

Conformal[®] Extended Checks Reference

Product Version 23.2
October 2023

© 1997-2023 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

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

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

All other trademarks are the property of their respective holders.

Restricted Print Permission: This publication is protected by copyright and any unauthorized use of this publication may violate copyright, trademark, and other laws. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. This statement grants you permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used solely for personal, informational, and noncommercial purposes;
2. The publication may not be modified in any way;
3. Any copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement; and
4. Cadence reserves the right to revoke this authorization at any time, and any such use shall be discontinued immediately upon written notice from Cadence.

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

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

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

<u>About This Manual</u>	13
<u>Audience</u>	13
<u>How This Manual Is Organized</u>	13
<u>Conventions</u>	14

2

<u>Command Reference</u>	15
<u>Command Syntax</u>	16
<u>Wildcards</u>	17
<u>Using UNIX Commands with Conformal</u>	19
<u>ADD ALIAS</u>	21
<u>ADD ASSERTION CONSTRAINTS</u>	23
<u>ADD BLACK BOX</u>	25
<u>ADD CDC CHECK</u>	27
<u>ADD CDC FILTER</u>	32
<u>ADD CLOCK</u>	36
<u>ADD CLOCK ASSOCIATION</u>	38
<u>ADD DATA ASSOCIATION</u>	40
<u>ADD DISPLAY LIST</u>	43
<u>ADD DYNAMIC CONSTRAINTS</u>	44
<u>ADD FIFO INFO</u>	46
<u>ADD GENERATED CLOCK</u>	49
<u>ADD IGNORE RESET CONSTRAINT</u>	50
<u>ADD IGNORED PROPERTY</u>	52
<u>ADD INITIAL STATE</u>	57
<u>ADD INSTANCE CONSTRAINTS</u>	58
<u>ADD NOTTRANSLATE FILEPATHNAMES</u>	59
<u>ADD NOTTRANSLATED LINES</u>	61
<u>ADD NOTTRANSLATE MODULES</u>	63
<u>ADD PIN CONSTRAINTS</u>	65
<u>ADD PIN EQUIVALENCES</u>	67

Conformal Extended Checks Reference Manual

<u>ADD PRIMARY INPUT</u>	69
<u>ADD RENAMING RULE</u>	70
<u>ADD RESET_SYNC MODULE</u>	73
<u>ADD RESET SOURCE</u>	74
<u>ADD RULE FILTER</u>	76
<u>ADD RULE WAIVER</u>	80
<u>ADD SEARCH PATH</u>	82
<u>ADD SET SOURCE</u>	84
<u>ADD STATIC PROPERTY</u>	86
<u>ADD SYNC INPUT</u>	91
<u>ADD SYNCHRONIZATION RULE</u>	93
<u>ADD TIED SIGNALS</u>	99
<u>ADD VECTOR DEFINITION</u>	101
<u>ASSIGN PIN DIRECTION</u>	102
<u>BREAK</u>	104
<u>CHECKPOINT</u>	105
<u>CLOSE SCHEMATICS</u>	108
<u>CONTINUE</u>	109
<u>DELETE ALIAS</u>	110
<u>DELETE ASSERTION CONSTRAINTS</u>	111
<u>DELETE BLACK BOX</u>	112
<u>DELETE CDC CHECK</u>	113
<u>DELETE CDC FILTER</u>	118
<u>DELETE CLOCK</u>	119
<u>DELETE CLOCK ASSOCIATION</u>	120
<u>DELETE DATA ASSOCIATION</u>	121
<u>DELETE DISPLAY LIST</u>	122
<u>DELETE DYNAMIC CONSTRAINTS</u>	123
<u>DELETE FIFO INFO</u>	124
<u>DELETE GENERATED CLOCK</u>	126
<u>DELETE IGNORE RESET CONSTRAINT</u>	127
<u>DELETE IGNORED PROPERTY</u>	128
<u>DELETE INITIAL STATE</u>	133
<u>DELETE INSTANCE CONSTRAINTS</u>	134
<u>DELETE NOTTRANSLATE FILEPATHNAMES</u>	135
<u>DELETE NOTTRANSLATE MODULES</u>	136

Conformal Extended Checks Reference Manual

<u>DELETE PIN CONSTRAINTS</u>	138
<u>DELETE PIN EQUIVALENCES</u>	139
<u>DELETE PRIMARY INPUTS</u>	140
<u>DELETE RENAMING RULE</u>	141
<u>DELETE RESET_SYNC MODULE</u>	142
<u>DELETE RESET SOURCE</u>	143
<u>DELETE RULE FILTER</u>	144
<u>DELETE RULE WAIVER</u>	145
<u>DELETE SEARCH PATH</u>	146
<u>DELETE SET SOURCE</u>	147
<u>DELETE STATIC PROPERTY</u>	148
<u>DELETE SYNC INPUT</u>	153
<u>DELETE SYNCHRONIZATION RULE</u>	154
<u>DELETE TIED SIGNALS</u>	155
<u>DELETE VECTOR DEFINITION</u>	157
<u>DIAGNOSE CDC CHECK</u>	158
<u>DIAGNOSE STATIC PROPERTY</u>	160
<u>DOFILE</u>	163
<u>ELABORATE DESIGN</u>	164
<u>EXIT</u>	166
<u>EXPLORE</u>	167
<u>GENERATE CCD DOFILE</u>	171
<u>GENERATE FIFO INFO</u>	172
<u>HELP</u>	173
<u>HISTORY</u>	175
<u>INFO CHECKPOINT</u>	176
<u>LICENSE</u>	177
<u>MAN</u>	178
<u>NCENCRYPT</u>	181
<u>OPEN SCHEMATICS</u>	182
<u>PRINTENV</u>	183
<u>PROPAGATE CLOCK DOMAIN</u>	184
<u>PROVE</u>	185
<u>READ DESIGN</u>	187
<u>READ FSM ENCODING</u>	204
<u>READ INITIAL STATE</u>	205

Conformal Extended Checks Reference Manual

<u>READ LEF FILE</u>	208
<u>READ LIBRARY</u>	209
<u>READ RULE CHECK</u>	219
<u>READ SDC</u>	221
<u>REMOVE</u>	223
<u>REPORT ALIAS</u>	225
<u>REPORT ASSERTION CONSTRAINTS</u>	226
<u>REPORT BLACK BOX</u>	228
<u>REPORT CDC CHECK</u>	230
<u>REPORT CDC FILTER</u>	235
<u>REPORT CLOCK</u>	236
<u>REPORT CLOCK ASSOCIATION</u>	242
<u>REPORT CLOCK DOMAIN</u>	243
<u>REPORT CLOCK DOMAIN PRIORITY</u>	245
<u>REPORT COMMAND PROFILE</u>	246
<u>REPORT DATA ASSOCIATION</u>	247
<u>REPORT DESIGN DATA</u>	248
<u>REPORT DISPLAY LIST</u>	251
<u>REPORT DYNAMIC CONSTRAINTS</u>	252
<u>REPORT ENVIRONMENT</u>	253
<u>REPORT EXTRACTED PROPERTY</u>	254
<u>REPORT FIFO FAILURE</u>	259
<u>REPORT FIFO INFO</u>	260
<u>REPORT FLOATING SIGNALS</u>	262
<u>REPORT FSM</u>	264
<u>REPORT GATE</u>	266
<u>REPORT GENERATED CLOCK</u>	271
<u>REPORT IGNORE RESET CONSTRAINT</u>	272
<u>REPORT IGNORED PROPERTY</u>	274
<u>REPORT INITIAL STATE</u>	279
<u>REPORT INSTANCE CONSTRAINTS</u>	280
<u>REPORT LIBRARY DATA</u>	281
<u>REPORT MODULES</u>	283
<u>REPORT NOTTRANSLATE FILEPATHNAMES</u>	286
<u>REPORT NOTTRANSLATE MODULES</u>	287
<u>REPORT PATH</u>	288

Conformal Extended Checks Reference Manual

<u>REPORT PATH STATISTICS</u>	290
<u>REPORT PIN CONSTRAINTS</u>	293
<u>REPORT PIN DIRECTION</u>	294
<u>REPORT PIN EQUIVALENCES</u>	295
<u>REPORT PREDEFINED SYNCH_RULE</u>	296
<u>REPORT PRIMARY INPUTS</u>	297
<u>REPORT PRIMARY OUTPUTS</u>	298
<u>REPORT PROPERTY STATISTICS</u>	299
<u>REPORT PROVED DATA</u>	303
<u>REPORT RENAMING RULE</u>	310
<u>REPORT RESET_SYNC MODULE</u>	311
<u>REPORT RESET SOURCE</u>	312
<u>REPORT RULE CHECK</u>	313
<u>REPORT RULE FILTER</u>	318
<u>REPORT SEARCH PATH</u>	319
<u>REPORT SET SOURCE</u>	320
<u>REPORT STATIC PROPERTY</u>	321
<u>REPORT SYNC INPUT</u>	326
<u>REPORT SYNCHRONIZATION RULE</u>	327
<u>REPORT TIED SIGNALS</u>	328
<u>REPORT VALIDATED DATA</u>	330
<u>REPORT VECTOR DEFINITION</u>	340
<u>RESET</u>	341
<u>RESET CLP</u>	342
<u>RUN LIBRARY CHECK</u>	343
<u>SAVE DOFILE</u>	344
<u>SEARCH</u>	345
<u>SET ATTR INPUT PRAGMA KEYWORD</u>	346
<u>SET CASE SENSITIVITY</u>	348
<u>SET CDC OPTION</u>	349
<u>SET CLOCK_DOMAIN PRIORITY</u>	353
<u>SET CLOCK_DOMAIN RULE</u>	355
<u>SET COMMAND ECHO</u>	361
<u>SET COMMAND PROFILE</u>	363
<u>SET DIRECTIVE</u>	364
<u>SET DOFILE ABORT</u>	369

Conformal Extended Checks Reference Manual

<u>SET DW DEFINITION</u>	370
<u>SET FLATTEN MODEL</u>	372
<u>SET GUI</u>	375
<u>SET HDL OPTIONS</u>	376
<u>SET INSTANTIATION DEPTH</u>	397
<u>SET LOG FILE</u>	398
<u>SET NAMING RULE</u>	400
<u>SET NAMING STYLE</u>	413
<u>SET PARAMETER</u>	415
<u>SET PREDEFINED SYNCH RULE</u>	417
<u>SET PROVE EFFORT</u>	422
<u>SET PROVE OPTIONS</u>	424
<u>SET SPICE OPTION</u>	426
<u>SET ROOT MODULE</u>	428
<u>SET RTL TYPE</u>	429
<u>SET RULE HANDLING</u>	431
<u>SET SCREEN DISPLAY</u>	434
<u>SET STATETABLE</u>	435
<u>SET SYNTHESIS OFF COMMAND</u>	436
<u>SET SYNTHESIS ON COMMAND</u>	438
<u>SET SYSTEM MODE</u>	440
<u>SET UNDEFINED CELL</u>	441
<u>SET UNDEFINED PORT</u>	443
<u>SET UNDRIVEN SIGNAL</u>	444
<u>SET X HANDLING</u>	445
<u>SET WEB INTERFACE</u>	447
<u>SETENV</u>	449
<u>SUBSTITUTE BLACKBOX WRAPPER</u>	450
<u>SYSTEM</u>	452
<u>TCLMODE</u>	453
<u>TEST RENAMING RULE</u>	454
<u>USAGE</u>	455
<u>VALIDATE</u>	456
<u>VERSION</u>	457
<u>VPXMODE</u>	458
<u>WRITE BLACKBOX WRAPPER</u>	459

Conformal Extended Checks Reference Manual

<u>WRITE CDC CHECK</u>	464
<u>WRITE DESIGN</u>	470
<u>WRITE DIAGNOSIS DATA</u>	472
<u>WRITE LIBRARY</u>	473
<u>WRITE RULE CHECK</u>	474

3

<u>Modeling Rule Checks</u>	477
<u>Introduction to Modeling Messages</u>	478
<u>Modeling Rule Priority Levels</u>	479
<u>Rule Categories</u>	481
<u>Structural Rules</u>	482
<u>STRC1.1</u>	483
<u>STRC1.2</u>	485
<u>STRC2</u>	486
<u>STRC3</u>	489
<u>STRC4</u>	492
<u>STRC5.1</u>	493
<u>STRC5.2</u>	494
<u>STRC5.3</u>	495
<u>STRC6.1</u>	496
<u>STRC6.2</u>	497
<u>STRC7</u>	498
<u>Clock Definition Rules</u>	499
<u>CLK1</u>	500
<u>CLK2</u>	501
<u>CLK3</u>	503
<u>Initialization Rules</u>	505
<u>INIT1</u>	506
<u>INIT2</u>	507
<u>INIT3</u>	508
<u>INIT4</u>	510
<u>Finite State Machine Rules</u>	511
<u>FSM1</u>	512
<u>FSM2</u>	513

Conformal Extended Checks Reference Manual

<u>FSM3</u>	514
<u>FSM4.1</u>	515
<u>FSM4.2</u>	516
<u>Asynchronous Clock Domain Crossing</u>	517
<u>ACX1</u>	518
<u>ACX2</u>	519
<u>ACX3</u>	520
<u>ACX4</u>	521
<u>ACX5</u>	522
<u>ACX6</u>	523
<u>ACX7</u>	524
<u>Constraints</u>	525
<u>CNST1</u>	526

4

<u>Tcl Command Entry Mode Support</u>	527
<u>add_attribute</u>	529
<u>add_to_collection</u>	530
<u>append_to_collection</u>	531
<u>cfm_is_gui_mode</u>	532
<u>compare_collection</u>	533
<u>copy_collection</u>	534
<u>delete_attribute</u>	535
<u>echo_result</u>	536
<u>encrypt</u>	537
<u>exit</u>	538
<u>find</u>	539
<u>find_cfm</u>	542
<u>foreach_in_collection</u>	545
<u>get_current_module</u>	546
<u>get_fanin</u>	547
<u>get_fanout</u>	550
<u>get_handle_type</u>	553
<u>get_instances</u>	554
<u>get_license_mode</u>	555

Conformal Extended Checks Reference Manual

<u>get module definition</u>	556
<u>get names</u>	557
<u>get nets</u>	558
<u>get parent</u>	559
<u>get pins</u>	560
<u>get ports</u>	561
<u>get primitive type</u>	562
<u>get property</u>	563
<u>get relative path</u>	565
<u>get root module</u>	566
<u>get supply sets</u>	567
<u>get version info</u>	569
<u>index collection</u>	570
<u>help</u>	571
<u>ncdecrypt</u>	572
<u>objtype</u>	573
<u>query collection</u>	574
<u>query state</u>	575
<u>redirect</u>	577
<u>remove from collection</u>	578
<u>report attribute</u>	579
<u>set current module</u>	580
<u>sizeof collection</u>	581
<u>sort collection</u>	582
<u>tcl set command name echo</u>	583
<u>unset attribute</u>	584
<u>usage</u>	585

Conformal Extended Checks Reference Manual

About This Manual

This manual documents commands and modeling messages for the Conformal[®] Extended Checks software.

- Conformal Extended Checks

Conformal Extended Checks has equivalency checking capabilities with functional checks for ASIC design flows.

- Conformal XL

Conformal XL includes Extended Checks and extends equivalency checking capabilities to datapath synthesis and layout.

- Conformal GXL

Conformal GXL includes Conformal XL and extends equivalency checking capabilities to digital custom logic and custom memories.

- Conformal LowPower

Conformal LowPower enables low power equivalence and functional checks for isolation cells, level-shifter cells, and retention-register cells.

Audience

This manual is written for experienced designers of digital integrated circuits who must be familiar with RTL, synthesis, and design verification; as well as having a solid understanding of UNIX and Tcl/Tk programming.

How This Manual Is Organized

This manual contains the following chapters:

- Chapter 2, “Command Reference”

Lists Conformal Extended Checks commands, listed in alphabetical order. Each command entry includes information, such as syntax and description, related to the specific command.

Conformal Extended Checks Reference Manual

About This Manual

■ Chapter 3, “Modeling Rule Checks”

Contains the modeling rule check messages for modeling errors encountered during the analysis and modeling of the design in the Conformal Extended Checks software.

Conventions

The command syntax structure is as follows:

Convention	Definition
Bold Case	Indicates the command name.
UPPERCASE	Indicates the required minimum character entry.
< >	Indicates required arguments. Do not type the angle brackets.
[]	Indicates optional arguments. Do not type the square brackets.
	Indicates a choice among alternatives. Do not type the vertical bar.
\	The backslash character (\) at the end of a line indicates that the command you are typing continues on the next line.
. . .	Indicates multiple entries of an argument.
*	Indicates that the entry can use the wildcard (*) to represent zero or more characters.

Command Reference

This chapter describes the Conformal Extended Checks commands. The commands are presented in alphabetical order.

This chapter also includes the following sections:

- [Command Syntax](#) on page 16
- [Wildcards](#) on page 17
- [Using UNIX Commands with Conformal](#) on page 19

Command Syntax

- Conformal commands are *not* case sensitive.
- For every Conformal `ADD` command, there are corresponding `DELETE` and `REPORT` commands. For example:

`ADD INITIAL STATE`

`DELETE INITIAL STATE`

`REPORT INITIAL STATE`

- Conformal commands adhere to the "3-2-1" rule, which reduces the number of characters you must type.
 - ❑ 3: Type the leading three characters of the first term.
 - ❑ 2: Then type the leading two characters of the second term.
 - ❑ 1: End with the leading character of the third term.

In some cases, you must use more characters to resolve ambiguity. In this manual, the minimal sets of characters you must type are shown as *uppercase* letters in the syntax.

When you use the 3-2-1 rule in conjunction with the syntax guide to resolve any possible ambiguity, you reduce the number of characters in a command, as the following example shows:

`REPort SYnchronization Rule`

becomes

`rep sy r`

- Reduce the number of characters you type for command options to the characters shown in uppercase in the syntax, as the following example shows:

`DELete BLack Box -Module top`

becomes

`del bl b -m top`

Wildcards

On an as-needed basis, Cadence adds wildcard pattern-matching support to Conformal commands. The syntax convention that alerts you to wildcard support is the asterisk (*).

If you use a pattern where a filename or design object is expected, Conformal Extended Checks expands the pattern using the same conventions as in the UNIX shell.

■ Triggering pattern matching for filenames

To trigger pattern matching for filenames, a string must include at least one asterisk (*), question mark (?), or a pair of square brackets ([]).

■ Triggering pattern matching for design objects

To trigger pattern matching for design objects, a string must include at least one asterisk (*) or question mark (?).

In arguments that are considered patterns, the following characters have special meaning: ^, {, }, [,], ?, *. The dash (-) also has special meaning when it falls between square brackets.

Note: When you use wildcards for *design objects*, a wildcard can match a string that includes the hierarchical delimiter (/). For example, the pattern `*[10]` matches the design object `a/b/c[10]`.

When you use wildcards for *filenames*, every wildcard applies to part of a single directory or filename (this convention is the normal UNIX convention). For example, the pattern `*.v` does not match the filename `a/b/c.v`.

Special Characters for Filename and Design Object Pattern-Matching

Wildcard Character	Definition	Example
?	Match any single character.	<code>a?c</code> matches: <code>aac</code> , <code>abc</code> , <code>a4c</code> , <code>a?c</code>
*	Match any (possibly empty) string.	<code>a*c</code> matches the following: <code>ac</code> , <code>abc</code> , <code>a*c</code>

Conformal Extended Checks Reference Manual

Command Reference

Special Characters for Filename and Design Object Pattern-Matching

Wildcard Character	Definition	Example
[] That is, "[" followed by characters and "]"	<p>Match any single character listed between the square brackets: "[" and "]" .</p> <p>If the first character is "^", Conformal Extended Checks matches any single character <i>not</i> listed between the brackets.</p> <p>If the list shown between the brackets includes x–y, Conformal Extended Checks matches all characters in the range x–y.</p> <p>To match square brackets, you must include the escape character immediately preceding the square bracket.</p> <p>To be matched, the characters "-" and "]" must appear first in the list (possibly after ^).</p> <p>Note: For design objects, recall that a string triggers pattern matching with an asterisk or question mark. In those cases, this convention applies.</p>	<p>For filenames:</p> <p>a[145] matches the following:</p> <p>a1 a4 a5</p> <p>For design objects:</p> <p>a*[145] matches the following:</p> <p>ab1 a34 at5</p> <p>a*\[145\] matches the following:</p> <p>ab[145] a3[145] at[145]</p>
^	At the beginning of the pattern, the character "^" negates the result of the match:	^a* matches any name that does <i>not</i> begin with a .
\	Matches <i>only</i> the character that follows the "\" character.	<p>a\[10] matches the following:</p> <p>a[10]</p> <p>But it does not match:</p> <p>a1 a0</p>

Conformal Extended Checks Reference Manual

Command Reference

Special Characters for Filename and Design Object Pattern-Matching

Wildcard Character	Definition	Example
{p1,p2,...}	Matches any string matched by any of the sub-patterns listed.	<p>design/{top,sub{5,11}}/*.v matches the following:</p> <p>design/top/a.v design/sub5/b.v design/sub11/c.v</p> <p>Braces can nest. a/{d{e,f},g{h,i}}_0 matches the following:</p> <p>a/de_0 a/df_0 a/gh_0 a/gi_0</p>

Using UNIX Commands with Conformal

To execute a UNIX command from within the tools or a command script, start the line with an exclamation point "!" or with the `SYSTEM` command. When you execute commands in this way, they display to the standard output, and are recorded to a log file, if one is active.

Using the -all Option

This option applies within the given defaults. For example, the syntax for this example command is as follows:

```
REPort EXample
  <module_name...>
  [-All]
  [-Summary | -Verbose]
```

In the above syntax, `-Summary` is a default. Therefore, if you type the command with a module name and the `-all` option, but no other option, this command outputs a reports the *in the Summary format*.

Saving the Command's Output to a File

To save the command's output to a file, use the command line `>` operator. This works for all Conformal commands.

For example, to save the default output of the REPORT_GATE command to a file named `gate.out`, you would run the following:

```
report gate > gate.out
```

You can also use the `>>` operator to append output text to an existing file.

Note: Although some commands include a `-file <filename>` type option to save the command's output to a file, you should use the command line `>` operator instead.

ADD ALIAS

ADD ALias

```
<alias_name> <string>  
(Setup / Verify Mode)
```

Adds alias names for quick command entry. Assign an alias to long command names and arguments to avoid repeatedly typing them.

If you add an alias with an alias name that already exists, Conformal accepts the new alias and returns a warning as shown in the following example:

```
//Warning: Alias 'myread' is already defined, will be replaced by the new definition
```

To streamline your session, create aliases at the start of a Conformal session. Also, add aliases to an initial command file: `.verify.rc`.

The VERIFY_RC Environment Variable

Conformal Extended Checks checks for the `VERIFY_RC` environment variable. *If this variable is set, Conformal Extended Checks uses what the file this variable refers to and does not search for other files.*

If the `VERIFY_RC` variable is not set, Conformal Extended Checks continues the search as follows:

- First, the installation directory: `<install_dir>/share/cfm/lec/.verify.rc`
- Second, the user's home directory: `~/.verify.rc`
- Third, the current working directory: `./verify.rc`

If one or more `./verify.rc` files exist, Conformal Extended Checks runs them in the order noted above. This search order offers flexibility in the way you choose to use the initial command file. You can set up initial command files for any or all of the following purposes:

- Global initial command file for all users
- Global initial command file for an individual user
- Initial command file for a test case

Tcl Command

`add_alias`

Parameters

<code><alias_name></code>	Specifies the alias name.
<code><string></code>	Specifies the command that the alias represents.

Related Commands

DELETE ALIAS

REPORT ALIAS

ADD ASSERTION CONSTRAINTS

ADD ASsertion Constraints

```
<-ALL | < pathname*> ...>  
[-NOClocked | -Clocked]  
[-NORESET | -RESET]  
(Setup Mode)
```

Converts User Defined Assertions into Constraints. Conformal Extended Checks then uses these newly created constraints and the remaining assertion properties in the verification.

Use the `ADD ASSERTION CONSTRAINTS` command in either of the two ways: convert all assertions to constraints or specify one or more paths of the assertions you want Conformal Extended Checks to treat as constraints.

This command also specifies whether an assertion constraint is clocked (synchronous) to the clock signal that drives the assertion constraint. By default, Conformal Extended Checks does not consider assertion constraints to be synchronous to their driving clocks.

Note: When you write an assertion with the *options* parameter set to a value of 0 (which is the default), use this command to convert the *User Defined Assertion* into a *Constraint*. To specify whether the constraint is synchronized with the clock signal that drives it, use the `-noclocked` or `-clocked` option. (Refer to Chapter 2 of the *Open Verification Library Assertion Monitor Reference Manual* for additional information about the *options* parameter.)

Another way to define an assertion monitor as a constraint to formal tools is by setting the *options* parameter to a value of 1. Thus, when the value of an assertion's *options* parameter is 1, the assertion becomes a system constraint and the `SET FLATTEN MODEL` command (not the `ADD ASSERTION CONSTRAINTS` command) specifies whether assertion constraints are synchronous to the clock signals that drive them. This command option sets *all* system constraints.

Tcl Command

```
add_assertion_constraints
```

Parameters

<code>-ALL</code>	Adds all assertions as environmental constraints for the design.
-------------------	--

Conformal Extended Checks Reference Manual

Command Reference

<pathname*>	Adds assertions with the specified hierarchical path as environmental constraints for the design. This accepts wildcards.
-NOClOcked	Specifies that the assertion constraints are asynchronous to their driving clocks. <i>This option is the default.</i> Note: No-clocked constraints are valid at all times.
-ClOcked	Specifies that the assertion constraints are synchronous with their driving clocks. Note: Clocked constraints are valid on their respective rising clock edges.
-NORESET	Does not label the constraint as a reset constraint. <i>This option is the default.</i> A reset constraint is a constraint that keeps the design in functional mode, that is, not in reset. Refer to <u>ADD IGNORE RESET CONSTRAINT</u> for more information.
-RESET	Labels the constraint as a reset constraint. A reset constraint is a constraint that keeps the design in functional mode, that is, not in reset. Refer to <u>ADD IGNORE RESET CONSTRAINT</u> for more information.

Related Commands

ADD IGNORE RESET CONSTRAINT

DELETE ASSERTION CONSTRAINTS

REPORT ASSERTION CONSTRAINTS

SET FLATTEN MODEL

ADD BLACK BOX

ADD Black Box

```
[<[-Module | -Instance] [<name* ...>] [-File <filename>]>  
| <-All [ | -Design | -Library]>]  
(Setup Mode)
```

Specifies modules or instances that will be defined as blackboxes. These newly defined blackboxes are classified in the *User* class of blackboxes. Blackboxes already contained in the original design are classified in the *System* class of blackboxes.

Note: Conformal lets the wildcard (*) represent any zero or more characters in blackbox names.

Tcl Command

add_black_box

Parameters

-Module	Defines this list of module names as blackboxes. <i>This option is the default.</i>
-Instance	Defines this list of instance names as blackboxes.
<name* ...>	Defines the list of names of modules or instances. Note: Wildcard names are only supported for the module names. The wildcard (*) represents any zero or more characters in the blackbox module names.
-File <filename>	Specifies the name of the blackbox file. This file must contain only names of modules or instances, it is not a Verilog file. The names in the file are added to the [name...] list.
-All	Blackboxes all modules except the top module. -All applies within the given defaults.
-Design	(Used with the -all option only) Blackboxes all modules in the design. If you do not specify either -design or -library, blackboxing applies to both the library and the design.

Conformal Extended Checks Reference Manual

Command Reference

`-Library`

(Used with the `-all` option only) Blackboxes all modules in the library.

If you do not specify either `-design` or `-library`, blackboxing applies to both the library and the design.

Related Commands

ADD NOTRANSLATE MODULES

DELETE BLACK BOX

REPORT BLACK BOX

ADD CDC CHECK

ADD CDc Check

```
[-STRUctural
  | -FUNctional [SOURCE_DATA] [DESTINATION_DATA]
    [MUX_enable] [SINGLE_bit_change]
  | -SET
  | -RESET ]
[-SOURce <-ALL | <clock_domain*> ...>]
[-DESTination <-ALL | <clock_domain*> ...>]
[-FROM <-ALL | -REG | -PI | -BBOx | <instance* ...>>]
[-TO <-ALL | -REG | -PO | -BBOx | <instance* ...>>]
[<-MODule | -INStance> <name* ...>
  [-RECursive | -NORECursive] ]
(Verify Mode)
```

Turns on structural or functional Clock Domain Crossing (CDC) checks for the specified portion of the design. Options are order-dependent, since some option names are duplicated and have different purposes according to their placement (for example, `-source -all`, `-destination -all`, `-from -all`, `-to -all`). Thus for proper execution, you must adhere to the order shown in the syntax below.

Conformal Extended Checks extracts the clock domains from the circuit and runs CDC Checks for each synchronization rule. By default, Conformal Extended Checks checks only register to register paths. To specify other paths (primary inputs, primary outputs, and blackboxes) for structural CDC checks, use the `ADD DATA ASSOCIATION` command.

Functional CDC Checks build on the Structural CDC Checks; thus, *you must meet the following requirements for successful CDC Checks validation*:

- Run structural CDC checks before you add Functional CDC Checks. Although it is not necessary to run structural CDC checks on the entire design, you must run them on those portions where you want to run the functional CDC checks.
- After you add Functional CDC Checks, you can neither add nor delete structural CDC Checks until you delete all functional CDC Checks.

Conformal Extended Checks issues an error message if you do not satisfy these requirements.

Tcl Command

`add_cdc_check`

Parameters

`-STRUctural` Adds the Structural CDC Check as specified. Structural CDC Checks include many checks, such as `cdc_path_logic_type_check`, `cdc_path_destination_check`, `sync_chain_dff_number_check`. *This option is the default.*

See [REPORT VALIDATED DATA](#) for more information.

`-FUNctional` Adds the Functional CDC Check as specified. Functional CDC Checks include Source Data, Destination Data, MUX Enable, and Single-bit Change checks.

If you do not specify the type of Functional CDC Check, Conformal Extended Checks will run all checks by default.

`SOURCE_DATA` Adds the Source Data Functional CDC Check for the specified portion of the design.

The Source Data check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.

`DESTINATION_DATA`

Adds the Destination Data Functional CDC Check for the specified portion of the design.

The Destination Data check determines whether data entering the destination register is held stable for the destination register to latch it.

Conformal Extended Checks Reference Manual

Command Reference

	<code>MUX_enable</code>	<p>Adds the MUX Enable Functional CDC Check for the specified portion of the design.</p> <p>This check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.</p>
	<code>SINGLE_bit_change</code>	<p>Adds the Single-bit Change Functional CDC Check for the specified portion of the design.</p> <p>The Single-bit Change check ensures that a vector of signals crossing different clock domains can only be changed one bit at a time.</p>
<code>-SET</code>		Adds a check for asynchronous set pins of the flip-flops to ensure that they are consistent with the clock domain.
<code>-RESET</code>		Adds a check for asynchronous reset pins of the flip-flops to ensure that they are consistent with the clock domain.
<code>-SOURCE</code>		Adds the specified CDC Check to the paths with the specified source clock domain.
	<code>-ALL</code>	Uses all clock domains as the source.
	<code><clock_domain* ...></code>	Uses the specified clock domains as the source.
<code>-DESTINATION</code>		Adds the specified CDC Check to the paths with the specified destination clock domain.
	<code>-ALL</code>	Uses all clock domains as the destination.
	<code><clock_domain* ...></code>	Uses the specified clock domains as the destination.
<code>-FROM</code>		Adds the specified CDC Check to the paths that start from the specified key points.

Conformal Extended Checks Reference Manual

Command Reference

	-ALL	Uses all types of key points as the start of the paths. The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-REG	Uses registers as the start of the paths.
	-PI	Uses primary inputs as the start of the paths.
	-BBOx	Uses blackbox (outputs) as the start of the paths.
	<instance*>	Uses the specified instances as the start of the paths. This accepts wildcards. The instances can be registers, primary inputs, or blackbox outputs.
-TO		Adds the specified CDC check to the paths that end at the specified key points.
	-REG	Uses registers as the end of the paths.
	-PO	Uses primary outputs as the end of the paths.
	-BBOx	Uses blackbox (inputs) as the end of the paths.
	<instance*>	Uses the specified instances as the end of the paths. This accepts wildcards. The instances can be registers, primary outputs, or blackbox inputs.
-MODULE -INSTance	<name* ...>	Applies this check to the specified modules or module instance path names.
	-RECURSive	Includes submodules.
	-NORECURSive	Does not include submodules.

Related Commands

DELETE CDC CHECK

Conformal Extended Checks Reference Manual

Command Reference

PROPAGATE CLOCK DOMAIN

REPORT CDC CHECK

VALIDATE

ADD CDC FILTER

ADD Cdc Filter

```
<filter_name>
[-STRUctural
  | -FUNctional [SOURCE_DATA] [DESTINATION_DATA]
    [MUX_enable] [SINGLE_bit_change]
]
[-SYNC_RULE <rule_name*>...]
[-SOURce <-ALL | <clock_domain*> ...>]
[-DESTination <-ALL | <clock_domain*> ...>]
[-FROM <-ALL | -REG | -PI | -BBOx | instance* ...>]
[-TO <-ALL | -REG | -PO | -BBOx | instance* ...>]
[<-MODule | -INStance> <name* ...>
  [-RECurSive | -NORECurSive] ]
  | [ [-SET | -RESET] [-ALL| <ID|instance_name> ...] ]
]
(Verify Mode)
```

Filters CDC checks or sync rules on specified paths. These only filter out the checks from regions on which the checks were previously added by the `ADD CDC CHECK` command.

Tcl Command

`add_cdc_filter`

Parameters

<code><filter_name></code>	Specifies the CDC filter name to add.
<code>-STRUctural</code>	Filters the Structural CDC Check as specified. Structural CDC Checks include many checks, such as <code>cdc_path_logic_type_check</code> , <code>cdc_path_destination_check</code> , <code>sync_chain_dff_number_check</code> . <i>This option is the default.</i> See REPORT VALIDATED DATA for more information.

Conformal Extended Checks Reference Manual

Command Reference

-FUNctional

Filters the Functional CDC Check as specified. Functional CDC Checks include Source Data, Destination Data, MUX Enable, and Single-bit Change checks.

If you do not specify the type of Functional CDC Check, Conformal Extended Checks will run all checks by default.

Note: In order for Conformal Extended Checks to run the Functional CDC Checks on a path, that path must first pass the Structural CDC Check. Otherwise, Conformal Extended Checks will not allow you to add the Functional CDC Check for that path.

SOURCE_DATA

Filters the Source Data Functional CDC Check for the specified portion of the design.

The Source Data check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.

DESTINATION_DATA

Filters the Destination Data Functional CDC Check for the specified portion of the design.

The Destination Data check determines whether data entering the destination register is held stable for the destination register to latch it.

MUX_enable

Filters the MUX Enable Functional CDC Check for the specified portion of the design.

This check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.

Conformal Extended Checks Reference Manual

Command Reference

`SINGLE_bit_change`

Filters the Single-bit Change Functional CDC Check for the specified portion of the design.

The Single-bit Change check ensures that a vector of signals crossing different clock domains can only be changed one bit at a time.

<code>-SYNC_RULE</code>	Specifies to filter the checks only for these sync rules.
<code>-SOURCE</code>	Filters the specified CDC Check to the paths with the specified source clock domain.
<code>-ALL</code>	Uses all clock domains as the source.
<code><clock_domain* ...></code>	Uses the specified clock domains as the source.
<code>-DESTINATION</code>	Adds the specified CDC Check to the paths with the specified destination clock domain.
<code>-ALL</code>	Uses all clock domains as the destination.
<code><clock_domain* ...></code>	Uses the specified clock domains as the destination.
<code>-FROM</code>	Adds the specified CDC Check to the paths that start from the specified key points.
<code>-ALL</code>	Uses all types of key points as the start of the paths. The type of key points includes registers, primary inputs, and blackboxes (outputs).
<code>-REG</code>	Uses registers as the start of the paths.
<code>-PI</code>	Uses primary inputs as the start of the paths.
<code>-BBOX</code>	Uses blackbox (outputs) as the start of the paths.

Conformal Extended Checks Reference Manual

Command Reference

	<code><instance*></code>	Uses the specified instances as the start of the paths. This accepts wildcards. The instances can be registers, primary inputs, or blackbox outputs.
<code>-TO</code>	Adds the specified CDC check to the paths that end at the specified key points.	
	<code>-REG</code>	Uses registers as the end of the paths.
	<code>-PO</code>	Uses primary outputs as the end of the paths.
	<code>-BBOx</code>	Uses blackbox (inputs) as the end of the paths.
	<code><instance*></code>	Uses the specified instances as the end of the paths. This accepts wildcards. The instances can be registers, primary outputs, or blackbox inputs.
<code>-MODULE -INSTance</code>	<code><name* ...></code>	Applies this filter to the specified modules or module instance path names.
	<code>-RECurSive</code>	Includes submodules.
	<code>-NORECurSive</code>	Does not include submodules.
<code>-SET</code>	Adds a filter for asynchronous set pins of the flip-flops to ensure that they are consistent with the clock domain.	
<code>-RESET</code>	Adds a filter for asynchronous reset pins of the flip-flops to ensure that they are consistent with the clock domain.	

Related Commands

[ADD CDC CHECK](#)

[DELETE CDC FILTER](#)

[REPORT CDC FILTER](#)

[VALIDATE](#)

ADD CLOCK

ADD CLock

```
<0 | 1>  
[-WAVEform <offset> <width> <total_units>]  
<primary_pin* ...>  
[-NO_SUPpressible | -SUPpressible]  
(Setup / Verify Mode)
```

Defines the waveform for the specified clock pins in the design.

Specify a waveform with a default waveform (0 or 1):

- Default waveform 0 is 0 1 1 2
- Default waveform 1 is 1 1 1 2

Note: All clock waveforms are based on the same (implicit) unit.

Alternatively, use the `-waveform` option to specify the offset, width, and total units.

In the Conformal Extended Checks GUI mode, do the following to access the Clock window to create, view, and modify clock waveforms graphically:

- In the main Conformal Extended Checks window, choose the *Setup* pull-down menu.
- Choose *Clock*.

Tcl Command

`add_clock`

Parameters

- | | |
|---|---|
| 0 | <i>The default waveform is 0 1 1 2.</i> The waveform starts at value 0 at time unit 0, goes up to a value of 1 at time unit 1, and back to a value of 0 at time unit 2. |
| 1 | <i>The default waveform is 1 1 1 2.</i> The waveform starts at value 1 at time unit 0, goes down to a value of 0 at time unit 1, and back to a value of 1 at time unit 2. |

Conformal Extended Checks Reference Manual

Command Reference

-WAVEform	<p>Specifies the properties of the clock waveform.</p> <p><offset> is the initial value of the clock waveform for the specified number of time units.</p> <p><width> is the value of the clock waveform changes from its initial value (0 -> 1 or 1 -> 0) and remains at the new value for width time units.</p> <p><total_units> is the period of the clock waveform in time units.</p>
<primary_pin* ...>	<p>Specifies the primary pin for which the clock waveform is defined.</p>
-NO_SUPpressible	<p>Does not suppress the pulse of the defined clock. <i>This option is the default.</i></p>
-SUPpressible	<p>Arbitrarily suppresses the pulse of the defined clock.</p>

Examples

For examples of this command and waveform specification, see the [Examples of Clock Waveforms](#) section in the *Conformal Extended Checks User Guide*.

Related Commands

[DELETE CLOCK](#)

[REPORT CLOCK](#)

ADD CLOCK ASSOCIATION

ADD CLock Association

```
< <pathname> [<pathname* ...>]>  
[-Name <name>]  
(Verify Mode)
```

In asynchronous clock designs, synchronization failures can occur when signals are generated in one clock domain and sampled in another clock domain. Conformal Extended Checks clock domain checking analyzes the designs and identifies these potential failures.

In Conformal Extended Checks, each defined clock is considered to be in its own unique clock domain, even though the clocks are defined with the same phase and frequency. Use this command to associate a group of synchronous clocks as a single clock domain. After the clocks are associated, signals belonging to these associated clocks are considered to be in a single clock domain. It is also possible to add clocks to an existing association by issuing the command again with the same representative domain (the first argument).

A clock domain is defined as any clock signals driving D-latches or D flip-flops.

Note: Some input pins are not clock inputs, but they can be sampled by D flip-flops, D-latches, and output pins. Therefore, input pins can be considered separate domains so that Conformal Extended Checks can identify potential synchronization failures from input pins. Similarly, output pins that can be sampled by another chip not in the design can also be considered as separate clocks so that a potential synchronization failure at the output pins can be identified. To assign domains to primary inputs, primary outputs, and blackboxes, use the ADD DATA ASSOCIATION command, not the ADD CLOCK ASSOCIATION command.



Adding a clock association changes the clock IDs for other clock domains, and makes the data association invalid. As a result, running this command after ADD DATA ASSOCIATION deletes the data association.

Tcl Command

```
add_clock_association
```

Parameters

pathname [<pathname* ...>]

Specifies the name of the path(s). The first path identifies a known clock domain as the representative domain. Other clock domains are associated with (synchronous to) the representative domain.

Subsequent paths specify the clock domains that are grouped into the same domain as the representative clock domain.

-Name <label_name>

Specifies a label name for associated clocks. After using this setting, you can refer to a clock domain using its label name.

Related Commands

ADD CLOCK

ADD GENERATED CLOCK

DELETE CLOCK ASSOCIATION

PROPAGATE CLOCK DOMAIN

REPORT CLOCK ASSOCIATION

REPORT CLOCK DOMAIN

ADD DATA ASSOCIATION

ADD Data Association

```
< <pathname> [<pathname* ...>]>  
  <-Domain <clock_domain> | -NEW [-Name <name>]  
  | -UNIVersal <-PI | -PO | -PIO | <pathname* ...>>  
  | -INDividual <-PI | -PO | PIO | -BBOX> ...  
(Verify Mode)
```

Assigns the specified data signals to a clock domain. The clock domain can be any of the followings:

- Existing clock domain
- Newly created clock domain
- Universal domain

This command can be applied only to PI, PO, and blackbox pins. You cannot apply this command to wire instances.

A signal belonging to the universal domain is considered to be synchronous to any clock domain. Hence, Conformal Extended Checks does not mark any violations for signal crossings that involve the universal domain.



Running the ADD CLOCK ASSOCIATION or ADD GENERATED CLOCK command after running this command deletes the data association.

Tcl Command

add_data_association

Parameters

`<pathname> [<pathname* ...>]`

Specifies the name of the path(s). The first path identifies a data signal that will be assigned to the specified clock domain and serve as the representative name for the association.

Subsequent paths specify the data signals that are assigned to the specified clock domain.

`-Domain <clock_domain>`

Associates an existing clock domain with the specified data signals.

`-NEW`

Associates a newly created clock domain with the specified data signals.

`-Name <name>`

Specifies a label name for the newly created domain. Signals that are associated with the domain are grouped together and the specified name serves as the group name.

Note: Wildcards are not supported in the first argument if you use `-NEW` without specifying `-Name`.

`-UNIVersal <-PI | -PO | PIO | pathname* ...>`

Associates the following data signals with the universal domain.

A signal belonging to the universal clock domain is considered to be synchronous to any clock domain. Hence, Conformal Extended Checks does not mark any violations for signal crossings that involve the universal domain.

You can choose from the followings:

`-PI` for primary input pins

`-PO` for primary output pins

`-PIO` for primary inout pins

Conformal Extended Checks Reference Manual

Command Reference

-INDividual <-PI | -PO | PIO | -BBOX>...

Associates each input and/or output pin to a new and distinct clock domain if it was not already explicitly associated to any clock domain.

You can choose from the followings:

-PI for primary input pins

-PO for primary output pins

-PIO for primary inout pins

-BBOX for black box input, output and inout pins.

Related Commands

DELETE DATA ASSOCIATION

PROPAGATE CLOCK DOMAIN

REPORT DATA ASSOCIATION

ADD DISPLAY LIST

ADD Display List

```
<-PO | <pathname | id> ...>  
(Verify Mode)
```

Adds signals to the list Conformal displays during the `DIAGNOSE STATIC PROPERTY` command.

Tcl Command

```
add_display_list
```

Parameters

- | | |
|--|---|
| <code>-PO</code> | Display the list of primary outputs after verification. |
| <code><pathname id> ...</code> | Add the specified gate to the display list. |
- Note:** ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

`DELETE DISPLAY LIST`

`DIAGNOSE STATIC PROPERTY`

`REPORT DISPLAY LIST`

ADD DYNAMIC CONSTRAINTS

ADD Dynamic Constraints

```
<PI | 0 | 1 | ONEHOT | ZEROONEHOT>  
<id | pathname*> ...  
(Verify Mode)
```

Adds dynamic constraints, which you can use to explore properties. Use this command with the `EXPLORE` command.

Note: Dynamic constraints are recorded, but are not processed. Dynamic constraints provide a convenient way to perform design exploration and analysis using *temporary* assumptions. You can use dynamic constraints to explore a property in different cases, either to get more meaningful results or to debug a failed property.

