to be selected and also enables the Add Prototype file button to allow a netlist restructuring file to be included in the project.</p> <p>Tcl equivalent: <code>set_option -enable_nfilter 0 1</code></p>                                                                                                                                                                                                                                                             |

| Feature                               | Description                                                                                                                                                                                                                                                                                              |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Feedthrough optimization              | Prototyping tool option that eliminates unused routes through a module after it has been compiled.<br>Tcl equivalent: <b>set_option -popfeed 0 1</b>                                                                                                                                                     |
| Constant propagation                  | Prototyping tool option that eliminates the I/O pin count associated with tying a logic input signal high or low by making the connection internal to the instance rather than through the instance interface.<br>Tcl equivalent: <b>set_option -constprop 0 1</b>                                       |
| Create always/process level hierarchy | Prototyping tool option that creates an additional level of hierarchy based on the always (Verilog) or process (VHDL) blocks in the source RTL design file. Allows large modules or entities to be broken up into a set of logical submodules.<br>Tcl equivalent: <b>set_option -createhierarchy 0 1</b> |
| Optimize Netlist                      | Prototyping tool option that removes unnecessary and redundant logic.<br>Tcl equivalent: <b>set_option -optimize_netlist 0 1</b>                                                                                                                                                                         |
| Add Prototype file                    | Brings up an Add File dialog box to allow a prototyping tool file (bit-slice definition) or Tcl file to be specified for the project.<br>Tcl equivalent: <b>set_option -nfilter_user_path filename</b>                                                                                                   |

## Place and Route Panel

The Place and Route panel allows you to run selected place-and-route jobs after design synthesis. To create a place-and-route job, see [Add P&R Implementation Popup Menu Command, on page 626](#) or [Options for Place & Route Jobs Popup Menu Command, on page 630](#) for details.



# Import Menu

The Import menu provides support for handling various IPs in the Synopsys FPGA synthesis tools.

The following table describes the Import IP menu commands.

| Command                       | Description                                                                                                                                                                                                                                         |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add Vivado IP                 | Opens the initial Add IP dialog box that lets you import Vivado IP into your project. See <a href="#">Add Vivado IP Command , on page 486</a> .<br>Tcl equivalent: <code>add_vivado_ip IPsource IPimpMode additionalOptions</code>                  |
| Import Xilinx EDK/ISE Project | Opens the ise2syn utility that lets you convert ISE or EDK projects to synthesis projects. See <a href="#">Import Xilinx EDK/ISE Project Command , on page 489</a> .<br>Tcl equivalent: <code>ise2syn -ise projectName</code>                       |
| Import Intel SOPC Project     | Opens the sopc2syn interface that lets you include subsystems created with SOPC builder into the synthesis project. See <a href="#">Import Intel SOPC Project Command , on page 492</a> .<br>Tcl equivalent: <code>sopc2syn -ptf projectName</code> |
| Import Intel QSF Project      | Opens the qsf2syn utility that lets you convert QSF projects to synthesis projects. See <a href="#">Import Intel QSF Project Command , on page 495</a> .<br>Tcl equivalent: <code>qsf2syn -qsf projectName</code>                                   |
| Import IP Package             | Imports an IP package into your project. You need to consolidate the IP core files into a single folder or a compressed zip file before importing. See <a href="#">Import IP Package Command , on page 496</a> .                                    |
| Launch Vivado IP Catalog      | Launches the Vivado place-and-route tool with the specified options.<br>See <a href="#">Launch Vivado Command , on page 497</a> .<br>Tcl equivalent: <code>launch_vivado -project projectName</code>                                                |

## Add Vivado IP Command

This command adds Vivado IP to the current project with the specified options. You can add the IP individually for a single XCI or DCP file, or as a group using a directory or a list file. You can import the IP as part of the project or as a subproject. You can also specify whether the IP is absorbed into the design or treated as a white box. For details about using this command, see [Incorporating Vivado IP, on page 740](#) in the *User Guide*.

The GUI consists of multiple pages, which are described below. For the Tcl equivalent, see [add\\_vivado\\_ip, on page 31](#).

### Add Vivado IP: Page 1

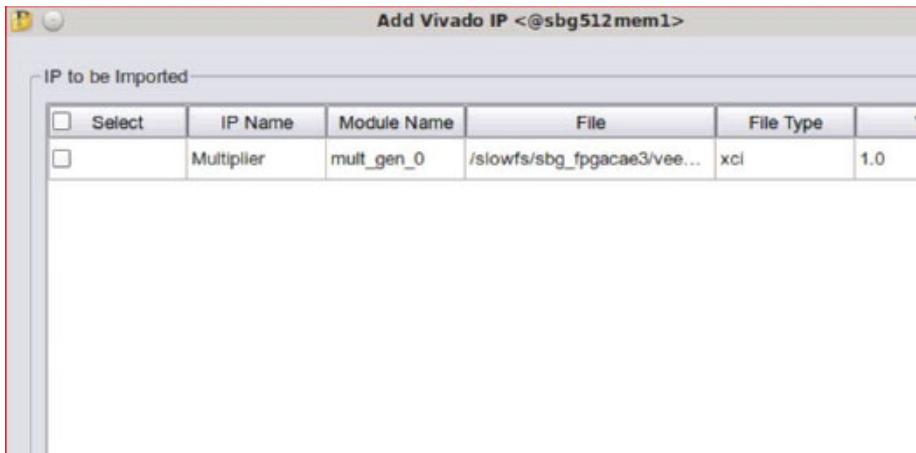
The first page contains the IP specifications:



| Field                | Description                                                                                                                                                                                                                                                                                   |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Specify IP Directory | Specifies a directory that contains the IP blocks. The tool searches this directory for IP files and then generates a table of IP blocks that were found on the second GUI page.                                                                                                              |
| Specify IP file      | Specifies one of several file types: XCI, DCP, or TXT. The XCI or DCP file specifies a single IP block to import. A text file contains a list of IP blocks to import. For a TXT file, the tool generates a table of IP blocks on the second GUI page, which is similar to the directory mode. |

## Add Vivado IP: Page 2

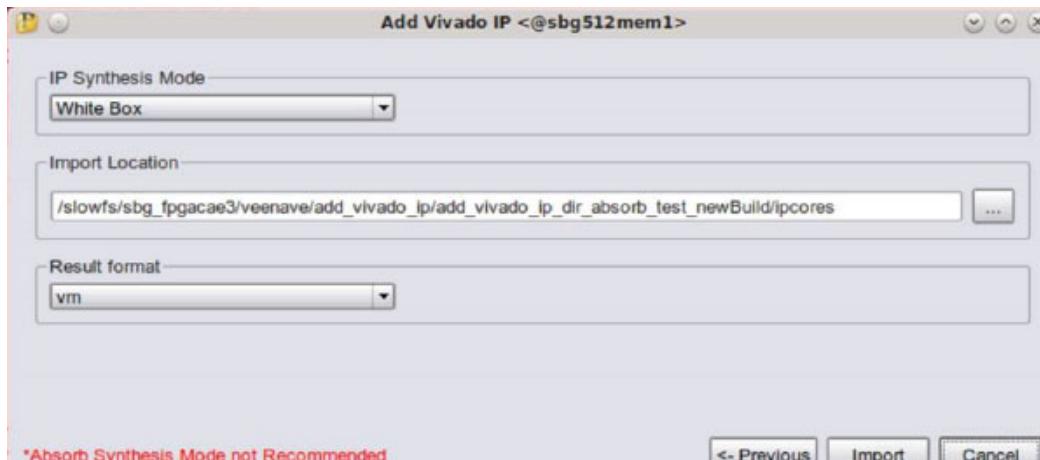
The second page of the dialog box displays a table of the IP blocks to import.



| Field       | Column Description                                                                          |
|-------------|---------------------------------------------------------------------------------------------|
| Select      | Checkbox status indicates whether to import the IP block.                                   |
| IP Name     | Specifies the name of the IP block.                                                         |
| Module Name | Specifies the name of the IP module.                                                        |
| File        | Specifies the location of the IP block.                                                     |
| Version     | Specifies the version of the IP block.                                                      |
| Settings    | Indicates if the IP block is to be synthesized before it is imported (Synthesize & Import). |
| Import      | Imports the IP using the criteria set.                                                      |

## Add Vivado IP: Page 3

The third page of the dialog box describes the IP project options.

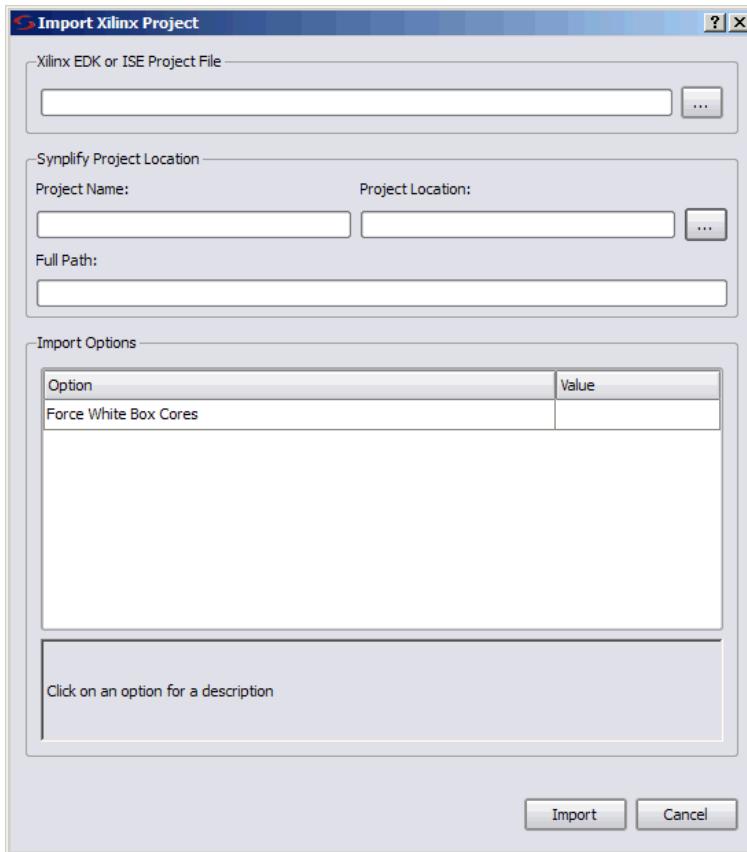


| Field             | Description                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IP Synthesis Mode | Specifies how the IP block is to be included in the design. Select one of the following modes: <ul style="list-style-type: none"><li>• Absorb - Allows the IP to be optimized and included in the final design netlist.</li><li>• White Box - Uses the IP for timing estimation, but does not optimize or include it in the final netlist. A DCP file is used to provide the content of the IP block for place and route.</li></ul> |
| Import Location   | Identifies a location to place the IP-related files generated during the import IP process.                                                                                                                                                                                                                                                                                                                                         |
| Import            | Imports the IP using the criteria set.                                                                                                                                                                                                                                                                                                                                                                                              |

## Import Xilinx EDK/ISE Project Command

Use the Import Xilinx EDK/ISE Project command to open the GUI interface to the ise2syn conversion utility. Use this utility to convert a Xilinx .ise, .xise, or .xmp project into a Synplify Pro or Synplify Premier project for synthesis. For information about using the utility to convert a Xilinx project, see [Converting Xilinx Projects with ise2syn, on page 761](#) in the *User Guide*.

For information about the equivalent Tcl command for batch mode, see [ise2syn, on page 95](#).



The following table describes the options you can set.

| Field                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Xilinx ISE or EDK Project File | Specifies an ISE project (ise/xise) or an EDK project (xmp) to be imported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Project Name                   | Specifies a name for the synthesis project file to be created. The default name for the file is <i>baseNameOfInputProject</i> /synplify. If the input file is system.xmp, the default project file name is system.prj.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Project Location               | Specifies the directory for the synthesis project file. You must have write access to this directory.<br>The default location for the file is the directory for the input project file. If the input file is E:\ise2syn\Br2_Exbox2\system.xmp, the default project location is E:\ise2syn\Br2_Exbox2\synplify.                                                                                                                                                                                                                                                                                                                                                     |
| Full Path                      | Automatically updated field that reflects the values specified in Project Name and Project Location.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Force White Box Cores          | Specifies cores to be treated as white boxes. This option is only applicable to EDK designs. Use this option only if the tool errors out because it cannot read the core information because it is encrypted or in non-standard HDL. See <a href="#">Force White Box Cores Option (EDK)</a> , on page 491 for more information about using this option.<br><br>When this option is enabled, the utility uses .ngc files instead of HDL files for the core. Use this option for EDK cores and cores that use non-standard HDL.<br><br>To specify multiple core names, separate the names with commas. Do not leave white space between the core name and the comma. |

When you click Import, the ise2syn utility generates the following output files:

|                |                                                                         |
|----------------|-------------------------------------------------------------------------|
| .prj           | Synplify Pro or Synplify Premier synthesis project file                 |
| _conv.sdc      | Synthesis constraint file, with converted ucf, ncf, or xcf constraints  |
| _unapplied.ucf | Xilinx User Constraint file, that contains any unconverted constraints. |

## Force White Box Cores Option (EDK)

Use this option to specify EDK cores to be white-boxed. Use this option only if the tool errors out because it cannot read the core information. By default, the tool automatically white boxes certain core configurations, and you do not have to enable the Force White Box Cores option for them. The tool automatically white boxes the following cores:

|                                     |                                                                                                                                                                        |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Xilinx specific core configurations | ddr<br>ddr2<br>ddr2_sdram<br>ipic_to_temac<br>opb_ddr<br>mch_opb_ddr<br>plb_ddr2<br>plb_master_llink<br>plb_rapidio_lvds<br>plb_temac<br>plbv46_master<br>plbv46_pcnie |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                               |                                        |
|---------------------------------------------------------------|----------------------------------------|
| Unsupported HDL constructs (in the Synplify Pro/Premier tool) | clock_generator<br>mpmc<br>xps_mch_emc |
|---------------------------------------------------------------|----------------------------------------|

|                 |  |
|-----------------|--|
| Encrypted cores |  |
|-----------------|--|

Enable the Force White Box Cores option if your project has cores that are not listed in the previous table. For EDK project cores, specify any cores that are not listed in the previous table and which meet either of these criteria:

- Xilinx-specific core configurations that are not supported by the synthesis tools. For some cores, Xilinx tools configures the cores during synthesis based on some Xilinx specific directives.
- Xilinx cores that use non-standard HDL constructs and which are not supported by the synthesis tool compiler, like signal width mismatch for example.

To specify cores to be white-boxed, do the following:

1. Type the name of the core to be white-boxed in the Force White Box Cores option.

- To locate the core name that corresponds to a HDL file, open the Synplify Pro or Synplify Premier project as text and locate the HDL file in it.
- The core name is displayed as a comment at the beginning. For example:

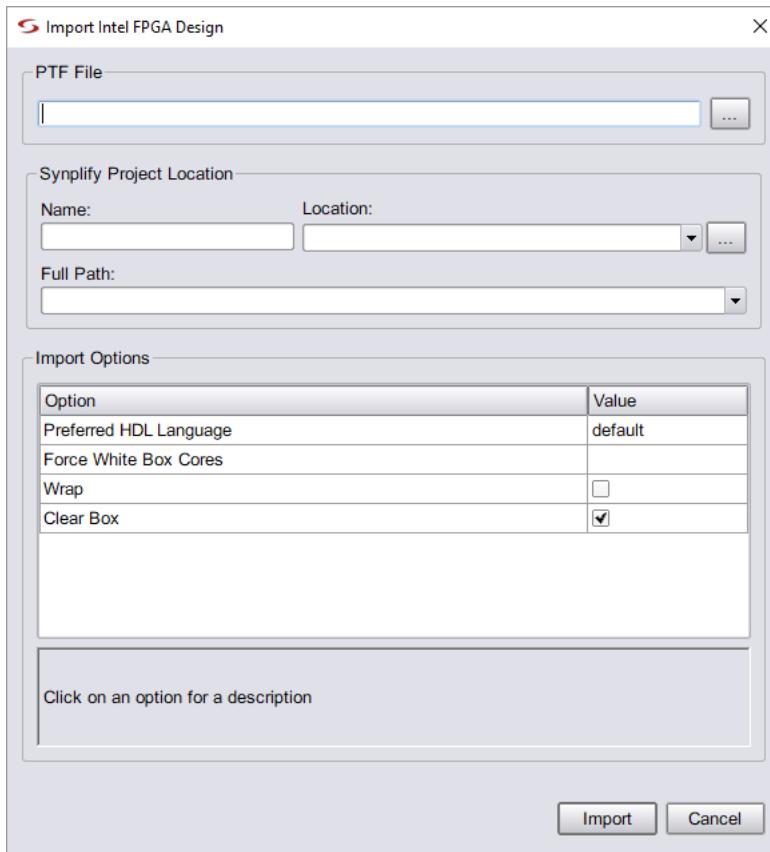
```
#Core Name = ppc405_virtex4>Core Instance = ppc405_0, Core Version
add_file -verilog $XILINX_XMP_PRJ_DIR/pcores/ppc405_virtex4_v2_01_b/h
add_file -vhdl -lib ppc405_virtex4_v2_01_b $XILINX_XMP_PRJ_DIR/pcores
add_file -vhdl -lib work $XILINX_XMP_PRJ_DIR/hdl/ppc405_0_wrapper.vhd
```

2. To specify a list of cores, separate the names with commas. Alternatively, specify a file that contains a comma-separated list of files to be white-boxed.

## Import Intel SOPC Project Command

The Import Intel SOPC Project command allows you to include subsystems created with the SOPC Builder into the Synplify Pro or Synplify Premier project for synthesis. SOPC Builder is the Intel embedded system development tool, and is used to design embedded processor systems and subsystems in Intel FPGAs with embedded NIOS soft processor cores. The Import Intel FPGA SOPC Project command uses the *sopc2syn* utility, whose command line syntax is described in [sopc2syn, on page 197](#).

For information about using these components and cores in your design, see [Working with SOPC Builder Components, on page 713](#) and [Including Intel FPGA Processor Cores from SOPC Builder, on page 707](#) in the *User Guide*.



The following table describes the Import Intel SOPC Project menu options.

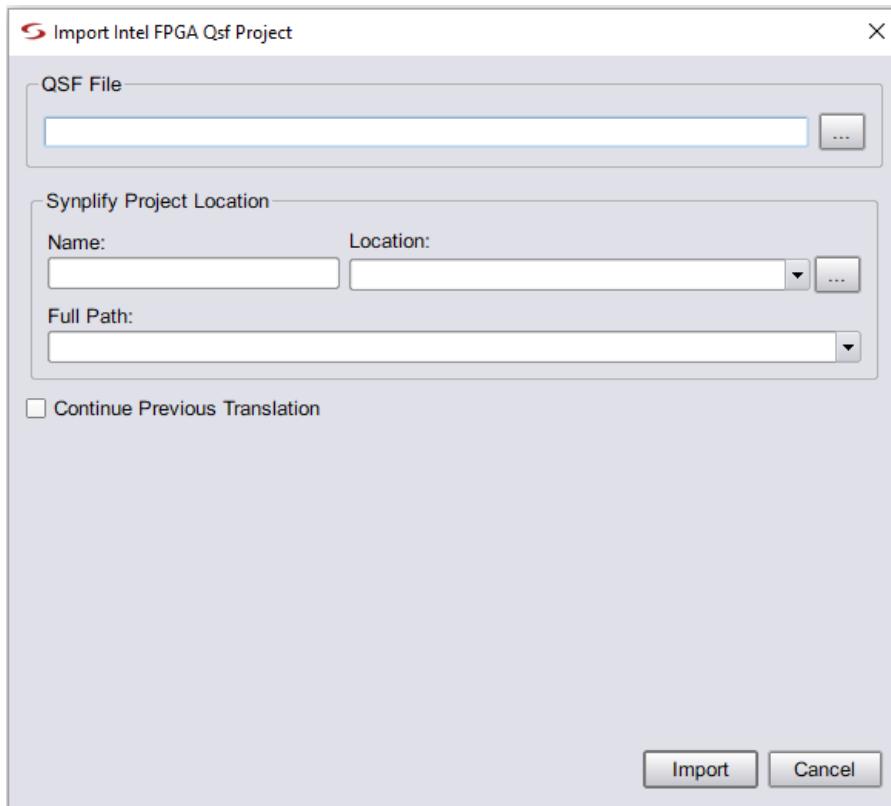
| Command          | Description                                                                       |
|------------------|-----------------------------------------------------------------------------------|
| PTF File         | Specifies the SOPC Builder plain test file (ptf) to be imported into the project. |
| Project Name     | Specifies a name for the synthesis project.                                       |
| Project Location | Specifies a location for the project.                                             |
| Full Path        | Specifies the full path to the project directory.                                 |

| Command                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Preferred HDL Language | Sets the preferred HDL language in case of duplicate files. You can select one of the following: <ul style="list-style-type: none"> <li>• VHDL</li> <li>• Verilog</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Force White Box Cores  | This is an advanced option that forces listed cores to be treated as white boxes. This means that the utility uses .vqm files for core definitions. This option has no default. <ul style="list-style-type: none"> <li>• <i>string</i> is a comma-separated list of cores to be white-boxed.</li> <li>• <i>file</i> is a file containing a comma-separated list of cores to be white-boxed.</li> </ul> See <a href="#">Defining SOPC Components as Black Boxes and White Boxes, on page 716</a> of the <i>User Guide</i> for more information.                                                                                                                                                                                                                                               |
| Wrap                   | This advanced option enables or disables the generation of missing core wrappers. <ul style="list-style-type: none"> <li>• When enabled, it generates missing core wrappers.</li> <li>• When disabled, it does not generate missing core wrappers. This is the recommended setting and the default.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Clear Box              | This option specifies whether or not to use the clear box flow for encrypted cores. This also determines if the cores are treated as black boxes. (See <a href="#">Defining SOPC Components as Black Boxes and White Boxes, on page 716</a> of the <i>User Guide</i> .) <ul style="list-style-type: none"> <li>• When enabled, it uses the clear box flow. The tool adds black box wrapper files for encrypted cores. It adds the HDL files for the encrypted cores to the project, but only uses them to run place-and-route. This is the default.</li> <li>• When disabled, it does not use the clear box flow. Black box wrapper files for the encrypted cores are added to the project and the HDL files for the encrypted cores are copied to the place-and-route directory.</li> </ul> |

## Import Intel QSF Project Command

Use the Import Intel QSF Project command to converts an Intel Quartus project into a Synplify Pro or Synplify Premier project for synthesis. This command uses the `qsf2syn` utility, the syntax for which is described in [qsf2sdc, on page 124](#).

For detailed information about using the command and converting Quartus projects, see [Importing Projects from Quartus, on page 717](#) in the *User Guide*.



The following table describes the Import Intel QSF Project menu options.

| Command                       | Description                                                                                                                                                                                                                                      |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Intel QSF Project File        | Specifies the .qsf file for the Quartus project to be imported. See <a href="#">Imported Quartus Project Settings and Timing Constraints, on page 722</a> in the <i>User Guide</i> for a list of the supported constraints and project settings. |
| Project Name                  | Specifies a name for the synthesis project to be generated. See <a href="#">Synthesis Files Generated After Importing the Quartus Project, on page 721</a> in the <i>User Guide</i> for a list of the files that are created.                    |
| Project Location              | Specifies a location for the synthesis project.                                                                                                                                                                                                  |
| Full Path                     | Echoes the full path to the synthesis project directory.                                                                                                                                                                                         |
| Continue Previous Translation | Lets you continue translation from the point at which you stopped to debug or fix your design. For details on using this option, see <a href="#">Importing Quartus Projects, on page 718</a> in the <i>User Guide</i> .                          |

## Import IP Package Command

Use the Import IP Package command to help you import IP core files to your Project view.

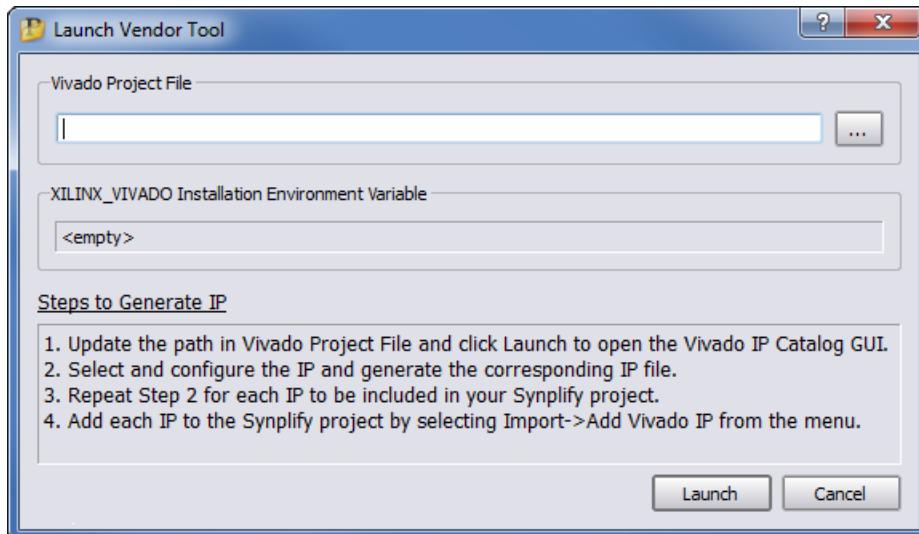


The following table describes the Import IP Package menu options.

| Command      | Description                                                                                                                                                     |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IP Package   | Location of the consolidated ZIP file containing the IP core files. The file format can be zip or tar. Use the Browse button to help you locate the IP package. |
| IP Directory | Directory location of the consolidated IP core files. Use the Browse button to help you locate the IP directory.                                                |
| Package Name | The name of the top-level module for the IP core.                                                                                                               |

## Launch Vivado Command

The Launch Vivado command launches Vivado with the specified options from the indicated project file. Fill in the Vivado Project File field and the XILINX\_VIVADO Installation Environment Variable field (the Vivado executable used is \$VIVADO\_ROOTDIR/bin/vivado) and then click the Launch button.



# Run Menu

You use the Run menu to perform the following tasks:

- Compile a design, without mapping it.
- Synthesize (compile and map) or resynthesize a design.
- Check design syntax and synthesis code, and check source code errors.
- Check constraint syntax and how/if constraints are applied to the design.
- Run Tcl scripts.
- Run all implementations at once.
- Check the status of the current job.

The following table describes the Run menu commands.

| Command          | Description                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Run              | <p>Synthesizes (compiles and maps) the top-level design. For the compile point flow, this command also synthesizes any compile points whose constraints, implementation options, or source code changed since the last synthesis run. You can view the result of design synthesis in the RTL and Technology views.</p> <p>Same as clicking the Run button in the Project view.</p> <p>Tcl equivalent: <b>project -run</b></p> |
| Resynthesize All | <p>Resynthesizes (compiles and maps) the entire design, including the top level and <i>all compile points</i>, whether or not their constraints, implementation options, or source code changed since the last synthesis. If you do <i>not</i> want to force a <i>recompilation of all compile points</i>, then use Run-&gt;Run instead.</p> <p>Tcl equivalent: <b>project -run synthesis -clean</b></p>                      |
| Compile Only     | <p>Compiles the design into technology-independent high-level structures. You can view the result in the RTL view.</p> <p>Tcl equivalent: <b>project -run compile</b></p>                                                                                                                                                                                                                                                     |

| Command                        | Description                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Write Output Netlist Only      | <p>Generates an output netlist after synthesis has been run. This command generates the netlists you specify on the Implementation Results tab of the Implementation Options dialog box.</p> <p>You can also use this command in an incremental timing analysis flow. See <a href="#">Generating Custom Timing Reports with STA</a>, on page 482 for details.</p> <p>Tcl equivalent: <b>project -run write_netlist</b></p> |
| Estimate Area                  | <p><i>Synplify Premier</i></p> <p>For details, see <a href="#">Run Menu: Design Planner Commands , on page 666</a>.</p> <p>Tcl equivalent: <b>project -run estimator</b></p>                                                                                                                                                                                                                                               |
| Compile Physical Hierarchy     | <p><i>Synplify Premier</i></p> <p>For details, see <a href="#">Run Menu: Design Planner Commands , on page 666</a>.</p> <p>Tcl equivalent: <b>project -run partitioner</b></p>                                                                                                                                                                                                                                             |
| FSM Explorer                   | <p><i>Synplify Pro, Synplify Premier</i></p> <p>Analyzes finite state machines contained in a design, and selects the optimum encoding style. This menu command is not available in some views.</p> <p>Tcl equivalent: <b>project -run fsm_explorer</b></p>                                                                                                                                                                |
| Translate Vendor IO            | <p>Intel, <i>Xilinx</i></p> <p>Displays a dialog box where you can specify an Intel PIN or Xilinx PAD file to translate into an FPGA constraint file (.sdc).</p> <p>See <a href="#">Translate Vendor IO Command , on page 504</a>.</p>                                                                                                                                                                                     |
| Post Place & Route Resynthesis | <p><i>Synplify Premier</i><br/><i>Xilinx</i></p> <p>Provides timing closure and better routability for the design.</p> <p>For details, see <a href="#">Post Place &amp; Route Resynthesis Command , on page 504</a>.</p> <p>Tcl equivalent:</p> <p><b>post_par_resynthesis -dcp <i>fileName</i> -rundir <i>directoryPath</i></b></p>                                                                                       |

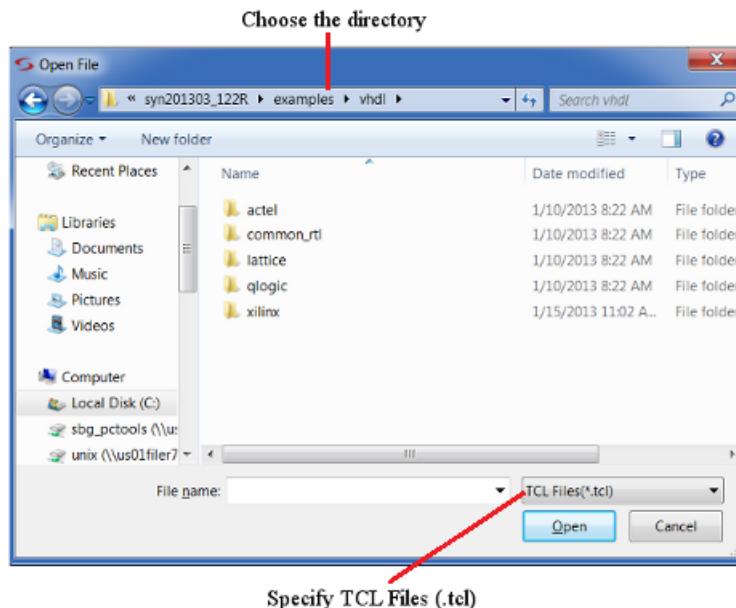
| Command                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Syntax Check                 | <p>Runs a syntax check on design code. The status bar at the bottom of the window displays any error messages. If the active window shows an HDL file, then the command checks only that file; otherwise, it checks all project source code files.</p> <p>Tcl equivalent: <b>project -run syntax_check</b></p>                                                                                                                                                          |
| Synthesis Check              | <p>Runs a synthesis check on your design code. This includes a syntax check and a check to see if the synthesis tool could map the design to the hardware. No optimizations are carried out. The status bar at the bottom of the window displays any error messages. If the active window shows an HDL file, then the command checks only that file; otherwise, it checks all project source code files.</p> <p>Tcl equivalent: <b>project -run synthesis_check</b></p> |
| Constraint Check             | <p>Checks the syntax and applicability of the timing constraints in the .fdc/.sdc file for your project and generates a report (<i>projectName_cck.rpt</i>). The report contains information on the constraints that can be applied, cannot be applied because objects do not exist, and wildcard expansion on the constraints.</p> <p>See <a href="#">Constraint Checking Report, on page 218</a>.</p> <p>Tcl equivalent: <b>project -run constraint_check</b></p>     |
| Arrange VHDL files           | <p>Reorders the VHDL source files for synthesis.</p> <p>Tcl equivalent: <b>project -run hdl_info_gen fileorder</b></p>                                                                                                                                                                                                                                                                                                                                                  |
| Launch Identify Instrumentor | <p>Launches the Identify Instrumentor within the FPGA synthesis tools; See <a href="#">Working with the Identify Tools, on page 1159</a> to launch the debugger tools.</p>                                                                                                                                                                                                                                                                                              |
| Launch Identify Debugger     | <p>Launches the Identify debugger tool. For more information, see: <a href="#">Working with the Identify Tools, on page 1159</a> of the <i>User Guide</i>.</p> <p>To launch the Identify debugger in batch mode, use the <code>set_option -identify_debug_mode 1</code> Tcl command to create an Identify implementation.</p>                                                                                                                                           |
| Launch SYNCORE               | <p>Opens the Synopsys FPGA IP Core Wizard. This tool helps you build IP blocks such as memory or FIFO models for your design.</p> <p>See the <a href="#">Launch SYNCORE Command , on page 508</a> for details.</p>                                                                                                                                                                                                                                                      |

| Command                            | Description                                                                                                                                                                                                                                                                                                                                                         |
|------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Configure and Launch VCS Simulator | Allows you to configure and launch the VCS simulator. See <a href="#">Configure and Launch VCS Simulator Command , on page 508.</a>                                                                                                                                                                                                                                 |
| SpyGlass RTL Analysis              | Use the SpyGlass tool to run quick lint checks with easy setup for your design. See <a href="#">Using the SpyGlass Tool for RTL Analysis, on page 93.</a>                                                                                                                                                                                                           |
| Export Verdi                       | Allows the Synplify tool to export a Verdi® script for your project. See <a href="#">Exporting Verdi Scripts, on page 755</a>                                                                                                                                                                                                                                       |
| Run Tcl Script                     | Displays the Open dialog box, where you choose a Tcl script to run. See <a href="#">Run Tcl Script Command , on page 502.</a>                                                                                                                                                                                                                                       |
| Run Implementations Setup          | <i>Synplify Pro, Synplify Premier</i><br>Runs all selected implementations for one project at the same time. See <a href="#">Run Implementations Setup Command , on page 502.</a><br>Tcl equivalent: <code>run -impl "implementation1 implementation2..." -parallel</code>                                                                                          |
| Job Status                         | During compilation, tells you the name of the current job, and gives you the runtime and directory location of your design. This option is enabled during synthesis. See <a href="#">Job Status Command , on page 505.</a> Clicking in the status area of the Project view is a shortcut for this command.                                                          |
| Next Error/Warning                 | Shows the next error or warning in your source code file.                                                                                                                                                                                                                                                                                                           |
| Previous Error/Warning             | Shows the previous error or warning in your source code file.                                                                                                                                                                                                                                                                                                       |
| Log File Message Filter            | Allows messages in the current session to be elevated in severity (for example, promoted to an error from a warning), lowered in severity (for example, demoting a warning to a note), or suppressed from the log file after the next run through the Log File Filter dialog box. For more information, see <a href="#">Log File Message Controls, on page 314.</a> |

## Run Tcl Script Command

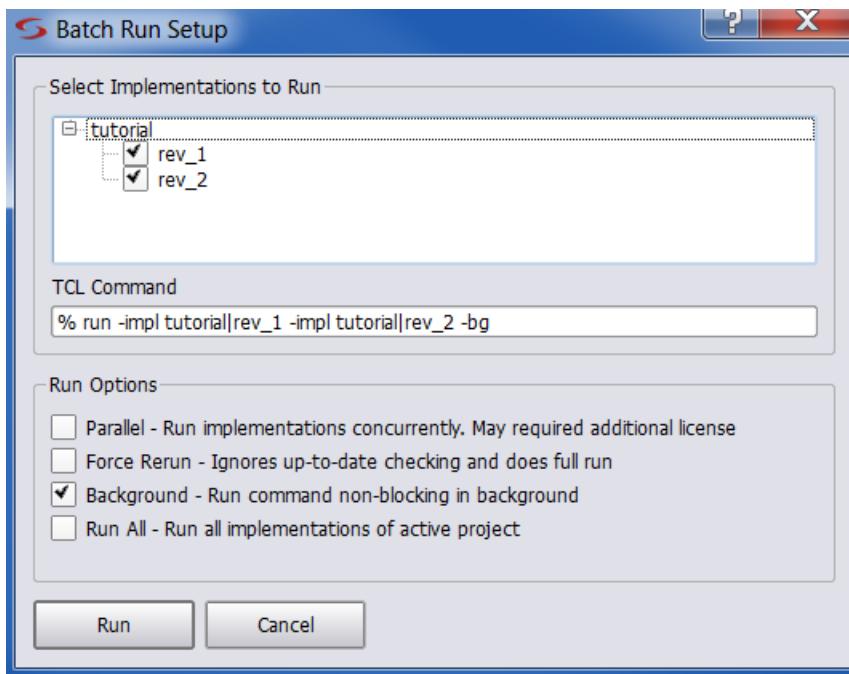
Select Run->Run Tcl Script to display the Open dialog box, where you specify the Tcl script file to execute. The File name area is filled automatically with the wildcard string “\*.tcl”, corresponding to Tcl files.

This dialog box is the same as that displayed with File->Open, except that no Open as read-only check box is present. See [Open Project Command, on page 401](#), for an explanation of the features in the Open dialog box.



## Run Implementations Setup Command

Select Run->Run Implementations Setup to run selected implementations of a Project file in batch mode. To use the Batch Run Setup dialog box, check one or more implementations from the list displayed and click the Run button.



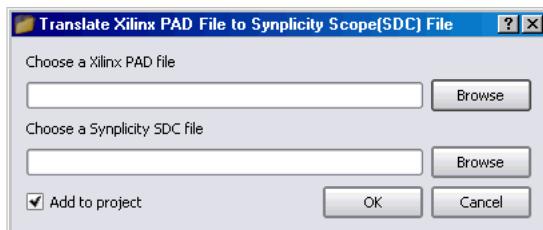
You can also choose to run the selected implementations with one or more of the following options:

| Command     | Description                                                                                                                                                                                  |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parallel    | Runs specified implementations concurrently. This may require additional licenses.<br>Tcl equivalent: <b>run -impl implName -parallel</b>                                                    |
| Force Rerun | Runs specified implementations, while ignoring up-to-date checking. This option clears all previous results and forces a complete rerun.<br>Tcl equivalent: <b>run -impl implName -clean</b> |
| Background  | Runs specified implementations in non-blocking background mode.<br>Tcl equivalent: <b>run -impl implName -bg</b>                                                                             |
| Run All     | Runs all implementations of the active project.<br>Tcl equivalent: <b>run -all</b>                                                                                                           |

## Translate Vendor IO Command

Intel, Xilinx

Select Run->Translate Vendor IO to translate the Xilinx PAD or Intel PIN file into an sdc constraint file. This command displays the appropriate translation dialog box for your target vendor (Translate Xilinx PAD File to Synplicity Scope (SDC) File or Translate Intel PIN File to Synplicity Scope (SDC) File), where you specify the input and output files for the translation. Enable Add to project to add the constraint file to your project.

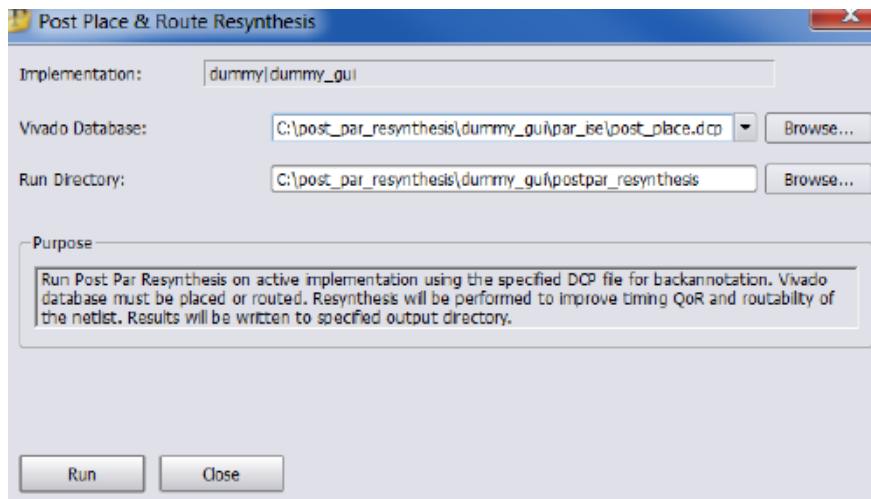


## Post Place & Route Resynthesis Command

Synplify Premier

Intel, Xilinx

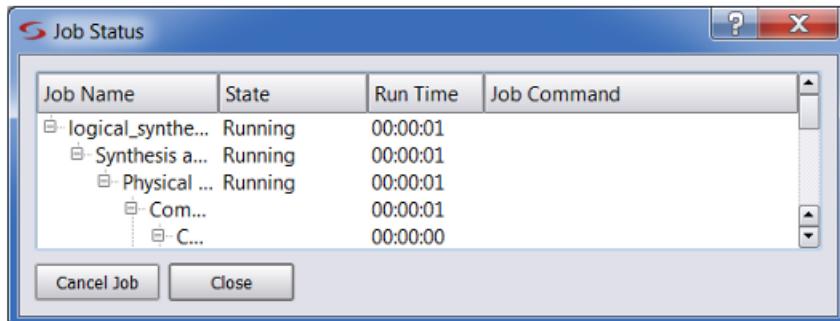
Select Run->Post Place & Route Resynthesis after running logic synthesis and place and route, if your design does not meet timing performance or fails to route completely to fix these problems. This feature implements timing closure and better routability for the design with fast turnaround time. For details, see [Using Post Place and Route Resynthesis, on page 1151](#).



## Job Status Command

Select Run->Job Status to monitor the synthesis jobs that are running, their run times, and their associated commands. This information appears in the Job Status dialog box. This dialog box is also displayed when you click in the status area of the Project view (see [The Project View](#), on page 34).

You can cancel a displayed job by selecting it in the dialog box and clicking Cancel Job.

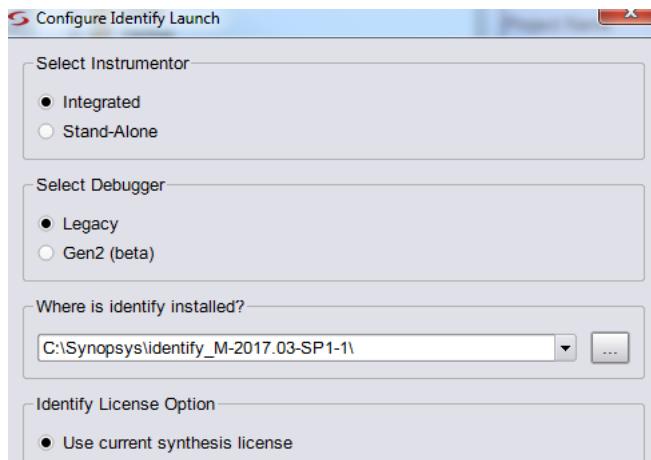


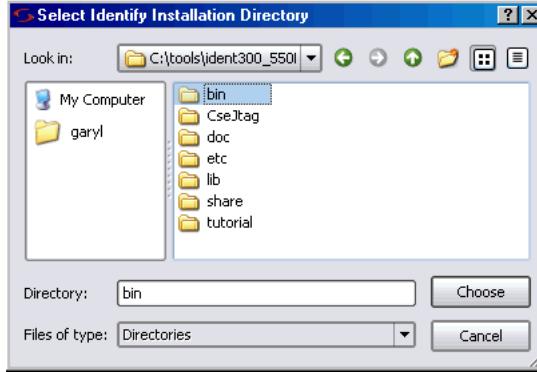
## Identify Instrumentor Command

The Identify Instrumentor command lets you start the integrated or stand-alone Identify instrumentor from within the synthesis interface. Before you can use this command, you must define an Identify implementation in the project view. For a description of the work flow using the Identify instrumentor, see [Working with the Identify Tools, on page 1159](#) in the *User Guide*.

### Configure Identify Launch Dialog Box

Select Options->Configure Identify Launch to display this dialog box or which is automatically displayed when the location of the Identify executable has not been previously defined.



| Command                                                  | Description                                                                                                                                                                                                                                                                                                                                                           |
|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Select Instrumentor                                      | Click the radio button to select which version of the instrumentor to use. You can choose either the integrated or stand-alone Identify Instrumentor.                                                                                                                                                                                                                 |
| Locate Identify Installation (for the Identify Debugger) | A pointer to the Identify install directory. Use the (...) button to navigate to the directory location.                                                                                                                                                                                                                                                              |
|                                                          |                                                                                                                                                                                                                                                                                     |
| Identify License Option                                  | Radio buttons to select the Identify license option. Select Use current synthesis license when only a single TSL license is available; select Use separate Identify Instrumentor license when multiple licenses are available. With a single TSL license, you are prohibited from compiling or mapping in the synthesis tool while the Identify instrumentor is open. |

## Launch Identify Debugger Command

The Launch Identify Debugger command launches a stand-alone version of the Identify Debugger software from the synthesis interface. Before you can use this command, you must have an active Identify implementation and an instrumented design. For a description of the work flow using the Identify/Identify RTL Debugger software, see [Working with the Identify Tools, on page 1159](#) in the *User Guide*.

## Launch SYNCORE Command

The SYNCORE wizard helps you build IP cores. The wizard can compile RAM and ROM memories including a byte-enable RAM, a FIFO, an adder/subtractor, and a counter. The resulting Verilog models can be synthesized and simulated. For details about using the wizard to build these models, see the following topics:

- [SYNCORE FIFO Compiler, on page 334](#)
- [SYNCORE RAM Compiler, on page 365](#)
- [SYNCORE Byte-Enable RAM Compiler, on page 387](#)
- [SYNCORE ROM Compiler, on page 403](#)
- [SYNCORE Adder/Subtractor Compiler, on page 418](#)
- [SYNCORE Counter Compiler, on page 442](#)

## Configure and Launch VCS Simulator Command

The Configure and Launch VCS Simulator command enables you to launch VCS simulation from within the Synopsys FPGA synthesis tools. Additionally, configuration information, such as libraries and options can be specified on the Run VCS Simulator dialog box before running VCS simulation. You can launch this simulation tool from the synthesis tools on Linux platforms only.

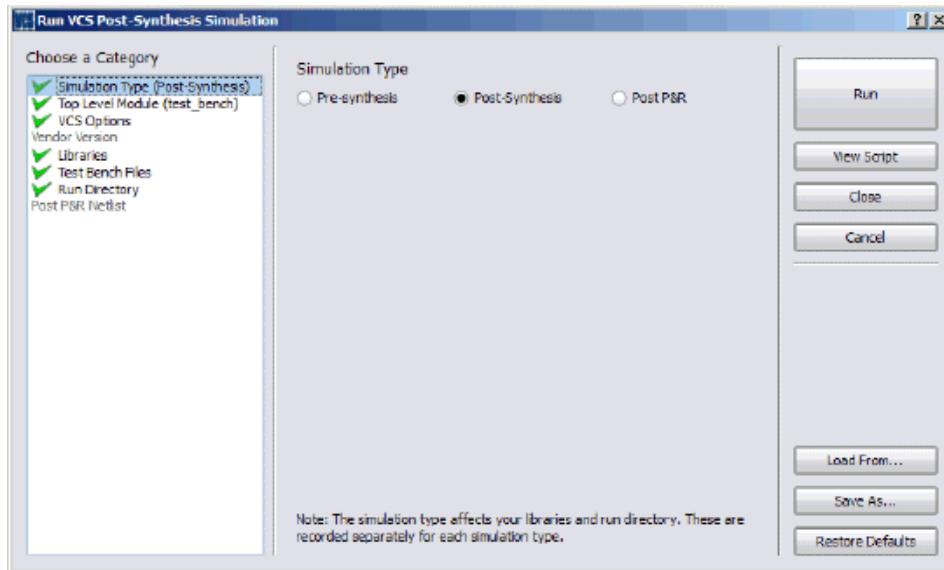
For a step-by-step procedure on setting up and launching this tool, see [Simulating with the VCS Tool, on page 1174](#) in the *User Guide*.

The Run VCS *Simulation Type* Simulation dialog box contains unique pages for specific tasks, such as specifying simulation type, VCS options, and libraries or test bench files. From this dialog box:

- Choose a category, which simplifies the data input for each task.
- A task marked with (✓) means that data has automatically been filled in; however, an (●) requires that data must be filled in.
- You are prompted to save, after canceling changes made in the dialog box.

## Simulation Type

The following dialog box displays the Simulation Type task.



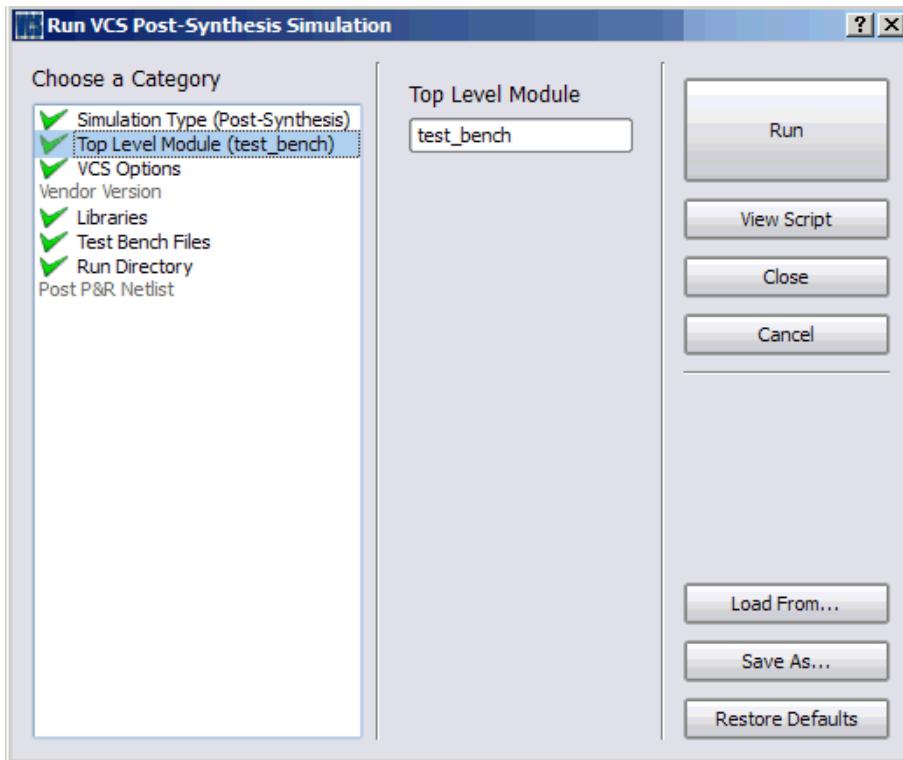
The Run VCS Simulator dialog box contains the following options:

| Command                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Choose a Category<br>Simulation Type  | Select Simulation Type and choose the type of simulation to run: <ul style="list-style-type: none"> <li>• Pre-synthesis - RTL simulation</li> <li>• Post-synthesis - Post-synthesis netlist simulation</li> <li>• Post-P&amp;R - Post-P&amp;R netlist simulation</li> </ul> See <a href="#">Simulation Type , on page 508</a> to view the dialog box.                                                                                                                                                                                                                                                                                                      |
| Choose a Category<br>Top Level Module | Select Top Level Module and specify the top-level VCS module or modules for simulation. You can use any combination of the semi-colon (;), comma (,), or a space to separate multiple top-level modules.<br>See <a href="#">Top Level Module , on page 512</a> to view the dialog box.                                                                                                                                                                                                                                                                                                                                                                     |
| Choose a Category<br>VCS Options      | Select VCS Options and specify options for each VCS step: <ul style="list-style-type: none"> <li>• Verilog compiler - VLOGAN command options for compiling and analyzing Verilog, such as the -q option.</li> <li>• VHDL compiler - VHDLAN options for compiling and analyzing VHDL.</li> <li>• Elaboration - VCS command options. The default setting is -debug_all.</li> <li>• Simulation - SIMV command options. The default setting is -gui.</li> </ul> The default settings use the FPGA version of VCS and open the VCS GUI for the debugger (DBE) and the waveform viewer.<br>See <a href="#">VCS Options , on page 513</a> to view the dialog box. |
| Choose a Category<br>Vendor Version   | For Xilinx devices, you can select Vendor Version and specify the Xilinx ISE version (10.3 or 11.2) to use.<br>See <a href="#">Vendor Version , on page 513</a> to view the dialog box.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Choose a Category<br>Libraries        | Select Libraries and specify library files typically used for Post-synthesis or Post-P&R simulation. These library files are automatically populated in the display window. You can choose to: <ul style="list-style-type: none"> <li>• Add a library</li> <li>• Edit the selected library</li> <li>• Remove the selected library</li> </ul> See <a href="#">Libraries , on page 514</a> and <a href="#">Changing Library and Test Bench Files , on page 516</a> for more information.                                                                                                                                                                     |

| Command                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Choose a Category<br>Test Bench Files | Select Test Bench Files and specify the test bench files typically used for Post-synthesis or Post-P&R simulation. These test bench files are automatically populated in the display window. You can choose to: <ul style="list-style-type: none"> <li>• Add a test bench file</li> <li>• Edit the selected test bench file</li> <li>• Remove the selected test bench file</li> </ul> See <a href="#">Test Bench Files , on page 514</a> and <a href="#">Changing Library and Test Bench Files , on page 516</a> for more information. |
| Choose a Category<br>Run Directory    | Select Run Directory and specify the results directory to run the VCS simulation.<br><br>See <a href="#">Run Directory , on page 515</a> to view the dialog box.                                                                                                                                                                                                                                                                                                                                                                       |
| Choose a Category<br>Post P&R Netlist | Select Post P&R Netlist and specify the post place-and-route netlist to run the VCS simulation.<br><br>See <a href="#">Post P&amp;R Netlist , on page 515</a> to view the dialog box.                                                                                                                                                                                                                                                                                                                                                  |
| Run                                   | Runs VCS simulation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| View Script                           | View the script file with the specified VCS commands and options before generating it. For an example, see <a href="#">VCS Script File , on page 517</a> .                                                                                                                                                                                                                                                                                                                                                                             |
| Load From                             | Use this option to load an existing VCS script.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Save As                               | Generates the VCS script. The tool generates the XML script in the directory specified.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Restore Defaults                      | Restores all the default VCS settings.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

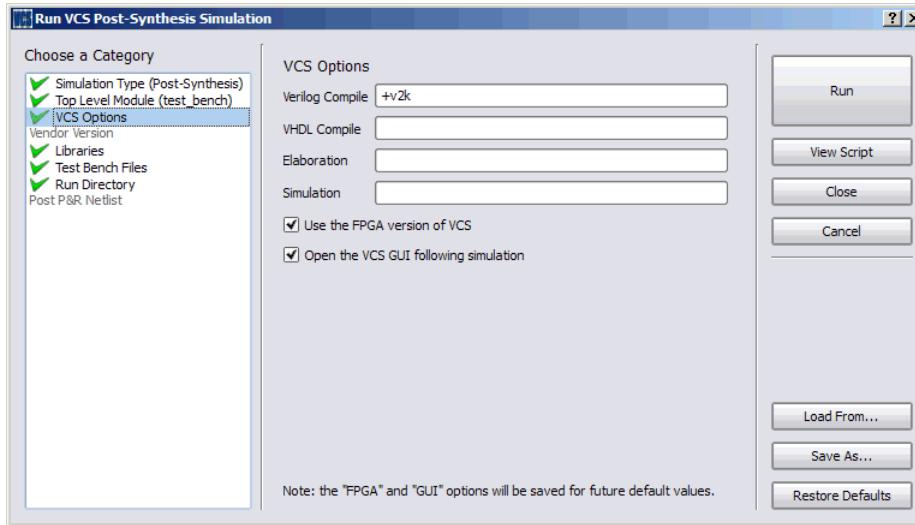
## Top Level Module

The following dialog box displays the Top Level Module task.



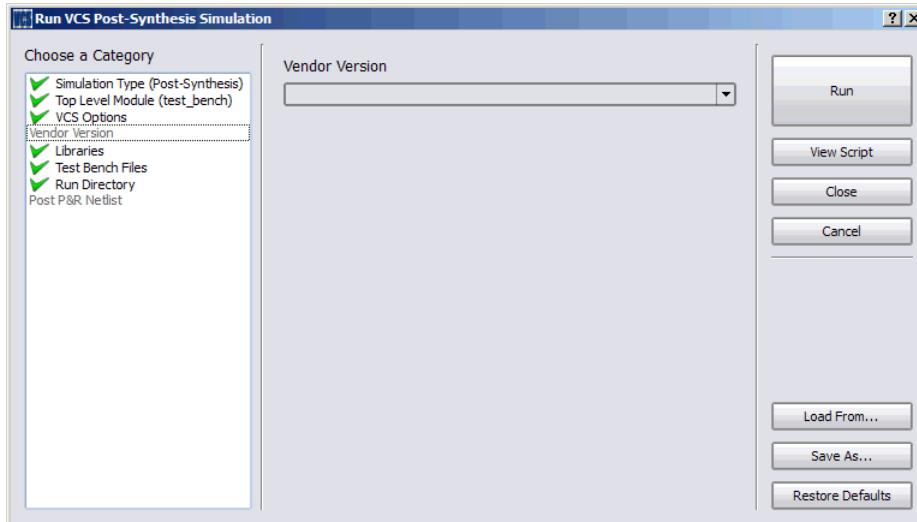
## VCS Options

The following dialog box displays the VCS Options task.



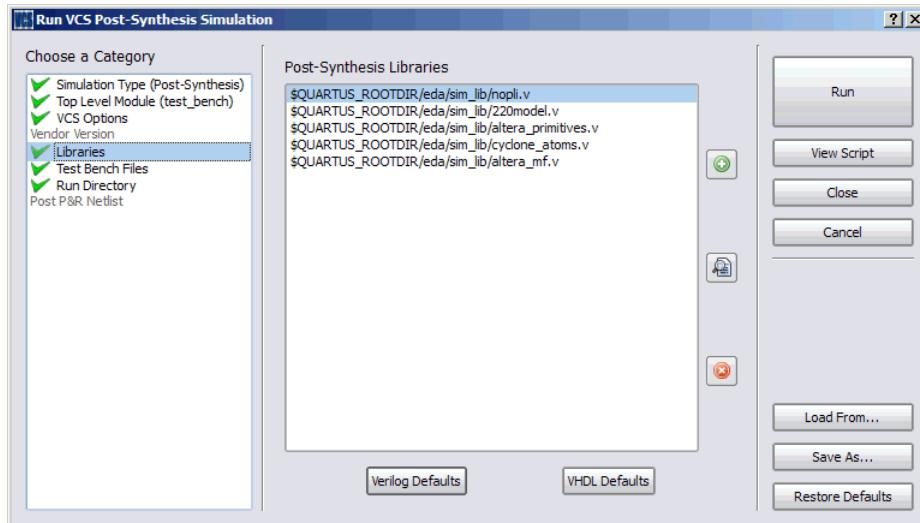
## Vendor Version

The following dialog box displays the Vendor Versions task.



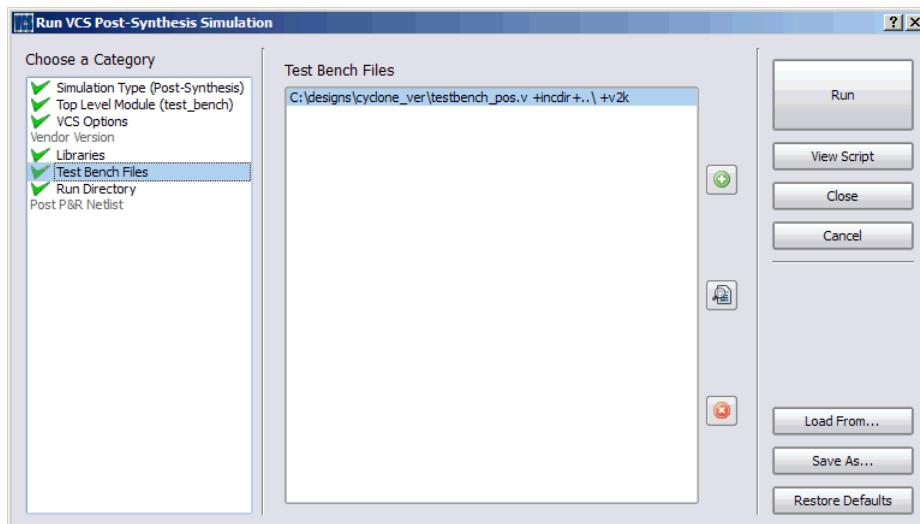
## Libraries

The following dialog box displays the Libraries task.