Tcl Command

```
add_dynamic_constraints
```

Parameters

PI	Designates the specified flattened gates as primary inputs. This option introduces more behavior into the design.
0	Cuts the outputs of the specified flattened gates and ties them to zero.
1	Cuts the outputs of the specified flattened gates and ties them to one.
ONEHOT	Applies a one-hot condition to the specified flattened gates.
ZEROONEHOT	Applies a zero-one-hot condition to the specified flattened gates.
<id pathname*> ...	Adds dynamic constraints to the flattened gate specified by the ID or path. The pathname accepts wildcards.

Examples

For a set of sample commands that shows this and related commands in context, see the example for the [EXPLORE](#) command.

Related Commands

[DELETE DYNAMIC CONSTRAINTS](#)

[EXPLORE](#)

[REPORT DYNAMIC CONSTRAINTS](#)

ADD FIFO INFO

ADD Fifo Info

```
<-Index <index>
  | -INStance <instance_name>
  | -Module <module_name> >
[ [-MEM <reg1 reg2 ...>]
[-REAd <data1 data2 ...>]
[-RADr <reg1 reg2 ...>]
[-WADr <reg1 reg2 ..>]
[-RSYNc <<reg1_1 reg1_2> ...>]
[-WSYNc <<reg1_1 reg1_2> ...>]
[-RGRay <reg1 reg2 ...>]
[-WGRay <reg1 reg2 ...>] ]
(Verify Mode)
```

Use this command to add FIFO instances or add components to a FIFO instance.

Note: When adding components to FIFO instances using the `-module` option, you must specify the path relative to the FIFO module. When adding components/FIFO instances using the `-index/-instance` options, you must use the full hierarchical path.

Tcl Command

`add_fifo_info`

Parameters

- | | |
|--|---|
| <code>-Index <index></code> | Specifies an existing FIFO using its index.

To report the indices of FIFOs, use the <code>REPORT FIFO INFO</code> command. |
| <code>-INStance <instance_name></code> | Selects a FIFO instance with the specified name. If it does not exist, one will be created with the specified name. |
| <code>-Module <module_name></code> | Adds FIFO instances for each instance of the specified module. |
| <code>-MEM <reg1 reg2 ...></code> | Adds specified memory registers to the FIFO. |

Conformal Extended Checks Reference Manual

Command Reference

-REAd <data1 data2 ...>

Add the specified registers or primary outputs as FIFO memory output.

-RADr <reg1 reg2 ...> Adds specified registers as FIFO read address registers.

-WADr <reg1 reg2 ...> Adds specified registers as FIFO write address synchronizer registers.

-RSYNc <<reg1_1 reg1_2 ...>

Adds specified registers as FIFO read address synchronizer registers.

Note: Registers must be specified in a strict order. For example, if a gray code register has three bits, there would be three first-level DFF synchronizers and three second-level DFF synchronizers. In this case, specify the register names in the following order:

1. First level register for the first bit
 2. Second level register for first bit
 3. First level register for the second bit
 4. Second level register for second bit
- and so on.

-WSYNc <<reg1_1 reg1_2 ...>

Adds specified registers as FIFO write address synchronizer registers.

Note: Registers must be specified in a strict order. See example for -RSYNc.

-RGRay <reg1 reg2 ...>

Adds specified registers as read gray code registers of the FIFO.

-WGRay <reg1 reg2 ...>

Adds specified registers as write gray code registers of the FIFO.

Example

If a module with the name `myfifo` has two instances in the design, the following command adds two FIFO instances:

```
add fifo info -module myfifo
```

You can then use the `REPORT FIFO INFO` command to verify the creation of the two FIFO instances and to determine their index number so that you can add FIFO components/information to those instances.

If the added FIFO instances are called `fifo1` and `fifo2` and have an indices of 1 and 2, respectively, you can add components to them using the `-index` option. For example:

```
add fifo info -index 1 -memory fifo1/mem_reg1 fifo1/mem_reg2
add fifo info -index 1 -read fifo1/read_reg1 fifo1/read_reg2
```

The following command illustrates how to add a FIFO instance and components within a single command:

```
add fifo info -module myfifo -memory mem_reg1 mem_reg2 -read read_reg1 read_reg2
```

If user is using "-instance/-index" option, he should give the component name with full hierarchical path.

Related Commands

DELETE FIFO INFO

GENERATE FIFO INFO

REPORT FIFO INFO

ADD GENERATED CLOCK

ADD GEnErated Clock

```
<id | pathname*> ...  
(Verify Mode)
```

Creates a new clock domain for the specified ID or path of an internally generated clock.

Note: This clock domain is for use in Clock Domain Checking only.



Adding this new clock domain changes the clock IDs for other clock domains, and makes the data association invalid. As a result, running this command after ADD DATA ASSOCIATION deletes the data association.

Tcl Command

```
add_generated_clock
```

Parameters

`<id | pathname*> ...` Creates a new clock domain for the signal with the specified ID or path. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

DELETE GENERATED CLOCK

PROPAGATE CLOCK DOMAIN

REPORT CLOCK DOMAIN

REPORT GENERATED CLOCK

ADD IGNORE RESET_CONSTRAINT

```
ADD IGNORE RESET_constraint
    <-ALL
    | <-BRANCH_enable
    | -FSMTransition
    | -FSMReachability
    | -TRI_state
    >>
(Setup / Verify Mode)
```

Ignores reset constraints for the specified types of checks.

To obtain meaningful results, Conformal Extended Checks normally runs functional checks on a design when it is in functional mode. And constraints (called the reset constraints) are used to keep a design in functional mode. However, for certain types of functional checks, such as Branch Enable, FSM Transition, FSM Reachability, and Tristate Stuck-on/Stuck-off, applying reset constraints to the design actually invalidates some of the results. For example, during a Branch Enable check, if a zero constraint is applied to the "reset" signal to keep the design in functional mode, then the Branch Enable check for the "reset" signal branch will fail. However, this FAIL result is caused by the reset constraint and not by a design flaw. Ideally, for these types of checks, Conformal should ignore these reset constraints. To remedy this situation, use this command to ignore reset constraints for the specified checks.

Note: Other checks such as bus and dont_care still require the reset constraints and are not affected by this command.

Tcl Command

```
add_ignore_reset_constraint
```

Parameters

-ALL	Ignores reset constraints for Branch Enable, FSM Transition, FSM Reachability, and Tristate Stuck-on/Stuck off checks. <i>This option is the default.</i>
-BRANCH_enable	Ignores reset constraints for Branch Enable checks.
-FSMTransition	Ignores reset constraints for FSM Transition checks.
-FSMReachability	Ignores reset constraints for FSM Reachability checks.

Conformal Extended Checks Reference Manual

Command Reference

-TRI_state	Ignores reset constraints for Tristate Stuck-on/Stuck off checks.
------------	---

Related Commands

ADD ASSERTION CONSTRAINTS

ADD PIN CONSTRAINTS

DELETE IGNORE RESET CONSTRAINT

REPORT IGNORE RESET CONSTRAINT

ADD IGNORED PROPERTY

ADD IGNORED PROPERTY

```
<-ALL
|<-TRI_state [-ALL | -PO | <id | pathname*> ...]
| -DONT_CARE [-ALL | -X_ASSIGNment | -RANGE_overflow
| -FULL_case | -PARAllel_case | <id | pathname*> ...
]
| -BRANCH_enable
| [-ALL | -IF_else | -CASE
| -DEFault | -FOR_loop | <id | pathname*> ...
]
| <-BUS | -MULTI_PORT
| -SET_RESET
| -FSM | -FSMReachability
| -FSMTransition | -FSMDeadlock
| -NO_DIVide_by_zero
| -ENCoding_completeness
>
[ -ALL | <id | pathname*> ...]
> ...
>
(Verify Mode)
```

Excludes specified properties from the "Prove" list. When you choose the Verify mode, Conformal automatically extracts all possible predefined properties. The correct sequence of commands is as follows:

1. Run the `ADD IGNORED PROPERTY` command to exclude specified properties from the "Prove" list.
2. Run the `ADD STATIC PROPERTY` command to add properties to the "Prove" list.
3. Run the `PROVE` command. At this point, Conformal proves all the properties included on the "Prove" list.

After you add ignored properties, use the `REPORT IGNORED PROPERTY` command to review your additions. And use the `DELETE IGNORED PROPERTY` command to remove the ignored status.

Note: When you specify properties with the `ADD IGNORED PROPERTY` command, you cannot specify these same properties with the `ADD STATIC PROPERTY` command until you make them available. Use the `DELETE IGNORED PROPERTY` command for this purpose. Likewise, when you add specified properties to the prove list, you cannot designate these same properties as "ignored" until you delete them from the prove list. Use the `DELETE STATIC PROPERTY` command for this purpose.

Tcl Command

add_ignored_property

Parameters

-ALL	Ignores all predefined properties.
-TRI_state	Ignores tristate properties, as specified.
-ALL	Ignores all tristate properties. <i>This option is the default.</i>
-PO	Ignores tristates connected to the primary output.
<id pathname* ...>	Ignores tristate properties for pins with the specified IDs or paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-DONT_CARE	Ignores Don't Care properties, as specified.
-ALL	Ignores Don't Care properties for all cases. <i>This option is the default.</i>
-X_ASSIGNment	Ignores Don't Care properties for X-assignment cases.
-RANGE_overflow	Ignores Don't Care properties for array index overflow cases.
-FULL_case	Ignores Don't Care properties for full_case synthesis directive cases.
-PARAllel_case	

Conformal Extended Checks Reference Manual

Command Reference

Ignores Don't Care properties for `parallel_case` synthesis directive cases.

<id | pathname* ...>

Ignores Don't Care properties for the specified IDs or paths. This accepts wildcards.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BRANCH_enable

Ignores Branch Enable properties, as specified.

-ALL

Ignores all Branch Enable properties. That is, do not check whether case, if-else, or for loop conditional branches can be enabled. *This option is the default.*

-IF_else

Ignores Branch Enable properties for if-else conditional branches. That is, do not check whether if-else conditional branches can be enabled.

-CASE

Ignores Branch Enable properties for case conditional branches. That is, do not check whether case conditional branches can be enabled. (This option does not include default branches.)

-DEFAULT

Ignores Branch Enable properties for case-default conditional branches. That is, do not check whether case-default conditional branches can be enabled.

-FOR_loop

Ignores Branch Enable `for_loop` properties. That is, do not check whether `for_loops` can be executed.

<id | pathname* ...>

Conformal Extended Checks Reference Manual

Command Reference

Ignores the specified Branch Enable properties. That is, do not check whether the specified Branch Enable properties can be enabled.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BUS	Ignores Bus properties, as specified.
-MULTI_PORT	Ignores Multi-port properties, as specified.
-SET_RESET	Ignores Set-Reset properties, as specified.
-FSM	Ignores all finite state machine properties, as specified.
-FSMREachability	Ignores finite state machine Reachability properties, as specified.
-FSMTransition	Ignores finite state machine Transition properties, as specified.
-FSMDeadlock	Ignores finite state machine Deadlock properties, as specified.
-NO_DIVide_by_zero	Ignores No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
-ENCoding_completeness	Ignores Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
-ALL	<p>Ignores all properties, as specified. For example, the following command excludes all Bus properties from the "Prove" list:</p> <pre>add ignored property -bus -all</pre> <p><i>Does not</i></p>
<id pathname* ...>	Ignores the specified properties with the specified ID or hierarchical path.

Related Commands

ADD BLACK BOX

ADD NOTRANSLATE MODULES

ADD STATIC PROPERTY

DELETE IGNORED PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

REPORT FSM

REPORT GATE

REPORT IGNORED PROPERTY

REPORT PROVED DATA

REPORT STATIC PROPERTY

ADD INITIAL STATE

ADD INitial State

<0 | 1 | X> <instance_pathname*...>
(Setup Mode)

Adds the initial state value to instances specified by the instance path. If `instance_pathname` points to a module, Conformal initializes the initial state value to all state elements within this module.

Tcl Command

`add_initial_state`

Parameters

0	Sets the initial state of the specified instance path to 0.
1	Sets the initial state of the specified instance path to 1.
X	Sets the initial state of the specified instance path to X.
<instance_pathname*> ...	Sets initial states for the specified instance. This accepts wildcards.

Related Commands

DELETE INITIAL STATE

READ INITIAL STATE

REPORT INITIAL STATE

ADD INSTANCE CONSTRAINTS

ADD INstance Constraints

```
<0 | 1> <instance_pathname*...>  
[-REPlace]  
(Setup Mode)
```

Places constraints on the specified instances by placing a state value on their output. This command only takes a value of 0 or 1 on the output of the instances. This command can apply to DFFs and D-Latches.

Tcl Command

```
add_instance_constraints
```

Parameters

0	Constrains the specified instance paths to a 0-state.
1	Constrains the specified instance paths to a 1-state.
instance_pathname*...	Places constraints on the specified instances. The instances are either D flip-flops or D-latches. This accepts wildcards.
-REPlace	Changes the previously specified instance constraint.

Related Commands

DELETE INSTANCE CONSTRAINTS

REPORT INSTANCE CONSTRAINTS

ADD NOTTRANSLATE FILEPATHNAMES

ADD NOTtranslate Filepathnames

```
<filepath_names* ...>  
[ | -Library | -Design]  
(Setup Mode)
```

Specifies library or design files, where modules defined in these files will not be translated when running the `READ LIBRARY` or `READ DESIGN` command. These modules will automatically become blackboxes.

This command is applied during initial parsing, so name matching applies only to original module names. For parameterized or VHDL generic modules whose names are determined and applied by the Conformal software after parsing and preprocessing, you must use the `ADD BLACK BOX` command.

The following are examples of modules that should not be compiled:

- RAM models
- Analog blocks
- Modules that contain non synthesizable RTL constructs

Wildcard: The wildcard (*) represents any zero or more characters in module names.

Tcl Command

```
add_nottranslate_filepathnames
```

Parameters

<filepath_names* ...>

Specifies the filepath name, which can be directory names and Verilog filenames. The wildcard (*) and search path is supported.

-Library

Does not translate the modules in the specified library. If you do not specify `-library` or `-design`, Conformal applies this command to both the library *and* design modules.

Conformal Extended Checks Reference Manual

Command Reference

`-Design` Does not translate the modules in the specified design. If you do not specify `-library` or `-design`, Conformal applies this command to both the library *and* design modules.

Related Commands

ADD NOTTRANSLATE MODULES

DELETE NOTTRANSLATE FILEPATHNAMES

DELETE NOTTRANSLATE MODULES

REPORT NOTTRANSLATE FILEPATHNAMES

REPORT NOTTRANSLATE MODULES

ADD NOTTRANSLATED LINES

ADD NOTtranslated Lines

```
<filename>  
<[start:end | line] ... >  
(Setup Mode)
```

Specifies lines to skip in a Verilog or VHDL file. This has the same effect as commenting out the lines in the file. With this command, you can instruct the Conformal software to skip certain lines in the design file without having to modify the file.

Tcl Command

```
add_nottranslated_lines
```

Parameters

<filename>	Specifies the Verilog or VHDL file.
start:end	Specifies the a line number range to skip. For example, 6:8 would skip lines 6, 7, and 8.
line	Specifies a line number to skip.

Example

The following commands skip lines 6 through 8 and line 17 in the `foo.v` file when reading in the design:

```
add nottranslated lines foo.v 6:8 17  
read des foo.v
```

The following shows the lines (in bold) that were skipped in the `foo.v` file:

```
(1) module test(clk, rst, set, d, z);  
(2) input clk, set, rst, d;  
(3) output z;  
(4) reg z;  
(5)  
(6)   garbage 1.....  
(7)   initial begin end  
(8)   garbage 2.....  
(9)  
(10)  always @(negedge clk or negedge set or negedge rst) begin  
(11)      if (!rst) z <= 1'b0;  
(12)      else if (!set) z <= 1'b1;
```

Conformal Extended Checks Reference Manual

Command Reference

```
(13)         else z <= d;  
(14)     end  
(15)  
(17)     garbage 3....  
(18)  
(19)endmodule
```

Related Commands

READ DESIGN

READ LIBRARY

ADD NOTRANSLATE MODULES

ADD NOTranslate Modules

```
<module_name* ...>  
[ | -Library | -Design]  
(Setup Mode)
```

Specifies library or design modules that are parsed but will not be translated when you execute the `READ LIBRARY` or `READ DESIGN` command. These modules automatically become blackboxes. *You must execute this command before you execute the `READ LIBRARY` and `READ DESIGN` commands.* Some examples of modules that should not be compiled include RAM models, analog blocks, and modules that contain non-synthesizable RTL constructs.

Tcl Command

```
add_notranslate_modules
```

Parameters

<code><module_name* ...></code>	Does not compile the listed library or design modules.
<code>-Library</code>	Does not compile the specified library modules. If you do not specify <code>-library</code> or <code>-design</code> , Conformal applies this command to both the library and design modules.
<code>-Design</code>	Does not compile the specified design modules. If you do not specify <code>-library</code> or <code>-design</code> , Conformal applies this command to both the library and design modules.

Related Commands

[ADD BLACK BOX](#)

[DELETE BLACK BOX](#)

[DELETE NOTRANSLATE MODULES](#)

Conformal Extended Checks Reference Manual

Command Reference

READ DESIGN

READ LIBRARY

REPORT BLACK BOX

REPORT NOTRANSLATE MODULES

ADD PIN CONSTRAINTS

ADD Pin Constraints

```
<0 | 1 | ONE_Hot | ONE_Cold | ZERO_ONE_Hot | ZERO_ONE_Cold>  
<primary_pin* ...>  
[-REPlace]  
[-ROot | -Module <module_name*> | -All]  
[-NORESET | -RESET]  
(Setup Mode)
```

Places constraints on input pins of the specified module. Place any of the following constraints on the input pins:

- 0-state
- 1-state
- One-hot
- One-cold
- Zero-one-hot
- Zero-one-cold
- Base 2, 8, 10, and 16 Verilog constants

The *one-hot* constraint lets only one pin be at a 1-state. The remaining pins must be at a 0-state.

The *one-cold* constraint lets only one pin to be at a 0-state. The remaining pins will be at a 1-state.

Tcl Command

```
add_pin_constraints
```

Parameters

0	Holds the specified primary input pins to a 0-state.
1	Holds the specified primary input pins to a 1-state.
ONE_Hot	Specifies that only one of the pins can have a high value.
ONE_Cold	Specifies that only one of the pins can have a low value.

Conformal Extended Checks Reference Manual

Command Reference

ZERO_ONE_Hot	All pins can have a low value, but only one can have a high value.
ZERO_ONE_Cold	All pins can have a high value, but only one can have a low value.
<primary_pin* ...>	Specifies a list of primary input names that will be constrained to a certain state. This accepts wildcards.
-REPlace	Changes the previously specified pin constraint.
-ROot	Applies the constraints to the root module. <i>This option is the default.</i>
-Module <module_name*>	Applies the constraints to the specified module. This accepts wildcards.
-All	Applies the constraints to all modules.
-NORESET	Does not label this constraint as a reset constraint. <i>This option is the default.</i> A reset constraint is a constraint that keeps the design in functional mode, that is, not in reset. Refer to ADD IGNORE RESET CONSTRAINT for more information.
-RESET	Labels this constraint as a reset constraint. A reset constraint is a constraint that keeps the design in functional mode, that is, not in reset. Refer to ADD IGNORE RESET CONSTRAINT for more information.

Related Commands

[DELETE PIN CONSTRAINTS](#)

[REPORT PIN CONSTRAINTS](#)

ADD PIN EQUIVALENCES

ADD PIn Equivalences

```
<primary_pin> <primary_pin* ...>  
[-Invert <primary_pin* ...> ]  
[-R0ot | -Module <module_name*> | -All]  
(Setup Mode)
```

Specifies input pin equivalences or inverted pin equivalences on module input pins. The first specified input pin is the representative pin. The subsequent primary input pins refer to the representative pin.

Tcl Command

add_pin_equivalences

Parameters

<primary_pin> <primary_pin* ...>

Lists the primary input pins that are equivalent. The first input pin is classified as the representative pin. This option accepts wildcards, multiple names, and buses.

You can also use this command on buses; in this case, define the bus range using <name [msb:lsb]>.

-Invert <primary_pin* ...>

Specifies that the named primary pins are inversions of each other. If you do not specify this option, the pins are defined as equivalent. This accepts wildcards.

Use the REPORT PIN EQUIVALENCES command to identify the inverted primary input pins. The (-) denotes inverted pins.

-R0ot

Applies pin equivalence to the root module. *This is the default.*

-Module <module_name*>

Applies pin equivalence to the specified module. This accepts wildcards.

-All

Applies pin equivalence to all modules in the design.

Related Commands

DELETE PIN EQUIVALENCES

REPORT PIN EQUIVALENCES

ADD PRIMARY INPUT

ADD Primary Input

<<pathname*> [-Net] | <pathname*> -Pin>>
(Setup Mode)

Adds a new primary input pin to a specified net or pin name. This new primary input pin is classified in the *User* class of primary inputs. (The original primary inputs of the design are classified in the *System* class of primary inputs.)

Tcl Command

add_primary_input

Parameters

<pathname*> -Net	Add the primary input to the specified net path. <i>The default option is -net.</i> This accepts wildcards.
<pathname> -Pin	Add the primary input to the specified pin path. This accepts wildcards.

Related Commands

DELETE PRIMARY INPUTS

REPORT PRIMARY INPUTS

REPORT PATH

ADD RENAMING RULE

ADD REnaming Rule

```
<rulename> <pattern> <substitution>
[-REPlace]
(Setup Mode)
```

Adds renaming rules. A *renaming rule* defines the pattern to be matched and the temporary substitution string. Renaming rules are applied in the same order in which they were created.

Renaming Rule Structure

When defining renaming rules, the first string specifies the pattern to be matched; the second string specifies how Conformal is to rename or make substitutions. The first string can contain expressions of the following types:

%d	Matches 1 or more digits, [0-9]+
%a	Matches 1 or more alphabetical characters, [a-zA-Z]+
%s	Matches 1 or more digits or alphabetical characters, [0-9a-zA-Z]+
x	Matches character "x"
abc	Matches string "abc"
.	Matches any single character
*	Matches 0 or more repetitions of the preceding expression
+	Matches 1 or more repetitions of the preceding expression
^bol	Matches "bol" only when it occurs at the beginning of a string
eol\$	Matches "eol" only when it occurs at the end of a string
x abc	Matches "x" or "abc"
[oz!]	Matches "o", "z", or "!"
(pattern)	Matches anything that is matched by "pattern"; you can then refer to it using @n in the substitution string.

Any character can be preceded by the escape character "\" to cancel any special meaning it has. Use the escape character whenever any of the following special characters represents a simple character.

% . * + ^ \$ | () [] \

Conformal Extended Checks Reference Manual

Command Reference

The second string can contain expressions of the following types:

<code>abc</code>	Replaces string "abc" for each matched string.
<code>@n</code>	Replaces the string that matches the <code>n</code> th <code>%d</code> , <code>%a</code> , <code>%s</code> , or a pattern enclosed in parentheses. The <code>n</code> is a digit other than 0, and you can use <code>@{nn}</code> to refer to further matches (that is, 10 ... 99).
<code>#(expr)</code>	Where "expr" is an arbitrary expression that can only contain constant integers, <code>@n</code> expressions, and the operators <code>+</code> , <code>-</code> , <code>*</code> , <code>/</code> , <code>%</code> , and <code>()</code> . Operator precedence does not apply when using expressions; use parenthesis to modify the evaluation order.

Testing Renaming Rules

You can use the `TEST RENAMING RULE` command to test your renaming rules.

test renaming rule `<original_name>`

where `original_name` indicates the object name for which you would like to apply the renaming rules. For example:

1. You created two renaming rules using these commands:

```
add renaming rule r1 "_reg_%d" "_reg[@1]"
add renaming rule r2 "/N01" "/Q"
```

2. You can use the `rn` command to test these rules against the "fsm_state_reg_2/N01" object name:

```
test renaming rule fsm_state_reg_2/N01
```

Conformal CD displays messages similar to the following:

```
Renaming rule: r1. Str: fsm_state_reg_2/N01
Result: fsm_state_reg[2]/N01
Renaming rule: r2. Str: fsm_state_reg[2]/N01
Result: fsm_state_reg[2]/Q
```

Tcl Command

`add_renaming_rule`

Parameters

<code><rulename></code>	Specifies a rule identification name assigned to a specific renaming rule.
<code><pattern></code>	Represents the pattern to be matched.
<code><substitution></code>	Represents the substitution pattern.
<code>-REPlace</code>	Redefines an existing renaming rule.



Tip

The difference between using `add renaming rule -replace` as opposed to using `delete renaming rule` followed by `add renaming rule` is that renaming rules that are redefined remain in the same position in the list of renaming rules. By deleting and adding a rule, the new definition will appear at the end of the list. In some cases, the order in which renaming rules are applied might affect the result.

Example

```
add renaming rule rule1 "/N01"  "/"Q"  
add renaming rule r1  "_reg_%d" "_reg[@1]"
```

Related Command

DELETE RENAMING RULE

ADD RESET_SYNC MODULE

```
ADD RReset_sync Module
    <reset_sync_module_name*> ...>
    (Verify Mode)
```

Declares modules as set or reset synchronizers.

While performing set and reset checks, the Conformal software traces the driver of the set and reset port. The driver can be one of the followings.

- Primary input
- Register
- Any gate with more than one effective inputs (ignoring constant inputs, and signals that belong to the universal domain).

For a valid set and reset synchronization, ensure the followings:

1. The domain of the driver should be the same as the domain of the register for set or reset check.
2. The driver should either belong to a user-defined set or reset synchronizer module, or should be an output of a generally accepted set or reset synchronizer chain (two back to back DFFs). In the user-defined case, the software does not check the functionality of the module itself.

Tcl Command

```
add_reset_sync_module
```

Parameters

```
<reset_sync_module_name> ...
```

Specifies the name, or names, of the module(s) to declare.

Related Commands

- DELETE RESET_SYNC MODULE
- REPORT RESET_SYNC MODULE

ADD RESET SOURCE

ADD REset Source
 <pathname> ...
 (*Verify Mode*)

Specifies hierarchical paths for the reset input signal for reset synchronization checks. Only these signals can derive the reset pin of the reset synchronizer.

When using this command, be aware of the following situations:

- Adding any reset source automatically enables the check for reset synchronizers, equivalent to running the `SET CDC OPTION -reset_source_check` command.
- Having no reset source does not mean that the `SET CDC OPTION` command's `-reset_source_check` option is disabled.

After adding reset sources, if you want to see the reset result without source check, you must disable the appropriate CDC option. You do not have to delete the reset sources.

- The software will still show a PASS status for the reset synchronizer flops even if their reset signal is not from the specified sources. But it will produce a FAIL status for the output flop whose reset signal is driven by the output of the synchronizer.

Note: You must run the `ADD CDC CHECK -reset` command to validate the reset synchronization.

Tcl Command

`add_reset_source`

Parameters

<pathname> ...	Specifies the hierarchical path, or paths, of the reset signal for reset synchronization checks.
----------------	--

Example

The following commands specify `rst1` as the input reset signal for reset synchronization checks, then adds a check for asynchronous reset pins of the flip-flops, and verifies clock domain crossing with the `VALIDATE` command.

```
add reset source rst1
```

Conformal Extended Checks Reference Manual

Command Reference

add cdc check -reset
validate

Related Commands

- DELETE RESET SOURCE
- REPORT RESET SOURCE

ADD RULE FILTER

ADD Rule Filter

```
<filter_name> [<condition ...>]
[-DESCRiption <description>]
[-FILTER <filter_expression> [-REGEXP]]
[-RULE <rule_name*>] ...
[-REPlace]
(Setup Mode)
```

Creates and modifies report rule filters.

Tcl Command

add_rule_filter

Parameters

<filter_name>	Specifies the name of the rule filter to add or apply. If <condition> is not specified, the rule filter must already exist.
<condition> ...	Specifies the condition used to determine if a rule occurrence should be filtered or not. If the condition is true, the rule occurrence will be filtered. A condition is required when adding a new rule filter.

For <condition>, specify one or more of the following filter conditions:

-RULE <rulename*>

Filters out all occurrences of specified rule(s).

-SEVerity <Error | Warning | Note | Ignore>

Filters out all occurrences of specified severity levels.

-MESsage <pattern*>

Conformal Extended Checks Reference Manual

Command Reference

Matches rule occurrences whose verbose message matches the specified pattern. For the condition `-message <pattern>` to be true, the pattern must match the entire verbose message of the rule occurrence.

For example, if the pattern is based on some words in the middle of the message, the pattern should start and end with the asterisk (`'*'`) wildcard character.

Some characters (such as, `'*'`, `'?'`, `'\'`, `'['`, and `']'`) have a special meaning in wildcard patterns. To match these characters literally, escape them with a backslash (`'\'`). For example:

```
add rule filter rule1 -RULE CPF_ISO1 -AND -MESSAGE \
    '*from ff1/out_f_reg\[0\]/Q*'
```

`-Object [-Hierarchical] [-Recursive] <name*>`

Matches rule occurrences related to objects that match the specified pattern.

This condition is only available for SDC rules.

`-Hierarchical` matches any instance in the design hierarchy against the specified pattern (analogous to the SDC command `'get_pins -hier <pinname>'`)

`-Recursive` matches any object under an instance that matches the specified pattern (analogous to the Unix command `'grep -r ... <dirname>'`)

`-Hierarchical` and `-Recursive` can be used together.

`-SDCMatch <pattern*>`

Matches rule occurrences related to SDC statements matching the specified pattern. Note: the string representing the SDC statement is the one displayed in the SDC command browser, not the one from the SDC file.

This condition is only available for SDC rules.

`-NOT <condition>`

Matches rule occurrences not matched by the sub-condition.

`<condition> [-AND] <condition>`

Matches rule occurrences matched by both sub-conditions.

`<condition> -OR <condition>`

Conformal Extended Checks Reference Manual

Command Reference

Matches rule occurrences matched by either sub-condition.

(<condition>)

Conditions can be grouped using parentheses to enforce precedence. White space is required around the parentheses.

-DESCRiption <description>

Specifies a description for the added rule filter.

-FILTER <filter_expression>

The rule filter will apply to occurrences for which the specified `filter_expression` evaluates to true.

The syntax of the `filter_expression` is the same as for the `find Tcl` command.

-REGEXp

Specifies that the pattern matching operators used inside `filter_expression` (such as `=~`) should use regular expressions as patterns, rather than glob-style patterns.

-RULE <rule_name*>

Specifies the rule occurrences to which the filter should be applied. By default, `-rule` is part of the condition. In the native 1801 flow (set `lowpower option -native_1801`), `-rule` is not part of the condition and must be specified to apply the filter. Without the `-rule` option in the native 1801 flow, the filter will not have any effect.

This accepts wildcards.

-REPlace

Specifies that the filter name can be that of an existing filter, which is then modified.

-INClude

Specifies that the filter will cause any matching occurrence not to be filtered out, unless this is reversed by another filter down the list.

Examples

Example 1:

The following adds a filter called `f1` that filters out occurrences where the message contains the string `"special_cell"`:

```
add rule filter f1 -message "*special_cell"
```

Conformal Extended Checks Reference Manual

Command Reference

The following adds a filter called `f2` that filters out occurrences of `PDM4d` where the message contains the string `"special_cell"`:

```
add rule filter f2 -rule PDM4d -AND -message "*special_cell"
```

Example 2:

The following adds a filter called `f3` that filters out occurrences of `CROSSING_OFF_TO_ON_NOLP` rule where the attribute contains `driver_supply_set == SS_AREA0`:

```
add_rule_filter f3 -rule CROSSING_OFF_TO_ON_NOLP -filter "driver_supply_set == SS_AREA0"
```

The user can find the checker attribute by using the command `"report_rule_check <rule name> -attribute"`.

The following adds a filter called `f4` that filters out occurrences of `1801_ISO_CLAMP_VALUE_CONFLICT` rule where the attribute contains `strategy == PD_BLOCK2.iso_rule1`, `element == inst2/d[0]`, and `clamp_value == 0`:

```
add_rule_filter f4 -rule 1801_ISO_CLAMP_VALUE_CONFLICT -filter  
"(strategy==PD_BLOCK2.iso_rule1) && (element==inst2/d\[0\]) && (clamp_value==0)"
```

Related Commands

[DELETE RULE FILTER](#)

[READ RULE CHECK](#)

[REPORT RULE FILTER](#)

[WRITE RULE CHECK](#)

ADD RULE WAIVER

ADD Rule Waiver

```
<rulename* <[<occurrence#>] [ ' .. ' ] [<occurrence#>] > ... > ...  
(Setup Mode)
```

Marks the specified occurrences as waived.

Note: This mark is lost when the rule is re-checked.

Tcl Command

add_rule_waiver

Parameters

<rulename*>	Specifies the rule(s). This accepts wildcards.
<occurrence#>	Specifies the number of occurrences to apply the waiver.
'..'	Specifies a range of occurrence numbers. All occurrences between the two numbers surrounding this argument will be affected. If no number precedes this argument, the range begins with the first occurrence. If no number succeeds this argument, the range extends to the last occurrence. Note: Spaces are required between the numbers and ' .. '. See example.

Example

The following command will waive all but the first and fifth occurrences of rule CCD_EXC_OLP1:

```
add rule waiver CCD_EXC_OLP1 2 .. 4 6 ..
```

Related Commands

DELETE RULE WAIVER

READ RULE CHECK

Conformal Extended Checks Reference Manual

Command Reference

WRITE RULE CHECK

ADD SEARCH PATH

ADD SEarch Path

```
<pathname ...>  
[ |-Design | -Library | -POWER_intent]  
(Setup Mode)
```

Defines additional search paths outside the current directory for filenames you use in the READ DESIGN and READ LIBRARY commands. This command is necessary because the default is to search for filenames in the current directory; but your design or library can include filenames that are housed in other directories. The default search path is the current working directory.

When you add multiple search paths to the list, Conformal does the search in the order paths were added to the list.

Use the REPORT SEARCH PATH command to display all search paths. Use the tilde character (~) to shorten the specified path.

Notes:

- By default, Conformal begins the search for files in the current directory. Use the DELETE SEARCH PATH command to remove the current directory from the search path.
- Define search paths relative to the current directory. See the following *Relative* search path examples:

```
lib/testlib  
./vsrc
```
- Specify search paths using full paths. See the following *Full* search path examples:

```
/usr/home/jones/design1/lib  
~/lib/testlib
```
- Specify multiple paths, separated by spaces. See the following multiple path example:

```
./vsrc ~smith/design1/vsrc /home/design1/vsrc
```
- If you use the ADD SEARCH PATH command repeatedly, the paths are added to the end of the current search path list.

Tcl Command

```
add_search_path
```

Parameters

<pathname ...>	Specifies a list of paths for directories that you want Conformal to search.
-Design	<p>The <code>READ DESIGN</code> command uses the specified search path.</p> <p>If you do not specify <code>-library</code> or <code>-design</code>, Conformal applies this command to both the <code>READ DESIGN</code> and <code>READ LIBRARY</code> commands.</p>
-Library	<p>The <code>READ LIBRARY</code> command uses the specified search path.</p> <p>If you do not specify <code>-library</code> or <code>-design</code>, Conformal applies this command to both the <code>READ DESIGN</code> and <code>READ LIBRARY</code> commands.</p>
-POWER_intent	This is a low power command option. Specifies the search path for power intent files.

Examples

```
add search path /home/library/txelib -library
add search path ~/txmain/src primitives src -design
add search path ~lib/hflib ~lib/lflib ~lib/prim -library
```

Related Commands

[DELETE SEARCH PATH](#)

[READ DESIGN](#)

[READ LIBRARY](#)

[REPORT SEARCH PATH](#)

ADD SET SOURCE

ADD SET Source
 <pathname> ...
 (*Verify Mode*)

Specifies hierarchical paths for the set input signal for set synchronization checks. Only these signals can derive the set pin of the set synchronizer.

When using this command, be aware of the following situations:

- Adding any set source automatically enables the check for set synchronizers, equivalent to running the `SET CDC OPTION -set_source_check` command.
- Having no set source does not mean that the `SET CDC OPTION` command's `-set_source_check` option is disabled.

After adding set sources, if you want to see the set result without source check, you must disable the appropriate CDC option. You do not have to delete the set sources.

- The software will still show a PASS status for the set synchronizer flops even if their set signal is not from the specified sources. But it will produce a FAIL status for the output flop whose set signal is driven by the output of the synchronizer.

Note: You must run the `ADD CDC CHECK -set` command to validate the reset synchronization.

Tcl Command

`add_set_source`

Parameters

<pathname> ...	Specifies the hierarchical path, or paths, of the set signal for set synchronization checks.
----------------	--

Example

The following commands specify `set1` as the input set signal for set synchronization checks, then adds a check for asynchronous set pins of the flip-flops, and verifies clock domain crossing with the `VALIDATE` command.

```
add set source set1
```

Conformal Extended Checks Reference Manual

Command Reference

```
add cdc check -set  
validate
```

Related Commands

- DELETE SET SOURCE
- REPORT SET SOURCE

ADD STATIC PROPERTY

ADD Static Property

```
[  -ALL
|< -TRI_state      [-PO | -ALL |<id | pathname*> ...]
|  -DONT_CARE
|    [ -ALL          | -X_ASSIGNment  | -RANGE_overflow
|      | -FULL_case  | -PARAllel_case |<id | pathname*>...
|    ]
|  -BRANCH_enable
|    [ -ALL | -IF_else  | -CASE
|      | -DEFAULT | -FOR_loop |<id | pathname*>...
|    ]
|  <-BUS | -MULTI_PORT
|  -SET_RESET
|  -FSM | -FSMReachability
|  -FSMTransition | -FSMDeadlock
|  -NO_DIVide_by_zero
|  -ENCoding_completeness
|  >
|  [-ALL |<id | pathname*>...]
|  >...
]
```

(Verify Mode)

Adds specified extracted properties to the "Prove" list. When you choose the Verify mode, Conformal automatically extracts all possible predefined properties. The correct sequence of commands is as follows:

1. Run the `ADD IGNORED PROPERTY` command to exclude specified properties from the "Prove" list.
2. Run the `ADD STATIC PROPERTY` command to add properties to the "Prove" list.
3. Run the `PROVE` command. At this point, Conformal Extended Checks proves all the properties included on the "Prove" list.

When you use the `ADD IGNORED PROPERTY` command to exclude properties from the "Prove" list, you cannot add these same properties with the `ADD STATIC PROPERTY` command until you release them from the "Ignored Property" list. Use the `DELETE IGNORED PROPERTY` command to reinstate ignored properties.

After you add properties, use the `REPORT STATIC PROPERTY` command to review your additions. And use the `DELETE STATIC PROPERTY` command to delete properties from the "Prove" list.

Note: You can execute the `ADD STATIC PROPERTY` command repeatedly to add individual, groups, or all properties.

Tcl Command

`add_static_property`

Parameters

<code>-ALL</code>	Adds all properties. <i>This option is the default.</i>
<code>-TRI_state</code>	Adds tristate properties, as specified.
<code>-PO</code>	Adds tristate properties connected to the primary output.
<code>-ALL</code>	Adds all tristate properties.
<code><id pathname*></code>	Adds tristate properties for pins with the specified IDs or paths. This accepts wildcards. ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
<code>-DONT_CARE</code>	Adds Don't Care properties, as specified.
<code>-ALL</code>	Adds all Don't Care properties. <i>This option is the default.</i>
<code>-X_ASSIGNment</code>	Adds only Don't Care properties for X-assignment cases.
<code>-RANGE_overflow</code>	Adds only Don't Care properties for array index overflow cases.
<code>-FULL_case</code>	Adds only Don't Care properties for <code>full_case</code> synthesis directive cases.
<code>-PARAllel_case</code>	Adds only Don't Care properties for <code>parallel_case</code> synthesis directive cases.

Conformal Extended Checks Reference Manual

Command Reference

<id | pathname*>

Adds the Don't Care properties for the specified IDs or paths. This accepts wildcards.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BRANCH_enable

Add Branch Enable properties, as specified.

-ALL Adds all Branch Enable properties. That is, check whether case and if-else conditional branches can be enabled.
This option is the default.

-IF_else Adds Branch Enable properties for if-else conditional branches. That is, check whether if-else conditional branches can be enabled.

-CASE Adds Branch Enable properties for case conditional branches. That is, check whether case conditional branches can be enabled. (This option does not include default branches.)

-DEFAULT Adds Branch Enable properties for case-default conditional branches. That is, check whether case-default conditional branches can be enabled.

-FOR_loop Adds Branch Enable `for_loop` properties. That is, check whether `for_loops` can be executed.

<id | pathname*>

Conformal Extended Checks Reference Manual

Command Reference

Adds the specified Branch Enable properties. That is, check whether the specified Branch Enable properties can be enabled.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BUS	Adds Bus properties, as specified.
-MULTI_PORT	Adds Multi-port properties, as specified.
-SET_RESET	Adds Set-Reset properties, as specified.
-FSM	Adds all finite state machine properties, as specified.
-FSMReachability	Adds finite state machine Reachability properties, as specified.
-FSMTransition	Adds finite state machine Transition properties, as specified.
-FSMDeadlock	Adds finite state machine Deadlock properties, as specified.
-NO_DIVide_by_zero	Adds No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
-ENCoding_completeness	Adds Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
-ALL	<p>Adds all properties, as specified. For example, the following command adds all Bus properties to the "Prove" list:</p> <pre>add static property -bus -all</pre> <p><i>This option is the default.</i></p>

Conformal Extended Checks Reference Manual

Command Reference

<id | pathname*>

Adds the specified properties with the specified ID or hierarchical path. This accepts wildcards.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Examples

For a set of sample commands that shows this and related commands in context, see the example for the EXPLORE command.

Related Commands

ADD IGNORED PROPERTY

DELETE IGNORED PROPERTY

DELETE STATIC PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

REPORT EXTRACTED PROPERTY

REPORT FSM

REPORT GATE

REPORT IGNORED PROPERTY

REPORT PROVED DATA

REPORT STATIC PROPERTY

ADD SYNC INPUT

ADD Sync Input
 <pathname> ...
 (Verify Mode)

Specifies that some inputs of the current design are properly synchronized outside of the design while performing set or reset checks.

Tcl Command

add_sync_input

Parameters

<pathname> ...	Specifies the name of path(s). Paths specify the data signal which are assumed to be properly synchronized outside the design.
----------------	--

Example

The following command sequence specifies that input port `rstn` is already synchronized appropriately external to the design, then associates this signal with the proper synchronous clock domain, and specifies that only sets or resets in other domains will be flagged as failing:

```
add sync input rstn
add data association rstn -domain clkb
add cdc check -reset
validate
```

`rstn` goes to both `clka` and `clkb` resets in this design, resulting in the following summary:

```
=====
= Reset Synchronization Validated Data (Summary)
=====
```

Clock Domain	Pass	Fail	Total
clkb	10	0	10
clka	0	9	9
Total	10	9	19

Conformal Extended Checks Reference Manual

Command Reference

=====

Related Commands

DELETE SYNC INPUT

REPORT SYNC INPUT

ADD SYNCHRONIZATION RULE

ADD Synchronization Rule

```
<rule_name> <-SYNC_MODULE <module_name ...>
|-DFF <min> [INFinity |<max>]
  [-FIRST_DFF | -FIRST_DLAT_OR_DFF]
  [-SYNC_CHAIN [BUFFer | WIRe | LOGic]
  [MULTiple_chain | SINGle_chain] ]
  |-MUX <min> <max> [-HOLD | -NOHOLD] >
  [-CDC_PATH [LOGic | WIRe | BUFFer]
  [MULTiple_destination | SINGle_destination] ]
  [-SOURCE <-ALL | <clock_domain*>...>]
  [-DESTINATION <-ALL | <clock_domain*>...>]
  [-FROM <-ALL | -REG | -PI | -BBOX | instance*...>]
  [-TO <-ALL | -REG | -PO | -BBOX | instance*...>]
  [<-MODULE | -INSTance> <name*...>]
  [-RECURSive | -NORECURSive]]
(Verify Mode)
```

Specifies the kind of synchronizers that are used in specified portions of the design.

Use the `-mux` option to identify data paths in a design. The meanings of some typical parameter values are:

- `-mux 0 0`: The multiplexer synchronizer is located in the CDC path.
- `-mux 1 1`: The multiplexer synchronizer is located in the sync chain, that is, after 1 destination D flip-flop.
- `-mux 0 1`: The multiplexer synchronizers are located in the CDC path, the sync chain, or both.

Tcl Command

`add_synchronization_rule`

Parameters

`rule_name` Specifies an identification name assigned to a specific synchronization rule.

`-SYNC_MODULE module_name...`

Specifies that modules are used as synchronizers.

This option specifies that Conformal Extended Checks will first check to ensure that a domain crossing path's destination D flip-flop is inside the specified module. If this condition is satisfied, Conformal Extended Checks runs a check on the `cdc_path` to ensure that logic outside the module boundary complies with all other parameters of the synchronization rule.

Example:

```
add sync rule r1 -sync_module m1 -cdc_path buffer
```

For this example, the CDC path check passes if the CDC path satisfies both of the following conditions:

1. The destination D flip-flop is inside module `m1`.
2. Only the buffer or inverter is in the `cdc_path` outside the module boundary.

-DFF

Specifies that D flip-flops are used as a synchronizer.

min

Specifies the minimum number of flip-flops in the DFF synchronizer. (This `min` is an integer value: 1 or greater.)

INFINITY

Specifies the maximum number of flip-flops in the DFF synchronizer.

If you do not set the `max` parameter, the default is infinity.

max

Specifies the maximum number of flip-flops in the DFF synchronizer. (This `max` is an integer value that is greater than or equal to `min`.)

-FIRST_DFF

Specifies that the first state element in the DFF synchronizer must be a D flip-flop (that is, a D-latch is not allowed as the first element in the DFF synchronizer). *This option is the default.*

-FIRST_DLAT_OR_DFF

Specifies that the first state element in the DFF synchronizer can be either a D flip-flop or D-latch.

Conformal Extended Checks Reference Manual

Command Reference

-SYNC_CHAIN		Identifies what is allowed between the D flip-flops (or D-latch and D flip-flops) of the DFF synchronizer.
		If you do not specify this option, <code>buffer</code> is the default, which means that only wire, buffers, and inverters are allowed between the state elements of the DFF synchronizer. Additionally, multiple fan-outs are allowed from within the DFF synchronizer chain.
	BUFFer	Allows only wire, buffers, and inverters between the state elements of the DFF synchronizer.
	WIRe	Allows only wire between the state elements of the DFF synchronizer.
	LOGic	Allows any logic gate between the state elements of the DFF synchronizer.
	MULTiple_chain	Allows multiple fan-out from within the DFF synchronizer chain.
	SINGle_chain	Does not allow fan-out from within the DFF synchronizer chain.
-MUX	Multiplexers are used as a synchronizer.	
	min	Specifies the minimum number of destination D flip-flops before the multiplexer.
	max	Specifies the maximum number of destination D flip-flops before the multiplexer.
	-HOLD	Specifies that the data hold condition must exist for the MUX synchronizer. <i>This option is the default.</i> The data hold condition describes a feedback loop that starts from the output of the destination register and goes back to the input of the MUX synchronizer.

Conformal Extended Checks Reference Manual

Command Reference

	-NOHOLD	Specifies that the data hold condition does not have to exist for the MUX synchronizer. The data hold condition describes a feedback loop that starts from the output of the destination register and goes back to the input of the MUX synchronizer.										
-CDC_PATH		Identifies what is allowed in the CDC path. The CDC path definition is different for different types of synchronizers: For DFF synchronizers, the CDC path is the signal path between source flip-flop and the destination flip-flop. For the MUX synchronization scheme, the CDC path is the signal path between the source flip-flop and destination flip-flop, excluding the MUX synchronizer element. For Sync module-based synchronizers, the CDC path is the signal path between source flip-flop and destination flip-flop, excluding the gates inside the specified Sync module. The defaults for this option are <code>logic</code> and <code>multiple_destination</code> , which means that Conformal Extended Checks supports any logic and multiple fan-outs after the source domain. <table><tr><td>LOGic</td><td>Allows any logic in the CDC path. Note: Logic is restricted when driven by multiple clock domains.</td></tr><tr><td>WIRe</td><td>Allows only wire in the CDC path.</td></tr><tr><td>BUFfer</td><td>Allows only wire, buffer, and inverters in the CDC path.</td></tr><tr><td>MULtiple_destination</td><td>Allows multiple fan-outs from the CDC.</td></tr><tr><td>SINggle_destination</td><td>Does not allow any fan-out from the CDC path.</td></tr></table>	LOGic	Allows any logic in the CDC path. Note: Logic is restricted when driven by multiple clock domains.	WIRe	Allows only wire in the CDC path.	BUFfer	Allows only wire, buffer, and inverters in the CDC path.	MULtiple_destination	Allows multiple fan-outs from the CDC.	SINggle_destination	Does not allow any fan-out from the CDC path.
LOGic	Allows any logic in the CDC path. Note: Logic is restricted when driven by multiple clock domains.											
WIRe	Allows only wire in the CDC path.											
BUFfer	Allows only wire, buffer, and inverters in the CDC path.											
MULtiple_destination	Allows multiple fan-outs from the CDC.											
SINggle_destination	Does not allow any fan-out from the CDC path.											
-SOURce		Applies the synchronization rule to the paths with the specified source clock domain.										

Conformal Extended Checks Reference Manual

Command Reference

	-ALL	Allows the synchronization rule to all source clock domains.
	<clock_domain*>	Applies the synchronization rule to the paths with the specified source clock domain(s). This accepts wildcards.
-DESTination		Applies the synchronization rule to the paths with the specified destination domain.
	-ALL	Applies the synchronization rule to all destination clock domains.
	<clock_domain*>	Applies the synchronization rule to the paths with the specified destination clock domain(s). This accepts wildcards.
-FROM		Applies the synchronization rule to the paths that start from the specified key points. The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-ALL	Applies the synchronization rule to the paths that start from all the allowed key points. The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-REG	Applies the synchronization rule to the paths that start from registers.
	-PI	Applies the synchronization rule to the paths that start from primary inputs.
	-BBOx	Applies the synchronization rule to the paths that start from blackbox output pins.
	<instance*>	Applies the synchronization rule to the paths that start from the specified instance(s). The instances can be a register, primary input, or blackbox output. This accepts wildcards.

Conformal Extended Checks Reference Manual

Command Reference

-TO	Applies the synchronization rule to the paths that end at the specified key points. The type of key points includes registers, primary outputs, and blackboxes (inputs).
-ALL	Applies the synchronization rule to the paths that end at all the allowed key points. The type of key points includes registers, primary outputs, and blackboxes (inputs).
-REG	Applies the synchronization rule to the paths that end at registers.
-PO	Applies the synchronization rule to the paths that end at primary outputs.
-BBOX	Applies the synchronization rule to the paths that end at blackbox input pins.
<instance*>	Applies the synchronization rule to the paths that end at the specified instance(s). This accepts wildcards. The instances can be a register, primary output, or blackbox input.
-MODULE INSTANCE <name* ...>	Applies this synchronization rule to the specified modules or module instance path names.
-RECURSIVE	Includes submodules.
-NORECURSIVE	Does not include submodules.

Related Commands

DELETE SYNCHRONIZATION RULE

PROPAGATE CLOCK DOMAIN

REPORT SYNCHRONIZATION RULE

SET PREDEFINED SYNCH_RULE

VALIDATE

ADD TIED SIGNALS

ADD Tied Signals

```
<0 | 1>  
<name* ...>  
[-Net |-Pin]  
[-ROot |-Module <module_name*> | -All]  
[ | -Design | -Library]  
(Setup Mode)
```

Assigns the specified floating nets or pins to a 0-state or a 1-state in the design. These tied signals are classified in the *User* class of tied signals. Tied signals included in the *original* design are classified in the *System* class of tied signals.

Tcl Command

add_tied_signals

Parameters

0	Ties the floating nets or pins to a 0-state.
1	Ties the floating nets or pins to a 1-state.
<name* ...>	Specifies a list of names that correspond to either only floating net names or only floating pin names where you intend to add the tied signal. This accepts wildcards.
-Net	Ties off an internal net, which must be undriven. <i>This option is the default.</i>
-Pin	Ties off an input pin.
-ROot	Specifies that the floating net or pin resides in the current root module. <i>This option is the default.</i>
-Module <module_name*>	Specifies that the floating net or pin resides in the specified module. This accepts wildcards.
-All	Applies the tied signals to all modules.

Conformal Extended Checks Reference Manual

Command Reference

-Design	Applies the tied signals to the design. If you do not specify <code>-design</code> or <code>-library</code> , Conformal applies tied signals to both designs and libraries.
-Library	Applies the tied signals to the library. If you do not specify <code>-design</code> or <code>-library</code> , Conformal applies tied signals to both designs and libraries.

Related Commands

DELETE TIED SIGNALS

REPORT TIED SIGNALS

ADD VECTOR DEFINITION

ADD V_{ector} Definition

```
<vectorname> <id | pathname*> ...  
(Verify Mode)
```

Adds vector definitions and reports the number of cells that belong to the new vector. All the matched cells must be of the same class. The vector name cannot be the name of an existing cell from one of these supported classes.

- PI/O/IO
- DFF
- DLAT
- BBOX-I/O/IO

Note: You cannot specify a name that would be extracted as a vector by the automatic word-level CDC extraction algorithm. For example, if there is a cell called `a[1]`, then the name `a[]` is not available for vector definition.

Note: This command requires new extraction of word-level CDC checks. After issuing it, you must re-add any existing functional CDC checks.

Tcl Command

```
add_vector_definition
```

Parameters

<code><vectorname></code>	Specifies the name of the vector.
<code><id pathname*></code>	Specifies that the new vector will include all the cells that match the ID or pathname.

Related Commands

- DELETE VECTOR DEFINITION
- REPORT VECTOR DEFINITION

ASSIGN PIN DIRECTION

ASSign PIn Direction

```
<IN | OUT | IO>
<module_name*>
<pin_name*>
[-FROM_Dir <IN | OUT | IO>]
(Setup Mode)
```

Assigns an input, output, or I/O direction to a pin in a module. This command is especially useful when a SPICE netlist is read in, since all module ports will be treated as I/O pins. Conformal determines pin directions automatically. In cases where Conformal fails to determine the direction of a pin, this command specifies the proper pin direction.

Note: Conformal supports wildcard matching for pin names.

Refer to the *Conformal Equivalence Checking Reference Manual* and *Conformal Equivalence Checking User Guide* for additional information about this command.

Tcl Command

```
assign_pin_direction
```

Parameters

IN	Specifies that the assigned pin direction is input.
OUT	Specifies that the assigned pin direction is output.
IO	Specifies that the assigned pin direction is I/O.
<module_name*>	Specifies that the pin resides in the specified module. To specify all modules, use *.
<pin_name*>	Assigns a direction to the specified pin.
-FROM_Dir	Specifies that only pins which have the direction specified by this option's argument will be redirected.

Examples

```
assign pin direction in mux2p sela
assign pin direction out mux4p y
```

Conformal Extended Checks Reference Manual

Command Reference

```
//Assigns direction to pins sela and y.  
assign pin direction IN mod pin_* -from_type IO  
//Changes all IO pins whose name matches 'pin_*' in
```

Related Commands

ASSIGN PIN DIRECTION

REPORT PIN DIRECTION

BREAK

BREa**k**

(Setup / Verify Mode)

Interrupts script execution and returns to the `SETUP` or `VERIFY` command prompt when running a dofile script. This has no effect if you type this command at the command prompt.

Tcl Command

`break`

Related Commands

`CONTINUE`

`DOFILE`

`SAVE DOFILE`

`SET DOFILE ABORT`

CHECKPOINT

CHECKPOINT

```
<checkpoint_file_name>  
[-CONFIG [GZIP | BZIP2 | LRZIP]]  
[-protect <password>]  
[-REPlace]  
(Setup/Verify Mode)
```

Creates a *checkpoint* file. A checkpoint file contains a complete snapshot of the Conformal run up until the CHECKPOINT command is issued. The tool preserves the entire memory contents of the Conformal run. The checkpoint file includes:

- Hierarchical and flattened databases
- Environment settings
- Constraints
- Verification results
- User-defined variables
- User-defined procedures

To restart a session using a checkpoint file, invoke Conformal using the `-RESTART_CHECKPOINT <checkpoint_file_name> [-protect <password>]` option. For example:

```
lec -restart_checkpoint mycpfile -protect mypass
```

Review the set of limitations at the end of this section before using the CHECKPOINT command.

For more information on the checkpoint and restart facility, refer to "[Checkpoint and Restart Facility](#)" in the *Conformal Constraint Designer User Guide*.

Tcl Command

checkpoint

Parameters

checkpoint_file_name Name of the checkpoint file.

Conformal Extended Checks Reference Manual

Command Reference

<code>-CONFIG</code>	Enables checkpoint file to be compressed using one of these three formats GZIP, BZIP2 or LRZIP. The default format is GZIP. Note: User must check if their environment has one of these binary paths available by "which gzip/bzip2/lrzip" before generating the checkpoint file.
<code>-protect <password></code>	Encrypts the checkpoint file with the specified password. The password cannot lead with a dash ("-"); for example, <code>-mypassword</code> is not allowed.
<code>-REPlace</code>	Removes any existing file of the same name and replaces them with the specified file.

Limitations

- Checkpoint and restarts works on only Linux, and only on the following platforms:
32/64-bit Linux kernel versions 2.6.9-34, 2.6.9-42, 2.6.9-55, 2.6.9-67, 2.6.9-78, 2.6.9-89, 2.6.10, 2.6.14, 2.6.16, 2.6.18, 2.6.25, 2.6.26, 2.6.27, 2.6.32, 3.0.101, 3.0.13, 3.10.0-957, 3.10.0-1062, 4.4.21-69, 4.18.0-147, 4.18.0-193, 4.18.0-240, 4.18.0-305, 4.18.0-348, 4.18.0-372, 5.3.18-24, and 5.14.21-150400.
- Checkpoint files should be created and restarted using the same machine.

The tool issues warnings when you try to restart a checkpoint file from a different machine. The warning message can be ignored if the restart is always successful. For example, in a homogenous virtual machine computation environment, the host name might be different, but the OS version and configuration are the same. Thus, the warning message can be ignored.

If you are creating a checkpoint file that you plan to restart using a different license server, add the restart license server to the `LM_LICENSE_FILE` variable before invoking Conformal and before creating the checkpoint file; otherwise, you will not be able to restart the checkpoint file with the new server. For example:

```
setenv LM_LICENSE_FILE "$LM_LICENSE_FILE":5280@mylic01
```

- Do not enter the GUI mode if you plan to create a checkpoint file that you will want to run later in the GUI mode.

If a checkpoint file is created after having entered GUI mode, when the checkpoint file is restarted, it will restart and run in non-GUI mode and the GUI mode is disabled. If a checkpoint file is created before entering the GUI mode, the checkpoint file can enter the GUI mode when it is restarted.

Conformal Extended Checks Reference Manual

Command Reference

The Conformal `-restart_checkpoint` option tries to restore the process to the same status as when it was checkpointed. If a file cannot be re-opened during restart (permission problems, for instance), Conformal issues an error and opens it at `/dev/null` with only read permissions. You can restart again in the correct directory, or ask the file owner to change the permissions and try restarting the checkpoint file.

- You cannot specify the stack limit in a restarted tool process. You can, however, specify the stack limit when you issue the `CHECKPOINT` command:

```
CHECKPOINT -stack <multiplier>
```

Default multiplier is 1 (in other words, 64MB).

Related Commands

INFO CHECKPOINT

```
<Conformal_tool> -restart_checkpoint [-protect <password>]
```

CLOSE SCHEMATICS

CLOse SCheMatics
(*Setup / Verify Mode*)

Closes the schematic viewer.

Note: This command is only used in the GUI mode.

Tcl Command

`close_schematics`

Related Commands

[LICENSE](#)

[OPEN SCHEMATICS](#)

CONTINUE

CONTinue
(Setup / Verify Mode)

Use this command in conjunction with the `BREAK` command in a dofile. When a dofile executes the `BREAK` command, Conformal issues a warning and prompts you to use the `CONTINUE` command. The `CONTINUE` command has no effect if you type it without being prompted by Conformal.

The `CONTINUE` command supports mixed GUI and non-GUI mode. For example, you can run a dofile in non-GUI mode, encounter a break in the dofile, issue `set gui on`, and run `continue` from the GUI. The same applies when you break in the GUI mode: switch to command mode and type `continue`.

Conformal also supports nested breaks inside dofiles. This command works in a stack fashion. For example, when you execute `continue` from a lower-level dofile, Conformal proceeds until it encounters the next `BREAK` command.

Tcl Command

`continue`

Example

```
//Warning: Break dofile 'my_dofile' at line 32. Use 'continue' command to continue.  
% continue
```

Related Commands

[BREAK](#)

[DOFILE](#)

[SAVE DOFILE](#)

[SET DOFILE ABORT](#)

DELETE ALIAS

DELeTe ALias

<alias_name* ...>
(Setup / Verify Mode)

Deletes any aliases created with the ADD ALIAS command. Use the REPORT ALIAS command to display a list of all current aliases.

Tcl Command

delete_alias

Parameters

<alias_name* ...> Specifies the alias(es) to delete. This accepts wildcards.

Related Commands

ADD ALIAS

REPORT ALIAS

DELETE ASSERTION CONSTRAINTS

DELeTe ASsertion Constraints

<-ALL | <pathname*> ...>
(Setup Mode)

Deletes the assertion constraints that were added with the `ADD ASSERTION CONSTRAINTS` command. Delete all assertion constraints or individual assertion constraints, as specified by paths.

To recall the assertion constraints you have added, use the `REPORT ASSERTION CONSTRAINTS` command. This report displays the ID and the path of the assertion constraints.

Tcl Command

`delete_assertion_constraints`

Parameters

<code>-ALL</code>	Deletes all assertion constraints.
<code><pathname*> ...</code>	Deletes assertion constraints with the specified hierarchical paths. This accepts wildcards.

Related Commands

[ADD ASSERTION CONSTRAINTS](#)

[ADD STATIC PROPERTY](#)

[DELETE STATIC PROPERTY](#)

[REPORT ASSERTION CONSTRAINTS](#)

[REPORT STATIC PROPERTY](#)

DELETE BLACK BOX

DELEte Black Box

```
<blackbox_name* ...> [-Module] | <blackbox_name* ...> -Instance | -All>  
(Setup Mode)
```

Deletes specified blackboxes from the design. These blackboxes were either created with the ADD BLACK BOX command or were a part of the original design.

Use the REPORT BLACK BOX command to display a list of all the blackboxes.

Tcl Command

```
delete_black_box
```

Parameters

```
<blackbox_name* ...> -Module
```

Deletes blackboxes specified by this list. *The default option is -module.* This accepts wildcards.

```
<blackbox_name* ...> -Instance
```

The specified blackbox names are instance names. This accepts wildcards.

```
-All
```

Deletes all defined blackboxes.

Related Commands

ADD BLACK BOX

ADD NOTRANSLATE MODULES

DELETE NOTRANSLATE MODULES

REPORT BLACK BOX

REPORT NOTRANSLATE MODULES

DELETE CDC CHECK

DELEte CDc Check

```
[ <-ALL | -STRUctural
  | -FUNctional [SOURCE_DATA] [DESTINATION_DATA]
  [MUX_enable] [SINGLE_bit_change]
  | -STATus <<PASS | FAIL> ..> [-SYNC_rule <rule> ...] >
[-SOURce      <-ALL | <clock_domain*>...>]
[-DESTination <-ALL | <clock_domain*>...>]
[-FROM        <-ALL | -REG | -PI | -BBOx | instance*...>]
[-TO          <-ALL | -REG | -PO | -BBOx | instance*...>]
[<-MODule | -INStance> <name*>... [-RECurSive | -NORECurSive] ]
| [ [-SET | -RESET] [-ALL| <ID|instance_name> ...] ]
]
(Verify Mode)
```

Turns off the Clock Domain Crossing (CDC) Check for the specified portion of the design. These checks were originally specified with the `ADD CDC CHECK` command.

Note: This command deletes convergence check results. You must run the `VALIDATE` command after the `DELETE SDC CHECK` command.

Due to the nature of the FIFO structure, clock domain crossing paths to and from FIFO memory are inherently synchronized. These clock domain crossing paths do not need additional MUX or DFF synchronization. Therefore, the synchronization violations reported for these paths can be ignored or filtered. In the tool, these paths are automatically removed by the clock domain crossing list through by implicit `DELETE CDC CHECK` commands. Therefore, the following commands can affect the behavior `DELETE CDC CHECK`.

- `ADD FIFO INFO`
- `GENERATE FIFO INFO`
- `SET CDC OPTON -fifo_detect`

Tcl Command

`delete_cdc_check`

Parameters

<code>-ALL</code>	Deletes all CDC checks (Includes structural, functional, set, and reset checks).
-------------------	--

Conformal Extended Checks Reference Manual

Command Reference

-STRUctural	<p>Deletes the Structural CDC Check as specified.</p> <p>Structural CDC Checks include many checks such as <code>cdc_path_logic_type_check</code>, <code>cdc_path_destination_check</code>, <code>sync_chain_dff_number_check</code>.</p> <p>See REPORT VALIDATED DATA for more information.</p>						
-FUNctional	<p>Deletes the Functional CDC Check as specified. Functional CDC Checks include Source Data, Destination Data, MUX Enable, and Single-bit Change checks.</p> <p>If you do not specify the type of Functional CDC Check, Conformal Extended Checks will run all checks by default.</p> <tr><td>SOURCE_DATA</td><td><p>Deletes the Source Data Functional CDC Check for the specified portion of the design.</p><p>The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.</p></td></tr> <tr><td>DESTINATION_DATA</td><td><p>Deletes the Destination Data Functional CDC Check for the specified portion of the design.</p><p>The Destination Data Functional CDC Check determines whether data entering the destination register is held stable for the destination register to latch it.</p></td></tr> <tr><td>MUX_enable</td><td><p>Deletes the MUX Enable Functional CDC Check for the specified portion of the design.</p><p>This check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.</p></td></tr>	SOURCE_DATA	<p>Deletes the Source Data Functional CDC Check for the specified portion of the design.</p> <p>The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.</p>	DESTINATION_DATA	<p>Deletes the Destination Data Functional CDC Check for the specified portion of the design.</p> <p>The Destination Data Functional CDC Check determines whether data entering the destination register is held stable for the destination register to latch it.</p>	MUX_enable	<p>Deletes the MUX Enable Functional CDC Check for the specified portion of the design.</p> <p>This check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.</p>
SOURCE_DATA	<p>Deletes the Source Data Functional CDC Check for the specified portion of the design.</p> <p>The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.</p>						
DESTINATION_DATA	<p>Deletes the Destination Data Functional CDC Check for the specified portion of the design.</p> <p>The Destination Data Functional CDC Check determines whether data entering the destination register is held stable for the destination register to latch it.</p>						
MUX_enable	<p>Deletes the MUX Enable Functional CDC Check for the specified portion of the design.</p> <p>This check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.</p>						

Conformal Extended Checks Reference Manual

Command Reference

`SINGLE_bit_change`

Deletes the Single-bit Change Functional CDC Check for the specified portion of the design.

The Single-bit Change Functional CDC Check ensures that a vector of signals crossing different clock domains can only be changed one bit at a time.

`-STATUS` Deletes the specified CDC checks by the status of the checks.

`PASS` Deletes the passed CDC checks.

`FAIL` Deletes the failed CDC checks.

Note: You can specify both `PASS` and `FAIL`.

`-SYNC_rule <rule> ...`

Deletes the passed or failed checks only for the specified sync rules. If no rule is specified, all sync rule results with the specified status are deleted.

`-SOURCE` Deletes the specified CDC checks from the paths with the specified source clock domain.

`-ALL` Uses all clock domains as the source.

`<clock_domain* ...>`

Uses the specified clock domains as the source. This accepts wildcards.

`-DESTination` Deletes the specified CDC checks from the paths with the specified destination clock domain.

`-ALL` Uses all clock domains as the destination.

`<clock_domain* ...>`

Uses the specified clock domains as the destination. This accepts wildcards.

`-FROM` Deletes the specified CDC checks from the paths that start from the specified key points.

Conformal Extended Checks Reference Manual

Command Reference

	-ALL	Uses all types of key points as the start of the paths. The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-REG	Uses registers as the start of the paths.
	-PI	Uses primary inputs as the start of the paths.
	-BBOX	Uses blackbox (outputs) as the start of the paths.
	<instance*>	Uses the specified instances as the start of the paths. This accepts wildcards. The instances can be registers, primary inputs, or blackbox outputs.
-TO		Deletes the specified CDC check from the paths that end at the specified key points.
	-REG	Uses registers as the end of the paths.
	-PO	Uses primary outputs as the end of the paths.
	-BBOX	Uses blackbox (inputs) as the end of the paths.
	<instance*>	Uses the specified instances as the end of the paths. This accepts wildcards. The instances can be registers, primary outputs, or blackbox inputs.
-MODULE INSTANCE	<name* ...>	Deletes this check from the specified modules or module instance path names.
	-RECURSIVE	Includes submodules.
	-NORECURSIVE	Does not include submodules.
-SET		Deletes the check for asynchronous set pins of the flip-flops to ensure that they are consistent with the clock domain.
-RESET		Deletes the check for asynchronous reset pins of the flip-flops to ensure that they are consistent with the clock domain.

Related Commands

ADD CDC CHECK

ADD FIFO INFO

GENERATE FIFO INFO

REPORT CDC CHECK

REPORT VALIDATED DATA

SET CDC OPTION

VALIDATE

DELETE CDC FILTER

DELEte CDc Filter

```
<<filter_name>... | -All>  
(Verify Mode)
```

Deletes CDC filters that were originally specified with the ADD CDC FILTER command. The validation should be rerun for the results of these checks to be available.

Tcl Command

```
delete_cdc_filter
```

Parameters

<filter_name>	Specifies the CDC filter(s) to delete.
-ALL	Deletes all CDC filters.

Related Commands

ADD CDC FILTER

REPORT CDC FILTER

DELETE CLOCK

DELeTe CLoCk

<-ALL | <primary_pin*> ... | <id | pathname*> ...>
(Setup Mode)

Deletes clocks that were defined using the ADD CLOCK command.

Tcl Command

delete_clock

Parameters

-ALL	Deletes all defined clocks.
<primary_pin*>	Deletes the specified clocks. This accepts wildcards.
<id pathname>	Deletes the clocks with the specified ID or path.
Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.	

Related Commands

ADD CLOCK

REPORT CLOCK

DELETE CLOCK ASSOCIATION

DELEte CLock Association

<-ALL | <label_name* ... [-Gate | -Association]>
(Verify Mode)

Deletes the clock associations that were created with the ADD CLOCK ASSOCIATION command.

Tcl Command

delete_clock_association

Parameters

-ALL	Deletes all clock associations.
<label_name* ...>	Deletes only those clock associations with the specified label name(s). This accepts wildcards.
-Gate	Specifies whether the names you specified refer to gates. <i>This option is the default.</i>
-Association	Specifies whether the names you specified refer to associations.

Related Commands

ADD CLOCK ASSOCIATION

PROPAGATE CLOCK DOMAIN

REPORT CLOCK ASSOCIATION

REPORT CLOCK DOMAIN

DELETE DATA ASSOCIATION

DELEte DAta Association

<-ALL | <pathname* ... [-Gate | -Association]>
(Verify Mode)

Deletes the data associations that were created with the ADD DATA ASSOCIATION command.

Tcl Command

delete_data_association

Parameters

-ALL	Deletes all data associations.
<pathname* ...>	<p>Deletes only those data associations with the specified paths. This accepts wildcards.</p> <p>If you specify the representative path, Conformal Extended Checks deletes the entire association.</p> <p>If you specify paths other than the representative path, Conformal Extended Checks removes the specified paths from the association. The association remains.</p>
-Gate	Indicates that the specified paths, or paths, contain a list of gates to be removed from existing data associations.
-Association	Indicates that the specified paths, or paths, contain a list of association names to be deleted. The valid names are clock domain names, or names that were defined with the add data association -NEW -Name command.

Related Commands

ADD DATA ASSOCIATION

PROPAGATE CLOCK DOMAIN

REPORT DATA ASSOCIATION

DELETE DISPLAY LIST

DELEte DIspLay LIst

<-All | <pathname | id> ...>
(Verify Mode)

Deletes the specified gates from the display list. The specified gates were originally added to the display list using the `ADD DISPLAY LIST` command.

Tcl Command

`delete_display_list`

Parameters

-All

Removes all gates from the display list.

<pathname | id>

Removes gates with the specified paths or gate IDs from the display list.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

[ADD DISPLAY LIST](#)

[REPORT DISPLAY LIST](#)

DELETE DYNAMIC CONSTRAINTS

DELEte DYnamic Constraints

```
< -ALL | <id | pathname*> ...>  
(Verify Mode)
```

Deletes dynamic constraints originally added with the `ADD DYNAMIC CONSTRAINTS` command.

Use the `REPORT DYNAMIC CONSTRAINTS` command to display a list of all added dynamic constraints.

Tcl Command

```
delete_dynamic_constraints
```

Parameters

<code>-All</code>	Deletes "all" dynamic constraints within the given defaults.
<code><id pathname*></code>	Specifies the gate ID or path of the flattened gate that you want to delete dynamic constraints from. The pathname option accepts wildcards.

Related Commands

[ADD DYNAMIC CONSTRAINTS](#)

[EXPLORE](#)

[REPORT DYNAMIC CONSTRAINTS](#)

DELETE FIFO INFO

DELEte FIfO Info

```
<-Index <index>
  | -INSTance <instance_name> [-Force]
  | -Module <module_name> >
[ [-MEM <reg1 reg2 ...>]
[-RADr <reg1 reg2 ...>]
[-WADr <reg1 reg2 ..>]
[-RSYNc <<reg1_1 reg1_2> ...>]
[-WSYnc <<reg1_1 reg1_2> ...>]
[-RGRay <reg1 reg2 ...>]
[-WGRay <reg1 reg2 ...>] ]
| -All]
(Verify Mode)
```

Deletes information regarding the FIFOs in the design.

Tcl Command

delete_fifo_info

Parameters

- | | |
|---|---|
| <code>-Index <index></code> | Specifies the index of an existing FIFO from which to delete information. To see the indices, use the <code>REPORT FIFO INFO</code> command. |
| <code>-INSTance <instance_name> [-Force]</code> | <p>Specifies an instance name to delete information from the FIFO.</p> <p>If there are multiple FIFOs with the same instance name, you will get an error if you attempt to delete a FIFO with this option. To override this, specify the <code>-Force</code> option to delete all the instances with the same name.</p> |
| <code>-Module <module_name></code> | Specifies that the information of all the instances of the module will be deleted from the FIFO. |
| <code>-MEM <reg1 reg2 ...></code> | Deletes information for one or more memory registers from the FIFO. |

Conformal Extended Checks Reference Manual

Command Reference

- RADr <reg1 reg2 ...> Deletes information for one or more read address registers from the FIFO.
- WADr <reg1 reg2 ...> Deletes information for one or more write address registers from the FIFO.
- RSYNc <<reg1_1 reg1_2 ...>
Deletes information for one or more pairs of read DFF synchronizers from the FIFO.
- WSYNc <<reg1_1 reg1_2 ...>
Deletes information for one or more pairs of write DFF synchronizers from the FIFO.
- RGRay <reg1 reg2 ...>
Deletes information for one or more read gray code registers from the FIFO.
- WGRay <reg1 reg2 ...>
Deletes information for one or more write gray code registers from the FIFO.
- All Deletes all information from the FIFO.

Related Commands

ADD FIFO INFO

GENERATE FIFO INFO

REPORT FIFO INFO

DELETE GENERATED CLOCK

DELEte GEnerated Clock

<-ALL | <id | pathname*> ...>
(Verify Mode)

Deletes clock domains created with the ADD GENERATED CLOCK command.

Note: This clock domain is for use in Clock Domain Checking only.

Tcl Command

delete_generated_clock

Parameters

-ALL	Deletes all clock domains previously created with the ADD GENERATED CLOCK command.
<id pathname*>	Deletes clock domains with the specified ID or path. Wildcards are supported for paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

ADD GENERATED CLOCK

PROPAGATE CLOCK DOMAIN

REPORT CLOCK DOMAIN

REPORT GENERATED CLOCK

DELETE IGNORE RESET_CONSTRAINT

DELeTe IGNore RESET_constraint

```
<-ALL
| <-BRANCH_enable
| -FSMTransition
| -FSMReachability
| -TRI_state
|
>
> (Setup / Verify Mode)
```

Reapplies the reset constraints to the specified checks.

See [ADD IGNORE RESET_CONSTRAINT](#) for more information.

Tcl Command

`delete_ignore_reset_constraint`

Parameters

-ALL	Re-applies reset constraints for Branch Enable, FSM Transition, FSM Reachability, and Tristate Stuck-on/Stuck off checks.
-BRANCH_enable	Re-applies reset constraints for Branch Enable checks.
-FSMTransition	Re-applies reset constraints for FSM Transition checks.
-FSMReachability	Re-applies reset constraints for FSM Reachability checks.
-TRI_state	Re-applies reset constraints for Tristate Stuck-on/Stuck off checks.

Related Commands

[ADD ASSERTION CONSTRAINTS](#)

[ADD IGNORE RESET_CONSTRAINT](#)

[ADD PIN CONSTRAINTS](#)

[REPORT IGNORE RESET_CONSTRAINT](#)

DELETE IGNORED PROPERTY

DELeTe IGnored Property

```
<-ALL
| <-TRI_state [-ALL | -PO | <id | pathname*> ... ]
| -DONT_CARE [-ALL | -X_ASSIGNment | -RANGE_overflow
| -FULL_case | -PARAllel_case | <id | pathname*> ...
]
| -BRANCH_enable [ -ALL | -IF_else | -CAsE
| -DEFault | -FOR_loop | <id | pathname*> ...
]
| <-BUS | -MULTI_PORT | -SET_RESET
| -FSM | -FSMReachability
| -FSMTransition | -FSMDeadlock
| -NO_DIVide_by_zero
| -ENCoding_completeness
>
[-ALL | <id | pathname*> ...]
>...
>
(Verify Mode)
```

Reinstates all or specific properties. This reverses the ADD IGNORED PROPERTY command.

Note: If you specify properties with the ADD IGNORED PROPERTY command, you cannot specify these same properties with the ADD STATIC PROPERTY command until you reinstate them.

Tcl Command

delete_ignored_property

Parameters

-ALL	Reinstates all properties.
-TRI_state	Reinstates tristate properties, as specified.
-ALL	Reinstates all tristate properties. <i>This option is the default.</i>
-PO	Reinstates tristate properties connected to the primary output.
<id pathname*>	

Conformal Extended Checks Reference Manual

Command Reference

Reinstates tristate properties with the specified IDs or paths. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-DONT_CARE

Reinstates Don't Care properties, as specified.

-ALL

Reinstates all Don't Care properties. *This option is the default.*

-X_ASSIGNment

Reinstates properties for Don't Care X-assignment cases.

-RANGE_overflow

Reinstates properties for Don't Care array index overflow cases.

-FULL_case

Reinstates Don't Care properties for `full_case` synthesis directives.

-PARAllel_case

Reinstates Don't Care properties for `parallel_case` synthesis directives.

<id | pathname*>

Reinstates Don't Care properties with the specified IDs or paths. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BRANCH_enable

Reinstates Branch Enable properties, as specified.

Conformal Extended Checks Reference Manual

Command Reference

-ALL	Reinstates all Branch Enable properties. <i>This options is the default.</i>
-IF_else	Reinstates Branch Enable properties for if-else conditional branches.
-CAsE	Reinstates Branch Enable properties for case conditional branches. (This does not include default branches.)
-DEFault	Reinstates Branch Enable properties for case-default conditional branches.
-FOR_loop	Reinstates Branch Enable <code>for_loop</code> properties.
<id pathname*>	Reinstates the specified Branch Enable properties. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-BUS	Reinstates the specified Bus properties.
-MULTI_PORT	Reinstates the specified Multi-port properties.
-SET_RESET	Reinstates the specified Set-Reset properties.
-FSM	Reinstates finite state machine checks.
-FSMReachability	Reinstates finite state machine Reachability checks.
-FSMTransition	Reinstates finite state machine Transition checks.
-FSMDeadlock	Reinstates finite state machine Deadlock checks.
-NO_DIVide_by_zero	Reinstates No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
-ENCoding_completeness	Reinstates Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.

Conformal Extended Checks Reference Manual

Command Reference

-ALL

Reinstates all properties, as specified. For example, the following command reinstates all Bus properties that were formerly ignored:

```
delete ignored property -bus -all
```

This option is the default.

<id | pathname*>

Reinstates properties with the specified ID or hierarchical path. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

ADD BLACK BOX

ADD IGNORED PROPERTY

ADD NOTTRANSLATE MODULES

ADD STATIC PROPERTY

DELETE BLACK BOX

DELETE NOTTRANSLATE MODULES

DELETE STATIC PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

REPORT BLACK BOX

REPORT FSM

REPORT GATE

REPORT IGNORED PROPERTY

REPORT NOTTRANSLATE MODULES

Conformal Extended Checks Reference Manual

Command Reference

REPORT PROVED DATA

REPORT STATIC PROPERTY

DELETE INITIAL STATE

DELEte INitial State

```
<<-All | instance_pathname* ...>>  
(Setup Mode)
```

Deletes the initial state values that were added with the `ADD INITIAL STATE` command.

Use the `REPORT INITIAL STATE` command to display a list of all added initial states.

Tcl Command

```
delete_initial_state
```

Parameters

`-All` Removes all previously added initial states.

`<instance_pathname* ...>`

Removes the initial states for the specified instance paths. This accepts wildcards.

Related Commands

[ADD INITIAL STATE](#)

[REPORT INITIAL STATE](#)

DELETE INSTANCE CONSTRAINTS

DELEte INstance Constraints

```
<<instance_pathname ...> | -All>  
(Setup Mode)
```

Deletes instance constraints that were added with the `ADD INSTANCE CONSTRAINTS` command.

Use the `REPORT INSTANCE CONSTRAINTS` command to display a list of all added instance constraints.

Tcl Command

```
delete_instance_constraints
```

Parameters

<instance_pathname ...>

Deletes constraints on the specified instance paths.

-All

Deletes all instance constraints.

Related Commands

`ADD INSTANCE CONSTRAINTS`

`REPORT INSTANCE CONSTRAINTS`

DELETE NOTRANSLATE FILEPATHNAMES

DELeTe NOttranslate Filepathnames

```
<<filepath_name* ...> | -All>  
[ | -Library | -Design]  
(Setup Mode)
```

Deletes the specified file pathnames originally added with the ADD NOTRANSLATE FILEPATHNAMES command.

Use the REPORT NOTRANSLATE FILEPATHNAMES command to display a list of all of the library and design file pathnames.

Wildcard: The wildcard (*) represents any zero or more characters in module names.

Tcl Command

```
delete_notranslate_filepathnames
```

Parameters

<filepath_name* ...>

Deletes the listed notranslate file pathnames.

-All

Deletes "all" previously added notranslate file pathnames. -All applies within the given defaults.

-Library

Deletes the specified library file pathnames. *This is the default.*

-Design

Deletes the specified design file pathnames.

Related Commands

ADD NOTRANSLATE FILEPATHNAMES

ADD NOTRANSLATE MODULES

DELETE NOTRANSLATE MODULES

REPORT NOTRANSLATE FILEPATHNAMES

REPORT NOTRANSLATE MODULES

DELETE NOTRANSLATE MODULES

DELeTe NOttranslate Modules

```
<<module_name* ...> | -All>  
[-Library | -Design]  
(Setup Mode)
```

Deletes the specified module names that were added with the ADD NOTRANSLATE MODULES command. Run this command before READ LIBRARY and READ DESIGN.

Use the REPORT NOTRANSLATE MODULES command to display a list of all library and design module names that Conformal will not compile.

Tcl Command

```
delete_notranslate_modules
```

Parameters

<module_name* ...>

Deletes the specified modules. This accepts wildcards.

-All

Deletes all previously added module names.

-Library

Deletes the specified library module names. *This option is the default.*

-Design

Deletes the specified design module names.

Related Commands

ADD BLACK BOX

ADD NOTRANSLATE MODULES

DELETE BLACK BOX

READ DESIGN

READ LIBRARY

REPORT BLACK BOX

Conformal Extended Checks Reference Manual

Command Reference

REPORT NOTRANSLATE MODULES

DELETE PIN CONSTRAINTS

DELEte PIn Constraints

```
<-ALL_Pin | primary_pin*...>  
[-Module <module_name>]  
(Setup Mode)
```

Deletes constraints that were placed on the design's named primary input pins. Constraints were originally placed on primary input pins with the `ADD PIN CONSTRAINTS` command.

Use the `REPORT PIN CONSTRAINTS` command to display a list of all added pin constraints.

Tcl Command

```
delete_pin_constraints
```

Parameters

<code>-ALL_Pin</code>	Deletes all constraints placed on primary input pins. <i>This option is the default.</i>
<code>primary_pin*...</code>	Deletes pin constraints from the listed primary inputs. (These primary inputs were originally constrained with the <code>ADD PIN CONSTRAINTS</code> command.) This accepts wildcards.
<code>-Module <module_name></code>	Deletes pin constraints from the specified module.

Related Commands

`ADD PIN CONSTRAINTS`

`REPORT PIN CONSTRAINTS`

DELETE PIN EQUIVALENCES

DELEte Pin Equivalences

```
<-ALL_Pin | primary_pin*...>  
[-ROot | -Module <module_name> | -ALL_Module]
```

Deletes the added pin equivalences from the specified primary input pins. The equivalences were originally placed on primary input pins with the `ADD PIN EQUIVALENCES` command.

Use the `REPORT PIN EQUIVALENCES` command to display the list of all added pin equivalences.

Tcl Command

```
delete_pin_equivalences
```

Parameters

<code>-ALL_Pin</code>	Deletes "all" added pin equivalences within the given defaults.
<code>primary_pin*...</code>	Deletes pin equivalences from the listed primary input pins. This accepts wildcards. Pin equivalence was originally added with the <code>ADD PIN EQUIVALENCE</code> command.
<code>-ROot</code>	Deletes pin equivalences from the root module. <i>This is the default.</i>
<code>-Module <module_name></code>	Deletes pin equivalences from the specified module.
<code>-ALL_Module</code>	Deletes pin equivalences from all modules.

Related Commands

[ADD PIN EQUIVALENCES](#)

[REPORT PIN EQUIVALENCES](#)

DELETE PRIMARY INPUTS

DELeTe PRImary Inputs

<-All | pathname*...>
(Setup Mode)

Deletes the primary input and cut point established by the `ADD PRIMARY INPUT` command.

Tcl Command

`delete_primary_inputs`

Parameters

<code>-All</code>	Deletes all user-defined primary inputs.
<code>pathname*...</code>	Deletes the specified primary input pins. This accepts wildcards.

Related Commands

`ADD PRIMARY INPUT`

`REPORT PRIMARY INPUTS`

DELETE RENAMING RULE

DELeTe RENaming Rule

<-All | rulename*>
(Setup Mode)

Deletes renaming rules that were originally added with the `ADD RENAMING RULE` command.

Tcl Command

`delete_renaming_rule`

Parameters

<code>-All</code>	Specifies that all renaming rules are to be deleted.
<code>rulename*</code>	Specifies the rule name(s) to be deleted. This accepts wildcards.

Related Command

`ADD RENAMING RULE`

DELETE RESET_SYNC MODULE

DELeTe REset_sync Module

```
<-ALL | <reset_sync_module_name*> ...>  
(Verify Mode)
```

Deletes set and reset synchronizers defined by the ADD RESET_SYNC MODULE command.

Tcl Command

delete_reset_sync_module

Parameters

-ALL Deletes all set and reset synchronizers.

<reset_sync_module_name*> ...

Specifies the set and reset synchronizer(s) to delete. This accepts wildcards.

Related Commands

ADD RESET_SYNC MODULE

REPORT RESET_SYNC MODULE

DELETE RESET SOURCE

DELeTe REset Source

<-ALL | <pathname*> ...>
(Verify Mode)

Deletes the hierarchical paths of the reset input signals that were specified with the ADD RESET SOURCE command.

Tcl Command

delete_reset_source

Parameters

-ALL	Deletes all hierarchical paths of the reset signals.
<pathname*> ...	Specifies the hierarchical path, or paths, of the reset signals to delete. This accepts wildcards.

Related Commands

- ADD RESET SOURCE
- REPORT RESET SOURCE

DELETE RULE FILTER

DELeTe RULe Filter

<filtername*> ...
(Setup Mode)

Deletes the specified rule filters.

Tcl Command

delete_rule_filter

Parameters

<filtername*> ... Specifies the name(s) of the filter(s) to delete.

Related Commands

ADD RULE FILTER

REPORT RULE FILTER

DELETE RULE WAIVER

DELEte RuLe Waiver

```
<rulename* [[occurrence#] [ ' .. ' ] [occurrence#]] ... > ...  
(Setup Mode)
```

Unmarks the specified occurrences as waived. If no occurrence number is specified, all the occurrences of the rules that match the specified pattern are deleted.

Tcl Command

delete_rule_waiver

Parameters

rulename*	Specifies the rule(s). This accepts wildcards.
occurrence#	Specifies the number of occurrences to apply the waiver.
'..'	Specifies a range of occurrence numbers. All occurrences between the two numbers surrounding this argument will be affected. If no number precedes this argument, the range begins with the first occurrence. If no number succeeds this argument, the range extends to the last occurrence. Note: Spaces are required between the numbers and ' .. '. See example for ADD RULE WAIVER.

Related Commands

ADD RULE WAIVER

READ RULE CHECK

WRITE RULE CHECK

DELETE SEARCH PATH

DELeTe SEArch Path

```
<-All | pathname ... >  
[-Design | -Library | -POWER_intent]  
(Setup Mode)
```

Deletes search paths Conformal uses for the READ DESIGN and READ LIBRARY commands.