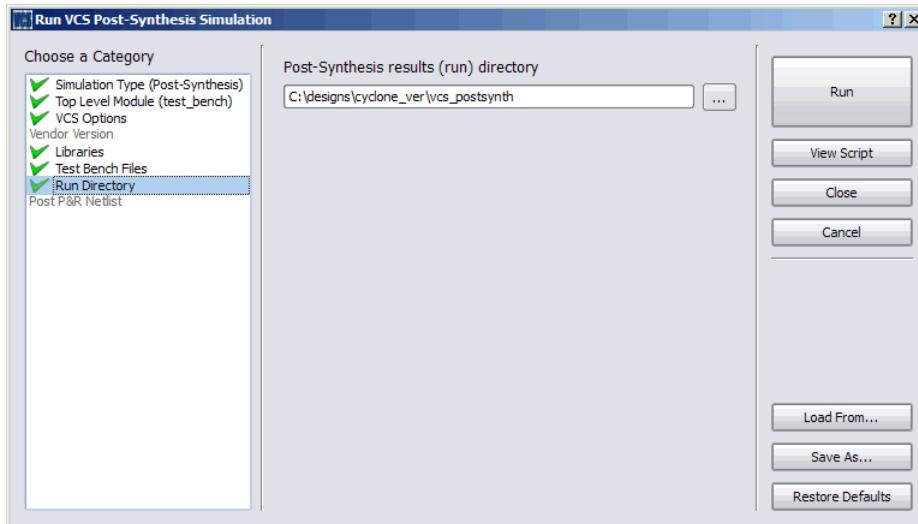
## Test Bench Files

The following dialog box displays the Test Bench Files task.



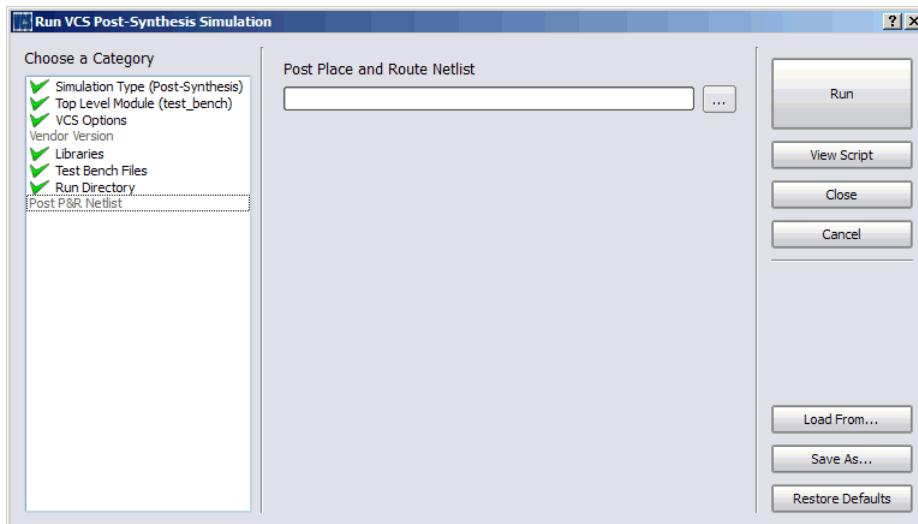
## Run Directory

The following dialog box displays the Run Directory task.



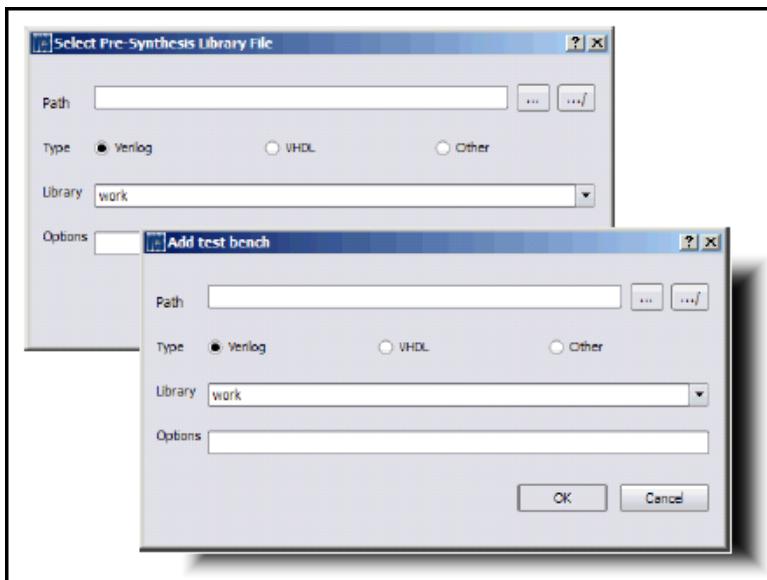
## Post P&R Netlist

The following dialog box displays the Post P&R Netlist task.

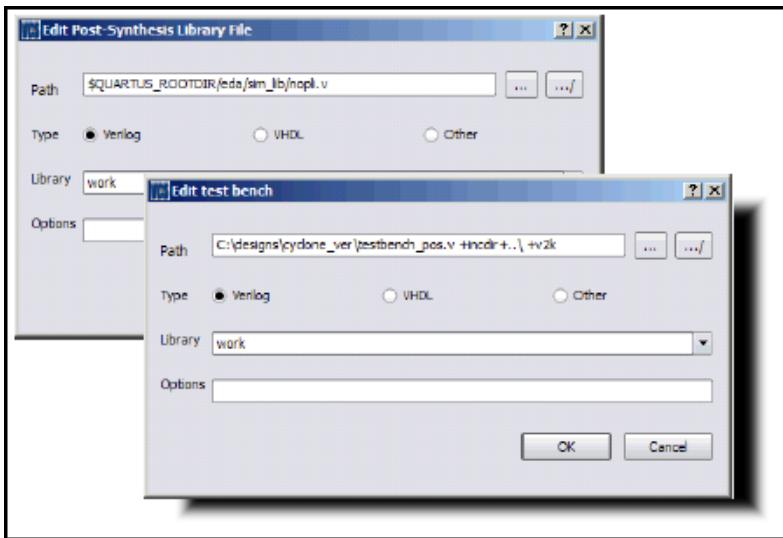


## Changing Library and Test Bench Files

You can add Post-synthesis or Post place-and-route library files and test bench files before you launch the VCS simulator. For example, specify options on the following dialog box.

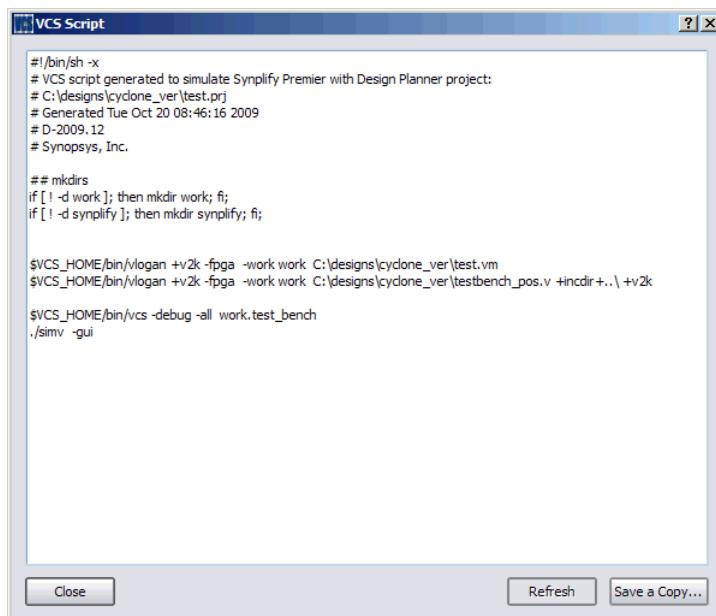


You can also edit library files and test bench files before you launch the VCS simulator. For example: specify options on the following dialog box.



## VCS Script File

When you select the VCS Script button on the Run VCS Simulator dialog box, you can view the VCS script generated by the synthesis software for this VCS run. You can also save this VCS script to a file by clicking Save a Copy.



The screenshot shows a Windows application window titled "VCS Script". The window contains a text area with a generated VCS script. The script includes comments indicating it was generated for a Synplify Premier project with Design Planner, dated Tuesday October 20, 2009, at 08:46:16. It uses shell commands like '#!/bin/sh -x' and 'mkdirs' to set up work and synplify directories. It also includes Verilog compilation commands using vlogan and vlog, and a simulation command using ./simv -gui.

```
#!/bin/sh -x
VCS script generated to simulate Synplify Premier with Design Planner project:
C:\designs\cyclone_ver\test.pj
Generated Tue Oct 20 08:46:16 2009
D-2009.12
Synopsys, Inc.

mkdirs
if [! -d work]; then mkdir work; fi;
if [! -d synplify]; then mkdir synplify; fi;

$VCS_HOME/bin/vlogan +v2k -fpga -work work C:\designs\cyclone_ver\test.vm
$VCS_HOME/bin/vloga +v2k -fpga -work work C:\designs\cyclone_ver\testbench_pos.v +includer..\\+v2k

$VCS_HOME/bin/vcs -debug -all work.test_bench
./simv -gui
```

## Analysis Menu

When you synthesize a design, a default timing report is automatically written to the log file (*projectName.srr*), located in the results directory. This report provides a clock summary, I/O timing summary, and detailed critical path information for the design. However, you can also generate a custom timing report that provides more information than the default report (specific paths or more than five paths) or one that provides timing based on additional analysis constraint files without rerunning synthesis.

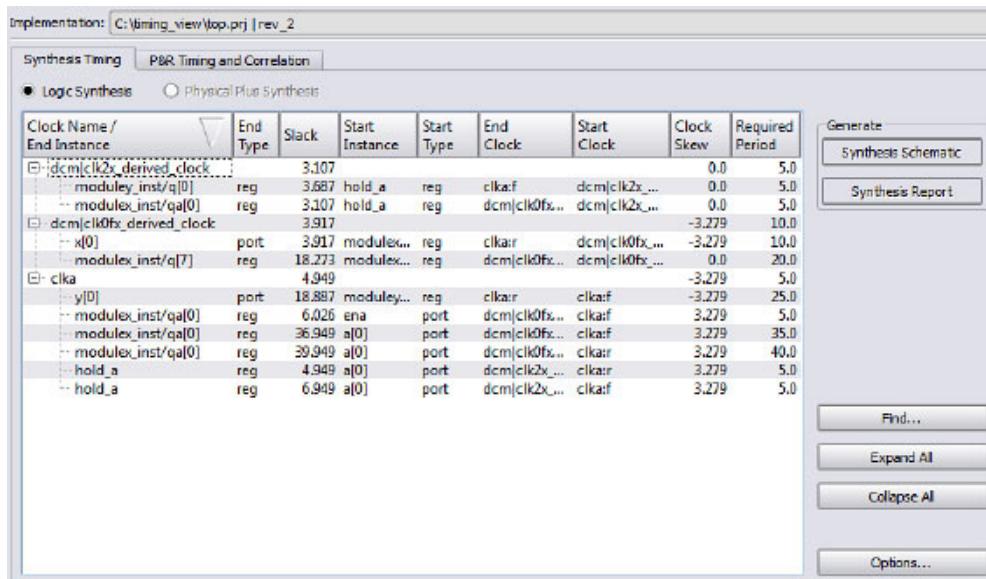
| Command                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Timing Report View           | <p><i>Synplify Premier</i></p> <p>The Timing Report View displays the synthesis timing report summary and lets you compare synthesis timing results with place-and-route (P&amp;R) Static Timing Analysis (STA) results and determine if paths for timing end points, start points, and requested periods match. See <a href="#">Timing Report View , on page 520</a>.</p>                                                                                                                                                                                                            |
| Timing Analyst               | <p><i>Synplify Pro, Synplify Premier</i></p> <p>Displays the Timing Report Generation dialog box to specify parameters for a stand-alone customized report. See <a href="#">Timing Report Generation Parameters , on page 530</a> for information on setting these options, and <a href="#">Analyzing Timing in Schematic Views, on page 474</a> in the <i>User Guide</i> for more information.</p> <p>If you click OK in the dialog box, the specified parameters are saved to a file. To run the report, click Generate. The report is created using your specified parameters.</p> |
| Generate Timing              | <p>Generates and displays a report using the timing option parameters specified above. See the following:</p> <ul style="list-style-type: none"> <li>• <a href="#">Generating Custom Timing Reports with STA</a>, on page 482 for specifics on how to run this report.</li> <li>• <a href="#">Timing Report Generation Parameters , on page 530</a> for information on setting parameters for the report. This includes information on filtering and options for running backannotation data and power consumption reports.</li> </ul>                                                |
| Generate Congestion Analysis | Generates a congestion report for the active implementation. See <a href="#">Generate Congestion Analysis , on page 541</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Timing Report View

### Synplify Premier

Use the Analysis->Timing Report View command or its associated icon (  ) or the Timing Report View link in the Project Status to display the Timing Report View tool. The Timing Report View displays the synthesis timing report summary and the P&R timing and correlation report. This tool lets you compare synthesis timing results with place-and-route (P&R) Static Timing Analysis (STA) results and determine if paths for timing end points, start points, and requested periods match.

For more information, see [Using the Timing Report View, on page 497](#).



## Synthesis Timing Tab

The Synthesis Timing tab allows you to view and manipulate timing results after synthesis. The following table describes the options you can set on the Synthesis Timing tab.

For Virtex 7 Series devices, see [Synthesis Timing and Correlation Tab, on page 522](#).

The Synthesis Timing and P&R Timing and Correlation tabs are mutually exclusive. The panel in front is active and uses the selected paths to run the options for the synthesis timing report or the P&R timing correlation report.

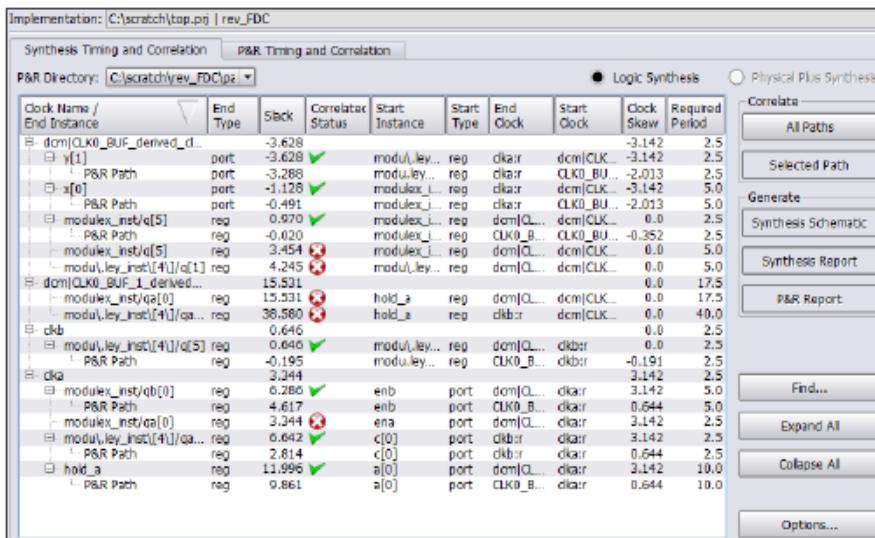
| Field                                                                                                             | Description                                                                                                                                                                                  |
|-------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Synthesis Timing Options</b>                                                                                   |                                                                                                                                                                                              |
| Logic Synthesis                                                                                                   | Specify displaying logical timing results.                                                                                                                                                   |
| Generate                                                                                                          |                                                                                                                                                                                              |
| <ul style="list-style-type: none"> <li>• Synthesis Schematic Button</li> <li>• Synthesis Report Button</li> </ul> | <ul style="list-style-type: none"> <li>• Generate the selected and filtered HDL Analyst view for the path.</li> <li>• Generate the synthesis timing report for the selected path.</li> </ul> |
| Find                                                                                                              | Locate clock and instance names in this view.                                                                                                                                                |
| Expand All                                                                                                        | Expand all paths for the specified clock or instance.                                                                                                                                        |
| Collapse All                                                                                                      | Collapse all paths for the specified clock or instance.                                                                                                                                      |
| Options                                                                                                           | Specify the number of P&R paths per end point to include in the synthesis timing report. The default is 1.<br>For details, see <a href="#">Synthesis Timing Options , on page 527</a> .      |
| <b>Synthesis Timing Report</b>                                                                                    |                                                                                                                                                                                              |
| Clock Name / End Instance                                                                                         | Clock name or name of end point instance                                                                                                                                                     |
| End Type                                                                                                          | Type of end point instance from the P&R tool that can be specified as <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                                |
| Slack                                                                                                             | Slack value                                                                                                                                                                                  |
| Start Instance                                                                                                    | Name of start point instance                                                                                                                                                                 |
| Start Type                                                                                                        | Type of start point instance from the P&R tool that can be specified as <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                              |

| Field           | Description                |
|-----------------|----------------------------|
| End Clock       | End rising/falling clock   |
| Start Clock     | Start rising/falling clock |
| Clock Skew      | Clock skew value           |
| Required Period | Requested period           |

## Synthesis Timing and Correlation Tab

For Xilinx 7 Series projects, the Timing Report View runs Vivado place and route. You can access place-and-route timing paths that can be compared and correlated with their corresponding synthesis paths. When the synthesis path correlates to a place-and-route path, the P&R path is displayed below the synthesis path.

For all other devices, see [Synthesis Timing Tab, on page 520](#).



The following table describes the options you can set on the Synthesis Timing and Correlation tab.

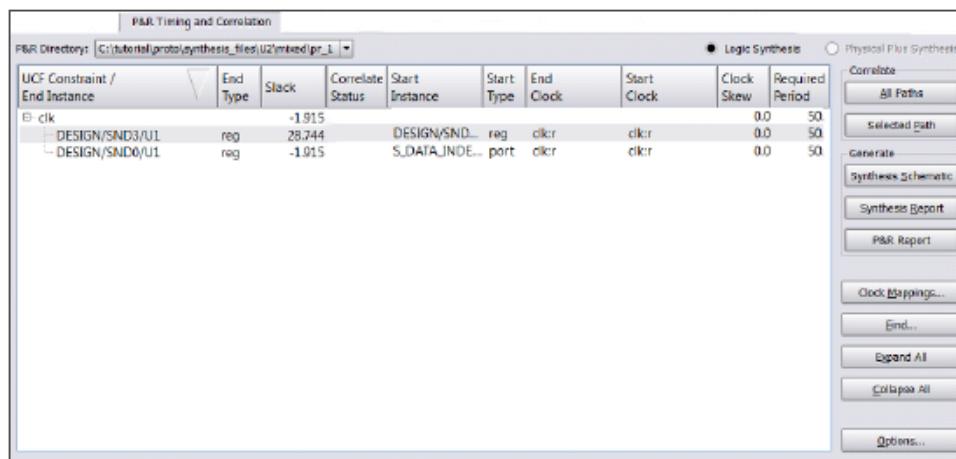
The Synthesis Timing and Correlation and P&R Timing and Correlation tabs are mutually exclusive. The panel in front is active and uses the selected paths to run the options for the synthesis timing correlation report or the P&R timing correlation report.

| Field                           | Description                                                                                                                                                                             |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Synthesis Timing Options</b> |                                                                                                                                                                                         |
| Logic Synthesis                 | Specify displaying logical timing results.                                                                                                                                              |
| Correlate                       | Run the timing correlation report for all paths or only selected path(s).                                                                                                               |
| • All Paths Button              |                                                                                                                                                                                         |
| • Selected Path Button          |                                                                                                                                                                                         |
| Generate                        |                                                                                                                                                                                         |
| • Synthesis Schematic Button    | • Generate the selected and filtered HDL Analyst view for the path.                                                                                                                     |
| • Synthesis Report Button       | • Generate the synthesis timing report for the selected path.                                                                                                                           |
| • P&R Report Button             | • Display the P&R report file ( <code>twr</code> ) for the selected path.                                                                                                               |
| Find                            | Locate clock and instance names in this view.                                                                                                                                           |
| Expand All                      | Expand all paths for the specified clock or instance.                                                                                                                                   |
| Collapse All                    | Collapse all paths for the specified clock or instance.                                                                                                                                 |
| Options                         | Specify the number of P&R paths per end point to include in the synthesis timing report. The default is 1.<br>For details, see <a href="#">Synthesis Timing Options , on page 527</a> . |
| <b>Synthesis Timing Report</b>  |                                                                                                                                                                                         |
| Clock Name / End Instance       | Clock name or name of end point instance                                                                                                                                                |

| Field             | Description                                                                                                                                                                                                                                                                              |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| End Type          | Type of end point instance from the P&R tool that can be specified as <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                                                                                                                            |
| Slack             | Slack value                                                                                                                                                                                                                                                                              |
| Correlated Status | Status of timing report paths: <ul style="list-style-type: none"> <li>• Green Check Mark- Paths passed correlation</li> <li>• Red X Mark- Paths failed correlation - Float over the cell for a tool tip describing the error.</li> <li>• Gray - Correlation not run for paths</li> </ul> |
| Start Instance    | Name of start point instance                                                                                                                                                                                                                                                             |
| Start Type        | Type of start point instance from the P&R tool that can be specified as <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                                                                                                                          |
| End Clock         | End rising/falling clock                                                                                                                                                                                                                                                                 |
| Start Clock       | Start rising/falling clock                                                                                                                                                                                                                                                               |
| Clock Skew        | Clock skew value                                                                                                                                                                                                                                                                         |
| Required Period   | Requested period                                                                                                                                                                                                                                                                         |

## P&R Timing Correlation Tab

The P&R Timing Correlation tab allows you to set options for timing.



The following table describes the options you can set on the P&R Timing and Correlation tab.

The Synthesis Timing/Synthesis Timing and Correlation and P&R Timing and Correlation tabs are mutually exclusive. The panel in front is active and uses the selected paths to run the options for the synthesis timing report or the P&R timing correlation report.

| Field                                                                                                                                              | Description                                                                                                                                                                                                                                                          |
|----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>P&amp;R Timing Options</b>                                                                                                                      |                                                                                                                                                                                                                                                                      |
| P&R Directory                                                                                                                                      | Select the P&R directory for an implementation or other project.                                                                                                                                                                                                     |
| Logic Synthesis                                                                                                                                    | Specify displaying logical timing results.                                                                                                                                                                                                                           |
| Correlate                                                                                                                                          | Run the timing correlation report for all paths or only selected path(s).                                                                                                                                                                                            |
| <ul style="list-style-type: none"> <li>• All Paths Button</li> <li>• Selected Path Button</li> </ul>                                               |                                                                                                                                                                                                                                                                      |
| Generate                                                                                                                                           |                                                                                                                                                                                                                                                                      |
| <ul style="list-style-type: none"> <li>• Synthesis Schematic Button</li> <li>• Synthesis Report Button</li> <li>• P&amp;R Report Button</li> </ul> | <ul style="list-style-type: none"> <li>• Generate the selected and filtered HDL Analyst view for the path.</li> <li>• Generate the synthesis timing report for the selected path.</li> <li>• Display the P&amp;R report file (twr) for the selected path.</li> </ul> |

---

|                |                                                                                                                                                                                                                                                         |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Clock Mappings | The Clock Correlation table links clock names in synthesis to clock names in P&R. The table is populated after Clock Rules is run. The synthesis clock is a clock alias name that appears in the synthesis log file (.srr) or timing report file (.ta). |
| Find           | Locate clock and instance names in this view.                                                                                                                                                                                                           |
| Expand All     | Expand all paths for the specified clock or instance.                                                                                                                                                                                                   |
| Collapse All   | Collapse all paths for the specified clock or instance.                                                                                                                                                                                                 |
| Options        | <p>Set P&amp;R timing correlation options.</p> <p>For more information, see <i>P&amp;R Timing and Correlation Options</i>, on page 528.</p>                                                                                                             |

---

**P&R Timing Report**

|                               |                                                                                                                                                                                                                                                                                          |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UCF Constraint / End Instance | Paths listed by end point for each UCF Constraint.                                                                                                                                                                                                                                       |
| End Type                      | Type of end point instance from the P&R tool that can be specified as: <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                                                                                                                           |
| Slack                         | Slack value                                                                                                                                                                                                                                                                              |
| Correlated Status             | Status of timing report paths: <ul style="list-style-type: none"> <li>• Green Check Mark- Paths passed correlation</li> <li>• Red X Mark- Paths failed correlation - Float over the cell for a tool tip describing the error.</li> <li>• Gray - Correlation not run for paths</li> </ul> |
| Start Instance                | Name of the start point instance                                                                                                                                                                                                                                                         |
| Start Type                    | Type of start point instance from the P&R tool that can be specified as: <ul style="list-style-type: none"> <li>• reg</li> <li>• port</li> </ul>                                                                                                                                         |
| End Clock                     | End rising/falling clock                                                                                                                                                                                                                                                                 |
| Start Clock                   | Start rising/falling clock                                                                                                                                                                                                                                                               |
| Clock Skew                    | Clock skew value                                                                                                                                                                                                                                                                         |
| Required Period               | Requested period                                                                                                                                                                                                                                                                         |

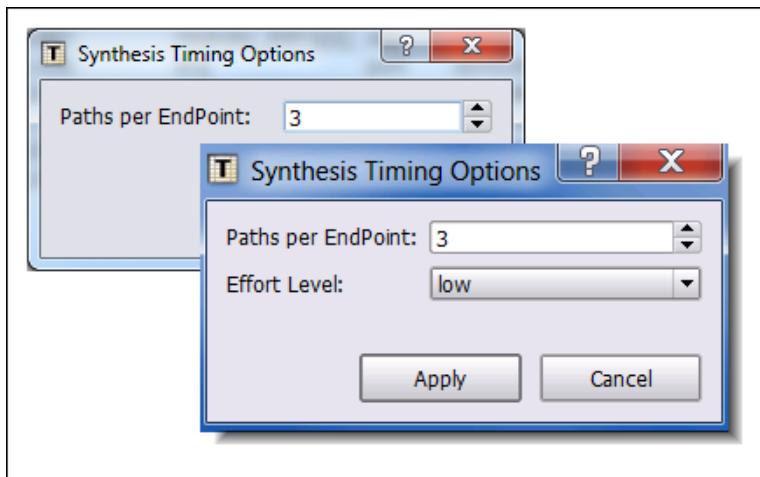
---

## Synthesis Timing Options

The Synthesis Timing Options dialog box lets you specify the number of paths per end point to include in the synthesis timing report.

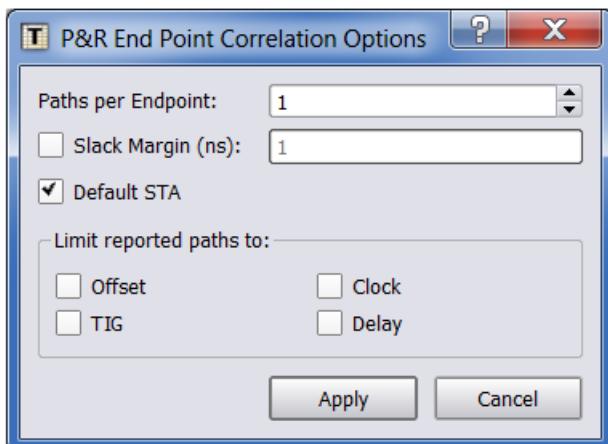
For the Synthesis Timing and Correlation that correlates to place and route results, you can specify the following effort levels:

- low - Runs a default timing report for Vivado (no -from/-to filtering). From this report a first-pass correlation is run. For the paths that did not correlate, all -from/-to filters are specified as a collection of uncorrelated start and end points and is run through a second-pass timing report. You can correlate the remaining paths individually, by clicking the Selected Path button under Correlate.
- medium - Performs a low effort run, but continues to run the second-pass correlation until there is no further convergence of results. This occurs when the number of correlation failures does not change from the previous run.
- high - Performs a medium effort level run. However, whenever convergence stops, it then continues running correlation on the remaining individual paths until all have been attempted. Be aware that effort level high can incur large runtimes, because Vivado is called repeatedly.



## P&R Timing and Correlation Options

The P&R End Point Correlation Options dialog box lets you specify the options to narrow the scope of paths included in the timing correlation report.



Enable any of the options in the following table:

| Options            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Paths per Endpoint | Specify the number of P&R paths per end point to include in the synthesis timing report. The default is 1.                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Slack Margin       | This option directs the timing comparison routines to compare the reported slack for a P&R path to the same path in synthesis, if found. When a P&R path successfully correlates to a synthesis path, this option further checks that reported slacks are within the given range (in nanoseconds) of each other. When enabled, slacks that are within the given range show up as green and slacks out of range show up as red. Use the tool tip to fly over red slacks with the cursor to see a report of the slack found in synthesis. |

For the following *Xilinx only* options, if none of them are selected, all options are active. Select them individually to narrow the scope of paths (based on UCF constraint) included in the End Point/Start Point Table.

|        |                                                     |
|--------|-----------------------------------------------------|
| Clock  | Include TIMESPEC...PERIOD constraints in the table. |
| Offset | Include OFFSET constraints in the table.            |

---

Default STA      Use the default timing report from P&R (*top.twr*) to populate the End Point/Start Point Table. Use the option early on to verify the P&R critical paths in a design. This is the default.

---

TIG      Include TIMESPEC...TIG constraints in the table.

---

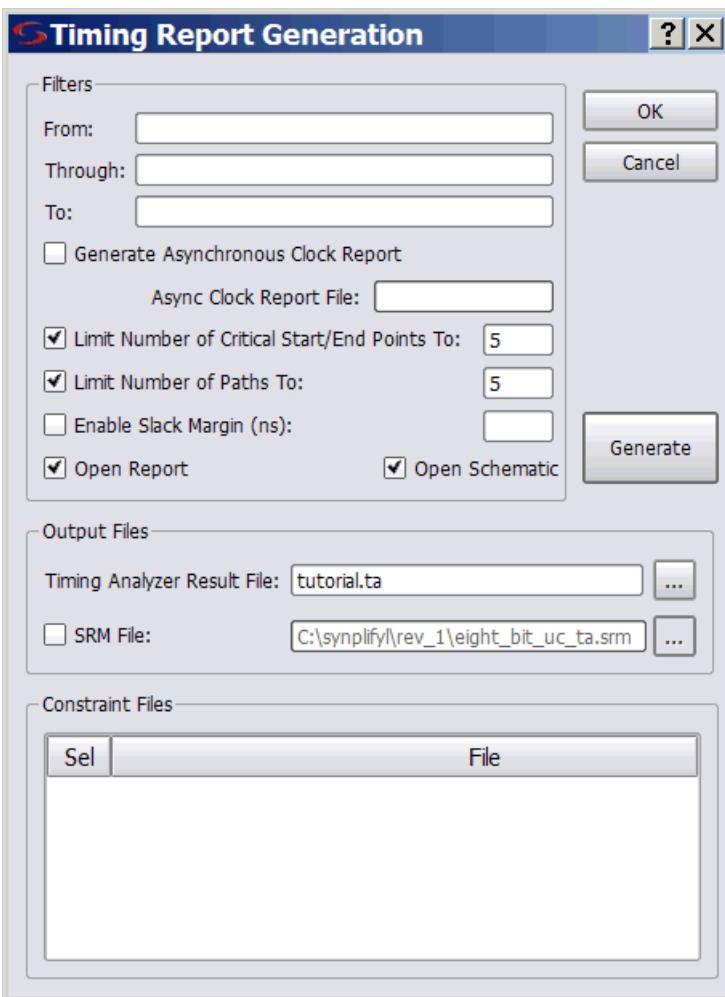
Delay      Include TIMESPEC...*delay* constraints in the table.

---

## Timing Report Generation Parameters

*Synplify Pro, Synplify Premier*

You can use the Analysis->Timing Analyst command to specify parameters for a stand-alone timing report. See [Timing Reports, on page 205](#) for information on the file contents.



The following table provides brief descriptions of the parameters for running a stand-alone timing report.

### Timing Report Option Description

|                                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| From or To                                   | <p>Specifies the starting (From) or ending (To) point of the path for one or more objects. It must be a timing start point (From) or end (To) point for each object. Use this option in combination with the others in the Filters section of the dialog box. See <a href="#">Combining Path Filters for the Timing Analyzer</a>, on page 534 for examples of using filters.</p> <p>Tcl equivalent: <code>set_option -reporting_filter "-from {object1} -to {object2}"</code></p>                                                                                                          |
| Through                                      | <p>Reports all paths through the specified point or list of objects. See for more information on using this filter. Use this option in combination with the others in the Filters section of the dialog box. See the following for additional information:</p> <ul style="list-style-type: none"> <li>• <a href="#">Timing Analyzer Through Points</a>, on page 533</li> <li>• <a href="#">Combining Path Filters for the Timing Analyzer</a>, on page 534.</li> </ul> <p>Tcl equivalent: <code>set_option -reporting_filter "-from {object1} -to {object2} -through {object3}"</code></p> |
| Generate Asynchronous Clock Report           | <p>Generates a report for paths that cross between clock groups. Generally paths in different clock groups are automatically handled as false paths. This option provides a file that contains information on each of the paths and can be viewed in a spreadsheet. This file is in the results directory (<i>projectName_async_clk.rpt.csv</i>). For details on the report, see <a href="#">Asynchronous Clock Report</a>, on page 213.</p> <p>Tcl equivalent: <code>set_option -reporting_async_clock 0 1</code></p>                                                                     |
| Limit Number of Critical Start/End Points to | <p>Specifies the maximum number of start/end paths to display for critical paths in the design. The default is 5. Use this option in combination with the others in the Filters section of the dialog box.</p> <p>Tcl equivalent: <code>set_option -num_startend_points numberOfRowsPaths</code></p>                                                                                                                                                                                                                                                                                       |
| Limit Number of Paths to                     | <p>Specifies the maximum number of paths to report. The default is 5. Use this option in combination with the others in the Filters section of the dialog box.</p> <p>Tcl equivalent: <code>set_option -reporting_number_paths numberOfRowsPaths</code></p>                                                                                                                                                                                                                                                                                                                                |

**Timing Report Option Description**

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Enable Slack Margin (ns) | Limits the report to paths within the specified distance of the critical path. Use this option in combination with the others in the Filters section of the dialog box.<br><br>Tcl equivalent: <b>set_option -reporting_margin slackValue</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Open Report              | When enabled, clicking the Generate button opens the Text Editor on the generated custom timing report specified in the timing report file (.ta).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Open Schematic           | When enabled, clicking the Generate button opens a Technology view showing the netlist specified in the timing report netlist file (.srm).<br><br>Tcl equivalent: <b>set_option -reporting_output_srm 0 1</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Output Files             | <p>Displays the name of the generated report:</p> <ul style="list-style-type: none"> <li>Async Clock Report File contains the spreadsheet data for the asynchronous clock report. This file is not automatically opened when report generation is complete. You can locate this file in the results directory. Default name is <i>projectName_async_clk.rpt.csv</i> (name cannot be changed).</li> </ul> <p>Tcl equivalent: <b>set_option -reporting_async_clock 0 1</b></p> <ul style="list-style-type: none"> <li>Timing Analyst Results File is the standard timing report file, located in the Implementation Results directory. The file is also listed in the Project view. Default filename is <i>projectName.ta</i>.</li> </ul> <p>Tcl equivalent: <b>set_option -reporting_filename filename.ta</b></p> <ul style="list-style-type: none"> <li>SRM File updates the Technology view so that you can display the results of the timing updates in the HDL and Physical Analyst tools. The file is also listed in the Project view.</li> </ul> <p>Tcl equivalent: <b>set_option -reporting_netlist filename</b></p> <p>For more details on any of these reports, see <a href="#">Timing Reports, on page 205</a>.</p> |
| Constraint Files         | Enables analysis design constraint files (.adc) to be used for stand-alone timing analysis only. See <a href="#">Input Files, on page 182</a> for information on this file.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Generate                 | Clicking this button generates the specified timing report file and timing view netlist file (.srm) if requested, saves the current dialog box entries for subsequent use, then closes the dialog box.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Timing Analyzer Through Points

You can specify through points for nets (n:), hierarchical ports (t:), or instantiated cell pins (t:). You can specify the through points in two ways:

- OR list** Enter the points as a space-separated list. The points are treated as an OR list and paths are reported if they crosses any of the points in the list. For example, when you type the following, the tool reports paths that pass through points b or c:

```
{n:b n:c}
```

See [Filtering Points: OR List of Through Points , on page 533.](#)

- AND list** Enter the points in a product of sums (POS) format. The tool treats them as an AND list, and only reports the path if it passes through all the points in the list. The POS format for the timing report is the same as for timing constraints. The POS format is as follows:

```
{n:b n:c}, {n:d n:e}
```

This constraint translates as follows:

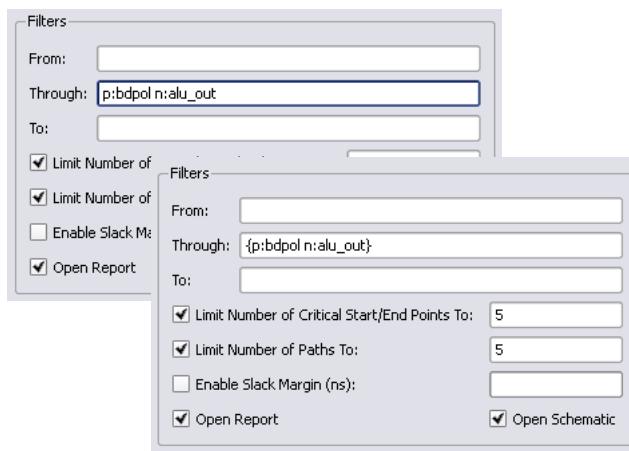
```
b AND d
OR b AND e
OR c AND d
OR c AND e
```

See [Filtering Points: AND List of Through Points , on page 534.](#)

See [Defining From/To/Through Points for Timing Exceptions, on page 221](#) in the *User Guide* for more information about specifying through points.

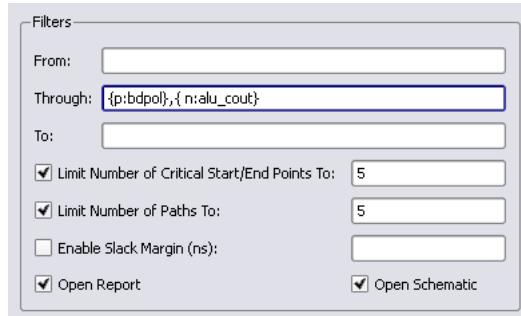
### Filtering Points: OR List of Through Points

This example reports the five worst paths through port bdpol or net aluout. You can enter the through points as a space-separated list (enclosing the list in braces is optional.)



## Filtering Points: AND List of Through Points

This example reports the five worst paths passing through port bdpol and net aluout. Enclose each list in braces {} and separate the lists with a comma.



## Combining Path Filters for the Timing Analyzer

This section describes how to use a combination of path filters to specify what you need and how to specify start and end points for path filtering.

### Number and Slack Path Filters

The Limit Number of Paths To option specifies the maximum number of paths to report and the Enable Slack Margin option limits the report to output only paths that have a slack value that is within the specified value. When you use these

two options together, the tighter constraint applies, so that the actual number of paths reported is the minimum of the option with the smallest value. For example, if you set the number of paths to report to 10 and the slack margin for 1 ns, if the design has only five paths within 1 ns of critical, then only five paths are reported (not the 10 worst paths). But if, for example, the design has 15 paths within a 1 ns of critical, only the first 10 are reported.

## From/To/Through Filters

You can specify the from/to points for a path. You can also specify just a from point or just a to point. The from and to points are one or more hierarchical names that specify a port, register, pin on a register, or clock as object (clock alias). Ports and instances can have the same names, so prefix the name with p: for top-level port, i: for instance, or t: for hierarchical port or instance pin. However, the c: prefix for clocks is required for paths to be reported.

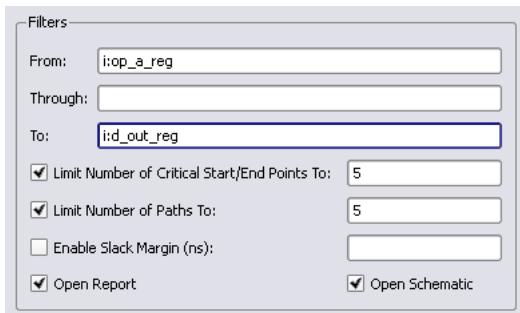
The timing analyst searches for the from/to objects in the following order: clock, port, bit port, cell (instance), net, and pin. Always use the prefix qualifier to ensure that all expected paths are reported. Remember that the timing analyst stops at the first occurrence of an object match. For buses, all possible paths from the specified start to end points are considered.

You can specify through points for nets, cell pins, or hierarchical ports.

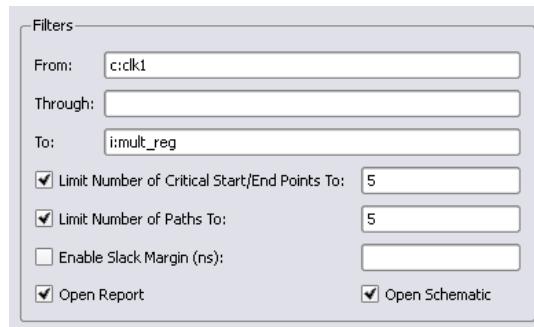
You can simply type in from/to or through points. You can also cut-and-paste or drag-and-drop valid objects from the RTL or Technology views into the appropriate fields on the Timing Report Generation dialog box. Timing analysis requires that constraints use the Tech View name space. Therefore, it is recommended that you cut-and-paste or drag-and-drop objects from the Technology view rather than the RTL view.

The following examples show how to specify start, end, or through point combinations for path filtering.

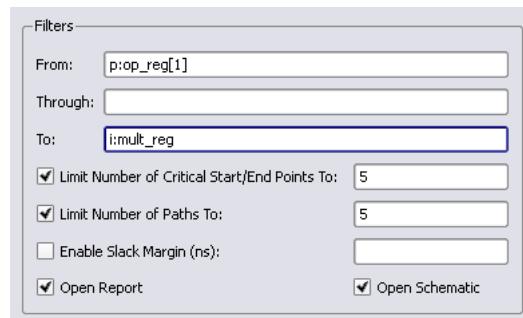
### Filtering Points: Single Register to Single Register



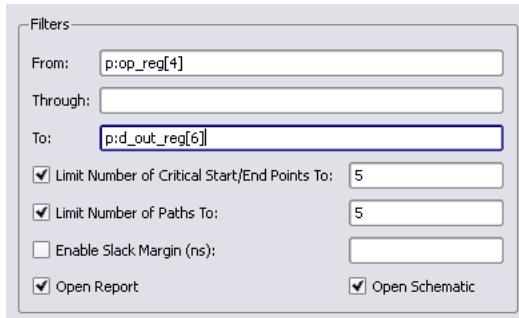
### Filtering Points: Clock Object to Single Register



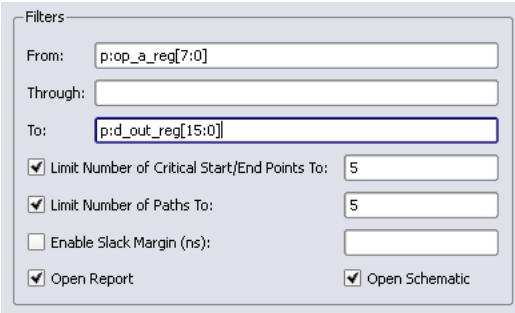
### Filtering Points: Single Bit of a Bus to Single Register



## Filtering Points: Single Bit of a Bus to Single Bit of a Bus

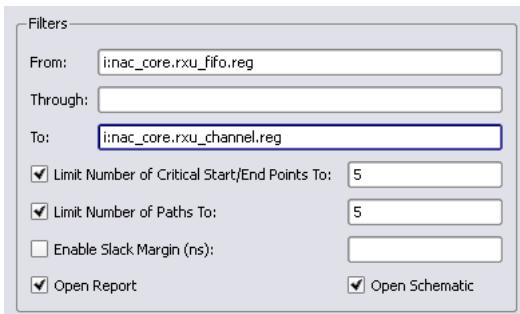


## Filtering Points: Multiple Bits of a Bus to Multiple Bits of a Bus

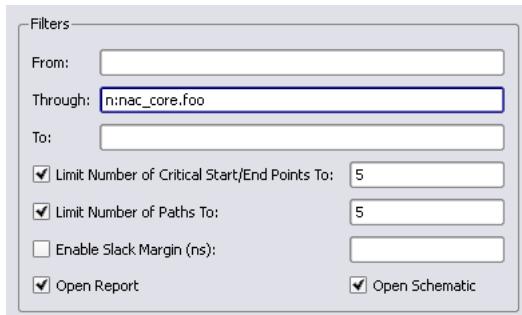


## Filtering Points: With Hierarchy

This example reports the five worst paths for the net foo:

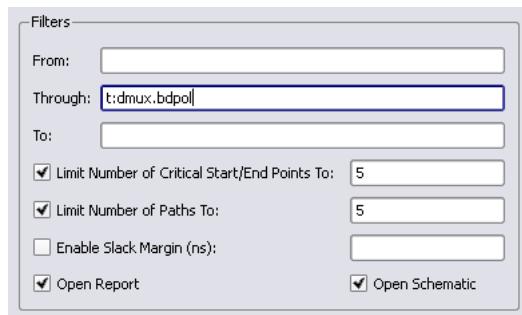


## Filtering Points: Through Point for a Net



## Filtering Points: Through Point for a Hierarchical Port

This example reports the five worst paths for the hierarchical port bdpol:



## Examples Using Wildcards

You can use the question mark (?) or asterisk (\*) wildcard characters for object searching and name substitution. These characters work the same way in the synthesis tool environment as in the Linux environment.

## The ? Wildcard

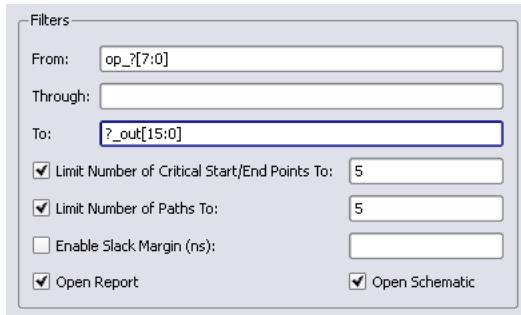
The ? matches single characters. If a design has buses op\_a[7:0], op\_b[7:0], and op\_c[7:0], and you want to filter the paths starting at each of these buses, specify the start points as op\_?[7:0]. See [Example: ? Wildcard in the Name, on page 539](#) for another example.

## The \* Wildcard

The \* matches a string of characters. In a design with buses op\_a2[7:0], op\_b2[7:0], and op\_c2[7:0], where you want to filter the paths starting at each of these objects, specify the start points as op\_\*[\*]. The report shows all paths beginning at each of these buses and for all of the bits of each bus. See [Example: \\* Wildcard in the Name \(With Hierarchy\), on page 540](#) and [Example: \\* Wildcard in the Bus Index, on page 540](#) for more examples.

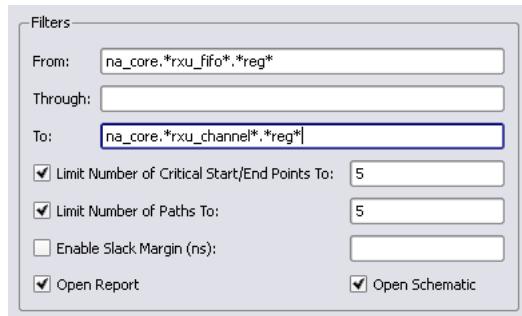
### Example: ? Wildcard in the Name

The ? is not supported in bus indices.



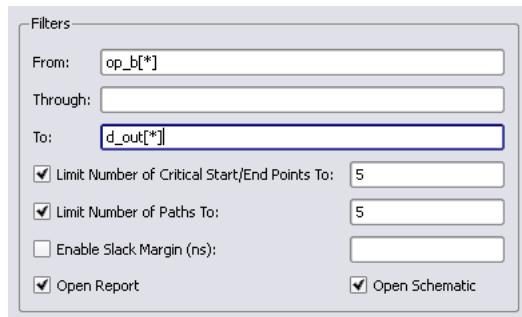
## Example: \* Wildcard in the Name (With Hierarchy)

This example reports the five worst paths, starting at block rxu\_fifo and ending at block rxu\_channel within module nac\_core. Each register in the design has the characters reg in the name.



## Example: \* Wildcard in the Bus Index

This example reports the five worst paths, starting at op\_b, and ending at d\_out, taking into account all bits on these buses.

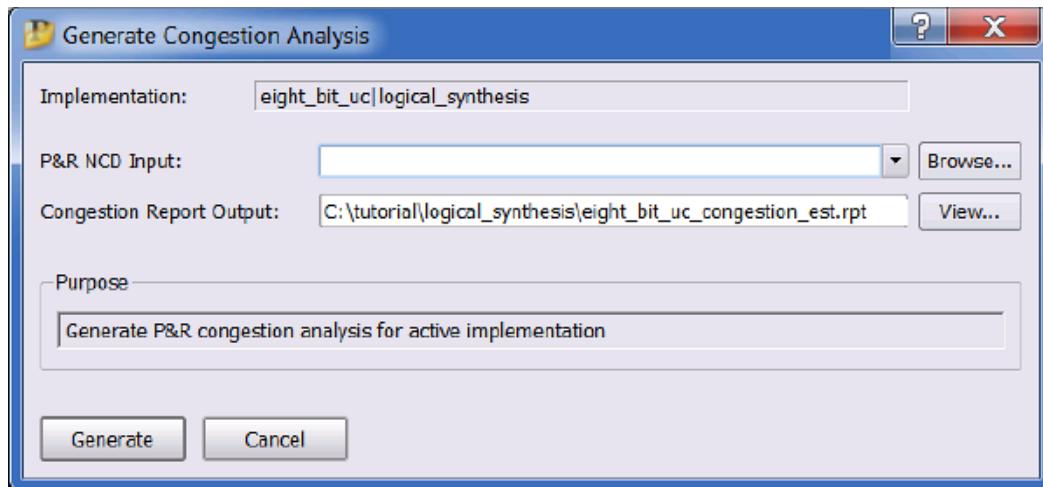


## Generate Congestion Analysis

*Synplify Premier  
Xilinx Virtex-5 and Virtex-6 Technologies*

The Generate Congestion Analysis command lets you generate a congestion report for the active implementation. This report provides congestion analysis and possible fixes for congestion problems, after running logic synthesis and Xilinx place and route when the design is fully placed. This feature can provide improved routability for the design.

For more information, see [Analyzing Congestion After Logic Synthesis, on page 330](#).



The Generate Congestion Analysis Report dialog box contains the following options:

| Command                  | Description                                                                                                                                                     |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Implementation           | Displays the current implementation.                                                                                                                            |
| P&R NCD Input            | Specifies the mapped NCD file from Xilinx place and route. You can use the pre-populated congestion report output file:<br><i>designName_congestion_est.rpt</i> |
| Congestion Report Output | Specifies the output congestion report destination location.                                                                                                    |

# HDL Analyst Menu

In the Project View, the HDL Analyst menu contains commands that provide project analysis in the following views:

- [RTL View](#)
- [Technology View](#)
- Physical Analyst (see [HDL Analyst Menu: Physical Analyst Command, on page 691](#)).

This section describes the HDL Analyst menu commands for the RTL and Technology views. Commands may be disabled, depending on the current context. Generally, the commands enabled in any context reflect those available in the corresponding popup menus. The descriptions in the table indicate when commands are context-dependent. For explanations about the terms used in the table, such as filtered and unfiltered, transparent and opaque, see [Filtered and Unfiltered Schematic Views, on page 105](#) and [Transparent and Opaque Display of Hierarchical Instances, on page 111](#). For procedures on using the HDL Analyst tool, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

For ease of use, the commands have been divided into sections that correspond to the divisions in the HDL Analyst menu.

- [HDL Analyst Menu: RTL and Technology View Submenus, on page 543](#)
- [HDL Analyst Menu: Hierarchical and Current Level Submenus, on page 544](#)
- [HDL Analyst Menu: Filtering and Flattening Commands, on page 546](#)
- [HDL Analyst Menu: Timing Commands, on page 549](#)
- [HDL Analyst Menu: Analysis Commands, on page 549](#)
- [HDL Analyst Menu: Selection Commands, on page 553](#)
- [HDL Analyst Menu: FSM Commands, on page 553](#)
- [HDL Analyst Menu: VCD Commands, on page 554](#)

## HDL Analyst Menu: RTL and Technology View Submenus

This table describes the commands that appear on the HDL Analyst->RTL and HDL Analyst->Technology submenus when the RTL or Technology View is active. For procedures on using these commands, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

| HDL Analyst Command                    | Description                                                                                                                                                                                                                                                                                        |
|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RTL->Hierarchical View                 | Opens a new, hierarchical RTL view. The schematic is unfiltered.                                                                                                                                                                                                                                   |
| RTL->Flattened View                    | Opens a new RTL view of your entire design, with a flattened, unfiltered schematic at the level of generic logic cells. See <a href="#">Usage Notes for Flattening , on page 547</a> for some usage tips.                                                                                          |
| RTL->Design Plan View                  | Displays an RTL schematic of the device and its design-planned regions in the Synplify Premier tool with Design Planner.<br>Same as double-clicking the partitioned RTL view (.srp) file. See <a href="#">Working in the Standard Schematic, on page 409</a> of the <i>User Guide</i> .            |
| Technology->Hierarchical View          | Opens a new, hierarchical Technology view. The schematic is unfiltered.                                                                                                                                                                                                                            |
| Technology->Flattened View             | Creates a new Technology view of your entire design, with a flattened, unfiltered schematic at the level of technology cells. See <a href="#">Usage Notes for Flattening , on page 547</a> for tips about flattening.                                                                              |
| Technology->Flattened to Gates View    | Creates a new Technology view of your entire design, with a flattened, unfiltered schematic at the level of Boolean logic gates. See <a href="#">Usage Notes for Flattening , on page 547</a> for tips about flattening.                                                                           |
| Technology->Hierarchical Critical Path | Creates a new Technology view of your design, with a hierarchical, <i>filtered</i> schematic showing only the instances and paths whose slack times are within the slack margin you specified in the Slack Margin dialog. This command automatically enables HDL Analyst->Show Timing Information. |

| HDL Analyst Command                 | Description                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Technology->Flattened Critical Path | <p>Creates a new Technology view of your design, with a flattened, <i>filtered</i> schematic showing only the instances and paths whose slack times are within the slack margin you specified in the Slack Margin dialog. This command automatically enables HDL Analyst-&gt;Show Timing Information.</p> <p>See <a href="#">Usage Notes for Flattening , on page 547</a> for tips about flattening.</p> |

## HDL Analyst Menu: Hierarchical and Current Level Submenus

This table describes the commands on the HDL Analyst->Hierarchical and HDL Analyst->Current Level submenus. For procedures on using these commands, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

| HDL Analyst Command                   | Description                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hierarchical->Expand                  | <p>Expands paths from selected pins and/or ports up to the nearest objects on any hierarchical level, according to pin/port directions. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.</p> <p>Successive Expand commands expand the paths further, based on the new current selection.</p> |
| Hierarchical->Expand to Register/Port | <p>Expands paths from selected pins and/or ports, in the port/pin direction, up to the next register, port, or black box. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.</p>                                                                                                               |
| Hierarchical->Expand Paths            | <p>Shows all logic, on any hierarchical level, between two or more selected instances, pins, or ports. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.</p>                                                                                                                                  |
| Hierarchical->Expand Inwards          | <p>Expands within the hierarchy of an instance, from the lower-level ports that correspond to the selected pins, to the nearest objects and no further. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.</p>                                                                                 |

| HDL Analyst Command                    | Description                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hierarchical->Goto Net Driver          | Displays the unfiltered schematic sheet that contains the net driver for the selected net. Operates hierarchically, on lower schematic levels as well as the current level.                                                                                                                                                                                                                |
| Hierarchical->Select Net Driver        | Selects the driver for the selected net. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.                                                                                                                                                                                                                       |
| Hierarchical->Select Net Instances     | Selects instances connected to the selected net. The result is a <i>filtered</i> schematic. Operates hierarchically, on lower schematic levels as well as the current level.                                                                                                                                                                                                               |
| Current Level->Expand                  | Expands paths from selected pins and/or ports up to the nearest objects on the current level, according to pin/port directions. The result is a <i>filtered</i> schematic. Limited to all sheets on the current schematic level. This command is only available if a HDL Analyst view is open.<br>Successive Expand commands expand the paths further, based on the new current selection. |
| Current Level->Expand to Register/Port | Expands paths from selected pins and/or ports, according to the pin/port direction, up to the next register, ports, or black box on the current level. The result is a <i>filtered</i> schematic. Limited to all sheets on the current schematic level.                                                                                                                                    |
| Current Level->Expand Paths            | Shows all logic on the current level between two or more selected instances, pins, or ports. The result is a <i>filtered</i> schematic. Limited to the current schematic level (all sheets).                                                                                                                                                                                               |
| Current Level->Goto Net Driver         | Displays the unfiltered schematic sheet that contains the net driver for the selected net. Limited to all sheets on the current schematic level.                                                                                                                                                                                                                                           |
| Current Level->Select Net Driver       | Selects the driver for the selected net. The result is a <i>filtered</i> schematic. Limited to all sheets on the current schematic level.                                                                                                                                                                                                                                                  |
| Current Level->Select Net Instances    | Selects instances on the current level that are connected to the selected net. The result is a <i>filtered</i> schematic. Limited to all sheets on the current schematic level.                                                                                                                                                                                                            |

## HDL Analyst Menu: Filtering and Flattening Commands

This table describes the filtering and flattening commands on the HDL Analyst menu. For procedures on filtering and flattening, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

### HDL Analyst Command Description

|                                                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  Filter Schematic | <p>Filters your entire design to show only the selected objects. The result is a <i>filtered</i> schematic. For more information about using this command, see <a href="#">Filtering Schematics, on page 456</a> of the <i>User Guide</i>.</p> <p>This command is only available with an open HDL Analyst view.</p>                                                                                                                                                                                                                                                                                                                                                                                                                |
| Flatten Current Schematic<br>(Unfiltered Schematic)                                                | <p>In an unfiltered schematic, the command flattens the current schematic, at the current level and all levels below. In an RTL view, the result is at the generic logic level. In a Technology view, the result is at the technology-cell level. See the next table entry for information about flattening a filtered schematic.</p> <p>This command does not do the following:</p> <ul style="list-style-type: none"><li>• Flatten your entire design (unless the current level is the top level)</li><li>• Open a new view window</li><li>• Take into account the number of Dissolve Levels defined in the Schematic Options dialog box</li></ul> <p>See <a href="#">Usage Notes for Flattening , on page 547</a> for tips.</p> |

**HDL Analyst Command Description**

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Flatten Current Schematic<br>(Filtered Schematic) | In a filtered schematic, flattening is a two-step process: <ul style="list-style-type: none"><li>Only unhidden transparent instances (including nested ones) are flattened in place, in the context of the entire design. Opaque and hidden hierarchical instances remain hierarchical. The effect of this command is that all hollow boxes with pale yellow borders are removed from the schematic, leaving only what was displayed inside them.</li><li>The original filtering is restored.</li></ul> In an RTL view, the result is at the generic logic level. In a Technology view, the result is at the technology-cell level. This command does not do the following: <ul style="list-style-type: none"><li>Flatten everything inside a transparent instance. It only flattens transparent instances and any nested transparent instances they contain.</li><li>Open a new view window.</li><li>Take into account the number of Dissolve Levels defined in the Schematic Options dialog box.</li></ul> See <a href="#">Usage Notes for Flattening , on page 547</a> for usage tips. |
| Unflatten Current Schematic                       | Undoes any flattening operations and returns you to the original schematic, as it was before flattening and any filtering.<br><br>This command is available only if you have explicitly flattened a hierarchical schematic using HDL Analyst->Flatten Current Schematic, for example. It is not available for flattened schematics created directly with the RTL and Technology submenus of the HDL Analyst menu.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Usage Notes for Flattening

It is usually more memory-efficient to flatten only parts of your design, as needed. The following are a few tips for flattening designs with different commands. For detailed procedures, see [Flattening Schematic Hierarchy, on page 463](#) of the *User Guide*.

## RTL/Technology->Flattened View Commands

- Use Flatten Current Schematic to flatten only the current hierarchical level and below.
- Flatten selected hierarchical instances with Dissolve Instances (followed by Flatten Current Schematic, if the schematic is filtered).
- To make hierarchical instances transparent without flattening them, use Dissolve Instances in a filtered schematic. This shows their details nested inside the instances.

### Flatten Current Schematic Command (Unfiltered View)

- Flatten selected hierarchical instances with Dissolve Instances.
- To see the lower-level logic inside a hierarchical instance, push into it instead of flattening.
- Selectively flatten your design by hiding the instances you do not need, flattening, and then unhiding the instances.
- Flattening erases the history of displayed sheets for the current view. You can no longer use View->Back. You can, however, use UnFlatten Schematic to get an unflattened view of the design.

### Flatten Current Schematic Command (Filtered View)

- Flatten selected hierarchical instances with Dissolve Instances, followed by Flatten Current Schematic.
- Selectively flatten your design by hiding the instances you do not need, flattening, and then unhiding the instances.
- Flattening erases the history of displayed sheets for the current view. You can no longer use View->Back. You can do the following:
  - Use View->Back for a view of the transparent instance flattened in the context of the entire design. This is the view generated after step 1 of the two-step flattening process described above. Use UnFlatten Schematic to get an unflattened view of the design.

## HDL Analyst Menu: Timing Commands

This table describes the timing commands on the HDL Analyst menu. For procedures on using the timing commands, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

| HDL Analyst Command                                                                                  | Description                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Set Slack Margin                                                                                     | Displays the Slack Margin dialog box, where you set the slack margin. HDL Analyst->Show Critical Path displays only those instances whose slack times are worse than the limit set here. Available only in a Technology view.                                                                                                                                                                                                        |
|  Show Critical Path | Filters your entire design to show only the instances and paths whose slack times exceed the slack margin set with Set Slack Margin, above. The result is flat if the entire design was already flat. This command also enables Show Timing Information (see below). Available only in a Technology view.                                                                                                                            |
| Show Timing Information                                                                              | When enabled, Technology view schematics are annotated with timing numbers above each instance. The first number is the cumulative path delay; the second is the slack time of the worst path through the instance. Negative slack indicates that timing has not met requirements. Available only in a Technology view. For more information, see <a href="#">Viewing Timing Information, on page 474</a> on the <i>User Guide</i> . |

## HDL Analyst Menu: Analysis Commands

This table describes the analysis commands on the HDL Analyst menu. For procedures on using the analysis commands, see [Analyzing With the Standard HDL Analyst Tool, on page 452](#) of the *User Guide*.

| HDL Analyst Command | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Isolate Paths       | <p>Filters the current schematic to display only paths associated with all the pins of the selected instances. The paths follow the pin direction (from output to input pins), up to the next register, black box, port, or hierarchical instance.</p> <p>If the selected objects include ports and/or pins on unselected instances, the result also includes paths associated with those selected objects.</p> <p>The range of the operation is all sheets of a filtered schematic or just the current sheet of an unfiltered schematic. The result is always a filtered schematic.</p> <p>In contrast to the Expand operations, which add to what you see, Isolate Paths can only remove objects from the display. While Isolate Paths is similar to Expand to Register/Port, Isolate Paths reduces the display while Expand to Register/Port augments it.</p> |
| Show Context        | Shows the original, unfiltered schematic sheet that contains the selected instance. Available only in a filtered schematic.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Hide Instances      | <p>Hides the logic inside the selected hierarchical (non-primitive) instances. This affects only the active HDL Analyst view; the instances are not hidden in other HDL Analyst views.</p> <p>The logic inside hidden instances is not loaded (saving dynamic memory), and it is unrecognized by searching, dissolving, flattening, expansion, and push/pop operations. (Crossprobing does recognize logic inside hidden instances, however.) See <a href="#">Usage Notes for Hiding Instances</a>, on page 552 for tips.</p>                                                                                                                                                                                                                                                                                                                                    |
| Unhide Instances    | Undoes the effect of Hide Instances: the selected hidden hierarchical instances become visible (susceptible to loading, searching, dissolving, flattening, expansion, and push/pop operations). This affects only the current HDL Analyst view; the instances are not hidden in other HDL Analyst views.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

| HDL Analyst Command | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show All Hier Pins  | Shows all pins on the selected transparent, non-primitive instances. Available only in a filtered schematic. Normally, transparent instance pins that are connected to logic that has been filtered out are not displayed. This command lets you display these pins that connect to logic that has been filtered out. Pins on primitives are always shown.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Dissolve Instances  | Shows the lower-level details of the selected non-hidden hierarchical instances. The number of levels dissolved is determined by the Dissolve Levels value in the HDL Analyst Options dialog box ( <a href="#">Standard HDL Analyst Options Command , on page 589</a> ). For usage tips, see <a href="#">Usage Notes for Dissolving Instances , on page 552</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Dissolve to Gates   | <p>Dissolves the selected instances by flattening them to the gate level. This command displays the lower-level hierarchy of selected instances, but it dissolves technology primitives as well as hierarchical instances. Technology primitives are dissolved to generic synthesis symbols. The command is only available in the Technology view.</p> <p>The number of levels dissolved is determined by the Dissolve Levels value in the HDL Analyst Options dialog box (<a href="#">Standard HDL Analyst Options Command , on page 589</a>).</p> <p>Dissolving an instance one level redraws the current sheet, replacing the hierarchical dissolved instance with the logic you would see if you pushed into it using Push/pop mode. Unselected objects or selected hidden instances are not dissolved.</p> <p>The effect of the command varies:</p> <ul style="list-style-type: none"> <li>• In an unfiltered schematic, this command <i>flattens</i> the selected instances. This means the history of displayed sheets is removed. The resulting schematic is unfiltered.</li> <li>• In a filtered schematic, this command makes the selected instances <i>transparent</i>, displaying their internal, lower-level logic inside hollow boxes. History is retained. You can use Flatten Schematic to flatten the transparent instances, if necessary. The resulting schematic is filtered.</li> </ul> |

## Usage Notes for Hiding Instances

The following are a few tips for hiding instances. For detailed procedures, see [Flattening Schematic Hierarchy, on page 463](#) of the *User Guide*.

- Hiding hierarchical instances soon after startup can often save memory. After the interior of an instance has been examined (by searching or displaying), it is too late for this savings.
- You can save memory by creating small, temporary working files: File->Save As .srs or .srm files does not save the hidden logic (hidden instances are saved as black boxes). Restarting the synthesis tool and loading such a saved file can often result in significant memory savings.
- You can selectively flatten instances by temporarily hiding all the others, flattening, then unhiding.
- You can limit the range of Edit->Find (see [Find Command \(HDL Analyst\), on page 407](#)) to prevent it looking inside given instances, by temporarily hiding them.

## Usage Notes for Dissolving Instances

Dissolving an instance one level redraws the current sheet, replacing the hierarchical dissolved instance with the logic you would see if you pushed into it using Push/pop mode. Unselected objects or selected hidden instances are not dissolved. For additional information about dissolving instances, see [Flattening Schematic Hierarchy, on page 463](#) of the *User Guide*.

The type (filtered or unfiltered) of the resulting schematic is unchanged from that of the current schematic. However, the effect of the command is different in filtered and unfiltered schematics:

- In an unfiltered schematic, this command flattens the selected instances. This means the history of displayed sheets is removed.
- In a filtered schematic, this command makes the selected instances transparent, displaying their internal, lower-level logic inside hollow boxes. History is retained. You can use Flatten Schematic to flatten the transparent instances, if necessary. This command is only available if an HDL Analyst view is open.

## HDL Analyst Menu: Selection Commands

This table describes the selection commands on the HDL Analyst menu.

| HDL Analyst Command                            | Description                                                                                                                                                               |
|------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Select All Schematic<br>->Instances<br>->Ports | Selects all Instances or Ports, respectively, on all sheets of the current schematic. All other objects are unselected. This does not select objects on other schematics. |
| Select All Sheet<br>->Instances<br>->Ports     | Selects all Instances or Ports, respectively, on the current schematic sheet. All other objects are unselected.                                                           |
| Unselect All                                   | Unselects all objects in all HDL Analyst views.                                                                                                                           |

## HDL Analyst Menu: FSM Commands

This table describes the FSM commands on the HDL Analyst menu.

| HDL Analyst Command | Description                                                                                                                                                                                          |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| View FSM            | <i>Synplify Pro, Synplify Premier</i><br>Displays the selected finite state machine in the FSM Viewer. Available only in an RTL view.                                                                |
| View FSM Info File  | Displays information about the selected finite state machine module, including the number of states, the number of inputs, and a table of the states and transitions. Available only in an RTL view. |

## HDL Analyst Menu: VCD Commands

This table describes the VCD/Identify-HDL Analyst commands for the VCD simulation or Identify Analysis tool on the HDL Analyst menu. Select HDL Analyst->VCD from the menu.

For more information, see [Using VCD/Identify with HDL Analyst, on page 1179](#).

| HDL Analyst Command: | Description |
|----------------------|-------------|
| <b>VCD Commands</b>  |             |

|                            |                                                                                                                                                                                                                                                                                       |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VCD Panel                  | Toggles the VCD Control Panel to be displayed in the HDL Analyst views. See the <a href="#">Simulation VCD Control Panel, on page 555</a> and <a href="#">Identify VCD Control Panel, on page 559</a> .                                                                               |
| Annotate Values            | Enable this option to display the net values in the HDL Analyst view.                                                                                                                                                                                                                 |
| Load VCD File              | Loads the simulation or Identify file (.vcd). The Control Panel changes to a gradient blue color and indicates an active session when the VCD file is loaded. See the <a href="#">Load Simulation VCD File, on page 556</a> and <a href="#">Load Identify VCD File, on page 559</a> . |
| Reload VCD File            | Reloads the simulation or Identify file (.vcd). The Control Panel changes to a gradient blue color and indicates an active session when the VCD file is loaded.                                                                                                                       |
| Unload the VCD File        | Unloads the simulation or Identify file (.vcd) to free up memory used by the simulation data without having to close and re-open the Analyst view.                                                                                                                                    |
| Watch Sheet                | Watch all nets on the current sheet.                                                                                                                                                                                                                                                  |
| Un-Watch Sheet             | Remove watching all nets on the current sheet.                                                                                                                                                                                                                                        |
| Automatically Watch Sheets | While navigating to a sheet, all nets on the sheet are added to the watch list.                                                                                                                                                                                                       |
| Next Transition            | You can view the next transition for an object being watched in the Waveform view.                                                                                                                                                                                                    |
| Previous Transition        | You can view the previous transition for an object being watched in the Waveform view.                                                                                                                                                                                                |

**DVE**

You can perform the following tasks:

- Launch DVE
- Configure DVE
- Import Watched Signals from DVE
- Export Watched Signals to DVE

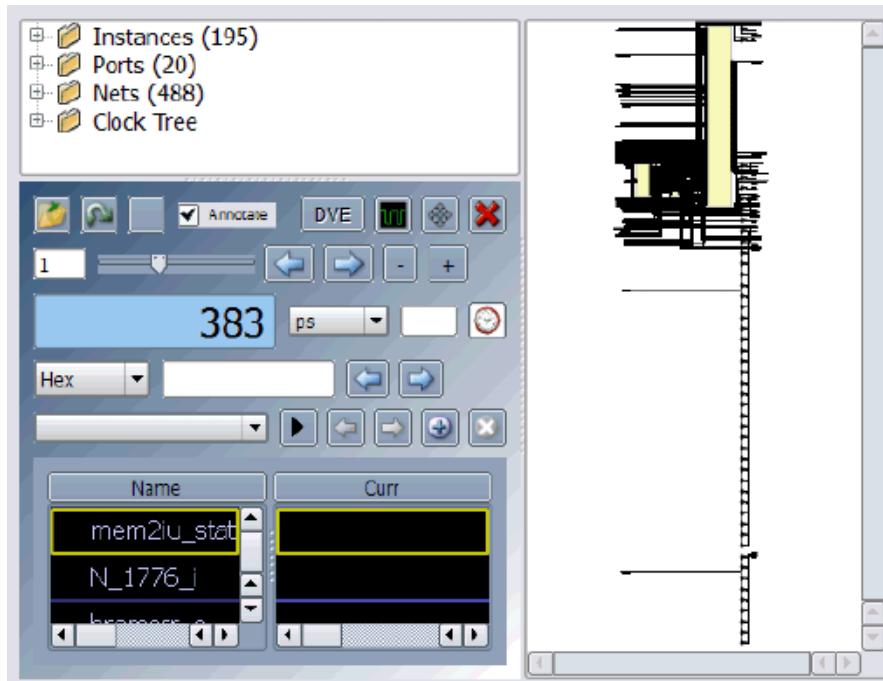
See [DVE Commands , on page 561.](#)

**VCD Properties**

Displays properties for the Waveform viewer global data and parameters. See [VCD Properties , on page 566.](#)

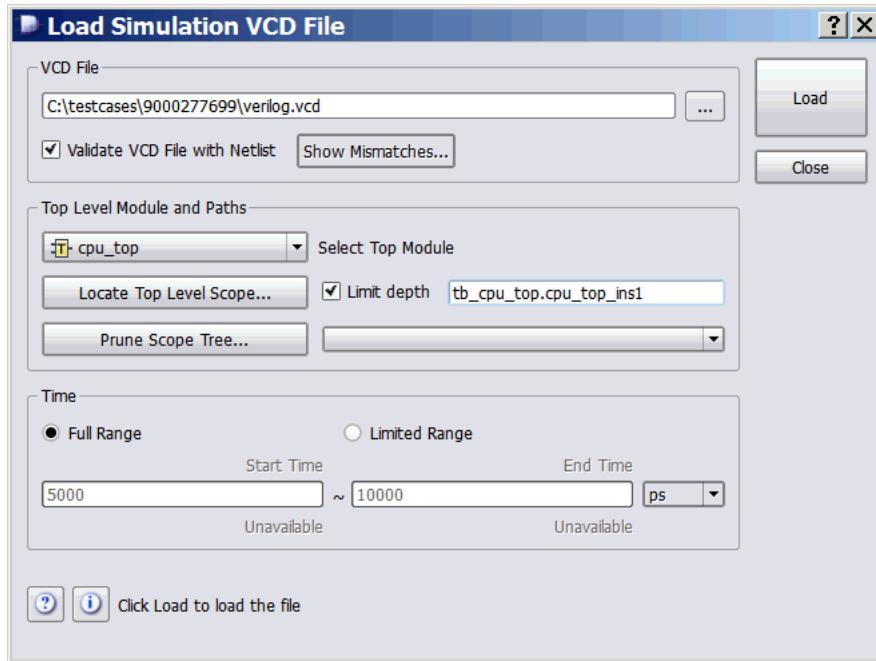
## Simulation VCD Control Panel

The Control Panel lets you use the VCD/Identify-Analyst feature, which enhances the interaction between the VCS Simulator and the HDL Analyst tools by providing a way to watch signals annotated in the HDL Analyst from the Control Panel watch list. For more information, see [Using VCD/Identify with HDL Analyst, on page 1179.](#)



## Load Simulation VCD File

To load the simulation file, click the Open a VCD File icon (  ) or select HDL-Analyst->VCD->Load VCD File from the Project menu.



This table describes the Load Simulation VCD File dialog box commands.

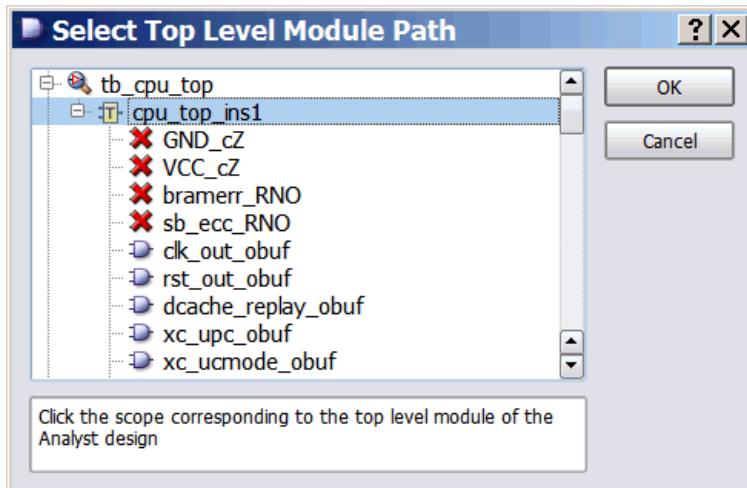
| HDL Analyst Command: | Description |
|----------------------|-------------|
| <b>VCD Commands</b>  |             |

|                                |                                                                                                                                                                                                                                                                                           |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VCD File                       | Specifies the simulation file (.vcd) to load.                                                                                                                                                                                                                                             |
| Validate VCD File with Netlist | When enabled, the tool reports net or signal mismatches at the bottom of the Load Simulation VCD File dialog box. Click Show Mismatches to display any mismatches.<br>If this option is disabled, the tool does not validate the VCD file and loads the VCD netlist without any warnings. |
| Show Mismatches                | Displays any mismatches.                                                                                                                                                                                                                                                                  |

|                            |                                                                                                                                                                                                                                                                         |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Top Level Module and Paths | Specifies the path through the test bench to the root module displayed by the Analyst. Use the drop-down menu to select the top level module of the VCD file. When Limit depth is enabled, the hierarchy level for the Top Level Scope is set to 3 levels of hierarchy. |
| Locate Top Level Scope     | Opens the Select Top Level Module Path dialog box where you can specify the top-level module for the HDL Analyst design. See <a href="#">Select Top Level Module Path Dialog Box , on page 557</a> .                                                                    |
| Limit depth                | When enabled, sets the hierarchy level for the root scope to three levels of hierarchy.                                                                                                                                                                                 |
| Prune Scope Tree           | Prune the scope tree for the root scope paths to be included in the hierarchy being loaded. See <a href="#">Prune Scope Hierarchy Dialog Box , on page 558</a> .                                                                                                        |
| Time                       | You can set the time to Full Range or Limited Range. The time range is determined from the simulation file to be loaded. The value is calculated when you browse the root scope. Limited Range lets you choose a range and desired unit of measurement.                 |
| Load button                | Loads the simulation VCD file.                                                                                                                                                                                                                                          |

## Select Top Level Module Path Dialog Box

Select the path to the top-level module of the design.

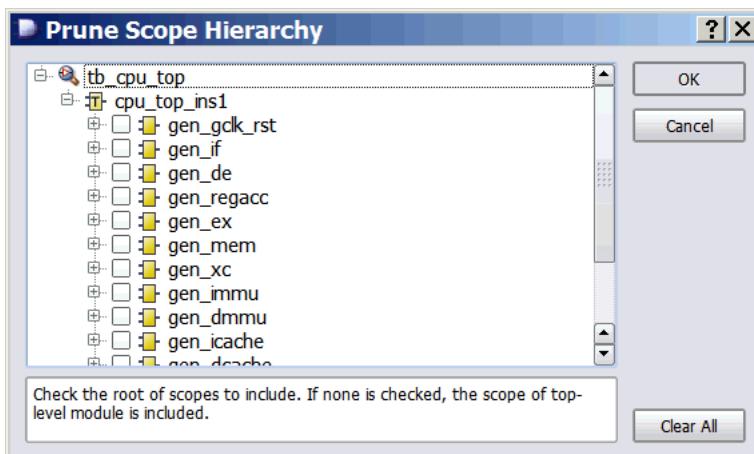


Icons next to the hierarchy tree provide information about how the scopes match the netlist. For example:

|  |                                                                                                                                                                                 |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | User specified top-level module                                                                                                                                                 |
|  | Hierarchical module                                                                                                                                                             |
|  | Primitive<br>(Scope is considered primitive by the HDL Analyst)                                                                                                                 |
|  | Scope is within the test bench                                                                                                                                                  |
|  | Scope does not match the hierarchy in the HDL Analyst netlist<br>(Some scopes, such as power, do not exist in the netlist or are considered to be primitive in the HDL Analyst) |

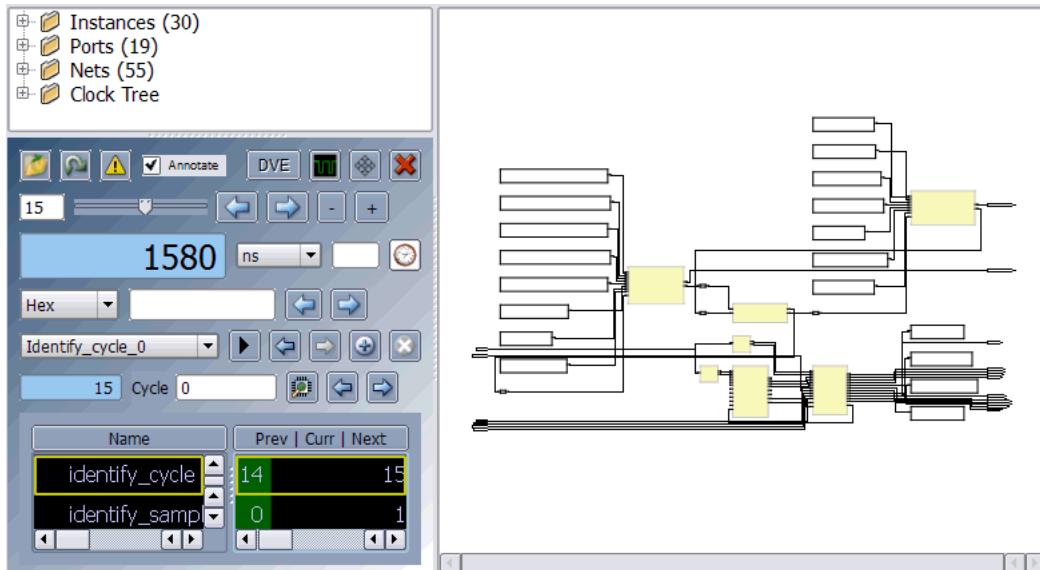
## Prune Scope Hierarchy Dialog Box

Prune the scope tree for the root scope paths to be included in the hierarchy being loaded. Check the root scopes to be included in the hierarchy. If none of the scopes are checked, then the scope for the top-level module is loaded. Click the Clear All button to ensure that none of the root scopes are checked. Once you have pruned the scope tree, the paths for the scopes to be loaded are shown in the complimentary box to the right of the Prune Scope Tree button.



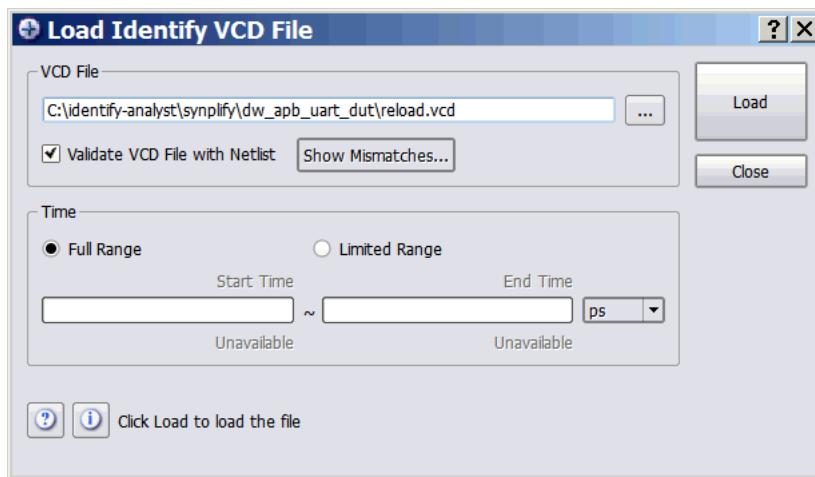
## Identify VCD Control Panel

The Control Panel lets you use the VCD/Identify-Analyst feature, which enhances the interaction between the Identify VCD and the HDL Analyst tools by providing a way to watch Identify instrumented signals annotated in the HDL Analyst from the Control Panel watch list. For more information, see [Using VCD/Identify with HDL Analyst, on page 1179](#).



## Load Identify VCD File

To load the Identify VCD file, click the Open a VCD File icon ( ) or select HDL-Analyst->VCD->Load VCD File from the Project menu.



This table describes the Load Simulation VCD File dialog box commands.

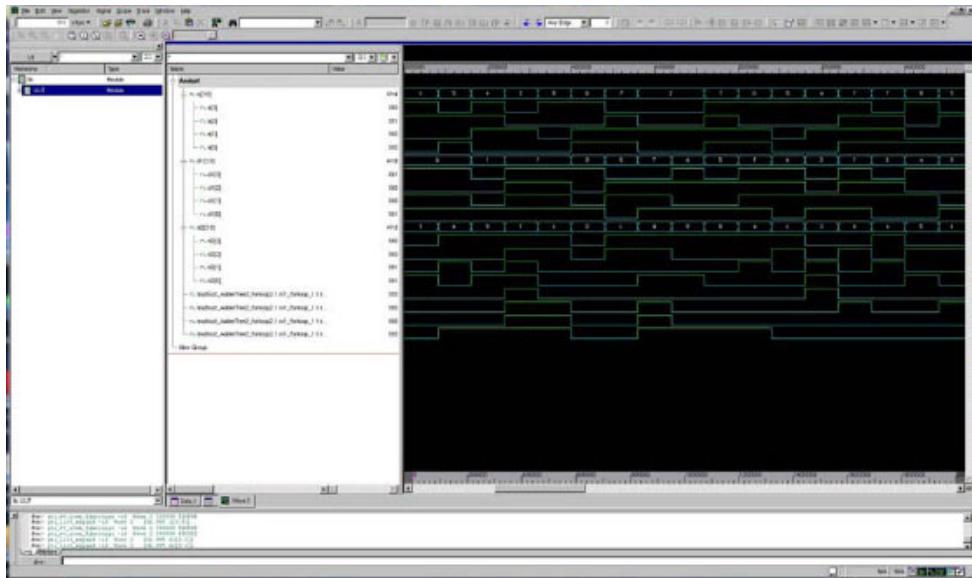
| HDL Analyst Command:<br>VCD Commands | Description                                                                                                                                                                                                                                                                                   |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VCD File                             | Specifies the Identify .vcd file to load.                                                                                                                                                                                                                                                     |
| Validate VCD File with Netlist       | When enabled, the tool reports net or signal mismatches at the bottom of the Load Simulation VCD File dialog box. Click Show Mismatches to display any mismatches.<br><br>If this option is disabled, the tool does not validate the VCD file and loads the VCD netlist without any warnings. |
| Show Mismatches                      | Displays any mismatches.                                                                                                                                                                                                                                                                      |
| Time                                 | You can set the time to Full Range or Limited Range. The time range is determined from the simulation file to be loaded. The value is calculated when you browse the root scope. Limited Range lets you choose a range and desired unit of measurement.                                       |
| Identify debug                       | Enables/disables the Identify-VCD HDL Analyst flow. If you do not check this option, the tool loads the VCD netlist without validating it or reporting warnings.                                                                                                                              |
| Load button                          | Loads the Identify VCD file.                                                                                                                                                                                                                                                                  |

## DVE Commands

Note: You must have a VCS license and run the DVE tool on a Linux platform.

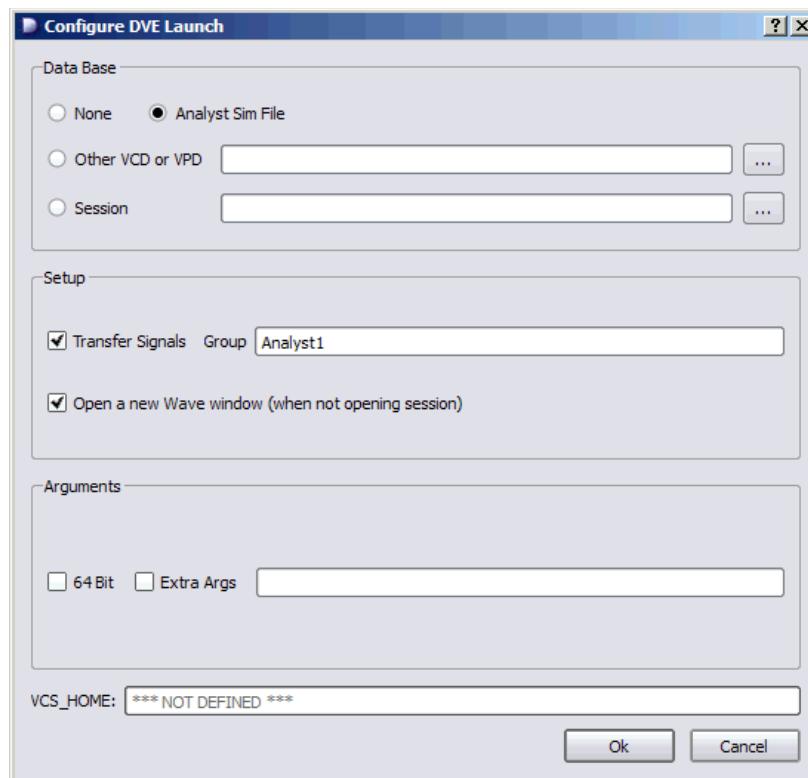
DVE (Discovery Virtual Environment) is an interactive waveform viewer that you can launch and configure within the VCD-HDL Analyst integration. Select a command from the HDL Analyst->VCD->DVE menu.

| Command        | Details                                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Launch DVE     | Launches the interactive DVE waveform viewer. This is the same as using the (  ) icon. |
| Configure DVE  | Configures the viewer. <i>Configure DVE Launch Dialog Box , on page 562</i>                                                                                             |
| Import Signals | Imports watched signals from the DVE viewer to the VCD-HDL Analyst integration. See <i>Import Signals Dialog Box , on page 563</i> .                                    |
| Export Signals | Exports watched signals from the VCD-HDL Analyst integration to the DVE waveform viewer. See <i>Export Signals Dialog Box , on page 564</i> .                           |



## Configure DVE Launch Dialog Box

To configure DVE options, select HDL Analyst->VCD->DVE->Configure DVE.

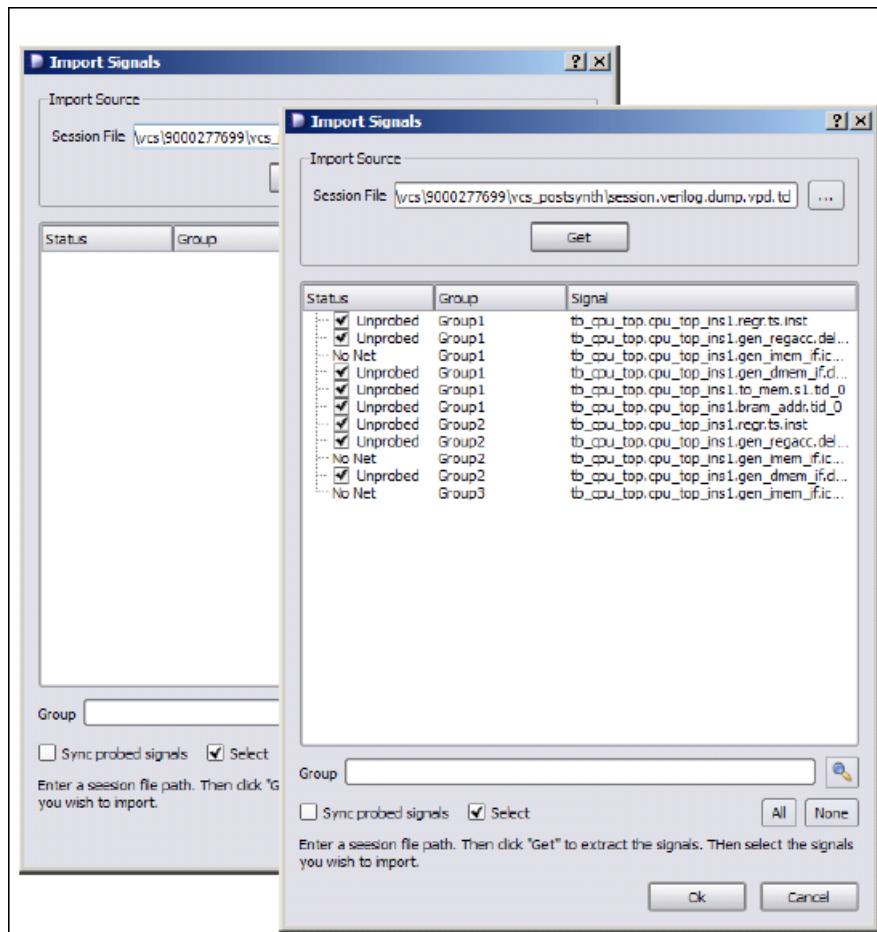


| Option                  | Description                                                                                                                                                                                            |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| None / Analyst Sim File | <ul style="list-style-type: none"><li>None: Does not select signals from the HDL Analyst simulation file.</li><li>Analyst Sim File<br/>Selects signals from the HDL Analyst simulation file.</li></ul> |
| Other VCD or VPD        | Specifies the path to another simulation file.<br>Session - you can re-load a saved DVE session Tcl file.                                                                                              |
| Session                 | Specifies a saved DVE session Tcl file to reload.                                                                                                                                                      |
| Transfer Signals Group  | Transfers specified simulation file signals to a user-specified group, specified in group.                                                                                                             |
| Open a new Wave Window  | When enabled, opens a new waveform window.                                                                                                                                                             |

|            |                                                                         |
|------------|-------------------------------------------------------------------------|
| 64-bit     | When enabled, specifies the 64-bit configuration for DVE.               |
| Extra Args | Specifies other configuration options for DVE.                          |
| VCS_HOME   | Read-only field that specifies the VCS_HOME/bin installation directory. |

## Import Signals Dialog Box

Import watched signals from DVE to the VCD-Analyst Integration tool. Select HDL Analyst->VCD->DVE->Import Watched Signals from DVE on the Project menu.

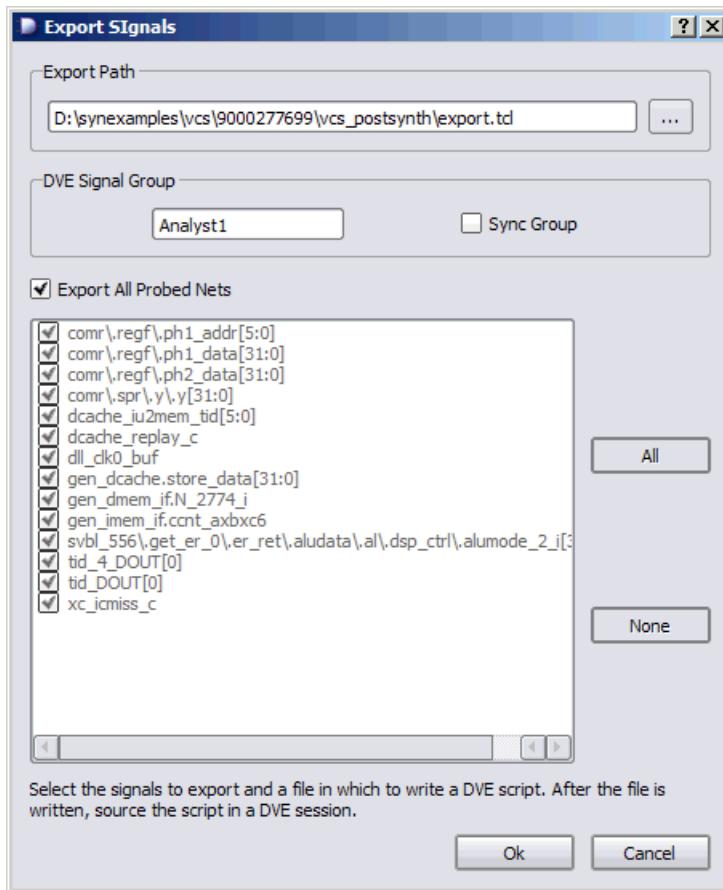


This table describes the Import Signals dialog box commands.

| Option               | Description                                                                           |
|----------------------|---------------------------------------------------------------------------------------|
| Session File         | Specify the DVE session file path containing the source input signals to be imported. |
| Group                | You can search for a specific DVE signal group.                                       |
| Sync watched signals | Synchronize signals to be the same signals being watched in the Simulation Panel.     |
| Get button           | Click the Get button to extract the signals to watch from the import source file.     |

## Export Signals Dialog Box

Export watched signals from the VCD-Analyst Integration tool to DVE. Select HDL Analyst->VCD->DVE->Export Watched Signals to DVE from the Project menu.



This table describes the Export Signals dialog box commands.

| HDL Analyst Command | Description                                                                                                                                               |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Export path         | Specify the DVE script file to which exported signals are written, then source the script in a DVE session.                                               |
| DVE Signal Group    | Specify a specific signal group to export. To synchronize the signal group to match the signals to be watched in the Simulation Panel, enable Sync Group. |

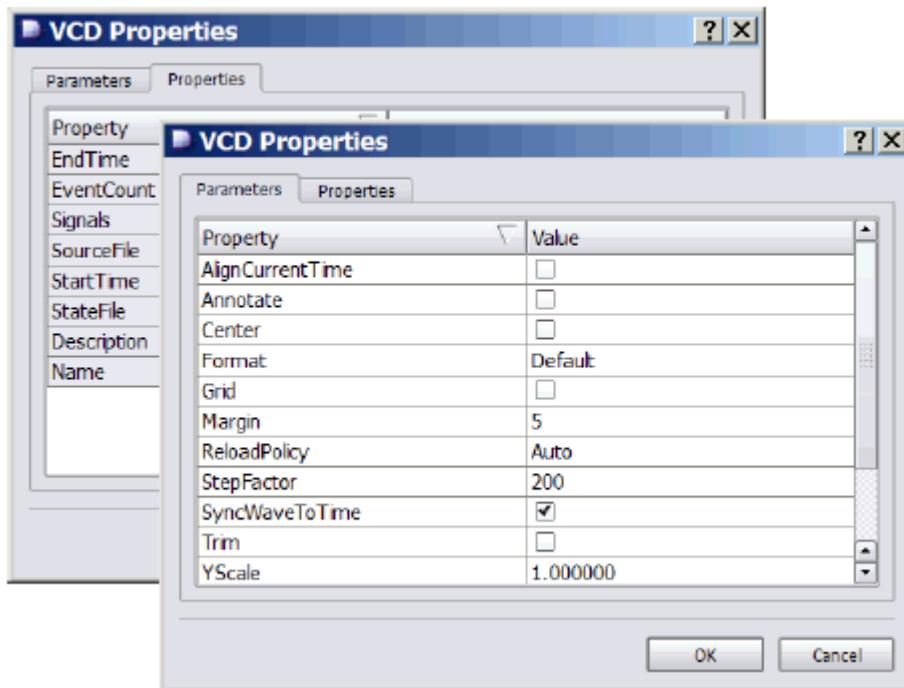
**Export All Watched Nets**

When Export All Watched Nets is enabled, this option extracts all the signals to watch from the import source file. When you disable this option, then select:

- All - all signals are checked.
- None - all signals are unchecked. Then, you can select specific nets to export.

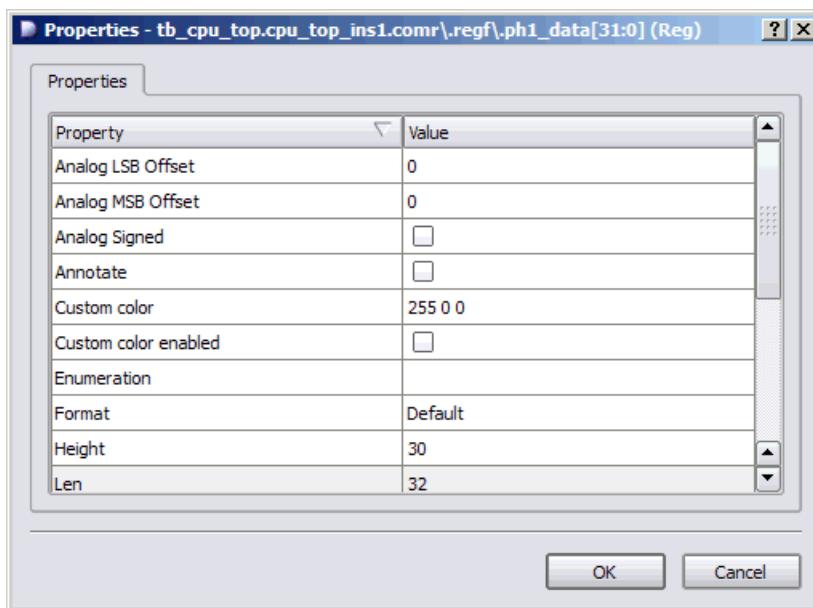
## VCD Properties

Displays properties for the Waveform viewer global data and parameters. Select HDL-Analyst->VCD->VCD Properties from the Project menu.



To display signal properties for a watched net in the Control Panel, do the following:

- Be sure at least one net is being watched.
- Right-click on the net and select Properties from the popup menu.



# Options Menu

Use the Options menu to configure the VHDL and Verilog compilers, customize toolbars, and set options for the Project view, Text Editor, and HDL Analyst schematics. When using certain technologies, additional menu commands let you run technology-vendor software from this menu.

The following table describes the Options menu commands.

| Command                                          | Description                                                                                                                                                                                                                                                                                                                                                                 |
|--------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Basic Options Menu Commands for all Views</b> |                                                                                                                                                                                                                                                                                                                                                                             |
| Configure VHDL Compiler                          | Opens the Implementation Options dialog box where you can set the top-level entity and the encoding method for enumerated types. State-machine encoding is automatically determined by the FSM compiler or FSM explorer, or you can specify it explicitly using the <code>syn_encoding</code> attribute. See <a href="#">Implementation Options Command , on page 444</a> . |
| Configure Verilog Compiler                       | Opens the Implementation Options dialog box where you can specify the top-level module and the include search path. See <a href="#">Implementation Options Command , on page 444</a> .                                                                                                                                                                                      |
| Configure Parallel or Compile Point Process      | Lets you specify the maximum number of parallel synthesis jobs that can be run and how errors are treated, for example with compile points. See <a href="#">Configure Parallel or Compile Point Process Command , on page 570</a> .                                                                                                                                         |
| Toolbars                                         | Lets you customize the toolbars.                                                                                                                                                                                                                                                                                                                                            |
| Project View Options                             | Sets options for organizing files in the Project view. See <a href="#">Project View Options Command , on page 573</a> .                                                                                                                                                                                                                                                     |
| Editor Options                                   | Sets your Text Editor syntax coloring, font, and tabs. See <a href="#">Editor Options Command , on page 580</a> .                                                                                                                                                                                                                                                           |
| P&R Environment Options                          | Displays the environmental variable options set for the place-and-route tool. See <a href="#">Place and Route Environment Options Command , on page 583</a> .                                                                                                                                                                                                               |

| Command                          | Description                                                                                                                                                                                                                       |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Configure 3rd Party Tool Options | Lets you invoke third-party tools, for example to modify the files generated or debug problems from the EDK tool within the FPGA synthesis products. See <a href="#">Configure 3rd Party Tools Options Command , on page 584.</a> |
| Project Status Page Location     | Saves the current project status to a location of your choice. See <a href="#">Project Status Page Location , on page 585.</a>                                                                                                    |
| HDL Analyst Options              | Sets display preferences for New HDL Analyst schematics (RTL and Technology views). See <a href="#">HDL Analyst Options Command , on page 587.</a>                                                                                |
| Standard HDL Analyst Options     | Sets display preferences for HDL Analyst schematics (RTL and Technology views). See <a href="#">Standard HDL Analyst Options Command , on page 589.</a>                                                                           |
| Configure External Programs      | Lets you set browser and Acrobat Reader options on Linux platforms. See <a href="#">Configure External Programs Command , on page 597.</a>                                                                                        |

### Options Menu Commands Specifically for the Project View

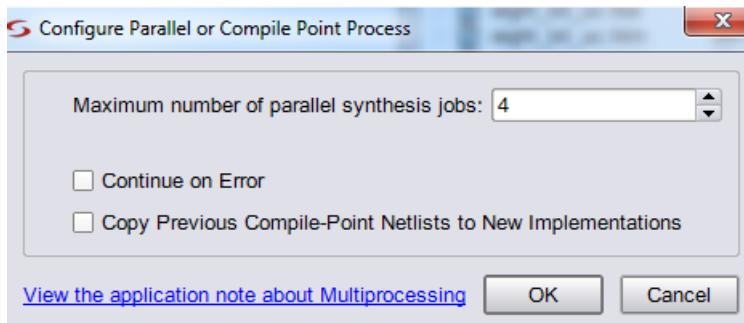
|                                                                                                         |                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Configure Identify Launch                                                                               | If Identify software is not properly installed, you might run into problems when you try to launch it from the synthesis tools. Use the Configure Identify Launch dialog box to help you resolve these issues. For guidelines to follow, see <a href="#">Handling Problems with Launching Identify, on page 1164</a> in the <i>User Guide</i> . |
| Quartus II (Intel only)<br>->Launch Quartus<br>->Run Background Compile                                 | These commands are only for Intel designs. They determine whether you run placement and routing interactively (Launch Quartus) or in batch mode (Run Background Compile).                                                                                                                                                                       |
| Xilinx (Xilinx only)<br>->Start Design Manager<br>->Start Floorplanner<br>->Start ISE Project Navigator | These commands are only available for Xilinx designs. They let you start Design Manager, the Xilinx Floorplanner, or ISE Project Navigator from the synthesis tool.                                                                                                                                                                             |

| Command                                                                 | Description                                                                             |
|-------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| <b>Options Menu Commands Specifically for the Physical Analyst View</b> |                                                                                         |
| Physical Analyst Control Panel                                          | For details, see <a href="#">Options Menu: Physical Analyst Commands</a> , on page 691. |
| Physical Analyst Color Schemes                                          | For details, see <a href="#">Options Menu: Physical Analyst Commands</a> , on page 691. |

## Configure Parallel or Compile Point Process Command

Use the Configure Parallel or Compile Point Process command to run parallel jobs for multiprocessing with compile points or distributed processing. For example, this option allows the software to run multiple, independent compile point jobs simultaneously, providing additional runtime improvements for the compile point synthesis flow.

This feature is supported on Windows and Linux for certain technologies only. The command is disabled for technologies that are not supported.



| Field/Option                                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Maximum Number of Parallel Synthesis Jobs                  | <p>Sets the maximum number of parallel synthesis jobs to run multiprocessing with compile points or distributed processing. It displays the current value from the .ini file, and allows you to reset it. Use this option for multiprocessing or distributed processing to run jobs in parallel.</p> <p>Set a value based on the number of available licenses. With each license, you can run four jobs in parallel by default. See <a href="#">Multiprocessing With Compile Points, on page 804</a>.</p> <p>When you set this option, it resets the MaxParallelJobs value in the .ini file. See <a href="#">Setting Number of Parallel Jobs, on page 782</a> in the <i>User Guide</i> and <a href="#">Maximum Parallel Jobs , on page 572</a> for other ways to specify this value.</p>                                                                                                                                                                                                                                                                                                                                           |
| Continue on Error                                          | <p><i>Synplify Pro, Synplify Premier</i></p> <p>Allows the software to continue on an error in a compile point and synthesize the rest of the design, even when there might be problems with a portion of the design.</p> <p>The Continue on Error mode automatically enables the MultiProcessing option to run with compile points using a single license; this is the default. For additional runtime improvements, you can specify multiple synthesis jobs that run in parallel. See <a href="#">Chapter 15, Improving Runtime</a> for details.</p> <p>For more information about Continue on Error mode during compile-point synthesis, see <a href="#">Combining Compile Points with Multiprocessing, on page 644</a> in the <i>User Guide</i>.</p> <p>The Continue on Error mode also allows design compilation in the Synplify Premier tools to continue for certain types of compiler errors. For more information about Continue on Error mode during compilation, see <a href="#">Continue on Error, on page 126</a> in the <i>User Guide</i>.</p> <p>Tcl equivalent: <code>set_option -continue_on_error 0 1</code></p> |
| Copy Previous Compile-Point Netlists to New Implementation | Allows you to copy compile point netlist (.srd) files generated from the previous implementation into a new implementation that has been created.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

## Maximum Parallel Jobs

There are three ways to specify the maximum number of parallel jobs:

**.ini File** Set this variable in the MaxParallelJobs variable in the product ini file:

```
[JobSetting]
MaxParallelJobs=<n>
```

This value is used by the UI as well as in batch mode, and is effective until you specify a new value. You can change it with the Options->Configure Parallel or Compile Point Process command.

**Tcl Variable** Set the following variable in a Tcl file, the project files, or from the Tcl window:

```
set_option -max_parallel_jobs=<n>
```

This is a global option that is applied to all project files and their implementations. This value takes effect immediately. If you set it in the Tcl file or project file, it remains in effect until you specify a new value. If you set it from the Tcl window, the max\_parallel\_jobs value is only effective for the session and will be lost when you exit the application.

**Configure Compile Point Process Command** The Maximum Number of Parallel Synthesis Jobs option displays the current ini file value and allows you to reset it.

## License Utilization for Multiprocessing

When you decide to run parallel synthesis jobs, additional licenses may be required for the compile point jobs. By default, four parallel jobs use one license. For example, if you set the Maximum number of parallel synthesis jobs to 12, the synthesis tool consumes one license to run 4 compile point jobs and can utilize the two additional licenses to run 8 more parallel jobs if they are available for your computing environment. Licenses are released as jobs complete, and then consumed by new jobs which need to run.

The actual number of licenses utilized depends on the following:

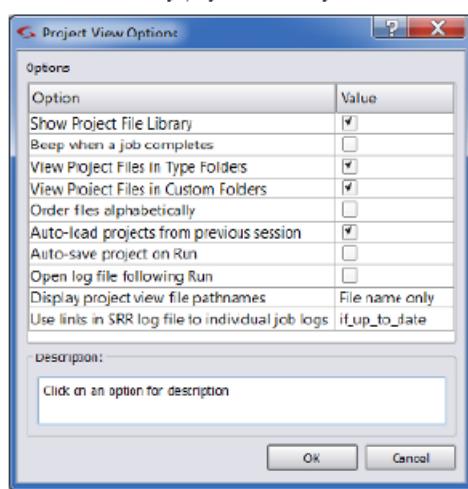
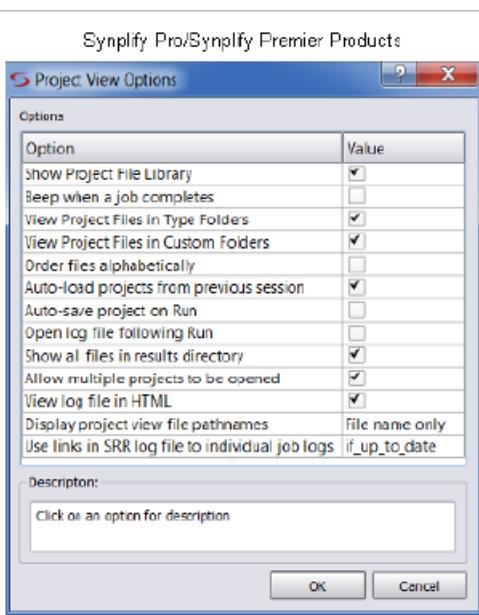
1. Synthesis software scheme for the compile point requirements used to determine the maximum number of parallel jobs or licenses a particular design tries to use.
2. Value set on the Configure Parallel or Compile Point Process dialog box.

3. Number of licenses actually available. You can use Help->Preferred License Selection to check the number of available license. If you need to increase the number of available licenses, you can specify multiple license types. For more information, see [Using Different License Types for Multiprocessing, on page 807](#).

Note that factors 1 and 3 above can change during a single synthesis run. The number of jobs equals the number of licenses, which then equates the lowest value of these three factors.

## Project View Options Command

Select Options->Project View Options to display the Project View Options dialog box, where you define how projects appear and are organized in the Project view.

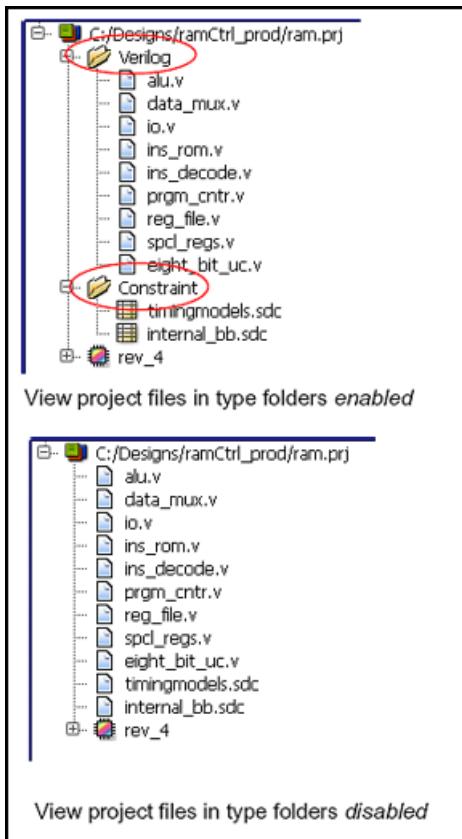


The following table describes the Project View Options dialog box features.

| Field/Option                            | Description                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show Project File Library               | <p>When enabled, displays the corresponding VHDL library next to each source VHDL filename, in the Project Tree view of the Project view. For example, with library dune, file pc.vhd is listed as [dune] pc.vhd if this option is enabled, and as pc.vhd if it is disabled.</p> <p>See also <a href="#">Set VHDL Library Command , on page 429</a>, for how to change the library of a file.</p> |
| Beep when a job completes               | <p>When enabled, sounds an audible signal whenever a project finishes running.</p>                                                                                                                                                                                                                                                                                                                |
| View Project Files in Type Folders      | <p>When enabled, organizes project files into separate folders by type. See <a href="#">View Project Files in Type Folders Option , on page 577</a> and <a href="#">add_file</a>, on page 25.</p>                                                                                                                                                                                                 |
| View Project Files in Custom Folders    | <p>When enabled, allows you to view files contained within the custom folders created for the project. See <a href="#">View Project Files in Custom Folders Option , on page 577</a>.</p>                                                                                                                                                                                                         |
| Order files alphabetically              | <p>When enabled, the software orders the files within folders alphabetically instead of in project order. You can also use the Sort Files option in the Project view.</p>                                                                                                                                                                                                                         |
| Autoload projects from previous session | <p>Enable/Disable automatically loading projects from the previous session. Otherwise, projects will not be loaded automatically. This option is enabled by default. See <a href="#">Loading Projects With the Run Command , on page 579</a>.</p>                                                                                                                                                 |
| Auto-save project on Run                | <p>Enable/Disable automatically saving projects when the Run button is selected. See <a href="#">Automatically Save Project on Run , on page 579</a>.</p>                                                                                                                                                                                                                                         |
| Open Log file following Run             | <p>Enable/Disable automatically opening and displaying log file after a synthesis run.</p>                                                                                                                                                                                                                                                                                                        |
| Show all files in results directory     | <p><i>Synplify Pro, Synplify Premier</i><br/>When enabled, shows all files in the Implementation Results view. When disabled, the results directory shows only files generated by the synthesis tool itself.</p>                                                                                                                                                                                  |

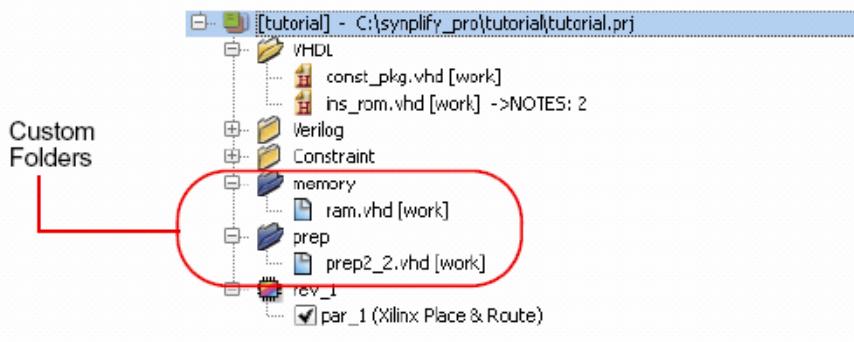
| Field/Option                                     | Description                                                                                                                                                                                                                                                                                                                     |
|--------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Allow multiple projects to be opened             | <i>Synplify Pro, Synplify Premier</i><br>When enabled, multiple projects are displayed at the same time. See <a href="#">Allow Multiple Projects to be Opened Option</a> , on page 578.                                                                                                                                         |
| View log file in HTML                            | Enable/Disable viewing of log file report in HTML format versus text format. See <a href="#">Log File, on page 199</a> .                                                                                                                                                                                                        |
| Project file name display                        | From the drop-down menu, select one the following ways to display project files: <ul style="list-style-type: none"><li>• File name only</li><li>• Relative file path</li><li>• Full file path</li></ul>                                                                                                                         |
| Use links in SRR log file to individual job logs | Determines if individual job logs use links in the srr log file. You can select: <ul style="list-style-type: none"><li>• off—appends individual job logs to the srr log file.</li><li>• on—always link to individual job logs.</li><li>• if_up_to_date—only links to individual job logs if the module is up-to-date.</li></ul> |

## View Project Files in Type Folders Option



## View Project Files in Custom Folders Option

Selecting this option enables you to view user-defined custom folders that contain a predefined subset of project files in various hierarchy groupings or organizational structures. Custom folders are distinguished by their blue color. For information on creating custom folders, see [Creating Custom Folders, on page 113](#) in the *User Guide*.



## Allow Multiple Projects to be Opened Option

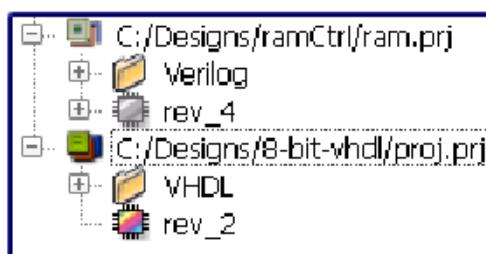
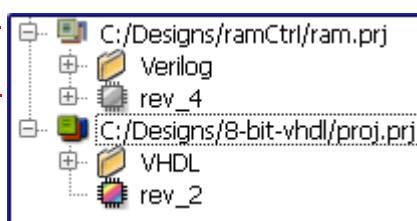
*Synplify Pro, Synplify Premier*

Project 1

Project 2

Project 1

Project 2



## Loading Projects With the Run Command

When you load a project that includes the project -run command, a dialog box appears in the Project view with the following message:

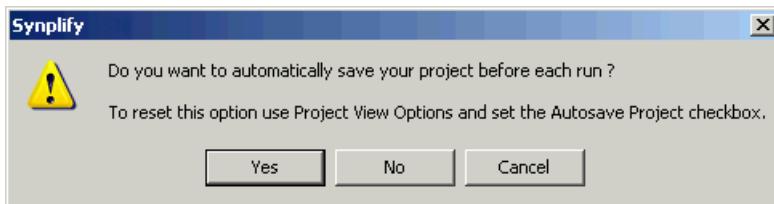
Project run command encountered during project load. Are you sure you want to run?

You can reply with either yes or no.

## Automatically Save Project on Run

If you have modified your project on the disk directory since being loaded into the Project view and you run your design, a message is generated that infers the UI is out-of-date.

The following dialog box appears with a message to which you must reply.

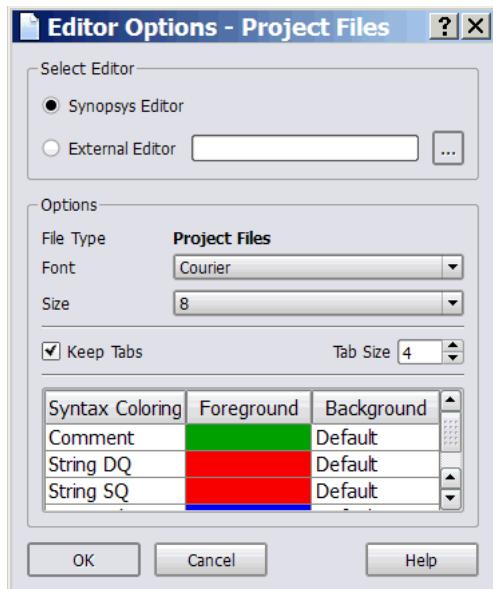


You can specify one of the following:

- Yes — The Auto-save project on Run switch on the Project View Options dialog box is automatically enabled, and then your design is run.
- No — The Auto-save project on Run switch on the Project View Options dialog box is not enabled, but your design is run.
- Cancel — Closes this message dialog box and does not run your design.

## Editor Options Command

Select Options->Editor Options to display the Editor Options dialog box, where you select either the internal text editor or an external text editor.