Use the REPORT SEARCH PATH command to display the list of all search paths.

Run this command before READ LIBRARY and READ DESIGN.

Tcl Command

```
delete_search_path
```

Parameters

-All	Deletes all search paths.
pathname...	Deletes the specified search paths.
-Design	Deletes search paths used by the READ DESIGN command. <i>This option is the default.</i>
-Library	Deletes search paths used by the READ LIBRARY command.
-POWER_intent	This is a low power command option. Deletes the specified search path for power intent files.

Related Commands

ADD SEARCH PATH

READ DESIGN

READ LIBRARY

REPORT SEARCH PATH

DELETE SET SOURCE

DELEte SET Source

<-ALL | <pathname*> ...>
(Verify Mode)

Deletes the hierarchical paths of the set input signals that were specified with the ADD SET SOURCE command.

Tcl Command

delete_set_source

Parameters

-ALL	Deletes all hierarchical paths of the set signals.
<pathname*> ...	Specifies the hierarchical path, or paths, of the set signals to delete. This accepts wildcards.

Related Commands

ADD SET SOURCE

REPORT SET SOURCE

DELETE STATIC PROPERTY

DELEte StAtic Property

```
<-ALL
| <-TRI_state [ -ALL | -PO | <id | pathname*>...]
| -DONT_CARE [ -ALL | -X_ASSIGNment | -RANGE_overflow
| -FULL_case | -PARAllel_case | <id | pathname*>...]
| -BRANCH_enable
| [ -ALL | -IF_else | -CASE | -DEFAULT | -FOR_loop | <id | pathname*>...]
| <-BUS | -MULTI_PORT | -SET_RESET | -FSM | -FSMReachability
| -FSMTransition | -FSMDeadlock | -NO_DIVide_by_zero
| -ENCoding_completeness
>
[-ALL | <id | pathname*>...]
>...
>
(Verify Mode)
```

Deletes properties from the "Prove" list. (Properties were originally added with the ADD STATIC PROPERTY command.)

Use the DELETE STATIC PROPERTY command to remove all or specified properties from the "Prove" list.

Tcl Command

delete_static_property

Parameters

-ALL	Deletes all predefined properties from the "Prove" list.
-TRI_state	Deletes the specified tristate properties from the "Prove" list.
-ALL	Deletes <i>all</i> tristate properties from the "Prove" list. <i>This option is the default.</i>
-PO	Deletes tristate properties connected to the primary output.
id pathname*...	

Conformal Extended Checks Reference Manual

Command Reference

Deletes tristate properties with the specified ID or hierarchical path. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-DONT_CARE

Deletes the specified Don't Care properties.

-ALL

Deletes all Don't Care properties. *This option is the default.*

-X_ASSIGNment

Deletes Don't Care properties for X-assignment cases.

-RANGE_overflow

Deletes Don't Care properties for array index overflow cases.

-FULL_case

Deletes Don't Care properties for full_case synthesis directive.

-PARAllel_case

Deletes Don't Care properties for parallel_case synthesis directive.

id | pathname*...

Deletes the Don't Care properties for the specified IDs or paths. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-BRANCH_enable

Deletes Branch Enable properties, as specified.

Conformal Extended Checks Reference Manual

Command Reference

-ALL	Deletes all Branch Enable properties. That is, do not check whether case and if-else conditional branches can be enabled. <i>This option is the default.</i>
-IF_else	Deletes Branch Enable properties for if-else conditional branches. That is, do not check whether if-else conditional branches can be enabled.
-CAsE	Deletes Branch Enable properties for case conditional branches. That is, do not check whether case conditional branches can be enabled. (This does not include default branches.)
-DEFault	Deletes Branch Enable properties for case-default conditional branches. That is, do not check whether case-default conditional branches can be enabled.
-FOR_loop	Deletes Branch Enable for_loop properties. That is, do not check whether for_loops can be executed.
id pathname*...	<p>Deletes the specified Branch Enable properties. That is, do not check whether the specified Branch Enable properties can be enabled. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-BUS	Deletes the specified Bus properties.
-MULTI_PORT	Deletes the specified Multi-port properties.
-SET_RESET	Deletes the specified Set-Reset properties.
-FSM	Deletes the specified finite state machine properties.

Conformal Extended Checks Reference Manual

Command Reference

<code>-FSMReachability</code>	Deletes the specified finite state machine Reachability properties.
<code>-FSMTransition</code>	Deletes the specified finite state machine Transition properties.
<code>-FSMDeadlock</code>	Deletes the specified finite state machine Deadlock properties.
<code>-NO_DIVide_by_zero</code>	Deletes No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
<code>-ENCoding_completeness</code>	Deletes Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
<code>-ALL</code>	<p>Deletes all properties of the specified type. For example, the following command deletes all Bus properties from the "Prove" list:</p> <pre>delete static property -bus -all</pre> <p><i>This option is the default.</i></p>
<code>id pathname*...</code>	<p>Deletes the specified property for the specified IDs or paths. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>

Related Commands

[ADD ASSERTION CONSTRAINTS](#)

[ADD IGNORED PROPERTY](#)

[ADD STATIC PROPERTY](#)

[DELETE ASSERTION CONSTRAINTS](#)

[DELETE IGNORED PROPERTY](#)

[DIAGNOSE STATIC PROPERTY](#)

Conformal Extended Checks Reference Manual

Command Reference

PROVE

REPORT ASSERTION CONSTRAINTS

REPORT EXTRACTED PROPERTY

REPORT FSM

REPORT GATE

REPORT PROVED DATA

REPORT STATIC PROPERTY

DELETE SYNC INPUT

DELeTe SYnc Input

< -ALL | <pathname*> ...>
(*Verify Mode*)

Removes synchronized signals that were added with the ADD SYNC INPUT command.

Tcl Command

delete_sync_input

Parameters

-ALL	Deletes all the signals added using "ADD SYnc Input" command
<pathname> ...	Specifies the name of path(s).

Related Commands

ADD SYNC INPUT

REPORT SYNC INPUT

DELETE SYNCHRONIZATION RULE

DELeTe SYNchronization Rule

<-ALL | <rule_name*...>>
(Verify Mode)

Deletes synchronization rules previously created with the `ADD SYNCHRONIZATION RULE` command.

Tcl Command

`delete_synchronization_rule`

Parameters

<code>-ALL</code>	Deletes all synchronization rules.
<code>rule_name*...</code>	Deletes synchronization rules with the specified rule names. This accepts wildcards.

Related Commands

`ADD SYNCHRONIZATION RULE`

`REPORT SYNCHRONIZATION RULE`

`VALIDATE`

DELETE TIED SIGNALS

DELEte Tied Signals

```
<-All | name...>  
[-Net | -Pin]  
[-Module <module_name>]  
[-Class <Full | User | System>]  
(Setup Mode)
```

Deletes specified tied signals from the design. Use the `REPORT TIED SIGNALS` command to display a list of all the tied signals.

Tcl Command

```
delete_tied_signals
```

Parameters

-All	Deletes all tied signals.
name...	Deletes the specified tied signals.
-Net	Specifies that the name of the tied signal is a net name. <i>This option is the default.</i>
-Pin	Specifies that the name of the tied signal is a pin name.
-Module <module_name>	Specifies that the name of the module where the floating net or pin resides.
-Class	Deletes tied signals of the following class:
Full	tied signals from both the user and system class. <i>This option is the default.</i>
User	tied signals that were added previously with the <code>ADD TIED SIGNALS</code> command.
System	tied signals from the original design.

Related Commands

ADD TIED SIGNALS

Conformal Extended Checks Reference Manual

Command Reference

REPORT TIED SIGNALS

DELETE VECTOR DEFINITION

DELeTe VEctor Definition

<vectorname*>
(Verify Mode)

Deletes the definition of the matching vectors defined by the ADD VECTOR DEFINITION command.

Tcl Command

delete_vector_definition

Parameters

<vectorname*>	Specifies the name of the vector(s) to delete the definition. This accepts wildcards.
---------------	---

Related Commands

ADD VECTOR DEFINITION

REPORT VECTOR DEFINITION

DIAGNOSE CDC CHECK

DIAGnose CDc Check

```
<-SOURCE_DATA | -DESTINATION_DATA  
    |-MUX_enable | -SINGLE_bit_change>  
<from-instance> <to-instance>  
[ |-SOURCE <clock_domain> ]  
[ |-DESTination <clock_domain>]  
[ |-SYNC_RULE <rule_name>]  
(Verify Mode)
```

Diagnoses the specified path that failed the Functional CDC Checks. Additionally, it generates a counter-example for the failed path. To save the counter-example to a file, use the `WRITE CDC CHECK` command.

Use the Conformal Extended Checks graphical user interface to further aid in diagnosis.

Tcl Command

```
diagnose_cdc_check
```

Parameters

<code>-SOURCE_DATA</code>	<p>Diagnoses the specified path that failed the Source Data Functional CDC Check.</p> <p>The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable in order for the destination register to latch it.</p>
<code>-DESTINATION_DATA</code>	<p>Diagnoses the specified path that failed the Destination Data Functional CDC Check.</p> <p>The Destination Data Functional CDC Check determines whether data entering the destination register is held stable in order for the destination register to latch it.</p>

Conformal Extended Checks Reference Manual

Command Reference

<code>-MUX_enable</code>	<p>Diagnoses the specified path that failed the MUX Enable Functional CDC Check.</p> <p>The MUX Enable Functional CDC Check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.</p>
<code>-SINGLE_bit_change</code>	<p>Diagnoses the specified path that failed the Single-bit Change Functional CDC Check.</p> <p>The Single-bit Change Functional CDC Check determines whether a vector of signals crossing different clock domains can only be changed one bit at a time.</p>
<code>from-instance</code>	Diagnoses the CDC path that has the specified beginning.
<code>to-instance</code>	Diagnoses the CDC path that has the specified ending.
<code>-SOURCE clock_domain</code>	<p>Diagnoses the CDC path with the specified source domain.</p> <p>Use this option with multi-port latches to uniquely identify paths with the same <code>from-instance</code> and <code>to-instance</code>.</p>
<code>-DESTination clock_domain</code>	<p>Diagnoses the CDC path with the specified destination domain.</p> <p>Use this option with multi-port latches to uniquely identify paths with the same <code>from-instance</code> and <code>to-instance</code>.</p>
<code>-SYNC_RULE rule_name</code>	<p>Diagnoses the CDC path with the specified <code>rule_name</code>.</p> <p>Use this option with a failed <code>mux_enable</code> CDC path to uniquely identify paths with the same <code>from-instance</code> and <code>to-instance</code>.</p> <p>Use this option when multiple MUX synchronization rules apply to the same path.</p>

Related Commands

[ADD CDC CHECK](#)

[ADD SYNCHRONIZATION RULE](#)

[WRITE CDC CHECK](#)

DIAGNOSE STATIC PROPERTY

DIAGnose STatic Property

```
[ | -FLOating | -MULTI_driven | -CONTention | -TRI_ON | -TRI_OFF  
  | <-SET_RESET | -SR>  
  | -FSMReachability | -FSMDeadlock]  
<id | pathname>  
[ | -COMbinational]  
(Verify Mode)
```

Diagnoses a property that failed during the verification process. Additionally, it generates a counter-example for the failed property, with simulation values.

To examine simulation values, follow the `DIAGNOSE STATIC PROPERTY` command with the `REPORT GATE` command including the `-simvalue` option. Alternatively, use the Conformal Extended Checks graphical user interface to view this information.

Tcl Command

```
diagnose_static_property
```

Parameters

<code>-FLOating</code>	<p>Generates a counter-example for the specified floating property.</p> <p>Use the <code>REPORT GATE</code> command or GUI to display the simulation values.</p>
<code>-MULTI_driven</code>	<p>Generates a counter-example for the specified multi-driven property.</p> <p>Use the <code>REPORT GATE</code> command or GUI to display the simulation values.</p>
<code>-CONTention</code>	<p>Generates a counter-example for the specified contention property.</p> <p>Use the <code>REPORT GATE</code> command or GUI to display the simulation values.</p>

Conformal Extended Checks Reference Manual

Command Reference

-TRI_ON	Generates a counter-example for the specified tristate stuck-on property. Use the <code>REPORT GATE</code> command or GUI to display the simulation values.
-TRI_OFF	Generates a counter-example for the specified tristate stuck-off property. Use the <code>REPORT GATE</code> command or GUI to display the simulation values.
-SET_RESET -SR	Generates a counter-example for the specified Set-Reset property. Use the <code>REPORT GATE</code> command or GUI to display the simulation values.
-FSMReachability	Generates an example for the specified FSM state that passed the FSM Reachability check.
-FSMDeadlock	Generates a counter-example for the specified FSM state that is deadlocked.
id pathname	Generates a counter-example for the property with the specified ID or path. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-COMbinational	Generates a counter-example for the specified property for combinational proof depth.

Related Commands

[ADD BLACK BOX](#)

[ADD NOTRANSLATE MODULES](#)

[ADD STATIC PROPERTY](#)

[DELETE BLACK BOX](#)

[DELETE NOTRANSLATE MODULES](#)

[DELETE STATIC PROPERTY](#)

Conformal Extended Checks Reference Manual

Command Reference

PROVE

REPORT BLACK BOX

REPORT GATE

REPORT NOTRANSLATE MODULES

REPORT PROVED DATA

REPORT STATIC PROPERTY

WRITE DIAGNOSIS DATA

DOFILE

DOFile

```
<filename>  
<-NOECHO>  
(Setup / Verify Mode)
```

Executes a set of commands that are placed in a specified file. If Conformal encounters an error while running a dofile script, it terminates the script and returns to the system mode prompt.

Use the `SET DOFILE ABORT` command to specify how Conformal responds to an error message.

Use the `BREAK` command in a dofile script to terminate the dofile script and return to the system mode prompt. Then, if it is applicable, use the `CONTINUE` command when prompted.

Tcl Command

dofile

Parameters

filename	Specifies the name of the file containing a set of commands Conformal executes one at a time.
-NOECHO	Does not echo the commands while executing the commands in the file.

Related Commands

BREAK

CONTINUE

SAVE DOFILE

SET DOFILE ABORT

ELABORATE DESIGN

ELaborate DEsign

```
[-RooT <module_name>]
[-ROOTONLY]
[-ROOTConfig <configuration_name>]
[-PARAMeter [-INT | -STR | -ENUM | -TYPE] <name> <value>]
[-RAngeconstraint]
(Setup Mode)
```

Completes the READ DESIGN command specified with the `-noelaborate` option. During this step, modules are synthesized and the complete design hierarchy is created. This command is typically used for mixed design flows where the Verilog modules and VHDL entity or architectures are read in separately. Then they can be elaborated using this command.

Tcl Command

```
elaborate_design
```

Parameters

`-RooT <module_name>`

Specifies the root module to be elaborated. If this option is not specified, the Conformal software automatically selects a root module.

`-ROOTONLY`

Elaborates the root module only and skips elaboration of the other modules that are not instantiated from the root module.

Because the elaboration stage is skipped for the uninstantiated modules, this option reduces memory usage and omits the elaboration time error checking.

`-ROOTConfig <configuration_name>`

Specifies that the design includes the specified configuration for the top-level module.

Note: This option applies to only VHDL designs.

Use this option when the design includes multiple configurations for the top-level module. When you use the `-rootconfig` option, you must also use the `-root module_name` option (above).

Conformal Extended Checks Reference Manual

Command Reference

`-PARAMeter [-INT | -STR | -ENUM] <name> <value>`

Assigns design parameters or replace existing design parameters. To specify multiple parameters, use the `-parameter` option for each parameter you want to set.

When using the `-parameter -int <parameter_name> <value>` command, the `<value>` will be converted to integer value, which can be a positive integer (1), negative integer (-1), an integer value recognized as a string ("1" / "-1"), or a Verilog style integer ("16'h0001"). When using a Verilog style integer, the value must be specified between double-quotes (" ").

When using the `-parameter -str <parameter_name> <value>` command, the `<value>` will be saved as a string.

When using the `-parameter -enum <parameter_name> <value>` command, the `<value>` will be converted to a VHDL enumeration literal.

When using the `-parameter -type <parameter_name> <value>` command, `<value>` is a Verilog/SystemVerilog built-in type or user-defined type compiled in the compilation unit scope `$unit`—types defined in a module or package are not supported.

Notes:

Any value that is not recognized as an unsigned decimal integer value is interpreted as string value.

If `-int` or `-str` is not specified, then the parameter value will be interpreted as an integer if it is not between double-quotes (" "), and as a string if it is between double-quotes. Therefore, if you want to specify a Verilog format value, it must be between double-quotes and used with the `-int` option.

See `READ DESIGN -parameter` for examples.

`-RAngeconstraint` Applies range constraints during verification. If this option is not specified, all range constraints are ignored.

Related Commands

[READ DESIGN](#)

[READ LIBRARY](#)

EXIT

EXIt

`[-Force]`
`[-Verbose]`
(*Setup / Verify Mode*)

Ends the existing Conformal session and returns to the operating system.

Exit Status Codes

On exiting, Conformal Extended Checks returns a status code. A nonzero status code indicates a potential error (for example, failed properties and modeling rule violations). Conversely, a zero status code indicates that all proved properties passed, there were no counter-example warnings or modeling rule errors, and all commands executed successfully. To see the exit code, use the `-verbose` option.

Saving GUI Settings

By default, Conformal *does not* automatically save GUI settings for future sessions. To save your preferred settings, use the GUI *Exit* window and click the *Save GUI settings* check box.

Refer to the *Conformal Extended Checks User Guide* for the procedure to save GUI settings.

Tcl Command

`exit`

Parameters

<code>-Force</code>	Exits without requiring an exit confirmation.
<code>-Verbose</code>	Displays exit codes on exiting Conformal Extended Checks.

EXPLORE

EXPLORE

```
< <id | pathname>
| < -SOURCE_DATA | -DESTINATION_DATA
| -MUX_enable | -SINGLE_bit_change
> <from-instance> <to-instance>
>
[-TIME_unit <INFINITY | depth#> | -COMbinational]
[-NOCHECK_constraint | -CHECK_constraint]
(Verify Mode)
```

Explores either pre-defined properties or CDC functional checks in an environment that has dynamic constraints. Using the EXPLORE command, you can explore results directly in Verify mode. Use the ADD DYNAMIC CONSTRAINTS command to specify constraints you want to use during this proof process.

Tcl Command

explore

Parameters

id pathname	Explores the pre-defined property with the specified ID or path.
-SOURCE_DATA	<p>Explores the specified path that failed the Source Data Functional CDC Check.</p> <p>The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable in order for the destination register to latch it.</p>
-DESTINATION_DATA	<p>Explores the specified path that failed the Destination Data Functional CDC Check.</p> <p>The Destination Data Functional CDC Check determines whether data entering the destination register is held stable in order for the destination register to latch it.</p>

Conformal Extended Checks Reference Manual

Command Reference

-MUX_enable	Explores the specified path that failed the MUX Enable Functional CDC Check. The MUX Enable Functional CDC Check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.
-SINGLE_bit_change	Explores the specified path that failed the Single-bit Change Functional CDC Check. The Single-bit Change Functional CDC Check determines whether a vector of signals crossing different clock domains can be changed only one bit at a time.
from-instance	Explores the CDC path that has the specified beginning.
to-instance	Explores the CDC path that has the specified ending.
-TIME_unit	Explores the added properties to the specified sequential depth. INFINITY Explores the added properties to depth=infinity. depth# Explores the added properties to the specified depth (integer).
-COMbinational	Explores the added properties with the combinational method.
-NOCHECK_constraint	Specifies that Conformal will not perform constraint consistency checks before exploring the specified properties. <i>This option is the default.</i>
-CHECK_constraint	Specifies whether Conformal performs constraint consistency checks before exploring the specified properties.

Examples

The following is a set of sample commands that shows this and related commands in context. The following set of commands assumes that you have read in your library, design, and have switched to the Verify mode.

1. Use the `ADD STATIC PROPERTY` command to add specified extracted properties to the prove list.

```
VERIFY> add static property -bus
```


Conformal Extended Checks Reference Manual

Command Reference

```
//1 property added
```

2. Use the PROVE command to verify the specified properties.

```
// Proving 1 properties (out of 1)...
```

Property Status Summary @ Infinity					
Assertion Property	Pass	Fail	ED	NR	Total
Bus	0	1	0	0	1
All Properties	Pass	Fail	ED	NR	Total
Total	0	1	0	0	1

3. Use the ADD DYNAMIC CONSTRAINT command to add a dynamic constraint to the design.

```
VERIFY> add dynamic constraint ONEHOT E1 E2
```

4. Use the REPORT DYNAMIC CONSTRAINTS to report the dynamic constraints in the design.

```
VERIFY> report dynamic constraints
```

Dynamic Constraint List	
OneHot:	
E1 (Id: 10)	
E2 (Id: 9)	

5. Use the EXPLORE command to explore the pre-defined properties and CDC functional checks in an environment that has dynamic constraints.

```
VERIFY> explore -check_constraint tmp$BUS
```

Explored Results	
Bus Property 6 is *Pass to Time Inf*	
Gate name:	tmp\$BUS
Type	: Internal BUS
Characteristic:	one-hot

Related Commands

ADD DYNAMIC CONSTRAINTS

DELETE DYNAMIC CONSTRAINTS

REPORT DYNAMIC CONSTRAINTS

GENERATE CCD DOFILE

GENerate CCd Dofile

```
-FILE <output_file>  
[-NO_DEL_CDC_STRuctural]  
[-REPLACE]  
(Verify Mode)
```

Automatically generates a dofile for running clock domain crossing checks in Conformal Constraint Designer (CCD). More information on migrating from LEC-Verify to CCD is available through the Web Interface (through the `SET WEB ON` command).

Tcl Command

```
generate_ccd_dofile
```

Parameters

<code>-file <output_file></code>	Specifies the output file name.
<code>-NO_DEL_CDC_STRuctural</code>	Does not process <code>DELETE CDC CHECK -structural</code> commands. By default, the command converts these commands into <code>filter_paths</code> in CCD.
<code>-REPLACE</code>	Overwrites the output file, if it already exists.

Example

For example:

```
generate ccd dofile -file test1 -replace
```

This command generates a CCD dofile called `test1.ccd` and a corresponding SDC file called `test1.sdc` in the current directory.

Related Commands

[SET WEB INTERFACE](#)

GENERATE FIFO INFO

GENERate FIfO Info
(Verify Mode)

Extracts information for FIFOs. Conformal Verify can extract FIFOs with one or more dimensional memory components.

Note: For this command to work properly, you must have already assigned the read data registers to a clock domain with the `ADD DATA ASSOCIATION` command (see example dofile below).



Before running this command, you must first use the `ADD CDC CHECK` command's `-structural` option to identify the part of the design to search for FIFOs because FIFO detection works on existing clock domain crossings.

Tcl Command

```
generate_fifo_info
```

Example

The following shows an example dofile of user-based FIFO identification:

```
set system mode verify
add data association read_data\[*\] -domain read_clk
add cdc check -structural -source -all -destination -all -from -all -to -all
generate fifo info
validate
```

Related Commands

[ADD FIFO INFO](#)

[DELETE FIFO INFO](#)

[REPORT FIFO INFO](#)

HELP

HELp

```
[<command_name> | <message_name> | -message]
[-Verbose]
(Setup / Verify Mode)
```

Note: Although the `HELP` command is still available, it is recommended that you use the `MAN` command.

Displays the Conformal Extended Checks commands and their command syntax. To display a category of commands, use a keyword such as `add`, `delete`, `report`, or `set`. To display the usage for all commands, use `help -help`.

While in the Tcl mode, the `HELP` command displays a list of all available Conformal Tcl mode commands.

Tcl Command

`help`

Parameters

<code><command_name></code>	Displays the command syntax for a given command name. If you do not specify a command name, Conformal Extended Checks displays all the commands.
<code><message_name></code>	Displays help for the corresponding rule check message.
<code>-message</code>	Displays all rule check messages.
<code>-Verbose</code>	Use this option with a command name to see expanded information about the command, including examples and related commands.

Examples

```
help
help read design
help read design -verbose
help -help
help add
```

Conformal Extended Checks Reference Manual

Command Reference

```
help f5  
help -message
```

Sample Tcl mode system prompt and command:

```
TCL_SETUP> help set_current_module
```

Related Command

[SEARCH](#)

[MAN](#)

HISTORY

HISTory
[int]
[-ALL]
(Setup/LEC mode)

Lists the commands typed in the current session.

Tcl Command

history

Parameters

[int]	Limits the number of commands to return. By default, this command returns the last 20 commands entered.
[-ALL]	List all commands entered in the current session.

INFO CHECKPOINT

INFo CHeckpoint

<checkpoint_file_name>
(Setup / Verify Mode)

Provides more information about the specified checkpoint file.

Tcl Command

info_checkpoint

Parameters

checkpoint_file_name

Specifies the name of the checkpoint file that you want to query.

Related Command

CHECKPOINT

LICENSE

LICense

[<license_string>]
(*Setup / Verify Mode*)

Displays information about the Conformal licenses that you currently have checked out.

Tcl Command

license

Parameters

<license_string> Displays information for a specific license.

Conformal Extended Checks Reference Manual

Command Reference

MAN

MAN

```
[<name>] [-Keyword] [-Verbose]  
[-COLOR | -NOCOLOR] [-PAGE | -NOPAGE]  
(All Modes)
```

This displays the manual pages for a given expression. By default, the MAN command searches through command names, TCL function names, and rule or modeling message IDs for a best match. You can also use the `-Keyword` option to search through descriptions. If an exact match is not found, this command returns a list of all matches. Running this command without any arguments returns a list of all available pages.

Note: In TCL mode, this command is `VPXMODE MAN`.

With the `man` command, results are categorized according to the type of information:

Category	Description
LEC-CMD	LEC Commands
CFM-RULE	HDL Rules
LEC-MODEL	LEC Modeling Rules
LEC-TCL	LEC TCL Commands
ECO-CMD	Conformal ECO Commands
CCD-CMD	Conformal CD Commands
CCD-LINT	Conformal CD Lint Rules
CCD-MODEL	Conformal CDCCD Modeling Rules
CCD-TCL	Conformal CD TCL Commands
SDC-RULE	SDC Rules
CLP-CMD	Conformal Low Power Commands
CPF-RULE	Common Power Format Rules
CLP-RULE	Conformal Low Power Rules
CDC-CMD	Conformal Extended Checks Commands
CDC-MODEL	Conformal Extended Checks Modeling Rules
CDC-TCL	Conformal Extended Checks TCL Commands

Conformal Extended Checks Reference Manual

Command Reference

For example, the following command:

```
CDC> man usage
```

has the following results:

```
USAGE (CDC-CMD)
usage (CDC-TCL)
```

To obtain the man page for the usage Tcl command, use the following command:

```
VERIFY> man usage ccd-tcl
```

Tcl Command

man

Parameters

<name>	Specifies the search expression.
-Verbose	By default, the MAN command display only the first section of a manual page. Use this option to display the entire manual page.
-Keyword	<p>Search for all commands, rules, or messages whose description contains the expression given by <code>name</code>.</p> <p>Using this option, you can also search based through specific sections. For example, the following command searches through all Example sections for the word <code>gated_clock</code>:</p> <pre>MODE> man -k example gated_clock</pre> <p>Keyword searching is done line by line. Therefore, the search expression cannot span over multiple lines.</p>
-COLOR	Displays manual page with color highlighting. <i>This is the default.</i>
-NOCOLOR	Disables color highlighting. You can also use the <code>c</code> keyboard shortcut to toggle color highlighting.

Conformal Extended Checks Reference Manual

Command Reference

-PAGE	<p>Displays manual pages one screen at a time.</p> <p>You can use the following keyboard shortcuts while in MAN:</p> <ul style="list-style-type: none">■ Spacebar—Forward one screen.■ Return Key—Forward one line.■ b Key—Back one screen.■ d Key—Forward 1/2 screen.■ u Key—Back 1/2 screen.■ a Key—Display all the remaining text.■ q—Quit. <p>Note: The displayed output is not saved in the logfile specified by the SET LOG FILE command.</p> <p>The pager is not enabled if the help text is less than a screen, when output is redirected to a file, or when MAN is run in the GUI window.</p>
-NOPAGE	<p>Displays the entire man page at once.</p>

Example

For example, the following command displays the manual page for the REPORT DESIGN DATA command:

```
MODE> man rep de d
```

For example, the following command retrieves all entries with the word `datapath`.

```
MODE> man datapath
ANALYZE DATAPATH (LEC)
REPORT DATAPATH OPTION (LEC)
```

The following lists all pages whose Syntax section contains the word `thread`.

```
MODE> man -k syntax thread
```

The following displays all RTL rules whose default severity is error:

```
MODE> man -k rule default severity error
```

Related Commands

HELP

NCENCRYPT

NCENCRYPT

```
<<input_file>  
<output_file>  
[-VERilog | -VHdl | -TCL]  
[-PRAGMA]  
[-REPlace]  
| -VERSion >  
(All modes)
```

Use the NC Protect protection scheme to encrypt the specified files. This command also works in Tcl mode.

Tcl Command

ncencrypt

Parameters

<input_file>	File to be encrypted.
<output_file>	Name of the encrypted file.
[-VERilog -VHdl -TCL]	Specifies the format of the input file. The default is Verilog.
-PRAGMA	Encrypt only the text between the <code>protect begin</code> and <code>protect end</code> NC Protect pragmas.
-REPLACE	Replaces existing file.
-VERSion	Reports the NC library version.

Related Commands

ncdecrypt (Tcl command)

OPEN SCHEMATICS

OPEn SCheMatics

[<full_pathname>]
(Setup / Verify Mode)

Opens the schematic viewer.



You cannot use this command in the non-graphical mode.

Tcl Command

open_schematics

Parameters

<full_pathname> Opens a schematic for the specified path name.

Related Command

CLOSE SCHEMATICS

PRINTENV

PRINTENV

[<variable>]
(*Setup / Verify Mode*)

Displays environment variable values. If you do not specify a variable, the PRINTENV command displays the value of every environment variable.

Tcl Command

printenv

Parameters

<variable> Prints the value of the specified variable.

Related Command

SETENV

PROPAGATE CLOCK DOMAIN

PROpagate **CL**ock **DO**main
(*Verify Mode*)

Recalculates the clock domain information for the design. Use this command when you must manually update clock domain information.

PROVE

PROve

```
[-Time_unit <INFINITY | <depth#> >  
| -COMbinational]
```

Starts verification of the specified checks. If you do not specify parameters, Conformal Extended Checks proves all added checks with `-time_unit infinity`.

Tcl Command

prove

Parameters

<code>-Time_unit</code>	Proves the added properties to the specified sequential depth.	
	<code>INFINITY</code>	Proves the added properties to depth=infinity.
	<code><depth#></code>	Proves the added properties to the specified depth (integer).
<code>-COMbinational</code>	Proves the added properties with the combinational method.	

Examples

```
prove  
prove -time_unit 32  
prove -combinational
```



Tip

For properties that return a proof status of ED, re-run the proof with greater effort. Conformal Extended Checks remembers the proof results from previous proofs and resumes the proof on only the unproven properties and from the depth that was previously proven. Example:

```
add static property -all // 10 properties  
prove // 5 passed, 5 ED depth 12  
set prove effort high  
prove  
// The 2nd proof begins where the previous proof  
// ended, that is, the 5 EDs at depth 12.
```

Conformal Extended Checks Reference Manual

Command Reference

```
// The result might now be 5 EDs at depth 20.  
// Any subsequent proofs will start on the  
// unproven properties from the depth already  
// proven.
```

Related Command

SET PROVE OPTIONS

READ DESIGN

REAd DEsign

```
<filename>
[-MIXvlog]
[-ROot <module_name>]
[-ROOTONLY]
[-ROOTConfig <configuration_name>]
[-VERilog | -VERILOG2K | -V1995 | -SYStemverilog | -SVA | -SV09
  | | -VHdl [ 93 | 87 | 2008] | -SPice [-NO_RESistor]]
[-NOSTRength]
[-File <command_filename>]
[ | -REPlace | -APPend]
[-Define <name>]
[-Map <library_name> <library_path>]
[-MAPRecursive <library_name> <library_path>]
[-MAPFile <library_name> [=<library_name2>] <filename ...>]
[-LIBRARY <library_name> <library_path>] (this option is the same as -Map)
[-STATEtable | -NOSTATEtable]
[ | -BBOXUNResolve | -NOBBOXEMpty]
[-BLAST_inst_port]
[-RANgeconstraint | -NORANgeconstraint]
[-INITial_value]
[-VHDLESCaped_to_verilog]
[-CONTINUOUSASSIGNment <BiDirectional | UNIdirectional>]
[-SUPPLY | -NOSUPPLY]
[-UNCompress <zip_file_name>]
[-UNZip <zip_file_name ...>]
[-PARAMeter [-INT | -STR | -ENUM | -TYPE] <name> <value>]
  (combined with -ROot option)
[-ARCHitecture <architecture_name>]
[-FUnctiondefault [0 | 1 | x]]
[-Keep_unreach | -NOKeep_unreach]
[-SEnsitive | -NOSEnsitive]
[-CONFIguration | -NOCONFIguration]
[-NOELaborate]
[-NOREName]
[-EXclude <exclude_file_name*>]
[-KEEP_ESCAPED_ID]
[-VERBose]
[-OPTimize | -NOOPTimize]
[ | -LAsTmod | -OVERWrite_mod]
[-LOCAlref]
[-LOGIC_ENCODING_OFF]
[-MERge BBox]
[-INSERT_FEEDTHROUGH_buffer]
[-SUMmary_report | -NOSUMmary_report]
(Setup Mode)
```

Conformal Extended Checks Reference Manual

Command Reference

Reads and parses an HDL design intended for verification by Conformal Extended Checks. The design can consist of one or more files containing synthesizable Verilog or VHDL.

If you use the `ADD SEARCH PATH` command before this command, Conformal Extended Checks uses your specified search path to locate files by name.

Note: Use the tilde "~" character in the file's path to replace the path to the user login home directory.

Prerequisites

- If the design has undefined cells, Conformal Extended Checks reports them as errors, unless you used the `SET UNDEFINED CELL` command before reading the design.
- If the design references modules from libraries, you must read the libraries before reading the design. Use the `READ LIBRARY` command to read the libraries.
- If Conformal Extended Checks finds duplicate modules, it uses the first module listed and ignores later ones. However, you can use the `-lastmod` option to specify that Conformal Extended Checks use the last module and ignore earlier ones.

Verilog Notes

- Use the `-file` command option to specify a Verilog command file list. (See the table in the [READ DESIGN](#) command for supported options.) In addition, you must add the `-yd` option to treat library modules as design modules.
- Conformal Extended Checks does not support multiple uses of the `-file` command option.

VHDL Notes

- If Conformal Extended Checks finds duplicate architectures, it uses the first architecture listed, ignores later ones, and issues a warning message.
- Conformal Extended Checks supports most synthesizable VHDL constructs and all standard and IEEE packages. Conformal Extended Checks issues warning messages for all non-synthesizable constructs.

VHDL and Verilog 2001 Library Mapping

You can specify how VHDL and Verilog 2001 libraries are mapped using the `READ DESIGN` command's `-map`, `-mapfile`, or `-library` options.

Conformal Extended Checks Reference Manual

Command Reference

The `-map` and `-library` options work the same in that they map logical library names to physical directories. You can use multiple `-map` commands to map multiple physical directories to one logical library. Use the `-mapfile` option for more specific library mapping, such as specifying that a list of files must be compiled into a specified library. If you read in a file without specifying its library mapping, that file is stored in a default library called `work`.

Note: You can map a file into more than one library. In this case, the file is stored in each library for which it is mapped.

See the "VHDL Support" and "Verilog Support" appendices in the *Conformal Equivalence Checking User Guide*.

IP Protection

Conformal can read in files that are encrypted using the `ncprotect` utility (available with the Cadence® NC-Verilog Simulator and Cadence® NC-VHDL Simulator) or Cadence encryption tools. Designs that are encrypted can be read into Conformal and compared with other non-encrypted designs.

When there are protected modules, the `WRITE DESIGN` command will write out the full netlist of the protected modules in either clear text or encrypted text, depending on the level of protection specified when the file was encrypted (for example, the `output_netlist` attribute of NC-Protect specifies the encryption level of the output netlist).

Note: Conformal does not support Cadence Verilog-XL protected files (they are only for simulations), and any other proprietary encryption files.

For more information on protection and synthesis rights, refer to the *Protecting IP Source Files* document of the Cadence simulation tool.

Tcl Command

`read_design`

Parameters

<code><filename></code>	Reads the specified design file.
-------------------------------	----------------------------------

Conformal Extended Checks Reference Manual

Command Reference

-MIXvlog	Automatically reads Verilog source file in Verilog 1995, Verilog2K, or SystemVerilog standard according to the file extension. Any other extensions not specified by <code>SET RTL TYPE</code> command will be read according to the language option specified in <code>READ DESIGN</code> command.
-ROot <module_name>	Identifies the top root module.
-ROOTONLY	Elaborates the root module only and skips elaboration of the other modules that are not instantiated from the root module. Because the elaboration stage is skipped for the uninstantiated modules, this option reduces memory usage and omits the elaboration time error checking.
-ROOTConfig <configuration_name>	<p>The design includes the specified configuration for the top-level module.</p> <p>Note: This option applies to VHDL designs, only. Use this option when the design includes multiple configurations for the top-level module. When you use the <code>-rootconfig</code> option, you must also use the <code>-root module_name</code> option (above).</p>
-VErilog	Specifies that this design is a Verilog design. (Use this option for Verilog designs that comply with IEEE 1364-2001.) <i>This is the default.</i>
-VERILOG2K	Specifies that this design is a Verilog2K design (Use this option for Verilog designs that comply with IEEE 1364-2001).
-V1995	Specifies that this design is a Verilog 1995 design (Use this option for Verilog designs that comply with IEEE 1364-1995).
-SYStemverilog	Specifies that this design is a SystemVerilog design.
-SVA	Enables SystemVerilog Assertion (SVA) support.
-SV09	Specifies that this design is a SystemVerilog 1800-2009 design.
-VHdl	Specifies that this design is written in VHDL with the specified standard:
93	VHDL-93 (Use this option for VHDL designs that comply with IEEE Std 1076-1993.) <i>This is the default.</i>
87	VHDL-87 (Use this option for VHDL designs that comply with IEEE Std 1076-1987.)

Conformal Extended Checks Reference Manual

Command Reference

	2008	<p>VHDL-2008 (Use this option for VHDL designs that comply with IEEE Std 1076-2008.)</p> <p>VHDL 2008 cannot be used with other versions of VHDL as some standard libraries are defined differently.</p> <p>Note: The Conformal software supports multiple uses the <code>-vhdl 93</code> and <code>-vhdl 87</code> options. See the example.</p>
<code>-SPice</code>		Specifies that this design is a SPICE netlist design.
<code>-NO_RESistor</code>		Removes resistor instances specified in the SPICE netlist.
<code>-NOSTrength</code>		Specifies that drive strengths are automatically ignored without being removed from design.
<code>-File <command_filename></code>		<p>Reads in the specified command file as a design.</p> <p>Note: This option is for Spice, Verilog, or VHDL command file lists.</p> <p>The options for <code><command_filename></code> are described in Table 2-1 on page 198 for Verilog and Table 2-2 on page 202 for VHDL.</p>
<code>-REPlace</code>		Removes all designs that were previously read in, and replaces them with the specified design.
<code>-APPend</code>		<p>Appends the design to the one that was previously read.</p> <p>For example, you can use this option to fix a top module and then read it in again without parsing the entire design file again:</p> <pre>read design top.v -append -lastmod</pre> <p>The top module cannot pass parameters to modules that are read in previously.</p>
<code>-Define <name></code>		<p>Defines <code>`ifdef</code> variable names in Verilog. To specify multiple definitions, use this option for each definition you want to set. For example:</p> <pre>read design filename -define definition1=value1 \ -define definition2=value2</pre> <p>The macro redefinition in Verilog source code is ignored if the text macro is specified at the Conformal command line or Verilog command file.</p>

Conformal Extended Checks Reference Manual

Command Reference

`-Map <library_name> <library_path>`

Reads in files for the specified `library_name` from `library_path`.

Use this option to read in all the VHDL files in the specified library path for the given library name. You can also map multiple library paths to a single library. For example:

```
read design -vhd1 top.vhd -map mylib /design/path1 \  
                                     -map mylib /design/path2
```

`-MAPRecursive <library_name> <library_path>`

This option has same function as `-Map`, but it searches for all VHDL files recursively down to the subdirectories of the `<library_path>`.

`-Map` searches VHDL files under the `<library_path>` and will not search any VHDL files under the subdirectories of `<library_path>`.

`-MAPFile <library_name> [=<library_name2>] <filename ...>`

Reads in the specified file(s) and includes them in the specified library.

Use this option to specify the files that belong to a given library. The file list terminates with the next option or the end of the `READ DESIGN` command.

You can also use multiple `-mapfile` options to specify multiple files in a library. For example, the following two commands are the same:

```
read design -vhd1 top.vhd -mapfile \  
mylib x1.vhd -mapfile mylib x2.vhd  
read design -vhd1 top.vhd -mapfile mylib x1.vhd x2.vhd
```

You can specify multiple aliases (or names) for a library using the equal sign (=). The real library should be specified last. For example, the following command reads in `test.vhd` and includes it in `lib1`; `lib2` and `lib3` are aliases of `lib1`:

```
read design -vhd1 -mapfile lib3=lib2=lib1 test.vhd
```

In this example, `lib1` is the real library; `lib2` and `lib3` are merely aliases.

`-LIBRARY <library_name> <library_path>`

Conformal Extended Checks Reference Manual

Command Reference

	Reads in the specified file in the given library and path for user-defined VHDL libraries. (This option is the same as <code>-map</code> .)
<code>-STATETable</code>	<p>Enables support for Synopsys Liberty state tables. <i>This is the default.</i></p> <p>Note: This option supersedes the <code>SET STATETABLE</code> command.</p>
<code>-NOSTATETable</code>	Disables support for Synopsys Liberty state tables.
<code>-BBOXUNResolve</code>	Specifies that unresolved semantics as unsupported constructs (in VHDL) will be blackboxed instead of erroring out.
<code>-NOBBOXEMpty</code>	Specifies that empty modules will be retained instead of being blackboxed.
<code>-BLAST_inst_port</code>	<p>Allows the cell model to have a bus pin while the instantiation is bit-blasted.</p> <p>By default, instantiations that contain bit-blasted connections error out. The Verilog standard does not allow these connections.</p>
<code>-RAngeconstraint</code>	<p>When a variable is of type integer with a value range, a value check is made against the range. If a value is out of range, it will be interpreted as don't care. <i>This is the default.</i></p> <p>For example:</p> <pre>variable v : integer range 3 to 5;</pre> <p>With <code>-RAngeconstraint</code>, <code>v</code> will be interpreted as:</p> <pre>((v>=3 && v<=5)? v : 3'bx).</pre> <p>Note: This option applies only to VHDL designs.</p>
<code>-NORAngeconstraint</code>	<p>Specifies that no value check is made against the integer value range.</p> <p>Note: This option applies only to VHDL designs.</p>
<code>-INITial_value</code>	<p>Specifies that the variable's initial value will not be ignored.</p> <p>Note: This option is for VHDL only.</p>
<code>-VHDL ESCaped_to_verilog</code>	

Conformal Extended Checks Reference Manual

Command Reference

Specifies that if the content between '\ ' pairs does not contain any white space, the new name is the escaped content.

For example, \A_B_C_ \ changes to \A_B_C_. In all other cases, the VHDL escaped name is unchanged. For example, \1 2 \ is unchanged.

`-CONTINUOUSASSIGNment <BIdirectional | UNIdirectional>`

Specifies that continuous assignment in the design should be interpreted as bi-directional or uni-directional assignment.

`-SUPPLY`

Keeps all Verilog `supply0` and `supply1` type nets unchanged. *This is the default.*

`-NOSUPPLY`

Converts the Verilog `supply0` and `supply1` type nets to Verilog wire type nets.

`-UNCompress <zip_file_name>`

Reads in the specified compressed file. By default, the Conformal software uses the `gunzip` tool to uncompress the file into the `/tmp` directory. Files created with `gzip` or `compress` are supported.

If the compressed file cannot be uncompressed using the `gunzip` tool, you specify another tool by setting UNIX variable `CONFORMAL_UNCOMPRESS`.

You can also set the UNIX variable `CONFORMAL_TMP` to a path other than the default `/tmp` directory.

`-UNZip <zip_file_name ...>`

This option has the same function as `-UNCompress` except that it can include a list of filenames. For example:

```
read design fileABC -UNZ zip1 zip2 zip3
```

The list of filenames end when a subsequent option is specified, or if it is at the end of the command line.

Note: Specifying `-uncompress` or `-unzip` is optional for gzipped files because the Conformal software can automatically recognize the file type created by the `gzip` tool. This includes command file lists specified with `-file`.