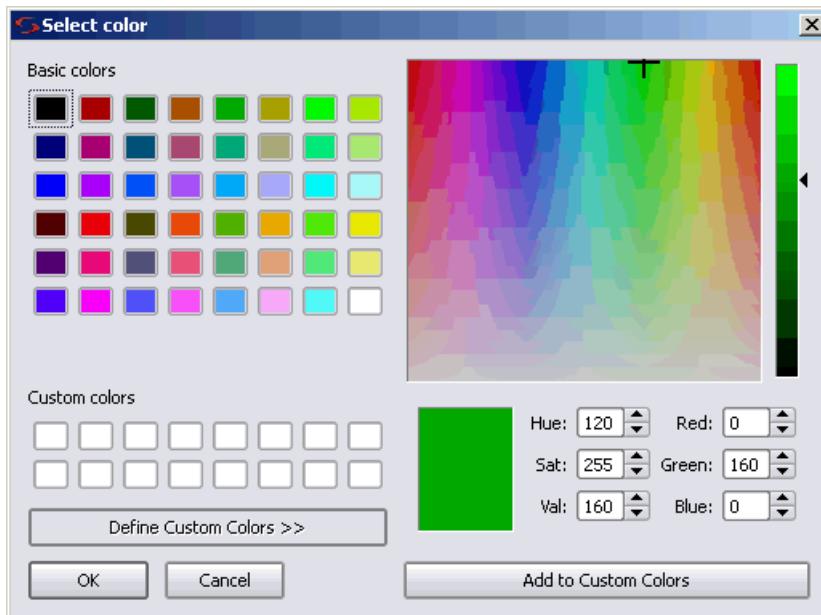
The following table describes the Editor Options dialog box features.

| Feature           | Description                                                                                                                                                                                                                                                                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Select Editor     | Select an internal or external editor.                                                                                                                                                                                                                                                                                                                    |
| • Synopsys Editor | Sets the Synopsys text editor as the default text editor.                                                                                                                                                                                                                                                                                                 |
| • External Editor | Uses the specified external text editor program to view text files from within the Synopsys tool. The executable specified must open its own window for text editing. See <a href="#">Using an External Text Editor, on page 77</a> of the <i>User Guide</i> for a procedure.<br><i>Note:</i> Files opened with an external editor cannot be crossprobed. |
| Options           | Set text editing preferences.                                                                                                                                                                                                                                                                                                                             |
| • File Type       | You can define text editor preferences for the following file types: project files, HDL files, log files, constraint files, and default files.                                                                                                                                                                                                            |

| Feature           | Description                                                                                                                             |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| • Font            | Lets you define fonts to use with the text editor.                                                                                      |
| • Font Size       | Lets you define font size to use with the text editor.                                                                                  |
| • Keep Tabs       | Lets you define whether to use tab settings with the text editor.                                                                       |
| • Tab Size        |                                                                                                                                         |
| • Syntax Coloring | Lets you define foreground or background syntax coloring to use with the text editor. See <a href="#">Color Options , on page 581</a> . |

## Color Options

Click in the Foreground or Background field for the corresponding object in the Syntax Coloring field to display the color palette.

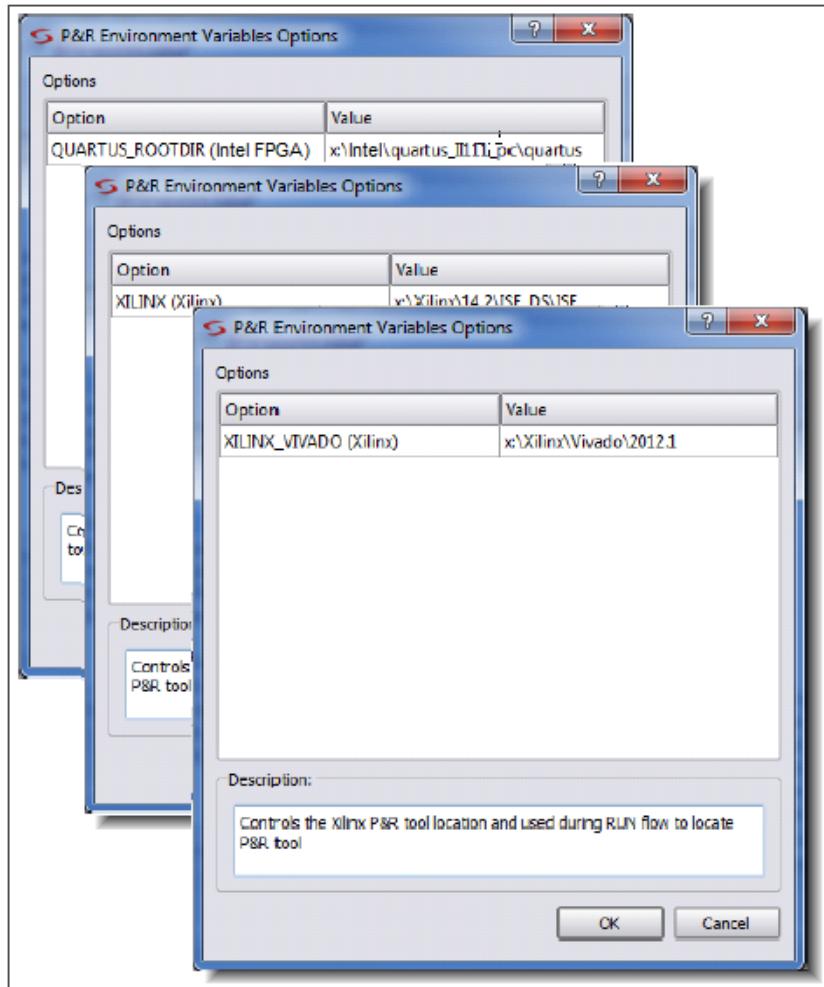


You can set syntax colors for some common syntax options listed in the following table.

| Syntax       | Description                                                                                                 |
|--------------|-------------------------------------------------------------------------------------------------------------|
| Comment      | Comment strings contained in all file types.                                                                |
| Error        | Error messages contained in the log file.                                                                   |
| Gates        | Gates contained in HDL source files.                                                                        |
| Info         | Informational messages contained in the log file.                                                           |
| Keywords     | Generic keywords contained in the project, HDL source, constraint, and log files.                           |
| Line Comment | Line comments contained in the HDL source, C, C++, and log files.                                           |
| Note         | Notes contained in the log file.                                                                            |
| SDCKeyword   | Constraint-specific keywords contained in the .sdc file.                                                    |
| Strength     | Strength values contained in HDL source files.                                                              |
| String DQ    | String values within double quotes contained in the project, HDL source, constraint, C, C++, and log files. |
| String SQ    | String values within single quotes contained in the project, HDL source, constraint, C, C++, and log files. |
| SVKeyword    | SystemVerilog keywords contained in the Verilog file.                                                       |
| Types        | Type values contained in HDL source files.                                                                  |
| Warning      | Warning messages contained in the log file.                                                                 |

## Place and Route Environment Options Command

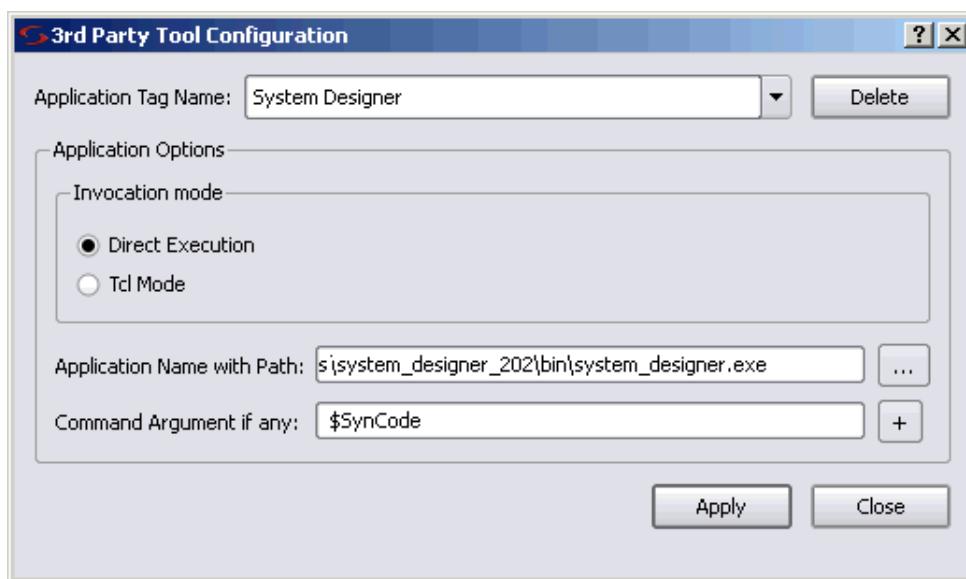
Select Options->P&R Environment Options to display the environment variable options set for the place-and-route tool. This option allows you to change the specified location of the selected place-and-route tool set on your system; the software locates and runs this updated version of the P&R tool for the current session of the synthesis tool.



## Configure 3rd Party Tools Options Command

Use the Configure 3rd Party Tools Option command to invoke third-party tools, such as the Embedded Development Kit (EDK) from within the Synopsys FPGA products. This allows you to modify source files or libraries added to your synthesis projects from within the third-party tool directly. Use the following dialog box to configure the location and common arguments for the tools.

For more information, see [Invoking Third-Party Vendor Tools, on page 799](#) in the *User Guide*.



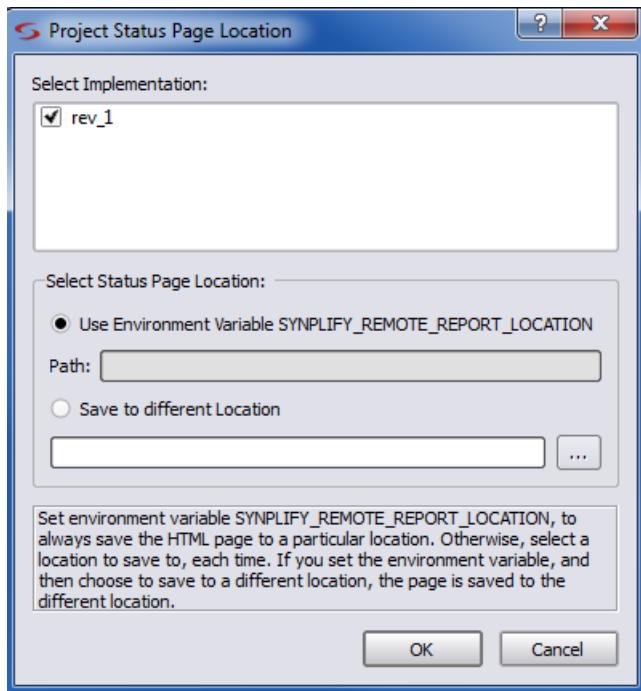
The 3rd Party Tool Configuration dialog box includes the following options:

| Feature              | Description                                                                                                                                                       |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Application Tag Name | Specifies an application or Tcl procedure name. Type in the name or select a preconfigured application from the list.                                             |
| Direct Execution     | Sets up the direct invocation of a third-party tool from within the FPGA synthesis tool, using the path defined for the executable in Application Name with Path. |

| Feature                    | Description                                                                                                                                                                                                                                                            |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tcl Mode                   | Sets up the tool to execute the Tcl procedure from within the FPGA synthesis tool, using the path defined for the procedure in Tcl Procedure Name. You must execute this Tcl script from the Tcl window to register the invocation procedure for the third-party tool. |
| Application Name with Path | When using direct execution, specifies the path to the executable for the application.                                                                                                                                                                                 |
| Tcl Procedure Name         | When using Tcl mode, specifies the Tcl procedure name.                                                                                                                                                                                                                 |
| Command Argument if any    | Defines any additional arguments for the third-party application. You can select arguments from the drop-down list or type them.<br>Note: For internal Synopsys tools, you must include the \$Syncode parameter.                                                       |
| Procedure Arguments if any | Defines additional arguments for the Tcl procedure. You can select arguments from the drop-down list or type them.                                                                                                                                                     |

## Project Status Page Location

Lets you save the current project status to a location of your choice. You can then view the project status offline with any browser on a mobile device.

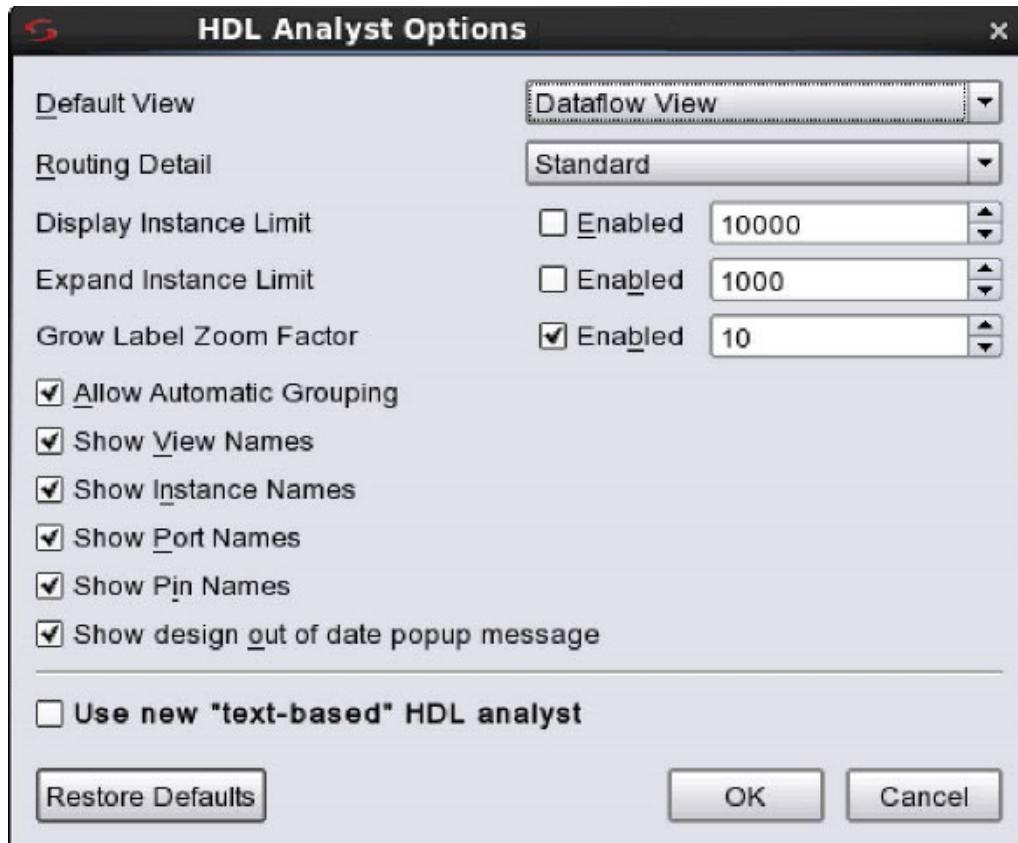


The following table describes the Project Status Page Location dialog box options.

| Option                                                                                                                                                                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Select Implementation                                                                                                                                                           | Select the implementation for the design for which you want synthesis results. You can select multiple implementations.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Select Status Page Location <ul style="list-style-type: none"><li>• Use Environment Variable<br/>SYNPLIFY_REMOTE_REPORT_LOCATION</li><li>• Save to Different Location</li></ul> | Select the location on your computer where you want to save the project status reports: <ul style="list-style-type: none"><li>• Use the environment variable to specify a standard location for the project status reports. Choose this option if you always want to save the reports to the same location.</li><li>• Choose a location for the project status reports for the current implementation. You can change this as often as you like.</li></ul> For more information, see <a href="#">Accessing Results Remotely, on page 295</a> in the <i>User Guide</i> . |

## HDL Analyst Options Command

Select Options->Schematic Options to display a dialog box where you define preferences for the HDL Analyst schematic. For details see [Setting Schematic Preferences, on page 356](#) in the *User Guide*.



The following options are on the HDL Analyst Options panel.

| Field/Option                          | Description                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Default View                          | <p>Specify how you want the schematic views to display:</p> <ul style="list-style-type: none"> <li>• Clocks View - Displays all sequential elements connected to clock nets so that clocks in the design can be debugged.</li> <li>• Dataflow View - Displays objects from a left to right datapath flow. This is the default.</li> </ul> |
| Routing Detail                        | <p>Specify how the tool determines the detailed routing for the design:</p> <ul style="list-style-type: none"> <li>• Standard - This is the default.</li> <li>• Quick - Direct net connections</li> </ul>                                                                                                                                 |
| Display Instance Limit                | <p>When enabled, uses the specified limit to display instances. The default is 10000.</p>                                                                                                                                                                                                                                                 |
| Expand Instance Limit                 | <p>When enabled, uses the specified limit to expand instances. The default is 1000.</p>                                                                                                                                                                                                                                                   |
| Grow Label Zoom Factor                | <p>Specify a zoom factor for labels displayed in the schematic view.</p> <p>Select a value between 1 and 10, where labels are shown increasing in size respectively. Changes will appear in the next opened schematic view. The default is 2.</p>                                                                                         |
| Allow Automatic Grouping              | <p>When enabled, automatic grouping is performed.</p>                                                                                                                                                                                                                                                                                     |
| Show View Names                       | <p>When enabled, module names are displayed.</p>                                                                                                                                                                                                                                                                                          |
| Show Instance Names                   | <p>When enabled, instance names are displayed.</p>                                                                                                                                                                                                                                                                                        |
| Show Port Names                       | <p>When enabled, port names are displayed.</p>                                                                                                                                                                                                                                                                                            |
| Show Pin Names                        | <p>When enabled, pin names are displayed.</p>                                                                                                                                                                                                                                                                                             |
| Show design out of date popup message | <p>When enabled, shows the design out of date popup message.</p>                                                                                                                                                                                                                                                                          |
| [BETA] Use new hierarchy browser      | <p>When enabled, the new Hierarchy Browser displays the RTL schematic at the instance level (Instance Hierarchy tab) and design level (Design View tab), in the next schematic displayed.</p>                                                                                                                                             |
| Restore Defaults                      | <p>Click this button to reset all options to their defaults.</p>                                                                                                                                                                                                                                                                          |

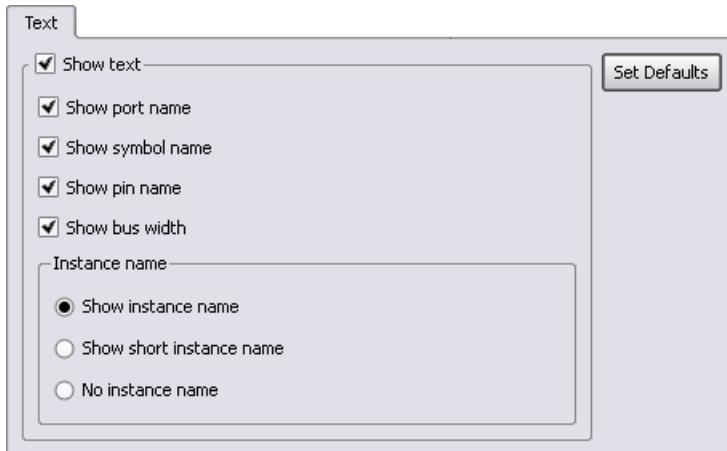
## Standard HDL Analyst Options Command

Select Options->HDL Analyst Options to display the HDL Analyst Options dialog box, where you define preferences for the HDL Analyst schematic views (RTL and Technology views). Some preferences take effect immediately, others only take effect in the next view that you open. For details, see [Setting Schematic Preferences, on page 420](#) in the *User Guide*.

For information about the options, see the following, which correspond to the tabs on the dialog box:

- [Text Panel, on page 589](#)
- [General Panel, on page 590](#)
- [Sheet Size Panel, on page 594](#)
- [Visual Properties Panel, on page 596](#)

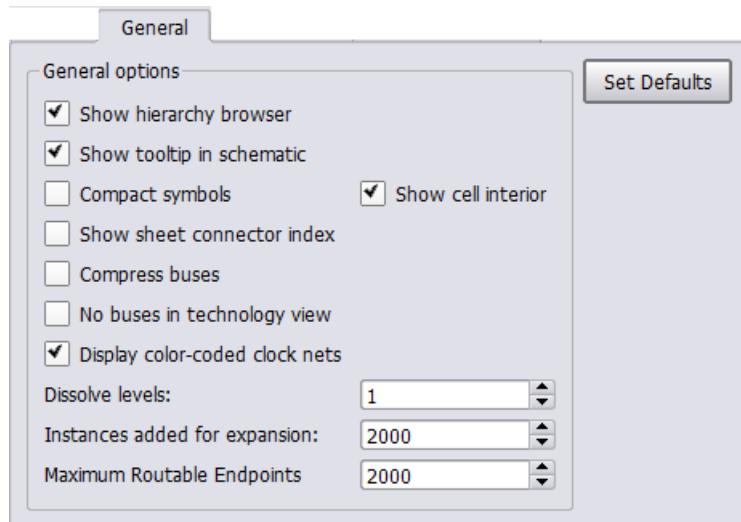
### Text Panel



The following options are on the Text panel.

| Field/Option     | Description                                                                                                                                                                      |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show text        | Enables the selective display of schematic labels. Which labels are displayed is governed by the other Show * features and Instance name, described below.                       |
| Show port name   | When enabled, port names are displayed.                                                                                                                                          |
| Show symbol name | When enabled, symbol names are displayed.                                                                                                                                        |
| Show pin name    | When enabled, pin names are displayed.                                                                                                                                           |
| Show bus width   | When enabled, connectivity bit ranges are displayed near pins (in square brackets: [ ]), indicating the bits used for each bus connection.                                       |
| Instance name    | Determines how to display instance names: <ul style="list-style-type: none"> <li>• Show instance name</li> <li>• Show short instance name</li> <li>• No instance name</li> </ul> |
| Set Defaults     | Set the dialog box to display the default values.                                                                                                                                |

## General Panel



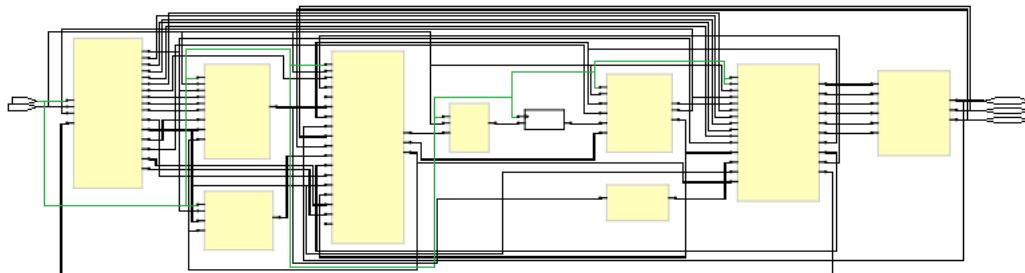
The following options are on the General panel.

| Field/Option                   | Description                                                                                                                                                                                                                                                                                                                        |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show hierarchy browser         | When enabled, a hierarchy browser is present at the left pane of RTL and Technology views.                                                                                                                                                                                                                                         |
| Show tooltip in schematic      | When enabled, displays tooltips that hover objects as you move over them in the RTL and Technology schematic views.                                                                                                                                                                                                                |
| Compact symbols                | When enabled, symbols are displayed in a slightly more compact manner, to save space in schematics.<br>When this is enabled, Show cell interior is disabled.                                                                                                                                                                       |
| Show cell interior             | When enabled, the internal logic of cells that are technology-specific primitives (such as LUTs) is shown in Technology views. This is not available if Compact symbols is enabled.                                                                                                                                                |
| Show sheet connector index     | When enabled, sheet connectors show connecting sheet numbers - see <a href="#">Sheet Connectors, on page 109</a> .                                                                                                                                                                                                                 |
| Compress buses                 | When enabled, buses having the same source and destination instances are displayed as bundles, to reduce clutter. A single bundle can connect to more than one pin on a given instance. The display of a bundle of buses is similar to that of a single bus.                                                                       |
| No buses in technology view    | When enabled, buses are not displayed, they are only indicated as bits in a Technology View. This applies only to flattened views created by HDL Analyst->Technology->Flattened View (or Flattened to Gates View), not to hierarchical views that you have flattened (using, for example, HDL Analyst->Flatten Current Schematic). |
| Display color-coded clock nets | Displays clock nets in the HDL Analyst View with the color green. See <a href="#">Color-coded Clock Nets , on page 592</a> .                                                                                                                                                                                                       |

| Field/Option                  | Description                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Dissolve levels               | The number of levels to dissolve, during HDL Analyst->Dissolve Instances. See <a href="#">Dissolve Instances</a> , on page 551.                                                                                                                                                                                                                                                                  |
| Instances added for expansion | The maximum number of instances to add during any operation (such as HDL Analyst->Hierarchical->Expand) that results in a <i>filtered</i> schematic. When this limit is reached, you are prompted to continue adding more instances.                                                                                                                                                             |
| Maximum Routable Endpoints    | <p>Specifies the maximum number of endpoints for nets, which the synthesis tool routes to their explicit connection endpoints in the design to improve HDL Analyst performance.</p> <p>The default value is set to 2000. You can use this option to change this value. For more information, see <a href="#">Results of Maximum Routable Endpoints in the HDL Analyst View</a>, on page 592.</p> |

## Color-coded Clock Nets

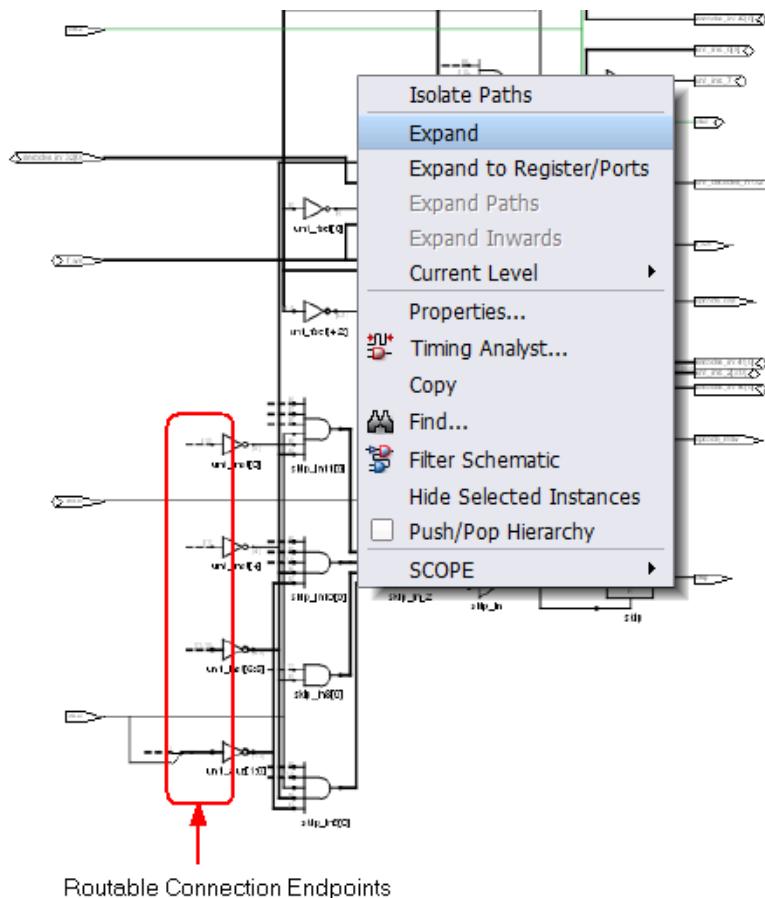
Clock nets are displayed with the color green in the RTL and Technology views.



## Results of Maximum Routable Endpoints in the HDL Analyst View

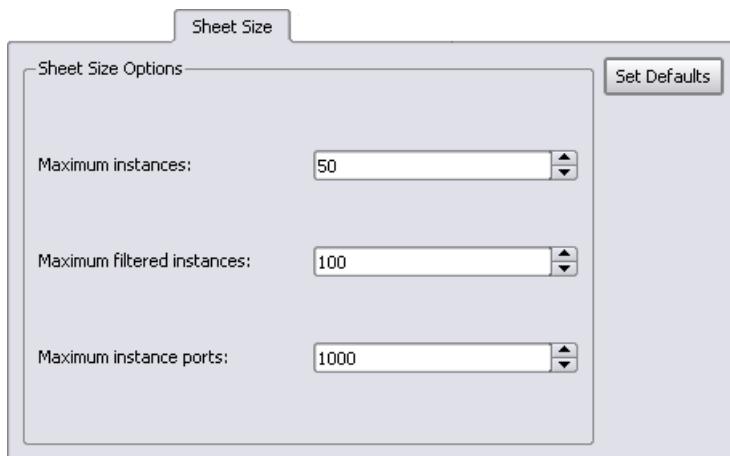
Use the Maximum Routable Endpoints option to specify the maximum number of endpoints for nets in the design to be explicitly routed to their connection endpoints. When you adjust the default value of 2000 sufficiently, improvements in performance can be seen in the HDL Analyst tool.

If the number of connection endpoints routed for the design has been reduced, you will see dashes (---) for these endpoints in the HDL Analyst (RTL or Technology) view. Note that you can still select these endpoints and perform any viable operation for these nets as shown in the RTL view below.



**Note:** Occasionally, the software does not route nets for some reason. You will see dashes (---) for these endpoints in the HDL Analyst (RTL or Technology) view. Note that you can still select these nets and perform any viable operation for them.

## Sheet Size Panel

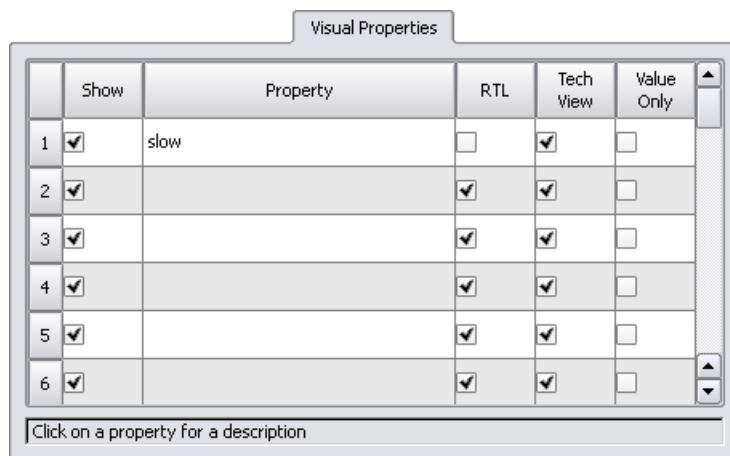


The following options are on the Sheet Size panel.

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Maximum instances          | Defines the maximum number of instances to display on a single sheet of an unfiltered schematic. If a given hierarchical level has more than this number of instances, then it will be partitioned into multiple sheets. See <a href="#">Multiple-sheet Schematics, on page 122</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Maximum filtered instances | <p>Defines the maximum number of instances to display on a filtered schematic sheet, at any visible hierarchical level. This limit is applied recursively, at each visible <i>level</i>, when</p> <ul style="list-style-type: none"><li>• the sheet itself is a level, and</li><li>• each transparent instance is a level (even if inside another transparent instance).</li></ul> <p>Whenever a given level has more child instances inside it than the value of Filtered Instances, it is divided into multiple sheets.</p> <p>(Only children are counted, not grandchildren or below. Instance A is a <i>child</i> of instance B if it is inside no other instance that is inside B.)</p> <p>In fact, at each level except the sheet itself, an additional margin of allowable child instances is added to the Maximum filtered instances value, increasing its effective value. This means that you can see more child instances than Maximum filtered instances itself implies.</p> <p>The Maximum filtered instances value must be at least the Maximum instances value. See <a href="#">Multiple-sheet Schematics, on page 122</a>.</p> |
| Maximum Instance Ports     | Defines the maximum number of instance pins to display on a schematic sheet.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

## Visual Properties Panel

Controls the display of the selected property in open HDL Analyst views. The properties are displayed as colored boxes on the relevant objects. To display these properties, the View->Visual Properties command must also be enabled. For more information about properties, see [Viewing Object Properties](#), on page 412 in the *User Guide*.

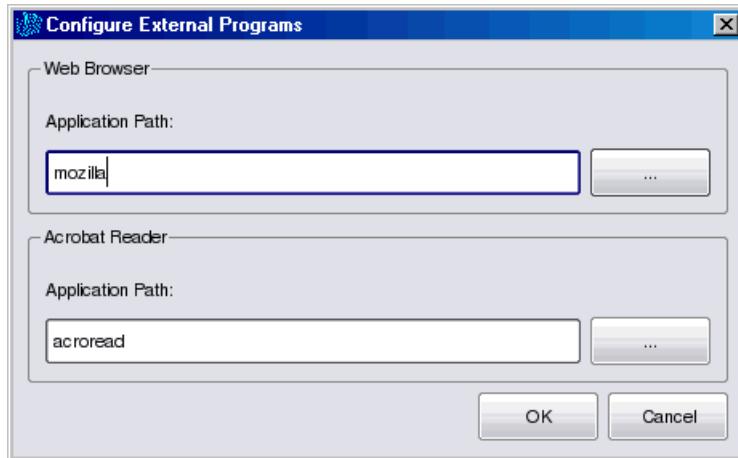


The following options are on the Visual Properties panel.

|            |                                                                                      |
|------------|--------------------------------------------------------------------------------------|
| Show       | Toggles the property name and value is displayed in a color-coded box on the object. |
| Property   | Sets the properties to display.                                                      |
| RTL        | Enables or disables the display of visual properties in the RTL view.                |
| Tech View  | Enables or disables the display of visual properties of in the Technology view.      |
| Value Only | Displays only the value of an item and not its property name.                        |

## Configure External Programs Command

This command is for Linux platforms only. It lets you specify the web browser and PDF reader for accessing Synopsys support (see [Web Menu, on page 601](#) for details) and online documentation.



| Field/Option   | Description                                                                                                                                                                                                                               |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Web Browser    | Specify your web browser as an absolute path. You can use the Browse button to locate the browser you need. The default is netscape. If your browser requires additional environment settings, you must do so outside the synthesis tool. |
| Acrobat Reader | Specify your PDF reader as an absolute path. You can use the Browse button to locate the reader you need. The default is acroread.                                                                                                        |

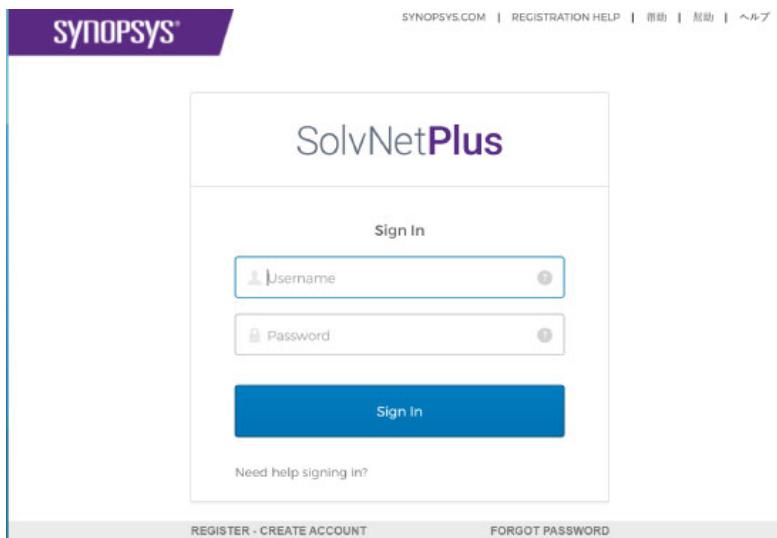
## Tech-Support Menu

The Tech-Support menu contains information and the actions you can take when you encounter problems running your designs or working with the Synopsys products.

| Command     | Description                                                                                                                                                                                                                                                                                                                                                                                |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Web Support | <p>Opens the Synopsys SolvNetPlus Support page from where you can:</p> <ul style="list-style-type: none"><li>• Log on to SolvNetPlus to request Synopsys technical support.</li><li>• Access the Synopsys Downloads, Training, and Documentation pages that have links to product information.</li></ul> <p>See <a href="#">Web Support Command</a>, on page 599 for more information.</p> |

## Web Support Command

Through the Synopsys SolvNetPlus Support page, you can access Downloads, Training, and Documentation pages on the Synopsys website as well as submit requests for technical support through SolvNetPlus. To use SolvNetPlus, you must have a user account. You are prompted for your SolvNetPlus account when you select Tech-Support->Web Support from the main menu.



Once you log on, the SolvNetPlus Support page is displayed.

The screenshot shows the Synopsys SolvNetPlus Support Community interface. At the top, there's a navigation bar with links for Home, Cases, STARS, Knowledge Base, Legacy Docs Search, and Feedback. A search bar is also present. Below the navigation, a welcome message reads "Welcome to the Synopsys Support Community! A place where you can easily find solutions and ask questions". There are four main sections: Documentation (with a document icon), Training (with a person icon), Downloads (with a download icon), and EFT (Electronic File Transfer) (with a folder icon). On the left, there's a sidebar titled "Needing My Response" showing "Cases (0)" with a search and filter button. It also displays a message "No data returned". On the right, there's a section titled "SolvNet Plus Getting Started" featuring the Synopsys logo and a link to "Read more...".

## Web Menu

This menu contains commands that access up-to-date information from Synopsys Support.

| Command                   | Description                                                                                                                                                                                                                                                                               |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Go to SolvNetPlus         | Opens the home page for the Synopsys SolvNetPlus Search support. This website contains links to useful technical information. You can search for new or updated articles or documentation, such as application notes, white papers, release notes, and other user-oriented documentation. |
| Go to Training Center     | Opens the Synopsys training web page for Synopsys products. Synopsys offers both online web-based training courses, as well as classroom training courses taught by Synopsys personnel. Select the FPGA Implementation courses from the drop-down menu.                                   |
| Synopsys Home             | Opens the Synopsys home web page for Synopsys products.                                                                                                                                                                                                                                   |
| FPGA Implementation Tools | Opens the Synopsys FPGA design solution web page for Synopsys FPGA products. You can find information about the full line of Synopsys FPGA Implementation products here.                                                                                                                  |

You can also click the **Search SolvNet** button in the Project view to open the dialog box that takes you to application notes and articles on SolvNetPlus.

# Help Menu

There are four help systems accessible from the Help menu:

- Help on the Synopsys FPGA synthesis tool (Help->Help Topics)
- Help on standard Tcl commands (Help->TCL)
- Help on using messages (Help->Error Messages)
- Help on using online help (Help->How to Use Help)

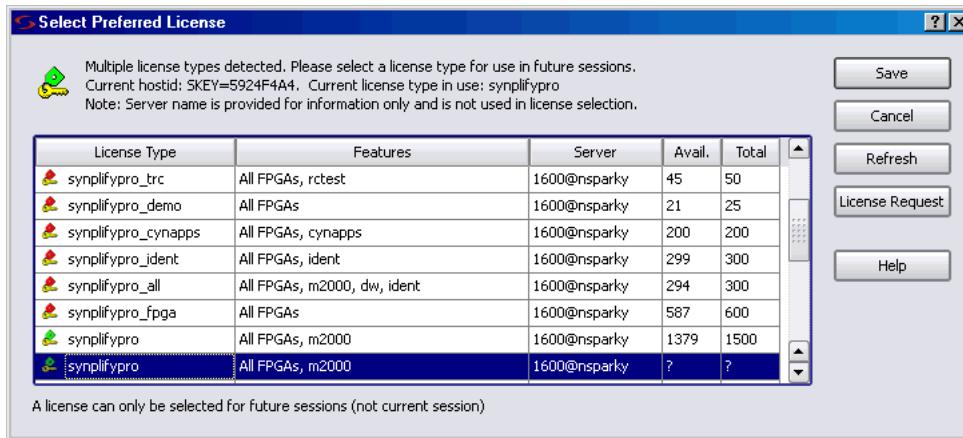
The following table describes the Help menu commands.

| Command                | Description                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Help Topics            | Displays hyperlinked online help for the product.                                                                                                                                |
| Demos & Examples       | Lets you quickly access information on key features, examples, and an attribute wizard. See <a href="#">Demos &amp; Examples, on page 67</a> for a description.                  |
| Additional Products    | Displays the Synopsys FPGA family of products with a brief description.                                                                                                          |
| How to Use Help        | Displays help on how to use Synopsys FPGA online help.                                                                                                                           |
| PDF Documents          | Displays an Open dialog box with hyperlinked PDF documentation for the product including user guide and reference manuals. You need Adobe Acrobat Reader® to view the PDF files. |
| Error Messages         | Displays help on the message viewer.                                                                                                                                             |
| TCL                    | Displays help for Tcl commands.                                                                                                                                                  |
| Tutorial               | Opens the appropriate tutorial for the synthesis tool you are using.                                                                                                             |
| Mouse Stroke Tutor     | Displays the Mouse Stroke Tutor dialog box which provides information on the available mouse strokes - see <a href="#">Using Mouse Strokes, on page 72</a> for details.          |
| License Agreement      | Displays the Synopsys software license agreement.                                                                                                                                |
| Floating License Usage | Specifies the number of floating licenses currently being used and their users.                                                                                                  |

| Command                                                                                              | Description                                                                                                                                                                                                                                                                                                                                                          |
|------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Preferred License Selection                                                                          | Displays the floating licenses that are available for your selection. See <i>Preferred License Selection Command</i> , on page 603.                                                                                                                                                                                                                                  |
| Tip of the Day                                                                                       | Displays a daily tip on how to use the Synopsys tools better. See <i>Tip of the Day Command</i> , on page 604.                                                                                                                                                                                                                                                       |
|  About this program | Displays the About dialog box, showing the tool product name, license expiration date, customer identification number, version number, and copyright.<br>Clicking the Versions button in the About dialog box displays the Version Information dialog box, listing the installation directory and the versions of all the compiler and mapper programs for the tool. |

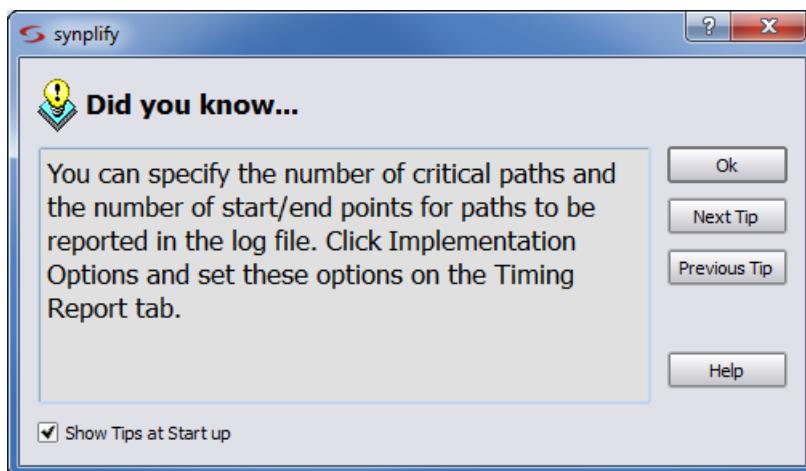
## Preferred License Selection Command

Select Help->Preferred License to display the Select Preferred License dialog box, listing the available licenses for you to choose from. Select a license from the License Type column and click Save. Close and restart the Synopsys tool, then the new session uses the preferred license you selected.



## Tip of the Day Command

Select Help->Tip of the Day to display a dialog box with a daily tip on how to best use the Synopsys tool. This dialog box also displays automatically when you first start the tool. To prevent it from redisplaying at product startup, deselect Show Tips at Startup.



## CHAPTER 6

# GUI Popup Menu Commands

In addition to the GUI menu commands described in [Chapter 5, User Interface Commands](#), the FPGA synthesis tools also have context-sensitive commands that are accessed from popup or right-click menus in different parts of the interface. Most of these commands have an equivalent menu command. This chapter only describes the unique commands that are not documented in the previous chapter.

See the following sections for details:

- [Popup Menus](#), on page 606
  - [Project View Popup Menus](#), on page 613
  - [RTL and Technology Views Popup Menus](#), on page 641

# Popup Menus

Popup menus, available by clicking the right mouse button, offer quick ways to access commonly used menu commands that are specific to the view where you click. Commands shown greyed out (dimmed) are currently inaccessible. Popup menu commands generally duplicate commands available from the regular menus, but sometimes have commands that are only available from the popup menu. The following table lists the popup menus:

| Popup Menu               | Description                                                                                                                   |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Project view             | See <a href="#">Project View Popup Menus , on page 613</a> for details.                                                       |
| SCOPE window             | Contains commonly used commands from the Edit menu.                                                                           |
| Watch Window             | See <a href="#">Watch Window Popup Menu , on page 607</a> for details.                                                        |
| Tcl window               | Contains commands from the Edit menu. For details, see <a href="#">Edit Menu Commands for the Text Editor , on page 403</a> . |
| Text Editor window       | See <a href="#">Text Editor Popup Menu , on page 607</a> for more information.                                                |
| RTL and Technology views | See <a href="#">RTL and Technology Views Popup Menus , on page 641</a> .                                                      |
| FSM viewer               | See <a href="#">FSM Viewer Popup Menu , on page 610</a> .                                                                     |
| Design Planner           | <i>Synplify Premier</i><br>See <a href="#">Design Planner Popup Menus, on page 673</a> .                                      |
| Physical Analyst         | <i>Synplify Premier</i><br>See <a href="#">Physical Analyst View Popup Menus, on page 692</a> .                               |

## Watch Window Popup Menu

*Synplify Pro, Synplify Premier*

The Watch window popup menu contains the following commands:

| Command          | Description                                                                                     |
|------------------|-------------------------------------------------------------------------------------------------|
| Configure Watch  | Displays the Log Watch Configuration dialog box, where you choose the implementations to watch. |
| Refresh          | Refreshes (updates) the window display.                                                         |
| Clear Parameters | Empties the Watch window.                                                                       |

For more information on the Watch window and the Configure Watch dialog box, see [Watch Window, on page 54](#).

## Tcl Window Popup Menu

*Synplify Pro, Synplify Premier*

The Tcl window popup menu contains the Copy, Paste, and Find commands from the Edit menu, as well as the Clear command, which empties the Tcl window. For information on the Edit menu commands available in the Tcl window, see [Edit Menu Commands for the Text Editor, on page 403](#).

## Text Editor Popup Menu

The popup menu in the Text Editor window contains the following commonly used text-editing commands from the Edit menu: Undo, Redo, Cut, Copy, Paste, and Toggle Bookmark. In addition, HDL Analyst specific commands appear when both an HDL Analyst view and its corresponding HDL source file is open. For details of these commands, see [Edit Menu Commands for the Text Editor, on page 403](#) and [HDL Analyst Menu, on page 542](#).

The following table lists the commands that are unique to the popup menu:

| Command        | Description                                                                                                                              |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Filter Analyst | Filters your design to show only the currently selected objects in the HDL text file. This is the same as HDL Analyst->Filter Schematic. |

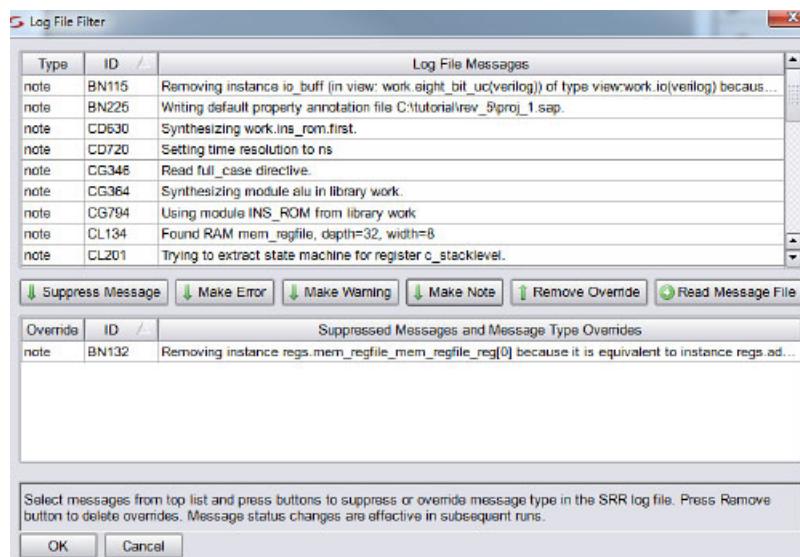
| Command           | Description                                                                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Select in Analyst | Crossprobes from the Text Editor and selects the objects in the HDL Analyst view. To use this command, the Enhanced Text Crossprobing (option must be engaged). |

## Log File Popup Menu

The popup menu in the log file contains commands that control operations in the log file. The popup menu differs when the log file is opened in the HTML mode or in the ASCII text mode.

### Log File Filter Dialog Box

The Log File Filter dialog box is available by selecting Log File Message Filter from the log file popup menu when the log file is opened in the HDML mode. The dialog box allows messages in the current session to be promoted or demoted in severity or suppressed from the log files for subsequent sessions. For additional information on using this dialog box, see [Log File Message Controls, on page 314](#) of the *User Guide*.



The following table describes the dialog box functions.

| Function                 | Description                                                                                                                                                                                                                                                               |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Log File Messages window | Displays the message ID and text and the default message type of messages generated during the current session.                                                                                                                                                           |
| Suppress Message button  | Suppresses the selected note, warning, or advisory message. The selected message is removed from the upper Log File Messages window and displayed in the lower window with the Override column indicating suppress status. Note that error messages cannot be suppressed. |
| Make Error button        | Promotes the status of the selected warning (or note) to an error. The selected message is removed from the upper Log File Messages window and displayed in the lower window with the Override column indicating error status.                                            |
| Make Warning button      | Promotes the status of the selected note to a warning. The selected message is removed from the upper Log File Messages window and displayed in the lower window with the Override column indicating warning status.                                                      |
| Make Note button         | Demotes the status of the selected warning to a note. The selected message is removed from the upper Log File Messages window and displayed in the lower window with the Override column indicating note status.                                                          |
| Remove Override button   | Removes the override status on the selected message in the lower window and returns the message to the upper Log File Messages window.                                                                                                                                    |
| Read Message File        | Select the message filter file ( <i>project.pfl</i> ) to be read for the project.                                                                                                                                                                                         |
| lower window             | Lists the status of all messages that have been promoted, demoted, or suppressed.                                                                                                                                                                                         |
| OK button                | Updates the status of any changed messages in the .pfl file. Note that you must recompile/resynthesize the design before any message status changes become effective.                                                                                                     |

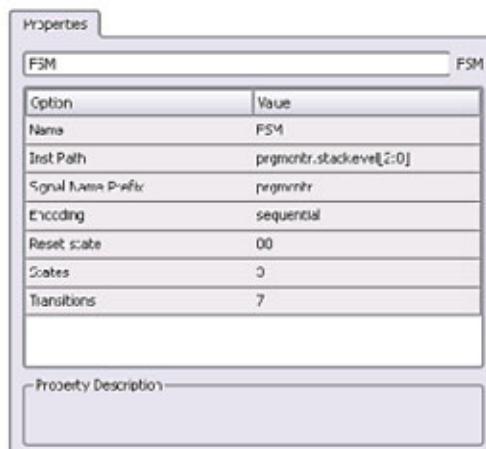
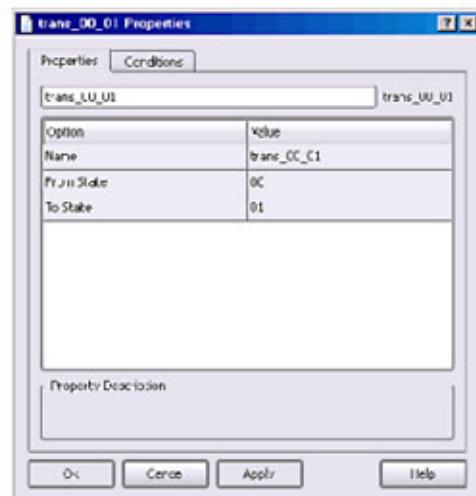
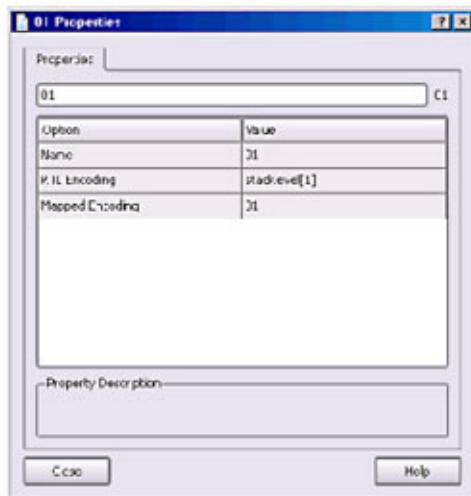
## FSM Viewer Popup Menu

*Synplify Pro, Synplify Premier*

The popup menu in the FSM Viewer contains commands that determine what is shown in the FSM Viewer. The following table lists the popup commands in the FSM Viewer.

| Command        | Description                                                                                                                                                                                                                                                                                                                                |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Properties     | Displays the Object Properties dialog box and view properties of a selected state or transition. Information about a selected transition includes the conditions enabling the transition and the identities of its origin and destination states. Information about a selected state includes its name, RTL encoding, and mapped encoding. |
| Filter         | See <a href="#">View Menu: FSM Viewer Commands , on page 417</a> .                                                                                                                                                                                                                                                                         |
| Unfilter       | See <a href="#">View Menu: FSM Viewer Commands , on page 417</a> .                                                                                                                                                                                                                                                                         |
| FSM Properties | Displays the Object Properties dialog box indicating the FSM identity and location, encoding style, reset state, and the number of states and transitions.                                                                                                                                                                                 |

FSM Properties

State properties  
(state selected)Transition properties  
(transition selected)

| Field/Option                                                                      | Description                                                                                                                                                                                     |
|-----------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | Icons indicating the object type: FSM, state, or transition.                                                                                                                                    |
| Name                                                                              | The name of the selected state or transition, or FSM if nothing is selected.                                                                                                                    |
| Inst Path                                                                         | The full name and position of the state machine in the hierarchy.                                                                                                                               |
| Signal Name Prefix                                                                | The position of the state machine in the hierarchy.                                                                                                                                             |
| Encoding                                                                          | The style of encoding used for the state machine. This can be onehot, sequential, gray, or safe. See <a href="#">syn_encoding, on page 259</a> , for information on changing the encoding type. |
| Reset State                                                                       | The initial state of the FSM: the active state after resetting.                                                                                                                                 |
| States                                                                            | The number of states in the state machine.                                                                                                                                                      |
| Transitions                                                                       | The number of transitions in the state machine.                                                                                                                                                 |
| RTL Encoding                                                                      | The name (address) of the selected state, as referred to in the RTL (HDL) file.                                                                                                                 |
| Mapped Encoding                                                                   | The encoding of the selected state.                                                                                                                                                             |
| From                                                                              | The origin state of the selected transition.                                                                                                                                                    |
| To                                                                                | The destination state of the selected transition.                                                                                                                                               |
| Conditions (min-terms)                                                            | The conditions enabling the selected transition, as defined in the RTL (HDL) file.                                                                                                              |

# Project View Popup Menus

The popup menu commands available in the Project view are context-sensitive, depending on what is currently selected and where in the view you click to open the popup menu. Most commands duplicate commands from the File, Project, Run, and Options menus.

## Project Management View Popup Commands

The following table describes the Project Management view commands that are not duplicated on other menus in the tool:

| Command                                       | Description                                                                                                                                                                                                                                                                                             |
|-----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Project Management View, No Selections</b> |                                                                                                                                                                                                                                                                                                         |
| Open Project                                  | Displays the Open Project Dialog. See <a href="#">Open Project Command , on page 401</a> .                                                                                                                                                                                                              |
| New Project                                   | Creates a new empty project in the Project Window.                                                                                                                                                                                                                                                      |
| Refresh                                       | Refreshes the display.                                                                                                                                                                                                                                                                                  |
| Project View Options                          | Displays the Project View Options dialog. See <a href="#">Project View Options Command , on page 573</a> .                                                                                                                                                                                              |
| <b>Project Selected</b>                       |                                                                                                                                                                                                                                                                                                         |
| Open as Text                                  | Opens the selected file in the Text Editor.                                                                                                                                                                                                                                                             |
| Add File                                      | Displays the Add Files to Project dialog. See <a href="#">Add Source File Command , on page 426</a> .                                                                                                                                                                                                   |
| Synthesize                                    | Compiles and maps the selected design.                                                                                                                                                                                                                                                                  |
| Compile Only                                  | Compiles the selected design.                                                                                                                                                                                                                                                                           |
| Write Output Netlist Only                     | Writes the mapped output netlist to structural Verilog (.vm) or VHDL (.vhd) format.<br>Same as enabling: <ul style="list-style-type: none"><li>• Write Mapped Verilog Netlist</li><li>• Write Mapped VHDL Netlist</li></ul> on the Implementation Results tab of the Implementation Options dialog box. |
| Arrange VHDL Files                            | Reorders the VHDL source files.                                                                                                                                                                                                                                                                         |

| Command                                | Description                                                                                                                                                                                                  |
|----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Save Project                           | Displays the Save Project As dialog box.                                                                                                                                                                     |
| Close Project                          | Closes your project.                                                                                                                                                                                         |
| <b>Project Folder or File Selected</b> |                                                                                                                                                                                                              |
| Add Folder                             | Creates a folder with the new name you specified and adds it to the Project view. See <a href="#">Add Folder Command , on page 619</a> .                                                                     |
| Rename Folder                          | Renames an existing folder with the new name you specified in the Project view. See <a href="#">Rename Folder Command , on page 619</a> .                                                                    |
| Delete Folder                          | Deletes the specified folder and all its contents as necessary. See <a href="#">Delete Folder Command , on page 619</a> .                                                                                    |
| Remove from Folder                     | Removes the selected file from its corresponding folder.                                                                                                                                                     |
| Place in Folder                        | Places the selected file into the folder you specify.                                                                                                                                                        |
| Launch Tools->Run Vendor Tool          | Launches the vendor application or Tcl procedure tool from the Project view for the selected file or folder. See <a href="#">Vendor Tool Invocation Popup Menu Command , on page 620</a> .                   |
| <b>Constraint File Selected</b>        |                                                                                                                                                                                                              |
| File Options                           | Displays the File Options dialog box. See <a href="#">File Options Popup Menu Command , on page 622</a> .                                                                                                    |
| Open                                   | Opens the SCOPE window.                                                                                                                                                                                      |
| Open as Text                           | Opens the selected file in the Text Editor.                                                                                                                                                                  |
| Copy File                              | Displays the Copy File dialog box, where you copy the selected file and add it to the current project. You specify a new name for the file. See <a href="#">Copy File Popup Menu Command , on page 624</a> . |
| Change File                            | Opens the Source File dialog box where you choose a new file to replace the selected file. See <a href="#">Change File Command , on page 429</a> .                                                           |
| Remove File From Project               | Removes the file from the project.                                                                                                                                                                           |
| <b>HDL File Selected</b>               |                                                                                                                                                                                                              |
| File Options                           | Displays the File Options dialog box. See <a href="#">File Options Popup Menu Command , on page 622</a> .                                                                                                    |

| Command                                | Description                                                                                                                                                                                                                                   |
|----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Open                                   | Opens the file in the Text Editor.                                                                                                                                                                                                            |
| Syntax Check                           | Runs a syntax check on your design code. Reports errors, warnings, or notes in the Tcl Window.                                                                                                                                                |
| Synthesis Check                        | Runs a synthesis check on your design code. This includes a syntax check and a check to see if the synthesis tool could map the design to the hardware. No optimizations are performed. Reports errors, warnings, or notes in the Tcl Window. |
| Copy File                              | Displays the Copy File dialog box, where you copy the selected file and add it to the current project. You specify a new name for the file. See <a href="#">Copy File Popup Menu Command , on page 624</a> .                                  |
| Change File                            | Opens the Source File dialog box where you choose a new file to replace the selected file. See <a href="#">Change File Command , on page 429</a> .                                                                                            |
| Remove File From Project               | Removes the file from the project.                                                                                                                                                                                                            |
| <b>Implementation Selected</b>         |                                                                                                                                                                                                                                               |
| Implementation Options                 | Displays the Implementation Options dialog box. See <a href="#">Implementation Options Command , on page 444</a> .                                                                                                                            |
| Change Implementation Name             | Displays the Implementation Name dialog box, where you rename the selected implementation. See <a href="#">Change Implementation Popup Menu Commands , on page 624</a> .                                                                      |
| New Design Plan or<br>Edit Design Plan | <i>Synplify Premier</i><br>Displays the new generation or existing Design Planner, where you can edit the design. For more information, see <a href="#">Chapter 18, Floorplanning with Design Planner</a> .                                   |
| Copy Implementation                    | Copies the selected implementation and adds it to the current project with the name you specify in the dialog box. See <a href="#">Change Implementation Popup Menu Commands , on page 624</a> .                                              |
| Remove Implementation                  | Removes the selected implementation from the project.                                                                                                                                                                                         |
| RTL View                               | Creates an RTL View based on the properties of the selected implementation.                                                                                                                                                                   |

| Command                | Description                                                                                                                                                                                    |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tech View              | Creates a Technology View based on the properties of the selected implementation.                                                                                                              |
| Add P&R Implementation | Displays the Add New Place & Route Task dialog box where you set options to run place & route after synthesis. See <a href="#">Add P&amp;R Implementation Popup Menu Command , on page 626</a> |
| Run                    | Starts a synthesis run on your design.                                                                                                                                                         |

### Identify Implementation Selected

|                          |                                                                                                                                                                                                   |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Identify Instrumentor    | Displays the Identify Instrumentor tool, so that you can instrument signals based on the RTL and SRS netlist (after compile). See <a href="#">Working with the Identify Tools, on page 1159</a> . |
| Launch Identify Debugger | Launches the Identify Debugger tool to debug the instrumented design. See <a href="#">Working with the Identify Tools, on page 1159</a> .                                                         |

### Place & Route Implementation Selected

|                          |                                                                                                                                                                                             |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add Place & Route Job    | Displays the Add New Place & Route Task dialog box, so you can set options and run placement and routing. See <a href="#">Add P&amp;R Implementation Popup Menu Command , on page 626</a> . |
| Remove Place & Route Job | Deletes the place-and-route implementation from the project.                                                                                                                                |
| Run Place & Route Job    | Runs the place-and-route job for the design.                                                                                                                                                |
| Run Backannotation Only  | Only runs backannotation with placement and timing data immediately following placement and routing.                                                                                        |

## Project Management Commands

*Synplify Pro, Synplify Premier*

The following table lists the popup commands in the Project Management views that are not available on the tool command menus. The Project Management view consists of two tabs, and the table lists the popup commands available in both tabs.