Conformal Extended Checks Reference Manual

Command Reference

`-PARAMeter [-INT | -STR | -ENUM] name value`

Assigns design parameters or replace existing design parameters. To specify multiple parameters, use the `-parameter` option for each parameter you want to set. For example:

```
read design filename -parameter parm1 value1 \  
-parameter -int parm2 value2
```

This option applies to both Verilog and VHDL files. (Combine with `-root`.)

When using the `-parameter -int <parameter_name> <value>` command, the `<value>` will be converted to integer value, which can be a positive integer (1), negative integer (-1), an integer value recognized as a string ("1" / "-1"), or a Verilog style integer ("16'h0001"). When using a Verilog style integer, the value must be specified between double-quotes (" ").

When using the `-parameter -str <parameter_name> <value>` command, the `<value>` will be saved as a string.

When using the `-parameter -enum <parameter_name> <value>` command, the `<value>` will be converted to a VHDL enumeration literal. For example, the following command sets the parameter P4 to VHDL enumeration literal GREEN:

```
read design -root mod1 filename \  
-parameter -enum P4 GREEN
```

When using the `-parameter -type <parameter_name> <value>` command, `<value>` is a Verilog/SystemVerilog built-in type or user-defined type compiled in the compilation unit scope `$unit`—types defined in a module or package are not supported. For example, the following command sets the parameter P2 to SystemVerilog `shortint` type:

```
read design -root mod1 filename -parameter \  
-type P2 shortint
```

The format "`$unit::xxx`" is not supported for `<value>`.

Conformal Extended Checks Reference Manual

Command Reference

Notes:

Any value that is not recognized as an unsigned decimal integer value is interpreted as string value.

If `-int` or `-str` is not specified, then the parameter value will be interpreted as an integer if it is not between double-quotes (" "), and as a string if it is between double-quotes. Therefore, if you want to specify a Verilog format value, it must be between double-quotes and used with the `-int` option.

<code>-ARchitecture</code>	<code><architecture_name></code>	Reads the specified design <code>architecture_name</code> .
<code>-FUnctiondefault</code>		Specifies the default return value for unspecified or incompletely specified functions
	0	Returns zero.
	1	Returns one.
	x	Returns x.
<code>-Keep_unreach</code>		Maps all unreachable key points. Unreachable key points are those that do not eventually affect the primary outputs of the design. <i>This option is the default.</i>
<code>-NOKeep_unreach</code>		Does not map unreachable key points.
<code>-SEnsitive</code>		Specifies that the design is case sensitive.
		The default case sensitivity is set by the input language. For example, the default case sensitivity for VHDL input is <code>-NOSEnsitive</code> , and the default case sensitivity for Verilog input is <code>-SEnsitive</code> .
<code>-NOSEnsitive</code>		Does not read the design as case-sensitive.
<code>-CONFiguration</code>		Supports V2K, SystemVerilog, and VHDL configuration constructs. <i>This is the default.</i>
<code>-NOCONFiguration</code>		Does not interpret V2K, SystemVerilog, and VHDL configuration configurations.
<code>-NOELaborate</code>		Reads in multiple files of different languages.

Conformal Extended Checks Reference Manual

Command Reference

-NOREName	<p>Does not rename duplicate pin/port names.</p> <p>For example, without this option, the duplicate port <code>dout</code> will be renamed to <code>Port4</code>:</p> <pre>module test (dout, clk, din, dout); input clk, din; output dout; endmodule</pre>
-EXClude <exclude_file_name*>	<p>Specifies files to exclude when reading in the design. This accepts the wildcard.</p> <p>Note: You cannot use multiple wildcards with this option.</p>
-KEEP_ESCAPED_ID	<p>When used, this option keeps escaped identifiers, as in Verilog 2001. See "Support for Macro Expansions" in the <i>Conformal Constraint Extended Checks User Guide</i> for more information.</p>
-VERBose	<p>Displays additional messages during execution.</p>
-OPTimize	<p>Optimizes redundant logic (in library cells) that affects the way Conformal Extended Checks interprets the design.</p> <p>Note: Using this option does not <i>a/ways</i> optimize <i>a//</i> redundant logic.</p> <p>See the following example:</p> <pre>read design file1 file2 file3 -nooptimize -optimize \ -nooptimize -replace</pre> <p><i>This option is the default.</i></p>
-NOOPTimize	<p>Preserves redundant logic in Library cells.</p> <p>See the example listed above (<code>-optimize</code>).</p>
-LAsTmod	<p>If duplicate modules exist, Conformal uses the first module and ignores the later ones by default. Use this option to specify that Conformal use the last module and ignore earlier ones.</p>

Conformal Extended Checks Reference Manual

Command Reference

<code>-LOGIC_ENCODING_OFF</code>	<p>Does not interpret the user-defined <code>enum</code> types with literal value 0, 1, X, Z, and so on as one-bit logic values.</p> <p>For example, <code>MY_MVL</code> is the user-defined <code>enum</code> type:</p> <pre>type MY_MVL is ('X', '0', '1', 'Z');</pre> <p>The type of signal <code>val</code> is <code>MY_MVL</code>. For the statement:</p> <pre>val <= 'Z';</pre> <p>With this option, <code>val</code> is 2'd3; without this option, <code>val</code> is 1'bx.</p>
<code>-OVERWrite_mod</code>	<p>All module names must be unique within a given library, and same module name can exist in the different libraries. Use this option to keep the current module and delete all existing modules with the same name from all libraries. Without this option, only the module with the same name within the target library is affected.</p>
<code>-LOCalref</code>	<p>Keep all Verilog modules as local private modules if they are referenced by any module within the same Verilog file. Local private modules cannot be referenced from other Verilog files.</p>
<code>-MErge BBox</code>	<p>Replaces all blackboxed modules in the design space with modules in the library space.</p>
<code>-INSERT_FEEDTHROUGH_buffer</code>	<p>When a continuous assignment has a signal that inputs into a submodule and outputs directly, the tool will interpret this as a <code>through wire</code> (no direction). LEC will insert a buffer to indicate the wire direction.</p>
<code>-SUMmary_report</code>	<p>Print out HDL rule check messages while reading the library or design file(s). <i>This is the default.</i></p>
<code>-NOSUMmary_report</code>	<p>Does not print out HDL rule check messages while reading the library or design file(s).</p>

Table 2-1 Supported Verilog Command-Line Options

The following table lists the Verilog command options that Conformal supports.

<code><file></code>	(Design data file)	A list of the design files.
---------------------------	--------------------	-----------------------------

Conformal Extended Checks Reference Manual

Command Reference

<code>-v <file></code>	(Library file)	<p>A list of library files.</p> <p>Refer to “Search Path Order when using Verilog Command Options.” following this table for more information.</p>
<code>-y <directory></code>	(Library directory)	<p>A list of library directories.</p> <p>Conformal searches for modules not defined in the design files and reads in these modules as library modules.</p> <p>Refer to “Search Path Order when using Verilog Command Options.” following this table for more information.</p>
<code>+incdir+<dirname>...</code>	(Include directories)	<p>A list of directories used to resolve the Verilog files that have <code>`include</code> directives.</p> <p>Refer to “Search Path Order when using the <code>`include</code> Directive.” following this table for more information.</p>
<code>+libext+<extension>...</code>	(Library extensions)	<p>A list of library extensions you can include when using the <code>-y</code> option.</p> <p><i>The default is <code>.v</code>.</i></p>
<code>+define+<macro_name></code> ...	(Define macros)	<p>A list of macro names you can include in the <code>`define</code> statement</p>
<code>-yd <directory></code>	(Design directory)	<p>A list of directories.</p> <p>Conformal searches for modules that are not defined in the design files and reads these modules in as design modules.</p> <p>Refer to “Search Path Order when using Verilog Command Options.” following this table for more information.</p>

Conformal Extended Checks Reference Manual

Command Reference

<code>+search_yd_path</code>	Enable Conformal to search for files in the directories specified by the <code>-yd</code> option.
<code>-f <file></code>	(another command file)

Search Path Order when using Verilog Command Options

Verilog command options used to resolve Verilog module references (such as `-y/-yd`, `-v/-vd`, and `+libext+<ext_suffix>`) can also affect the search path order. When a Verilog module instance cannot be resolved to any existing modules in memory, the Verilog parser searches for the module from the directories (`-y/-yd`) or files (`-v/-vd`) specified by these options. The order in which module references are resolved is the order specified in the Verilog command file.

To search for files in the directories specified by the `-yd` option in the Verilog command file, use the `+search_yd_path`. The additional search paths specified by `-yd` are effective only in the current `READ DESIGN` command.

For example, a Verilog command file named `gol.f` contains:

```
+search_yd_path
+incdir+include
-yd hdl
-yd test
mux.vs
```

In `LEC`, `READ DESIGN -f gol.f` will search for `mux.vs` in the following directories (in the order listed):

- Current working directory "."
- `-yd hdl`
- `-yd test`
- Any other paths specified by `ADD SEARCH PATH`

Search Path Order when using the ``include` Directive

When a Verilog include file is specified using ``include` directive, the file search order is as follows:

1. Current working directory
2. Directories specified by `+incdir` option in the Verilog command file

Conformal Extended Checks Reference Manual

Command Reference

3. Directory that contains the file that specifies the ``include` directive
4. Search paths added by the `ADD SEARCH PATH` command
5. Directories specified by `-yd` in the Verilog command file
6. Directories specified by `-y` in the Verilog command file

When using the command `SET HDL OPTIONS -INCLUDE_SRC_DIR OFF` (note that the default is `ON`), which disables and prevents LEC from reading include files in the source's relative path, the search order becomes as follows.

1. Current working directory
2. Directories specified by `+incdir` option in the Verilog command file
3. Search paths added by the `ADD SEARCH PATH` command
4. Directories specified by `-yd` in the Verilog command file
5. Directories specified by `-y` in the Verilog command file

To customize the search order, use the `SET HDL OPTIONS -verilog_include_dir` command option.

Table 2-2 Supported VHDL Command-Line Options

The following table lists the VHDL command options Conformal supports.

Option	Definition
<file>	Individual design file
-map <libname> <dirname>	Map logical library name to a directory
-mapfile <libname> <filename ...>	Map logical library name to a list of files
-library <libname> <dirname>	Map logical library name to a directory

Example

In the following example, only `ent2.vhdl` and `arch2.vhdl` will be parsed according to VHDL 87 syntax rules. All the other files will be parsed according to VHDL 93 syntax rules.

```
read design ent0.vhdl arch0.vhdl \  
    -vhdl 93 ent1.vhdl arch1.vhdl \  
    -vhdl 87 ent2.vhdl arch2.vhdl \  
    -vhdl ent3.vhdl arch3.vhdl
```

Related Commands

ADD BLACK BOX

ADD NOTRANSLATE MODULES

ADD SEARCH PATH

READ LIBRARY

REPORT DESIGN DATA

REPORT LIBRARY DATA

SET DIRECTIVE

SET HDL OPTIONS

Conformal Extended Checks Reference Manual

Command Reference

SET NAMING RULE

SET ROOT MODULE

SET RTL TYPE

SET STATETABLE

SET UNDEFINED CELL

WRITE DESIGN

READ FSM ENCODING

READ Fsm Encoding

```
<filename>  
[-Append | -Replace]  
(Setup Mode)
```

Directs Conformal to read in a file that defines new Finite State Machine (FSM) encoding.

By default, Conformal reads binary encoding when building an FSM. Therefore, if your gate netlist uses different encoding (for example, one-hot), you must use the `READ FSM ENCODING` command to specify the correct encoding.

The following is an example of the contents of an encoding file:

```
.name MYENC  
.fromstates u1/current_state_reg[1] u1/current_state_reg[0]  
.tostates u1/current_state_reg[3] \  
u1/current_state_reg[2] \  
u1/current_state_reg[1] \  
u1/current_state_reg[0]  
.begin  
00 0001  
01 0010  
11 0100  
.end
```

Tcl Command

`read_fsm_encoding`

Parameters

<code><filename></code>	Reads in the specified file. This option is a required filename that contains the encoding differences.
<code>-Append</code>	Appends specified file to any existing files. <i>This option is the default.</i>
<code>-Replace</code>	Replaces any existing files.

READ INITIAL STATE

REAd INitial State

```
<filename>
[-SEquence [-NOSAVE_diagnosis_info | -SAVE_diagnosis_info]
  | -VCD [-ROOT instance_pathname]
  [-TIME units]
  [-SNAPshot <filename> [-NOReplace | -Replace] ]
]
(Setup Mode)
```

Specifies a file containing an initialization sequence or a VCD file. Conformal Extended Checks uses the specified file to establish the initial state values for the design.

For an initialization sequence file, each line in the file consists of:

```
<time> <value> < <cell_name*>...
  |< <type_specifier> < |cell_name*>... >...
  >
```

If you include only the time on a line, simulation is forced to advance to the specified time unit without changing the input values.

Note: The values specified in the initialization sequence file do not need to follow chronological order.

Tcl Command

```
read_initial_state
```

Parameters

For the normal initialization sequence flow, the argument are described as follows:

<filename>	Specifies the name of the file containing the initial state definitions.
-SEquence	Specifies the file is an initialization sequence file. <i>This option is the default.</i>
-NOSAVE_diagnosis_info	Does not save the diagnosis information when running design initialization. <i>This option is the default.</i>

Conformal Extended Checks Reference Manual

Command Reference

	<code>-SAVE_diagnosis_info</code>	Saves the diagnosis information when running design initialization.
<code>-VCD</code>		Specifies that the file is a VCD file.
	<code>-R0ot <instance_pathname></code>	This level in the VCD file is used as the root level. <i>The default is the top-level module.</i>
	<code>-TIME units</code>	The point in time at which the design is considered initialized
	<code>-SNAPshot <filename></code>	Writes a reduced VCD file for later use with Conformal Extended Checks. This reduced VCD file stores only the information needed for the initialization time you specified and can be dramatically smaller than a normal VCD file.
	<code>-NOReplace</code>	Does not replace an existing snapshot with the new snapshot. <i>This option is the default.</i> Conformal Extended Checks issues a warning if you use a filename that already exists.
	<code>-Replace</code>	If the snapshot file exists, replace its contents.

Example

```
read initial state init.seq
read initial state myinit.vcd -vcd -root bt_mod1 -time 750
```

init.seq Example

```
0 0 reset
2 1 reset
8 0 reset
0 0 tc_en
0 z datain
```

Related Commands

ADD INITIAL STATE

DELETE INITIAL STATE

REPORT GATE

REPORT INITIAL STATE

READ LEF FILE

REAd LEf File

```
<filename>  
[-MACRO_PINS_ONLY]  
[-SUMmary_report | -NOSUMmary_report]  
(Setup Mode)
```

Reads in the Library Exchange Format (LEF) file, which contains the design's library information.

During design elaboration, Conformal checks whether each cell used in the specified design has a corresponding cell in the library. This command reads in the LEF file and checks that the port declarations are in both the LEF and library. If a power/ground port declaration exists only in the LEF file, Conformal will add the port to the library cell.

Tcl Command

```
read_lef_file
```

Parameters

<filename>	Specifies the name of the LEF file.
-MACRO_PINS_ONLY	Reads in the LEF file, but uses a line-by-line parser to report information only on macros and pins.
-SUMmary_report	Print out HDL rule check messages while reading the library or design file(s). <i>This is the default.</i>
-NOSUMmary_report	Does not print out HDL rule check messages while reading the library or design file(s).

Related Commands

READ DESIGN

READ LIBRARY

READ LIBRARY

READ Library

```
< <filename> | -CPF <cpf_file_name> ... >
[-Verilog | -VERILOG2K | -V1995 | -SYStemverilog | -SVA | -SV09
 | -VHdl [93 | 87 | 2008] | -Liberty [-LP [ALL | STD] [1.1 | 2.0]]
 | -CPF [-LP [ALL | STD] [1.1 | 2.0]] | -SPice [-NO_RESistor]]
[-NOSTRength]
[-File <command_filename>]
[-SEnsitive | -NOSEnsitive]
[-EXtract]
[ | -REPlace | -APPend]
[-Define <var_name>]
[-Map <library_name> <library_path>]
[-MAPRecursive <library_name> <library_path>]
[-MAPFile <library_name> [=<library_name2>] <filename ...>]
[-CONFIguration | -NOCONFIguration]
[-KEEP_ESCAPED_ID]
[-VERBose]
[-SUPPLY | -NOSUPPLY]
[-UNCompress <zip_file_name>]
[-UNZip <zip_file_name ...>]
[-OPTimize | -NOOPTimize]
[-STATEtable | -NOSTATEtable]
[-RESPECT_timing_type_for_flop]
[-LAsmod]
[-LOGIC_ENCODING_OFF]
[-MULTIPLE_LIBraries]
[-CIRcuit_view]
[-NOELaborate]
[-BBOXSolver]
[-EXclude <exclude_file_name*>]
[-NOKeep_unreach | -Keep_unreach]
[-SUMmary_report | -NOSUMmary_report]
(Setup Mode)
```

Reads in the library model descriptions for Verilog or VHDL designs. The library is either a Verilog simulation library or a Synopsys Liberty library.

If there are duplicate modules, the Conformal software uses the first module and ignores the later ones. However, you can use the `-lastmod` option so that the software uses the last module and ignores earlier ones.

Note: Library in this context refers to the technology library, such as ASIC cell and memory definitions. See [READ DESIGN](#) for information on reading VHDL libraries and packages.

Note: Use the tilde "~" character to shorten the specified path.



You must use this command before the `READ DESIGN` command if the design is Verilog or VHDL.

VHDL and Verilog File Encryption

Conformal tools support Verilog and VHDL files which are encrypted by the NC-Protect and Cadence encryption Cadence tools. NC-Protect is available from Cadence NC-VHDL and Verilog simulators. Cadence encryption is Cadence proprietary tool.

The Cadence Verilog-XL protected files are for simulations only and are unsupported by Conformal tools. Other proprietary encryption files are unsupported by Conformal tools.

Tcl Command

`read_library`

Parameters

<code><filename></code>	A required filename that contains the Verilog simulation library or the Synopsys Liberty library.
<code>-CPF <cpf_file_name> ...</code>	Reads in a CPF file that contains Library definitions which define locations of the Liberty library and contains CPF file definitions for the low power library cells, such as level shifters and isolation cells.
<code>-V<code>erilog</code></code>	The library file contains the Verilog library model descriptions. <i>This option is the default.</i> Note: This supports NC-Protect and Cadence encryption.
<code>-VERILOG2K</code>	Contains Verilog2k library model descriptions.
<code>-V1995</code>	Contains Verilog 95 library model descriptions.
<code>-SYStemverilog</code>	Contains SystemVerilog library model descriptions.
<code>-SVA</code>	Enables SystemVerilog Assertion (SVA) support.
<code>-SV09</code>	Contains SystemVerilog 1800-2009 library model descriptions.

Conformal Extended Checks Reference Manual

Command Reference

-VHdl	This library is written in VHDL. The VHDL file is of the specified standard:	
	93	VHDL-93 (Use this option for VHDL designs that comply with IEEE Std 1076-1993.) <i>This is the default.</i>
	87	VHDL-87 (Use this option for VHDL designs that comply with IEEE Std 1076-1987.)
	2008	VHDL-2008 (Use this option for VHDL designs that comply with IEEE Std 1076-2008.) VHDL 2008 cannot be used with other versions of VHDL as some standard libraries are defined differently.
	Note: The VITAL format is unsupported.	
	Note: This supports NC-Protect and Cadence encryption.	
-Liberty	Specifies that the library filename is in the Synopsys Liberty format. This file can contain functional and low power attributes.	
	Notes:	
	This supports Cadence encryption.	
	Supports LDB library files. LDB files contain binary database information and compressed Liberty source text. Conformal will independently parse the compressed Liberty source text portion of the LDB file.	
	The <code>-LP</code> option is for Conformal Low Power. This option extracts low power attributes into library power intent database.	
	<code>-LP ALL</code>	Use Liberty attributes to extract the standard cells (isolation, level shifter, always on, retention, power switch, ground switch) and the IO Pad, and macro model cells into the Conformal Low Power library and macro model power intent database. <i>This is the default.</i> For example, when you specify only <code>read library -liberty -lp</code> , the tool uses <code>ALL</code> by default.

Conformal Extended Checks Reference Manual

Command Reference

-LP	STD	Use Liberty attributes to extract standard low power cells (isolation, level shifter, always on, retention, power switch, and ground switch cells) into the Conformal LP library power intent database.
	1.1	Map the extracted cells to CPF 1.1 features. <i>This is the default.</i>
	2.0	Map the extracted cells to CPF 2.0 features.
	-CPF	Specifies that filename is a CPF file which contain references to Liberty file paths defined by CPF library set (<code>library_set</code>) objects.
		The <code>-LP</code> option is for Conformal Low Power. This option allows extraction of low power cells defined by Liberty attributes.
-LP	ALL	Extracts the standard cells (isolation, level shifter, always on, retention, power switch, ground switch) and the IO Pad, and macro model cells into the Conformal Low Power library and macro model power intent database.
		<i>This is the default.</i> For example, when you specify only <code>read library -liberty -cpf</code> , the tool uses <code>ALL</code> by default.
	STD	Extracts standard low power cells (isolation, level shifter, always on, retention, power switch, and ground switch cells) into the Conformal LP library power intent database.
	1.1	Map the extracted cells to CPF 1.1 features. <i>This is the default.</i>
	2.0	Map the extracted cells to CPF 2.0 features.
-SPice		Specifies that the library filename is in the SPICE format.
-NO_RESistor		Removes resistor instances specified in the SPICE netlist.
-NOSTRength		Specifies that drive strengths are automatically ignored without being removed from library.

Conformal Extended Checks Reference Manual

Command Reference

-File <command_filename>

Reads in the specified command file as a library.

Note: This option is for Verilog or VHDL command file lists. It also supports Liberty command files (-v/-y commands are not supported in Liberty command files).

-SEnsitive

Regards the library model descriptions as case-sensitive. The default case sensitivity is set by the input language. For example, the default case sensitivity for VHDL input is -NOSEnsitive, and the default case sensitivity for Verilog input is -SEnsitive.

-NOSEnsitive

Does not treat the library model descriptions as case-sensitive.

-EXtract

Abstracts the gate information from any transistor library models.

-REPlace

Replaces the existing library. The designs are also deleted.

-APPend

Appends the library to the one that was previously read.

For example, you can use this option to fix a top module and then read it in again without parsing the entire library file again:

```
read library top.v -append -lastmod
```

Note: The top module cannot pass parameters to modules that are read in previously.

-Define <var_name>

Defines `ifdef variable names in Verilog.

The macro redefinition in Verilog source code is ignored if the text macro is specified at the Conformal command line or Verilog command file.

-Map <library_name> <library_path>

Reads in files for the specified <library_name> from <library_path>.

Use this option to read in all of the VHDL or Verilog files in the specified library path for the given library name. You can also map multiple directories to a single library. For example:

```
read library -vhd1 top.vhd -map mylib /design/path1 \
-map mylib /design/path2
```

Conformal Extended Checks Reference Manual

Command Reference

`-MAPRecursive <library_name> <library_path>`

This option has the same function as `-Map`, but it searches for all VHDL or Verilog files recursively down to the subdirectories of the `<library_path>`.

`-Map` searches VHDL or Verilog files under the `<library_path>` and will not search any VHDL files under the subdirectories of `<library_path>`.

`-MAPFile <library_name> [=<library_name2>] <file_name ...>`

Reads in the specified file(s) and includes them in the specified library.

Use this option to specify the files that belong to a given library. The file list terminates with the next option or the end of the `READ LIBRARY` command.

You can also use multiple `-mapfile` options to specify multiple files in a library.

For example, the following two commands are the same:

```
read library -vhd1 top.vhd -mapfile \  
    mylib x1.vhd -mapfile mylib x2.vhd  
read library -vhd1 top.vhd -mapfile mylib x1.vhd x2.vhd
```

You can specify multiple aliases (or names) for a library using the equal sign (`=`). The real library should be specified last. For example, the following command reads in `test.vhd` and includes it in `lib1`; `lib2` and `lib3` are aliases of `lib1`:

```
read library -vhd1 -mapfile lib3=lib2=lib1 test.vhd
```

In this example, `lib1` is the real library; `lib2` and `lib3` are merely aliases.

`-CONFIguration` Supports VHDL configuration constructs. *This is the default.*

`-NOCONFIguration` Does not interpret VHDL configuration.

`-KEEP_ESCAPED_ID` When used, this option keeps escaped identifiers, as in Verilog 2001. See "[Support for Macro Expansions](#)" in the *Conformal Constraint Extended Checks User Guide* for more information.

`-VERBoSe` Displays the verbose messages of parsing and translating each module in the libraries.

Conformal Extended Checks Reference Manual

Command Reference

- SUPPLY** Keeps all Verilog `supply0` and `supply1` type nets unchanged. *This is the default.*
- NOSUPPLY** Converts the Verilog `supply0` and `supply1` type nets to Verilog wire type nets.
- UNCompress <zip_file_name>**
- Reads in the specified compressed file. By default, the Conformal software uses the `gunzip` tool to uncompress the file into the `/tmp` directory. Files created with `gzip` or `compress` are supported.
- If the compressed file cannot be uncompressed using the `gunzip` tool, you specify another tool by setting UNIX variable `CONFORMAL_UNCOMPRESS`.
- You can also set the UNIX variable `CONFORMAL_TMP` to a path other than the default `/tmp` directory.
- UNZip <zip_file_name ...>**
- This option has the same function as `-UNCompress` except that it can include a list of filenames. For example:
- ```
read library fileABC -UNZ zip1 zip2 zip3
```
- The list of filenames end when a subsequent option is specified, or if it is at the end of the command line.
- Note:** Specifying `-uncompress` or `-unzip` is optional for gzipped files because the Conformal software can automatically recognize the file type created by the `gzip` tool. This includes command file lists specified with `-file`.
- OPTimize** Optimizes redundant logic that affects the way Conformal Extended Checks interprets the design.
- Note:** Using this option does not *always* optimize *all* redundant logic.
- See the following example:
- ```
read library file1 file2 file3 -nooptimize -optimize \
-nooptimize -replace
```
- This option is the default.*
- NOOPTimize** Preserves redundant logic in the Library cell.
- See the example listed above (`-optimize`).

Conformal Extended Checks Reference Manual

Command Reference

-STATETable	<p>Enables support for Synopsys Liberty state tables. <i>This is the default.</i></p> <p>Note: This option supersedes the <code>SET STATETABLE</code> command.</p>
-NOSTATETable	<p>Disables support for Synopsys Liberty state tables.</p>
-RESPECT_timing_type_for_flop	<p>If the timing type and the clock phase conflict with the master-slave flip-flop in Liberty file, this honors the timing type as the clock phase of the flip-flop.</p>
-LAsTmod	<p>If duplicate modules exist, Conformal uses the first module and ignores the later ones by default. Use this option to specify that Conformal use the last module and ignore earlier ones.</p>
-LOGIC_ENCODING_OFF	<p>Does not interpret the user-defined <code>enum</code> types with literal value 0, 1, X, Z, and so on as one-bit logic values.</p> <p>For example, <code>MY_MVL</code> is the user-defined <code>enum</code> type:</p> <pre>type MY_MVL is ('X', '0', '1', 'Z');</pre> <p>The type of signal <code>val</code> is <code>MY_MVL</code>. For the statement:</p> <pre>val <= 'Z';</pre> <p>With this option, <code>val</code> is 2'd3; without this option, <code>val</code> is 1'bx.</p>
-MULTIPLE_LIBraries	<p>If duplicate modules exist, Conformal uses the first modules and ignores the later ones by default. Use this option to specify that Conformal store duplicated library modules if they are in a different library.</p>
-CIRcuit_view	<p>Note: This option requires a GXL license.</p> <p>Specifies that the file is read in as a library but it is not used as library for the design. This option only provides the circuit views to examine the transistor level for those checks only, and all the other low power checking ignores the circuit view library.</p>
-NOELaborate	<p>Reads in multiple files of different languages. With this option, you can defer the binding of entity or module instantiations.</p> <p>This is for cases when you have mixed library files in VHDL and Verilog languages, where a VHDL entity in one library file instantiates a Verilog module, and a Verilog module in another library file instantiates a VHDL entity.</p>

Conformal Extended Checks Reference Manual

Command Reference

<code>-BBOXSolver</code>	Does not select blackbox modules from duplicated module declarations.
<code>-EXClude <exclude_file_name*></code>	<p>Specifies files to exclude when reading in the library. This accepts the wildcard.</p> <p>Note: You cannot use multiple wildcards with this option.</p>
<code>-NOKeep_unreach</code>	Remove any unreachable DFF or D-Latch in a module during RTL synthesis. <i>This is the default.</i>
<code>-Keep_unreach</code>	Keeps any unreachable DFF or D-Latch in a module during RTL synthesis.
<code>-SUMmary_report</code>	Print out HDL rule check messages while reading the library or design file(s). <i>This is the default.</i>
<code>-NOSUMmary_report</code>	Does not print out HDL rule check messages while reading the library or design file(s).

Examples

The following commands specify a Verilog simulation library `myLib.v`, and a circuit view library for `myLib.spi` to perform transistor level structure checks:

```
read library -verilog myLib.v
read library -spice -circuit_view myLib.spi
```

In the following example, `read library mod1.v mod2.v` will select blackbox module `sub1()` from `mod1.v`, and `read library -lastmod mod1.v mod2.v` will select blackbox module `sub2()` from `mod2.v`.

To avoid selecting blackboxes, use `read library -bboxesolver mod1.v mod2.v`, which will select `sub1()` from `mod2.v`, and `sub2()` from `mod1.v`.

File `mod1.v`:

```
module sub1(aa, oo);
  input aa;
  output oo;
  // This is a blackbox
endmodule

module sub2(aa, oo);
  input aa;
  output oo;
  assign oo = aa;
endmodule
```

File `mod2.v`:

Conformal Extended Checks Reference Manual

Command Reference

```
module sub1(aa, oo);  
  input aa;  
  output oo;  
  assign oo = !aa;  
endmodule  
  
module sub2(aa, oo);  
  input aa;  
  output oo;  
  // This is a blackbox  
endmodule
```

Related Commands

ADD BLACK BOX

ADD NOTRANSLATE MODULES

ADD SEARCH PATH

READ DESIGN

REPORT DESIGN DATA

REPORT LIBRARY DATA

REPORT MODULES

SET DIRECTIVE

SET STATETABLE

SET UNDEFINED CELL

WRITE LIBRARY

READ RULE CHECK

READ RULE Check

```
<filename> <-EXCLude |-INCLude>  
[-WAIVe | -UNWAive]  
[-Design | -Library]  
(Setup Mode)
```

Hides the rule occurrences listed in the file. The first time you run a session, write the rule violations into a rule file using the `write rule check <filename>` command. For later runs, exclude the violations already flagged with the `read rule check -exclude <filename>` command.

Use the tilde character (~) to shorten the path of the file.

Tcl Command

```
read_rule_check
```

Parameters

<filename>	A required filename. It contains rule violations from a previous session.
-EXCLude	Exclude checks for violations noted in the specified file. This option works as a filter; therefore, use it <i>after</i> the READ DESIGN command.
-INCLude	Include checks for violations noted in the specified file. This option lets you reinstate violations that were previously excluded.
-WAIVe	Waives all rule occurrences in the file.
-UNWAive	Unwaives all rule occurrences in the file. Note: By default, or by specifying <code>-WAIVe</code> or <code>-EXCLude</code> , it waives all rule occurrences in the file. Using <code>-UNWAive</code> or <code>-INCLude</code> , it unwaives them.
-Design	Read only design rule check violations. If you do not specify <code>-design</code> or <code>-library</code> , Conformal reads rule check violations from both designs and libraries.

Conformal Extended Checks Reference Manual

Command Reference

`-Library` Read only library rule check violations. If you do not specify `-design` or `-library`, Conformal reads rule check violations from both designs and libraries.

Examples

In the following example, the second `report rule check` will not report any rules.

```
read design des.v
write rule check rule.des -replace
read design des.v -replace
report rule check -verbose
read rule check rule.des -exclude
report rule check -verbose
```

Related Command

[ADD RULE WAIVER](#)

[DELETE RULE WAIVER](#)

[WRITE RULE CHECK](#)

READ SDC

READ SDC

```
[-SUMmary_report | -NOSUMmary_report]
[-NOReplace | -REPlace ]
[-Sensitive | -NOSensitive]
<sdccfile* ...>
(Setup Mode)
```

Reads in one or more SDC files. This command supports version 1.3 and earlier.

If the design instantiates Liberty library cells that include generated clock definitions, the Conformal software creates these clocks with this command just before reading the SDC file. Thus, the SDC file can refer to these generated clocks. For example, if the cell instance `some/clock/generator` defines a generated clock `gclk`, the SDC file can refer to it as `[get_clocks some/clock/generator/gclk]`.

Note: If you want to have these clocks created but you do not have any SDC file, you must create an empty SDC file and read it using this command.

Note: The SDC rules that apply to generated clocks defined in Liberty libraries are the same as those that apply to generated clocks defined in SDC files.

Conformal Extended Checks supports the following SDC commands:

<code>all_clocks</code>	<code>get_attribute</code>
<code>all_inputs</code>	<code>get_lib_pins</code>
<code>all_outputs</code>	<code>current_design</code>
<code>create_clock</code>	<code>current_instance</code>
<code>create_generated_clock</code>	<code>set_hierarchy_separator</code>
<code>get_cells</code>	<code>set_case_analysis</code>
<code>get_clocks</code>	<code>set_input_delay</code>
<code>get_generated_clocks</code>	<code>set_output_delay</code>
<code>get_lib_cells</code>	
<code>get_libs</code>	
<code>get_nets</code>	
<code>get_pins</code>	
<code>get_ports</code>	

Tcl Command

read_sdc

Parameters

-SUMmary_report	Writes a summary SDC rule report after reading the SDC file(s). <i>This is the default.</i>
-NOSUMmary_report	Does not write a summary SDC rule report after reading the SDC file(s).
-NOReplace	Does not read in the specified files if one has already been read. <i>This option is the default.</i>
-REPlace	Clears the results of the previous SDC reads and read in a new SDC file. Note: SDC reads do not affect the design, but they result in a number of annotations to the design network's internal representation.
-Sensitive	Treats the SDC file contents as case-sensitive. <i>This is the default.</i>
-NOEnsitive	Ignores case-sensitivity for sdc file contents.
<sdcfiler* ...>	Specifies the SDC file(s) to read in.

Related Commands

READ DESIGN

READ LIBRARY

REPORT RULE CHECK -sdc

REMOVE

REMOve

```
<name* ...>  
[-INSTance | -INS_Module]  
[-MODule <module_name*> | -ALL]  
(Setup Mode)
```

Removes instances from the database.

Tcl Command

remove

Parameters

<name* ...>	Specifies the name(s) of the instance(s) to remove. This accepts wildcards.
-INSTance	Indicates that the specified name(s) are instance names. This specifies to remove all instances whose instance names match <name* ...>. <i>This is the default.</i>
-INS_Module	Indicates that the specified name(s) are module names. This specifies to remove all instances whose module names match <name* ...>.
-MODule <module_name*>	Specifies the module names to remove. This accepts wildcards. Note: By default, if you do not specify <mod_name*>, the REMOVE command removes the specified instances from only the root module.
-ALL	Removes the specified instances from all modules in the design.

Example

For these lines:

```
module top (...);
```

Conformal Extended Checks Reference Manual

Command Reference

```
mod1 u01 (...); // inst1
mod2 u02 (...); // inst2
mod3 u03 (...); // inst3
endmodule
module mod3 (...);
  mod1 u01 (...); // inst4
  mod2 u02 (...); // inst5
endmodule
```

- The following command removes `u01` from the root module `top`:

```
remove u01 -ALL // remove inst1, inst4
```

- The following command removes `u01` from module `mod3`:

```
remove u01 -MODULE mod3 // remove inst4
```

- The following command removes all `mod1` instances from the root module `top`:

```
remove mod1 -INS_Module // remove inst1
```

- The following command removes all `mod1` instances from all modules:

```
remove mod1 -INS_Module -ALL // remove inst1, inst4
```


REPORT ALIAS

REPort ALias

[<alias_name*>]
(Setup / Verify Mode)

Displays aliases created with the `ADD ALIAS` command. By default, this displays a list of all aliases.

Tcl Command

`report_alias`

Parameters

<alias_name*> Specifies the name of the alias.

Related Commands

`ADD ALIAS`

`DELETE ALIAS`

REPORT ASSERTION CONSTRAINTS

REPort Assertion Constraints

```
[ -CONSTRAINT [ -Full | -User | -System ]
               | -ALL
               | <pathname* ...>
]
[ -SUMmary | -VERbose ]
(Setup / Verify Mode)
```

Displays all of the assertion constraints information that was added by the `ADD ASSERTION CONSTRAINT` command. The displayed information includes assertion ID, gate ID, path, and a flag indicating if the assertion has been added as a primary constraint.

Tcl Command

`report_assertion_constraints`

Parameters

<code>-CONSTRAINT</code>	Displays the following class of assertion constraints. <i>This option is the default.</i>
<code>-Full</code>	Assertion constraints from both the user and system classes. <i>This option is the default.</i>
<code>-User</code>	Assertion constraints added previously with the <code>ADD ASSERTION CONSTRAINTS</code> command.
<code>-System</code>	Assertion constraints that are contained in the original design.
<code>-ALL</code>	Displays all the assertion constraints.
<code><pathname* ...></code>	Displays the assertion constraints with the specified hierarchical paths. This accepts wildcards.
<code>-SUMmary</code>	Displays a summary of the specified assertion constraints. <i>This option is the default.</i>
<code>-VERbose</code>	Displays a detailed report of the specified assertion constraints.

Related Commands

ADD ASSERTION CONSTRAINTS

ADD STATIC PROPERTY

DELETE ASSERTION CONSTRAINTS

DELETE STATIC PROPERTY

PROVE

REPORT EXTRACTED PROPERTY

REPORT STATIC PROPERTY

REPORT BLACK BOX

REPort Black Box

```
[-Module | -Instance]
[-DEtail]
[-Class <Full | User | System | UNDEFINED | UNSupported | EMPTy | NOTranslate>]
(Setup / Verify Mode)
```

Displays blackboxes from the design. These blackboxes were either created with the `ADD BLACK BOX` command or were a part of the original design.

Tcl Command

```
report_black_box
```

Parameters

-Module	Reports only the blackbox modules. <i>This option is the default.</i>
-Instance	Reports the blackbox instances.
-DEtail	<p>Displays details about blackboxes, where:</p> <p>USER—Indicates that the blackbox was added by the <code>ADD BLACK BOX</code> command.</p> <p>SYSTEM (empty)—Indicates that the module contains no logic.</p> <p>SYSTEM (GRAYBOX)—Indicates that a Liberty cell has an incomplete functional definition. The cell has undriven outputs and a tied output, but no well-functioning output pin.</p> <p>SYSTEM (LEF)—Indicates that the macro is defined in the LEF file, but not defined in library/design modules.</p> <p>SYSTEM (notranslate)—Indicates that the blackbox was added by the <code>ADD NOTRANSLATE MODULES</code> command.</p> <p>SYSTEM (undefined)—Indicates that the blackbox was added by the <code>SET UNDEFINED CELL</code> blackbox command.</p> <p>SYSTEM (unsupported)—Indicates that the module contains unsupported statements.</p>
-Class	Displays one of the following classes of blackboxes:

Conformal Extended Checks Reference Manual

Command Reference

Full	Blackboxes from both the user and system classes. <i>This option is the default.</i>
User	Blackboxes added previously with the <code>ADD BLACK BOX</code> command.
System	Blackboxes that are contained in the original design.
UNDefined	Blackboxes for undefined modules.
UNSupported	Blackboxes for unsupported modules.
EMPTy	Blackboxes for empty modules.
NOTranslate	Blackboxes for notranslate modules.

Related Commands

`ADD BLACK BOX`

`ADD NOTRANSLATE MODULES`

`DELETE BLACK BOX`

`DELETE NOTRANSLATE MODULES`

`REPORT NOTRANSLATE MODULES`

REPORT CDC CHECK

REPort CDc Check

```
[|-STRUctural
  |-FUNcTional [SOURCE_DATA] [DESTINATION_DATA]
    [MUX_enable] [SINGLE_bit_change]
  |-SET
  |-RESET
]
[-SOURce <-ALL | <clock_domain* ...> >]
[-DESTination <-ALL | <clock_domain* ...> >]
[-FROM <-ALL | -REG | -PI | -BBOx | <instance* ...> >]
[-TO <-ALL | -REG | -PO | -BBOx | <instance* ...> >]
[<-MODule | -INStance> <name* ...>
  [-RECurSive | -NORECurSive]]
[-SUMmary | -VERbose]
[-WORD_level | -BIT_level]
[ |-REPORT_PATH
  [-LIBrary_boundary | -NOLIBrary_boundary]
  [-NOFULL_NAME | -FULL_NAME]
]
[ |-LOGic      |-DIRect]
(Verify Mode)
```

Lists Clock Domain Crossing (CDC) Checks specified with the ADD CDC CHECK command.

Tcl Command

report_cdc_check

Parameters

-STRUctural	Reports the Structural CDC Check as specified. Structural Checks include many checks such as cdc_path_logic_type_check, cdc_path_destination_check, sync_chain_dff_number_check.
-------------	---

See [REPORT VALIDATED DATA](#) for more information.

Conformal Extended Checks Reference Manual

Command Reference

-FUNctional

Reports the Functional CDC Check as specified.

Functional CDC Checks include Source Data, Destination Data, MUX Enable, and Single-bit Change checks.

If you do not specify the type of Functional CDC Check, Conformal Extended Checks runs all checks by default.

Note: In order for Conformal Extended Checks to run the Functional CDC Checks on a path, that path must first pass the Structural CDC Check. Otherwise, Conformal Extended Checks will not allow you to add the Functional CDC Check for that path.

SOURCE_DATA Reports the Source Data Functional CDC Check for the specified portion of the design.

The Source Data Functional CDC Check determines whether data leaving the source register (that is, the output of the source register) is held stable for the destination register to latch it.

DESTINATION_DATA

Reports the Destination Data Functional CDC Check for the specified portion of the design.

The Destination Data Functional CDC Check determines whether data entering the destination register is held stable for the destination register to latch it.

MUX_enable Reports the MUX Enable Functional CDC Check for the specified portion of the design.

The MUX Enable Functional CDC Check determines whether data at the output of the multiplexer synchronizer and the select line of the synchronizer are held stable (after changing at the select input) for the destination register to latch it.

SINGLE_bit_change

Conformal Extended Checks Reference Manual

Command Reference

	Reports the Single-bit Change Functional CDC Check for the specified portion of the design.
	The Single-bit Change Functional CDC Check ensures that a vector of signals crossing different clock domains can only be changed one bit at a time.
-SET	Reports the check for asynchronous set pins of the flip-flops to ensure that they are consistent with the clock domain.
-RESET	Reports the check for asynchronous reset pins of the flip-flops to ensure that they are consistent with the clock domain.
-SOURCE	Reports the specified CDC Check for the paths with the specified source clock domain.
	-ALL Uses all clock domains as the source.
	<clock_domain* ...>
	Uses the specified clock domains as the source. This accepts wildcards.
-DESTination	Reports the specified CDC Check for the paths with the specified destination clock domain.
	-ALL Uses all clock domains as the destination.
	<clock_domain* ...>
	Uses the specified clock domains as the destination. This accepts wildcards.
-FROM	Reports the specified CDC Check for the paths that start from the specified key points.
	-ALL Uses all types of key points as the start of the paths.
	The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-REG Uses registers as the start of the paths.
	-PI Uses primary inputs as the start of the paths.
	-BBOX Uses blackbox (outputs) as the start of the paths.
	<instance* ...>

Conformal Extended Checks Reference Manual

Command Reference

	Uses the specified instances as the start of the paths. This accepts wildcards.
	The instances can be registers, primary inputs, or blackbox outputs.
-TO	Report the specified CDC Check for the paths that end at the specified key points.
-ALL	Uses all types of key points as the end of the paths.
	The type of key points includes registers, primary outputs, and blackboxes (inputs).
-REG	Uses registers as the end of the paths.
-PO	Uses primary outputs as the end of the paths.
-BBOx	Uses blackbox (inputs) as the end of the paths.
<instance* ...>	Uses the specified instances as the end of the paths. This accepts wildcards.
	The instances can be registers, primary outputs, or blackbox inputs.
-MODule INSTance <name* ...>	Reports this check for the specified modules or module instance path names.
	-RECURSive Includes submodules.
	-NORecursive Does not include submodules.
-SUMmary	Reports with the "Summary" format. <i>This is the default.</i>
-VERbose	Reports with the "Verbose" format.
-WORD_level	Lists paths at the word-level. <i>This is the default.</i>
-BIT_level	Lists paths at the bit-level.
-REPORT_PATH	Reports CDC Checks with the specified display options.
	-LIBrary_boundary

Conformal Extended Checks Reference Manual

Command Reference

	Reports CDC Checks to the boundary of library cells. This report does not show the contents of the library cells in the path. <i>This is the default.</i>
-NOLIBrary_boundary	
	Reports CDC Checks down to the primitive cell (display the full path).
-NOFULL_NAME	Limits the width of the report to 80 characters. <i>This is the default.</i>
-FULL_NAME	Ignores the character limit to allow full names in the report.
-LOGic	Reports CDC Checks that have logic in the CDC path. "Logic" refers to logic other than wire, buffer, or inverter.
-DIRect	Reports CDC Checks that have only wire, buffers, or inverters in the CDC path.