For the Design Hierarchy tab, the tools support all the project management commands listed in [Project Management View Popup Commands, on page 613](#), as well as the unique commands listed here.

| Command                                                               | Description                                                                                                                                                                                                                                                                                                |
|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Project Management View -&gt; Project Files Tab Popup Commands</b> |                                                                                                                                                                                                                                                                                                            |
| All Synplify project management commands                              | Refer to the table in <a href="#">Project Management View Popup Commands , on page 613</a> for descriptions of these commands.                                                                                                                                                                             |
| Project Options                                                       | With the project selected, displays project properties such as name and location. See <a href="#">Project Options Popup Menu Command , on page 626</a> .                                                                                                                                                   |
| Show Compile Points                                                   | Displays the compile points of the selected implementation and lets you edit them. See <a href="#">Show Compile Points Popup Menu Command , on page 625</a> .                                                                                                                                              |
| P & R Options                                                         | With a place-and-route implementation selected, displays the Options for Place & Route on Implementation dialog box, so you can change options and rerun placement and routing. See <a href="#">Options for Place &amp; Route Jobs Popup Menu Command , on page 630</a> for a description of the features. |
| Create Subproject (Instance)                                          | Makes an instance into a subproject of the top-level project. See <a href="#">Create Subproject (Instance) Command , on page 632</a> .                                                                                                                                                                     |
| Set as Black Box                                                      | When enabled, specifies that the design block be implemented as a black box during synthesis. Only available when the subproject is selected.                                                                                                                                                              |
| Design Block Source                                                   | Takes you to the design block or instance block definition in the HDL source file. Only available when the subproject is selected.                                                                                                                                                                         |
| Refresh Hierarchy                                                     | Refreshes the Design Hierarchy view after design blocks or instance blocks have changed.                                                                                                                                                                                                                   |
| Properties                                                            | Displays the design block or instance block properties. For details, see <a href="#">Design Block/Instance Properties Popup Menu Command , on page 634</a> .                                                                                                                                               |
| Hierarchical Project Options                                          | Configures synthesis run for subprojects or top-level projects. For details, see <a href="#">Hierarchical Project Options Command , on page 441</a> .<br>Same as the Project->Hierarchical Project Options command.                                                                                        |

| Command                        | Description                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add SubProject Implementations | Adds new implementations to the blocks in the top-level project. This command is only available when the top-level implementation is selected. Same as the Project-> SubProject Implementation command.<br>See <a href="#">Working with Multiple Implementations in Hierarchical Projects, on page 166</a> in the <i>User Guide</i> for information about using this command. |
| Insert SubProject              | Allows you to nest subprojects within a hierarchy. For details, see <a href="#">Insert Subproject Command , on page 442</a> .<br>Only available as a popup menu command.                                                                                                                                                                                                      |
| Subproject Parameter Sync      | Synchronizes parameters for all the subprojects from the top-level project. See <a href="#">Subproject Parameter Sync , on page 636</a> .                                                                                                                                                                                                                                     |

### Project Management View -> Design Hierarchy Tab Commands

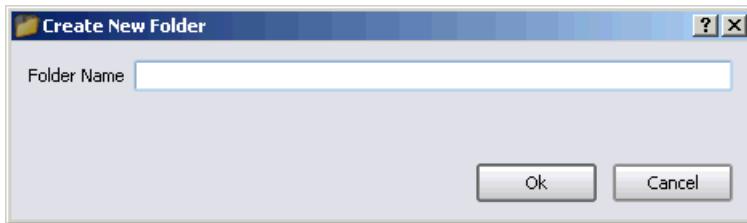
|                                      |                                                                                                                                                                                                                                                                                                                                                                                               |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Create Subproject (Design Block)     | Defines a design or instance block as a subproject of the top-level project. See <a href="#">Create Subproject Popup Menu Commands , on page 631</a> for a description.                                                                                                                                                                                                                       |
| Insert and Link Subproject to Module | Specifies a design as a subproject and links it to the top-level module. See <a href="#">Insert &amp; Link Subproject to Module Command , on page 637</a> or a description.                                                                                                                                                                                                                   |
| Set as Black Box                     | Specifies that the design block be implemented as a black box during synthesis.                                                                                                                                                                                                                                                                                                               |
| Design Block source                  | Takes you to the design block or instance block definition in the HDL source file.                                                                                                                                                                                                                                                                                                            |
| Refresh Hierarchy                    | Refreshes the Design Hierarchy view after design blocks or instance blocks have changed.                                                                                                                                                                                                                                                                                                      |
| Properties                           | Displays the design block or instance block properties. For details, see <a href="#">Design Block/Instance Properties Popup Menu Command , on page 634</a> .                                                                                                                                                                                                                                  |
| Allocate Timing and Resource Budgets | Generates the timing and resource constraints for the instance-based subproject(s) of a hierarchical design. For a description of the dialog box, see <a href="#">Allocate Timing and Resource Budgets , on page 639</a> .<br>The Tcl equivalent is <code>generate_instance_constraints</code> . For the syntax description, see <a href="#">generate_instance_constraints , on page 84</a> . |

## Project Management View Popup Folder Commands

The Project view popup menu includes commands for manipulating folders.

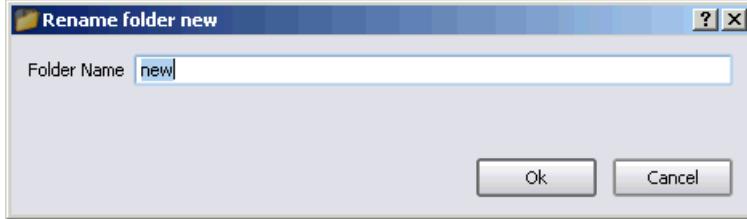
### Add Folder Command

Use this option to add a folder to the Project view.



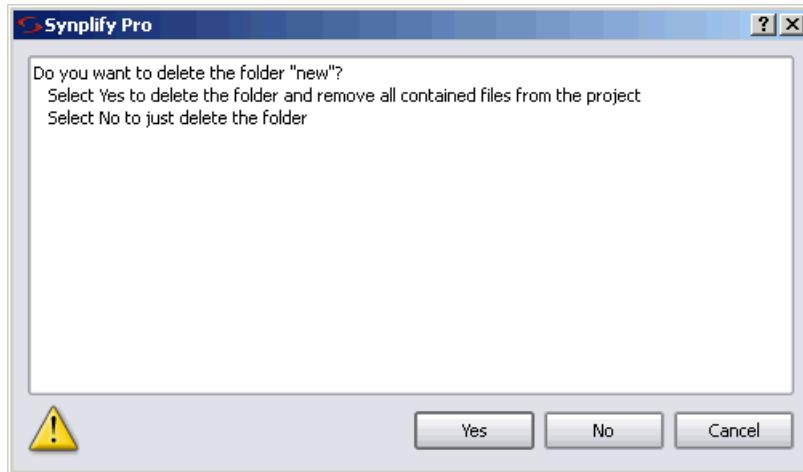
### Rename Folder Command

Use this option to rename an existing folder in the Project view.



### Delete Folder Command

Use this option to delete a folder from the Project view.



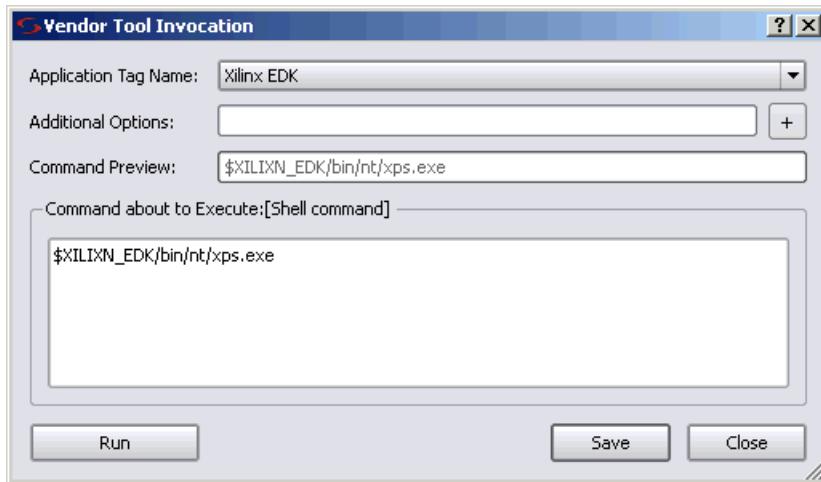
This dialog box includes the following options:

| Feature | Description                                                                                  |
|---------|----------------------------------------------------------------------------------------------|
| Yes     | Select Yes to delete the folder and all files contained in the folder from the Project view. |
| No      | Select No to delete just the folder from the Project view.                                   |
| Cancel  | Select Cancel, to discontinue the operation.                                                 |

## Vendor Tool Invocation Popup Menu Command

Use the Vendor Tool Invocation command to invoke third-party tools, such as the Embedded Development Kit (EDK) from within the Synopsys FPGA products. This allows you to modify source files or libraries added to your synthesis projects from within the third-party tool directly. Use the following dialog box to run the vendor tools.

For more information, see [Invoking Third-Party Vendor Tools, on page 799](#) in the *User Guide*.



The Vendor Tool Invocation dialog box includes the following options:

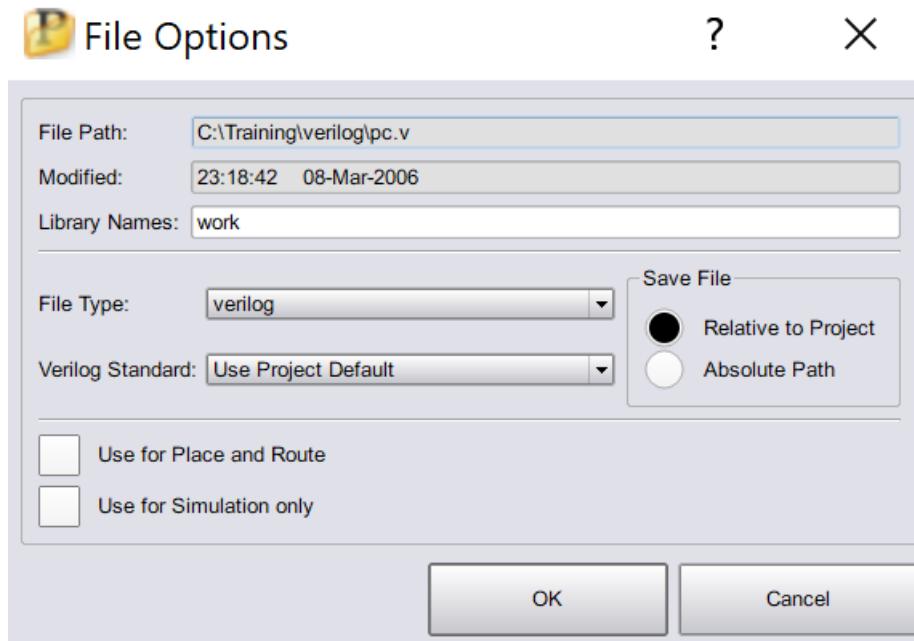
| Feature              | Description                                                                                                                                                                                                                                                                                           |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Application Tag Name | Specifies an application or Tcl procedure name. Type in the name or select a preconfigured application from the list.                                                                                                                                                                                 |
| Additional Options   | Defines any additional arguments for the Tcl procedure or third-party application. You can select arguments from the drop-down list or type them.<br>Note: For internal Synopsys tools, you must include the \$Syncode parameter.                                                                     |
| Command Preview      | Sets up the direct invocation of a third-party tool from within the FPGA synthesis tool, using the path defined for the executable in Application Name with Path or to execute the Tcl procedure from within the FPGA synthesis tool, using the path defined for the procedure in Tcl Procedure Name. |
| Run                  | The synthesis tool launches the third-party tool or runs the Tcl procedure with the arguments you specified.                                                                                                                                                                                          |

## File Options Popup Menu Command

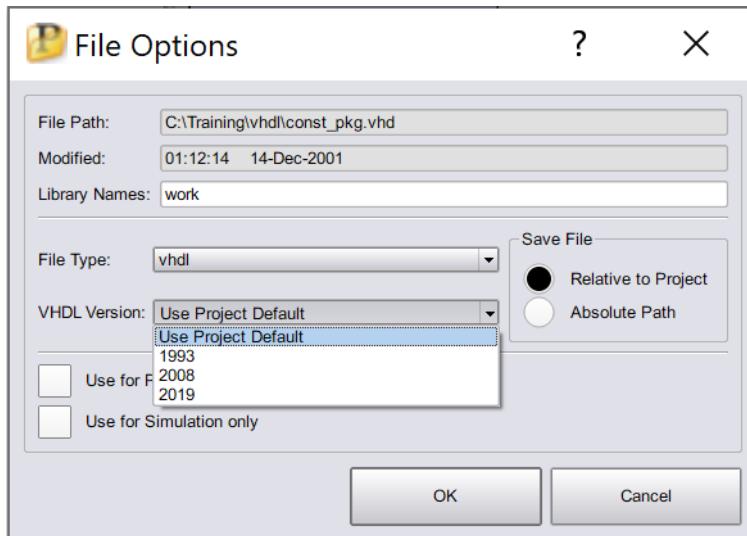
To display the File Options dialog box, right-click a project file and select File Options from the popup menu. Specify the path as relative or absolute when listing the file in the project (.prj) file and if the file is to be passed to the place-and-route tool or used only for simulation.

| Field/Option                       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| File Path                          | Path to the selected file.                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| File Type                          | <p>The folder type for the selected file. You can select the file folder type from a large list of file types.</p> <p>Changing the folder file type does <i>not</i> change the file contents or its extension; it simply places the file in the specified Project view folder. For example, if you change the file type of a VHDL file to Verilog, the file retains its Verilog extension, but is moved from the VHDL folder to the Verilog folder.</p>                         |
| Library Names                      | <p>Name of the library which must be compatible with the HDL simulator. For VHDL files, the dialog box is the same as that accessed by Project-&gt;Set VHDL Library - see <a href="#">Set VHDL Library Command , on page 429</a>.</p>                                                                                                                                                                                                                                           |
| Last modified                      | Date the file was last modified.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Save file                          | The format for the path type: choose either Relative to Project (the default) or with an Absolute Path.                                                                                                                                                                                                                                                                                                                                                                         |
| Verilog Standard<br>(Verilog only) | <p>Select the Verilog file type from the menu: Use Project Default, Verilog 95, Verilog 2001, or SystemVerilog.</p> <p>Use Project Default sets the type of the selected file to the default for the project (new projects default to SystemVerilog).</p>                                                                                                                                                                                                                       |
| Use for Place and Route            | <p>Determines if files are automatically passed to the backend place-and-route tool. The files are copied to the place-and-route implementation directory and then invoked when the place-and-route tool is run.</p> <p>For example, IP core wrappers generated by the Intel MegaWizard, as well as their associated instantiated components (including encrypted files) must be passed on to the place-and-route tool since they are not written out to the final netlist.</p> |
| Use for Simulation Only            | Determines if files are only to be used for simulation. For example, files such as test benches containing HDL constructs used only for simulation can be specified using this option.                                                                                                                                                                                                                                                                                          |

The following is the Verilog dialog box:



The following is the VHDL dialog box:



From the File Options dialog box, you can now specify the VHDL version (1993, 2008, or 2019) on each file, using the VHDL Version option.

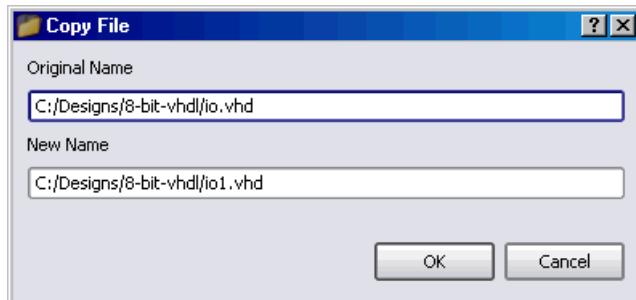
---

**Note:** If any file is compiled under VHDL 2008 or VHDL 2019, then the IEEE packages used for the entire project will be the 2008 version. For example, while the parsing rules would be VHDL 1993 for a file set to VHDL 1993, any methods used from IEEE packages would come from the VHDL 2008 IEEE packages.

---

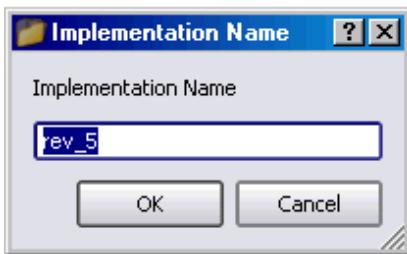
## Copy File Popup Menu Command

With a file selected, select the Copy File popup menu command to copy the selected file and add it to the current project. This displays the Copy File dialog box where you specify the name of the new file.



## Change Implementation Popup Menu Commands

With an implementation selected, right-click and select the Change Implementation Name or Copy Implementation popup menu commands to display a dialog box where you specify the new name.

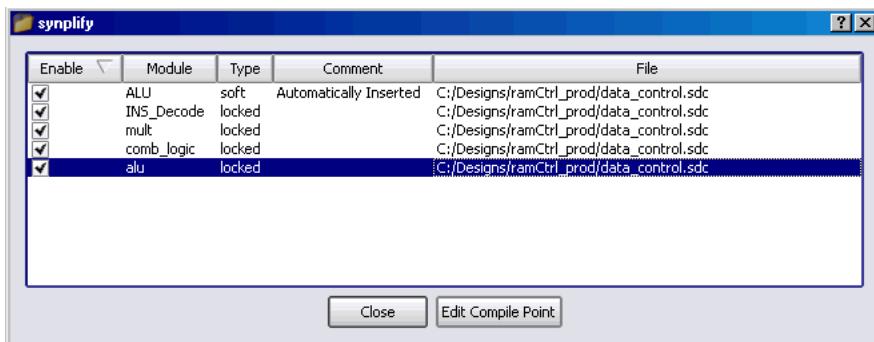


## Show Compile Points Popup Menu Command

*Synplify Pro, Synplify Premier*

With an implementation selected, select the Show Compile Points popup menu command to display the Compile Points dialog box and view or edit the compile points of the selected implementation.

Compile points are only available for certain technologies. For more information on compile points and the compile-point synthesis flow, see [Compile Point Types, on page 610](#) and [Synthesizing Compile Points, on page 626](#) of the *User Guide*.



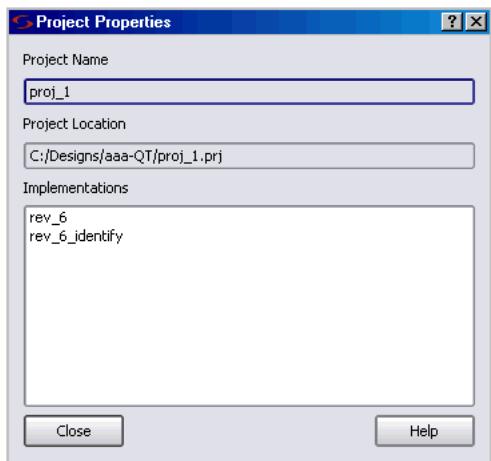
The columns **Enable**, **Module**, **Type**, and **Comment** in the dialog box correspond to the columns **Enabled**, **Module**, **Type**, and **Comment** in the SCOPE spreadsheet for the compile point. The **File** column lists the top-level constraint file where the compile point is defined.

To open and edit the SCOPE spreadsheet for a compile point, either double-click the row of the compile point or select it and click the Edit Compile Point button.

## Project Options Popup Menu Command

With a project selected, select the Project Options popup menu command to display the Project Properties dialog box and change the implementation of a project.

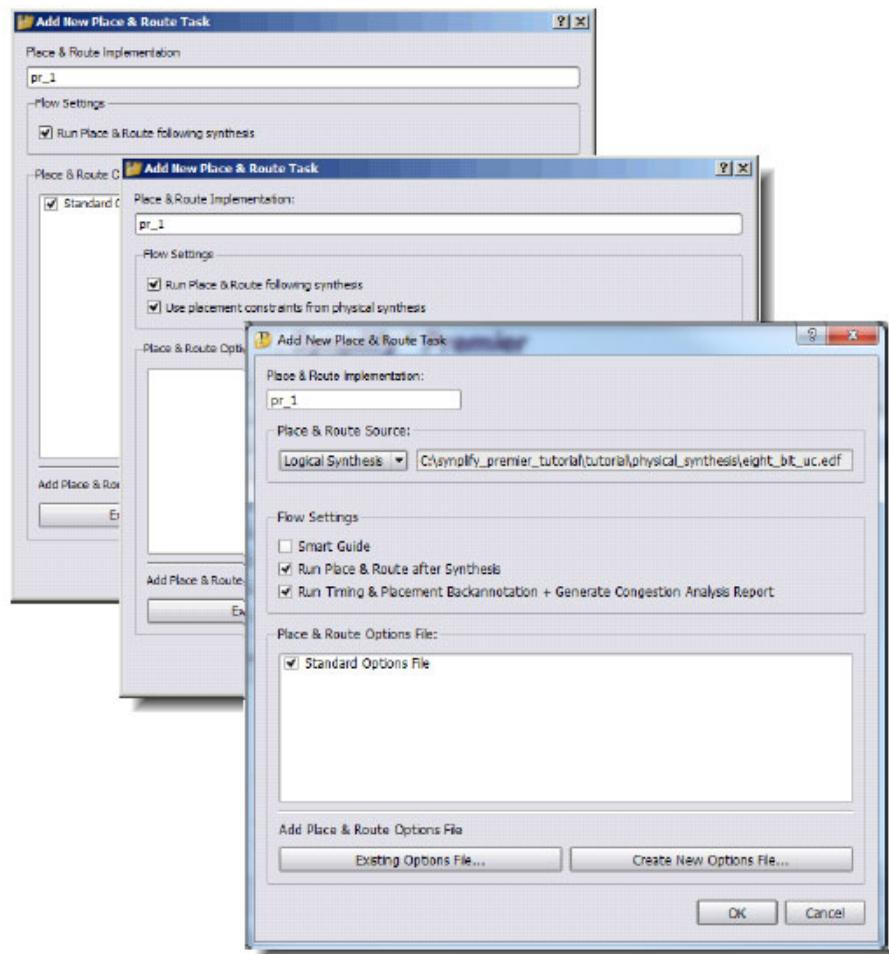
In the dialog box, select an implementation in the Implementations list, then click OK or Apply to make it the active implementation of the project.



## Add P&R Implementation Popup Menu Command

*Synplify Pro, Synplify Premier*

Displays the Add New Place & Route Task dialog box. For information about using this command for place-and-route encapsulation, see [Running P&R Automatically after Synthesis, on page 1102](#) in the *User Guide*.



| Command                               | Description                                                                                                   |
|---------------------------------------|---------------------------------------------------------------------------------------------------------------|
| Place & Route Implementation Name     | Enter a name for the place & route implementation. Do not use spaces for the implementation name.             |
| Flow Settings                         |                                                                                                               |
| Run Place & Route following synthesis | Enable/disable the running of the place & route tool from the synthesis tool immediately following synthesis. |
| Smart Guide                           | Enable/disable running the SmartGuide flow.                                                                   |

| Command                                                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Run Timing & Placement Backannotation + Generate Congestion Analysis Report | <p><i>Synplify Premier</i></p> <p>Enable/disable backannotation of placement and timing data immediately following place &amp; route and generate a congestion analysis report.</p>                                                                                                                                                                                                                                                                                                                          |
| Use placement constraints from physical synthesis                           | <p>This option is only available in the Synplify Premier tools for Intel Stratix II, Stratix II GX, Stratix III, Stratix IV, Stratix IV GT, and Stratix V devices.</p> <ul style="list-style-type: none"> <li>• By default, the Quartus II place-and-route tool uses the placement constraints forward annotated from the physical synthesis tool.</li> <li>• When this option is disabled (not checked), the Quartus II place-and-route tool determines how it will handle physical constraints.</li> </ul> |
| Place & Route Options File                                                  | <p>This option lets you specify a place &amp; route options file. You can select either the:</p> <ul style="list-style-type: none"> <li>• Smart Guide Options File - use this option to run the Xilinx SmartGuide incremental place-and-route flow.</li> <li>• Standard Options File - use this option to run the standard synthesis place-and-route flows.</li> </ul> <p>For more information about the SmartGuide flow, see the <a href="#">Xilinx SmartGuide Flow, on page 1139</a>.</p>                  |
| Add Place & Route Options File<br>Existing Options File                     | <p>This option opens the Select Place &amp; Route option file dialog box where you browse for an existing place &amp; route options file. See <a href="#">Running P&amp;R Automatically after Synthesis, on page 1102</a> for information about using this feature.</p>                                                                                                                                                                                                                                      |

| Command                                                   | Description                                                                                                                                                                                                                                               |
|-----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                           |                                                                                                                                                                                                                                                           |
| Add Place & Route Options File<br>Create New Options File | This option opens the Create Place & Route Options File dialog box where you specify a new place & route options file. See <a href="#">Running P&amp;R Automatically after Synthesis, on page 1102</a> for information about creating a new options file. |



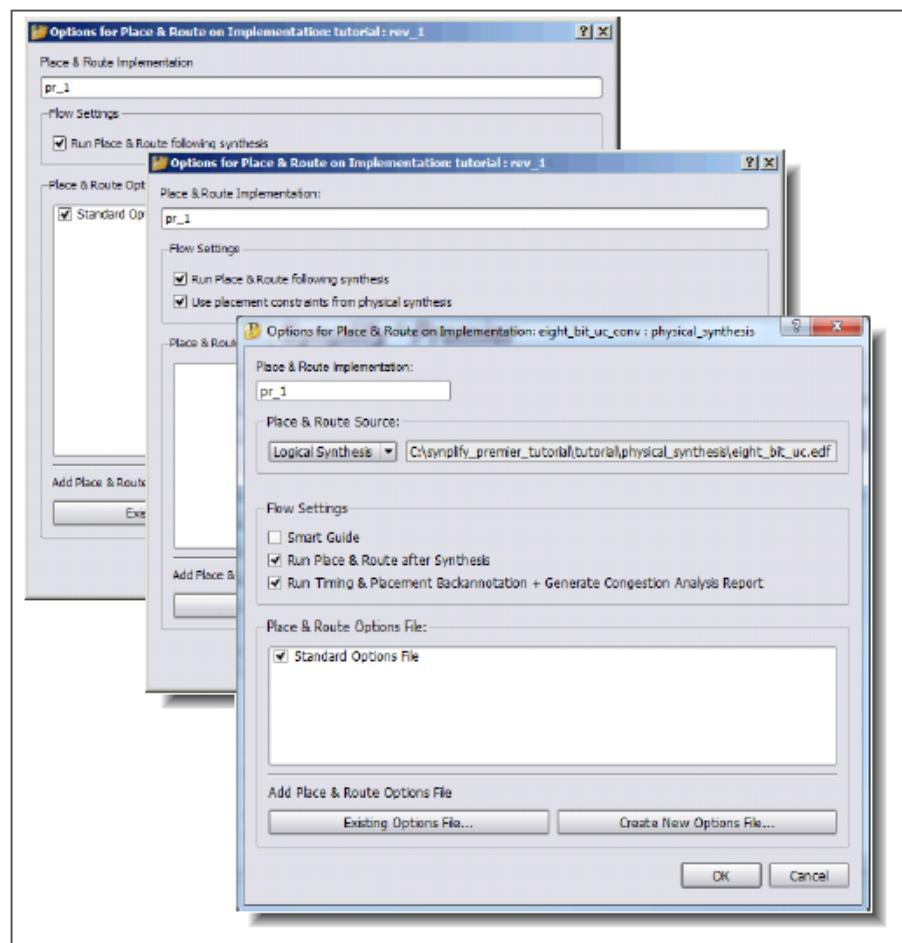
Once the par implementation is created, then you can right-click and perform any of the following options:

- P&R Options—See [Options for Place & Route Jobs Popup Menu Command](#), on page 630.
- Add Place & Route Job—See [Add P&R Implementation Popup Menu Command](#), on page 626.

- Run Place & Route Job—Runs the place-and-route job.

## Options for Place & Route Jobs Popup Menu Command

You can select a place-and-route job for a particular implementation, easily change options and then rerun the job. These options are the same found on the Options for Place & Route on Implementation dialog box. For a description of these options, see [Add P&R Implementation Popup Menu Command, on page 626](#).



## Create Subproject Popup Menu Commands

*Synplify Pro, Synplify Premier*

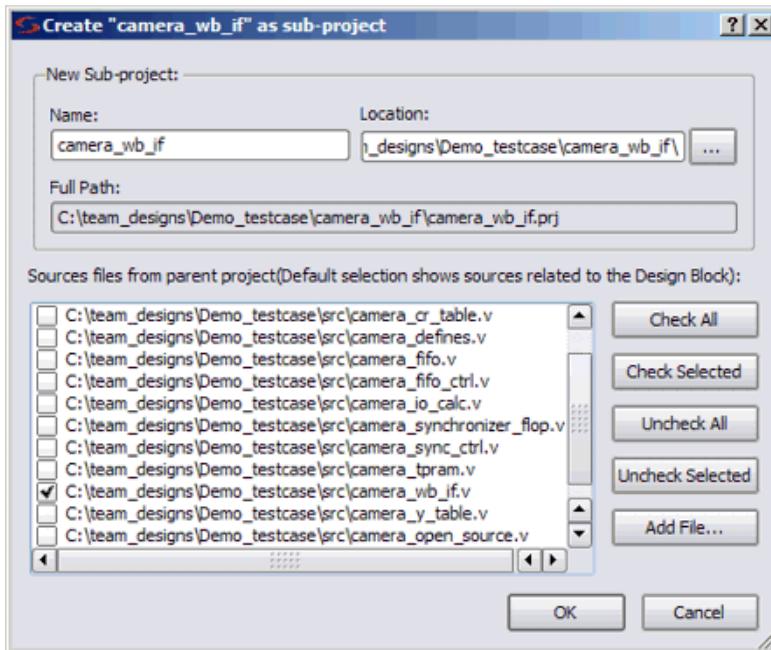
There are two subproject creation commands:

- [Create Subproject \(Design Block\)](#), on page 631
- [Create Subproject \(Instance\) Command](#), on page 632

### Create Subproject (Design Block)

The Create Subproject (Design Block) command displays the Create “*blockName*” as Subproject dialog box, which lets you specify a name, location, and source files for the subproject you are creating. You can create a subproject for an instance block or a design block.

For more information about using this command, refer to [Creating Hierarchical Subprojects by Exporting Blocks](#), on page 149 in the *User Guide*.



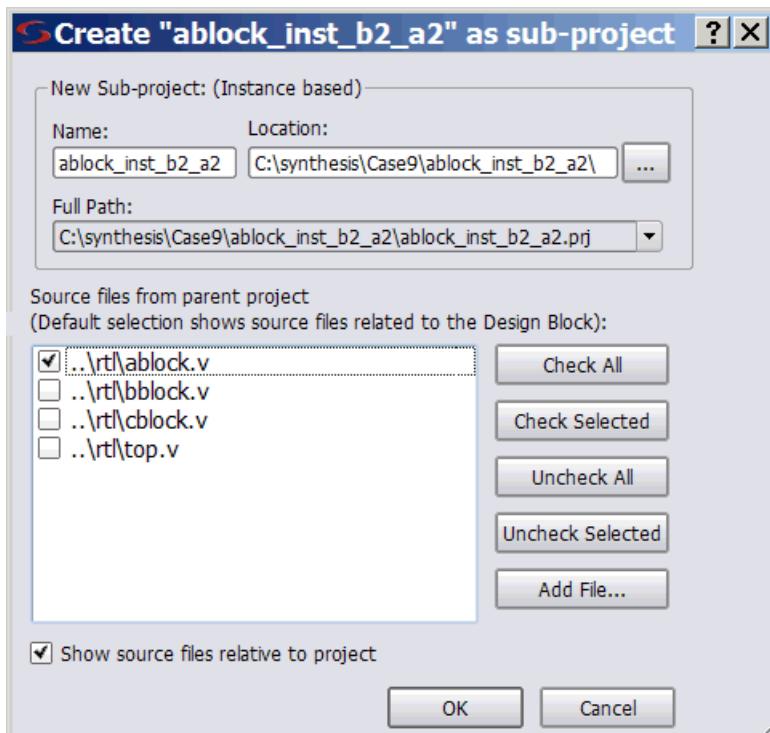
The following table describes the options:

| Option                           | Description                                                                                                                                                                                                                                                      |
|----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Project Name                     | Specifies the name of the block subproject to be created for the block.                                                                                                                                                                                          |
| Project Location                 | Specifies the location for the block subproject.                                                                                                                                                                                                                 |
| Full Path                        | Specifies the full path name of the subproject file.                                                                                                                                                                                                             |
| Source files from parent project | Lists the source file(s) from the parent project. The selected design block is enabled by default in the source file display window.                                                                                                                             |
| Check All                        | Selects all source files listed in the source file display window. The selected source files are added to the block project when you click OK.                                                                                                                   |
| Check Selected                   | Re-selects the recently unchecked source files in the source file display window. The selected source files are added to the block project when you click OK.                                                                                                    |
| Uncheck all                      | Disables all the source files selected in the source file display window. Deselected files are not added to the block project when you click OK.                                                                                                                 |
| Uncheck Selected                 | Disables the selected source files in the source file display window. Deselected files are not added to the block project when you click OK.                                                                                                                     |
| Add File                         | Opens the Add Files to Project dialog box, so you can select source files to add to the displayed list. The compiler might not always identify all the source files that belong to the parent project, but you can use Add File to add the missing source files. |

## Create Subproject (Instance) Command

The Create Subproject (Instance) command displays the Create “*instanceName*” as Subproject dialog box, which lets you specify a name, location, and source files for the subproject you are creating. You can create a subproject for an instance block.

For more information about using this command, refer to [Creating Hierarchical Subprojects by Exporting Instances, on page 152](#) in the *User Guide*.



The following table describes the options:

| Option                           | Description                                                                                                                                                                                                                                                                                                    |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Project Name                     | Specifies the name of the subproject to be created for the instance.                                                                                                                                                                                                                                           |
| Project Location                 | Specifies the location for the instance subproject.                                                                                                                                                                                                                                                            |
| Full Path                        | Specifies the full path name of the subproject file.                                                                                                                                                                                                                                                           |
| Source files from parent project | Lists the source file(s) from the parent project. The selected instance is enabled by default in the source file display window. The synthesis tool can miss auxiliary files such as `include files that define macros or packages. You must manually add these files by checking the corresponding check box. |
| Check All                        | Selects all source files listed in the source file display window. The selected source files are added to the instance project when you click OK.                                                                                                                                                              |

| Option           | Description                                                                                                                                                                                                                                                      |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Check Selected   | Re-selects the recently unchecked source files in the source file display window. The selected source files are added to the instance project when you click OK.                                                                                                 |
| Uncheck all      | Disables all the source files selected in the source file display window. Deselected files are not added to the instance project when you click OK.                                                                                                              |
| Uncheck Selected | Disables the selected source files in the source file display window. Deselected files are not added to the instance project when you click OK.                                                                                                                  |
| Add File         | Opens the Add Files to Project dialog box, so you can select source files to add to the displayed list. The compiler might not always identify all the source files that belong to the parent project, but you can use Add File to add the missing source files. |

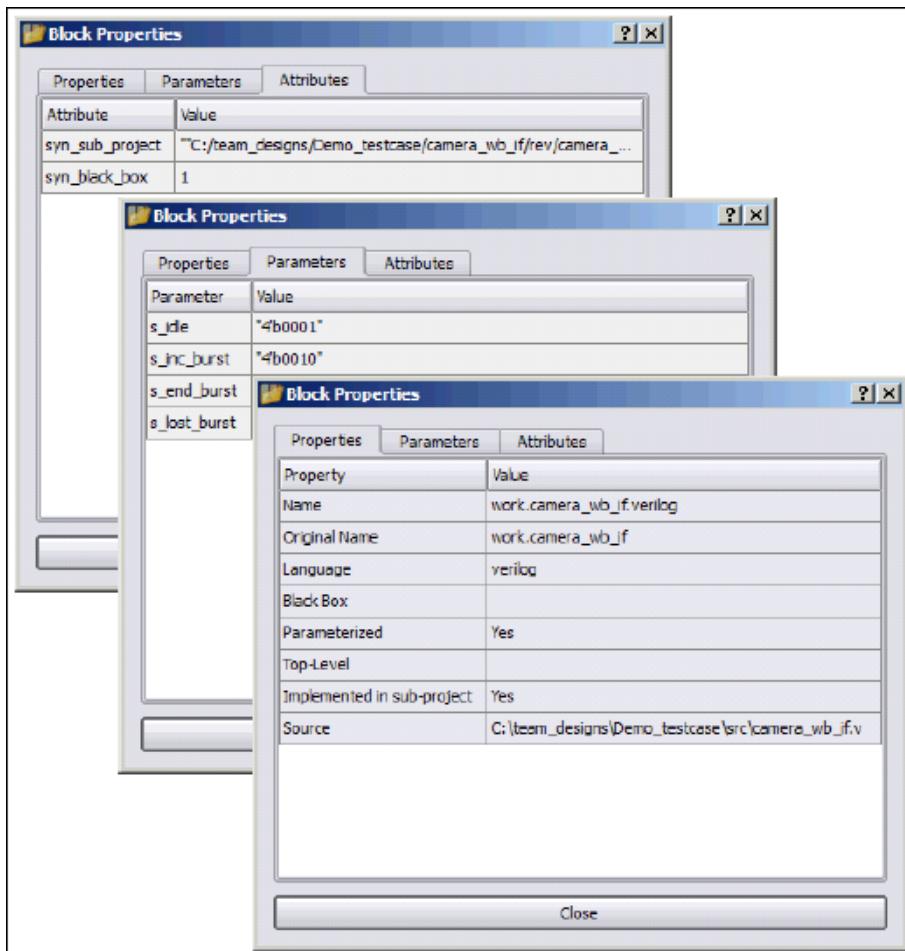
## Design Block/Instance Properties Popup Menu Command

*Synplify Pro, Synplify Premier*

The Properties dialog box in the Design Hierarchy view displays properties for a hierarchical design block or instance block. The properties displayed vary, according to the status of the design block or instance block. The Properties tab is always displayed and lists properties such as the following:

- Current block name
- Original block name
- Language used for the block, such as Verilog
- The kind of block. For example, it could be any of the following:
  - A black box
  - Parameterized (the block can have multiple combinations of parameters)
  - The top-level block
  - Implemented as a subproject
- Block source file

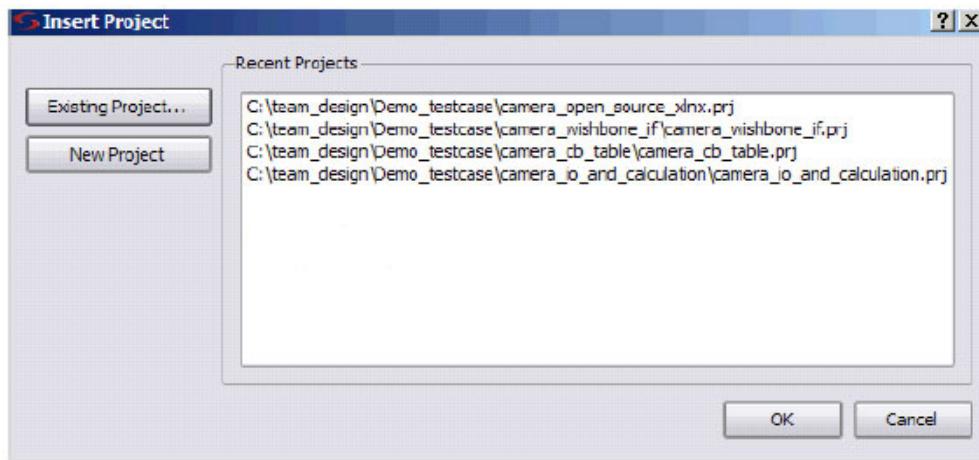
The Parameters or Attributes tabs are displayed for some design or instance blocks. Parameters are user-defined variables in the HDL source code, such as register width. Attributes can be specified in the HDL source or created by the compiler, such as `syn_black_box` or `syn_sub_project`. See the following figure.



## Insert Subproject Command

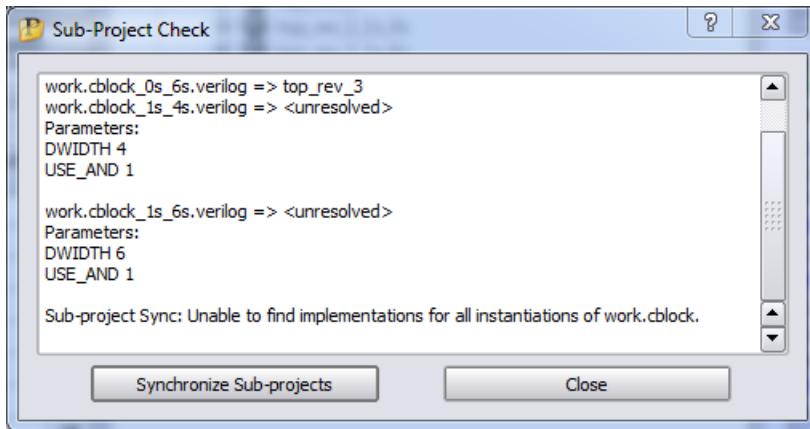
*Synplify Pro, Synplify Premier*

This command in the Project Files view lets you nest subprojects within a subproject hierarchy. To do this, right-click on a subproject and select Insert Subproject from the popup menu. You can add an existing project or a new project from the Insert Project dialog box. Then, begin adding design files to your subproject after you have created the new project.



## Subproject Parameter Sync

The Subproject Parameter Sync command synchronizes parameter properties for all the subprojects from the top-level project. To do this, simply click the Synchronize Subprojects button shown on the dialog box.



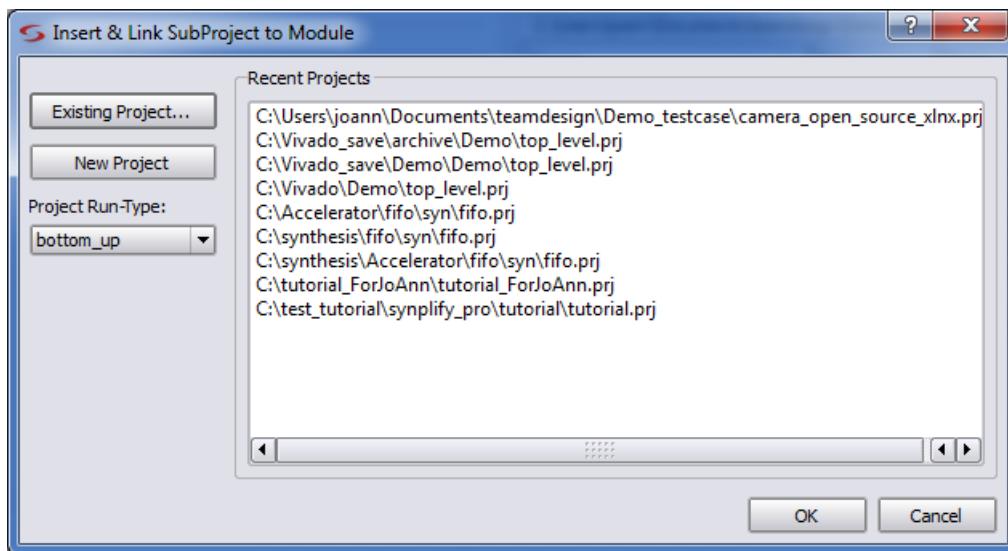
## Insert & Link Subproject to Module Command

*Synplify Pro, Synplify Premier*

This popup command in the Design Hierarchy view adds the specified design module/instance as a subproject and links it to the top-level module.

## Insert & Link Subproject to Design Block

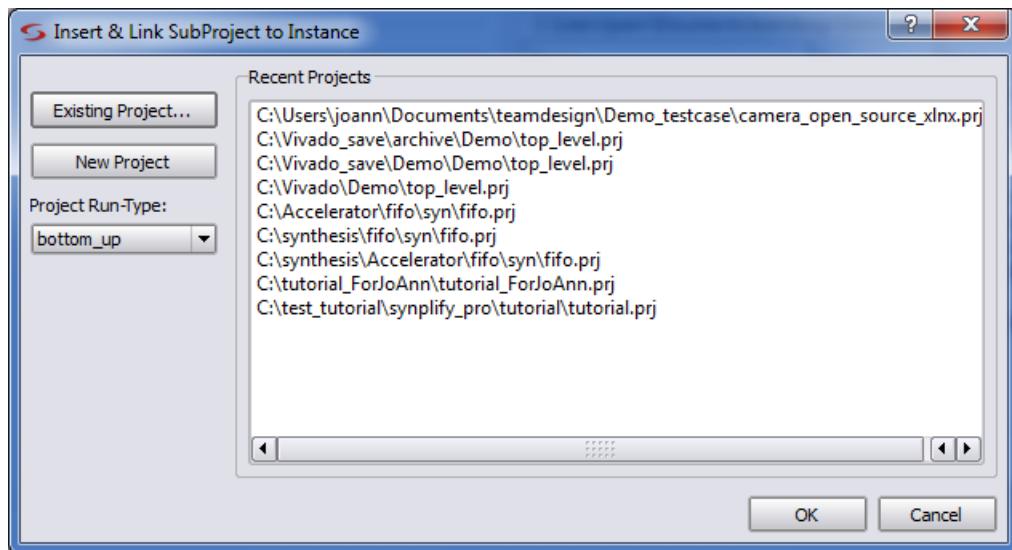
Highlight the desired module, right-click and Insert & Link Subproject to Design Block. The following dialog box appears.



| Option           | Description                                                                                                |
|------------------|------------------------------------------------------------------------------------------------------------|
| Existing Project | Adds the specified module to an existing subproject and links it to the top-level module.                  |
| New Project      | Adds the specified module to a new subproject and links it to the top-level module.                        |
| Project Run-Type | Specifies the run type for how you want the modules synthesized for the subproject: top-down or bottom-up. |

## Insert & Link Subproject to Instance

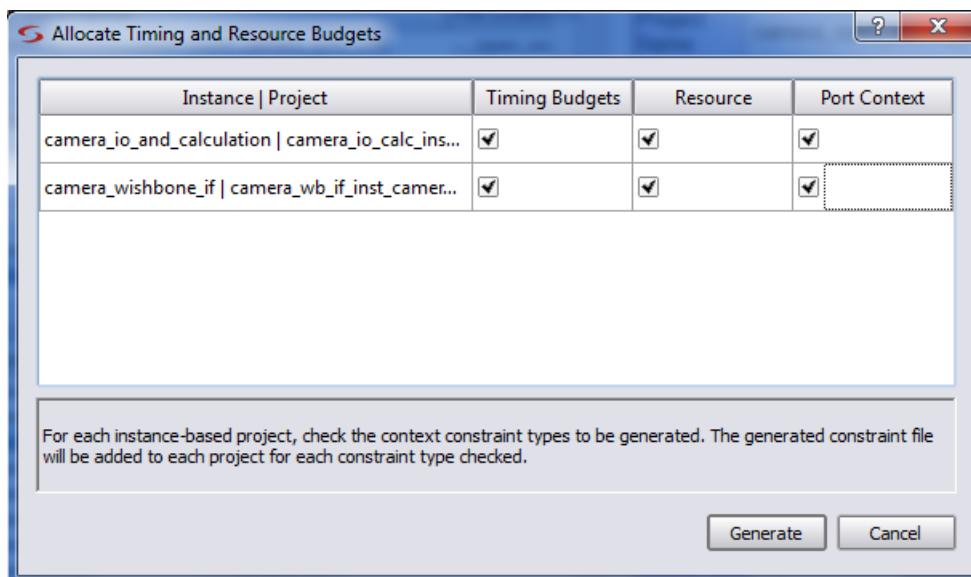
Highlight the desired module, right-click and Insert & Link Subproject to Instance. The following dialog box appears.



| Option           | Description                                                                                                  |
|------------------|--------------------------------------------------------------------------------------------------------------|
| Existing Project | Adds the specified instance to an existing subproject and links it to the top-level module.                  |
| New Project      | Adds the specified instance to a new subproject and links it to the top-level module.                        |
| Project Run-Type | Specifies the run type for how you want the instances synthesized for the subproject: top-down or bottom-up. |

## Allocate Timing and Resource Budgets

The Allocate Timing and Resource Budgets command generates the timing and resource constraints for the instance-based subproject(s) of a hierarchical design. You can access this command by right-clicking on an instance in the Design Hierarchy view or by right-clicking anywhere in the Design Hierarchy view.



| Option           | Description                                                                                                                                                                                                                                                                                        |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Instance/Project | Specifies the name of the instance-based subproject.                                                                                                                                                                                                                                               |
| Timing Budgets   | When enabled, sets initial timing constraints for the instance-based subproject, based on the top-level constraints. See <a href="#">Setting Initial Timing Budgets for Instance-Based Subprojects, on page 174</a> in the User Guide for information on using this option.                        |
| Resource         | When enabled, allocates RAM and DSP resources to the instance-based subproject, based on top-level resources. See <a href="#">Allocating Resources for Instance-Based Subprojects, on page 172</a> in the User Guide for information on using this option.                                         |
| Port Context     | When enabled, generates port context information, such as ports tied to a fixed value or unused ports for the instance-based subproject with the bottom-up flow. See <a href="#">Generating Port Information for Instance-Based Subprojects, on page 176</a> for information on using this option. |

# RTL and Technology Views Popup Menus

Some commands are only available from the popup menus in the RTL and Technology views, but most of the commands are duplicates of commands from the HDL Analyst, Edit, and View menus. The popup menus in the RTL and Technology views are nearly identical. See the following:

- [Hierarchy Browser Popup Menu Commands](#), on page 641
- [RTL View and Technology View Popup Menu Commands](#), on page 641
- [Simulation Popup Menu Commands in the RTL/Technology View](#), on page 647

## Hierarchy Browser Popup Menu Commands

The following commands become available when you right-click in the Hierarchy Browser of an RTL or Technology view. The Filter, Hide Instances, and Unhide Instances commands are the same as the corresponding commands in the HDL Analyst menu. The following commands are unique to this popup menu.

| Command               | Description                                                                                                                                                               |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Collapse All          | Collapses all trees in the Hierarchy Browser.                                                                                                                             |
| Filter                | Highlights and filters objects such as ports, instances, and primitives in the HDL analyst window.                                                                        |
| Reload                | Refreshes the Hierarchy Browser. Use this if the Hierarchy Browser and schematic view do not match.                                                                       |
| Hide/Unhide Instances | Hides or unhides selected instances in the HDL analyst window. For more information on hidden instances, see <a href="#">Hidden Hierarchical Instances</a> , on page 113. |

## RTL View and Technology View Popup Menu Commands

The commands on the popup menu are context-sensitive, and vary depending on the object selected, the kind of view, and where you click. In general, if you have a selected object and you right-click in the background, the menu includes global commands as well as selection-specific commands for the objects. In the Synplify Premier tool, physical regions are treated as

hierarchical instances by the HDL Analyst tool. The popup commands available for a selected region are the same as those available for a selected instance.

Most of the commands duplicate commands available on the HDL Analyst menu (see [HDL Analyst Menu, on page 542](#)). The following table lists the unique commands.

### Common Commands

| Command                                 | See ...                                                                                                                                                                                                                                                                                                         |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show Critical Path                      | <a href="#">HDL Analyst Menu: Timing Commands</a> , on page 549.                                                                                                                                                                                                                                                |
| Timing Analyst                          | <a href="#">HDL Analyst Menu: Timing Commands</a> , on page 549.                                                                                                                                                                                                                                                |
| Find                                    | <a href="#">Find Command (HDL Analyst)</a> , on page 407.                                                                                                                                                                                                                                                       |
| Filter Schematic                        | <a href="#">HDL Analyst Menu: Filtering and Flattening Commands</a> , on page 546.                                                                                                                                                                                                                              |
| Push/Pop Hierarchy                      | <a href="#">HDL Analyst Menu: RTL and Technology View Submenus</a> , on page 543.                                                                                                                                                                                                                               |
| Select All Schematic                    | <a href="#">HDL Analyst Menu: Selection Commands</a> , on page 553.                                                                                                                                                                                                                                             |
| Select All Sheet                        | <a href="#">HDL Analyst Menu: Selection Commands</a> , on page 553.                                                                                                                                                                                                                                             |
| Unselect All                            | <a href="#">HDL Analyst Menu: Selection Commands</a> , on page 553.                                                                                                                                                                                                                                             |
| Flatten Schematic                       | <a href="#">HDL Analyst Menu: Filtering and Flattening Commands</a> , on page 546.                                                                                                                                                                                                                              |
| Unflatten Current Schematic             | <a href="#">HDL Analyst Menu: Filtering and Flattening Commands</a> , on page 546.                                                                                                                                                                                                                              |
| HDL Analyst Options                     | <a href="#">HDL Analyst Options Command</a> , on page 587.                                                                                                                                                                                                                                                      |
| SCOPE->Edit Attributes<br>(object name) | Opens a SCOPE window where you can enter attributes for the selected object. It displays the Select Constraint File dialog box ( <a href="#">Edit Attributes Popup Menu Command</a> , on page 646), where you select the constraint file to edit. If no constraint file exists, you are prompted to create one. |

|                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SCOPE->Edit Compile Point Constraints (module <i>moduleName</i> ) | <i>Synplify Pro, Synplify Premier</i><br>For technologies that support compile points, it opens a SCOPE window where you can enter constraints for the selected compile point. It displays the Select Compile Point Definition File dialog box and lets you create or edit a compile-point constraint file for the selected region or instance. See <a href="#">Edit Attributes Popup Menu Command</a> , on page 646. |
| SCOPE->Edit Module Constraints (module <i>moduleName</i> )        | <i>Synplify Pro, Synplify Premier</i><br>Opens a SCOPE window so you can define module constraints for the selected module). If you do not have a constraint file, it prompts you to create one. The file created is a separate, module-level constraint file.                                                                                                                                                        |
| Simulation                                                        | <i>Synplify Premier</i><br>Allows you to watch nets in the Simulation Panel and annotate the following values: <ul style="list-style-type: none"><li>• None</li><li>• Previous, Current, Next</li><li>• Current</li></ul> For more information, see <a href="#">Simulation Popup Menu Commands in the RTL/Technology View</a> , on page 647.                                                                          |

**Instance Selected**

| Command                    | See ...                                                       |
|----------------------------|---------------------------------------------------------------|
| Isolate Paths              | <a href="#">Isolate Paths</a> , on page 550.                  |
| Expand Paths               | <a href="#">Hierarchical-&gt;Expand Paths</a> , on page 544.  |
| Current Level Expand Paths | <a href="#">Current Level-&gt;Expand Paths</a> , on page 545. |
| Show Context               | <a href="#">Show Context</a> , on page 550.                   |
| Hide Instance              | <a href="#">Hide Instances</a> , on page 550.                 |
| Unhide Instance            | <a href="#">Unhide Instances</a> , on page 550.               |
| Show All Hier Pins         | <a href="#">Show All Hier Pins</a> , on page 551.             |
| Dissolve Instance          | <a href="#">Dissolve Instances</a> , on page 551.             |
| Dissolve to Gates          | <a href="#">Dissolve to Gates</a> , on page 551.              |

**Generate Dependent File List** Generates a comprehensive list of all of the dependent files for a subproject. For details, see [Generating Dependent File Lists, on page 165](#) in the *User Guide*.

### Port Selected

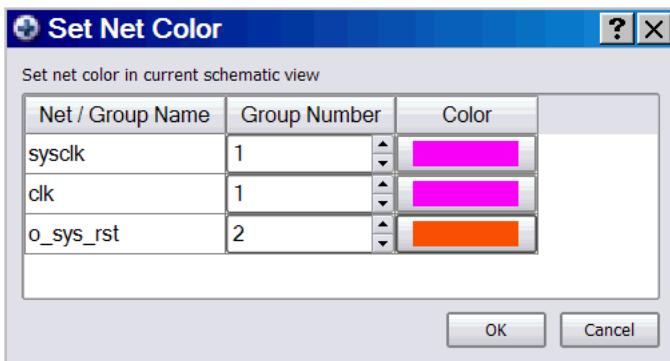
| Command                                | See ...                                                                  |
|----------------------------------------|--------------------------------------------------------------------------|
| Expand to Register/Port                | <a href="#">Hierarchical-&gt;Expand to Register/Port , on page 544.</a>  |
| Expand Inwards                         | <a href="#">Hierarchical-&gt;Expand Inwards , on page 544.</a>           |
| Current Level->Expand                  | <a href="#">Current Level-&gt;Expand , on page 545.</a>                  |
| Current Level->Expand to Register/Port | <a href="#">Current Level-&gt;Expand to Register/Port , on page 545.</a> |
| Current Level->Expand Paths            | <a href="#">Current Level-&gt;Expand Paths , on page 545.</a>            |
| Properties                             | <a href="#">Properties Popup Menu Command, on page 645.</a>              |

### Net Selected

| Command                             | See ...                                                                                                                                    |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Goto Net Driver                     | <a href="#">Hierarchical-&gt;Goto Net Driver , on page 545.</a>                                                                            |
| Select Net Driver                   | <a href="#">Hierarchical-&gt;Select Net Driver , on page 545.</a>                                                                          |
| Select Net Instances                | <a href="#">Hierarchical-&gt;Select Net Instances , on page 545.</a>                                                                       |
| Current Level->Goto Net Driver      | <a href="#">Current Level-&gt;Goto Net Driver , on page 545.</a>                                                                           |
| Current Level->Select Net Driver    | <a href="#">Current Level-&gt;Select Net Driver , on page 545.</a>                                                                         |
| Current Level->Select Net Instances | <a href="#">Current Level-&gt;Select Net Instances , on page 545.</a>                                                                      |
| Set Net Color                       | Sets the color of the selected net from a color pallet.<br>For details, see <a href="#">Set Net Color Popup Menu Command, on page 645.</a> |

## Set Net Color Popup Menu Command

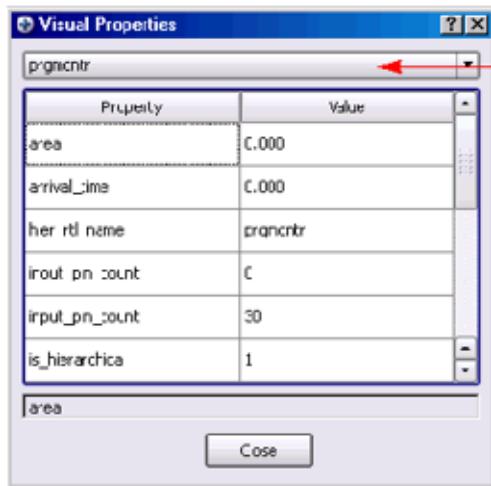
The set net color command sets the color of the selected net in the HDL Analyst for the current session. To use the command, select the desired net or nets in the RTL view and select set net color from the popup menu to display the dialog box.



Double click the corresponding color in the Color column to display the color pallet and then double click the desired color and click OK. Nets can be grouped and assigned to the same color by selecting the same group number in the Group Number column.

## Properties Popup Menu Command

The software displays property information about the selected object when you right-click on a net, instance, pin, or port in an HDL Analyst view. See [Visual Properties Panel, on page 596](#) or [Viewing Object Properties, on page 412](#) in the *User Guide* for more information about viewing object properties.

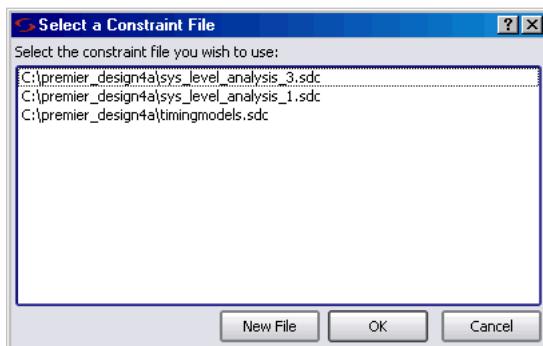


Lists pins, if the selected object is an instance or net.  
Lists bits, if the selected object is a port.

## Edit Attributes Popup Menu Command

*Synplify Pro, Synplify Premier*

You use the Select a Constraint File dialog box to choose or create a constraint file. You can open the constraint file and edit it. For technologies that support the compile points, it lets you create or edit a compile-point constraint file for the selected region or instance.