Related Commands

ADD CDC CHECK

DELETE CDC CHECK

REPORT CDC FILTER

REPort CDc Filter

```
< <filter_name>... | -All>  
(Verify Mode)
```

Reports CDC filters that were originally specified with the `ADD CDC FILTER` command.

Tcl Command

```
report_cdc_filter
```

Parameters

<code><filter_name></code>	Specifies the CDC filter(s) to report.
<code>-ALL</code>	Reports all CDC filters.

Related Commands

[ADD CDC FILTER](#)

[DELETE CDC FILTER](#)

REPORT CLOCK

REPort CLock

```
[ -ALL
| <primary_pin* ...>
| -Candidate [-CLK3 | -CLK2 | -OCCR1 | <id | pathname> ...]
| -INFLuence
|   [-ALL_Clocks
|   | -PIN <primary_pin>
|   | -DFF <id | pathname> [ -ALL_Inputs | -Set | -Reset
|   |   | -Data <0 | port_number>
|   |   | -Clock <0 | port_number>
|   | ]
|   | -DLAT <id | pathname> [-ALL_Inputs | -Set | -Reset
|   |   | -Data <0 | port_number>
|   |   | -Enable <0 | port_number>
|   | ]
|   | -INSTance <id | pathname>
|   | -NOT_INFLuenced [ -ALL_Devices | -DFF | -DLAT | -BBOX | -PO | -PIO]
|   | [-REConvergent | -NOREConvergent]
|   | [ | -LIST_REConvergent]
|   ]
| [-Reduce | -VERbose]
| [-COLUMN <2 | number>]
| [-NOMUX_SELECT_stop | -MUX_SELECT_stop]
| [-NOTRISTATE_stop | -TRISTATE_stop]
]
```

(Setup / Verify Mode)

Displays information about the design's clocks. Additionally, you can use the `-influence` option to list flip-flops, latches, and blackboxes that are driven by a specific clock.

You must define clocks using the `ADD CLOCK` command before you use this command.

Note: The `-candidate`, `-influence`, and `-not_influenced` options are only allowed in Verify mode.

Tcl Command

`report_clock`

Parameters

`-ALL` Displays all clocks. *This is the default.*

Conformal Extended Checks Reference Manual

Command Reference

<primary_pin* ...>	Displays all clocks with the specified primary pins. This accepts wildcards.
-Candidate	Displays the clocks associated with the following: Note: Only use this option in Verify mode.
-CLK3	Displays the clock candidates for D flip-flops and D-latches that violate the Clk3 clocking rule: DFF/DLATCH driven by unknown clock pins. <i>This is the default.</i>
-CLK2	Reports the clock candidates for D flip-flops and D-latches that violate the Clk2 clocking rule: DFF/DLATCH with gated clock.
-OCCR1	Reports the clock candidates for D flip-flops and D-latches that violate the OCCR1 clocking rule: Clock port of DFF/DLATCH is annotated as an unknown domain.
<id pathname>	Displays the violating clock that drives the specified D flip-flop or D-latch. Note: Specify the D flip-flop or D-latch by id or path. (This clock violates the CLK2, CLK3, or OCCR1 clock rule.) Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-INFLuence	Displays all D flip-flops, D-latches, and blackboxes driven by the following clocks. Note: Only use this option in Verify mode.
-ALL_Clocks	Displays all clocks that are driving D flip-flops, D-latches, and blackboxes. <i>This is the default when you use the -influence option.</i>
-PIN <primary_pin>	Displays all D flip-flops, D-latches, and blackboxes driven by the specified clocks.
-DFF	Displays the clocks driving the specified D flip-flop.

Conformal Extended Checks Reference Manual

Command Reference

<id pathname>	<p>The ID number or path of the D flip-flop.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-ALL_Inputs	<p>Displays all clocks driving each of the inputs of the flip-flop. <i>This is the default when you use the -dff option.</i></p>
-Set	<p>Displays all clocks driving D flip-flops for the set input.</p>
-Reset	<p>Displays all clocks driving D flip-flops for the reset input.</p>
-Data	<p>Reports all clocks driving a multi-port D flip-flop affecting a specific data port input.</p> <p>0 specifies port 0. <i>This is the default.</i></p> <p><port_number> specifies a port by number.</p>
-Clock	<p>Reports all clocks driving a multi-port D flip-flop affecting a specific clock port input.</p> <p>0 specifies port 0. <i>This is the default.</i></p> <p><port_number> specifies a port by number.</p>
-DLAT	<p>The clocks affecting a specified D-latch are determined and all clock influences are reported.</p>
<id pathname>	<p>The ID number or path of the D-latch.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-ALL_Inputs	<p>Reports all clocks driving all inputs of the D-latch. <i>This is the default when you use the -dlat option.</i></p>
-Set	<p>Reports all clocks driving the D-latch for the set input.</p>
-Reset	<p>Reports all clocks driving the D-latch for the reset input.</p>
-Data	<p>Reports all clocks driving a multi-port D-latch affecting a specific data port input.</p> <p>0 specifies port 0. <i>This is the default.</i></p> <p><port_number> specifies a port by number.</p>

Conformal Extended Checks Reference Manual

Command Reference

-Enable	<p>Reports all clocks driving a multi-port D-latch affecting a specific enable input.</p> <p>0 specifies port 0. <i>This is the default.</i></p> <p><port_number> specifies a port by number.</p>
-INSTance	<p>The specified names are instance names.</p>
<id pathname>	<p>The ID number or path of the instance.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-NOT_INFLuenced	<p>Displays the devices (as specified) not driven by any clocks.</p> <p>Note: Only use this option in Verify mode.</p>
-ALL_Devices	<p>Displays all D flip-flops, D-latches, and blackboxes that are not driven by any clocks. <i>This is the default when you use -not_influenced.</i></p>
-DFF	<p>Displays all D flip-flops not driven by any clocks.</p>
-DLAT	<p>Displays all D-latches not driven by any clocks.</p>
-BBOX	<p>Displays all blackboxes not driven by any clocks.</p>
-PO	<p>Displays all primary outputs not driven by any clocks.</p>
-PIO	<p>Displays all bidirectional I/Os not driven by any clocks.</p>
-REConvergent	<p>Displays reconvergent paths found during clock influence analysis (see note, below) and include the number of single clocks and multiple clocks in the summary section.</p> <p>Note: If Conformal Extended Checks finds reconvergent paths for a specified clock, the convergent paths are marked with (#) in the clock influence report.</p> <p>If Conformal Extended Checks finds reconvergent paths only for multiple clocks, the reconvergent paths are not marked with (#) in the clock influence report.</p> <p><i>This option is the default when you use the -influence option.</i></p>
-NOREConvergent	<p>Does not analyze for reconvergent paths.</p>

Conformal Extended Checks Reference Manual

Command Reference

-LIST_REConvergent	Lists all D flip-flops, D-latches, primary outputs, primary I/Os, and blackboxes that have reconvergent paths.
-Reduce	Use this option with <code>-influence</code> . <i>This is the default.</i> Show register names as: <div style="margin-left: 40px;"><code>*_reg</code></div>
-VERbose	Use this option with <code>-influence</code> . Show each bit of the register and show the full path of the register.
-COLumn	Specifies the number of spaces to be used for each logic level. 2 specifies two spaces for each logic level. <number> specifies a number of spaces for each logic level. (Specify an integer.)
-NOMUX_SElect_stop	If during reporting, Conformal Extended Checks reaches a multiplexer select input, continue analysis through the select line and on to the rest of the clock tree. <i>This is the default.</i> This option applies to clocks that drive the select input of a multiplexer. It has no effect on reports of clocks that drive other multiplexer inputs.
-MUX_SElect_stop	If during reporting, Conformal Extended Checks reaches a multiplexer select input, discontinue analysis and reporting. This option applies to clocks that drive the select input of a multiplexer. It has no effect on reports of clocks that drive other multiplexer inputs.
-NOTRISTATE_stop	If during reporting, Conformal Extended Checks reaches the data input pin of the tristate, continue clock influence analysis. <i>This is the default.</i>
-TRISTATE_stop	If during reporting, Conformal Extended Checks reaches a tristate, discontinue analysis and reporting.

Related Commands

ADD CLOCK

DELETE CLOCK

Conformal Extended Checks Reference Manual

Command Reference

REPORT RULE CHECK

REPORT CLOCK ASSOCIATION

REPort CLock Association
(*Verify Mode*)

Displays clock associations created with the `ADD CLOCK ASSOCIATION` command.

Tcl Command

`report_clock_association`

Related Commands

`ADD CLOCK ASSOCIATION`

`DELETE CLOCK ASSOCIATION`

REPORT CLOCK DOMAIN

REPort CLock Domain

```
[ [ | -USER | -SYSTEM ]
  | [-CLOCK_PORT
    [ | -Error | -Unknown | -Domain <clock_domain>]
  | -ERROR_SOURCE
  | <id | pathname* ...>
  ]
  [-FANIn <num>] [-FANOut <num>] [-INDent <num>]
  | <-DFF | -DLAT>
]
```

(Verify Mode)

Reports clock domains that have been defined by the user, determined by Conformal Extended Checks, or labeled as error or unknown according to the defined clock domain rule.

This command performs clock domain analysis. If there are any violations, an ACX* rule violation is issued.

Note: The SET CLOCK_DOMAIN RULE command specifies clock domain rules.

Tcl Command

```
report_clock_domain
```

Parameters

-USER	Reports clock domains that result from user specification (see ADD CLOCK , ADD CLOCK ASSOCIATION , and ADD GENERATED CLOCK).
-SYSTEM	Reports clock domains identified by Conformal Extended Checks. See SET CLOCK DOMAIN RULE for more information on how Conformal Extended Checks determines clock domains.
-CLOCK_PORT	Reports the clock port signals that are in the specified clock domain. A clock port signal is the immediate fan-in of a D flip-flop or D-latch from its clock input.
-Error	Reports the clock port signals that are in the Error clock domain.

Conformal Extended Checks Reference Manual

Command Reference

	-Unknown	Reports the clock port signals that are in the unknown clock domain.
	-Domain <code>clock_domain</code>	Reports the clock port signals that are in the specified <code>clock_domain</code> .
-ERROR_SOURCE		Reports gates with assignment conflict errors.
<id pathname* ...>		Reports the clock domain for the specified ID or path. This accepts wildcards. ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-FANIn <num>		Reports clock domains for gates in the fan-in cone to the <code>num</code> depth.
-FANOut <num>		Reports clock domains for gates in the fan-out cone to the <code>num</code> depth.
-INDent <num>		Indents each fan-in or fan-out depth by the number of blank spaces specified by <code>num</code> .
-DFF		Reports clock domain information of all DFFs in the design. The report includes the clock domain of a DFF itself and its inputs pins except the clock pin.
-DLAT		Reports clock domain information of all DLATs in the design. The report includes the clock domain of a DLAT itself and its inputs pins except the enable pin.

Related Commands

ADD CLOCK

ADD CLOCK ASSOCIATION

ADD DATA ASSOCIATION

ADD GENERATED CLOCK

PROPAGATE CLOCK DOMAIN

SET CLOCK DOMAIN RULE

REPORT CLOCK_DOMAIN PRIORITY

REPort CLock_domain Priority
(*Verify Mode*)

Displays clock domain priorities defined by the SET CLOCK_DOMAIN PRIORITY command.

Tcl Command

report_clock_domain_priority

Related Commands

SET CLOCK DOMAIN PRIORITY

SET CLOCK_DOMAIN RULE

REPORT COMMAND PROFILE

REPort COmmand PProfile
[-SUMmary | -Detail]
(Setup / Verify Mode)

Displays a profile of all commands that were executed by users while the Command Profile was set to "on." The profile report includes the order in which commands were executed and the memory use. The profile also reports commands run in the GUI mode.

Tcl Command

report_command_profile

Parameters

- | | |
|----------|---|
| -SUMmary | Lists a summary table of all the commands in alphabetical order. <i>This option is the default.</i> |
| -Detail | Displays a detailed list of all commands in the order they were executed. |

Related Commands

SET COMMAND PROFILE

SET LOG FILE

REPORT DATA ASSOCIATION

REPort DAta Association

[-ALL | -UNAssociated <-PI | -PO | -PIO | -BBOX> ...]
(Verify Mode)

Displays the data associations created with the ADD DATA ASSOCIATION command.

Tcl Command

report_data_association

Parameters

-ALL	Reports all existing data associations.
-UNAssociated	Reports unassociated I/O pins. Choose one of the following supported I/O pins: PI — primary inputs PO — primary outputs PIO — primary inputs and outputs BBOX — blackboxes

Related Commands

ADD DATA ASSOCIATION

DELETE DATA ASSOCIATION

REPORT DESIGN DATA

REPort DESign Data

```
[<module_name>]
[-Summary | -Verbose]
[-INST_COUNT [<cell_name1> <cell_name2> ... ]]
(Setup / Verify Mode)
```

Displays the following:

- Number of design modules
- Library cells
- Inputs
- Outputs
- Primitives
- One-to-one mapped state points

This report includes word-level information about the design in terms of the number of arithmetic/keyword operations. This report includes data path elements such as WMUX, WAND, WXOR and other word-level representations of Boolean logic.

Note: If the display report is too long, use `Ctrl-c` to interrupt the key point listing.

Tcl Command

```
report_design_data
```

Parameters

<code><module_name></code>	Reports design data for the specified module. <i>By default, design data is reported on the top root design module.</i>
----------------------------------	---

Conformal Extended Checks Reference Manual

Command Reference

- Summary** Summarizes the design data including the total number of:
- Design modules
 - Library cells
 - Inputs
 - Outputs
 - Primitives
- This is the default.*
- Verbose** Reports a detailed list of the total number of:
- Design modules
 - Library cells
 - Inputs
 - Outputs
 - Primitives in the design
- INST_COUNT** [`<cell_name1>` `<cell_name2>` ...]
- Reports the instance count (number of copies of a cell being used in the design) for library cells. You can specify a cell or a list of cell names. Without the cell names, this counts all library cells. The cell name list must be enclosed in double-quotes.
- Note:** This option only counts library cells. The `REPORT DESIGN DATA -v` command can also print out the count, but it includes both design and library cells.

Examples

- The following command reports the instance count for all library cells:

```
report design data -inst_count
```
- The following command reports the instance count for library cell `cellA`:

```
report design data -inst_count cellA
```
- The following command reports the instance count for library cells `cellA` and `cellB`:

```
report design data -inst_count "cellA cellB"
```

Related Commands

READ DESIGN

READ LIBRARY

REPORT LIBRARY DATA

REPORT DISPLAY LIST

REPort DIsplay List
(*Verify Mode*)

Lists the added signals that will be displayed during the `DIAGNOSIS` command.

Tcl Command

`report_display_list`

Related Commands

ADD DISPLAY LIST

DELETE DISPLAY LIST

REPORT DYNAMIC CONSTRAINTS

REPort Dynamic Constraints

(Verify Mode)

Displays all the dynamic constraints you added with the `ADD DYNAMIC CONSTRAINTS` command.

Tcl Command

`report_dynamic_constraints`

Examples

For a set of sample commands that shows this and related commands in context, see the example for the [EXPLORE](#) command.

Related Commands

[ADD DYNAMIC CONSTRAINTS](#)

[DELETE DYNAMIC CONSTRAINTS](#)

[EXPLORE](#)

REPORT ENVIRONMENT

REPort ENvironment

```
[|-LP]
[|-NON_DEFAULT]
(Setup Mode)
```

Displays global designs, system and lowpower option settings. The default is to report all global settings.

Tcl Command

```
report_environment
```

Parameters

-LP	Reports environment related to lowpower options.
-NON_DEFAULT	Reports environment that is not the default values.

Related Commands

SET LOG FILE

SET NAMING RULE

SET ROOT MODULE

SET LOWPOWER OPTION

REPORT EXTRACTED PROPERTY

REPort EXtracted Property

```
[ -ALL
  | <-TRI_state [ -ALL | -PO | <id | pathname*...>]
    | -DONT_CARE
      [-ALL          | -X_ASSIGNment | -RANGE_overflow
        | -FULL_case | -PARAllel_case | <id | pathname* ...>
      ]
    | -BRANCH_enable
      [-ALL          | -IF_else | -CAsE
        | -DEFault | -FOR_loop | <id | pathname* ...>
      ]
    | <-BUS          | -MULTI_PORT
      | -SET_RESET
      | -FSM | -FSMReachability
      | -FSMTransition | -FSMDeadlock
      | -NO_DIVide_by_zero
      | -ENCoding_completeness
    >
  [-ALL | <id | pathname*...>]
  > ...
]
[-SUMmary | -VERbose]
[ | -FILE <filename>]
(Verify Mode)
```

In the transition from Setup mode to Verify mode, Conformal Extended Checks analyzes the design and extracts all possible predefined properties for proof. This report displays those extracted properties according to your specification.

Tcl Command

report_extracted_property

Parameters

-ALL	Reports all extracted properties. <i>This is the default.</i>
-TRI_state	Reports extracted tristate properties, as specified.
-ALL	Reports all extracted tristate properties. <i>This is the default.</i>

Conformal Extended Checks Reference Manual

Command Reference

-PO	Reports only extracted tristate pins that are connected to the primary output.
<id pathname* ...>	<p>Reports extracted tristate properties with the specified IDs or hierarchical paths. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-DONT_CARE	Reports extracted Don't Care properties, as specified.
-ALL	Reports all extracted Don't Care properties. <i>This is the default.</i>
-X_ASSIGNment	Reports extracted Don't Care <code>X_assignment</code> properties.
-RANGE_overflow	Reports extracted Don't Care array index overflow properties.
-FULL_case	Reports extracted Don't Care <code>full_case</code> synthesis directive properties.
-PARAllel_case	Reports extracted Don't Care <code>parallel_case</code> synthesis directive properties.
<id pathname* ...>	<p>Reports extracted Don't Care properties with the specified IDs or paths. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>

Conformal Extended Checks Reference Manual

Command Reference

-BRANCH_enable	Reports extracted Branch Enable properties, as specified.
-ALL	Reports all extracted Branch Enable properties. <i>This is the default.</i>
-IF_else	Reports extracted Branch Enable properties for if-else conditional branches.
-CAsE	Reports extracted Branch Enable properties for case conditional branches. (This does not include default branches.)
-DEFault	Reports extracted Branch Enable properties for case-default conditional branches.
-FOR_loop	Reports extracted Branch Enable for_loop properties.
<id pathname* ...>	<p>Reports the specified extracted Branch Enable properties. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-BUS	Reports extracted Bus properties, as specified.
-MULTI_PORT	Reports extracted Multi-port properties, as specified.
-SET_RESET	Reports extracted Set-Reset properties, as specified.
-FSM	Reports extracted finite state machine properties, as specified.
-FSMReachability	Reports extracted finite state machine Reachability properties, as specified.
-FSMTransition	Reports extracted finite state machine Transition properties, as specified.
-FSMDeadlock	Reports extracted finite state machine Deadlock properties, as specified.

Conformal Extended Checks Reference Manual

Command Reference

<code>-NO_DIVide_by_zero</code>	Reports the extracted No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
<code>-ENCoding_completeness</code>	Reports the extracted Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
<code>-ALL</code>	Reports all extracted properties of the specified type. For example, the following command displays all Bus properties: <pre>report extracted property -bus -all</pre> <i>This is the default.</i>
<code><id pathname* ...></code>	Reports extracted properties with the specified IDs or paths. Wildcards are supported for paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
<code>-SUMmary</code>	Reports with the "Summary" format. This format lists the number of instances of each property type. <i>This is the default.</i>
<code>-VERbose</code>	Reports with the "Verbose" format. This format includes property type and each instance ID number and name.
<code>-FILE <filename></code>	Prints the report to the specified file.

Related Commands

ADD IGNORED PROPERTY

ADD STATIC PROPERTY

DELETE IGNORED PROPERTY

DELETE STATIC PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

Conformal Extended Checks Reference Manual

Command Reference

REPORT FSM

REPORT GATE

REPORT IGNORED PROPERTY

REPORT PROVED DATA

REPORT STATIC PROPERTY

REPORT FIFO FAILURE

REPort Fifo Failure

```
<fifo_memory_registers*>  
[<read_data_registers*>]  
(Verify Mode)
```

Reports the causes for non-identified FIFOs in the design. If the software can detect the FIFO under the present setup, this command will inform you to run the `GENERATE FIFO INFO` command to identify the FIFO. If the software cannot detect the FIFO under the present setup, the command will inform you the reason for the same.



Before running this command, you must first use the `ADD CDC CHECK` command's `-structural` option to identify the part of the design to search for FIFOs because FIFO detection works on existing clock domain crossings.

Tcl Command

```
report_fifo_failure
```

Parameters

```
<fifo_memory_registers*>
```

Specifies the the name of the FIFO memory register(s) to report. This accepts wildcards.

```
<read_data_registers*>
```

Specifies the the name of the data output of FIFO (read data registers or primary outputs) register(s) to report. This accepts wildcards.

Related Commands

[GENERATE FIFO INFO](#)

[REPORT FIFO INFO](#)

REPORT FIFO INFO

REPort FIfO InFo

```
[ | < < [-Index <index1 index2>]
| [-INStance <instance_name>]
| [-Module <module_name>] > >]
[ [-MEMory]
    [-SYNC | [RSYNC | WSYNC] ]
    [-GRAY | [RGRAY | WGRAY] ]
    [-ADR | [ RADR| WADR] ]
    [-NAME]
| -Verbose]
(Verify Mode)
```

Reports the FIFOs and their components in the design.

Tcl Command

```
report_fifo_info
```

Parameters

-Index <index>	Specifies the index information to report on an existing FIFO.
-INStance <instance_name>	Specifies an instance name to report information on the FIFO. The instance name is the instance name of the module containing FIFO memory.
-Module <module_name>	Specifies that the information of all the instances of the module will be reported on the FIFO.
-MEMory	Reports the information for the memory registers.
-SYNC	Reports the information for all synchronizer registers.
-RSYNc	Reports the information for the read synchronizer registers.
-WSYNc	Reports the information for the write synchronizer registers.
-GRAY	Reports the information for all gray code registers.
-RGRAY	Reports the information for the read gray code registers.
-WGRAY	Reports the information for the write gray code registers.

Conformal Extended Checks Reference Manual

Command Reference

-ADR	Reports the information for all address registers.
-RADr	Reports the information for the read address registers.
-WADr	Reports the information for the write address registers.
-NAME	Reports the name of the FIFO.
-Verbose	Reports all the information about the selected FIFO.

Related Commands

ADD FIFO INFO

DELETE FIFO INFO

GENERATE FIFO INFO

REPORT FLOATING SIGNALS

REPort Floating Signals

```
[-R0ot | -Module <module_name> | -All]  
[-UNDriven | -UNUsed]  
[ | -Net | -Pin]  
(Setup / Verify Mode)
```

Displays all floating signals in the design. The floating signals are either nets or pins and are either undriven or unused. Use the `SET UNDRIVEN SIGNAL` command to specify the global behavior of the undriven floating signals in the design.

Tcl Command

```
report_floating_signals
```

Parameters

-R0ot	Displays all the floating signals in the root module. <i>This option is the default.</i>
-Module <module_name>	Displays all the floating signals in the specified module within the given defaults.
-All	Displays all the floating signals in all design modules within the given defaults.
-UNDriven	Displays only undriven floating signals. <i>This option is the default.</i>
-UNUsed	Displays only unused floating signals.
-Net	Displays only floating nets. If you do not specify <code>-net</code> or <code>-pin</code> , Conformal Extended Checks displays both floating nets and floating pins.
-Pin	Displays only floating pins. If you do not specify <code>-net</code> or <code>-pin</code> , Conformal Extended Checks displays both floating nets and floating pins.

Related Command

SET UNDRIVEN SIGNAL

REPORT FSM

REPort FSM

```
[-ALL | <id | pathname* ... > | <-MODule <module_name*> > ...]  
[-NO_FULL_name | -FULL_name]  
[-SUMmary | -VERbose]  
(Setup / Verify Mode)
```

Displays automatically extracted finite state machines.

Tcl Command

report_fsm

Parameters

-ALL Displays all the extracted finite state machines. *This option is the default.*

<id | pathname* ...>

Displays finite state machines with the specified IDs or path names. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

-MODule <module_name*>

Displays the extracted finite state machines in the specified modules.

-NO_FULL_name Displays the width of the report to 80 characters. *This option is the default.*

-FULL_name Overrides the character limit to allow full names in the report.

-SUMmary Reports with the "Summary" format. The default format returns an FSM list with columns for the module name, FSM name, and size. It concludes with the total number of FSMs reported.

-VERbose Reports with the "Verbose" format. The default format lists each FSM by name with expanded information.

Related Commands

ADD IGNORED PROPERTY

ADD STATIC PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

REPORT EXTRACTED PROPERTY

REPORT GATE

REPORT PROVED DATA

REPORT GATE

REPort Gate

```
-SUMmary
| [
|   [-Type < PI          | PO          | PIO          | BBOX:IN
|       | BBOX:OUT      | BBOX:IO      | BBOX         | BUS
|       | DFF           | DLAT         | TRI_state    | DC
|       | BRANCH
|       | FSM           | FSMState     | FSMTrans
|       | DIVZERO
|       | ENCCOMP
|       | ALWAYS        | CHANGE       | DECREMENT    | DELTA
|       | EVENPARITY     | HANDSHAKE    | INCREMENT    | NEVER
|       | NOOVERFLOW    | NOTTRANSITION | NOUNDERFLOW  | ODDPARITY
|       | OA1HOT        | PROPOSITION  | RANGE        | TIME
|       | TRANSITION   | UNCHANGE     | WINCHANGE    | WINUNCHANGE
|       | WINDOW        | OA01HOT      | PROP1HOT     | PROP01HOT
|       | PROP1COLD     | PROP11COLD   | FRAME        | OA1COLD
|       | NEXT          | ONEDGE       | WIDTH        | FIFOIDX
|       | OAIMPLY      | QSTATE       | CYCLESEQ
|   ]
|   [ <pathname* | id> ]
| ...
|
| [
|   | -SUPport
|   | <-SIMvalue [ | <time_unit>] | -INITvalue>
|   | -FANIn <integer> [-Full]
|   | -FANOut <integer> [-Full]
|   | -COLLapse [ALL | BUfFer | INVerter | NONCONTRol]
|   | -INDent <integer>
| ] ...
(Verify Mode)
```

Displays information about the circuit structure in a variety of ways. For example:

- Summary of gate counts
- Fan-in / fan-out cones
- Tied / simulation value
- Support variables

Tcl Command

report_gate

Parameters

-SUMmary	Reports gate counts for each gate type. <i>This option is the default if you do not specify an option.</i>
-Type	Reports gates of the specified type:
PI	Reports primary inputs.
PO	Reports primary outputs.
PIO	Reports bidirectional I/Os.
BBOX:IN	Reports blackbox inputs.
BBOX:OUT	Reports blackbox outputs.
BBOX:IO	Reports blackbox I/O ports.
BBOX	Reports blackboxes.
BUS	Reports buses.
DFF	Reports D flip-flops.
DLAT	Reports D-latches.
TRI_state	Reports tristate buffers.
DC	Reports Don't Care properties.
BRANCH	Reports Branch-Enable properties.
FSM	Reports all instantiated finite state machine properties.
FSMState	Reports finite state machine state cells.
FSMTrans	Reports finite state machine deadlock cells.
DIVZERO	Reports No-Divide-By-Zero properties.
ENCCOMP	Reports Encoding-Completeness properties.
ALWAYS	Reports <code>assert_always</code> instances.
CHANGE	Reports <code>assert_change</code> instances.
DECREMENT	Reports <code>assert_decrement</code> instances.
DELTA	Reports <code>assert_delta</code> instances.
EVENPARITY	Reports <code>assert_even_parity</code> instances.

Conformal Extended Checks Reference Manual

Command Reference

HANDSHAKE	Reports <code>assert_handshake</code> instances.
INCREMENT	Reports <code>assert_increment</code> instances.
NEVER	Reports <code>assert_never</code> instances.
NOOVERFLOW	Reports <code>assert_no_overflow</code> instances.
NOTRANSITION	Reports <code>assert_no_transition</code> instances.
NOUNDERFLOW	Reports <code>assert_no_underflow</code> instances.
ODDPARITY	Reports <code>assert_odd_parity</code> instances.
OA1HOT	Reports <code>assert_one_hot</code> instances.
PROPOSITION	Reports <code>assert_proposition</code> instances.
RANGE	Reports <code>assert_range</code> instances.
TIME	Reports <code>assert_time</code> instances.
TRANSITION	Reports <code>assert_transition</code> instances.
UNCHANGE	Reports <code>assert_unchange</code> instances.
WINCHANGE	Reports <code>assert_win_change</code> instances.
WINUNCHANGE	Reports <code>assert_win_unchange</code> instances.
WINDOW	Reports <code>assert_window</code> instances.
OA01HOT	Reports <code>assert_zero_one_hot</code> instances.
PROP1HOT	Reports <code>assert_prop_one_hot</code> instances.
PROP01HOT	Reports <code>assert_prop_zero_one_hot</code> instances.
PROP1COLD	Reports <code>assert_prop_one_cold</code> instances.
PROP11COLD	Reports <code>assert_prop_one_one_cold</code> instances.
FRAME	Reports <code>assert_frame</code> instances.
OA1COLD	Reports <code>assert_one_cold</code> instances.

Conformal Extended Checks Reference Manual

Command Reference

NEXT	Reports <code>assert_next</code> instances.
ONEDGE	Reports <code>assert_always_on_edge</code> instances.
WIDTH	Reports <code>assert_width</code> instances.
FIFOIDX	Reports <code>assert_fifo_index</code> properties.
OAIMPLY	Reports <code>assert_implication</code> instances.
QSTATE	Reports <code>assert_quiescent_state</code> instances.
CYCLESEQ	Reports <code>assert_cycle_sequence</code> instances.
<code><pathname* id></code>	<p>Displays the specified path or ID. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
<code>-SUPport</code>	Displays only the support points.
<code>-SIMvalue <time_unit></code>	<p>Displays the counter-example logic values at the specified time cycle. If you omit <code>time_unit</code>, Conformal Extended Checks displays values for the last cycle of the counterexample.</p> <p>Note: You must use the <code>DIAGNOSE STATIC PROPERTY</code> command before logic values can be displayed.</p>
<code>-INITvalue</code>	<p>Displays the logic values immediately after initialization.</p> <p>Note: These values are before, and can differ from, counterexample time 0.</p>
<code>-FANIn <integer></code>	Displays the fan-in cone to an "integer" depth for the specified entities or properties.
<code>-Full</code>	Includes all duplicated subcones in the display.
<code>-FANOut <integer></code>	Displays the fan-out cone to an "integer" depth for the specified entities or checks.

Conformal Extended Checks Reference Manual

Command Reference

	-Full	Includes all duplicated subcones in the display.
-COLLapse	Collapses fan-in and fan-out cones.	
	ALL	Collapses all buffers, inverters, and non-controlling logic. <i>This option is the default.</i>
	BUFFer	Collapses buffers.
	INVerter	Collapses inverters.
	NONCONTrol	Collapses non-controlling logic.
-INDent <integer>	Indents each fan-in or fan-out depth by the number of blank spaces specified by the integer.	
	Note: By default, each fan-in and fan-out depth is indented by <i>two</i> spaces.	

Related Command

DIAGNOSE STATIC PROPERTY

REPORT GENERATED CLOCK

REPort GEnerated Clock
(*Verify Mode*)

Lists the clocks created by the `ADD GENERATED CLOCK` command.

Tcl Command

`report_generated_clock`

Related Commands

`ADD GENERATED CLOCK`

`DELETE GENERATED CLOCK`

REPORT IGNORE RESET_CONSTRAINT

REPort IGNore RESET_constraint

```
<-ALL
| <-BRANCH_enable
| -FSMTransition
| -FSMReachability
| -TRI_state
>
>
(Setup / Verify Mode)
```

Reports whether the specified checks have reset constraints ignored or applied. See the [ADD IGNORE RESET_CONSTRAINT](#) command for more information.

Tcl Command

report_ignore_reset_constraint

Parameters

-ALL	Displays whether Branch Enable, FSM Transition, FSM Reachability, and Tristate Stuck-on/Stuck-off checks have reset constraints ignored or applied. <i>This option is the default.</i>
-BRANCH_enable	Displays whether Branch Enable checks have reset constraints ignored or applied.
-FSMTransition	Displays whether FSM Transition checks have reset constraints ignored or applied.
-FSMReachability	Displays whether FSM Reachability checks have reset constraints ignored or applied.
-TRI_state	Displays whether Tristate Stuck-on/Stuck-off checks have reset constraints ignored or applied.

Related Commands

[ADD ASSERTION CONSTRAINTS](#)

[ADD IGNORE RESET_CONSTRAINT](#)

Conformal Extended Checks Reference Manual

Command Reference

ADD PIN CONSTRAINTS

DELETE IGNORE RESET CONSTRAINT

REPORT IGNORED PROPERTY

REPort IGnored Property

```
[ -ALL
| < -TRI_state [ -ALL | -PO | <id | pathname*> ...]
|-DONT_CARE
  [-ALL | -X_ASSIGNment | -RANGE_overflow
  | -FULL_case | -PARAllel_case | <id | pathname*> ...
  ]
|-BRANCH_enable
  [ -ALL | -IF_else | -CAsE
  | -DEFault | -FOR_loop | <id | pathname*> ...
  ]
| <-BUS | -MULTI_PORT
  | -SET_RESET
  | -FSM |-FSMReachability
  | -FSMTransition | -FSMDeadlock
  | -NO_DIVide_by_zero
  | -ENCoding_completeness
  >
  [ -ALL | <id | pathname*> ... ]
> ...
]
[-SUMmary | -VERbose]
(Verify Mode)
```

Reports information about properties that were specifically excluded from the "Prove" list with the `ADD IGNORED PROPERTY COMMAND`. The default report is a summary format that lists the number of instances of each property.

Conformal Extended Checks generates this report for individual properties, properties by type, or all properties.

To exclude properties from the "Prove" list, use the `ADD IGNORED PROPERTY` command. To reinstate properties and make them available for the "Prove" list, use the `DELETE IGNORED PROPERTY` command.

Note: View all properties with the Static Property Manager in the Conformal Extended Checks graphical user interface.

Tcl Command

`report_ignored_property`

Conformal Extended Checks Reference Manual

Command Reference

Parameters

-ALL	Reports all "ignored" properties. <i>This is the default.</i>										
-TRI_state	Reports the specified "ignored" tristate properties: <table><tr><td>-ALL</td><td>Reports all "ignored" tristate properties. <i>This is the default.</i></td></tr><tr><td>-PO</td><td>Reports only "ignored" tristate pins that are connected to the primary output.</td></tr></table>	-ALL	Reports all "ignored" tristate properties. <i>This is the default.</i>	-PO	Reports only "ignored" tristate pins that are connected to the primary output.						
-ALL	Reports all "ignored" tristate properties. <i>This is the default.</i>										
-PO	Reports only "ignored" tristate pins that are connected to the primary output.										
<id pathname*> ...	Reports "ignored" tristate properties with the specified IDs or hierarchical paths. Wildcards are supported for paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.										
-DONT_CARE	Reports the specified "ignored" Don't Care properties: <table><tr><td>-ALL</td><td>Reports all "ignored" Don't Care properties. <i>This is the default.</i></td></tr><tr><td>-X_ASSIGNment</td><td>Reports "ignored" Don't Care X_assignment properties.</td></tr><tr><td>-RANGE_overflow</td><td>Reports "ignored" Don't Care array index overflow properties.</td></tr><tr><td>-FULL_case</td><td>Reports "ignored" Don't Care full_case synthesis directive properties.</td></tr><tr><td>-PARAllel_case</td><td>Reports "ignored" Don't Care parallel_case synthesis directive properties.</td></tr></table>	-ALL	Reports all "ignored" Don't Care properties. <i>This is the default.</i>	-X_ASSIGNment	Reports "ignored" Don't Care X_assignment properties.	-RANGE_overflow	Reports "ignored" Don't Care array index overflow properties.	-FULL_case	Reports "ignored" Don't Care full_case synthesis directive properties.	-PARAllel_case	Reports "ignored" Don't Care parallel_case synthesis directive properties.
-ALL	Reports all "ignored" Don't Care properties. <i>This is the default.</i>										
-X_ASSIGNment	Reports "ignored" Don't Care X_assignment properties.										
-RANGE_overflow	Reports "ignored" Don't Care array index overflow properties.										
-FULL_case	Reports "ignored" Don't Care full_case synthesis directive properties.										
-PARAllel_case	Reports "ignored" Don't Care parallel_case synthesis directive properties.										
<id pathname*> ...											

Conformal Extended Checks Reference Manual

Command Reference

	Reports "ignored" Don't Care properties with the specified IDs or paths. Wildcards are supported for paths.
	Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-BRANCH_enable	Reports "ignored" Branch Enable properties, as specified.
-ALL	Reports all "ignored" Branch Enable properties. <i>This is the default.</i>
-IF_else	Reports "ignored" Branch Enable properties for if-else conditional branches.
-CAsE	Reports "ignored" Branch Enable properties for case conditional branches. (This does not include default branches.)
-DEFault	Reports "ignored" Branch Enable properties for case-default conditional branches.
-FOR_loop	Reports "ignored" Branch Enable <code>for_loop</code> properties.
<id pathname*> ...	Reports the specified "ignored" Branch Enable properties. Wildcards are supported for paths.
	Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-BUS	Reports the specified "ignored" Bus properties.
-MULTI_PORT	Reports the specified "ignored" Multi-port properties.
-SET_RESET	Reports the specified "ignored" Set-Reset properties.
-FSM	Reports the specified "ignored" finite state machine properties.
-FSMReachability	Reports the specified "ignored" finite state machine Reachability properties.

Conformal Extended Checks Reference Manual

Command Reference

-FSMTransition	Reports the specified "ignored" finite state machine Transition properties.
-FSMDeadlock	Reports the specified "ignored" finite state machine Deadlock properties.
-NO_DIVide_by_zero	Reports the extracted No-Divide-By-Zero properties, as specified. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
-ENCoding_completeness	Reports the extracted Encoding-Completeness properties, as specified. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
-ALL	<p>Reports all of the specified "ignored" properties. For example, the following command reports all "ignored" Bus properties:</p> <pre>report ignored property -bus -all</pre> <p><i>This is the default.</i></p>
<id pathname*> ...	<p>Reports "ignored" properties with the specified IDs or hierarchical paths. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
-SUMmary	Reports with the "Summary" format. This format lists the number of instances of each property type. <i>This is the default.</i>
-VERbose	Reports with the "Verbose" format. This format includes property type and each instance ID number and name.

Related Commands

[ADD IGNORED PROPERTY](#)

[ADD STATIC PROPERTY](#)

[DELETE IGNORED PROPERTY](#)

[DELETE STATIC PROPERTY](#)

Conformal Extended Checks Reference Manual

Command Reference

DIAGNOSE STATIC PROPERTY

PROVE

REPORT EXTRACTED PROPERTY

REPORT FSM

REPORT GATE

REPORT PROVED DATA

REPORT STATIC PROPERTY

REPORT INITIAL STATE

REPort INitial State
(Setup / Verify Mode)

Displays all the initial state values that were added with the `ADD INITIAL STATE` command.

Tcl Command

`report_initial_state`

Related Commands

`ADD INITIAL STATE`

`DELETE INITIAL STATE`

REPORT INSTANCE CONSTRAINTS

REPort INstance Constraints
(*Setup / Verify Mode*)

Displays the constraints placed on instances. These instances were specified with the `ADD INSTANCE CONSTRAINTS` command.

Tcl Command

```
report_instance_constraints
```

Related Commands

ADD INSTANCE CONSTRAINTS

DELETE INSTANCE CONSTRAINTS

REPORT LIBRARY DATA

REPort Library Data

```
[-SOURCE]
[-SORT <NAME | REference | INstance>]
(Setup / Verify Mode)
```

Displays the following columns:

- *ID*—Specifies the cell ID.
- *Name*—Specifies the cell name.
- *Cost*—Specifies the cost of each library cell, which is the product of the number of instances of primitive gates within each library cell (*Ins*) and the number of times the library cell is instantiated (*Ref*).
- *Ins*—Displays the number of instances of primitive gates within each library cell that is instantiated in the design.
- *Ref*—Displays the number of times the library cell is instantiated in the design.
- *TOT*—Displays the number of primitive gates within each library cell.
- *DFF*—Specifies whether the cell contains a D flip-flop.
- *DLAT*—Specifies whether the cell contains a D-latch.
- *BUF*—Specifies whether the cell contains a buffer.
- *NOT*—Specifies whether the cell contains an inverter gate.
- *BBOX*—Specifies whether the cell contains a blackbox.
- *UDP*—Specifies whether the cell is a UDP.

Tcl Command

```
report_library_data
```

Parameters

- | | |
|---------|---|
| -SOURCE | Displays the source filename and line number for each cell. |
| -SORT | Sorts report data as specified: |

Conformal Extended Checks Reference Manual

Command Reference

NAME	Sorts report data alphabetically by library cell name.
REference	Sorts report data according to the Reference column in descending order.
INStance	Sorts report data according to the Instance column in descending order.

Related Commands

READ DESIGN

READ LIBRARY

REPORT DESIGN DATA

WRITE LIBRARY

REPORT MODULES

REPort MOdules

```
[-COMPILER_directives <directive name...> | -ALL_DIRectives]
[-ROot | <module_name> [-Down | -Up] | -All | -Top]
[-Source]
[-INSTantiation [-PARAMinfo | -CONST]]
[-USer]
[-VHDLname]
[-LEVEL <value>]
[-Llibrary]
(Setup / Verify Mode)
```

Displays module information for the design. If a module name is given, additional information on modules and library cells can be displayed up or down the hierarchy of the given module name.

Tcl Command

report_modules

Parameters

-COMPILER_directives <directive name...>	Reports the modules which fall under the mentioned compiler directive(s) from this list: celldefine, default_nettype, define, else, elsif, endcelldefine, endif, ifdef, ifndef, include, line, resetall, undef.
-ALL_DIRectives	Reports all the modules/cells/parameters/values that are used by the any of the listed compiler directives.
-ROot	Displays module information for the root module. <i>This option is the default.</i>
<module_name>	Reports module information on a specified module. An additional option lets you specify how the display is ordered:
-Down	Reports modules and library cells in descending order. <i>This is the default.</i>
-Up	Reports modules and library cells in ascending order.

Conformal Extended Checks Reference Manual

Command Reference

-All	Displays information for all modules in the design hierarchy.
-Top	Displays the top modules.
-Source	Displays the source filename and line number for each module.
-INSTantiation	Displays instantiation information for modules.
-PARAMinfo	Display parameter value information for the instance of a Verilog parameterized module or a VHDL generic entity.
-CONST	Display parameter and enum value information for top module and instances.
-USer	Reports only the modules defined in the design files, and skips the internal modules which are not defined in the design files. Note: Some internal modules can be created by the Conformal tools after reading the design files. These internal modules are not defined in the design files.
-VHDLname	Displays the full name, rather than just the entity name. For example: <code>libname.entityname(architecturename)</code>
-LEVEL <value>	Displays modules in the design hierarchy to a given level. <value> must be an integer.
-Library	Displays library cells in the design hierarchy.

Examples

This example shows the difference between running the `REPORT MODULES` command without any options versus running the command with the `-USer` option.

The following design file contains only one module named `test`. Module `VDW_mult_nbw_u8_u8_16` is internal module which is not defined in this file:

```
module test(aa, bb, oo);
input [7:0] aa, bb;
output oo;
    assign oo = aa * bb;
endmodule
```

Running the following command:

```
report modules
```

Conformal Extended Checks Reference Manual

Command Reference

the Conformal software reports the `test` and `VDW_mult_nbw_u8_u8_16` modules. However, when running the following command:

```
report modules -user
```

the Conformal software reports only the `test` module.

Related Commands

[READ DESIGN](#)

[READ LIBRARY](#)

REPORT NOTRANSLATE FILEPATHNAMES

REPort NOtranslate Filepathnames

[| -Library | -Design]
(Setup / LEC Mode)

Displays all of the library and design file pathnames originally added with the `ADD NOTRANSLATE FILEPATHNAMES` command. The Conformal software will not compile these modules defined in libraries and design files.

Tcl Command

`report_notranslate_filepathnames`

Parameters

<code>-Library</code>	Displays only the added library file pathnames.
<code>-Design</code>	Displays only the added design file pathnames.

Related Commands

`ADD NOTRANSLATE FILEPATHNAMES`

`ADD NOTRANSLATE MODULES`

`DELETE NOTRANSLATE FILEPATHNAMES`

`DELETE NOTRANSLATE MODULES`

`REPORT NOTRANSLATE MODULES`

REPORT NOTRANSLATE MODULES

REPort NOtranslate Modules
(Setup / Verify Mode)

Lists all the library and design modules added with the `ADD NOTRANSLATE MODULES` command. These modules are not compiled when you execute the `READ LIBRARY` or `READ DESIGN` command.