For more information about creating constraint files, see [Specifying Xilinx Constraints \(Legacy\), on page 270](#) of the *User Guide*.

## Simulation Popup Menu Commands in the RTL/Technology View

When you use the VCD-Analyst Integration tool, the following simulation popup commands are available if you right-click in the views.

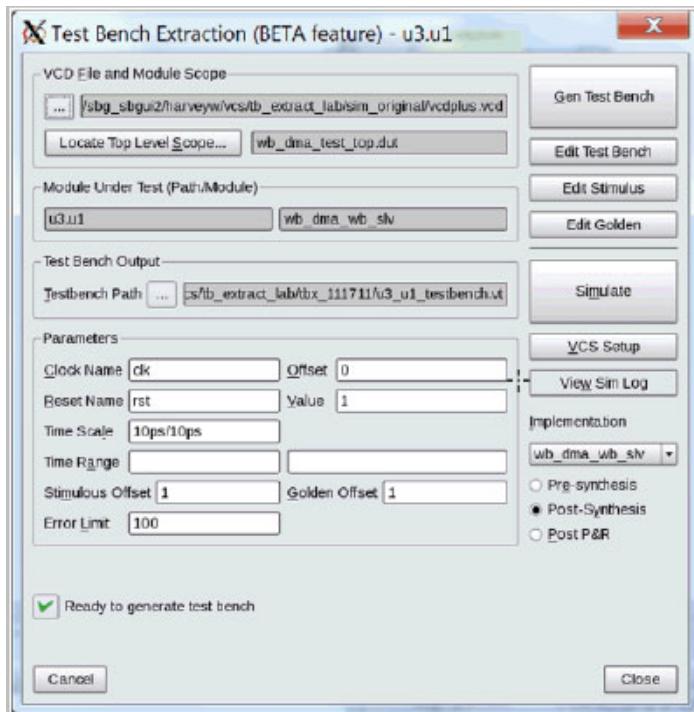
For more information see [HDL Analyst Menu: VCD Commands, on page 554](#) and [Using VCD/Identify with HDL Analyst, on page 1179](#) in the *User Guide*.

| Command                                                            | Description                                                                                                                                             |
|--------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Popup Menu Commands in the Simulation Panel View</b>            |                                                                                                                                                         |
| Launch DVE                                                         | Launches DVE.                                                                                                                                           |
| Configure DVE                                                      | Allows you to configure DVE.                                                                                                                            |
| Export Signals to DVE                                              | Exports watched signals from VCS-Analyst tool to DVE.                                                                                                   |
| Import Signals from DVE                                            | Imports watched signals from DVE to the VCS-Analyst tool.                                                                                               |
| Simulation Properties                                              | Displays properties for the waveform viewer global data and parameters.                                                                                 |
| VCD->Test Bench Extraction                                         | Generates the test bench files and launches the VCS simulator for the xDUT module. See the <a href="#">Test Bench Extraction Command, on page 649</a> . |
| <b>Popup Menu Commands Within the Name View Fields</b>             |                                                                                                                                                         |
| Show Full Name                                                     | Displays full name for the signal. Otherwise, use Show Leaf Name.                                                                                       |
| <b>Popup Menu Commands Within the Signal Value Fields</b>          |                                                                                                                                                         |
| Show Current Value Only                                            | Displays the current value for the watched signal. Otherwise, select Show Previous, Current and Next Values.                                            |
| <b>Popup Menu Commands Within the Name and Signal Value Fields</b> |                                                                                                                                                         |
| Collapse / Collapse all                                            | Collapses selected signals or all signals.                                                                                                              |
| Re-Expand                                                          | Expand signals that have been collapsed.                                                                                                                |
| Remove all                                                         | Removes all signals from the Simulation Panel watch list.                                                                                               |

| Command    | Description                                                                                                                                                                                                   |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Format     | Allows you to change the signal format to: <ul style="list-style-type: none"><li>• Enum</li><li>• Binary</li><li>• Octal</li><li>• Unsigned decimal</li><li>• Decimal</li><li>• Hex</li><li>• ASCII</li></ul> |
| Color      | Allows you to change the default color for the signal name or values.                                                                                                                                         |
| Remove     | Removes the selected signal from the Simulation Panel watch list.                                                                                                                                             |
| Properties | Displays the register properties associated with the selected signal.                                                                                                                                         |

## Test Bench Extraction Command

Use the VCD->Test Bench Extraction command to generate test bench files and launch the VCS simulator for the Design Under Test module (xDUT). For information about the Test Bench Extraction flow, see [Extracting VCS Test Benches for Submodules, on page 1201](#).



The following table describes the options:

| Option                          | Description                                                                                                                                                                                                                                                                                        |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VCD File and Module Scope       | Specifies the path to the simulation VCD file of the RTL DUT and the scope path within the file to the top-level module of the DUT.                                                                                                                                                                |
| Module Under Test (Path/Module) | Shows the xDUT instance path and module name that are read-only fields. To choose another instance path, close the dialog box and select another xDUT.                                                                                                                                             |
| Test Bench Output               | Displays a file browser to select the main xDUT test bench file. The directory for this file contains the stimulus and golden test bench files generated by vcat, as well as, log files and other working files.<br><br>Note: The names of the test bench files are generated by vcat.             |
| <b>Parameters</b>               |                                                                                                                                                                                                                                                                                                    |
| • Clock Name                    | Specifies the name of the clock signal for the xDUT. The default is clk.                                                                                                                                                                                                                           |
| • Offset                        | Compares the xDUT outputs with the Golden outputs at the positive transition of the clock. To delay the comparison (for example, to account for propagation delay through the xDUT), you can specify a positive offset in VCD time units. This is typically a small percentage of the clock cycle. |
| • Reset Name                    | Specifies the name of the reset signal for the xDUT. The default is rst                                                                                                                                                                                                                            |
| • Value                         | Specifies the value of the reset signal that enables comparison of the xDUT and golden values. The default is 1.                                                                                                                                                                                   |
| • Time Scale                    | Displays the VCD time scale in read-only mode.                                                                                                                                                                                                                                                     |
| • Time Range                    | Enter a time range for the simulation in VCD time units. If left blank, uses the entire duration of the original simulation.                                                                                                                                                                       |

| Option            | Description                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| • Stimulus Offset | Enter the time in VCD units for delaying application of the stimulus input signals.<br>You might need to correct race conditions by adjusting signal arrival times. Beginning at time 0, vcat applies the signals with blocking assignments in both the stimulus and module generation test bench files. Specify an offset, to offset this time. You can also manually edit the files for more complex modifications. |
| • Golden Offset   | Enter the time in VCD units for delaying the application of golden output signals.<br>You might need to correct race conditions by adjusting signal arrival times. Beginning at time 0, vcat applies the signals with blocking assignments in both the golden and module generation test bench files. Specify an offset, to offset this time. You can also manually edit the files for more complex modifications.    |
| • Error Limit     | Enter a positive number to limit the number of times to report mismatches.                                                                                                                                                                                                                                                                                                                                            |

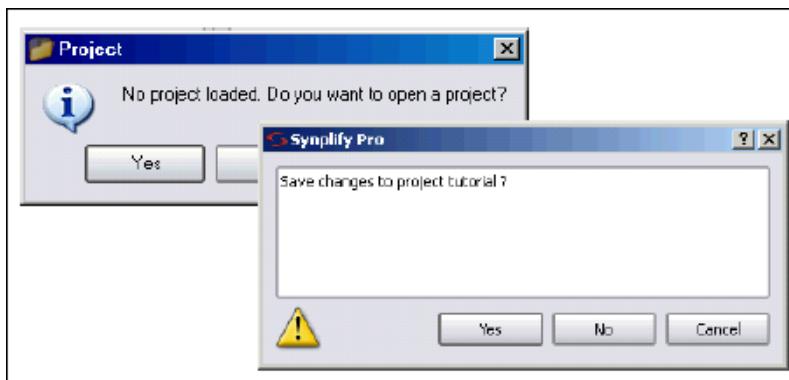
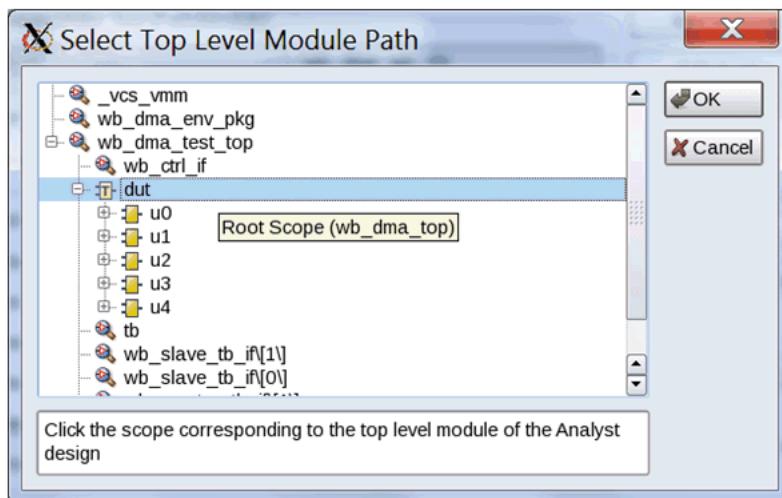
**Test Bench Extraction actions that can be performed ...**

|                 |                                                                                                                                                                                                               |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Gen Test Bench  | Generates the test bench files. The following files are generated/updated: <ul style="list-style-type: none"><li>• Main xDUT test bench file</li><li>• Stimulus file for xDUT</li><li>• Golden file</li></ul> |
| Edit Test Bench | Allows you to edit the main xDUT test bench file. You are prompted to over-write the existing file.                                                                                                           |
| Edit Stimulus   | Allows you to edit the stimulus file for the xDUT. You are prompted to over-write the existing file.                                                                                                          |
| Edit Golden     | Allows you to edit the golden file. You are prompted to over-write the existing file.                                                                                                                         |

| Option           | Description                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Simulate         | <p>Launches VCS simulation for the xDUT. Uses the Synplify Premier/VCS Integration utility to:</p> <ul style="list-style-type: none"> <li>• Add the xDUT test bench file, stimulus file, and golden file to any test bench files you may have configured manually.</li> <li>• Set the top-level module to the main test bench module.</li> <li>• Specify the -sverilog switch.</li> <li>• Run the simulation script.</li> </ul> |
| VCS Setup        | Displays the VCS setup dialog box. Allows you to configure VCS setup.                                                                                                                                                                                                                                                                                                                                                           |
| View Sim Log     | Lets you view the simulation log file. This view is continuously updated as simulation progresses.                                                                                                                                                                                                                                                                                                                              |
| Implementation   | Selects the xDUT implementation to use. You must have a xDUT implementation in order to perform simulation.                                                                                                                                                                                                                                                                                                                     |
| • Pre-synthesis  | Pre-synthesis simulation is performed.                                                                                                                                                                                                                                                                                                                                                                                          |
| • Post-synthesis | Post-synthesis simulation is performed. This is the default.                                                                                                                                                                                                                                                                                                                                                                    |
| • Post P&R       | Post P&R synthesis is performed.                                                                                                                                                                                                                                                                                                                                                                                                |

## Select Top Level Module Path

click the Locate Top Level Scope button to display a dialog box shown below that designates the top-level scope module. As you click on a module, the scopes are dynamically matched to the netlist.





# CHAPTER 7

# Design Planner Commands

This chapter describes Design Planner commands. The following sections contain the details:

- [Design Planner Menu \(Generation 2\)](#), on page 656
  - [Design Planner UI Commands](#), on page 658
  - [Design Planner Tools Menu](#), on page 667
  - [Design Planner Popup Menus](#), on page 673
  - [Design Planner View Commands](#), on page 681
  - [Physical Analyst UI Commands](#), on page 686
  - [Physical Analyst View Popup Menus](#), on page 692

# Design Planner Menu (Generation 2)

## *Synplify Premier*

When the Design Planner (Generation 2) is open, the Planner menu lets you:

- Create partitions and assign logic to these regions on the device.
- Use filter options to hide or zoom in on partitions created and selected on the device.
- Set preferences for various design planner features.

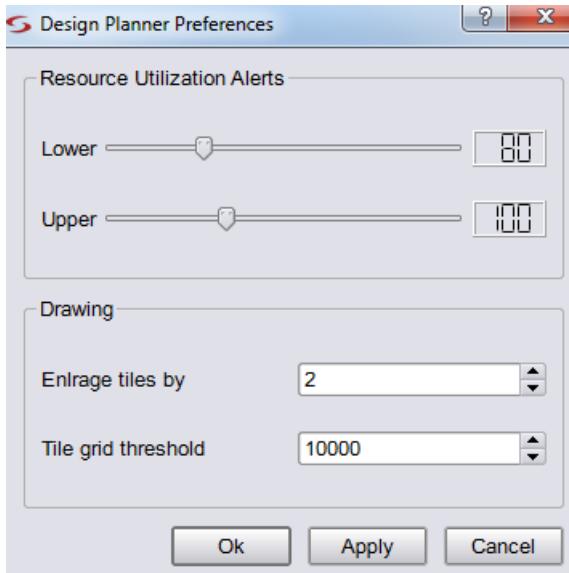
The Planner menu commands and any associated dialog boxes are described as follows:

- Partition Tool – Use this option to create partitions on the device and assign logic to these regions.
- Filter Actions – For example, you can use Filter Actions such as Zoom Selected to zoom in on a highlighted partition region.
- Planner Preferences – See [Design Planner Preferences, on page 656](#).

See [Using Design Planner \(Generation 2\), on page 978](#) for more information.

## Design Planner Preferences

Use the Design Planner Preferences command to control the Design Planner display for better visibility. For details, see [Viewing Resources on the Device, on page 987](#).



The following dialog box options are described below.

| Command                                                                                                  | Description                                                                                                                                                                                                                       |
|----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Resource Utilization <ul style="list-style-type: none"><li>• Lower</li><li>• Upper</li></ul>             | Allows you to change the lower and upper percentage limits for receiving resource utilization alerts.                                                                                                                             |
| Drawing <ul style="list-style-type: none"><li>• Enlarge tiles by</li><li>• Tile grid threshold</li></ul> | You can: <ul style="list-style-type: none"><li>• Increase the drawing size of tiles so they can be displayed larger.</li><li>• Have the tile grid drawn if the tile grid spacing exceeds the Tile grid threshold value.</li></ul> |

# Design Planner UI Commands

## *Synplify Premier*

The commands for Design Planner functionality appear on various menus in the Design Planner graphical user interface (GUI).

- [File Menu: New Command](#), on page 658
- [Edit Menu: Design Planner Commands](#), on page 661
- [View Menu: Design Planner Commands](#), on page 662
- [Implementation Options: Design Planner](#), on page 665
- [Run Menu: Design Planner Commands](#), on page 666

## File Menu: New Command

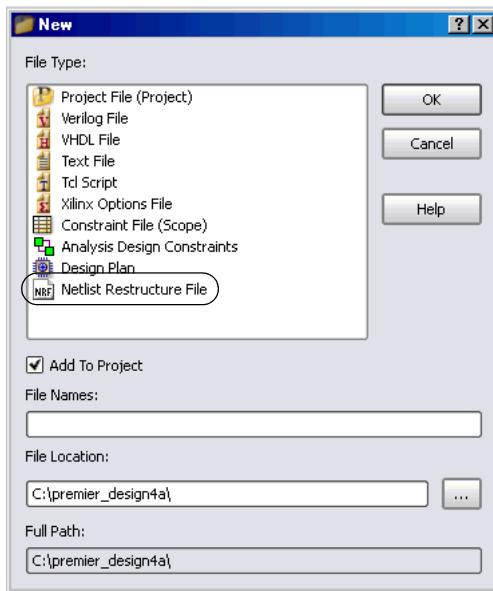
### *Synplify Premier*

The File->New command includes functionality for Design Planner-specific files:

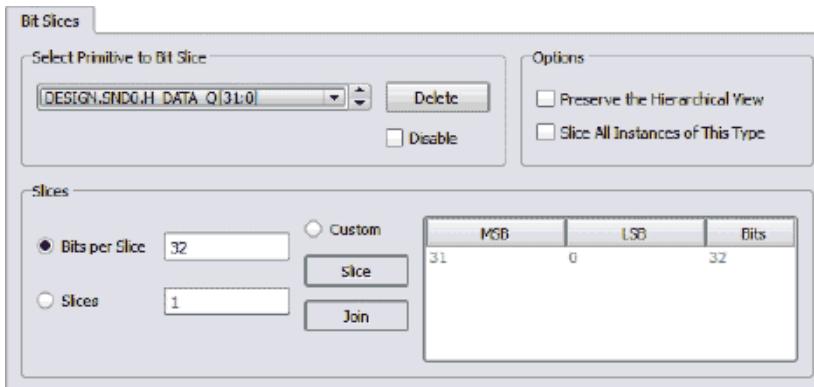
| Command | Description                                                                                                                                                                                                                      |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| New...  | Lets you create netlist restructure and Xilinx options files. <ul style="list-style-type: none"><li>• <a href="#">Netlist Restructure File</a>, on page 658</li><li>• <a href="#">Xilinx Options File</a>, on page 660</li></ul> |

## Netlist Restructure File

File->New displays the New dialog box, where you can select the file type of Netlist Restructure File. You can choose to add this file to your project, specifying a file name and location. After you click OK, the netlist restructure file view opens along with the HDL Analyst RTL view.

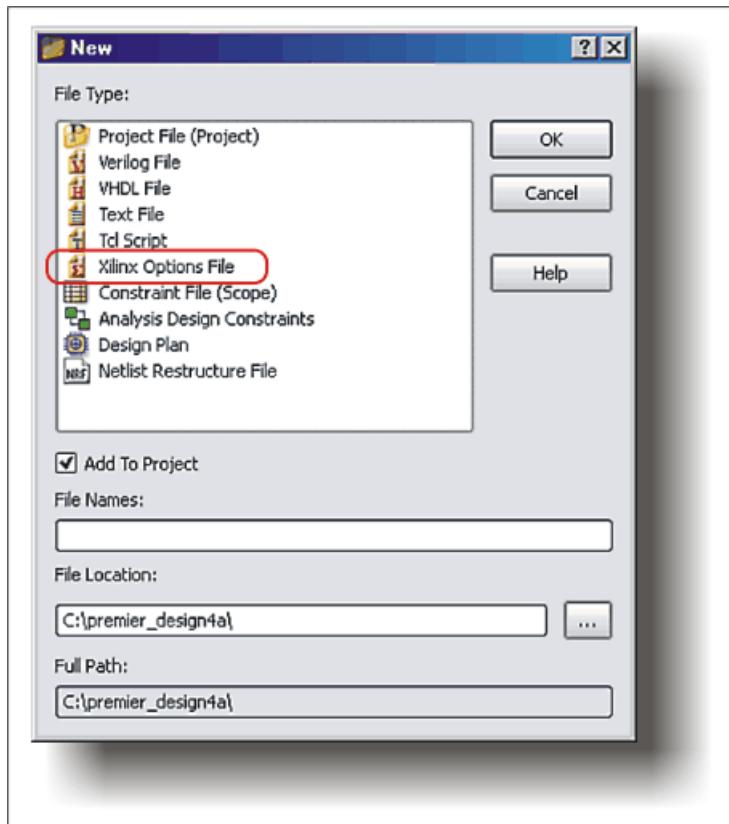


The figure below shows the view for creating the netlist restructure file used to perform bit-slicing. For more information, see [Bit Slicing, on page 1035](#) in the *User Guide*.

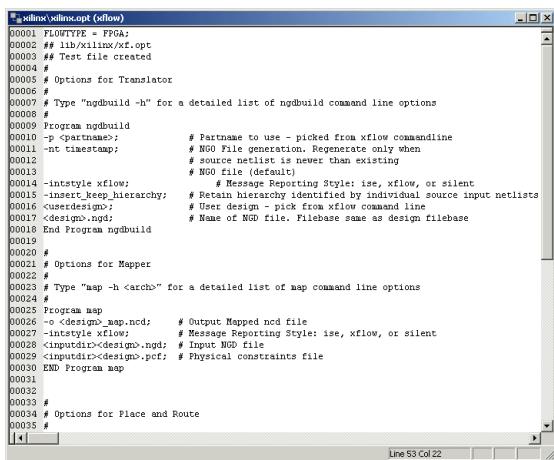


## Xilinx Options File

File->New displays the New dialog box, where you can select the file type of Xilinx Options File. You can choose to add this file to your project, specifying a file name and location. After you click OK, the default Xilinx place-and-route options file is displayed. To run Xilinx place-and-route, see [Running P&R Automatically after Synthesis, on page 1102](#) in the *User Guide*.



The following figure shows the partial contents of one of the default Xilinx place-and-route options files (`lib/xilinx/xilinx_name.opt`). Edit the contents of the file as required, and then save the file. To run Xilinx place-and-route, see [Running P&R Automatically after Synthesis, on page 1102](#) in the *User Guide*.



```

#xilinx\>xilinxopt (xflow)
00001 FLOWTYPE = FPGA;
00002 ## lib/xilinx/xf.opt
00003 ## Test file created
00004 #
00005 # Options for Translator
00006 #
00007 # Type "ngdbuild -h" for a detailed list of ngdbuild command line options
00008 #
00009 Program ngdbuild
00010 -P <Partname> # Partname to use - picked from xflow commandline
00011 -nt timestamp # NGD File generation. Regenerate only when
00012 # source netlist is newer than existing
00013 # NGD file (default)
00014 -lifestyle xflow; # Message Reporting Style: ise, xflow, or silent
00015 -insert_hierachy; # Retain hierarchy identified by individual source input netlists
00016 <userdesign>.ndg; # User design - pick from xflow command line
00017 <design>.ngd; # Name of NGD file. Please same as design filebase
00018 End Program ngdbuild
00019
00020 #
00021 # Options for Mapper
00022 #
00023 # Type "map -h <arch>" for a detailed list of map command line options
00024 #
00025 Program map
00026 -o <design>.map.ngd; # Output Mapped ngd file
00027 -lifestyle xflow; # Message Reporting Style: ise, xflow, or silent
00028 <inputdir><design>.ngd; # Input NGD file(s)
00029 <inputdir><design>.pcf; # Physical constraints file
00030 END Program map
00031
00032 #
00033 # Options for Place and Route
00034 #
00035 #

```

Line 53 Col 22

## Edit Menu: Design Planner Commands

*Synplify Premier*

The following table lists the Design Planner commands on the Edit menu:

### Edit Menu Commands — Design Planner

|                             |                                                                                                                                          |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| Paste - Replicate           | Adds assignments from one region to another region.                                                                                      |
| Delete Region Assignments   | Deletes all assignments from all regions.                                                                                                |
| Delete Selected Assignments | Deletes the instance(s) selected in the Design Plan Hierarchical, Design Plan, or Design Plan Editor view from the corresponding region. |
| Edit Regions                | Enables regions in the Design Plan Editor for sizing, moving, and aligning (Xilinx only).                                                |
| Delete Region               | Deletes the selected region(s) (Xilinx only).                                                                                            |
| Align Regions               | Aligns the selected regions (right, left, up, down) in the Design Plan Editor (Xilinx only).                                             |

## View Menu: Design Planner Commands

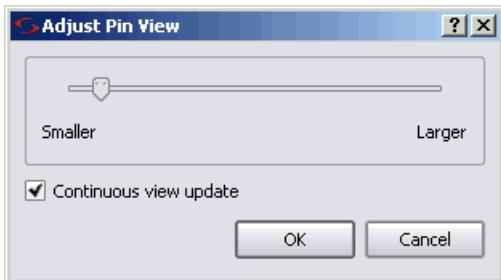
*Synplify Premier*

These View menu commands are available when the Design Planner window is active. These commands are in addition to the commands described in [View Menu Commands: All Views, on page 415](#) and [View Menu: Zoom Commands, on page 416](#).

| Command                  | Description                                                                                                                                                                                             |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show All Regions         | Displays all regions.                                                                                                                                                                                   |
| Expanded Pin View        | Enables or disables the expanded I/O pins on the device displayed in the Design Plan Editor.                                                                                                            |
| Adjust Pin View...       | Zooms in or out of the expanded pin view so that you can adjust the size of the I/O pins on the device displayed in the Design Plan Editor. See <a href="#">Adjust Pin View Command , on page 662</a> . |
| Rats Nest->Show          | Shows all, hides all, or shows selected rats nesting. Rats nesting displays connectivity between I/O pads and logic for all regions on the device in the Design Plan Editor.                            |
| Rats Nest->Hide          |                                                                                                                                                                                                         |
| Rats Nest->Show Selected |                                                                                                                                                                                                         |
| Properties               | Displays the Device or Region Properties or the Properties dialog box, showing the properties of the selected object. See <a href="#">Design Planner Properties Command , on page 663</a> .             |

### Adjust Pin View Command

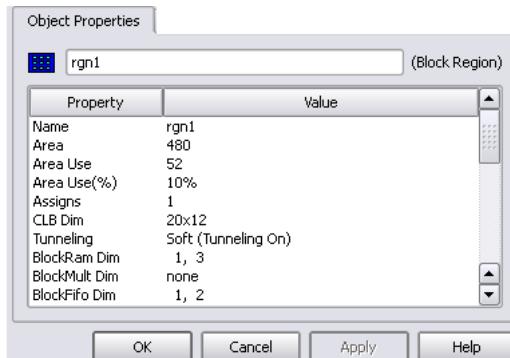
Use this command to adjust the size of the pins in the Design Plan Editor for viewing and pin assignments.



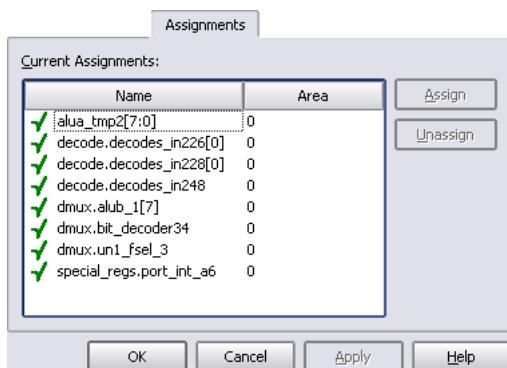
| Command                | Description                                                                                                                                                                                  |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Slider                 | Move the slider to increase or decrease the size of the pins.                                                                                                                                |
| Continuous view update | When enabled, dynamically displays the pin size changes in the Design Plan Editor as you drag the slider. If disabled, the pin size display updates only after you release the mouse button. |

## Design Planner Properties Command

Select View->Properties in the Synplify Premier Design Planner to display the Region Properties or Properties dialog box, depending on whether the selected object is a region or not. These two dialog boxes are similar; they both display the properties of the selected object in the Object Properties pane. The Region Properties dialog box also lets you change the region name in this pane.



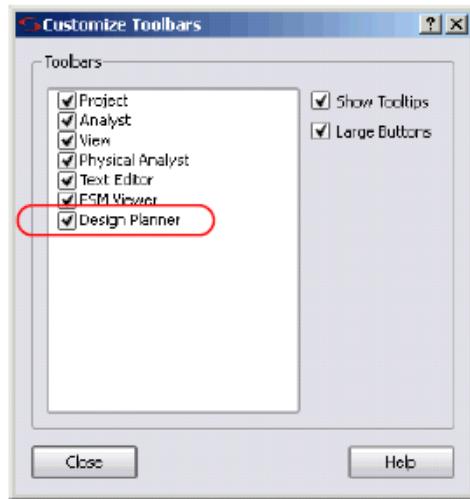
The Region Properties dialog box has an additional pane, Assignments, where you can unassign selected logic instances from the region. Before exiting the dialog box, you can reassign any instances that you mistakenly unassigned.



## Design Planner Toolbar Command

Select View->Toolbars to display the Toolbars dialog box, where you can:

- choose the toolbars to display
- customize their appearance



For details, see [Toolbar Command](#), on page 418.

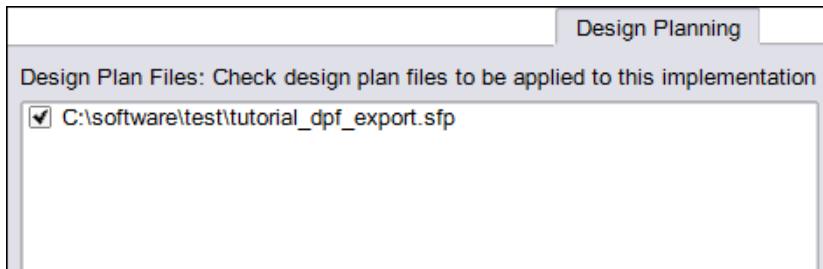
## Implementation Options: Design Planner

You can use the Project menu to set implementation options or create new implementations. For the Project->Implementation Options... and Project->New Implementation... commands, the following features apply specifically to the Synplify Premier Design Planner:

- [Design Planning Panel, on page 665](#)
- [GCC & Prototyping Tools Panel, on page 665](#)

### Design Planning Panel

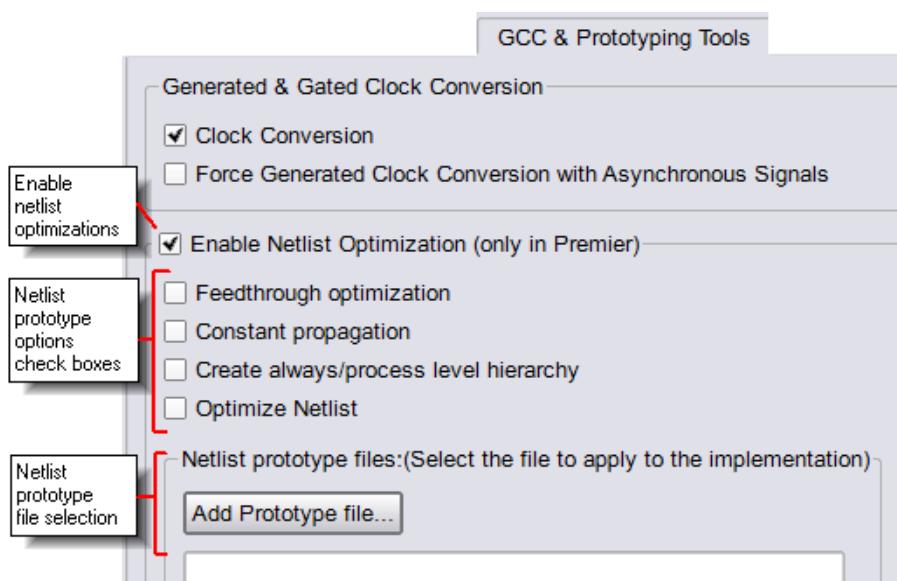
In the Synplify Premier with Design Planner tool, you use the Design Planning panel to select the design plan file to use when you synthesize your design for the implementation. See [Chapter 18, Floorplanning with Design Planner](#) of the *User Guide* for details.



### GCC & Prototyping Tools Panel

*Synplify Premier*

The GCC & Prototyping Tools panel also allows you to specify one or more netlist prototyping file options (the Enable prototyping tools check box must be enabled) and any netlist restructure or Tcl files to compile with the design. Use the Add Prototype file button to add netlist restructure files and Tcl files to the project. See [Using Design Planner, on page 998](#) of the *User Guide* for details.



## Run Menu: Design Planner Commands

You use the Run menu to perform tasks such as the following:

- Create a physical hierarchy.
- Estimate area required by each design module.

| Command                    | Description                                                                                                                |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Estimate Area              | Estimates the area required by each design module. Creates an estimation (est) file and an estimation log (_est.srr) file. |
| Compile Physical Hierarchy | Creates a netlist that includes regions as modules (Design Planner option).                                                |

# Design Planner Tools Menu

When the Synplify Premier Design Planner is open, the Tools menu lets you:

- Display and assign/unassign replicated modules.
- Run area estimation for selected regions or all regions on the device.
- Set preferences for various tool features.

The Tools menu commands and their associated dialog boxes are described in the following sections:

- [Tools Menu Commands](#), on page 667
- [Show Replicated Assignments Command](#), on page 668
- [Preferences Command](#), on page 668

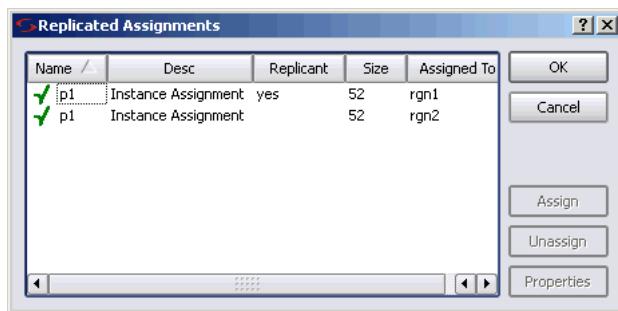
## Tools Menu Commands

The following table lists the Tools menu commands.

| Command                     | Description                                                                                                                                                                                                             |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show Replicated Assignments | Displays all replicated modules assigned to various regions on the device. Assign or unassign replicated logic as necessary. See <a href="#">Show Replicated Assignments Command</a> , on page 668 for further details. |
| Estimate Regions            | Runs area estimation for selected regions on the device.                                                                                                                                                                |
| Estimate All Regions        | Runs area estimation on all regions on the device.                                                                                                                                                                      |
| Design Planner Preferences  | Allows you to select preferences for the Design Planner general operations and region assignments. See <a href="#">Preferences Command</a> , on page 668 for further details.                                           |

## Show Replicated Assignments Command

Select Tools->Show Replicated Assignments to show all replicated modules assigned to various regions on the device. This displays the Replicated Assignments dialog box, where you can assign or unassign replicated logic, as necessary.



## Preferences Command

Select Tools->Design Planner Preferences to define how the Synplify Premier Design Planner displays windows and rats nesting, and handles warnings and pin assignments in this view. The Preferences dialog box is displayed, with tabs to the following panels:

- [General Panel](#)
- [Assignments Panel](#)
- [Pins Panel](#)

### General Panel

Sets preferences for displaying the Synplify Premier Design Planner windows and rats nesting.



| Command                        | Description                                                                                                                                                      |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Maximum Undo                   | For the series of operations you perform in the Design Planner view, specify the maximum number of operations that you can undo while in a session of this view. |
| Tile with Analyst when opening | If enabled, the HDL Analyst RTL view will also be tiled with the Design Planner view when it is opened.                                                          |
| Include Pins                   | When you turn on rats nesting, you can choose to display connections to I/O pins or not.                                                                         |
| Show Number of Connections     | When you turn on rats nesting, you can choose to display the number of connections between regions or I/O pins after making assignment to these resources.       |

## Assignments Panel

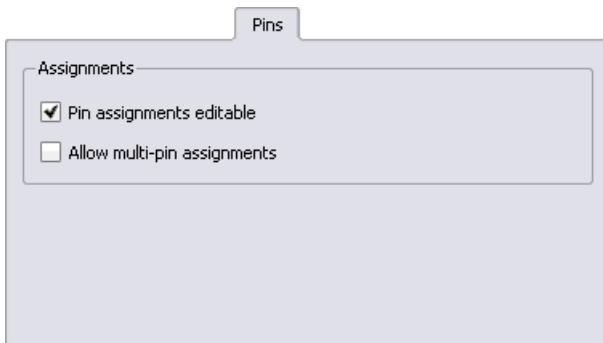
Sets preferences for displaying assignment warnings, area estimate warnings, and rats nests when dragging and dropping assignments.



| Command                       | Description                                                                                                                                                                    |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Show assignment warnings      | When enabled, this option might display messages when you assign logic to regions. For example, see <a href="#">Region Assignment Messages Dialog Box , on page 671</a> .      |
| No area estimate warning      | When enabled, this option displays a dialog box (see <a href="#">Estimation Needed and Running Estimation Dialog Boxes , on page 672</a> ) when you first open Design Planner. |
| Show rats' nest when dragging | When enabled, shows the attached nets to regions as you drag an instance or pin from the RTL view to the Design Planner view.                                                  |

## Pins Panel

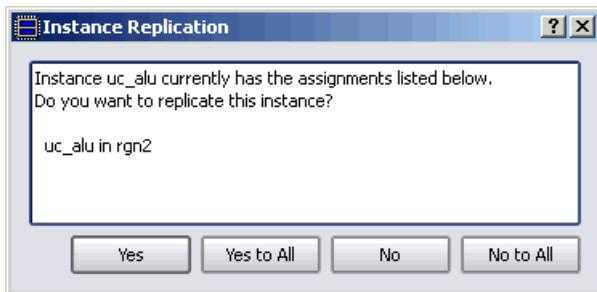
Specifies if pin assignments can be edited and if multiple-pin assignments are allowed.



| Command                        | Description                                                                                                                        |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Pin assignments<br>editable    | When enabled, this option will not allow you to assign ports to pin locations. By default, this option is disabled.                |
| Allow multi-pin<br>assignments | When enabled, this option allows you to assign ports to multiple pin locations. Otherwise, you will be prohibited from doing this. |

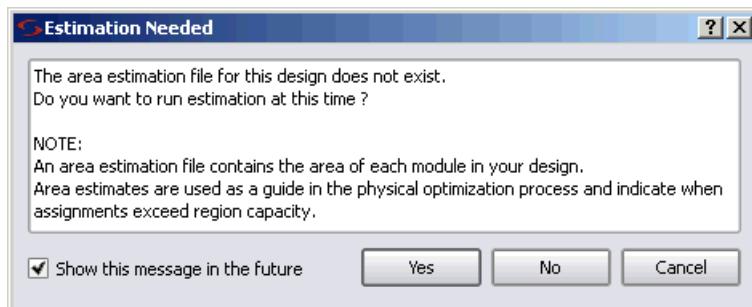
## Region Assignment Messages Dialog Box

The Instance Replication dialog box is displayed when you try to assign the same logic to different regions, if you have enabled Show assignment warnings on the Assignments tab of Tools->Design Planner Preferences.

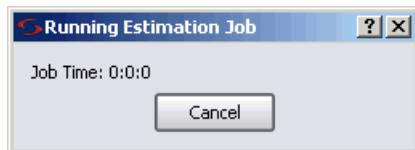


## Estimation Needed and Running Estimation Dialog Boxes

The Estimation Needed dialog box opens when you first open the Design Planner, if you have enabled No area estimate warning on the Assignments tab of Tools->Design Planner Preferences.



The Running Estimation dialog box shows the time taken to get an area estimate. It opens if you click Yes on the Estimation Needed dialog box.



# Design Planner Popup Menus

The commands available in the Synplify Premier Design Planner popup menu vary, depending on the technology, the current status of the design, what is selected, and the panel in which you are working. This section describes the following popup menus and dialog boxes:

- [Design Planner Hierarchical Browser Popup Menu](#), on page 673
- [Design Planner Popup Menu](#), on page 673
- [Design Plan Editor Popup Menu](#), on page 674
- [Properties Popup Menu Command \(Design Planner\)](#), on page 675
- [Show/Hide Columns Popup Menu Command](#), on page 676
- [Assign Dialog Box](#), on page 679
- [Instance Replication Dialog Box](#), on page 680

The following tables describe the Synplify Premier Design Planner popup menu commands. Popup menu commands from the Design Planner are added to the initial commands listed in these tables as you create physical regions and make assignments to regions and I/O pins. This lets you access the same commands from different locations within the synthesis tool.

## Design Planner Hierarchical Browser Popup Menu

This table shows the popup menu commands available in the Hierarchical Browser of the Design Planner.

| Command    | Description                                                                                                                                                          |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Properties | Displays the Properties dialog box, listing statistics (device, region, unassigned bins, or temporary assigns) for the object selected, depending on its technology. |

## Design Planner Popup Menu

This table shows the popup menu commands available in the Design Plan view of the Design Planner.

| Command                    | Description                                                                                                                                                                                                                                      |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Auto-size columns          | Automatically adjusts the column widths to fit in the Physical Report view.                                                                                                                                                                      |
| Hide column <i>colname</i> | Hides the column called <i>colname</i> .                                                                                                                                                                                                         |
| Show Column...             | Lets you add columns by choosing from those in this submenu.                                                                                                                                                                                     |
| Show/Hide Columns...       | Displays the Select Columns dialog box, where you can choose to enable or disable device options for reporting statistics in the column display for the current device. See <a href="#">Show/Hide Columns Popup Menu Command , on page 676</a> . |

## Design Plan Editor Popup Menu

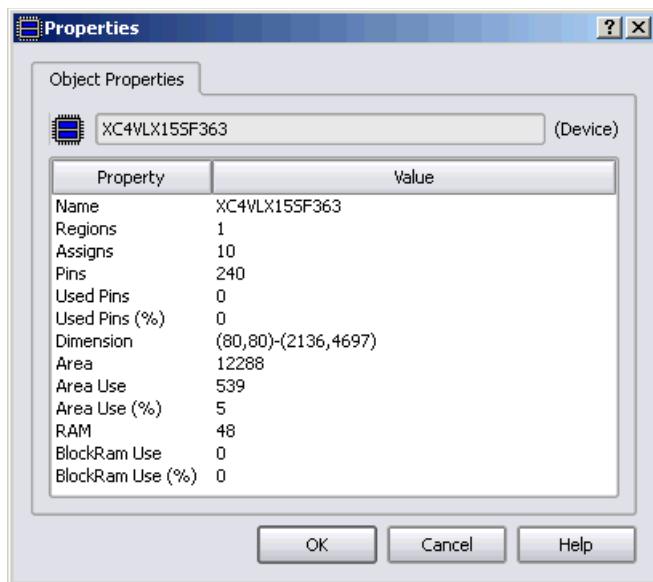
This table shows the popup menu commands available in the Design Plan Editor view of the Design Planner.

| Command                              | Description                                                                                                                                                                                                                                                                                                            |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RTL View                             | Displays the RTL view for the selected or unselected physical regions, depending on which you choose.                                                                                                                                                                                                                  |
| Delete Region Assignments            | Deletes assignments for the selected physical region.                                                                                                                                                                                                                                                                  |
| Delete Region                        | Deletes the selected physical regions.                                                                                                                                                                                                                                                                                 |
| Estimate Regions                     | Estimates utilization of mapped resources, such as LUTs and registers, for the selected physical regions.                                                                                                                                                                                                              |
| Region Type                          | <ul style="list-style-type: none"> <li>• Soft (Tunneling On) - in or out assignments allowed.</li> <li>• Hard (Tunneling Off) - only in assignments, no out.</li> <li>• Keep-Out - ensures that no placement occurs in the region. Use this option to create decongestion areas for optimizing your design.</li> </ul> |
| LogicLock<br>->Enabled<br>->Disabled | Enables or disables assignment of the selected regions as LogicLock™ Regions.                                                                                                                                                                                                                                          |
| Show/Hide Region                     | Shows or hides the selected physical regions.                                                                                                                                                                                                                                                                          |
| Paste                                | Pastes the assignments of the most recently copied physical region to the selected physical region.                                                                                                                                                                                                                    |

| Command              | Description                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Paste Replicate      | Pastes the assignments of the most recently copied replicated logic to the selected physical region.                                                                    |
| Edit Regions         | Locks or unlocks the selected physical regions for changing its size or location.                                                                                       |
| Add                  | Creates a new physical region. The available commands in this submenu depend on the current device. See <a href="#">Chapter 18, Floorplanning with Design Planner</a> . |
| Show All Regions     | Displays all physical regions, including those that were hidden.                                                                                                        |
| Estimate All Regions | Estimates utilization of mapped resources, such as LUTs and registers, for all physical regions.                                                                        |
| Rats Nest            | Displays the interconnects between the physical resources on the device.                                                                                                |
| Properties           | Displays the Properties dialog box, listing statistics (device, region, unassigned bins, or temporary assigns) for the object selected, depending on its technology.    |

## Properties Popup Menu Command (Design Planner)

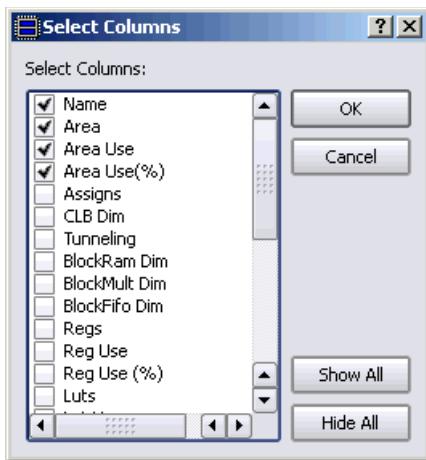
The Properties dialog box displays properties of the currently selected object. The properties shown depend on the technology and the current status of your design.



## Show/Hide Columns Popup Menu Command

In the Design Plan view, select the Show/Hide columns popup menu command to display the Select Columns dialog box. Customize your view by selecting or deselecting the available column headings.

The table below lists most of the resources, only some of which are available at any time, depending on the device selected.



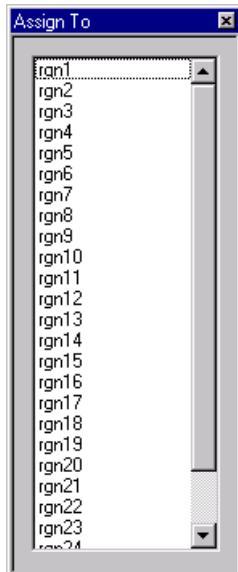
| Column Selection                             | Description                                                                                                                                       |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Area, Area Use, Area Use (%)                 | Total number of mapped resources, number of resources used, and percentage of resource utilization after you run Estimate Regions for the region. |
| Assigns                                      | Number of assignments in the region.                                                                                                              |
| BlockFifos, BlockFifo Use, BlockFifo Use (%) | Number of Block Fifos, number of Block Fifos used, and percentage of Block Fifos used after you run Estimate Regions in the region. Xilinx only.  |
| BlockMults, BlockMult Use, BlockMult Use (%) | Number of Block Mults, number of Block Mults used, and percentage of Block Mults used after you run Estimate Regions in the region. Xilinx only.  |
| BlockRams, BlockRams Use, BlockRams Use (%)  | Number of Block RAMs, number of Block RAMs used, and percentage of Block RAMs used after you run Estimate Regions in the region. Xilinx only.     |
| BlockFifo Dim                                | The dimension of the Block Fifo(s) for the region. Xilinx only.                                                                                   |
| BMULT Dim                                    | The dimension of the Block Mult(s) for the region. Xilinx only.                                                                                   |
| BRAM Dim                                     | The dimension of the Block RAM(s) for the region. Xilinx only.                                                                                    |

| Column Selection                                           | Description                                                                                                                                                                               |
|------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CLB Dim                                                    | The dimension of the CLB(s) for the region. Xilinx only.                                                                                                                                  |
| Dimension                                                  | The dimension of the ROW regions.                                                                                                                                                         |
| DSP, DSP Use, DSP Use (%)                                  | Number of DSPs, number of DSPs used, and percentage of DSPs used after you run Estimate Regions in the region. Intel Stratix only.                                                        |
| ESB Dim                                                    | The dimension of the ESB(s) in the region. Intel only.                                                                                                                                    |
| ESB, ESB Use, ESB Use (%)                                  | Number of ESBs, number of ESBs used, and percentage of ESBs used after you run Estimate Regions in the region. Intel only.                                                                |
| LAB Dim                                                    | The dimension of the LAB(s) for the region. Intel only.                                                                                                                                   |
| Luts, Lut Use, Lut Use (%)                                 | Number of LUTs, number of LUTs used, and percentage of LUTs used in the region after you run Estimate Regions.                                                                            |
| M4K, M4K Use, M4K Use (%)                                  | Number of M4Ks, number of M4Ks used, and percentage of M4Ks used after you run Estimate Regions in the region. Intel Stratix only.                                                        |
| M512, M512 Use, M512 Use (%)                               | Number of M512s, number of M512s used, and percentage of M512s used after you run Estimate Regions in the region. Intel Stratix only.                                                     |
| MaxChainLength, MaxChainLength Use, MaxChainLength Use (%) | Length of longest carry chain, length of the longest carry chain assigned to the region, and percentage of the longest carry chain assigned to the region after you run Estimate Regions. |
| MegaRAM, MegaRAM Use, MegaRAM Use (%)                      | Number of MegaRAMs, number of MegaRAMs used, and percentage of MegaRAMs used after you run Estimate Regions in the region. Intel Stratix only.                                            |

| Column Selection           | Description                                                                                                                   |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Name                       | Name of the region.                                                                                                           |
| Regs, Reg Use, Reg Use (%) | Number of registers, number of registers used, and percentage of registers used after you run Estimate Regions in the region. |
| Tunneling                  | Tunneling if enabled/disabled.                                                                                                |

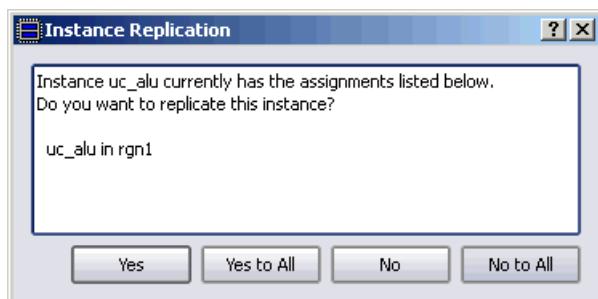
## Assign Dialog Box

In the Synplify Premier tool, you can assign modules or primitives from an HDL Analyst, Design Plan Hierarchy Browser, or Design Plan view to regions you create in the Design Plan Editor by selecting Assign->*regionName* from the popup menu. The regions listed in the popup menu are in order of most recent use. If there are more than 25 regions, choosing More in the menu displays the Assign To dialog box, where you can choose from additional regions. The additional regions are listed alphabetically in the dialog box.



## Instance Replication Dialog Box

When you assign the same instance logic from the HDL Analyst RTL view to different regions created in the Design Plan Editor, an Instance Replication dialog box appears as shown in the following figure. Confirm whether or not you want to replicate the selected logic instance to the specified region. You can choose to confirm replication for each instance separately (Yes or No) or simultaneously (Yes to All or No to All) for multiple instances.



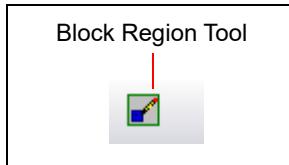
# Design Planner View Commands

## Synplify Premier

The Design Planner graphical user interface includes a toolbar icon and keyboard shortcuts in addition to the menu commands.

### Design Planner Toolbar Icon

The Design Planner toolbar allows you to create block regions on the device floorplan. You can assign critical path logic to these regions, then run synthesis and the place-and-route tool to help improve performance for the design.



The following table describes the default Design Planner icons.

| Icon              | Description                                                                                                                            |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| Block Region Tool | Create block regions on the device floorplan, to which you can assign critical path logic to help improve performance for your design. |

### Design Planner Keyboard Shortcuts

Keyboard shortcuts are key sequences that you type in order to run a command. Menus list keyboard shortcuts next to the corresponding commands. For example, you can press and hold the Ctrl key while you type the e key, instead of using the menu command View ->Expand Pin View.

The following table describes the Design Planner keyboard shortcuts.

| Keyboard Shortcut | Description                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ctrl-e            | In the Design Planner view, toggles Expanded Pin View on or off.                                                                                                   |
| F9                | In the Synplify Premier tool, obtains area utilization for the design.<br>Same as Run->Estimate Area: (see <a href="#">Run Menu , on page 498</a> ).               |
| Shift-F9          | In the Synplify Premier tool, compiles the design to include regions as compile point modules in the compiled netlist.<br>Same as Run->Compile Physical Hierarchy. |

# Physical Analyst User Interface

## Synplify Premier

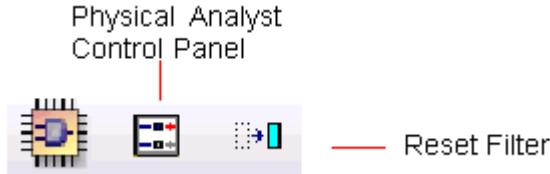
This section describes the graphical user interface (GUI) for the Synplify Premier Physical Analyst tool. See the following topics for details:

- [Physical Analyst Toolbar](#), on page 683
- [Physical Analyst Keyboard Shortcuts](#), on page 684

## Physical Analyst Toolbar

Toolbars provide a quick way to access common menu commands by clicking their icons.

In the Synplify Premier tool, the Physical Analyst toolbar provides access to a visual display of the floorplan, placement, and global routing of the design after running synthesis and place-and-route. The following standard toolbars are available:



The following table describes the default Physical Analyst icons.

| Icon | Description                                                                                                                                                                                                                                                                                                                                           |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|      | Physical Analyst<br>Opens a new Physical Analyst view of your synthesized design. The view displayed is flat, although the hierarchy of the instance name is retained. Use the Physical Analyst commands to control how objects are displayed in this viewer. There are also a number of commands to use for analyzing the netlist that is generated. |
|      | Physical Analyst Control Panel<br>Toggles the display of the Physical Analyst Control panel. Same as selecting Options->Physical Analyst Control Panel                                                                                                                                                                                                |
|      | Reset Filter<br>Resets the original view to hide all nets in the display. Same as selecting Unfilter->Show All Instances, Hide All Nets                                                                                                                                                                                                               |

## Physical Analyst Keyboard Shortcuts

Keyboard shortcuts are key sequences that you type in order to run a command. Menus list keyboard shortcuts next to the corresponding commands. For example, you can press and hold the Ctrl key while you type the g key, instead of using the menu command Go to Location.

The following table describes the Physical Analyst keyboard shortcuts.

| Keyboard Shortcut | Description                                                                                                                                                                        |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ctrl-b            | In the Physical Analyst view, expands paths to all pins for two or more instances selected.                                                                                        |
| Ctrl-e            | In the Design Planner view, toggles Expanded Pin View on or off.                                                                                                                   |
| Ctrl-g            | In the Physical Analyst view, displays the Go to Location dialog box so you can find objects at a particular location.                                                             |
| Ctrl-h            | In the Physical Analyst view, shows the drivers for selected nets.                                                                                                                 |
| Ctrl-i            | In the Physical Analyst view, selects all instances for nets expanding them to their input pins.                                                                                   |
| Ctrl-k            | In a Physical Analyst view, opens the Physical Analyst control panel. Same as Options->Physical Analyst Control Panel (see <a href="#">Using Physical Analyst, on page 1044</a> ). |

| Keyboard Shortcut | Description                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------|
| Ctrl-r            | In the Physical Analyst view, routes selected instances connecting them to their output pins.     |
| Ctrl-u            | In the Physical Analyst view, refreshes the display for the current view.                         |
| Ctrl-Shift-h      | In the Physical Analyst view, selects the drivers for the selected nets.                          |
| Ctrl-Shift-i      | In the Physical Analyst view, only highlights instances expanded to input pins for selected nets. |

# Physical Analyst UI Commands

## *Synplify Premier*

The following sections describe the commands and ways to access these commands in the Synplify Premier Physical Analyst graphical user interface (GUI).

- [Edit Menu: Physical Analyst Commands, on page 686](#)
- [View Menu: Physical Analyst Commands, on page 686](#)
- [HDL Analyst Menu: Physical Analyst Command, on page 691](#)
- [Options Menu: Physical Analyst Commands, on page 691](#)

## Edit Menu: Physical Analyst Commands

The following table describes all the Edit menu Physical Analyst commands:

### Edit Menu Commands for the Physical Analyst View



Find...

Displays the Object Query dialog box, which lets you search your design for instances, symbols, and nets, by name and object type. See [Physical Analyst Find Command , on page 699](#).

## View Menu: Physical Analyst Commands

Use the View menu to set the display and viewing options, choose toolbars, and display result files. The commands in the View menu might vary depending on what is selected within the active view.

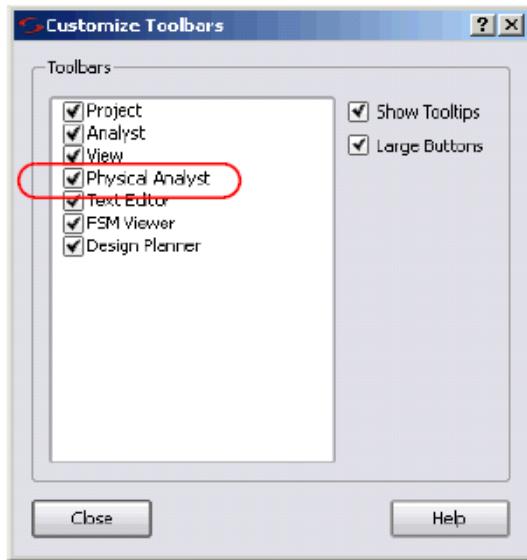
The commands are available when the Physical Analyst view is active. These commands are available in addition to the commands described in [View Menu Commands: All Views, on page 415](#) and [View Menu: Zoom Commands, on page 416](#).

| Command                             | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Filter                              | <p>Shows only the desired objects in the Physical Analyst view. You can filter any of the following:</p> <ul style="list-style-type: none"> <li>• Show Selected - only show selected objects</li> <li>• Hide Selected - hide selected objects</li> <li>• Hide All - hide all objects</li> <li>• Hide All Instances - hide all instances</li> <li>• Hide All Nets - hide all nets</li> <li>• Forward - move forward from a previous filtered display</li> </ul> |
| Unfilter                            | <p>Expands the view to display the desired objects in the Physical Analyst view. You can show any of the following:</p> <ul style="list-style-type: none"> <li>• Reset - resets the design to show all instances and ports, but hide all nets</li> <li>• Reset All - resets the design to show all instances, ports, and nets</li> <li>• Back - moves back to a previous filtered display</li> </ul>                                                           |
| Show Critical Path                  | Enters a filtered state where only instances and nets belonging to the critical timing path are shown in the Physical Analyst view.                                                                                                                                                                                                                                                                                                                            |
| Unselect All                        | Unselect all objects selected in the view.                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Cross Probing                       | Enables/disables cross probing to/from the Physical Analyst tool. See <a href="#">Cross Probing Command (Physical Analyst) , on page 688</a> .                                                                                                                                                                                                                                                                                                                 |
| Configure Enhanced Instance Display | Selects the enhanced display mode where core cells are displayed as fix-sized diamonds in the Physical Analyst view. See <a href="#">Configure Enhanced Instance Display (Physical Analyst) , on page 689</a> .                                                                                                                                                                                                                                                |
| Selection Transcription             | When enabled, allows you to select an object in the Physical Analyst view, then displays the object's tool tip information in the TCL window. This gives you a running history of the objects you have examined. By default, this option is turned on.                                                                                                                                                                                                         |
| Tool Tips                           | Controls the display of tool tips when the cursor is placed over objects in the view.                                                                                                                                                                                                                                                                                                                                                                          |
| Physical Analyst Properties         | Displays information about the design and device in the Physical Analyst view.                                                                                                                                                                                                                                                                                                                                                                                 |

| Command             | Description                                                                                                                                                                       |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Zoom->Zoom Selected | Zooms into the selected object. For other zoom commands, refer to <a href="#">View Menu: Zoom Commands , on page 416</a> .                                                        |
| Go to Location      | Zooms in on the object at the specified coordinate pair location.                                                                                                                 |
| Refresh             | When drawing large designs in the view, you can use the ESC key to cancel the current display. To restart drawing in the view again, use the Refresh option to start the display. |

## Physical Analyst Toolbar Command

Select View->Toolbars to display the Toolbars dialog box, where you can choose the toolbars to display and customize their appearance.



For details, see [Toolbar Command, on page 418](#).

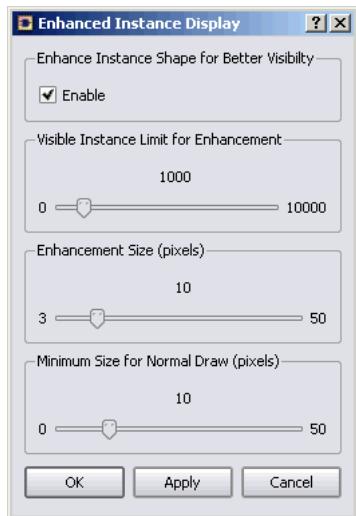
## Cross Probing Command (Physical Analyst)

View->Cross Probing is available in the Physical Analyst view of the Synplify Premier tool. The following enable/disable cross-probing options are available:

| Command                         | Description                                                                                                                                                                                     |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Send Crossprobes when selecting | Enables or disables cross probes from the Physical Analyst. For efficiency reasons, you might want to disable this option (i.e., not send cross probes).                                        |
| Cross Probing from RTL Analyst  | Enables or disables cross probing from the RTL Analyst view. Mapped or placed instances are highlighted in the Physical Analyst view in response to selection of objects in the RTL view.       |
| Cross Probing from Tech Analyst | Enables or disables cross probing from the Technology view. Mapped or placed instances are highlighted in the Physical Analyst view in response to selection of objects in the Technology view. |
| Cross Probing to HDL Source     | Enables or disables cross probing to the HDL source.                                                                                                                                            |
| Auto route cross probed insts   | Enables or disables instances that are cross probed to be automatically routed.                                                                                                                 |

## Configure Enhanced Instance Display (Physical Analyst)

The enhanced display mode, when enabled, causes core cells to be drawn as diamonds of fixed size regardless of zoom level. Selecting View->Configure Enhanced Instance Display in an active Physical Analyst view brings up the Enhanced Instance Display dialog box which is used to set the enhanced display parameters.



| Option                                       | Description                                                                                                               |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| Enhance Instance Shape for Better Visibility | Enables enhanced instance display when checked.                                                                           |
| Visible Instance Limit for Enhancement       | Sets the maximum number of visible core cells that can be displayed in enhanced instance display mode.                    |
| Enhancement Size (pixels)                    | Sets the size (in pixels) of the instance; instances are drawn as diamonds to differentiate them from normal cell shapes. |
| Minimum Size for Normal Draw (pixels)        | Sets the minimum size of an average core cell when the cell is drawn in normal mode and not enhanced                      |

## HDL Analyst Menu: Physical Analyst Command

### *Synplify Premier*

The Project view HDL Analyst->Physical Analyst command displays the Physical Analyst view, which you can use to analyze the project. The Physical Analyst tool provides a visual display of the floorplan, placement, and global routing of the design after synthesis and place-and-route have been run. For more information, see [Using Physical Analyst, on page 1044](#) of the *User Guide*.