Tcl Command

`report_notranslate_modules`

Related Commands

`ADD BLACK BOX`

`ADD NOTRANSLATE MODULES`

`DELETE BLACK BOX`

`DELETE NOTRANSLATE MODULES`

`READ DESIGN`

`READ LIBRARY`

`REPORT BLACK BOX`

REPORT PATH

REPort PAth

```
< <src gate> <dest gate>
[-COMBinatorial | -SEQuential [INFinity | <depth#>]
| -LOOP [-ALL | <id | pathname>] [-BOTH | -STRONG | -WEAK]
>
(Verify Mode)
```

Lists the combinational paths between the source and destination gates, and lists the combinational loops. If the source gate is the same as the destination gate, Conformal Extended Checks reports the combinational loop (if any exists) starting from the specified gate.

Tcl Command

report_path

Parameters

src gate	Reports the path between the specified source ID or path and the specified destination ID or path.
dest gate	
	Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-COMBinatorial	Reports a combinational path (including loops found in rules STRC1.1 and STRC1.2). <i>This option is the default.</i>
-SEQuential [-INFinity <depth#>]	Reports a sequential path. -INFinity (<i>the default for this option</i>), specifies that there is no depth restrictions for sequential path reporting. #depth specifies the depth (integer) for sequential path reporting.
-LOOP	Reports combinational loops found in rules STRC1.1 and STRC1.2, as specified:
-ALL	Reports all combinational loops found in rules STRC1.1 and STRC1.2. <i>This option is the default.</i>
<id pathname>	

Conformal Extended Checks Reference Manual

Command Reference

	Reports combinational loops for the specified ID or path.
-BOTH	Reports strong or weak combination loops. <i>This option is the default.</i>
-STRONG	Reports only combinational loops with strong devices (i.e., combinational loops without any weak devices).
-WEAK	Reports only combinational loop with weak devices (i.e., combinational loops with at least one weak device).

Related Commands

REPORT RULE CHECK

SET RULE HANDLING

REPORT PATH STATISTICS

```
REPort Path Statistics
  [-DOMAIN <-ALL | <clock_domain* ...> >]
  [-FROM <-ALL | instances* ...> >]
  [-TO <-ALL | instances* ...> >]
  [-WEIGHT <gate_weight_definition_filename>]
  [-RANGE <min> <max>]
  [-GROUP <num>]
  [-SUMmary [ -HIStogram | -TABle] | -VERbose [-NOLimit | -Limit <num>]]
  (Setup / Verify Mode)
```

Displays the path statistics for the paths selected in the design. You can use the `-domain` option to limit the reporting to one clock domain.

Tcl Command

```
report_path_statistics
```

Parameters

<code>-DOMAIN</code>	Specifies the clock domains for which you want to collect path statistics.
<code>-ALL</code>	Displays path statistics for all clock domains.
<code><clock_domain* ...></code>	Displays path statistics for the specified clock domain. This accepts wildcards.
<code>-FROM</code>	Collects path statistics only for paths that start from the specified instances.
<code>-ALL</code>	Collects path statistics for all paths.
<code><instances* ...></code>	Collects path statistics for paths starting at the specified instance.
<code>-TO</code>	Collects path statistics only for paths ending at the specified instance.
<code>-ALL</code>	Collects path statistics for all paths.
<code><instances* ...></code>	

Conformal Extended Checks Reference Manual

Command Reference

Collects path statistics for paths ending at the specified instance.

`-WEIght <gate_weight_definition_filename>`

Specifies the weight for each type of gate using the specified file. If you do not use this option, each gate type is counted as 1.

The specified file should use the following format, where each line defines the weight of a gate type:

`<gate_type> <integer>`

Conformal CD supports the following gate types: BUF, INV, AND, NAND, OR, NOR, XOR, XNOR, BUFIF1, BUFIF0, NOTIF1, NOTIF0, and MUX.

`-RANge <min> <max>`

Collects path statistics only for paths that have lengths between `<min>` and `<max>`.

`-GROUP <num>`

Specifies the report format. This option divides the paths based on the number of groups, which is specified by `<num>`, and the longest path. For example, if `<num>` is 4 and your longest path is 30, then your paths are divided into the following four groups:

0 <= length < 8
8 <= length < 16
16 <= length < 24
24 <= length

By default, the maximum number of groups is 10.

Note: If a group does not contain a path, it is compressed automatically and not reported.

`-SUMmary [-HIStogram | -TABle]`

Specifies how the report is formatted.

`-HIStogram` Reports path statistics in histogram format.

`-TABle` Reports path statistics in a table format.

`-VERbose [-NOLimit | -Limit <num>]`

Provides a verbose report, including the starting point, the ending point, and the path length—from the longest to the shortest path.

`-NOLimit` Does not limit to the number of entries that it displays.

Conformal Extended Checks Reference Manual

Command Reference

<code>-Limit <num></code>	Limits the number of entries displayed using <code><num></code> .
---------------------------------	---

REPORT PIN CONSTRAINTS

REPort PIn Constraints

```
[-CLASS < ALL | USER | LOWPOWER >]  
[-ROot | -Module <module_name> | -All]  
(Setup / Verify Mode)
```

Lists the constraints placed on a design's primary input pins. The constraints were originally specified with the `ADD PIN CONSTRAINTS` command.

Tcl Command

```
report_pin_constraints
```

Parameters

-CLASS	ALL: Report all pin constraints. <i>This is the default.</i> USER: Report only constraints added using the <code>ADD PIN CONSTRAINT</code> command. LOWPOWER: Report only power and ground constraints inferred by power intent.
-ROot	Displays the pin constraints from the root module. <i>This is the default.</i>
-Module <module_name>	Displays the pin constraints from the specified module.
-All	Displays pin constraints in all" modules within the given defaults.

Related Commands

[ADD PIN CONSTRAINTS](#)

[DELETE PIN CONSTRAINTS](#)

REPORT PIN DIRECTION

REPort PIn Direction

```
[-IO | -IN | -OUT]
[<module_name>]
[-Summary | -Verbose]
(Setup / Verify Mode)
```

Displays the assigned pin directions for each module. *The default is to display only a summary message.*

Note: Use the `ASSIGN PIN DIRECTION` command to assign pin direction to module I/O pins.

Refer to the *Conformal Equivalence Checking Reference Manual* and *Conformal Equivalence Checking User Guide* for additional information about this command.

Tcl Command

```
report_pin_direction
```

Parameters

-IO	Reports assigned module I/O pins. <i>This is the default.</i>
-IN	Reports assigned module input pins.
-OUT	Reports assigned module output pins.
<module_name>	Reports pin direction for the specified module. <i>By default, pin direction is reported for all modules.</i>
-Summary	Displays only a summary message of assigned pin directions. <i>This option is the default.</i>
-Verbose	Displays all assigned pin directions.

Related Commands

ASSIGN PIN DIRECTION

REPORT PIN EQUIVALENCES

REPort PIn Equivalences

```
[-R0ot | -Module <module_name> | -All]  
(Setup / Verify Mode)
```

Lists all added pin equivalences and inverted pin equivalences. The equivalences were originally added using the `ADD PIN EQUIVALENCE` command. Inverted pin equivalences are denoted by a (-) next to the primary input pin name.

Tcl Command

```
report_pin_equivalences
```

Parameters

-R0ot	Displays pin equivalences from the root module. <i>This option is the default.</i>
-Module <module_name>	Displays pin equivalences from the specified module.
-All	Displays pin equivalences in all modules within the given defaults.

Related Commands

[ADD PIN EQUIVALENCES](#)

[DELETE PIN EQUIVALENCES](#)

REPORT PREDEFINED SYNCH_RULE

REPort PREDEFIned Synch_rule
[-ALL | <rule_name* ...>]
(Verify Mode)

Displays predefined synchronization rules. See [SET PREDEFINED SYNCH_RULE](#) for more information on how to customize these rules. For information on how to interpret this report, see "Using Predefined Synchronization Rules" in the Clock Domain Crossing Checks chapter of the *Conformal Extended Checks User Guide*.

Tcl Command

report_predefined_synch_rule

Parameters

-ALL	Displays all predefined synchronization rules. <i>This is the default.</i>
<rule_name* ...>	Displays the specified predefined synchronization rule(s).

Related Commands

[ADD SYNCHRONIZATION RULE](#)

[SET PREDEFINED SYNCH_RULE](#)

REPORT PRIMARY INPUTS

REPort PRimary Inputs

`[-Class <Full | System | User>]`
(Setup / Verify Mode)

Lists the design's primary input pins.

Tcl Command

`report_primary_inputs`

Parameters

<code>-Class</code>	Displays the specified class of primary inputs:	
	<code>Full</code>	Displays primary inputs from both the User and System classes. <i>This option is the default when you do not specify options.</i>
	<code>System</code>	Displays primary inputs from the original design.
	<code>User</code>	Displays primary inputs added with the <code>ADD PRIMARY INPUT</code> command.

Related Commands

[ADD PRIMARY INPUT](#)

[DELETE PRIMARY INPUTS](#)

REPORT PRIMARY OUTPUTS

REPort PRimary Outputs
(*Setup / Verify Mode*)

Lists the design's primary output pins.

Tcl Command

`report_primary_outputs`

REPORT PROPERTY STATISTICS

REPort PProperty Statistics

```
[ -Structural [ -NOLIST
                    | -LIST [ -ALL | -INputs | -SEQelm | -TIEd | -NOTIEd ] ]
    | -Expanded <depth#> ]
[ -TIME_unit <INFINITY | <depth#>> | -COMbinational]
[ -STOP_at <PI | TIED>]
[ -ANY_Gate
    | -Prop_session [ | -FLOating | -MULTI_driven
                      | -CONTention | -TRI_ON
                      | -TRI_OFF
                      | <-SET_RESET | -SR>
                      | -FSMReachability | -FSMDeadlock
                    ]
]
<pathname | id> ...
(Verify Mode)
```

Shows information about the logic cone of any gate or property.

Tcl Command

report_property_statistics

Parameters

-Structural	Reports the information for the fan-in cone of the gate with the specified path or ID. <i>This is the default.</i>
-NOLIST	Reports the information for the fan-in cone of the gate with the specified path or ID. <i>This is the default when you use -structural.</i>
-LIST	Reports the information for the fan-in cone of the gate with the specified path or ID, providing more detail on specified items.
-ALL	Report the information for the fan-in cone of the gate with the specified path or ID, including a listing of primary inputs, sequential elements, tied signals, and non-tied signals. <i>This is the default when you use -list.</i>

Conformal Extended Checks Reference Manual

Command Reference

-INputs	Reports the information for the fan-in cone of the gate with the specified path or ID, including a listing of the primary inputs.
-SEQelm	Reports the information for the fan-in cone of the gate with the specified path or ID, including a listing of the sequential elements.
-TIEd	Reports the information for the fan-in cone of the gate with specified path or ID, including a listing of the tied signals. Tied signals are those with constraints attached.
-NOTIEd	Reports the information for the fan-in cone of the gate with the specified path or ID, including a listing of the non-tied signals. Non-tied signals are those without constraints attached.
-Expanded <depth#>	Reports the information for the fan-in cone of the gate with the specified path or ID, unrolled <depth#> sequential levels.
-TIME_unit	Reports the information for the fan-in cone of the gate with the specified path or ID, stopping at the specified time units. Conformal Extended Checks reports <code>-time_unit infinity</code> if you do not specify either <code>-time_unit <depth#></code> or <code>-combinational</code> .
INFINITY	Reports the information for the entire fan-in cone of the gate with the specified path or ID.
<depth#>	Reports the information for the fan-in cone of the gate with the specified path or ID, stopping at the specified time units (that is, <depth#>). This option can be used to limit the fan-in cone by register levels. Note: This <depth#> does not mean clock cycles, it means time units.
-COMbinational	Reports the information for the fan-in cone of the gate with the specified path or ID, stopping at the first level of registers.

Conformal Extended Checks Reference Manual

Command Reference

-STOP_at	Reports the information for the fan-in cone of the gate with the specified path or ID stopping at the specified pins.
PI	Reports the information for the fan-in cone of the gate with the specified path or ID, stopping at primary inputs.
TIED	Reports the information for the fan-in cone of the gate with the specified path or ID, stopping at tied signals.
-ANY_Gate	Regards the specified ID or path as a gate instance and report the information for its fan-in cone. <i>This is the default.</i>
-Prop_session	Regards the specified ID or path as a property instance and report the information for its fan-in cone. Note: If the specified ID or path is not a property instance, Conformal Extended Checks will issue an error message. If the specified ID or path is a property instance but has many atomic properties (for example, -assert, -no_overflow), Conformal Extended Checks supplies a list of atomic properties for you to choose so that the command can be completed properly. If the specified ID or path is a property instance but has exactly one atomic property, Conformal Extended Checks reports the information for its fan-in cone.
-FLOating	Regards the specified ID or path as a floating bus property instance and report the information for its fan-in cone.
-MULTI_driven	Regards the specified ID or path as a multiply-driven bus property instance and report the information for its fan-in cone.
-CONTention	Regards the specified ID or path as a bus contention property instance and report the information for its fan-in cone.
-TRI_ON	Regards the specified ID or path as a tristate stuck-on property instance and report the information for its fan-in cone.

Conformal Extended Checks Reference Manual

Command Reference

`-TRI_OFF` Regards the specified ID or path as a tristate stuck-off property instance and report the information for its fan-in cone.

`-SET_RESET` | `-SR`
Regards the specified ID or path as a set-reset property instance and report the information for its fan-in cone.

`-FSMReachability`
Regards the specified ID or path name as an FSM Reachability property instance and report the information for its fan-in cone.

`-FSMDeadlock` Regards the specified ID or path name as an FSM Deadlock property instance and report the information for its fan-in cone.

`<pathname | id> ...` Reports the information for the fan-in cone of the gate with the specified path or ID.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Command

DIAGNOSE STATIC PROPERTY

REPORT PROVED DATA

REPort PRoVED Data

```
[-SUMmary | -NOSUMmary | -ONLYSummary]
[-ALL
|<-TRI_state [-ALL | -PO | <id | pathname*> ...]
| -DONT_CARE
| [-ALL | -X_ASSIGNment | -RANGE_overflow
| -FULL_case | -PARAllel_case | <id | pathname*> ...
|
| -BRANCH_enable
| [-ALL | -IF_else | -CASE
| -DEFault | -FOR_loop | <id | pathname*> ...
|
| < -BUS | -MULTI_PORT
| -SET_RESET
| -FSM | -FSMReachability
| -FSMTransition | -FSMDeadlock
| -NO_DIVide_by_zero
| -ENCoding_completeness
>
| [-ALL | <id | pathname*> ...]
> ...
]
[-TIME_unit <INFinity | <depth#> > | -COMbinational]
[-STatus | -VERbose]
[ | -PASS | -FAIL | -EXPlored | -NOTrun] ...
[ | -WARNing | -NOWARNing ]
[ | -FILE <filename>]
(Verify Mode)
```

This report has information about proof results. Refer to the *Conformal Extended Checks User Guide* for specific information about how Conformal derives proof results.

The default report reflects the parameters specified by the last `PROVE` command. It includes the following information grouped by category (for example: Bus, tristate):

- ID number and path

- Type

This column appears for applicable properties. For example: Don't Care shows parallel-case and full-case in the "Type" column.

- Status

Indicates proof status; for example, Pass.

- Depth

Conformal Extended Checks Reference Manual

Command Reference

Indicates proof depth; for example: an integer, infinity, or combinational. This information applies to instances with a status other than "Pass".

The default report ends with a summary that lists property type totals with a breakdown of the number of properties in each category that passed, failed, or were not run.

Customize the Proved Data report to show specific categories of checks. For example, specify a report that shows only those properties that failed and were processed with a depth of infinity.

Tailor the scope of the return by specifying format and summary choices:

- Status (default)
- Verbose
- Summary (default)
- Nosummary
- Onlysummary

Tcl Command

`report_proved_data`

Parameters

<code>-SUMmary</code>	Concludes the specified report with a list of property IDs, status, and subtotals for each pass/fail/not run status type. <i>This is the default.</i>
<code>-NOSUMmary</code>	Does not include a summary with the specified report.
<code>-ONLYSummary</code>	Reports only a table of subtotals for each pass/fail/not run status type.
<code>-ALL</code>	Reports all proof results. <i>This is the default.</i>
<code>-TRI_state</code>	Reports the proof results for the specified tristate properties.
<code>-ALL</code>	Reports the proof results for all tristate properties. <i>This is the default.</i>

Conformal Extended Checks Reference Manual

Command Reference

-PO	Reports the proof results for only tristates directly connected to primary outputs.
<id pathname*> ...	Reports the proof results for tristate properties with the specified IDs or paths. Wildcards are supported for paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-DONT_CARE	Reports the proof results for the specified Don't Care properties.
-ALL	Reports the proof results for all Don't Care properties. <i>This is the default.</i>
-X_ASSIGNment	Reports the proof results for Don't Care X-assignment properties.
-RANGE_overflow	Reports the proof results for Don't Care array index overflow properties.
-FULL_case	Reports the proof results for Don't Care properties generated by the Synopsys <code>full_case</code> directive.
-PARAllel_case	Reports the proof results for Don't Care properties generated by the Synopsys <code>parallel_case</code> directive.

Conformal Extended Checks Reference Manual

Command Reference

	<code><id pathname*> ...</code>	<p>Reports the proof results for Don't Care properties with the specified IDs or paths. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
<code>-BRANCH_enable</code>		<p>Reports the proof results for Branch Enable properties, as specified.</p>
	<code>-ALL</code>	<p>Reports the proof results for all Branch Enable properties. <i>This is the default.</i></p>
	<code>-IF_else</code>	<p>Reports the proof results for Branch Enable properties for if-else conditional branches.</p>
	<code>-CAsE</code>	<p>Reports the proof results for Branch Enable properties for case conditional branches. (This does not include default branches.)</p>
	<code>-DEFAult</code>	<p>Reports the proof results for Branch Enable properties for case-default conditional branches.</p>
	<code>-FOR_loop</code>	<p>Reports the proof results for Branch Enable <code>for_loop</code> properties.</p>
	<code><id pathname*> ...</code>	<p>Reports proof results for the specified Branch Enable properties. Wildcards are supported for paths.</p> <p>Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.</p>
<code>-BUS</code>		<p>Reports the proof results for the specified Bus properties.</p>
<code>-MULTI_PORT</code>		<p>Reports the proof results for the specified Multi-port properties.</p>

Conformal Extended Checks Reference Manual

Command Reference

-SET_RESET	Reports the proof results for the specified Set-Reset properties.
-FSM	Reports the proof results for the specified finite state machine properties.
-FSMReachability	Reports the proof results for the specified finite state machine reachability properties.
-FSMTransition	Reports the proof results for the specified finite state machine transition properties.
-FSMDeadlock	Reports the proof results for the specified finite state machine deadlock properties.
-NO_DIVide_by_zero	Reports the proof results for the specified No-Divide-By-Zero properties. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
-ENCoding_completeness	Reports the proof results for the specified Encoding-Completeness. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
-ALL	Reports the proof results for all properties of the specified type (for example, all predefined). <i>This is the default.</i>
<id pathname*> ...	
	Reports the proof results for properties with the specified IDs or paths. Wildcards are supported for paths.
	Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-TIME_unit	Reports proof results (for the specified category of properties) for the specified sequential depth. Conformal Extended Checks reports <code>-time_unit infinity</code> if you do not specify either <code>-time_unit depth</code> or <code>-combinational</code> .
INFINITY	Proof depth = infinity.
<depth#>	Proof depth = N (N is an integer)

Conformal Extended Checks Reference Manual

Command Reference

-COMbinational	Reports proof results (for the specified category of properties) that were proved with the combinational method.
-STatus	Reports the proof results in the "Status" format. This format includes expanded information and concludes with a summary. <i>This is the default.</i>
-VERbose	Reports the proof results in the "Verbose" format. This format includes expanded information about each instance.
-PASS	Reports only the properties with status = PASS. Note: If you do not specify <code>-pass</code> , <code>-fail</code> , <code>-explored</code> , or <code>-notrun</code> , Conformal Extended Checks reports properties of all statuses.
-FAIL	Reports only the properties with status = FAIL.
-EXPlored	Reports only the properties with status = EXPLORED.
-NOTrun	Reports only the properties with status = NOT RUN.
-WARNing	Reports properties that have warnings. ("Y" in the Warning column.) Note: If you do not specify either <code>-warning</code> or <code>-nowarning</code> , Conformal Extended Checks reports both.
-NOWARNing	Reports properties that have no warnings. ("N" in the Warning column.) Note: If you do not specify either <code>-warning</code> or <code>-nowarning</code> , Conformal Extended Checks reports both.
-FILE <filename>	Saves the report to the specified file. Note: If you use this option the report is not displayed on your screen.

Related Commands

ADD STATIC PROPERTY

ADD IGNORED PROPERTY

DIAGNOSE STATIC PROPERTY

Conformal Extended Checks Reference Manual

Command Reference

PROVE

REPORT EXTRACTED PROPERTY

REPORT FSM

REPORT GATE

REPORT STATIC PROPERTY

REPORT RENAMING RULE

REPort RENaming Rule

`<rulename*>`
(*Setup / Verify Mode*)

Displays the list of all renaming rules that you created using `ADD RENAMING RULE`.

Tcl Command

`report_renaming_rule`

Parameters

`<rulename*>` Specifies the rule name to report. This accepts wildcards.

Related Commands

`ADD RENAMING RULE`

`DELETE RENAMING RULE`

REPORT RESET_SYNC MODULE

REPort REset_sync Module

```
<-ALL | <reset_sync_module_name* ...> >  
(Verify Mode)
```

Reports set and reset synchronizers defined by the `ADD RESET_SYNC MODULE` command.

Tcl Command

```
report_reset_sync_module
```

Parameters

`-ALL` Reports all set and reset synchronizers.

`<reset_sync_module_name*>`

Specifies the set and reset synchronizer(s) to report. This accepts wildcards.

Related Commands

- ADD RESET_SYNC MODULE
- DELETE RESET_SYNC MODULE

REPORT RESET SOURCE

REPort REset Source

(Verify Mode)

Reports the hierarchical paths of the reset input signals that were specified with the `ADD RESET SOURCE` command.

Tcl Command

`report_reset_source`

Related Commands

- ADD RESET SOURCE
- DELETE RESET SOURCE

REPORT RULE CHECK

REPort Rule Check

```
[ -All | -MODIfied | -RTL | -MODeling | -SDC | -LP | -CPF | -1801 | <rule_name*  
...>  
[-CREATE_RULE_FILTER]  
[-SETTING] ]  
[-File <filename> [<linenumber>] ]  
[-FILTER <filter_expression> [-REGEXP]]  
[-FILTERED_BY <filter_name*>]  
[-ATTRIBUTES]  
[-MODULE <module_name>]  
[ | -Design | -Library]  
[-SUMmary | -Verbose | -OVERVIEW]  
[-STATUS < ALL | FAIL | PASS | IGNORED | NOT-APPLICABLE>]  
[-DEFAULT_SEVERITY]  
[-Ignore]  
[-Note]  
[-Warning]  
[-Error]  
[-CATEGORY [-XML <filename>]]  
[-LIMIT [<natural_number>]]  
[-NOHidden | -HIDden | -COMplete  
| -FILTERED_out | -NOFILTERED_out  
| -WAIved | -NOWaived]  
[-HELP]  
[-OCCURRENCE_COUNT | -MAX_PRINT_COUNT <limit>]  
[-OLD_RULE_NAME]  
[-RULE_SEParation <num>]  
(Setup / Verify Mode)
```

Displays the status of the predefined rules. The status information consists of the number of occurrences and severity of violations.

Note: Rule violation severity is set with the [SET RULE HANDLING](#) command. Rules with a severity of "Ignore" are not reported except with the `rule_name` option; and in this case, Conformal reports only the number of occurrences.

Note: This command will report any limit set on the number of occurrences of any reported rule.

To view a complete list of the predefined rule messages that Conformal Extended Checks has generated for your design, use the following command:

```
report rule check -summary
```

The Conformal Extended Checks graphical user interface has rule managers for the RTL and Modeling rules. These windows allow you to view and debug violations in a more visually

accessible manner. Refer to the *Conformal Extended Checks User Guide* for specific information about GUI rule managers.

Note: A rule occurrence is "hidden" if it is filtered out (excluded by one or more rule filters), or if it is waived, or both.

Tcl Command

report_rule_check

Parameters

-All	Displays all predefined rule messages. <i>This option is the default.</i>
-MODIfied	Reports all the rule check violations that have a different severity level than the original default.
-RTL	Displays only RTL rule messages.
-MODeling	Displays only modeling rule messages.
-SDC	Displays only the SDC rule messages. There is only one SDC rule check for Conformal Extended Checks, which is called SDC_ERROR. This rule check represents all SDC violations. For more information on SDC rule checks that fall under this one message, refer to "Understanding SDC Rule Checks" in the <i>Conformal Constraint Designer User Guide</i> .
-LP	Displays power intent quality checks for the library or design, and all structural low power checks for the implemented logical/physical design.
-CPF	Displays CPF rule checks.
-1801	Displays native 1801 rule checks.
<rule_name* ...>	Displays only the specified predefined rule messages. This accepts wildcards.
-CREATE_RULE_FILTER	Creates Tcl-based filters for all the occurrences reported by REPORT RULE CHECK.
-SETTING	Displays the current severity level for the rule.

Conformal Extended Checks Reference Manual

Command Reference

-File <filename> [<linenumber>]

Reports all rule check messages in a file. With the <linenumber> option, you can report all rule check messages for a specific line number.

-FILTER <filter_expression>

Report only occurrences for which the specified `filter_expression` evaluates to true. The syntax for `filter_expression` is the same as for the Tcl `find` command.

This is a Conformal Low Power command option.

-FILTERED_BY <filter_name*>

Reports only the occurrences filtered by the specified filter(s). Wildcards are accepted.

-ATTRIBUTES

Reports attributes of occurrences.

-REGEXp

Specifies that the pattern matching operators used inside the `filter_expression` (such as `=~`) should use regular expressions as patterns, rather than glob-style patterns.

-Module <module_name>

Reports the rule checks that are specific to the specified module.

-Design

Reports the design rule violations.

If you do not specify `-design` or `-library`, the Conformal software reports rule violations from *both* designs and libraries.

-Library

Reports the library rule violations.

If you do not specify `-design` or `-library`, the Conformal software reports rule violations from *both* designs and libraries.

-Summary

Displays the status in summary format. *This is the default.*

-Verbose

Displays the status in verbose format.

-OVERVIEW

Displays an overview of the rule checker status (includes rule name, severity, applied analysis style, and status). You can also list all the critical rules using `REPORT RULE CHECK - overview -critical`.

-STATUS

Selects rules to be displayed based on their execution status.

Conformal Extended Checks Reference Manual

Command Reference

	ALL—All rules. <i>This is the default.</i>
	FAIL—Rules with failure occurrences.
	PASS—Rules without failure occurrences.
	IGNORED—Rules that are not run because their severity is 'ignore'.
	NOT-APPLICABLE—Rules not applicable for the run.
-DEFAULT_SEVERITY	Include the default rule severity. This option can only be used with -OVERVIEW option.
-Ignore	Displays violations that have a severity level of ignore.
-Note	Displays violations that have a severity level of note.
-Warning	Displays violations that have a severity level of warning.
-Error	Displays violations that have a severity level of error.
-CATegory	Reports RTL rules that fall under "Design intent mismatch" and Synthesis/simulation mismatch".
-XML <filename>	Writes the results of -CATegory to an XML file. The report is also written to the project directory (if set) and can be viewed through the web interface.
-LIMit [<natural_number>]	Limits the verbose display of all the reported rules to the specified number of occurrences. Used with -help, it reports the rule checking limit set by the set rule handling -limit command.
-NOHidden	Reports only occurrences that are not hidden. <i>This is the default.</i>
-HIDden	Reports only occurrences that are hidden.
-COMplete	Reports all occurrences regardless whether they are hidden or not
-FILTERED_out	Reports only occurrences excluded by filters (regardless whether they are waived or not).
-NOFILTERED_out	Reports only occurrences not excluded by filters (regardless whether they are waived or not).
-WAIVED	Reports only occurrences that are waived (regardless whether they are excluded by filters or not).

Conformal Extended Checks Reference Manual

Command Reference

-NOWaived	Reports only occurrences that are not waived (regardless whether they are excluded by filters or not).
-HELP	Displays the rule name, default severity level, and message.
-OCCURRENCE_COUNT	Report the number of occurrences even if it is zero.
-MAX_PRINT_COUNT <limit>	Limits the verbose display of all the reported rules to the specified number of occurrences. Used with <code>-help</code> , it reports the rule checking limit set by the <code>set rule handling -limit</code> command.
-OLD_RULE_NAME	<p>For 1801 rule checks, lists the current rule name and its old rule name (before 15.1). For example:</p> <pre>SETUP> report rule check PG_CONN_SUPPLY_NET_SRC_CONFLICT -old_rule_name</pre> <p>PG_CONN_SUPPLY_NET_SRC_CONFLICT (PG_CONN5): Supply net driven by a power source of conflicting type. Severity: Error Occurrence:3</p> <p>In this example, PG_CONN_SUPPLY_NET_SRC_CONFLICT is the current rule name and PG_CONN5 was the rule check's name before 15.1.</p> <p>To view the old rule names for all 1801 rule checks, use "report rule check -1801 -old_rule_name".</p>
-RULE_SEParation <num>	<num> specifies the number of blank lines to print before each rule is reported. Without this option, no blank lines are printed before each reported rule.

Related Commands

SET DOFILE ABORT

SET RULE HANDLING

WRITE RULE CHECK

REPORT RULE FILTER

REPort RUle Filter

<filtername* ...>
(Setup / Verify Mode)

Reports the currently defined rule filters, in dofile format.

Tcl Command

report_rule_filter

Parameters

<filtername* ...> Specifies the name(s) of the filter(s) to report.

Examples

the following command saves the ABC filter report to a file named XYZ:

```
report rule filter ABC > XYZ
```

Related Commands

ADD RULE FILTER

DELETE RULE FILTER

REPORT SEARCH PATH

REPort SEarch Path

[| -Design | -Library | -POWER_intent]
(Setup / Verify Mode)

Displays the paths Conformal searches to locate filenames included in the READ DESIGN and READ LIBRARY commands.

Tcl Command

report_search_path

Parameters

-Design	Report the design search paths. If you do not specify -design or -library, Conformal reports search paths for both designs and libraries.
-Library	Report the library search paths. If you do not specify -design or -library, Conformal reports search paths for both designs and libraries.
-POWER_intent	This is a low power command option. Reports the search path(s) for power intent files.

Related Commands

ADD SEARCH PATH

DELETE SEARCH PATH

READ DESIGN

READ LIBRARY

REPORT SET SOURCE

REPort SEt Source
(*Verify Mode*)

Reports the hierarchical paths of the set input signals that were specified with the `ADD SET SOURCE` command.

Tcl Command

`report_set_source`

Related Commands

- [ADD SET SOURCE](#)
- [DELETE SET SOURCE](#)

REPORT STATIC PROPERTY

REPort Static Property

```
[ -ALL
| < -TRI_state [-ALL |-PO |<id | pathname*>... ]
|-DONT_CARE
  [ -ALL | -X_ASSIGNment | -RANGE_overflow
  |-FULL_case | -PARAllel_case | <id | pathname*> ...
  ]
|-BRANCH_enable
  [ -ALL | -IF_else | -CAsE
  |-DEFault | -FOR_loop | <id | pathname*> ...
  ]
|< -BUS | -MULTI_PORT
  |-SET_RESET
  |-FSM |-FSMReachability
  |-FSMTransition | -FSMDeadlock
  |-NO_DIVide_by_zero
  |-ENCoding_completeness
  >
  [-ALL |<id | pathname*> ... ]
> ...
]
[-SUMmary | -VERbose]
(Verify Mode)
```

Displays information about properties added to the "Prove" list with the `ADD STATIC PROPERTY` command. The default report is a summary format that lists the number of instances of each property.

Conformal Extended Checks maintains a "prove list" of properties it will verify using the `Prove` command. Add or remove checks from this list with the `ADD STATIC PROPERTY` and `DELETE STATIC PROPERTY` commands.

Conformal Extended Checks generates this report for individual properties, properties by type, or all properties.

Note: The `REPORT STATIC PROPERTY` command displays only those checks currently on the prove list. Alternatively, you can view all extracted properties with the `REPORT EXTRACTED PROPERTY` command or the Static Property Manager in the Conformal Extended Checks GUI.

Tcl Command

`report_static_property`

Parameters

-ALL	Reports all properties that are on the "Prove" list. <i>This is the default.</i>
-TRI_state	Reports the specified tristate properties that are on the "Prove" list.
-ALL	Reports all tristate properties that are on the "Prove" list. <i>This is the default.</i>
-PO	Reports only tristate pins on the "Prove" list that are connected to the primary output.
<id pathname*> ...	Reports tristate properties on the "Prove" list with the specified IDs or hierarchical paths. Wildcards are supported for paths. Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.
-DONT_CARE	Reports the specified Don't Care properties that are on the "Prove" list.
-ALL	Reports all Don't Care properties that are on the "Prove" list. <i>This is the default.</i>
-X_ASSIGNment	Reports Don't Care <code>X_assignment</code> properties that are on the "Prove" list.
-RANGE_overflow	Reports Don't Care array index overflow properties that are on the "Prove" list.
-FULL_case	Reports Don't Care <code>full_case</code> synthesis directive properties that are on the "Prove" list.
-PARAllel_case	

Conformal Extended Checks Reference Manual

Command Reference

Reports Don't Care `parallel_case` synthesis directive properties that are on the "Prove" list.

`<id | pathname*> ...`

Reports Don't Care properties on the "Prove" list that have the specified IDs or paths. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

`-BRANCH_enable`

Reports the specified Branch Enable properties that are on the "Prove" list.

`-ALL` Reports all Branch Enable properties that are on the "Prove" list. *This is the default.*

`-IF_else` Reports Branch Enable if-else conditional branch properties that are on the "Prove" list.

`-CAsE` Reports Branch Enable case conditional branch properties that are on the "Prove" list. (This does not include default branches.)

`-DEFAult` Reports Branch Enable case-default conditional branch properties that are on the "Prove" list.

`-FOR_loop` Reports Branch Enable `for_loop` properties that are on the "Prove" list.

Conformal Extended Checks Reference Manual

Command Reference

`<id | pathname*> ...`

Reports the specified Branch Enable properties that are on the "Prove" list. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

<code>-BUS</code>	Reports the specified Bus properties that are on the "Prove" list.
<code>-MULTI_PORT</code>	Reports the specified Multi-port properties that are on the "Prove" list.
<code>-SET_RESET</code>	Reports the specified Set-Reset properties that are on the "Prove" list.
<code>-FSM</code>	Reports the specified finite state machine properties that are on the "Prove" list.
<code>-FSMReachability</code>	Reports the specified finite state machine reachability properties that are on the "Prove" list.
<code>-FSMTransition</code>	Reports the specified finite state machine transition properties that are on the "Prove" list.
<code>-FSMDeadlock</code>	Reports the specified finite state machine deadlock properties that are on the "Prove" list.
<code>-NO_DIVide_by_zero</code>	Reports the specified No-Divide-By-Zero properties. The No-Divide-By-Zero predefined check ensures that divisors are never zero.
<code>-ENCoding_completeness</code>	Reports the specified Encoding-Completeness properties. The Encoding-Completeness predefined check ensures that any reachable state is listed under <code>fromstates</code> in the FSM encoding file.
<code>-ALL</code>	<p>Reports all properties for the specified type. For example, the following command reports all Bus properties that are on the "Prove" list:</p> <pre>report static property -bus -all</pre> <p><i>This is the default.</i></p>

Conformal Extended Checks Reference Manual

Command Reference

<id | pathname*> ...

Reports properties for the specified IDs or paths. Wildcards are supported for paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different software version.

-SUMmary

Reports with the "Summary" format. This format lists the number of instances of each property type. *This is the default.*

-VERbose

Reports with the "Verbose" format. This format includes property type and each instance ID number and name.

Related Commands

ADD IGNORED PROPERTY

ADD STATIC PROPERTY

DELETE IGNORED PROPERTY

DELETE STATIC PROPERTY

DIAGNOSE STATIC PROPERTY

PROVE

REPORT EXTRACTED PROPERTY

REPORT FSM

REPORT GATE

REPORT IGNORED PROPERTY

REPORT PROVED DATA

REPORT SYNC INPUT

REPort Sync Input
(Verify Mode)

Displays all the synchronized inputs that were declared with the `ADD SYNC INPUT` command.

Tcl Command

```
report_sync_input
```

Related Commands

`ADD SYNC INPUT`

`DELETE SYNC INPUT`

REPORT SYNCHRONIZATION RULE

REPort SYNchronization Rule

`[-ALL | <rule_name* ...>]`
(Verify Mode)

Lists the specified synchronization rules that were previously created with the ADD SYNCHRONIZATION RULE command.

Tcl Command

`report_synchronization_rule`

Parameters

<code>-ALL</code>	Displays all synchronization rules. <i>This is the default.</i>
<code><rule_name* ...></code>	Reports synchronization rules with the specified rule names.

Related Commands

ADD SYNCHRONIZATION RULE

DELETE SYNCHRONIZATION RULE

REPORT TIED SIGNALS

REPort Tied Signals

```
[-ROot | -Module <module_name>]  
[ |-TIE0 | -TIE1 | -TIEZ | -TIEX]  
[ |-Net | -Pin]  
[-Class <Full | System | User >]  
(Setup / Verify Mode
```

Displays signals in the design that are assigned to 0 or 1 using the `ADD TIED SIGNALS` command or to 0, 1, X, or Z within the design.

Tcl Command

```
report_tied_signals
```

Parameters

<code>-ROot</code>	Displays tied signals in the root module. <i>This is the default.</i>
<code>-Module <module_name></code>	Displays tied signals in the specified module.
<code>-TIE0</code>	Displays signals tied to logic 0. If you do not specify the logic, Conformal Extended Checks displays signals tied to logic 0, 1, Z, and X.
<code>-TIE1</code>	Displays signals tied to logic 1. If you do not specify the logic, Conformal Extended Checks displays signals tied to logic 0, 1, Z, and X.
<code>-TIEZ</code>	Displays signals tied to logic Z. If you do not specify the logic, Conformal Extended Checks displays signals tied to logic 0, 1, Z, and X.
<code>-TIEX</code>	Displays signals tied to logic X. If you do not specify the logic, Conformal Extended Checks displays signals tied to logic 0, 1, Z, and X.
<code>-Net</code>	Displays net names with assigned tied signals. If you do not specify either <code>-net</code> or <code>-pin</code> , Conformal Extended Checks displays both net and pin names that with assigned tied signals.
<code>-Pin</code>	Displays pin names with assigned tied signals. If you do not specify either <code>-net</code> or <code>-pin</code> , Conformal Extended Checks displays both net and pin names that with assigned tied signals.
<code>-Class</code>	Display one of the following classes of tied signals:

Conformal Extended Checks Reference Manual

Command Reference

Full	Tied signals from both the user and system classes. <i>This is the default.</i>
System	Tied signals from the original design.
User	Tied signals added with the ADD TIED SIGNALS command.

Related Commands

ADD TIED SIGNALS

DELETE TIED SIGNALS

REPORT VALIDATED DATA

In the categorization flow, you have the following options:

REPort Validated Data

```
[-CATegory <rule_name* | MIXed_rule | FAIL>]
[-NOCLock_name | -CLock_name]
[-NOPIN_name | -PIN_name]
[-SUMmary | -VERbose]
[-WORD_level | -BIT_level]
(Verify Mode)
```

In the normal validation flow, you have the following options:

REPort Validated Data

```
[ | -STRUctural
  | -FUNctional [SOURCE_DATA] [DESTINATION_DATA]
                  [MUX_enable] [SINGLE_bit_change]
  | -SET [-ALL | <ID | instance_name> ...] ]
  | -RESET [-ALL | <ID | instance_name> ...] ]
]
[-SOURce <-ALL | <clock_domain* ...> >]
[-DESTination <-ALL | <clock_domain* ...> >]
[-FROM <-ALL | -REG | -PI | -BBOx | instance* ...>]
[-TO <-ALL | -REG | -PO | -BBOx | instance* ...>]
[<-MODULE | -INSTance> <name* ...>
  [-RECURSive | -NORECURSive]
  [ | -INTERface]]
[| -PASS | -FAIL ] [| -CONvergent]
[-SYNC_RULE <rule_name* ...>]
[-NOCLock_name | -CLock_name]
[-SUMmary | -VERbose
  [-FAILURE_CAUSE]
  [-NOSPEC | -SPEC]
]
[-WORD_level | -BIT_level]
[ | -REPORT_PATH
  [-LIBRARY_boundary | -NOLIBRARY_boundary]
  [-NOFULL_NAME | -FULL_NAME]
]
[ | -LOGic | -DIRect]
(Verify Mode)
```

In the low power flow, you have the following options:

Conformal Extended Checks Reference Manual

Command Reference

REPort Validated Data LP

```
[ -ALL
  | -ISolation [-ASSERT]
    [-ALL | <name* ...> [-MODule | -INStance]]
  | -CROssing [-SOURce [-ALL | <power_domain>...]]
    [-DESTination [-ALL | power_domain>...]]
  | -REtention [-ASSERT] [-CLOCK_value]
    [-ALL | <name*> ... [-MODule | -INStance]]
]
[ -SUMmary | -VERbose]
[ | -PASS | -FAIL | -ED]
[ | -FILE <filename> [-NOReplace | -REPlace]]
]
(Verify Mode)
```

Displays information about validation results and includes debugging information.

The available options for `REPORT VALIDATED DATA` depend on your current flow. When you use the predefined synchronization rules alone (`SET PREDEFINED SYNCH_RULE`), Conformal automatically enters the *Categorization* flow when you issue the `VALIDATE` command. However, if you have user-defined synchronization rules (`ADD SYNCHRONIZATION RULE`), Conformal enters the normal *Validation* flow.

For the Low Power flow, you can use this command to report validated results of added power checks. To report the power check results, see the third syntax group for this command (for low power flow).

For more information on this command, its sample output, and the subchecks under each synchronization rule, see "Reporting Validated Data" in the Clock Domain Crossing Checks chapter of the *Conformal Extended Checks User Guide*.

Tcl Command

```
report_validated_data
```

Parameters

While in the Categorization flow, you have the following options:

<code>-CATegory</code>	Reports validation results for the specified predefined synchronization rule category only.
------------------------	---

Conformal Extended Checks Reference Manual

Command Reference

<code><rule_name*></code>	Reports the number of CDC paths that fall under this specific predefined synchronization rule.
<code>MIXed_rule</code>	Reports the number of CDC paths that fall under <i>MIXED</i> . A CDC path falls under <i>MIXED</i> when the vector has individual bits that are classified into different categories for word-level reporting.
<code>FAIL</code>	Reports the number of CDC paths that do not satisfy any predefined synchronization rule.
<code>-NOCLock_name</code>	Reports only clock domains, not actual clocks. <i>This is the default.</i>
<code>-CLOck_name</code>	Of the CDC Checks results, reports the clock domains and the actual clocks influenced by each instance.
<code>-NOPIN_name</code>	Does not report pin names. <i>This is the default.</i>
<code>-PIN_name</code>	Reports pin names.
<code>-SUMmary</code>	Displays a summary of the validation results, in tabular format.
<code>-VERbose</code>	Displays validation results in verbose format.
<code>-WORD_level</code>	Reports CDC results at the word-level. <i>This is the default.</i>
<code>-BIT_level</code>	Reports CDC results at the bit-level.

While in the normal Verification flow, you have the following options:

<code>-STRUctural</code>	Reports the Structural CDC Check results for the specified portion of the design. Structural Checks include many checks such as <code>cdc_path_logic_type_check</code> , <code>cdc_path_destination_check</code> , <code>sync_chain_dff_number_check</code> , and so forth. (Refer to the list in the definition for this command.)
<code>-FUNctional</code>	Reports the Functional CDC Check results for the specified portion of the design. Functional CDC Checks include Source Data, Destination Data, MUX Enable, and Single-bit Change checks. (Refer to the list in the definition for this command.)
<code>SOURCE_DATA</code>	Reports the Source Data Functional CDC Check results for the specified portion of the design.

Conformal Extended Checks Reference Manual

Command Reference

	DESTINATION_DATA	Reports the Destination Data Functional CDC Check results for the specified portion of the design.
	MUX_enable	Reports the MUX Enable Functional CDC Check results for the specified portion of the design.
	SINGLE_bit_change	Reports the Single-bit Change Functional CDC Check results for the specified portion of the design.
-SET		Reports the results for the check for asynchronous set pins of the flip-flops. This check ensures that they are consistent with the clock domain.
-RESET		Reports the results for the check for asynchronous reset pins of the flip-flops. This check ensures that they are consistent with the clock domain.
	-ALL	Specifies set or reset results for all instances.
	-ID	Specifies the assigned ID of the element for which the set or reset validation results must be reported.
	<instance_name ...>	Specifies the hierarchical name of the instance for which the set or reset validation results must be reported.
-SOURCE		Of the CDC Check results, reports only those with the specified source clock domain.
	-ALL	Uses all clock domains as the source.
	<clock_domain* ...>	Uses the specified clock domains as the source. This accepts wildcards.
-DESTination		Of the CDC Check results, reports only those with the specified destination clock domain.
	-ALL	Uses all clock domains as the destination.