## Options Menu: Physical Analyst Commands

### *Synplify Premier*

Use the Options menu to set options for the Physical Analyst view. When using certain technologies, additional menu commands let you run technology-vendor software from this menu.

### Options Menu Commands Specifically for the Physical Analyst View

|                                |                                                                                                                                                |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Physical Analyst Control Panel | Toggles the display of the Physical Analyst control panel. See <a href="#">Using Physical Analyst, on page 1044</a> in the <i>User Guide</i> . |
| Physical Analyst Color Schemes | Allows you to change to another pre-defined color scheme in the Physical Analyst view. Currently, only one standard color scheme is available. |

# Physical Analyst View Popup Menus

## *Synplify Premier*

The popup (right-click) menus in the different tool views are usually comprised of commonly-used menu or toolbar commands. Usually commands in the pop-ups are also available through UI menus or buttons. For example: right-click->Filter can also be run through View->Filter. However, there are some commands that are only available through popup menus. Refer to the following tables for a complete list of Physical Analyst commands:

- [Global Commands](#), on page 692
- [Physical Analyst Popup Commands with Instances Selected](#), on page 695
- [Physical Analyst Popup Commands with Nets Selected](#), on page 696
- [Physical Analyst Popup Menu Property Commands](#), on page 696
- [Physical Analyst Find Command](#), on page 699
- [Resolve Selection Dialog Box](#), on page 703

## Global Commands

### *Synplify Premier*

The following table describes global commands that are commonly used for selected objects and available through a popup (right-click) menu in the Physical Analyst view.

| Command        | Description                                                                                                                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Properties     | Displays the properties of the selected objects (for example, mapped instances or nets). See <a href="#">Instance Properties: Properties (Core Cell) Command</a> , on page 697.                                                                             |
| Selection Tool | Cancels operations, such as when using the measurement tool in the Physical Analyst view, so that you can select other objects in the view thereafter. You can also click the Selection Tool icon on the Physical Analyst toolbar to cancel this operation. |

| Command                                                                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Signal Flow                                                                           | Displays the signal flow of nets. When you select a net, the net is displayed with arrows showing the predominant direction of its signal flow (right, left, up, or down).                                                                                                                                                                                                                                                                                                                                                  |
| Measurement Tool                                                                      | This measurement tool calculates the total distance in microns from a start location to an end location in the X and Y directions and displays these values in the Physical Analyst and Tcl windows.                                                                                                                                                                                                                                                                                                                        |
| Filter                                                                                | Filters the design to display only the desired objects in the view. You can filter any of the following: <ul style="list-style-type: none"> <li>• Show Selected - only show selected objects</li> <li>• Hide Selected - hide selected objects only</li> <li>• Hide All - hide all objects in the view</li> <li>• Hide All Instances - hide all instances in the view</li> <li>• Hide All Nets - hide all nets in the view</li> <li>• Forward - move forward to a filtered display in history of the current view</li> </ul> |
| Unfilter                                                                              | Expands the view to display the desired objects in the view. You can show any of the following: <ul style="list-style-type: none"> <li>• Reset - resets the design to show all instances and ports, but hide all nets</li> <li>• Reset All - resets the design to show all instances, ports, and nets</li> <li>• Back - moves back to a previous filtered display in history of the current view</li> </ul>                                                                                                                 |
| Find                                                                                  | See <a href="#">Physical Analyst Find Command , on page 699</a> .                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Select<br>->All Instances and Nets<br>->All Instances<br>->All Nets<br>->Deselect All | Globally selects and highlights any of the following in the view: all instances and nets, all instances only, or all nets only. Conversely, you can globally deselect and unhighlight all selected objects in the view.                                                                                                                                                                                                                                                                                                     |

| Command                                                                                                                  | Description                                                                                                                                                                                                                                                                                                                                     |
|--------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Critical Path<br>->  Show Critical Path | Ways to display the critical path: <ul style="list-style-type: none"><li>• Enters a filtered state where only instances and nets belonging to the critical timing path are shown.</li><li>• Traces the critical path from the start point to the end point.</li><li>• Traces the critical path from the end point to the start point.</li></ul> |
| ->Expand Path Forward                                                                                                    |                                                                                                                                                                                                                                                                                                                                                 |
| ->Expand Path Backward                                                                                                   |                                                                                                                                                                                                                                                                                                                                                 |
| Zoom Selected                                                                                                            | Zooms in on the selected object.                                                                                                                                                                                                                                                                                                                |
| Markers<br>->Add marker<br>->Remove All<br>->Remove Selected<br>->Go to Next<br>->Go to Previous                         | Create any marker at a specified location. You can: <ul style="list-style-type: none"><li>• add a marker</li><li>• remove all markers</li><li>• remove a selected marker</li><li>• move forward to the next marker in the order they were created</li><li>• move backwards to the previous marker in the order they were created</li></ul>      |
| Go to Location                                                                                                           | Allows you to specify a coordinate pair location for an object and then zoom in on the object if requested.                                                                                                                                                                                                                                     |
| Selection Transcription                                                                                                  | When enabled, allows you to select an object in the Physical Analyst view, then displays the object's tool tip information in the TCL window. This gives you a running history of the objects you have examined. By default, this option is enabled.                                                                                            |
| Tool Tips                                                                                                                | Enables/disables tool tips to be displayed as you move the mouse over objects in the view, providing additional information for these objects at-a-glance.                                                                                                                                                                                      |
| Refresh                                                                                                                  | When drawing large designs in the view, you can use the ESC key to cancel the current display. To restart drawing in the view again, use the Refresh option to start the display.                                                                                                                                                               |
| Physical Analyst Properties                                                                                              | Displays information about the design and device. See <a href="#">Physical Analyst Popup Menu Property Commands</a> , on page 696.                                                                                                                                                                                                              |

## Physical Analyst Popup Commands with Instances Selected

### *Synplify Premier*

When one or more instances are selected in the Physical Analyst view, certain commands are included in popup menus, regardless of where you right-click. The following commands are available when instances are selected. Note that some commands are only available from this popup menu.

| Instance-specific Command                                                                 | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Properties (Core Cell)                                                                    | Displays information about the cell you select. See <a href="#">Instance Properties: Properties (Core Cell) Command</a> , on page 697.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Expand<br>->Output Pins<br>->Input Pins<br>->All Pins<br>->Selected Pins                  | <p>Expands logic starting from an instance or instances. In general, start with some visible instances and filter or hide the remainder. You can then expand either from:</p> <ul style="list-style-type: none"> <li>• all pins</li> <li>• selected pins</li> <li>• input pin</li> <li>• output pins</li> </ul> <p>Expansion reveals (unfilters and selects) instances to which the instance connects following the signal direction of pins. Expansion is stopped when reaching a register, black box, or port.</p> <p>As you expand, nets are shown connecting the instances. If you enable net pruning the portions of nets connecting to invisible instances are dimmed.</p> |
| Expand to Register/Port<br>->Output Pins<br>->Input Pins<br>->All Pins<br>->Selected Pins | See all cells that are connected to a pin, output pins, input pins, or all pins, up to the next register/port.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Expand Paths<br>->All Pins<br>->Selected Pins                                             | Shows all logic between two or more selected instances. Unfilters and selects instances and nets along the signal paths of the selected instances. As in the case of Expand you can control the pins to expand (all pins, selected pins).                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**Instance-specific Command Description**

|                          |                                                                                                                                             |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| Route Selected Instances | Shows nets connected to selected instances. As in the case of Expand you can control the pins to route (all pins, input pins, output pins). |
| ->Connected              |                                                                                                                                             |
| ->Output Pins            |                                                                                                                                             |
| ->Input                  |                                                                                                                                             |
| ->All Pins               |                                                                                                                                             |

**Physical Analyst Popup Commands with Nets Selected***Synplify Premier*

When one or more nets are selected, the following context-sensitive commands are included in popup menus, regardless of where you right-click.

**Net-specific Command Description**

|                                 |                                                                                                                                                                                                     |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Properties (Net)                | Displays information about the net you select. See <a href="#">Instance Properties: Properties (Core Cell) Command , on page 697</a> .                                                              |
| Select Net Instances            | Selects all instances connected to the selected net. You can expand the net from input pins, output pins, or all pins. Unfilters and selects instances along the signal paths of the selected pins. |
| ->Input Pins                    |                                                                                                                                                                                                     |
| ->Output Pins                   |                                                                                                                                                                                                     |
| ->All Pins                      |                                                                                                                                                                                                     |
| Highlight Visible Net Instances | Shows only nets connected to the selected instances. The net can be expanded from input pins, output pins, or all pins.                                                                             |
| ->Input Pins                    |                                                                                                                                                                                                     |
| ->Output Pins                   |                                                                                                                                                                                                     |
| ->All Pins                      |                                                                                                                                                                                                     |
| Select Net Driver               | Selects the driver for a net.                                                                                                                                                                       |
| Go to Net Driver                | Shows and scrolls to the driver for a selected net.                                                                                                                                                 |

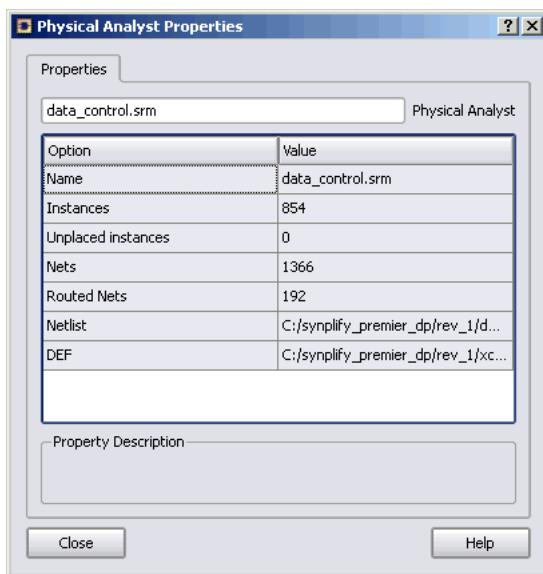
**Physical Analyst Popup Menu Property Commands***Synplify Premier*

You can display property information for the design or objects in the Physical Analyst view, using various popup commands:

- [Design Properties: Physical Analyst Properties Command](#), on page 697
- [Instance Properties: Properties \(Core Cell\) Command](#), on page 697
- [Net Properties: Properties \(Net\) Command](#), on page 698

## Design Properties: Physical Analyst Properties Command

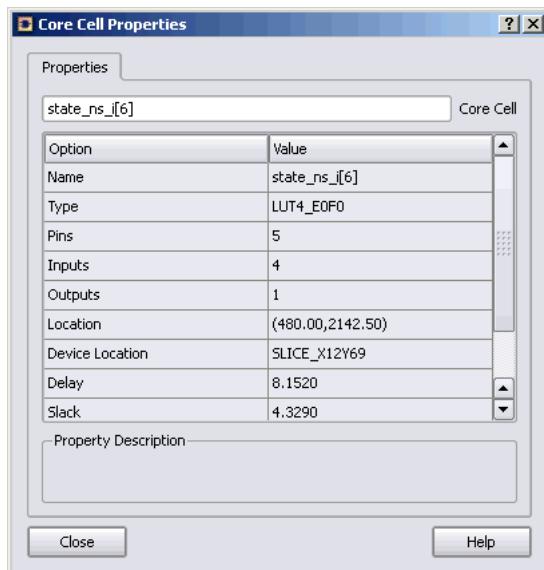
To display the Physical Analyst Properties dialog box, right-click anywhere in the Physical Analyst view and select Physical Analyst Properties from the bottom of the popup menu. The dialog box has a read-only Design pane, which lists information like the design name, the number of instances, unplaced instances, routed nets in the design, and the location of the netlist and floor-plan (def) files.



## Instance Properties: Properties (Core Cell) Command

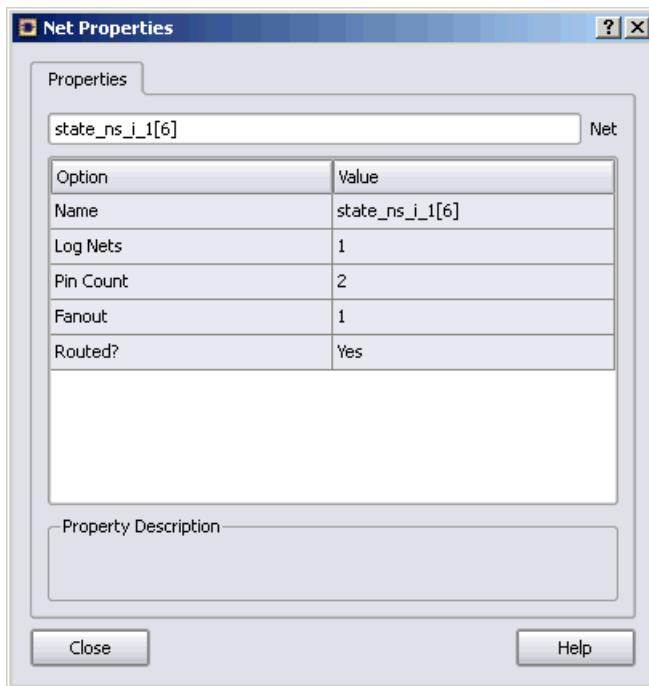
To display the properties for an instance, right-click the instance and select Properties (Core Cell). The dialog box lists read-only information like the instance name, type, number of pins, placement location, device-specific location, delay, slack, and clock signal. It also indicates if the instance is included in the critical path.

The following example shows the properties for a core cell that is selected in the Physical Analyst view.



## Net Properties: Properties (Net) Command

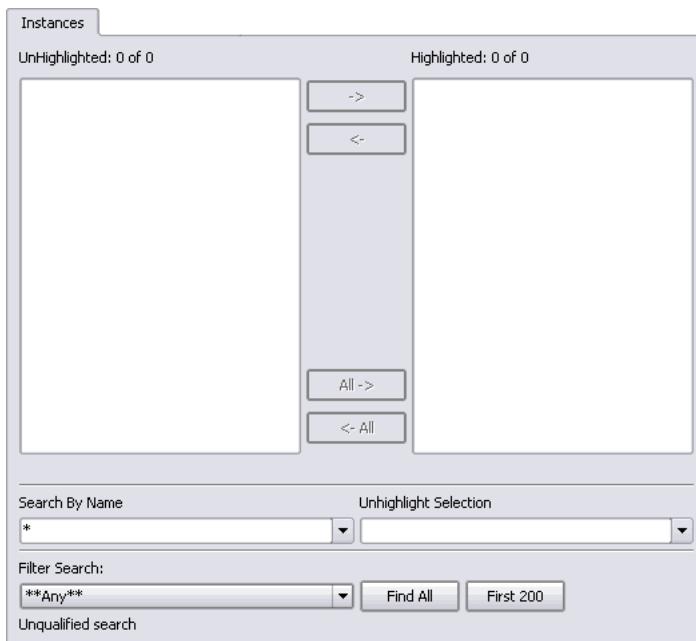
To display the properties for a net, right-click on the net and select Properties (Net). The dialog box lists read-only information like the net name, logical nets, pin count, and fanout. It also indicates if the net is globally routed, and if the net is a clock.



## Physical Analyst Find Command

*Synplify Premier*

In the Physical Analyst view, right-click and select Find from the popup menu or press Ctrl-f to open the Find Object dialog box. The view displayed is flat, although the hierarchy of instance names is retained.



Objects listed in the Highlighted window of Find Object are also highlighted in the HDL Analyst and/or Physical Analyst views. The following table provides descriptions for the parameters in this dialog box. More sections about filtering and searching for objects are presented after the table.

| Option                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Instances, Symbols, Nets, Ports | Specifies the design objects for which to search. In terms of memory consumption, searching for Instances requires the least amount, searching for Nets requires the most.                                                                                                                                                                                                                                                                                                     |
| Search                          | Limits/expands the search to the entire design.                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Search by Name                  | In the Physical Analyst tool, use with the Find 200 and Find All, for a case-sensitive search of the string that you specify in this field. The string can contain the following wildcard characters: <ul style="list-style-type: none"> <li>• * (asterisk) - matches any sequence of characters;</li> <li>• ? (question mark) - matches any single character;</li> <li>• . (period) - does not match any characters, but indicates a change in hierarchical level.</li> </ul> |

| Option                      | Description                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Filter Search               | In the Physical Analyst tool, use with the Find 200 and Find All, for a lists object names in the UnHighlighted area. Items in this field depend on the design object selected. See <a href="#">Object Filter Search for Find Command , on page 701</a> .                                                                                                                                          |
| UnHighlighted               | Lists all objects in the design.                                                                                                                                                                                                                                                                                                                                                                   |
| Highlighted                 | All objects in this section are highlighted in the current schematic or floorplan view. To highlight objects in the schematics, select them from the unhighlighted section and move them to the highlighted section using the arrows or by double-clicking the objects.<br>You can select multiple objects using the Ctrl or Shift key. To deselect an object, press Ctrl and click on the object. |
| Highlight Search (*?)       | In Physical Analyst, you can further define the search for instances, symbols, nets, or ports. See <a href="#">Object Filter Search for Find Command , on page 701</a> .                                                                                                                                                                                                                           |
| Un-Highlight Selection (*?) | Uses wildcards for filtering objects to deselect in the schematic views.                                                                                                                                                                                                                                                                                                                           |
| Jump to location            | Displays the target on any sheet in the schematic.                                                                                                                                                                                                                                                                                                                                                 |
| Find 200                    | Used with the Highlight Search (*) filter string, displays the first 200 objects that match the filter string in the UnHighlighted list. Each time you click the button, an additional 200 objects are added to the list. Find 200 does not highlight objects in the schematic or floorplan views, but filters what is listed in the Unhighlighted field.                                          |
| Find All                    | Used with the Highlight Search (*) filter string, displays all objects that match the filter string in the UnHighlighted list. Find All does not highlight objects in the schematic or floorplan views, but filters what is listed in the Unhighlighted field.                                                                                                                                     |

## Object Filter Search for Find Command

The Physical Analyst lets you fine-tune your search for design objects by filtering the kind of objects you are searching. In the Find Object dialog box (right-click->Find in a Physical Analyst view). The Highlight Search (\*) field has a pull-down menu that lists the additional object filters available for the current type of object, based on the Find object tab selected.

The following tables list the choices available for the Net, Instance, Port, and Symbol tabs. Design objects matching the search are listed in the Unhighlighted field in the dialog box. To highlight the objects in the view, use the arrows to move the desired objects to the Highlighted field.

## Instances

From the Instances panel, the pull-down list contains the following filters:

| <b>Selection</b>    | <b>Only searches ....</b>                                              |
|---------------------|------------------------------------------------------------------------|
| Combinational       | Combinational instances                                                |
| Cover placement     | Fixed macro placements that cannot be changed                          |
| Core cell           | Cells (instances) in the core area                                     |
| Critical path       | Instances on a critical path                                           |
| Critical path end   | Instances at the end of a critical path                                |
| Critical path start | Instances at the start of a critical path                              |
| Fixed placement     | Pre-placed or fixed macro placement                                    |
| Macro cell          | Macro cells (instance)                                                 |
| Module              | Groups of instances that form logical modules                          |
| Negative slack      | Instances with negative slack                                          |
| Pad cell            | Pad cells on the perimeter of the device                               |
| Placed              | Placed instances                                                       |
| Placed placement    | Pre-placed or fixed macro placement that can be changed by the program |
| Primitive           | Primitives                                                             |
| Sequential          | Sequential instances                                                   |
| Unplaced            | Unplaced instances                                                     |

## Symbols

From the Symbols panel, the pull-down list contains the following filters:

| <b>Selection</b> | <b>Only searches ...</b> |
|------------------|--------------------------|
| Module           | Hierarchical module      |
| Primitive        | Primitives               |

| Module | Hierarchical module |
| Primitive | Primitives |

## Nets

From the Nets panel, the pull-down list contains the following filters

| <b>Selection</b> | <b>Only searches ...</b> |
|------------------|--------------------------|
| Clock            | Nets from clock ports    |

| Clock | Nets from clock ports |

## Ports

From the Ports panel, the pull-down list contains the following filters

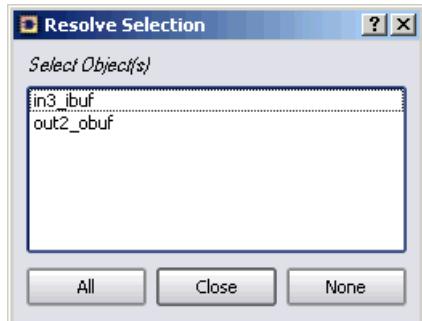
| <b>Selection</b> | <b>Only searches ...</b> |
|------------------|--------------------------|
| Bidir            | Bidirectional ports      |
| Clock            | Clocks                   |
| Input            | Input ports              |
| Output           | Output ports             |

| Bidir | Bidirectional ports |
| Clock | Clocks |
| Input | Input ports |
| Output | Output ports |

## Resolve Selection Dialog Box

### *Synplify Premier*

This box lists overlapping objects at a particular cursor location in the Physical Analyst and helps you resolve selection by selecting objects from the list. This box opens when you click the cursor at a location where there are over overlapping objects.



## CHAPTER 8

# Unified Power Format Commands

---

The commands for IEEE Unified Power Format (UPF) in the FPGA flow provide portable, low-power design specifications that allow interoperability across various tools in system design, verification, and validation. The relevant UPF support for FPGAs is limited to the commands that specify the following functionality:

- Isolation
- Retention
- Power Domains
- Scope Resolution

For more information, see [Configuring UPF for FPGA Designs, on page 1223](#).

The FPGA synthesis tool only supports the following UPF commands and their respective options in a .upf script file. All other commands and options in the .upf script file are parsed and ignored.

|                                    |                                  |
|------------------------------------|----------------------------------|
| <code>associate_supply_set</code>  | <code>corrupt_pd</code>          |
| <code>create_power_domain</code>   | <code>create_power_switch</code> |
| <code>create_supply_set</code>     | <code>load_upf</code>            |
| <code>map_retention_cell</code>    | <code>name_format</code>         |
| <code>set_domain_supply_net</code> | <code>set_equivalent</code>      |

[set\\_isolation](#)[set\\_isolation\\_control](#)[set\\_retention](#)[set\\_retention\\_control](#)[set\\_scope](#)

## associate\_supply\_set

*Synplify Premier  
UPF*

The `associate_supply_set` command associates the supply set or supply set handle to be used for the power domain.

### Syntax

```
associate_supply_set supplySetRef [-handle {supplySetHandle}]
```

*supplySetRef*

The root name of the supply set or supply set handle.

**-handle {supplySetHandle}**

Specifies the functionality for which the supply net is associated.

### Description

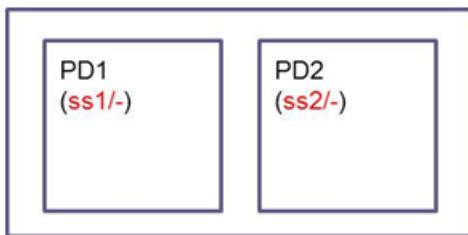
The `associate_supply_set` command associates the supply set or supply set handle to be used for the power domain. The supply set handle provides the functionality for which the supply net is associated.

### Examples

```
create_power_domain PD1
create_power_domain PD2

create_supply_set ss1
create_supply_set ss2

associate_supply_set ss1 -handle PD1.primary
associate_supply_set ss2 -handle PD2.primary
```



## corrupt\_pd

*Synplify Premier  
UPF*

Controls the corruption of sequential elements (RAM or flip-flops) for specific power domains when it is shut-off.

### Syntax

```
corrupt_pd -domain all | domainName | instanceName
 [-mode {all0s | all1s | random | none}]
```

#### -domain

Defines the power domains or instances for which corruption is applied.

- **all**  
Specifies all the switchable power domains to control corruption.
- ***domainName***  
Specifies the name of the power domain to control corruption.
- ***instanceName***  
Specifies the name of the instance to be ignored from corruption.

#### -mode

Implements sequential elements to one of the states below when the power domain is shut-off.

- **all0s**  
Corruption of sequential elements is set to all zeros. This is the default.
- **all1s**  
Corruption of sequential elements is set to all ones.
- **random**  
Corruption of sequential elements is randomly set to all0s and all1s.

none

No corruption occurs.

### Description

In the FPGA, when a power domain is powered off, the state of the domain becomes unknown or corrupted. UPF support implements isolation and retention logic so that the power domain is held to a known state during shut-off. Use the `corrupt_pd` command to specify how to implement corruption for instances or the sequential elements of specific power domains.

## Examples

Here are examples of various usage scenarios.

### Examples 1:

Enable all0s corruption for all domains globally.

```
corrupt_pd -domain all -mode all0s
```

Enable all1s corruption for power domain PD1.

```
corrupt_pd -domain PD1 -mode all1s
```

Ignore power domain PD2 from corruption.

```
corrupt_pd -domain PD2 -mode none
```

### Examples 2:

Enable all1s corruption for power domain PD1.

```
corrupt_pd -domain PD1 -mode all1s
```

Enable all0s corruption for power domain PD2.

```
corrupt_pd -domain PD2 -mode all0s
```

Enable random corruption for power domain PD3.

```
corrupt_pd -domain PD3 -mode random
```

### Examples 3:

Enable all0s corruption globally.

```
corrupt_pd -domain all -mode all0s
```

Ignore sequential instance inst from corruption.

```
corrupt_pd -domain {i:inst} -mode {none}
```

#### Examples 4:

Enable all1s corruption for power domain PD1.

```
corrupt_pd -domain {PD1} -mode all1s
```

Ignore sequential instance u1/inst from corruption.

```
corrupt_pd -domain {u1/inst} -mode {none}
```

Note that warnings are generated for the following conditions:

- Corruption for the power domain will be ignored because the specified power domain does not exist.
- Corruption for the power domain will be ignored because it does not include the specified switch, such as all1s.

## create\_power\_domain

*Synplify Premier  
UPF*

Creates a power domain, which provides a power supply distribution network.

### Syntax

```
create_power_domain domainName [-elements {list}] [-include_scope]
 [-scope {scopeInstance} [-supply {supplySetHandle [supplySetRef]}]]
```

*domainName*

The name of the power domain.

#### **-elements {list}**

Specifies a list of design elements assigned to the power domain. Note that the -elements option can be specified with the wildcard (\*) character for this command.

#### **-include\_scope**

Defines the extent of the power domain. This includes the active scope, and by default, all of its descendant scopes.

#### **-scope {scopeInstance}**

Defines the scope (hierarchical logic level) for the power domain that is created. By default, the power domain is created in the current scope.

#### **-supply {supplySetHandle [supplySetRef]}**

Specifies the supply set name associated with the supply set handle, where the specified supply set will be equivalent to the supply set handle. By default, the power domain uses the following supply net handles:

- primary – Primary supply set for the power domain.
- default\_isolation – Default isolation supply set if isolation power is not defined.
- default\_retention – Default retention supply set if retention power is not defined.

## Description

The `create_power_domain` command creates a power domain in the specified scope. A power domain contains a collection of design elements that share a primary power net and ground power net. The scope of the power domain is the hierarchical logic level in which the power domain is created. The set of design elements belonging to the power domain is referred to as the extent of that power domain. For more information, see [Specifying UPF Power Domains, on page 1225](#).

## Examples

```
create_power_domain PD_Top -include_scope
create_power_domain PD1 -elements {U1}
create_power_domain PD1 -supply {primary ss1}
create_power_domain PD2 -supply {primary ss2}
```

The following example shows how the wildcard (\*) character can be used with the `-elements` option for this command.

```
create_power_domain PD -elements {\
inst_top/inst[*].gen/mod*
inst_top/*/inst*
inst_top/inst_mod/mod
}
```

## create\_power\_switch

*Synplify Premier  
UPF*

The `create_power_switch` command used in combination with the `syn_corrupt_pd` attribute setting, implements logic to force corruption of the sequential elements (RAM or flip-flops) within a power domain when it is shut-off.

### Syntax

```
create_power_switch switchName -domain domainName
 {-control_port {portName netName}}*
 {-on_state {stateName inputSupplyPort {booleanExpression}}}*/
 [-off_state {stateName {booleanExpression}}]
 [-ack_port {portName netName {booleanExpression}}]
```

**switchName**

The name of the power switch for a power domain.

**-domain {domainName}**

Name of the power domain that contains the specified power switch.

**-control\_port {portName netName}**

Specifies a control port on the power switch and the net to which this port is connected. The `-control_port` argument is required and can be repeated.

**-on\_state {stateName inputSupplyPort {booleanExpression}}**

Implements the power domain on-state for the input supply port defined with its corresponding boolean expression. The `-on_state` argument is required and can be repeated.

**-off\_state**

`{stateName {booleanExpression}}`

Optional argument that Implements the power domain off-state defined with its corresponding boolean expression.

**-ack\_port {portName netName {booleanExpression}}**

Specifies the name of the acknowledge port for the power switch and the logical net to which this port is connected. Optionally, you can specify a boolean expression for the specified control ports.

For the Boolean expression syntax in the above arguments, only the following operator (plus parenthesis) can be used:

| Operators                    | Description                     |
|------------------------------|---------------------------------|
| <code>~ &amp;   ^</code>     | Bit-wise negation, AND, OR, XOR |
| <code>! &amp;&amp;   </code> | Logical negation, AND, OR       |

## Description

In the FPGA, when a power domain is powered off, the state of the domain becomes unknown or corrupted. When isolation and retention strategies are employed, the state of the power domain is restored on power up. With FPGA prototyping, UPF support implements isolation and retention logic so that the power domain is held to a known state during shut-off. The on/off state of the power domain is determined using the `create_power_switch` command, and corruption is enabled by setting the `syn_corrupt_pd` attribute that implements logic to force the sequential elements to one of the below states when the domain is shut-off:

- All ones
- All zeros
- Random ones and zeros

Adding power domain corruption to FPGA prototyping increases coverage and adds confidence that the design will continue to function reliably when the FPGA is fabricated. For more information, see [syn\\_cp\\_use\\_fast\\_synthesis, on page 181](#).

Additionally, the power switch drives the acknowledge port with the control ports based on the `on_state`/`off_state` condition or the boolean expression specified.

## Examples

```
create_power_switch sw1 \
 -domain PD1 \
 -control_port {ctrl_small ON1} \
 -control_port {ctrl_large ON2} \
 -control_port {ss SUPPLY_SELECT} \
 -on_state {full_s1 vin1 {ctrl_small & ctrl_large & ss}} \
 -off_state {not_required {!ctrl_small & !ctrl_large}} \
 -ack_port {ack_p ACKN {ctrl_small & !ctrl_large}}
```

## Restrictions

Control port names must not conflict with SystemVerilog keywords.

## create\_supply\_set

*Synplify Premier  
UPF*

The `create_supply_set` command explicitly creates supply sets. A supply set is a bundle of supply nets.

### Syntax

```
create_supply_set supplySetName
 [-function {power netName} -function {ground netName}]
 [-update]
```

*supplySetName*

The name of the supply set.

**-function {power netName} -function {ground netName}**

A power/ground pair that specifies the functionality for the corresponding supply net (*netName*). The supported function arguments are power and ground.

**-update**

Updates an existing supply set with the specified supply sets.

### Description

The `create_supply_set` command explicitly defines supply sets in the UPF file. The supply set can be used to provide more flexibility for isolating cells in a power domain when using the `-source/-sink/-diff_supply_only` arguments for the `set_isolation` command. To implicitly create supply sets, see [create\\_power\\_domain, on page 711](#).

### Example 1

```
create_supply_set ss1
create_supply_set ss2

set_domain_supply_net PD1 -primary_power_net ss1.power
 -primary_ground_net ss1.ground
set_domain_supply_net PD2 -primary_power_net ss2.power
 -primary_ground_net ss2.ground
```

```
create_supply_set ss1 -function {power VDD} -function {ground GND}
 -update
create_supply_set ss2 -function {power VDD1} -function{ground GND}
 -update
```



## Example 2

```
create_supply_set ss1
create_supply_set ss2

set_domain_supply_net PD1 -primary_power_net ss1.power
 -primary_ground_net ss1.ground
set_domain_supply_net PD2 -primary_power_net ss2.power
 -primary_ground_net ss2.ground

create_supply_set ss1 -function {power VDD} -function {ground GND}
 -update
create_supply_set ss2 -function {power ss1.power}
 -function {ground ss1.ground} -update
```



## load\_upf

*Synplify Premier  
UPF*

Loads the script file that executes the UPF commands.

### Syntax

**load\_upf** *upfFilename* [**-scope** *instanceName* ]

*upfFilename*

The name of the script file containing the UPF commands to be executed.

**-scope** *instanceName*

Specifies the scope for the UPF commands. The scope defines the hierarchical instance for which the UPF commands are applied.

### Description

The **load\_upf** command reads the UPF script file and executes its commands for the defined scope. For more information, see [Specifying UPF Power Domains, on page 1225](#).

### Examples

```
load_upf U1.upf -scope U1
```

## map\_retention\_cell

*Synplify Premier  
UPF*

The `map_retention_cell` command lets you customize retention for sequential elements. To specify how the `map_retention_cell` command can be implemented using the `set_option` command, see [set\\_option, on page 149](#).

### Syntax

```
map_retention_cell retentionName -domain domainName [-elements list]
[-lib_cells list] [-lib_cell_type libCellType]
```

**-map\_retention\_cell retentionName**

Name of the retention strategy.

**-domain domainName**

Specifies the power domain for which the retention strategy is applied.

**-elements {list}**

Specifies a list of design elements, sequential registers, or signals of sequential elements to be mapped for retention.

**-lib\_cells list**

Specifies library cells for the elements to be mapped for retention.

**-lib\_cell\_type libCellType**

Specifies the library cells identified by the retention attribute, with the same retention behavior as that of the inferred RTL for the sequential elements.

### Example

Here are examples of how the `map_retention_cell` command can be used:

```
set_retention RET -domain PD -elements {e1 e2}
map_retention_cell RET -domain PD -lib_cells RET_FF

map_retention_cell RET -domain PD -lib_cells RET_FF
 or
map_retention_cell RET -domain PD -lib_cell_type RET_FF_TYPE
 or
map_retention_cell RET -domain PD -lib_cells RET_FF
 -lib_cell_type RET_FF_TYPE
```

## name\_format

*Synplify Premier  
UPF*

The name\_format command lets you control how to specify the naming of isolation cells for UPF commands.

### Syntax

**name\_format [-isolation\_prefix *prefixName*] [-isolation\_suffix *suffixName*]**

**-isolation\_prefix *prefixName***

Specifies the prefix to be used with the isolation cell name. The default is "".

**-isolation\_suffix *suffixName***

Specifies the suffix to be used with the isolation cell name. The default is "\_UPF\_ISO".

### Example

```
name_format -isolation_suffix _U1_ISO
```

## set\_domain\_supply\_net

*Synplify Premier  
UPF*

The `set_domain_supply_net` command specifies the primary supply set for the power domain.

### Syntax

```
set_domain_supply_net domainName [-primary_power_net {supplyNetName}]
[-primary_ground_net {supplyNetName}]
```

*domainName*

The domain name for which the default supply nets are applied.

**-primary\_power\_net {*supplyNetName*}**

Specifies the primary power supply net.

**-primary\_ground\_net {*supplyNetName*}**

Specifies the primary ground net.

### Description

The `set_domain_supply_net` command specifies the primary supply set for the power domain.

### Example 1

```
set_domain_supply_net PD1 -primary_power_net VDD
-primary_ground_net GND

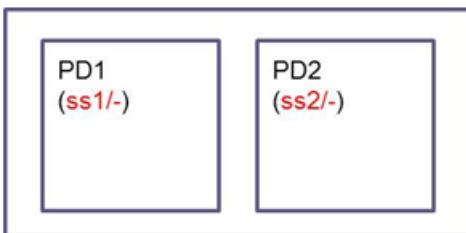
set_domain_supply_net PD2 -primary_power_net VDD1
-primary_ground_net GND
```



## Example 2

```
create_supply_set ss1
create_supply_set ss2

set_domain_supply_net PD1 -primary_power_net ss1.power
 -primary_ground_net ss1.ground
set_domain_supply_net PD2 -primary_power_net ss2.power
 -primary_ground_net ss2.ground
```



## set\_equivalent

*Synplify Premier  
UPF*

The **set\_equivalent** command allows you to specify two supply nets or supply sets to be equivalent.

### Syntax

```
set_equivalent {-net supplyNets | supplyPorts} | {-set supplySets | supplySetHandles}
-net1 {supplyNets | supplyPorts}
 Specifies the supply nets to be equivalent. When supply ports are specified for the argument, equivalence is applied to the supply nets connected to the port.
-set1 {supplySets | supplySetHandles}
 Specifies the supply sets to be equivalent. Equivalence is applied between the supply nets for the corresponding functions of the supply sets.
```

<sup>1</sup> When you specify the **set\_equivalent** command, only one of the arguments (-net or -set) can be used.

### Description

The **set\_equivalent** command enhances support for supply set association and can impact supply-dependent isolation strategies using the **-source/-sink** or **-diff\_supply\_only** options.

### Example

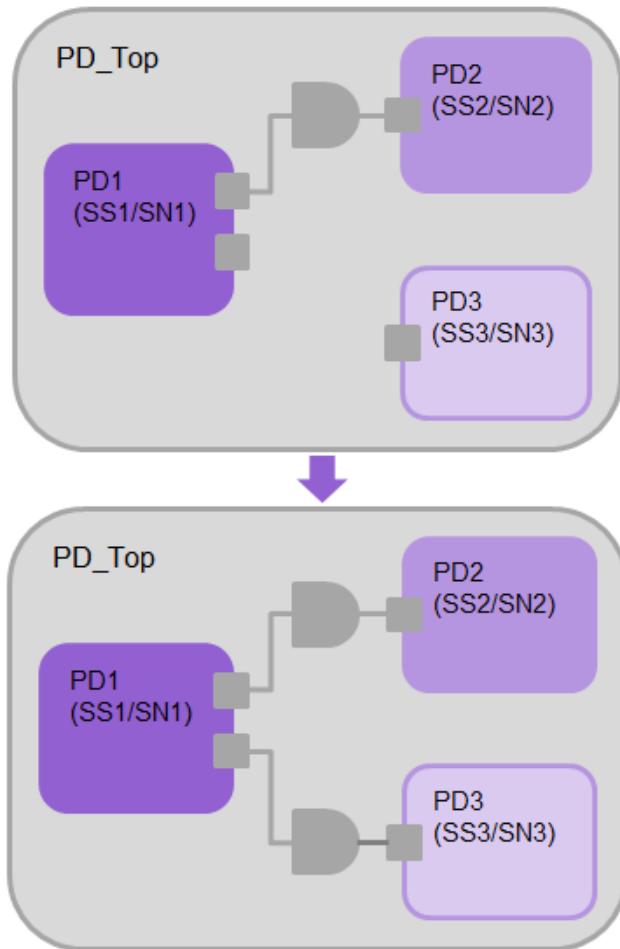
The **set\_equivalent** command can be used as follows:

- Specify isolation strategy with **-source/-sink**.

```
set_isolation iso_PD1 -domain PD1 -applies_to outputs -sink SS2
```

Then, add the **set\_equivalent** command.

```
set_equivalent -sets {SS2 SS3} / set_equivalent -sets
 {SS2 PD3.primary} / set_equivalent -nets {SN2 SN3}
set_isolation iso_PD1 -domain PD1 -applies_to outputs -sink SS2
```

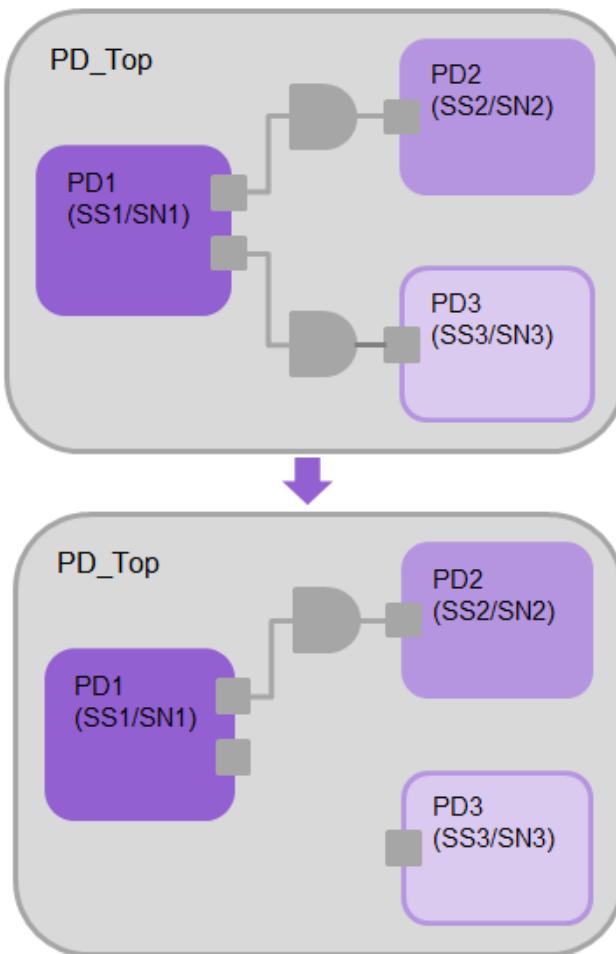


- Specify isolation strategy with `-diff_supply_only`.

```
set_isolation iso_PD1 -domain PD1 -applies_to outputs
 -diff_supply_only TRUE
```

Then, add the `set_equivalent` command.

```
set_equivalent -sets {SS1 SS3} / set_equivalent -sets
 {PD1.primary SS3} / set_equivalent -nets {SN1 SN3}
set_isolation iso_PD1 -domain PD1 -applies_to outputs
 -diff_supply_only TRUE
```



## set\_isolation

*Synplify Premier  
UPF*

Specifies the isolation strategy for a power domain and the elements applied with this strategy.

### Syntax

```
set_isolation strategyName [-domain domainName] [-elements {list}]
[-applies_to {inputs | outputs | both}] [-no_isolation]
[-clamp_value {0 | 1 | latch}] [-source {sourceSupplyRef}]
[-sink {sinkSupplyRef}] [-diff_supply_only {false | true}]
```

***strategyName***

Specifies the isolation strategy name.

The command must be specified with the same `set_isolation_control` strategy name.

**-domain *domainName***

Specifies the power domain for which the isolation strategy is applied.

**-elements {*list*}**

Specifies a list of power domain boundary ports to which the isolation strategy is applied.

Note that the `-elements` option can be specified with the wildcard (\*) character for this command. For example:

**For SystemVerilog Port**

**You can specify the isolation element as ...**

sv[1:0][1:0]

- sv\*
- sv[0][\*]
- sv[0]

- bus[0].ele1[7:0]
- bus[0].ele2[7:0]
- bus[0].ele3[7:0]

- bus[0].ele1\*
- bus[0].ele2[\*]
- bus[0].ele3

**-applies\_to {inputs | outputs | both}**

Defines whether input ports, output ports, or both input and output ports are isolated for the domain.

**-no\_isolation**

Prohibits isolation from occurring for the specified isolation strategy.

**-clamp\_value {0 | 1 | latch}**

Defines the clamp value of an isolated port for the isolation signal. The default is 0.

**-source {sourceSupplyRef}**

Defines the name of the supply set reference for the source to which the isolation strategy is applied.

**-sink {sinkSupplyRef}**

Defines the name of the supply set reference for the sink to which the isolation strategy is applied.

**-diff\_supply\_only {false | true}**

Inserts an isolation cell only if the driver and receiver supply sets are different. The default value is false.

## Description

The `set_isolation` command specifies the isolation strategy for a power domain and the elements in the domain that the strategy is applied. An isolation strategy specification includes the enable signal net and clamp value, and specifies whether inputs, outputs, or both inputs and outputs are isolated for the domain.

All isolation strategies defined with this command except those specifying no isolation, must have a corresponding set of control commands defined with the `set_isolation_control` command. This command is described in [set\\_isolation\\_control, on page 730](#).

## Use Tcl Variables with the `set_isolation` Command

The `set_isolation` command can be used with the following Tcl variables to control how the `-applies_to` option is implemented:

- `enable_upf_1_0_applies_to_default_output` – When the `-applies_to` option is not specified, use with the following syntax to control how the power domain is instrumented:

```
set_option enable_upf_1_0_applies_to_default_output true|false
```

- `upf_iso_filter_elements_with_applies_to` – When the `-applies_to` option is specified with the `-elements` option, use with the following syntax to filter elements base on the direction of the `-applies_to` for the power domain:

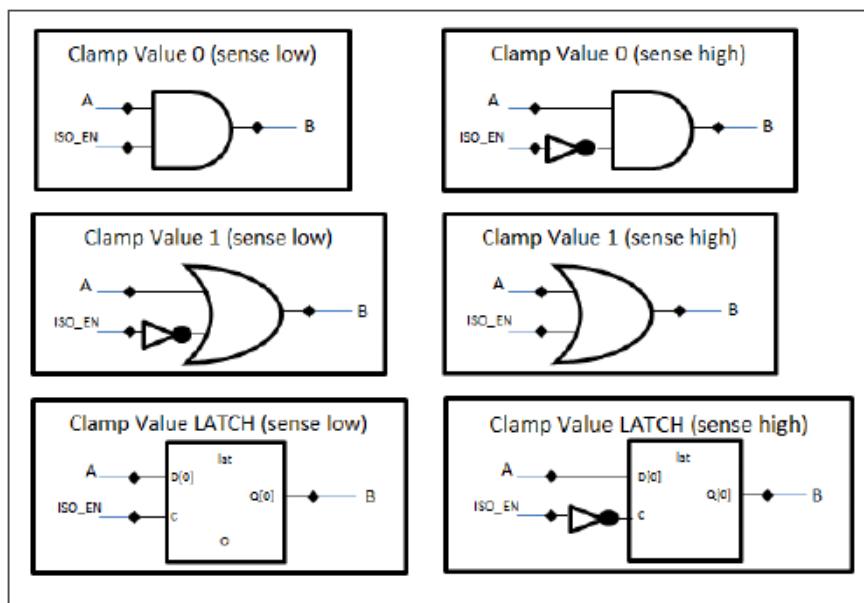
```
set_option upf_iso_filter_elements_with_applies_to enable|disable|error
```

Prior to UPF 1.0, the default isolation side (in the absence of `-applies_to`) was outputs which was changed to both beginning with the UPF2.0 standard. This TCL variable provides backwards compatibility to revert the isolation to the output side for UPF1.0 strategies instead of isolating both sides.

For details on setting this option, see [set\\_option, on page 149](#) in the *FPGA Synthesis Command Reference*.

## Isolation Cells

The following diagrams show how clamp value and sense are applied to isolation cells.



## Examples

```
set_isolation PD1_ret -domain PD1 -elements
{U1/out_reg U2/out_reg}
```

```
set_isolation PD1_iso -domain PD1 -clamp_value 1
-applies_to outputs

set_isolation PD1_no_iso -domain PD1 -no_isolation
-elements {port1 port2}

set_isolation iso_PD1 -domain PD1 -applies_to both
-diff_supply_only true
```

## set\_isolation\_control

*Synplify Premier  
UPF*

Identifies the isolation strategy and specifies the isolation control signals for this strategy.

### Syntax

```
set_isolation_control strategyName [-domain domainName]
 [-isolation_signal signalName] [-isolation_sense {high | low}]
 [-location {self | parent | fanout}]
```

*strategyName*

Specifies the isolation strategy name.

The `set_isolation_control` command must be specified with the same `set_isolation` strategy name.

**-domain** *domainName*

Specifies the power domain for which the isolation strategy is applied.

**-isolation\_signal** *signalName*

Specifies the isolation signal name for the net, port, or pin for which the isolation strategy is applied in the domain.

**-isolation\_sense** {high | low}

Specifies the logical sense for the control signal to be isolated in the domain. The default is high.

**-location** {self | parent | fanout}

Defines where the isolation cells are placed. The self option includes the isolation cells within the module and parent includes them in the top-level domain. The fanout option defines where the isolation cells are placed in the logic hierarchy. The default is self.

### Description

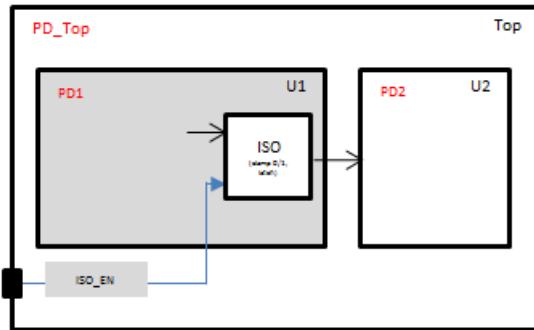
The `set_isolation_control` command identifies the isolation strategy and specifies the isolation control signals for this strategy. You can specify the logical sense for the isolation control signal in the power domain.

You must have a corresponding set of control commands for each isolation strategy defined with the `set_isolation` command, which is described in [set\\_isolation, on page 726](#).

## Examples

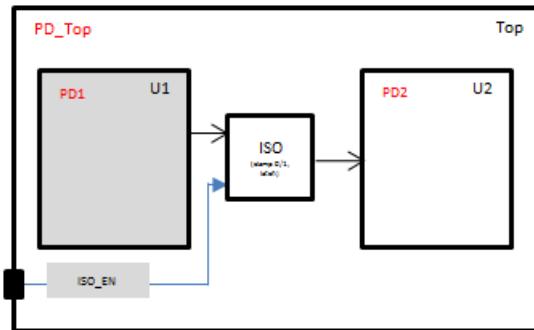
```
set_isolation_control ISO_on_outputs -domain PD1
 -isolation_signal iso_en1 -isolation_sense high
 -location self
```

The following figure shows that isolation cells are placed within module U1.



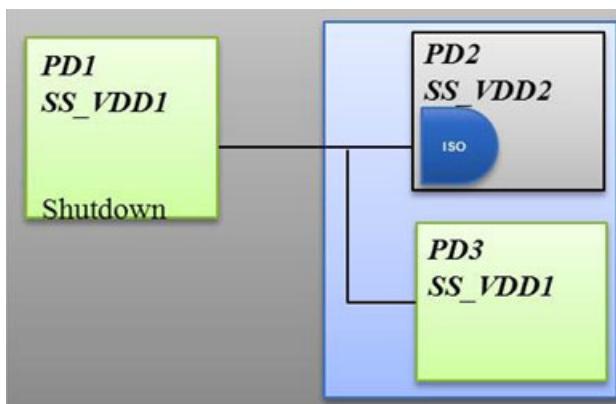
```
set_isolation_control ISO_on_outputs -domain PD1
 -isolation_signal iso_en1 -isolation_sense high
 -location parent
```

The following figure shows that isolation cells are placed in the top-level domain.



```
set_isolation PD1 -sink SS_VDD2
set_isolation_control -location fanout
```

The following figure shows that isolation cells are placed in the PD2 domain hierarchy.



## set\_retention

*Synplify Premier  
UPF*

Specifies registers in the domain that are implemented as retention registers and identifies the save and restore signals of the retention functionality for the registers.

### Syntax

```
set_retention strategyName [-domain domainName] [-elements {list}]
[-no_retention]
```

*strategyName*

Specifies the retention strategy name.

The `set_retention` command must be specified with the same `set_retention_-control` strategy name.

**-domain *domainName***

Specifies the power domain for which the retention strategy is applied.

**-elements {*list*}**

Specifies a list of design or storage elements in the power domain (registers, RAMs, sequential shifters, FSMs) to which the retention strategy is applied.

**-no\_retention**

When this option is enabled, retention capability is not added to the elements specified by the retention strategy.

### Description

The `set_retention` command specifies the registers in the domain that are implemented as retention registers and identifies the save and restore signals for the retention functionality. Only registers in the elements list are given retention capabilities. For more information, see [Specifying Retention Logic, on page 1237](#).

All isolation strategies defined with this command, except those specifying no retention, must have a corresponding set of control commands defined with the `set_retention_control` command. This command is described in [set\\_retention\\_control, on page 735](#).

## Examples

```
set_retention PD1_ret -domain PD1 -elements
{U1/out_reg U2/out_reg}

set_retention PD1_no_ret -domain PD1 -no_retention
-elements {R1 R2 R3}
```

See [Retention Results for Flip-Flops with the Same Save and Restore, on page 1238](#) and [Retention Results for RAMs, Sequential Shifters, and State Machines, on page 1240](#) for graphic examples.

## set\_retention\_control

*Synplify Premier  
UPF*

Allows you to specify the retention control signal and its logical sense for the strategy.

### Syntax

```
set_retention_control strategyName [-domain domainName]
 [-save_signal {/logicNet high | low}] [-restore_signal {/logicNet high | low}]
```

*strategyName*

Specifies the retention control strategy name.

The set\_retention\_control command must be specified with the same set\_retention strategy name.

**-domain *domainName***

Specifies the power domain for which the retention control strategy is applied.

**-save\_signal {/logicNet high | low}**

Specifies the signal (net/port/pin) that forces the register values to be saved into the shadow registers.

**-restore\_signal {/logicNet high | low}**

Specifies the signal (net/port/pin) that forces the register values to be restored.

### Description

The set\_retention\_control command allows you to specify the retention control signal and its logical sense for the strategy. For more information, see [Specifying Retention Logic, on page 1237](#).

You must have a corresponding set of control commands for each isolation strategy defined with the set\_retention command, which is described in [set\\_retention, on page 733](#).

### Example

```
set_retention_control PD1_ret -domain PD1 -save_signal
 {signal_net high} -restore_signal {signal_ret low}
```

## set\_scope

*Synplify Premier  
UPF*

Specifies the hierarchical scope for which the UPF commands are applied.

### Syntax

**set\_scope [instanceName]**

*instanceName*

Specifies the instance name of the scope for which the UPF commands are applied. If the instance path:

- is not specified, then the scope applies to the top-level current design.
- is specified with ".", then the scope applies to the working instance.
- is specified with "..", then the scope applies to the hierarchy one level up from the current design.
- is specified with "/", then the scope applies to the working instance of the design after the /.
- specifies that multiple levels of hierarchy can be traversed in a single call to the set\_scope command, then separate the cell names with a slash (/).

### Description

The set\_scope command defines the hierarchical scope for which the UPF commands are applied. For more information, see [Specifying UPF Power Domains, on page 1225](#).

### Examples

```
set_scope U1
```

## CHAPTER 9

# Netlist Editing Commands

---

Netlist editing commands must be written to a Tcl file and added to the synthesis project using Add Prototype file on the GCC & Prototyping Tools tab of the Implementation Options dialog box. For details, see [Netlist Editing, on page 1169](#) in the *User Guide*. You cannot use them at the command line.

|                                          |                                                                                        |
|------------------------------------------|----------------------------------------------------------------------------------------|
| <a href="#">connect_net</a>              | Connects a net to one or more pins or ports.                                           |
| <a href="#">copy_view</a>                | Duplicates a view of the current view.                                                 |
| <a href="#">create_cell</a>              | Creates an empty view.                                                                 |
| <a href="#">create_instance</a>          | Creates one or more instances in the current view.                                     |
| <a href="#">create_net</a>               | Creates new nets in the current view.                                                  |
| <a href="#">create_port</a>              | Creates ports of the specified direction in the current view.                          |
| <a href="#">create_process_hierarchy</a> | Creates an additional level of hierarchy based on blocks in the source RTL design file |
| <a href="#">define_current_view</a>      | Sets the current view.                                                                 |
| <a href="#">disconnect_connector</a>     | Disconnects pins and ports from their connected nets.                                  |
| <a href="#">disconnect_net</a>           | Disconnects pins and ports from the specified net.                                     |
| <a href="#">edit_netlist</a>             | Applies a Tcl editing script to a netlist file.                                        |
| <a href="#">get_net</a>                  | Returns the net name of a connector.                                                   |
| <a href="#">insert_buffer</a>            | Inserts buffers on specified ports and pins.                                           |
| <a href="#">load_database</a>            | Reads in an IP block.                                                                  |

|                                  |                                                                                      |
|----------------------------------|--------------------------------------------------------------------------------------|
| <code>load_library</code>        | Loads a primitive library.                                                           |
| <code>optimize_netlist</code>    | Removes redundant logic from the netlist.                                            |
| <code>pop_feedthroughs</code>    | Eliminates I/O pins associated with signals that pass through a design block.        |
| <code>propagate_constants</code> | Eliminates the I/O pin count associated with tying a logic input signal high or low. |
| <code>remove_buffer</code>       | Deletes the buffers specified.                                                       |
| <code>remove_instance</code>     | Deletes the specified instances from the current view.                               |
| <code>remove_net</code>          | Deletes the specified nets from the current view.                                    |
| <code>remove_port</code>         | Deletes the specified ports from the current view.                                   |
| <code>set_property</code>        | Sets properties on objects.                                                          |
| <code>swap_instance</code>       | Replaces the view of instances.                                                      |
| <code>tie_net</code>             | Ties the nets from a list of nets to a constant value.                               |
| <code>tie_pin</code>             | Ties connectors to a constant value.                                                 |

## Netlist Editing Object Definitions

All arguments used with the netlist editing commands must correspond to an existing object in the design, or to a constant. The objects can be connectors (pin or bit ports), nets, instances, views, library files, or database files.

Each of these objects can be identified with a simple name or hierarchical path name. Additionally, a name can be prefixed by a qualifier to further define the object type to the command. The qualifiers and examples of them are shown below:

| Qualifier:Object                        | Description                                                                                |
|-----------------------------------------|--------------------------------------------------------------------------------------------|
| <code>i:<i>instName</i></code>          | <code>i:</code> denotes an instance name.                                                  |
| <code>i:<i>instPath.instName</i></code> | This example is an instance in the netlist named in <code>instPath</code>                  |
| <code>t:<i>instName.pinName</i></code>  | <code>t:</code> denotes a pin name. This example specifies a pin on a particular instance. |
| <code>p:<i>portName</i></code>          | <code>p:</code> denotes a port name. This example is a top-level port.                     |

| Qualifier:Object           | Description                                                                         |
|----------------------------|-------------------------------------------------------------------------------------|
| <b>p:instName.portName</b> | This port example is a port of within the netlist of the instance <i>instName</i> . |
| <b>n:netName</b>           | <b>n:</b> denotes a net.                                                            |
| <b>n:instName.netName</b>  | This example is a net within the netlist of the instance <i>instName</i> .          |
| <b>v:viewName</b>          | <b>v:</b> denotes a view name.                                                      |

You can omit the qualifier in some commands, like single-argument commands. In multi-argument commands, the arguments must be separated by spaces.

## connect\_net

*Synplify Premier  
Netlist Editing*

Netlist editing command that connects a net to one or more pins or ports.

### Syntax

`connect_net netName [-hier] objectList`

### Arguments and Options

#### *netName*

The name of the net to be connected.

#### *objectList*

The list of pins and/or ports to be connected to the specified net.

#### -hier

Enables objects to be connected across hierarchies.

### Description

The connect\_net command connects pins and ports from *objectList* to the net specified in *netName*. The command can be used to connect objects across any hierarchy with the -hier option. Any additional ports for sub-hierarchies can be created as needed.

Objects can be single bit or multiple bits, but they must all be of the same size. Multiple-bit objects must be enclosed in curly brackets. With multi-bits objects, the connection is bit-to-bit, following the declarative order. For example, `connect_net foo[1:0] mem1.addr[6:7]` connects pin `mem1.addr[6]` to net `foo[1]` and connects pin `mem1.addr[7]` to net `foo[0]`.

When using the hier option to connect objects across hierarchies, the connection cannot be made if there is an intermediate hierarchy marked hard or fixed in the path.

An error message is generated for any of the following conditions:

- Net does not exist
- Port or pin does not exist

- Port or pin already connected to a net
- Ports and pins not in same level of hierarchy
- Intervening hierarchy marked hard or fixed
- Net, ports, and pins not the same size

## Examples

```
connect_net fast_cpm_clock clk_in u2.clk_in
connect_net {busA[15:0]} {dataA[15:0]}
```

## See Also

- [create\\_net, on page 746](#)

## copy\_view

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that duplicates a view of the current view.

### Syntax

```
copy_view currentView newView [-bbox]
```

### Arguments and Options

#### *currentView*

Name of the view to be duplicated.

#### *newView*

Name of new duplicated view.

#### **-bbox**

Treats new duplicated view as a black box.

### Description

The `copy_view` command makes a duplicate view of the current view with a name of *newView* (the `copy_view` command only copies hierarchical instances and does not copy Synplify primitives). When the optional `-bbox` argument is included, *newView* is copied as a black box with only the interconnect in place and without any internal logic. The `copy_view` command is commonly used to “uniquify” shared views.

### Examples

When devices U1 and U2 use the same view, the following sequence can be used to create a unique view for U2:

```
copy_view v:origview v:origview1
swap_instance U2 v:origview1
```

### See Also

- [\*define\\_current\\_view\*, on page 749](#)
- [\*swap\\_instance\*, on page 769](#)

## create\_cell

*Synplify Premier  
Netlist Editing*

Netlist editing command that creates an empty view.

### Syntax

`create_cell cellName`

### Arguments and Options

*cellName*

The name of the view you want to create.

### Description

The `create_cell` command creates an empty view. Add content to the view using the `create_port`, `create_instance`, and `create_net` commands. The *cellName* argument overwrites the view name contained in the .srs file to resolve conflicts with view names.

An error message is generated when the name of the view (from *cellName*) conflicts with the name of an existing view.

### Example

```
create_cell SKRAM
```

### See Also

- [create\\_instance, on page 744](#)
- [create\\_net, on page 746](#)
- [create\\_port, on page 747](#)

## create\_instance

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that creates one or more instances in the current view.

### Syntax

`create_instance instanceList referenceName`

### Arguments and Options

*instanceList*

A list of instance names separated by spaces.

*referenceName*

The type of view

### Description

The `create_instance` command creates one or more instances in the current view. Hierarchical names allow instance creation below the current level. In the command line, *instanceList* specifies a list of instance names separated by spaces, and *referenceName* specifies the type of view. The view type is either `library.cell` for a library primitive, or simply `view` for a netlist created from an `.srs` file with the `create_cell` command.

Note that an error message is generated when an instance name (from *instanceList*) conflicts with the name of an existing instance or when *referenceName* does not exist.

### Examples

Simple example:

```
create_instance {i:bb_inst} {FD}
```

The following example illustrates creating an instance of a Xilinx FD primitive from a technology library:

```
load_library $LIB/xilinx/xilinx.syn
create_instance {i:ff_inst} {FD}
create_port {p:out1} -direction out
create_port {p:in1} -direction in
create_net {n:in1} {n:out1}
connect_net {n:clk} {t:ff_inst.C}
connect_net {n:in1} {p:in1} {t:ff_inst.D}
connect_net {n:out1} {p:out1} {t:ff_inst.Q}
```

## create\_net

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that creates new nets in the current view.

### Syntax

```
create_net netList
```

### Arguments and Options

*netList*

A space-separated list of nets to be added to the netlist.

### Description

The `create_net` command creates new nets in the current view. Multiple net entries in the *netList* are separated by spaces, and both single- and multiple-bit nets can be specified (multiple-bit nets must be enclosed in curly brackets). Hierarchical names allow the creation of nets below the current hierarchy level.

Note that an error message is generated when a net name (from *netList*) conflicts with the name of an existing net.

### Examples

```
create_net fast_cpm_clock
create_net {mem1.addr[7:0]}
```

### See Also

- [create\\_port, on page 747](#)

## create\_port

*Synplify Premier  
Netlist Editing*

Netlist editing command that creates ports of the specified direction in the current view.

### Syntax

```
create_port portList [-direction {in|out|inout}]
```

### Arguments and Options

*portList*

A space-separated list of ports to be added to the netlist.

**-direction {in|out|inout}**

The direction of the port to be created.

### Description

The `create_port` command creates ports of the specified direction in the current view. Multiple port entries in the *portList* are separated by spaces, and both single- and multiple-bit ports can be specified (multiple-bit ports must be enclosed in curly brackets). When no direction is specified, input ports are created.

Note that an error message is generated when a port name (from *portList*) conflicts with the name of an existing port.

### Examples

```
create_port fast_cpm_clock -direction out
create_port {dataA[7:0]} {dataB[3:0]} -direction inout
```

### See Also

- [create\\_net, on page 746](#)
- [create\\_instance, on page 744](#)

## create\_process\_hierarchy

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that creates an additional level of hierarchy based on blocks in the source RTL design file.

### Syntax

`create_process_hierarchy`

### Arguments and Options

None

### Description

The `create_process_hierarchy` command creates an additional level of hierarchy based on the always (Verilog) or process (VHDL) blocks in the source RTL design file. This feature allows large modules or entities that cannot be partitioned into a single FPGA to be broken up into a set of logical submodules that can be more easily partitioned among the available FPGAs.

When including the `create_process_hierarchy` command in the netlist-editing script, be sure to uncheck the Create always/process level hierarchy check box on the GCC & Prototyping Tools tab. Disabling this check box allows control of when the effects of the `create_process_hierarchy` command occur relative to netlist optimizations. For additional information, see [Specifying Netlist Editing Commands, on page 1170](#) in the *Synopsys FPGA Synthesis User Guide*.

### Examples

`create_process_hierarchy`

### See Also

- [optimize\\_netlist, on page 760](#)
- [pop\\_feedthroughs, on page 761](#)
- [propagate\\_constants, on page 762](#)

## define\_current\_view

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that sets the current view.

### Syntax

```
define_current_view v:viewName
```

### Arguments and Options

*viewName*

Name of the view to be set.

### Description

The `define_current_view` command sets the current view to *viewName*. Because a design can be incrementally unqualified during editing operations, the actual machine-generated view name for a multiply-instantiated module may not be known. When specifying *viewName*, make sure that the entry is correct (no indication is given for a misspelled or non-existent view name).

### Examples

```
define_current_view v:scenic2
```

### See Also

- [copy\\_view, on page 742](#)

# disconnect\_connector

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that disconnects pins and ports from their connected nets.

## Syntax

`disconnect_connector objectList`

## Arguments and Options

*objectList*

A list of the ports or pins to be disconnected.

## Description

The `disconnect_connector` command disconnects pins and ports in *objectList* from the nets where they are connected. Objects can be single bit or multiple bits (multiple-bit nets must be enclosed in curly brackets). The command can be used to disconnect objects that are in different hierarchies.

Note that an error message is generated for any of the following conditions:

- Port or pin does not exist
- Port or pin already disconnected from net

## Examples

The following example disconnects pins `mem2.addr[1:0]` and `mem1.addr[7:0]` from their connected nets.

```
disconnect_connector {mem2.addr[1:0]} {mem1.addr[7:0]}
```

## See Also

- [disconnect\\_net, on page 751](#)

## disconnect\_net

*Synplify Premier  
Netlist Editing*

Netlist editing command that disconnects pins and ports from the specified net.

### Syntax

**disconnect\_net** *netName objectList | -all*

### Arguments and Options

*netName*

The name of the net.

*objectList*

The list of individual pins and ports to be disconnected from *netName*.

**-all**

Disconnects all pins and ports from *netName*.

### Description

The `disconnect_net` command disconnects pins and ports in *objectList* from the net specified by *netName*. The command can be used to disconnect objects that are in different hierarchies. Objects can be single bit or multiple bits, but they must all be of the same size. With multi-bits objects, the disconnection is bit-to-bit, following the declarative order. For example, `disconnect_net foo[1:0] mem1.addr[6:7]` disconnects pin `mem1.addr[6]` from net `foo[1]` and disconnects pin `mem1.addr[7]` from net `foo[0]`.

An error message can be generated for any of the following conditions:

- Net does not exist
- Port or pin does not exist
- Port or pin already disconnected from net
- Ports and pins not in same level of hierarchy
- Net, ports, and pins not the same size

## Examples

```
disconnect_net fast_cpm_clock clk_in u2.clk_in
disconnect_net {busA[15:0]} {dataA[15:0]}
```

## See Also

- *disconnect\_connector*, on page 750

## edit\_netlist

*Synplify Premier  
Netlist Editing*

Applies a Tcl editing script to a netlist file outside of a project.

### Syntax

```
edit_netlist -netlist pathToOrigNetlist -out pathToModNetlist -tcl filename
[-log filename]
```

### Arguments and Options

**-netlist** *pathToOrigNetlist*

Specifies the path to the original netlist file to be edited.

**-out** *pathToModNetlist*

Specifies the path to the updated (edited) netlist. The original netlist file is preserved.

**-tcl** *filename*

The name of the Tcl file containing the netlist commands to be applied to the original netlist file. For a list of the available netlist editing commands, see [Netlist Editing, on page 1169](#).

**-log** *filename*

Optional argument that redirects the results of the netlist edit to the specified log file. Results are written to the *modNetlistName\_edit\_netlist.log* file by default.

### Description

The `edit_netlist` command applies netlist-editing commands from the specified Tcl file to the named .srs netlist file outside of a project. The results are written to a separate file to preserve the original netlist.

### Examples

```
edit_netlist -netlist ./proto2/dsgn1A.srs
-out ./proto2/dsgn1ver1.srs
-tcl ./dsgn1.tcl -log ./proto2/netedit.log
```

## get\_net

*Synplify Premier  
Netlist Editing*

Returns the net name of a connector during netlist editing.

### Syntax

`get_net connectorName`

### Arguments and Options

*connectorName*

The name of the connector.

### Description

The `get_net` command returns the net name of a connector. The net must be a single-bit net, and the connector must be an instance pin or a netlist bit port. An error message is generated if the connector is not connected to any net.

### Example

The following command assigns the net name of pin I1 of instance inst1 to the variable `net`.

```
set net [get_net t:inst1.I1]
```

## insert\_buffer

*Synplify Premier  
Netlist Editing*

Netlist editing command that inserts buffers on specified ports and pins.

### Syntax

`insert_buffer [-inverter_pair] objectList bufferType`

### Arguments and Options

#### *objectList*

List of the ports and pins to buffer.

#### *bufferType*

The type of buffer to insert. Common buffer types include:

- IBUF
- OBUF
- BUFG
- BUF

#### **-inverter\_pair**

Specifies two inverters to buffer the ports or pins.

### Description

Inserts specified buffer types on the ports and pins in *objectList*. A buffer is defined as a view with either a single input or one or more outputs that function as either input or !input. Use the **-inverter\_pair** option to insert two inverters. A target technology library must be loaded before specifying any `insert_buffer` commands.

An error message is generated for any of the following conditions:

- Each pin or port is not connected to a net
- Bidirectional pin/port *bufferType* does not exist
- *bufferType* is not a buffer, as specified above
- Without **-inverter\_pair**, *bufferType* does not have non-inverting outputs

- With `-inverter_pair`, `bufferType` does not have inverting outputs

## Example

```
load_library $LIB/xilinx/xilinx.syn #loads target technology lib
insert_buffer {ua.q[7]} BUF #inserts a buffer
insert_buffer -inverter_pair {ua.q[3]} INV #inserts an
inverter pair
```

## See Also

- [\*load\\_library\*, on page 759](#)

# load\_database

*Synplify Premier  
Netlist Editing*

During netlist editing, reads in an IP block.

## Syntax

```
load_database netlist.srs [-base_name string | -auto_rename]
```

## Arguments and Options

### *netlist.srs*

The name of the .srs netlist file containing the IP block.

### -base\_name

An optional argument that prepends the specified base-name string and an underscore to each instance in the IP block.

### -auto\_rename

An optional argument that automatically renames modules in an IP block to resolve name conflicts with the remainder of the design.

## Description

The load\_database command reads an IP block into memory. The database is parallel to the top-level design and remains in memory for the current session. The Tcl find and collection commands cannot be used on the parallel database.

In the database, IP blocks can contain instance names that conflict with instance names in the core design. For example, an instance named state\_1 can appear independently in both an IP block and in the core design. Use the -basename or -auto\_rename argument to resolve this type of naming conflict.

- **-base\_name** - prepends the specified base-name string and an underscore to each instance in the IP block. For example, if the base name is bist and the IP block contains two modules, one module named aaa and the other named bbb, the modules are renamed bist\_aaa and bist\_bbb.
- **-auto\_rename** - automatically renames modules in an IP block to resolve name conflicts with the remainder of the design. Instance names are not

changed, and module names can be retrieved by either a `get_prop` or `get_attribute` command using the syntax:

`get_prop -prop view {instanceName}`

`get_attribute view {i:instanceName}`

## Examples

```
load_database ./tmp/add_select.srl -base_name nle
```

## See Also

- [get\\_prop, on page 247](#)

# load\_library

*Synplify Premier  
Netlist Editing*

Netlist editing command that loads a primitive library.

## Syntax

`load_library primitiveLibrary`

## Arguments and Options

*primitiveLibrary*

The location of the primitive library. For Intel primitives, the library is located in *installDirectory/lib/altera/altera.syn*, and for Xilinx primitives, the library is located in *installDirectory/lib/xilinx/xilinx.syn*.

## Description

The `load_library` command loads a primitive library for any netlist-editing commands that require the addition or replacement of a vendor primitive within the netlist.

## Examples

```
load_library $LIB/xilinx/xilinx.syn
```

```
load_library $LIB/altera/altera.syn
```

## See Also

- [insert\\_buffer, on page 755](#)

## optimize\_netlist

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that removes redundant logic from the netlist.

### Syntax

`optimize_netlist`

### Arguments and Options

None

### Description

The `optimize_netlist` command removes redundant logic from the netlist. The command operates on the entire design.

When including the `optimize_netlist` command in the netlist-editing script, be sure to uncheck the Optimize Netlist check box on the GCC & Prototyping Tools tab of the Implementation Options dialog box. Disabling this check box allows control of when the effects of the `optimize_netlist` command occur relative to netlist optimizations. For additional information, see [Specifying Netlist Editing Commands, on page 1170](#) in the *Synopsys FPGA Synthesis User Guide*.

### Examples

`optimize_netlist`

### See Also

- [\*create\\_process\\_hierarchy\*, on page 748](#)
- [\*pop\\_feedthroughs\*, on page 761](#)
- [\*propagate\\_constants\*, on page 762](#)

# pop\_feedthroughs

*Synplify Premier  
Netlist Editing*

Netlist editing command that eliminates I/O pins associated with signals that pass through a design block.

## Syntax

**pop\_feedthroughs**

## Arguments and Options

None

## Description

The `pop_feedthroughs` command eliminates I/O pins associated with signals that pass through a design block without driving any logic or that are unnecessarily routed through a block.

When including the `pop_feedthroughs` command in the netlist-editing script, be sure to uncheck the Feedthrough Optimization check box on the GCC & Prototyping Tools tab. Disabling this check box allows control of when the effects of the `pop_feedthroughs` command occur relative to netlist optimizations. For additional information, see [Specifying Netlist Editing Commands, on page 1170](#) in the *Synopsys FPGA Synthesis User Guide*.

## Examples

`pop_feedthroughs`

## See Also

- [\*create\\_process\\_hierarchy\*, on page 748](#)
- [\*optimize\\_netlist\*, on page 760](#)
- [\*propagate\\_constants\*, on page 762](#)

# propagate\_constants

*Synplify Premier  
Netlist Editing*

Netlist editing command that eliminates the I/O pin count associated with tying a logic input signal high or low.

## Syntax

**propagate\_constants [-effort high]**

## Arguments and Options

### **-effort high**

Sets the optimization level to include flip-flops and muxes.

## Description

The propagate\_constants command eliminates the I/O pin count associated with tying a logic input signal high or low by making the connection internal to the instance rather than through the instance interface. When the -effort high option is included, DFFRE and DFFSE flip-flops are optimized away when their input pins are tied to constants which would result in a constant output (latches are not optimized). Similarly, 2-to-1 muxes are optimized away when both inputs are tied to the same constant value. Constants are not propagated across keep buffers.

When including the propagate\_constants command in the netlist-editing script, be sure to uncheck the Constant Propagation check box on the GCC & Prototyping Tools tab. Disabling this check box allows control of when the effects of the propagate\_constants command occur relative to netlist optimizations. For additional information, see [Specifying Netlist Editing Commands, on page 1170](#) in the *Synopsys FPGA Synthesis User Guide*.

## Examples

```
propagate_constants -effort high
```

## See Also

- *create\_process\_hierarchy*, on page 748
- *optimize\_netlist*, on page 760
- *pop\_feedthroughs*, on page 761

## remove\_buffer

*Synplify Premier  
Netlist Editing*

Netlist editing command that deletes the specified buffers.

### Syntax

`remove_buffer instanceList`

### Arguments and Options

*instanceList*

A list of the buffers to be deleted.

### Description

The `remove_buffer` command deletes the buffers specified in *instanceList*. A buffer is defined by the `insert_buffer` command. To remove an inverter pair, the command must list both instances.

An error message is generated for any of the following conditions:

- Each instance is not a buffer as defined in the `insert_buffer` command
- The input pin of an instance is not connected to a net
- No output pins of an instance are connected to a net
- More than one output pin of an instance is connected to a net
- Inverting buffers are not cascaded with single fanout between them

### Examples

`remove_buffer ua.q[7] ux.ub.d[4]`

### See Also

- [remove\\_instance, on page 765](#)
- [remove\\_net, on page 766](#)
- [remove\\_port, on page 767](#)

# remove\_instance

*Synplify Premier*  
*Netlist Editing*

Netlist editing command that deletes the specified instances from the current view.

## Syntax

```
remove_instance instanceList | -all
```

## Arguments and Options

*instanceList*

List of the instances to be deleted.

-all

Delete all instances from the current view.

## Description

The `remove_instance` command deletes the specified instances from the current view. Hierarchical names allow deletion of instances below the current level. In the command line, *instanceList* specifies a list of instance names, separated by spaces, to be deleted. Instances can be single bit or multiple bits (multiple-bit instances must be enclosed in curly brackets). The `-all` argument deletes all instances from the current view.

An error message can be generated when an instance name (from *instanceList*) does not exist.

## Examples

```
remove_instance {inst_A.out[0:3]} {p:out[0:3]}
```

## See Also

- [remove\\_buffer, on page 764](#)
- [remove\\_net, on page 766](#)
- [remove\\_port, on page 767](#)

## remove\_net

*Synplify Premier  
Netlist Editing*

Netlist editing command that deletes the specified nets from the current view.

### Syntax

```
remove_net netList | -all
```

### Arguments and Options

*netList*

List of the nets to be deleted.

-all

Delete all nets from the current view.

### Description

The remove\_net command deletes the specified nets from the current view. Nets can be single bit or multiple bits (multiple-bit nets must be enclosed in curly brackets). Hierarchical names allow deletion below the current level.

An error message is generated when a net name (from *netList*) does not exist.

### Examples

```
remove_net fast_cpm_clock
remove_net {busA[15:0]}
```

### See Also

- [remove\\_buffer, on page 764](#)
- [remove\\_instance, on page 765](#)
- [remove\\_port, on page 767](#)

## remove\_port

*Synplify Premier  
Netlist Editing*

Netlist editing command that deletes the specified ports from the current view.

### Syntax

`remove_port portList | -all`

### Arguments and Options

*portList*

List of the ports to be deleted.

**-all**

Delete all ports from the current view.

### Description

The `remove_port` command deletes the specified ports from the current view. Multiple port entries in the *portList* are separated by spaces, and both single- and multiple-bit ports can be specified (multiple-bit ports must be enclosed in curly brackets).

Note that an error message is generated when a port name (from *portList*) does not exist.

### Examples

```
remove_port fast_cpm_clock {dataA[7:0]}
```

### See Also

- [remove\\_buffer, on page 764](#)
- [remove\\_instance, on page 765](#)
- [remove\\_net, on page 766](#)

## set\_property

*Synplify Premier  
Netlist Editing*

Netlist editing command that sets properties on objects.

### Syntax

```
set_property -type {integer| string} object property value
```

### Arguments and Options

#### -type {integer| string}

The property type; the default is string.

#### object

The name of the object. Object names are prefixed with a letter to identify the object type (v: view, i: instance, p: port, t: pin, n: net). If the object prefix is omitted, instance is assumed.

#### property

The name of the property assigned to the object.

#### value

The value of the assigned property.

### Description

The set\_property command sets properties on objects during netlist editing. In the command syntax, the property type is either integer or string (the default is string).

### Examples

```
set_property -type string p:portA direction inout
```

## **swap\_instance**

## *Synplify Premier Netlist Editing*

Replaces the view of the specified instances during netlist editing.

## Syntax

**swap\_instance** *instanceList viewName*

## Arguments and Options

### **sinstanceList**

A list of the instances to be replaced.

*viewName*

The new view for the swapped instances.

## Description

The `swap_instance` command replaces the view of the instances specified in `instanceList` with the view specified by `viewName`. The interface (pin count, pin names, pin directions) of the new view must be identical to the original view. The swapped instances retain their original properties.

### **Limitation**

The `swap_instance` command cannot be used for cell primitives without a view. As a workaround, use the following netlist editing commands instead:

- connect\_net
  - disconnect\_net

## Examples

```
swap instance U2 v:origview1
```

#### **See Also**

- *copy\_view*, on page 742

## tie\_net

*Synplify Premier  
Netlist Editing*

Ties the nets from the list of nets to a constant value during netlist editing.

### Syntax

```
tie_net netsList "value"
```

### Arguments and Options

#### *netsList*

List of nets; the *netList* argument can have one of the forms:

*netName* or **n:netName** - denotes a net

*instName.netName* or **n:instName.netName** - denotes a net within the netlist of an instance

#### *"value"*

The constant value; the *value* argument can be any of the following:

**0** - changes the net driver to GND

**1** - changes the net driver to VCC

**-** (dash) - keeps the net driver unchanged

**x** or **X** - disconnects the net driver

Note that it is not necessary to quote *value* and that no spaces are allowed within the value argument string.

### Description

The tie\_net command ties the nets from the list of nets to a constant value. The arguments can be scalar or vector names, but the widths of all arguments must match. The command assigns the same value to all connectors in the list. If the nets are vectors, the assignment is done in bit-wise manner.

## Examples

The command

```
tie_net {n:KONNECT6[0]} "x"
```

disconnects the net driver from bit 0 of KONNECT6.

The commands

```
tie_net {n:KONNECT6[0:3]} "x-01"
tie_net {n:KONNECT7[5:0]} "1-x100"
```

are equivalent to the following individual tie\_net commands:

```
tie_net {n:KONNECT6[0]} "x" (disconnect bit 0)
tie_net {n:KONNECT6[1]} "--" (leave bit 1 unchanged)
tie_net {n:KONNECT6[2]} "0" (set bit 2 to GND)
tie_net {n:KONNECT6[3]} "1" (set bit 3 to VCC)

tie_net {n:KONNECT7[5]} "1" (set bit 5 to VCC)
tie_net {n:KONNECT7[4]} "--" (leave bit 4 unchanged)
tie_net {n:KONNECT7[3]} "x" (disconnect bit 3)
tie_net {n:KONNECT7[2]} "1" (set bit 2 to VCC)
tie_net {n:KONNECT7[1]} "0" (set bit 1 to GND)
tie_net {n:KONNECT7[0]} "0" (set bit 0 to GND)
```

## See Also

[tie\\_pin, on page 772](#)

## tie\_pin

*Synplify Premier  
Netlist Editing*

Ties connectors to a constant value during netlist editing.

### Syntax

```
tie_pin connectorList "value"
```

### Arguments and Options

#### connectorList

List of connectors; the *connectorList* argument can have one of the following forms:

*instName.pinName* or *t:instName.pinName* - denotes an instance pin

**p:portName** - denotes a top-level output port

**p:instName.portName** - denotes an output port of the internal netlist of an instance

#### "value"

The constant value; the *value* argument can be any of the following:

**0** - changes the net driver to GND

**1** - changes the net driver to VCC

**-** (dash) - keeps the net driver unchanged

**x** or **X** - disconnects the net driver

Note that it is not necessary to quote *value* and that no spaces are allowed within the value argument string.

### Description

The tie\_pin command ties the connectors from the list of connectors to a constant value. The arguments can be scalar or vector names, but the widths of all arguments must match. The command assigns the same value to all connectors in the list. If the connectors are vectors, the assignment is done in bit-wise manner.

## Examples

The command

```
tie_pin {t:U6.ALARM_HRS[0]} "1"
```

ties pin ALARM\_HRS[0] of instance U6 to constant value 1 (VCC).

The commands

```
tie_pin {t:U6.ALARM_HRS[0:3]} "x-01"
tie_pin {t:U6.TIME_MINS[5:0]} "1-x100"
```

are equivalent to the following individual tie\_pin commands:

```
tie_pin {t:U6.ALARM_HRS[0]} "x" (disconnect pin 0)
tie_pin {t:U6.ALARM_HRS[1]} "--" (keep current connection of pin 1)
tie_pin {t:U6.ALARM_HRS[2]} "0" (tie pin 2 to constant value 0)
tie_pin {t:U6.ALARM_HRS[3]} "1" (tie pin 3 to constant value 1)

tie_pin {t:U6.TIME_MINS[5]} "1" (tie pin 5 to constant value 1)
tie_pin {t:U6.TIME_MINS[4]} "--" (keep current connection of pin 4)
tie_pin {t:U6.TIME_MINS[3]} "x" (disconnect pin 3)
tie_pin {t:U6.TIME_MINS[2]} "1" (tie pin 2 to constant value 1)
tie_pin {t:U6.TIME_MINS[1]} "0" (tie pin 1 to constant value 0)
tie_pin {t:U6.TIME_MINS[0]} "0" (tie pin 0 to constant value 0)
```

## See Also

[tie\\_net, on page 770](#)



# Index

---

## Symbols

\_ALLOWNESTEDBLOCKCOMMENTSTA  
    RT\_\_ directive [476](#)  
\_SEARCHFILENAMEONLY\_  
    directive [474](#)  
! character, find command [235](#)  
? wildcard  
    Timing Analyzer [539](#)  
.est file [666](#)  
.srr file  
    See log file  
.srs file  
    See srs file

## Numerics

64-bit mapping [449](#)

## A

aborting a synthesis run [505](#)  
About this program command [603](#)  
add files  
    -include tcl argument [26](#)  
Add Implementation command [424](#)  
Add P&R Implementation command [626](#)  
Add Place & Route Options File  
    command [628](#)  
Add Place and Route Job command [626](#)  
Add Source File command [423](#)  
add\_file Tcl command [25](#)  
add\_folder Tcl command [30](#)  
add\_to\_collection command [282](#)  
Additional Products command [602](#)  
Adjust Pin View command [662](#)  
advanced\_uram\_features\_on [152](#)  
Align Regions command [661](#)

annotated properties for analyst  
    object properties for filtering [230](#)  
append\_to\_collection command [284](#)  
archive utility  
    \_SEARCHFILENAMEONLY\_  
        directive [474](#)  
    copy tcl command [111](#)  
    unarchive tcl command [111](#)  
area estimation  
    design modules [666](#)  
    Design Planner regions [667](#)  
    warning settings [669](#)  
Arrange VHDL files command [500](#)  
assign\_to\_region Tcl command [39](#)  
asynchronous clock report  
    generation option [531](#)  
Attributes panel, SCOPE [313](#)  
auto constraints  
    Maximize option (Constraints tab) [452](#)

## B

Back command [417](#)  
batch mode [210](#)  
Build Project command [396](#)  
bus bundling [591](#)  
bus\_dimension\_separator\_style  
    command [390](#)  
bus\_naming\_style command [390](#)  
buses  
    compressed display [591](#)  
    enabling bit range display [590](#)  
    hiding in flattened Technology  
        views [591](#)  
By any transition command [418](#)  
By input transitions command [418](#)  
By output transitions command [418](#)

**C**

c\_diff command (collections) 305  
 c\_intersect command (collections) 305  
 c\_print command (collections) 305  
 c\_sub command (collections) 305  
 c\_symdiff command (collections) 305  
 c\_symdiff command, examples 245  
 c\_union command (collections) 305  
 camera mouse pointer 396  
 case sensitivity, Tcl find command 224  
 CDPL 154  
 cell interior display,  
     enabling/disabling 591  
 Change File command 423  
 Change Implementation Name  
     command 615  
 check\_fdc\_query command 40  
 check\_fdc\_query Tcl command 40  
 Clear Parameters command 607  
 Clearbox  
     set\_option parameters 180  
     setting options 180  
 clock alias 535  
 clock as object 535  
 clock groups, SCOPE 297  
 clock paths, ignoring 328  
 Clocks panel, SCOPE 296  
 Close command 396  
 Close Project command 397  
 Collapse All command 641  
 Collection Commands  
     get\_prop 247  
 collection commands  
     c\_diff 241  
     c\_intersect 242  
     c\_list 243  
     c\_print 243  
     c\_symdiff 245  
     c\_union 245  
     SCOPE 305  
 collections  
     Synopsys standard commands 282

Collections panel, SCOPE 304  
 commands  
     Add Place & Route Job 626  
     Hierarchy Browser 641  
     menu  
         See individual command entries  
     set\_modules (Tcl) 247  
     Tcl  
         See Tcl commands  
     Tcl collection 240  
     Tcl command equivalents 16  
     Tcl expand 237  
     Tcl find 221  
 Comment Code command 403  
 Common Distributed Processing  
     Library 154  
 Compile Only command 498  
 Compile Physical Hierarchy  
     command 499  
 compile point constraints  
     editing 643  
 Compile Points panel, SCOPE 316  
 compile\_strategy 150, 154  
 compiler directives  
     \_\_ALLOWNESTEDBLOCKCOMMENTS  
         TART\_\_ 476  
     \_\_SYN\_COMPATIBLE\_INCLUDEPATH\_\_  
         - 477  
     \_\_BETA\_FEATURES\_ON\_\_ 473  
     \_\_SEARCHFILENAMEONLY\_\_ 474  
     IGNORE\_VERILOG\_BLACKBOX\_GUTS  
         472  
     UI option 464  
     Verilog 470  
 Configure External Programs  
     command 597  
 Configure Mapper Parallel Job  
     command 568  
 Configure Verilog Compiler  
     command 568  
 Configure VHDL Compiler  
     command 568  
 Configure Watch command 607  
 connectivity, enabling bit range  
     display 590  
 constraint checker  
     check\_fdc\_query command 40

constraint file  
  define\_compile\_point 392  
  define\_current\_design 393  
  syn\_connect 207  
  syn\_create\_err\_net 208

constraint files  
  editing compile point files 643  
  SCOPE spreadsheet 294  
  translating 504

constraint\_file Tcl command 45

constraints  
  automatic. *See* auto constraints  
  check constraints 500  
  FPGA timing 342  
  translating Intel constraints 124

Constraints panel  
  Implementation Options dialog box 450

context-sensitive popup menus  
  *See* popup menus

Continue on Error  
  Configure Compile Point Process 571

Continue on Error compile point option 571

Copy command 402

Copy File command 614

Copy Implementation command 615

copy\_collection command 285

copying image  
  Create Image command 396

core cells  
  displaying 689

Create Image command 396

Create Place & Route Options file dialog box 629

Create Sub-project (Design Block) command 631, 632

create\_clock timing constraint 343

create\_generated\_clock timing constraint 345

create\_power\_domain Tcl command 711

create\_region Tcl command 48

critical paths  
  creating new schematics 543  
  custom timing reports 530  
  finding 549

Timing Report panel, Implementation Options dialog box 456

cross probing 688

custom folders  
  project\_folder Tcl command 123

Customize command 568

customizing  
  project files 575

Cut command 402

**D**

define\_compile\_point  
  Tcl 392

define\_current\_design  
  Tcl 393

defining I/O standards 314

delay paths  
  POS 334

Delay Paths panel, SCOPE 310

Delete all bookmarks command 403

Delete Region Assignments command 661

Delete Region command 661

Delete Selected Assignments command 661

Design Block Properties command (hierarchical project management) 634

design parameters (Verilog)  
  extracting 470

design plan view file (.srp) 543

Design Planner  
  popup menus 673  
  Preferences dialog box 668  
  Properties command 663

Design Planner tools menu  
  Preferences command 668  
  Show Replicated Assignments command 668

Design Planning panel, Implementation Options dialog box 665

DesignWare options  
  VHDL panel 461, 467

Device panel

- Conservative Register Optimization [170](#)
- Implementation Options dialog box [445](#)
- dh\_module\_sources Tcl command [63](#)
- dialog boxes
- Enhanced Instance Display [689](#)
  - Find Object (Physical Analyst) [699](#)
  - Implementation Options [444](#)
  - Physical Analyst Properties [697](#)
  - Region Properties [663](#)
  - Replicated Assignments [668](#)
  - Resolve Selection [703](#)
- directives
- `_SEARCHFILENAMEONLY_` [474](#)
  - beta features [473](#)
  - ignore syntax check [472](#)
  - `IGNORE VERILOG_BLACKBOX_GUTS` [472](#)
  - specifying for the compiler (Verilog) [470](#)
- disabling sequential optimizations [168](#)
- display settings
- Project view [575](#)
- Dissolve Instances command [551](#)
- Dissolve to Gates command [551](#)
- dissolving instances [551](#)
- distributed processing [154](#)
- duplicate modules (Verilog)
- Tcl option [152, 157](#)
- E**
- Edit Attributes command [642](#)
- Edit Compile Point Constraints command [643](#)
- Edit menu [402](#)
- Advanced submenu [403](#)
- Edit Module Constraints command [643](#)
- Edit Regions command [661](#)
- Edit Run Configuration command [441](#)
- Editor Options command [568](#)
- Enable Slack Margin [534](#)
- encoding
- enumeration, default (VHDL) [460](#)
  - state machine
    - displaying [612](#)
- encryptIP script
- command-line arguments [68](#)
  - syntax [68](#)
- encryptP1735 script [71](#)
- command-line arguments [71](#)
  - public keys repository file [72](#)
  - syntax [71](#)
  - use models [74](#)
- enhanced display mode [689](#)
- Enhanced Instance Display dialog box [689](#)
- enumeration encoding, default (VHDL) [460](#)
- environment variables
- accessing, `get_env` Tcl command [87](#)
- est file [666](#)
- Estimate All Regions command [667](#)
- Estimate Area command [499, 666](#)
- Estimate Regions command [667](#)
- estimation
- area [666, 667](#)
- estimation (setting in Design Planner) area
- warning settings [669](#)
- estimation file, area (.est) [666](#)
- examples
- Tcl find command syntax [225](#)
- Exit command [397](#)
- Expand command
- current level [545](#)
  - hierarchical [544](#)
- Expand Inwards command [544](#)
- Expand Paths command
- current level [545](#)
  - hierarchical [544](#)
- Expand to Register/Port command
- current level [545](#)
  - hierarchical [544](#)
- Expanded Pin View command [662](#)
- expanding
- paths between schematic objects [544](#)
- export project Tcl command [81](#)
- Extract Parameters [470](#)

**F**

false paths  
  architectural 328  
  clocks as from/to points 337  
  code-introduced 328  
  defined 328  
  POS 334

FDC  
  create\_clock constraint 343  
  create\_generated\_clock 345  
  reset\_path 349  
  set\_clock\_groups 353  
  set\_clock\_latency 360  
  set\_clock\_route\_delay 362  
  set\_clock\_uncertainty 363  
  set\_datapathonly\_delay 365  
  set\_false\_path 368  
  set\_input\_delay 371  
  set\_max\_delay 374  
  set\_multicycle\_path 380  
  set\_output\_delay 384  
  set\_reg\_input\_delay 387  
  set\_reg\_output\_delay 388  
  standard collection commands 282

File menu  
  Recent Projects submenu 397

File Options command 622

files  
  .est 666  
  .srp 543  
  .ta See also timing report file 532  
  adding to project 25, 426  
  area estimation (.est) 666  
  constraint 45  
  copying 614, 615  
  design plan view (.srp) 543  
  include 26  
  log. See log file  
  opening recent project 397  
  organization into folders 575  
  partitioned RTL view (.srp) 543  
  physical synthesis netlist (.srp) 543  
  project 121  
  removing from project 423  
  replacing in project 429  
  srs See srs file  
  stand-alone timing report (.ta) 519  
  temporary 552

timing report. *See also* timing report file 532

Filter Schematic command 546  
  popup menu 607

filtering  
  critical paths 549  
  FSM states and transitions 418  
  paths from pins or ports 550  
  selected objects 546  
  timing reports 531

filtering design objects  
  Physical Analyst 701

Find again command 403

Find command  
  HDL Analyst 407  
  Physical Analyst view 686  
  Text Editor 402

find command  
  batch mode 103  
  filter properties 230

Find command (Physical Analyst) 699

Find Object dialog box  
  filtering design objects 701

Find Object dialog box (Physical Analyst) 699

finding  
  critical paths 549

finding objects (Physical Analyst) 699

Flatten Current Schematic command  
  filtered schematic 547  
  unfiltered schematic 546

Flattened Critical Path command 544

flattened schematic, creating 543

Flattened to Gates View command 543

Flattened View command 543

flattening  
  instances 551  
  schematics 546

Floating License Usage command 602

Floorplan Editor  
  selecting objects 692

floorplan objects 688

Floorplanned View command 543

folders

adding to project 30  
folders for project files 575  
`foreach_in_collection` command 286  
Forward command 417  
FPGA Implementation Tools command 601  
FPGA timing constraints 342  
from points  
  clocks 336  
  multiple 331  
object search order (Timing Analyzer) 535  
objects 330  
timing analyzer 535  
FSM Explorer command 499  
FSM Table command 418  
FSM Viewer  
  popup menu 610  
  popup menu commands 610  
FSMs  
  optimizing with FSM Compiler 174  
Full View command 416

## G

gated-clock conversion 480, 481  
GCC & Prototyping Tools panel 481  
`generate_instance_constraints` Tcl command 84  
Generated Clocks panel, SCOPE 301  
generated-clock optimization 480, 481  
`get_env` Tcl command 87  
`get_object_name` command 288  
`get_option` Tcl command 87  
`get_prop`  
  design `get_prop` 60  
`get_prop` TCL command 247  
Go to SolvNet command 601  
Goto command 403  
Goto Net Driver command  
  current level 545  
  hierarchical 545

## H

HDL Analyst  
  Find command 407  
  Visual Properties 417  
HDL Analyst menu 542, 691  
  Current Level submenu 545  
  Hierarchical submenu 544  
  Physical Analyst submenu 691  
  RTL submenu 543  
  Select All Schematic submenu 553  
  Select All Sheet submenu 553  
  Technology submenu 543  
HDL Analyst Options command 569  
HDL Analyst tool  
  displaying timing information 549  
HDL parameter overrides 89  
`hdl_define` Tcl command 88  
`hdl_param` Tcl command 89  
Help command 602  
Help menu 602  
Hide command  
  View menu, Rats Nest submenu 662  
Hide Instances command 550  
hiding instances 550  
Hierarchical Critical Path command 543  
Hierarchical View command 543  
hierarchy  
  flattening 546  
Hierarchy Browser  
  commands 641  
  popup menu 641  
  refreshing 641  
hierarchy browser  
  enabling/disabling display 591  
hierarchy separator 389  
High Reliability panel  
  Implementation Options dialog box 456  
How to Use Help command 602

## I

I/O constraints  
  multiple on same port 308  
I/O Standards panel, SCOPE 314

impl Tcl command 93  
implementation options  
  Options Panel 447  
Implementation Options  
  command 423, 444  
Implementation Options dialog  
  box 430, 444  
  Constraints panel 450  
  Design Planning panel 665  
  Device panel 445  
  High Reliability panel 456  
  Netlist Restructure panel 665  
  Options panel 447  
  Place and Route panel 483  
  Timing Report panel 455  
  Verilog panel 462  
  VHDL panel 458  
implementation options, device  
  partdata tcl command 105  
Implementation Results panel  
  Options for implementation dialog  
    box 453  
implementations  
  creating 424  
  naming 615  
Import IP  
  commands 485  
Import IP menu 485  
Import IP Package command 496  
Import Xilinx ISE Project command 489  
include command  
  verilog library directories 466  
include files 26  
index\_collection command 288  
Inputs/Outputs panel, SCOPE 306  
Insert Sub-project command 442, 636  
Instance Properties command  
  (hierarchical project  
    management) 634  
instances  
  dissolving 551  
  expanding paths between 544  
  expansion maximum limit 592  
  expansion maximum limit (per filtered  
    sheet) 595  
expansion maximum limit (per unfiltered sheet) 595  
finding by name 402, 686  
hiding and unhiding 550  
isolating paths through 550  
making transparent 551  
name display 590  
properties (Physical Analyst) 697  
selecting all in schematic 553  
Instances command  
  schematic selection 553  
  sheet selection 553  
Intel  
  launching MegaWizard 497  
  translating PIN file to constraint  
    file 504  
Intel models  
  Clearbox 180  
  set\_option syntax 180  
IP  
  license queuing syntax 211  
IP core wizard 500  
IP cores (SYNCORE)  
  building ram models 508  
ise2syn utility  
  GUI command 489  
  Tcl syntax 95  
Isolate Paths command 550

## J

Job Status command 501, 505  
job tcl command 96

## K

keyboard shortcuts 681, 684

## L

labels, displaying 590  
Launch Identify Instrumentor  
  command 506, 507  
launch\_megawiz command 497  
launch\_vivado command 97  
levels  
  See hierarchy

license  
  floating 602  
  saving 603  
  specifying in batch mode 210  
License Agreement command 602  
license queuing 212  
Limit Number of Paths 534  
Linux, 64-bit mapping 449  
load\_upf Tcl command 718  
Log File  
  HTML 420  
  text 420  
log file  
  displaying 416  
  Tcl commands for filtering 216  
Log File command  
  View menu 420  
Log Watch window  
  popup menu 607  
Log Watch Window command 416  
log\_filter Tcl command  
  syntax 98  
log\_report Tcl command 100  
Lowercase command 403

**M**

maximum parallel jobs 572  
measurement tool  
  Physical Analyst 693  
MegaWizard  
  launching Intel 497  
memory compiler 508  
memory, saving 552  
menubar 18  
Menus  
  Import IP 485  
menus  
  context-sensitive  
    *See* popup menus  
  Edit 402  
  HDL Analyst 542, 691  
  Help 602  
  Options 568, 691  
  popup

*See* popup menus  
Project 423  
Run 498, 666  
Tools, Synplify Premier DP Design Planner 667  
View 415

Messages  
  Tcl Window command 415  
Mouse Stroke Tutor command 602  
multicycle paths  
  clocks as from/to points 336  
  examples 326  
  POS 334  
  using different start/end clocks 325  
multiple drivers  
  resolving 182  
Multiple File Compilation Unit  
  Verilog panel 465  
multiple projects  
  displaying project files 576  
MultiProcessing  
  Continue on Error mode 571  
multiprocessing  
  maximum parallel jobs 572

**N**

name\_format Tcl command 720  
naming  
  region 663  
naming rules 388  
net drivers  
  displaying and selecting 545  
Netlist Editing Commands  
  get\_prop 758  
netlist formats  
  Implementation Options dialog box,  
    Implementation Results  
    panel 455  
Netlist Restructure panel,  
  Implementation Options dialog box 665  
nets  
  expanding hierarchically from pins and  
    ports 544  
  finding by name 402, 686  
  properties (Physical Analyst) 698

- selecting instances on 545  
New command 396, 658  
New Implementation command 430  
New Project command 397  
Next Bookmark command 403  
Next Error command 501  
Next Sheet command 417  
Normal View command 416
- O**
- object prefixes  
Tcl find command 223  
object properties  
annotated properties for analyst 230  
object search order (Timing Analyzer) 535  
object types  
Tcl find command 223  
objects  
displaying compactly 591  
expanding paths between 544  
filtering 546  
finding 699  
selecting in Floorplan Editor 692  
selecting overlapping in Physical Analyst view 703  
unselecting  
all in schematic 553  
OEM Content  
subproject supported only for Lattice 636  
Open command  
File menu 396, 426  
Open Project command 396  
open\_design command 103  
open\_file command 104  
opening  
project 396  
operators  
Tcl collection 240  
option settings  
reporting 87  
options  
prototyping 481
- setting 149  
Options command  
Xilinx submenu 569  
Options for implementation dialog box  
Implementation Results panel 453  
Options menu 568, 691  
Quartus submenu (Intel) 569  
Options panel  
Implementation Options dialog box 447  
output files  
.est 666  
.srp 543  
log. See log file  
srs  
See srs file  
overlapping objects  
selecting in Physical Analyst view 703  
overriding FSM Compiler 168
- P**
- Pan command 416  
panels  
Implementation Options dialog box  
Design Planning 665  
Netlist Restructure 665  
parameters  
overriding HDL 89  
partdata tcl command 105  
partitioned RTL view file (.srp) 543  
Paste - Replicate command 661  
Paste command 402  
path delays  
clocks as from/to points 337  
path filtering 534  
paths  
expanding hierarchically from pins and ports 544  
Physical Analyst  
command summary 692  
cross probing 688  
Enhanced Instance Display 689  
instance commands 694  
net commands 696  
properties 697

Physical Analyst view  
Configure Enhanced Instance Display command 687  
Cross Probing command 687  
filter command 687  
Find command 699  
finding objects 699  
Go to Location command 688  
instance properties 697  
net properties 698  
Physical Analyst Properties command 687  
Selection Transcription command 687  
Show Critical Path command 687  
unfilter command 687, 693

physical constraints  
translating QSF constraints 124

physical synthesis  
translating QSF constraints 124

physical synthesis netlist file (.srp) 543

pins  
displaying names 590  
displaying on transparent instances 551  
expanding hierarchically from 544  
expanding paths between 544  
isolating paths from 550  
maximum on schematic sheet 595

Place & Route  
creating a new options file for Xilinx 629

place & route  
run from the synthesis tool 627

Place & Route options file 628  
adding for Xilinx 628

place & route options file  
specifying 628

place & route timing correlation tab 524

Place and Route panel  
Implementation Options dialog box 483

place and route tcl commands  
job 96

pointers, mouse  
zoom 416

popup menus  
FSM Viewer 610  
Hierarchy Browser 641

Log Watch window 607  
Project view 613  
RTL view 641  
Tcl window 607  
Technology view 641

ports  
displaying names 590  
expanding hierarchically from 544  
expanding paths between 544  
finding by name 402  
isolating paths from 550  
selecting all in schematic 553

Ports command  
schematic 553  
sheet 553

POS  
interface 333

preferences  
project file display 575

Preferences command  
Design Planner tools menu 668

Preferences dialog box  
Design Planner 668

Preferred License Selection command 603

prefixes  
Timing Analyzer points 535

Previous bookmark command 403

Previous Error/Warning command 501

Previous Sheet command 417

primitives  
dividing 195  
internal logic, displaying 591

Print command 396

Print Setup command 396

printing  
view 396

printing image  
Create Image command 396

Product of Sums  
See POS

program\_terminate command 109

program\_version command 110

project files  
organization into folders 575

Project menu 423  
  commands 423

Project Options command 617

project Tcl command 111

Project view  
  display settings 575  
  popup menu 613  
  setting up 573

Project View Options command 568

project\_data Tcl command 120

project\_file Tcl command 121

project\_folder  
  Tcl command 123

projects  
  adding files 426  
  closing 397  
  creating (Build Project) 396  
  creating (New) 397  
  displaying multiple 576  
  opening 396

properties  
  find command 230  
  Physical Analyst 697  
  project 120

Properties command  
  Design Planner 663  
  instances in Physical Analyst 697  
  nets in Physical Analyst 698  
  View menu  
    Synplify Premier DP Design  
      Planner 662

prototyping options 481

Push Tristates  
  Verilog panel 465

Push/Pop Hierarchy command 417

**Q**

qsf2sdc  
  syntax 124

qsf2syn utility 125  
  syntax 125

Quartus  
  converting constraints to sdc 124  
  importing projects from, qsf2syn  
    syntax 125

qsf2sdc utility. *See* qsf2sdc 124

quitting a synthesis run 505

**R**

recent projects, opening 397

recording command 133

Redo command 402

Refresh command 607

Region Properties dialog box 663

regions  
  assigning critical paths 39  
  creating 48

regions, naming 663

Registers panel, SCOPE 309

regular expressions  
  Tcl find command 224

Reload command 641

Remove Files From Project  
  command 423

Remove Implementation command 615

remove\_from\_collection command 290

Replace command  
  Text Editor 403

replacing  
  text 413

Replicated Assignments dialog box 668

report\_clocks command 134

report\_messages command 136

reports  
  timing report (.ta file) 519  
  reset\_path timing constraint 349

Resolve Multiple Drivers option 182

Resolve Selection dialog box 703

resolving conflicting timing  
  constraints 338

Resource Center  
  *See* Technical Resource Center

resource sharing  
  Resource Sharing option 450

Resynthesize All command 498

RTL view  
  displaying 104

- opening hierarchical view 543  
popup menu 641  
popup menu commands 641
- Run All Implementations command 501  
Run Background Compile command 569  
Run Foreground Compile command 569  
Run menu 498, 666  
Run Tcl Script command 501, 502  
running place & route 627
- ## S
- sar file  
Archive Project command 433
- Save All command 396  
Save As command 396  
Save command 396
- schematic objects  
displaying compactly 591  
expanding paths between 544  
filtering 546  
unselecting all 553
- schematics  
displaying labels 590  
flattening 546  
navigating sheets 416  
opening hierarchical RTL 543  
sheet connectors 591  
unselecting objects 553
- SCOPE  
Attributes panel 313  
clock groups 297  
Clocks panel 296  
Collections panel 304  
Compile Points panel 316  
Delay Paths panel 310  
Generated Clocks panel 301  
I/O Standards panel 314  
Inputs/Outputs panel 306  
Registers panel 309  
TCL View 319
- SCOPE spreadsheet  
popup menu commands 606  
starting 294
- SCOPE timing constraints summary 295
- sdc  
standard sdc collection commands 282  
sdc2fdc utility 147
- Select All command 403  
Select All States command 418  
Select in Analyst command 608  
Select Net Driver command  
current level 545  
hierarchical 545
- Select Net Instances command  
current level 545  
hierarchical 545
- Select Place & Route option file dialog box 628
- Selected command 418  
sequential elements  
naming 389
- sequential optimizations  
disabling 168
- Set Library command 424  
set modules command (collections) 305  
set modules\_copy command  
(collections) 305
- Set Slack Margin command 549  
Set VHDL Library command 424  
set\_case\_analysis timing constraint 351  
set\_clock\_groups timing constraint 353  
set\_clock\_latency timing constraint 360  
set\_clock\_route\_delay timing  
constraint 362  
set\_clock\_uncertainty timing  
constraint 363  
set\_datapathonly\_delay timing  
constraint 365  
set\_false\_path timing constraint 368  
set\_hierarchy\_separator command 389  
set\_input\_delay timing constraint 371  
set\_isolation Tcl command 726  
set\_isolation\_control Tcl command 730  
set\_max\_delay timing constraint 374  
set\_min\_delay constraint 377  
set\_multicycle\_path timing  
constraint 380

set\_option  
  Resolve Multiple Drivers 182  
set\_option Tcl command 149  
set\_output\_delay timing constraint 384  
set\_reg\_input\_delay timing constraint 387  
set\_reg\_output\_delay timing constraint 388  
set\_retention Tcl command 733  
set\_retention\_control Tcl command 735  
set\_rtl\_ff\_names command 389  
set\_scope Tcl command 736  
settings  
  reporting option 87  
sheet connectors 591  
shortcuts  
  keyboard  
    See keyboard shortcuts  
Show All Hier Pins command 551  
Show All Regions command 662  
Show command 662  
Show Compile Points command 617  
Show Context command 550  
Show Critical Path command 549  
Show Replicated Assignments command 667, 668  
Show Selected command 662  
Show Timing Information command 549  
sizeof\_collection command 291  
slack  
  margin  
    setting 549  
  slack margin 534  
slice\_primitive Tcl command 195  
SolvNet Support command 598  
SOPC2Syn 197  
spy\_glass Tcl command 201  
SpyGlass tool  
  spy\_glass Tcl command 201  
srm file  
  hidden logic not saved 552  
srp file 543  
srr file  
  See log file  
srs file  
  hidden logic not saved  
Start Design Manager command 569  
Start Floorplanner command 569  
Start ISE Project Navigator command 569  
start/end points  
  Timing Report panel, Implementation Options dialog box 456  
state machines  
  See also FSM Compiler, FSM viewer, FSMs.  
displaying in FSM viewer 553  
encoding  
  displaying 612  
filtering states and transitions 418  
Status Bar command 415  
status\_report Tcl command 202  
stopping a synthesis run 505  
sxml2pxml Tcl command  
  using the standalone XML converter 213  
symbols  
  enabling name display 590  
  finding by name 402, 686  
syn\_connect 207  
syn\_create\_err\_net 208  
syn\_reference\_clock attribute  
  effect on multiple I/O constraints 309  
syn\_sxml2pxml Tcl command 213  
syn\_tristatetomux attribute  
  effect of tristate pushing 469  
SYNCORE wizard 500, 508  
Synopsys FPGA implementation tools  
  product information 601  
Synopsys FPGA products 601  
Synopsys Home Page command 601  
Synopsys Training Page command 601  
synplify command-line command 210  
synplify premier command-line command 210

synplify premier dp command-line command 210  
synplify\_pro command-line command 210  
syntax  
  bus dimension separator 390  
  bus naming 390  
Syntax Check command 500  
synthesis  
  stopping 505  
Synthesis Check command 500  
synthesis jobs  
  monitoring 505  
synthesis strategy  
  Tcl command 175  
synthesis\_off directive, handling 460  
synthesis\_on directive, handling 460  
Synthesize command 498  
SystemVerilog 464

**T**

Tcl  
  c\_diff collection command 241  
  c\_intersect collection command 242  
  c\_list collection command 243  
  c\_print collection command 243  
  c\_symdiff collection command 245  
  c\_union collection command 245  
  collection commands 240  
  set\_modules collection command 247  
  -verilog argument 25  
  -vhdl argument 25  
Tcl (Tool Command Language) 16  
tcl argument  
  -\_include 26  
Tcl collection commands 240  
  c\_diff 241  
  c\_intersect 242  
  c\_list 243  
  c\_print 243  
  c\_symdiff 245  
  c\_union 245  
  set\_modules 247  
Tcl collection operators 240  
Tcl commands  
  add\_file 25  
  add\_folder 30  
  assign\_to\_region 39  
  collections 305  
  constraint\_file 45  
  create\_region 48  
  dh\_module\_sources 63  
  generate\_instance\_constraints 84  
  get\_env 87  
  get\_option 87  
  hdl\_param 89  
  impl 93  
  log file commands 216  
  project 111  
  project\_data 120  
  project\_file 121  
  project\_folder 123  
  set\_option 149  
  slice\_primitive 195  
  syn\_sxml2pxml 213  
Tcl conventions 16  
Tcl expand command 237  
Tcl find command 221  
  case sensitivity 224  
  examples 225  
  object prefixes 223  
  object types 223  
  regular expression syntax 224  
  special characters 224  
  wildcards 224  
TCL Help command 602  
Tcl Script  
  Tcl Window command 415  
Tcl scripts  
  running 501, 502  
Tcl shell command  
  sdc2fdc 147  
TCL View, SCOPE 319  
Tcl window  
  popup menu 607  
Tcl Window command 415  
Technical Resource Center  
  accessing 601  
  specifying PDF reader (UNIX) 597  
  specifying web browser (UNIX) 597  
Technology view  
  creating 543

popup menu 641  
popup menu commands 641  
technology view  
  displaying 104  
text  
  copying, cutting and pasting 402  
  replacing 413  
Text Editor  
  popup menu commands 607  
the 95  
through constraints  
  point-to-point delays 311  
through points  
  clocks 337  
  lists, multiple 333  
  lists, single 332  
  multiple 333  
  product of sums UI 333  
  single 332  
  specifying for timing exceptions 332  
  specifying for timing report 533  
tie\_pin command 772  
timing analyst  
  generating report 519  
timing analyzer  
  wildcards 538  
timing constraints  
  checking 500  
  conflict resolution 338  
  constraint priority 338  
  create\_clock 343  
  create\_generated\_clock 345  
  FPGA 342  
  reset\_path 349  
  set\_case\_analysis 351  
  set\_clock\_groups 353  
  set\_clock\_latency 360  
  set\_clock\_route\_delay 362  
  set\_clock\_uncertainty 363  
  set\_datapathonly\_delay 365  
  set\_false\_path 368  
  set\_input\_delay 371  
  set\_max\_delay 374  
  set\_min\_delay 377  
  set\_multicycle\_path 380  
  set\_output\_delay 384  
  set\_reg\_input\_delay 387  
  set\_reg\_output\_delay 388  
timing exceptions  
  False Paths panel 328  
  multicycle paths 325  
  priority 338  
  specifying paths/points 328  
timing information, displaying (HDL Analyst tool) 549  
timing report  
  asynchronous clock report 531  
  defining through points 533  
  file (.ta) 519  
  specifying slack margin 534  
  using path filtering 534  
timing report file  
  generating custom 530  
  stand-alone 532  
Timing Report panel  
  Implementation Options dialog box 455  
  Number of Critical Paths 456  
  Start/End Points 456  
Timing Report View 520  
  options 520  
timing reports  
  file. See timing report file  
  filtering 531  
  parameters 519  
  stand-alone 519  
  stand-alone (.ta file) 519  
Tip of the Day command 603  
to points 535  
  clocks 336  
  multiple 331  
  objects 330  
  Timing Analyzer 535  
Toggle bookmark command 403  
Toolbars command 415  
Tools menu 667  
  commands 667  
tooltips  
  displaying 419  
Translate Vendor IO command 499  
translating constraints 504  
transparent instances  
  displaying pins 551  
tristates  
  pushing tristates, description 468

- pushing tristates, example 468  
pushing tristates, pros and cons 468
- U**
- Uncomment Code 403  
Undo command 402  
Unfilter command 418  
unfiltering 550  
  FSM diagram 418  
  schematic 550  
Unflatten Current Schematic command 547  
Unhide Instances command 550  
unhiding hidden instance 550  
Unified Power Format commands 705  
UNIX  
  configure external programs 569  
Unselect All command 553  
  View menu (FSM Viewer) 418  
updates from the Resource Center 601  
UPF commands 705  
  create\_power\_domain 711  
  load\_upf 718  
  name\_format 720  
  set\_isolation 726  
  set\_isolation\_control 730  
  set\_retention 733  
  set\_retention\_control 735  
  set\_scope 736  
Uppercase command 403  
utilities  
  sdc2fdc 147
- V**
- variables  
  accessing, get\_env Tcl command 87  
  reporting 87  
VCS Simulator command 501  
Vendor Constraints  
  Implementation Results panel,  
    Implementation Options dialog box 455  
  writing 455  
Verilog  
  'ifdef and 'define statements 470  
  allow duplicate modules (Tcl option) 152, 157  
  beta features 473  
  compiler, configuring 568  
  extract design parameters 470  
  library directories 466  
  specifying compiler directives 470  
Verilog 2001  
  Verilog panel 464  
-verilog argument  
  Tcl 25  
Verilog include files  
  using \_SEARCHFILENAMEONLY\_ directive 474  
Verilog panel 464  
  Implementation Options dialog box 462  
  Multiple File Compilation Unit 465  
  options 464  
  Push Tristates 465  
  SystemVerilog 464  
version information 603  
VHDL  
  compiler, configuring 568  
  enumeration encoding, default 460  
  ignoring code with synthesis off/on 460  
-vhdl argument  
  Tcl 25  
VHDL libraries  
  setting up 429  
VHDL panel  
  DesignWare options 461, 467  
  Implementation Options dialog box 458  
View FSM command 553  
View FSM Info File command 553  
View Log File command 416  
View menu 415  
  Filter submenu 418  
  Log File command 420  
  Rats Nest submenu 662  
  RTL and Technology view commands 416  
View Result File command 416  
View Sheets command 417  
Visual Properties command 417  
Vivado

launching [97](#)

## W

web browser  
specifying for UNIX [597](#)

web updates [601](#)

white boxes  
ise2syn [491](#)

wildcards  
Tcl find command [224](#)  
text Find [404](#)  
text replacement [413](#)  
timing analyzer [538](#)

Windows, 64-bit mapping [449](#)

Write Output Netlist Only command [499](#)

write\_vhdl [152](#)

## X

Xilinx

launching Vivado [97](#)  
place & route options file [628](#)  
running from synthesis tool [569](#)  
translating PAD file to constraint file [504](#)

Xilinx forward-annotation  
datapath only [365](#)

Xilinx Partition Flow  
using the standalone XML converter [213](#)

Xilinx Partition flow  
syn\_sxml2pxml Tcl command [213](#)

Xilinx projects  
converting with ise2syn [489](#)

XML converter  
using Xilinx Partition Flow [213](#)

## Z

zoom mouse pointer [416](#)

Zoom Out command [416](#)

Zoom Selected command  
Physical Analyst view [688](#)