Conformal Extended Checks Reference Manual

Command Reference

	<code><clock_domain* ...></code>	Uses the specified clock domains as the destination. This accepts wildcards.
-FROM	Of the CDC Check results, reports only those that start from the specified key points.	
	-ALL	Uses all types of key points as the start of the paths.
		The type of key points includes registers, primary inputs, and blackboxes (outputs).
	-REG	Uses registers as the start of the paths.
	-PI	Use primary inputs as the start of the paths.
	-BBOX	Uses blackbox outputs as the start of the paths.
	<code><instance_name ...></code>	Uses the specified instances as the start of the paths. This accepts wildcards.
		The instances can be registers, primary inputs, or blackbox outputs.
-TO	Of the CDC Check results, reports only those that end at the specified key points.	
	-REG	Uses registers as the end of the paths.
	-PO	Uses primary outputs as the end of the paths.
	-BBOX	Uses blackbox inputs as the end of the paths.
	<code><instance_name ...></code>	Uses the specified instances as the end of the paths. This accepts wildcards.
		The instances can be registers, primary outputs, or blackbox inputs.
-MODULE INSTANCE	<code><name* ...></code>	Reports validation data for the specified modules or module instance path names.
-RECURSIVE		Reports all CDC paths under the specified module or instance hierarchically.

Conformal Extended Checks Reference Manual

Command Reference

-NOREcursive	Reports all CDC paths within the specified module or instance only.
-INTERface	Used with -REcursive, this reports all CDC paths which across hierarchical boundary and under the specified module or instance hierarchically. Used with -NOREcursive, this reports all CDC paths which across hierarchical boundary of the specified module or instance only.
-PASS	Of the CDC Checks results, reports the passed CDC paths. Note: For each CDC path, if <i>any</i> of its sync rules is PASS its validated result is PASS.
-FAIL	Of the CDC Checks results, reports the failed CDC paths. Note: For a CDC path, if <i>all</i> of its sync rules are FAIL, its validated result is FAIL.
-CONvergent	Of the CDC results, reports only those that relate to convergence checks.
-SYNC_RULE <rule_name* ...>	Of the CDC Checks results, reports only those related to the specified sync rules. This accepts wildcards.
-NOCLOck_name	Reports only clock domains, not actual clocks. <i>This is the default.</i>
-CLOck_name	Of the CDC Checks results, reports the clock domains and the actual clocks influenced by each instance.
-SUMmary	Reports CDC Checks results in the Summary format. <i>This is the default.</i>
-VERbose	Reports CDC Checks results in the Verbose format.
-FAILURE_CAUSE	Reports the failure cause of each sub-check in each synchronization rule.
-NOSPEC	Does not report the specification of the synchronization rule.
-SPEC	Reports the specification of the synchronization rule.
-WORD_level	Reports CDC results at the word-level. <i>This is the default.</i>
-BIT_level	Reports CDC results at the bit-level.

Conformal Extended Checks Reference Manual

Command Reference

-REPORT_PATH	Reports validation data with the specified display options.
-LIBrary_boundary	Reports validation data to the boundary of library cells. This report does not show the contents of the library cells in the path. <i>This is the default.</i>
-NOLIBrary_boundary	Reports validation data down to the primitive cell (display the full path).
-NOFULL_NAME	Limits the width of the report to 80 characters. <i>This option is the default.</i>
-FULL_NAME	Ignores the character limit to allow full names in the report.
-LOGic	Reports validation data that has logic in the CDC path. "Logic" refers to logic other than wire, buffer, or inverter.
-DIRect	Reports validation data that has only wire, buffers, or inverters in the CDC path.

While in the Low Power flow, you have the following options:

-ALL	Specifies all validated results of added power checks.
-ISOLation	Indicates that isolation cell(s) are to be reported.
-ASSERT	Displays the asserted isolation cell control pins.
-ALL <name* ...>	Specifies that all or specific isolation cell(s) will be reported.
-MODule -INSTance	Indicates if names specified after -ISOLation are module names or hierarchical instance path names.
-CROSSing	Indicates that crossing failure(s) are to be reported.
-SOURCE [-ALL <power_domain>...]	

Conformal Extended Checks Reference Manual

Command Reference

	Specifies that all or specific crossing from the specified power domain(s) will be reported. These are the starting (source) power domain(s) of the crossing failures.
	<code>-DESTination [-ALL power_domain ...>]</code>
	Specifies that all or specific crossing to the specified power domain(s) will be reported. These are the ending (destination) power domain(s) of the crossing failures.
<code>-RETention</code>	Indicates that retention-register cell(s) are to be reported.
	<code>-ASSERT</code>
	Displays the asserted retention cell control pins.
	<code>-CLOCK_value</code>
	Displays the clock value functional check for retention cells.
	<code>-ALL <name* ...></code>
	Specifies that all or specific retention-register cell(s) will be reported.
	<code>-MODule -INSTance</code>
	Indicates if names specified after <code>-RETention</code> are module names or hierarchical instance path names.
<code>-SUMmary</code>	Displays a summary of the validation results, in tabular format.
<code>-VERbose</code>	Displays validation results in verbose format.
<code>-PASS</code>	Filters the report so that it only outputs checks that have a PASS status.
<code>-FAIL</code>	Filters the report so that it only outputs checks that have a FAIL status.
<code>-ED</code>	Filters the report so that it only outputs checks that have an ED status.
<code>-FILE <filename></code>	Specifies the output filename for the report. This preserves long instance pathnames.
	<code>-NOReplace</code>
	Does not replace an existing file. <i>This is the default.</i>
	Conformal Extended Checks issues a warning if you use a filename that already exists.

Conformal Extended Checks Reference Manual

Command Reference

-REPlace

If the output file exists, this replaces its contents.

Examples

For sample output, see "Reporting Validated Data" in the Clock Domain Crossing Checks chapter of the *Conformal Extended Checks User Guide*.

- Report all validated data:

```
report validated data
```

- Provide a verbose report for all Structural CDC check results:

```
report validated data -structural
```

- Report all CDC paths with passed validated results:

```
report validated data -pass
```

- Report all CDC paths with failed validated results:

```
report validated data -fail
```

- Report CDC paths that have passed R1, R2, or both:

```
report validated data -pass -sync_rule R1 R2
```

- Report all CDC paths that failed at least one of their convergence checks:

```
report validated data -fail -convergent
```

- Report all CDC paths that have at least one of the given sync rules, and where that sync rule requires a convergence check:

```
report validated data -sync_rule R1 R2 -convergent
```

- Report all CDC paths that passed all of the convergence checks required by the given sync rules:

```
report validated data -pass -sync_rule sr1 sr0 -convergent
```

Related Commands

ADD CDC CHECK

ADD SYNCHRONIZATION RULE

SET CDC OPTION

SET PREDEFINED SYNCH_RULE

Conformal Extended Checks Reference Manual

Command Reference

VALIDATE

REPORT VECTOR DEFINITION

REPort VECtor Definition

```
<vectorname*>  
[-SUMmary | -VERbose]  
(Verify Mode)
```

Reports the definition of the matching vectors defined by the `ADD VECTOR DEFINITION` command.

Tcl Command

```
report_vector_definition
```

Parameters

<code><vectorname*></code>	Specifies the name of the vector(s) to report the definition. This accepts wildcards.
<code>-SUMmary</code>	Reports with the "Summary" format. This format lists the number of matching vectors. <i>This is the default.</i>
<code>-VERbose</code>	Reports with the "Verbose" format. This format includes the list of gates that belong to the vector.

Related Commands

- ADD VECTOR DEFINITION
- DELETE VECTOR DEFINITION

RESET

RESET

(Setup / Verify Mode)

Resets the system to the initial state. All existing designs and libraries are deleted, and all previously issued commands are cancelled.

Tcl Command

reset

Related Command

EXIT

RESET CLP

RESET CLP
(*Setup Mode*)

Resets all low power specific information except the library and design information. More specifically, this command resets the low power cell definitions and rule violations.

Note: Prior to running this command, if additional logic was inserted by another command, such as `COMMIT CPF -insert`, the inserted logic remains in the design. To undo the logic insertion, you must reset the entire library and design space and rerun the dofile.

Tcl Command

`reset_clp`

RUN LIBRARY CHECK

RUN Library Check
[-USED]
(Setup Mode)

Note: This is a Conformal Low Power command and a native 1801 feature.

Performs 1801 library checkers. This command can be run without reading the design.

Tcl Command

run_library_check

Parameters

-USED	Applies library consistency checking to the cells used in the design instead of checking every library cell.
-------	--

Example

The following performs 1801_LIB checkers at all read-in library files. Not require a design read-in.

```
TCL_SETUP > set_lowpower_option -native_1801
TCL_SETUP > read_library -liberty -lp <liberty files>
TCL_SETUP > run_library_check
```

The following performs 1801_LIB checkers at the library cells used at the design.

```
TCL_SETUP > set_lowpower_option -native_1801
TCL_SETUP > read_library -liberty -lp <liberty files>
TCL_SETUP > read_design <design>
TCL_SETUP > run_library_check -used
```

SAVE DOFILE

SAVe DOfile
<filename>
[-Replace]
(Setup / Verify Mode)

Saves the commands run during this session to a file. Use the saved dofile later as a batch file to repeat the Conformal Extended Checks session.

Note: The `SAVE DOFILE` command saves only commands that are issued using the command line. If you are using this command from within a dofile and there are no commands in the history buffer (in other words, no commands were issued using the command line), the tool will not create a dofile.

Tcl Command

save_dofile

Parameters

<filename>	Writes the dofile to the specified file. If the filename you specify already exists, you must use either the <code>-replace</code> or <code>-append</code> option. If you do not include an option, Conformal generates the following error message: Error: File 'filename' exists.
-Replace	Replaces the contents of the specified file with the commands run during the current session.

Related Commands

DOFILE

SET COMMAND PROFILE

SET LOG FILE

SEARCH

SEArch

`[-USage] <string1> [<string2 ...>]`
(*Setup / Verify Mode*)

Searches the database of commands and options and displays those commands that match all of the specified strings. Strings can be specified in any order; however, every specified string must match.

Tcl Command

search

Parameters

<code>-USage</code>	Displays the commands that have parameters that match the search string. This outputs the entire command syntax for each command.
<code><string1></code>	Displays commands that match the specified string.
<code><string2 ...></code>	Displays commands that match additional specified strings.

Related Command

[HELP](#)

SET_ATTR INPUT_PRAGMA_KEYWORD

SET_ATTR INPUT_PRAGMA_KEYWORD

<string>
(Setup Mode)

Specifies a keyword that the Conformal software must consider as an input pragma when it encounters it as the first word in a Verilog or VHDL source comment.

A pragma is a comment in the Verilog or VHDL source and is set off from ordinary comments by the pragma keyword. The pragma keyword is the first word listed in a pragma, and it notifies the Conformal software that the remainder of the comment is a command and not a comment. Changing this keyword allows you to set up compatibility with other tools

Tcl Command

```
set_attr_input_pragma_keyword
```

Parameters

<string>	Specifies the name of the keyword for a tool vendor. <i>Default:</i> pragma, synthesis, synopsys, cadence, ambit, verplex
----------	--

Examples

Sample Dofile:

```
set_attr_input_pragma_keyword rtl
set_synthesis_off_command turn_off
set_synthesis_on_command turn_on
```

After running these three commands, the Conformal and VHDL parsers will recognize the pragmas in the VHDL and Verilog Source files.

In a VHDL file, the code between `-- rtl turn_off` and `-- rtl turn_on` will not be synthesized.

In a Verilog file, the code between `// rtl turn_off` and `// rtl turn_on` will not be synthesized.

Related Commands

SET SYNTHESIS_OFF_COMMAND

SET SYNTHESIS_ON_COMMAND

SET CASE SENSITIVITY

SET CAse Sensitivity

<OFF | ON>

(Setup Mode)

Specifies whether any names you specify are case sensitive. The system default is no case sensitivity.

Run this command before `READ LIBRARY` and `READ DESIGN`. Use the `REPORT ENVIRONMENT` command to display the current setting for case sensitivity.

Tcl Command

```
set_case_sensitivity
```

Parameters

ON	Specifies that names are case sensitive.
OFF	Specifies that names are not case sensitive. <i>This handling is the system default.</i>

Related Command

`REPORT ENVIRONMENT`

SET CDC OPTION

SET Cdc Option

```
<
[-NODOMAIN_BY_SEL | -DOMAIN_BY_SEL]
[-NOMUX_DATA_SRC_DEST | -MUX_DATA_SRC_DEST]
[-NOMUX_SEL_SRC_DEST | -MUX_SEL_SRC_DEST]
[-NOBIDIR_IO_OPT | -BIDIR_IO_OPT]
[-NOCONVERGENCE_CHECK | -CONVERGENCE_CHECK]
[-NOUNIV_DOMAIN_THROUGH_SEQELM | -UNIV_DOMAIN_THROUGH_SEQELM]
[-DERIVE_PI_DOMAIN | -NODERIVE_PI_DOMAIN]
[-NOSETRESET_SOURCE_CHECK | -SETRESET_SOURCE_CHECK]
[ NOSET_SOURCE_CHECK | -SET_SOURCE_CHECK]
[-NORESET_SOURCE_CHECK | -RESET_SOURCE_CHECK]
[-NOFIFO_DETECT | -FIFO_DETECT]
[-NOSAME_DOMAIN_MULTI_DEST | -SAME_DOMAIN_MULTI_DEST]
[-TIED_DATA_MUX_OPT | -NOTIED_DATA_MUX_OPT]
[-NOPRINT_FROM_INSTANCE | -PRINT_FROM_INSTANCE]
> ...
(Setup / Verify Mode)
```

Defines how Conformal Extended Checks treats certain CDC structures.

Tcl Command

set_cdc_option

Parameters

-NODOMAIN_BY_SEL	The domain of the multiplexer output takes on the domain of the multiplexer inputs. If the multiplexer inputs have different domains, the domain of the multiplexer output will be a "mixed" domain. <i>This is the default.</i>
-DOMAIN_BY_SEL	The domain of the multiplexer output takes on the domain of the select line.
-NOMUX_DATA_SRC_DEST	Does not allow mixed source/destination domain to appear at any data input of the multiplexer. <i>This is the default.</i> Note: If it is not allowed, each data input must be in a single domain and the valid domains are <i>universal</i> , <i>source</i> , or <i>destination</i> , (that is, it cannot be mixed.)

Conformal Extended Checks Reference Manual

Command Reference

-MUX_DATA_SRC_DEST	Allows mixed source/destination domain to appear at any data input of the multiplexer.
-NOMUX_SEL_SRC_DEST	Does not allow mux synchronizers to be driven by signals that are in the mix of source and destination clock domains. <i>This is the default.</i>
-MUX_SEL_SRC_DEST	Allows mux synchronizers to be driven by signals that are in the mux of source and destination clock domains.
-NOBIDIR_IO_OPT	Does not globally optimize away bidirectional I/O paths (that is, run CDC Checks on bidirectional I/O paths). <i>This is the default.</i>
-BIDIR_IO_OPT	Globally optimizes away bidirectional I/O paths (that is, do not run CDC Checks on bidirectional I/O paths).
-NOCONVERGENCE_CHECK	Turns off the convergence check. <i>This is the default.</i>
-CONVERGENCE_CHECK	Turns on the convergence check, which ensures that two or more correlated CDC signals are not used by other logic in such a way that the outcome of the logic depends on both.
-NOUNIV_DOMAIN_THROUGH_SEQELM	The universal domain will not be propagated from data port of a sequential element to its output. <i>This is the default.</i>
-UNIV_DOMAIN_THROUGH_SEQELM	The universal domain will be propagated from data port of a sequential element to its output.
-DERIVE_PI_DOMAIN	Derives the domain of PI or BBOX:OUT automatically if this domain drives data input of registers that are all in the same clock domain. <i>This is the default.</i>
-NODERIVE_PI_DOMAIN	Specifies the domain of PI or BBOX:OUT as unknown unless explicit data association is applied.
-NOSETRESET_SOURCE_CHECK	Disables the check for both set and reset synchronizers. <i>This is the default.</i>
-SETRESET_SOURCE_CHECK	Enables the check for both set and reset synchronizers.
-NOSET_SOURCE_CHECK	Disables the check for only set synchronizers. <i>This is the default.</i>

Conformal Extended Checks Reference Manual

Command Reference

-SET_SOURCE_CHECK	Enables the check for only set synchronizers.
-NORESET_SOURCE_CHECK	Disables the check for only reset synchronizers. <i>This is the default.</i>
-RESET_SOURCE_CHECK	Enables the check for only reset synchronizers.
-NOFIFO_DETECT	Does not enable automatic FIFO identification. <i>This is the default.</i>
-FIFO_DETECT	Enables the automatic FIFO identification. In validation, this will remove false data crossing errors and reconvergence errors.
-NOSAME_DOMAIN_MULTI_DEST	Specifies that the CDC path multiple fanout check will be flagged if the same from-instance can reach more than one to-instance that is not in the same destination clock domain. <i>This is the default.</i>
-SAME_DOMAIN_MULTI_DEST	Specifies that the CDC path multiple fanout check will be flagged if the same from-instance can reach more than one to-instance that is in the same destination clock domain.
-TIED_DATA_MUX_OPT	Specifies that if both the inputs of a MUX are tied to the same constant, and the select line is not tied to any constant, the MUX is considered to be of constant domain. As a result, the entire fan-in cone of the MUX will be removed from the CDC path. <i>This is the default.</i>
-NOTIED_DATA_MUX_OPT	Specifies that the domain of the MUX will be mixed. The fan-in cone of the MUX will appear in the CDC path.
-NOPRINT_FROM_INSTANCE	When reporting validation results for set or reset, this option disables printing of multiple fan-in gates or sequential elements that drive the set or reset port of the register. <i>This is the default.</i>
-PRINT_FROM_INSTANCE	When reporting validation results for set or reset, this option prints multiple fan-in gates or sequential elements that drive the set or reset port of the register.

Examples

The following commands show an example of automatic FIFO identification:

```
set cdc option -fifo_detect
set system mode verify
add data association read_data\[*\] -domain read_clk
add cdc check -structural -source -all -destination -all -from -all -to -all
validate
```

Related Commands

REPORT ENVIRONMENT

VALIDATE

SET CLOCK_DOMAIN PRIORITY

SET Clock_domain Priority

```
<-FILE <filename> | -RESET | <id | pathname> ... >  
(Verify Mode)
```

Enables Conformal Extended Checks to resolve the clock domain assignment for an output of a logic gate in cases where the two inputs have different known domains. Specify a prioritized list of defined CDC clocks either on the command line or by calling up a specified file.

- Priority specified by file

The file format is one-clock-per-line in descending priority order.

- Priority specified by command line

List arguments in descending priority order. If you do not specify all defined CDC clocks in the command, all unspecified CDC clocks will have the same priority as the lowest specified CDC clock.

Note: This command works in conjunction with the command `set clock_domain rule -conflict_error`.

Tcl Command

```
set_clock_domain_priority
```

Parameters

<code>-FILE <filename></code>	Sets clock domain error resolution priority according to the list in the specified file.
<code>-RESET</code>	Resets all clock domain priorities.
<code><id pathname> ...</code>	Sets clock domain error resolution priority in order of the specified defined CDC clock IDs or paths.

Note: ID numbers can differ from one version of Conformal to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

Related Commands

REPORT CLOCK_DOMAIN PRIORITY

SET CLOCK_DOMAIN RULE

SET CLOCK_DOMAIN RULE

SET Clock_domain Rule

```
[ -UNKNOWN | -Derived | -NODerived ]  
[ -EXACT_phase [ | -IGNORE_NEGEDGE ]  
  | -WIRE [ | -IGNORE_INVerter | -IGNORE_NEGEDGE ]  
  | -EITHER_phase  
  | -LOGIC_Phase [ -Conflict_error | -NOConflict_error ]  
  | -LOGIC_NOPhase [ -Conflict_error | -NOConflict_error ]  
 ]  
[ -NOEXtract | -EXtract [ BUffer | WIRE ] ]  
(Setup / Verify Mode)
```

Specifies how Conformal Extended Checks automatically determines clock domains in a design. A clock domain rule consists of two parts: forward domain propagation and backward domain extraction.

■ Forward Domain Propagation

Forward domain propagation has two parts: propagation through sequential elements and propagation through combinational logic gates.

□ Propagation Through Sequential Elements.

Use the options `-unknown`, `-derived`, or `-noderived`.

□ Propagation Through Combinational Logic Gates.

Use the options `-exact_phase`, `-wire`, `-either_phase`, `-logic_phase`, or `-logic_nophase`. (You can further specify `-logic_phase` and `-logic_nophase` with the `-conflict_error` and `-noconflict_error` options.)

■ Backward Domain Extraction

Use the options `-noextract` or `-extract`. (You can further specify `-extract` with the `buffer` or `wire` option.)

Tcl Command

`set_clock_domain_rule`

Parameters

-UNKNOWN	The output of the state element has an unknown domain, regardless of the domain of the clock inputs. <i>This is the default.</i>
-Derived	<p>The output of the state element has the same domain as the clock input, if all clock inputs have the same known domain.</p> <p>The output of the state element has an unknown domain if all clock inputs have unknown domains.</p> <p>The output of the state element has an error domain if any clock input has an error domain.</p>
-NODerived	<p>The output of the state element has a new known domain (different from its clock inputs) if all clock inputs have the same known domain or different known domains.</p> <p>The output of the state element has an unknown domain if all clock inputs have unknown domains.</p> <p>The output of the state element has an error domain if any clock input has an error domain.</p>
-EXACT_phase	<p>If all inputs have unknown domains, the output has an unknown domain. <i>This is the default.</i></p> <p>If none of the logic gate's inputs has an error domain and at least one input has a known domain, the output has a new known domain.</p> <p>Exceptions:</p> <p>The output of an inverter has a domain that represents the inverted phase of the input domain. ($D_i \rightarrow \text{inv} \rightarrow \sim D_i$)</p> <p>The output of a buffer has the same domain as the input domain. ($D_i \rightarrow \text{buf} \rightarrow D_i$).</p>
-IGNORE_NEGEDGE	Ignores the inversion coming from the negative edge.
-WIRE	If all inputs have unknown domains, the output has an unknown domain. If none of the logic gate's inputs has an error domain and at least one input has a known domain, the output has a new known domain.
-IGNORE_INVerter	Specifies that the domain at the output of every inverter is the same as that at its input.

Conformal Extended Checks Reference Manual

Command Reference

-IGNORE_NEGEDGE	Specifies that the domain at the output of the inverter, which was synthesized because of the negedge keyword, is the same as that at its input.
-EITHER_phase	<p>If all inputs have unknown domains, the output has an unknown domain.</p> <p>If none of the logic gate's inputs has an error domain and at least one input has a known domain, the output has a new known domain.</p> <p>Exceptions:</p> <p>The output of a buffer has the same domain as the input domain. (Di -> buf -> Di)</p> <p>The output of an inverter has the same domain as the input domain. (Di -> inv -> Di)</p>
-LOGIC_Phase	<p>Further specify this option with one of the following:</p> <p>-Conflict_error</p> <p>If all inputs have unknown domains, the output has an unknown domain.</p> <p>If any input has an error domain, the output has an error domain.</p> <p>If any two inputs have different known domains, the output has an error domain.</p> <p>If all inputs have the same known domain, with the exception of zero or more unknown domains, the output has the same known domain as its inputs.</p> <p>Exception:</p> <p>The output of an inverter has a domain that represents the inverted phase of the input domain. (Di -> inv -> ~Di)</p> <p><i>The option -conflict_error is the default when you use -logic_phase.</i></p>

Conformal Extended Checks Reference Manual

Command Reference

-NOConflict_error

If all inputs have unknown domains, the output has an unknown domain.

If any two inputs have different known domains, the output has a new known domain.

If all inputs have the same known domain, with the exception of zero or more unknown domains, the output has the same known domain as its inputs.

Exception:

The output of an inverter has a domain that represents the inverted phase of the input domain. ($D_i \rightarrow \text{inv} \rightarrow \sim D_i$)

Note: This option is the same as `-logic_phase -conflict_error` except that there is no error domain; that is, the output of two different known domains is a new known domain.

-LOGIC_NOPhase

Further specify this option with one of the following:

-Conflict_error

If all inputs have unknown domains, the output has an unknown domain.

If any input has an error domain, the output has an error domain.

If any two inputs have different known domains, the output has an error domain.

If all inputs have the same known domain, with the exception of zero or more unknown domains, the output has the same known domain as its inputs.

$D_i \rightarrow \text{inv} \rightarrow D_i$

`-conflict_error` *is the default when you use* `-logic_nophase`.

Conformal Extended Checks Reference Manual

Command Reference

`-NOConflict_error`

If all inputs have unknown domains, the output has an unknown domain.

If any two inputs have different known domains, the output has a new known domain.

If all inputs have the same known domain, with the exception of zero or more unknown domains, the output has the same known domain as its inputs.

Exception:

The output of an inverter has the same domain as the input domain.

(Di -> inv -> Di)

Note: This option is the same as `-logic_nophase -conflict_error` except that there is no error domain; that is, the output of two different known domains is a new known domain.

`-NOEXtract`

Does not automatically extract domains. *This is the default.*

`-EXtract`

Further specify this option with one of the following:

`BUffer`

Automatically extracts domains starting from the unknown domain clock ports of D flip-flops and D-latches, and then backward trace through buffers and inverters.

Recognize the gate that stops the backward extraction as an extracted clock domain.

buffer is the default when you use -extract.

`WIRe`

Automatically extracts domains so that all unknown domain clock ports of D flip-flops and D-latches are treated as extracted clock domains.

Related Commands

REPORT ENVIRONMENT

Conformal Extended Checks Reference Manual

Command Reference

SET CLOCK DOMAIN PRIORITY

SET COMMAND ECHO

SET Command Echo
<ON | OFF>
(Setup / Verify Mode)

Controls whether the tool echos (or repeats on the screen) the command typed.

Tcl Command

set_command_echo

Parameters

ON	Enables command echoing.
OFF	Disables command echoing.

Example

For example, we save the following in the dofile and call it mydofile1.do:

```
read design ./ccr871626/test.v -verilog -root sub0
set command echo off
read design ./ccr871626/test.v -verilog -root sub0 -replace
set command echo on
read design ./ccr871626/test.v -verilog -root sub0 -replace
```

When you execute the dofile, the tool will only echo two of the READ DESIGN commands. It does not echo the second READ DESIGN command, because it was typed after the SET COMMAND ECHO OFF command was set:

```
// Command: dofile r
// Command: read design ./ccr871626/test.v -verilog -root sub0
// Parsing file ./ccr871626/test.v ...
// design root module is set to 'sub0'
// Note: Read VERILOG design successfully
// Read design summary: Error: 0, Warning: 10, Note: 0
// Note: Use 'report rule check -category library_design -verbose' for details.
// Command: set command echo off
// Warning: Existing design has been deleted
// Parsing file ./ccr871626/test.v ...
// design root module is set to 'sub0'
// Note: Read VERILOG design successfully
// Read design summary: Error: 0, Warning: 10, Note: 0
```

Conformal Extended Checks Reference Manual

Command Reference

```
// Note: Use 'report rule check -category library_design -verbose' for details.  
// Command: read design ./ccr871626/test.v -verilog -root sub0 -replace  
// Warning: Existing design has been deleted  
// Parsing file ./ccr871626/test.v ...  
// design root module is set to 'sub0'  
// Note: Read VERILOG design successfully  
// Read design summary: Error: 0, Warning: 10, Note: 0  
// Note: Use 'report rule check -category library_design -verbose' for details.
```

SET COMMAND PROFILE

SET Command Profile

[OFF | ON]

(Setup / Verify Mode)

Starts or stops recording a profile of commands executed by Conformal Extended Checks. This command records the order in which commands are executed and the memory use. The profile includes commands used in the graphical user interface mode.

Use `REPORT COMMAND PROFILE` to view the profile.

Tcl Command

`set_command_profile`

Parameters

OFF	Stops tracking executed commands. <i>This option is the default.</i>
ON	Starts tracking executed commands.

Related Commands

`REPORT COMMAND PROFILE`

`SET LOG FILE`

SET DIRECTIVE

SET Directive

```
<ON | OFF>  
[[synthesis | <vendor_name>] <directives>]  
[-File <filename*>]  
(Setup Mode)
```

Specifies whether to enable or disable the effects of the specified synthesis directives when reading in a Verilog or VHDL file. If you run this command and do not specify any directives, this command enables (or disables) *all* the directive effects. *The system default enables all directives.* (Thus, if you want Conformal Extended Checks to enable all directives, no action is necessary on your part.)

Run this command before READ LIBRARY and READ DESIGN.

For each disabled directive used in the HDL source code Conformal Extended Checks responds as follows:

- If the directive is supported but disabled, Conformal Extended Checks returns a message stating the directive is disabled.
- If the directive is unsupported and disabled, Conformal Extended Checks returns a message stating that the directive is unsupported.

Tcl Command

set_directive

Parameters

ON	Enables the specified directives. <i>This handling is the initial system default.</i>
OFF	Disables the specified directives. If you do not specify directives, all directives are disabled.
synthesis	Enables (or disables) the specified Synplicity® synthesis directives.

Conformal Extended Checks Reference Manual

Command Reference

`<vendor_name>` Enables (or disables) the specified synthesis directives when they are used with the specified `<vendor_name>` prefix. Supported vendors are listed below:

- `verplex`
- `synopsys`
- `ambit`
- `quickturn`

Conformal Extended Checks Reference Manual

Command Reference

<directives>	<p>Enables (or disables) the specified synthesis directives. If you do not specify any directives, all directives are enabled (or disabled), accordingly. Supported directives are listed below:</p> <p>assertion_library black_box built_in clock_hold compile_off compile_on dc_script_begin dc_script_end divider enum full_case infer_latch mem_rowselect multi_port multiplier operand parallel_case pragma state_vector synthesis_off synthesis_on template translate_off translate_on</p>
-File <filename*>	<p>Enables (or disables) a list of directives that are specified in a RTL file. This accepts wildcards.</p>

Examples

```
set directive off
set directive off translate_off translate_on
set directive on parallel_case full_case
```

Enabling Directives

The Verplex, Synopsys, and Ambit directives are enabled by default. The Quickturn directives are disabled by default. To recognize the Quickturn directives, you must first turn on all the directives for Quickturn using the following command:

```
set directive on quickturn
```

Enabling One Directive

When you employ the `SET DIRECTIVE` command and you do not specify a directive, the command applies to all directives. In the following example, the objective is to enable only the `parallel_case` directive. To do so, first disable all directives, then enable the appropriate directive (`parallel_case`).

```
//disable all directives
set directive off

//enable parallel_case
set directive on parallel_case
```

Disabling All Directives for One Vendor

In the following example, the objective is to disable all Synopsys directives (`synopsys translate_off`, `synopsys translate_on`, `synopsys full_case`..).

```
//disable all synopsys directives
set directive off synopsys
```

Disabling Specified Directives for One Vendor

In the following example, the objective is to disable `synopsys translate_off` and `synopsys translate_on`. This command has no effect on `verplex translate_off` and `verplex translate_on`.

Conformal Extended Checks Reference Manual

Command Reference

```
//disable synopsys translate_off and synopsys translate_on  
set directive off synopsys translate_off translate_on
```

Enabling a List of Directives from an RTL File

In the following examples, we have 2 RTL files: `test.v` and `test1.v`.

- In the following command, the synthesis directive `parallel_case` is on (enabled) in file `test.v`:

```
set directive on parallel_case -file test.v
```

- In the following command, the synthesis directive `parallel_case` is on (enabled) in file `test.v` and `test1.v`:

```
set directive on parallel_case -file *.v
```

Related Commands

READ DESIGN

READ LIBRARY

SET DOFILE ABORT

SET Dofile Abort

<ON | OFF | Exit>
(Setup / Verify Mode)

Specifies how Conformal handles a dofile if an error message occurs.

Note: This does not apply to LIB_LINT* rule checks or low power rules reported by ANALYZE POWER DOMAIN and the READ POWER INTENT commands.

Tcl Command

set_dofile_abort

Parameters

ON	Terminates the dofile if an error message occurs. <i>This handling is the system default.</i>
OFF	Allows the dofile to continue even if an error message occurs.
Exit	Exits the session if an error message occurs.

Related Commands

BREAK

CONTINUE

DOFILE

SAVE DOFILE

SET DW DEFINITION

SET DW Definition

```
<-USER_FIRST [-BBOX] | -USER_ONLY | -BUILTIN_only | -DW_MULT_div>  
(Setup Mode)
```

Specifies the DesignWare (DW*) modules' definition that the Conformal software will use. By default, the software specifies using the user-defined DW* modules first.

If you are using the built-in directory (where the DW* files are located), you can set the path with the following command:

```
setenv DW_DEFINE <pathname>
```

By default, the Conformal software uses the following path:

```
<install_dir>/share/cfm/lec/library/verilog/dw
```

Tcl Command

```
set_dw_definition
```

Parameters

-USER_FIRST [-BBOX]	Specifies using the user-defined DW* modules first. Resolved referenced but undefined DW* modules use built-in modules. <i>This is the software default.</i> Use the -BBOX option to specify that when attempting to resolve blackboxed DW* modules, it will use built-in modules.
-USER_ONLY	Specifies using user-defined DW* modules only.
-BUILTIN_only	Specifies using built-in DW* modules only and skips user-defined DW* modules.
-DW_MULT_div	Specifies using user-defined DW* modules only except DW_MULT* and DW_DIV* modules.

Example

The following shows an example of the RTL code in a file `absval.v`.

Conformal Extended Checks Reference Manual

Command Reference

```
module sample( A, ABSVAL );  
    input [7 : 0] A;  
    output [7 : 0] ABSVAL;  
    DW01_absval #(width) U1 ( .A(A), .ABSVAL(ABSVAL) ); endmodule
```

The command to read in the design is:

```
read design absval.v
```

The `DW01_absval.v` file is automatically read in from the built-in directory because there is no definition provided.

SET FLATTEN MODEL

SET Flatten Model

```
<-PIO_to_bus [PIO | PO | TRI_state]
| [-NOLATCH_Fold | -LATCH_Fold]
| [-NONONControlling_remodel | -NONControlling_remodel]
| [-NOCLOCKed_system_constraint | -CLOCKed_system_constraint]
| [-NOGATED_Clock | -GATED_Clock]
| [-NOINStance_cst_propagation | -INStance_cst_propagation]
| [-NOUSE_INS_NAME_only | -USE_INS_NAME_only]
>
[-TIED_AFTER_INIT]
(Setup Mode)
```

Specifies certain conditions for the flattened model.

Note: When OVL assertions have an options parameter of 1, formal tools interpret them as system constraints. Use the `set flatten model -noclocked_system_constraint` or `-clocked_system_constraint` option to specify whether these constraints are synchronized with their respective driving clock signals. (Refer to Chapter 2 of the *Open Verification Library Assertion Monitor Reference Manual* for additional information about the *options* parameter.)

Also, when an assertion's *options* value is 0 (which is the default), you must use the ADD ASSERTION CONSTRAINTS command with the `-noclocked` or `-clocked` option (note the `SET FLATTEN MODEL` command) to specify whether assertion constraints are synchronous to their respective driving clock signals.

Tcl Command

```
set_flatten_model
```

Parameters

<code>-PIO_to_bus</code>	Specifies how bidirectional pins are treated in the flattened model of the design. Pins are one of the followings:
<code>PIO</code>	bidirectional I/Os. <i>This is the default.</i>
<code>PO</code>	primary outputs
<code>TRI_state</code>	tristates

Conformal Extended Checks Reference Manual

Command Reference

- `-NOLATCH_Fold` Does not fold a master-slave latch into a D flip-flop. *This is the default.*
- `-LATCH_Fold` Folds a master-slave latch into a D flip-flop.
- `-NONONControlling_remodel`
Retains circuit logic that is optimized away by external constraints. *This is the default.*
- `-NONControlling_remodel`
Removes circuit logic that is optimized away by external constraints.
- `-NOCLOCKed_system_constraint`
Indicates that constraints are asynchronous to their driving clocks. *This is the default.*
Note: No-clocked constraints are valid at all times.
- `-CLOCKed_system_constraint`
Indicates that constraints are synchronous with their driving clocks.
Note: Clocked constraints are valid on their respective rising clock edges.
- `-NOGATED_Clock` Does not remodel gated-clock sequential instances. *This is the default.*
- `-GATED_Clock` Remodels the gated-clock logic of the clock port of a DFF. If the clock pin cannot be automatically determined, use the `ADD CLOCK` command to define the clock pin.
- `-NOINStance_cst_propagation`
Does not propagate the constrained value to the combinational fan-out cone of the constrained instance. *This is the default.*
- `-INStance_cst_propagation`
Propagates the constrained value to the combinational fan-out cone of the constrained instance.
- `-NOUSE_INS_NAME_only`
Generates flatten gate names based on instance names or net names of instance output pins. *This is the default.*
- `-USE_INS_NAME_only`
Generates flatten gate names from instance names only.

Conformal Extended Checks Reference Manual

Command Reference

-TIED_AFTER_INIT

Note: When a register's data input is tied to 1/0, it will be treated as having a constant value 1/0, respectively.

Related Commands

ADD ASSERTION CONSTRAINTS

REPORT ENVIRONMENT

REPORT RULE CHECK

SET GUI

SET GUI

[ON | OFF]

(Setup / Verify Mode)

Specifies a change in operating modes. While in the non-GUI mode, you can type this command to move to the GUI mode. Conversely, while in the GUI mode, you can type this command to move to the non-GUI mode.

Tcl Command

set_gui

Parameters

ON	Switches to the GUI mode. <i>This is the default.</i>
OFF	Switches to the non-GUI mode.

SET HDL OPTIONS

SET HDL Options

```
[-AUTOPkgsearch <ON | OFF | PREFIX <string>| SUFFIX <string>>]
[-BBOX_MODULE_WITH_NO_PI_USED <OFF | ON>]
[-CONTINUOUSASSIGNment <BiDirectional | UNIdirectional>]
[-CONVERT_ONEBIT_VECTOR_TO_SCALAR <OFF | ON>]
[-COPY_VERILOG_TO_LIBERTY
    <USE_LIBERTY_FUNCTION | USE_VERILOG_ONLY_IF_LIBERTY_EMPTY |
    USE_VERILOG_FUNCTION_ALWAYS>]
[-DFF_ASYNC_HOLD <OFF | ON>]
[-ERROR_OUT_POSITIONAL_ASSOCIATION_ON_LIBERTY_CELL <OFF | ON>]
[-EXCLUDE <cell_name* ...>]
[-FIX_PIN_DIRECTION <OFF | ON>]
[-FULLPATH <OFF | ON>]
[-HIEREF_TO_PORT_CONN <OFF | ON>]
[-INCLUDE_SRC_DIR <ON | OFF>]
[-KEEP_OT_SV_UNION_NAME <OFF | ON>]
[-LEGACY_DC_SCRIPT <OFF | ON>]
[-LIBERTY_MAP_FILE <name>]
[-MAX_FOR_LOOP_SIZE <integer_value>]
[-MAX_PARSE_STACK <integer_value>]
[-MAXMEM_address_space <threshold>]
[-MERGE_PHYSICAL_cell <AUTO | OFF|USER | AUTO_USER>]
[-MULTILINERead <OFF | ON>]
[-NO_NEGATIVE_INDEX_PORT <OFF | ZEROBASED | NOOVERLAPPED>]
[-NOPARSE_FILE_CMD | -PARSE_FILE_CMD <command>]
[-OUT_LIBERTY_DIR <path>]
[-OUT_LIBERTY_FILE <name>]
[-PHYSICAL_CELL_EXTRACT <LIBERTY_ATTRIBUTE | ATTRIBUTE_AND_STRUCTURE>]
[-PORT_SIZE_limit <size>]
[-PORTMISmatch <UNconnect | EXTend>]
[-PRIMITIVE_INPUT_Conversion <LSB | LOGIC>]
[-READ_TRANSLATE_MSFF <ON | OFF>]
[-REMOVE_CELL <cellnames*>]
[-SV_2STATE_DASSIGN_CHK <OFF | ON>]
[-SYNTHESIS_executable <syn_exe>]
[-UDP_IGN_VDD <module_name> <vdd_pin_name>]
[-UNIQUE_IF_CONSTRAINTS | -NOUNIQUE_IF_CONSTRAINTS]
[-UNREACHABLE_STMT_CHECK <OFF | ON>]
[-UNSIGNED_CONVERSION_OVERFLOW <OFF | ON>]
[-UNSIGNED_EXPRESSION_OVERFLOW <OFF | ON>]
[-USE_LIBRARY_FIRST <OFF | ON>]
[-USER_DEFINED_PHYSICAL_CELL <cell_name*>]
[-VARSIZE_limit <integer_value>]
[-VERILOG_INCLUDE_DIR <string>]
[-VERILOG_OUTOFBOUNdRead <PARTIAL_X | ALL_X | PARTIAL_0 | PARTIAL_1>]
[-VERILOG_OUTOFBOUNdWrite <Noeffect | X>]
[-VERILOG_TRIMINdex <OFF | ON>]
[-VERILOG_XL_LIBORDER <OFF | ON>]
```


Conformal Extended Checks Reference Manual

Command Reference

```
[-VHDL_OUTOFBOUNDWrite <X | Noeffect>]
[-VHDL_OUTOFBOUNDRead <ALL_X | PARTIAL_X | PARTIAL_0 | PARTIAL_1>]
[-VHDL_TRIMINDEX <OFF | ON>]
[-WAND_GATE [FALSE | TRUE]]
[-WOR_GATE [FALSE | TRUE]]
[-ZERO_REPLICATE_AS_ZERO <ON | OFF>]
(Setup Mode)
```

Controls the interpretation of some RTL semantics.

This command sets global attributes that control the behavior of subsequent READ LIBRARY, READ DESIGN, and ELABORATE DESIGN commands. Libraries and designs that have been successfully read in and elaborated previously will not be affected by newly issued SET HDL OPTIONS commands.

Tip: Use SET HDL OPTIONS without any options to display the current settings for SET HDL OPTIONS.

Tcl Command

set_hdl_options

Parameters

```
-AUTOPkgsearch <ON | OFF | PREFIX <string> | SUFFIX <string>>
```

Conformal Extended Checks Reference Manual

Command Reference

ON— Searches the work directory for referred packages or entities. *This is the default.*

OFF—Disables the automatic searching.

PREFIX *<string>*—Specifies that you want to search for packages/entities with the specified prefix. Separate multiple values with a comma (spaces are ignored).

SUFFIX *<string>*—Specifies that you want to search for packages/entities with the specified suffix. Separate multiple values with a comma (spaces are ignored).

When a prefix or suffix is not specified, the VHDL parser searches the working directory for `name.vhdl / name.vhd`.

When a prefix or suffix is specified, the tool first searches for the packages with the specified prefix or suffix. If there are no results, the tool resumes default searching. If there are multiple prefix/suffix values available, the tool searches in the order specified in the string.

Note: You cannot specify both a prefix and a suffix.

-BBOX_MODULE_WITH_NO_PI_USED

ON specifies that Verilog modules or VHDL entities will be treated as a blackbox if none of the inputs are used in the module entity. This option is OFF by default.

-CONTINUOUSASSIGNment <BiDirectional | UNIdirectional>

Specifies that continuous assignment in the design should be interpreted as bi-directional or uni-directional assignment. *The default is bidirectional.*

-CONVERT_ONEBIT_VECTOR_TO_SCALAR <OFF | ON>

Default is OFF. When set to ON, this option tells Conformal to treat all single-bit vectors as scalar objects. For example, both `dd` and `qq` in following module `t1` declaration are single-bit vectors. With this option, they are treated as scalar objects:

```
module t1(clk, dd, qq);
input [0:0] dd;          // changes to "input dd;"
input clk;
output reg [1:1] qq; // changes to "output reg qq;"
always @(posedge clk) begin
qq <= dd;
end
endmodule
```

-COPY_VERILOG_TO_LIBERTY Specifies the source from which the cell's logic function is derived.

USE_LIBERTY_FUNCTION— Tool always takes the logic function from the Liberty library model. *This is the default.*

USE_VERILOG_ONLY_IF_LIBERTY_EMPTY—If the cell's Liberty library model is not available, the tool looks for logic information in the Verilog model.

USE_VERILOG_FUNCTION_ALWAYS—Tool uses function information from the Verilog model, regardless of whether the Liberty model is available. However, PG pins (power/ground/bias) that are defined in Liberty but not in Verilog will be automatically added to stay consistent with the Liberty definition. To enable this feature, users need to read the liberty library into the library space first, then read the verilog library into library space with the `-append` option. For example:

```
set hdl option -copy_verilog_to_liberty \
user_verilog_function_always
read library -liberty std_lib.lib
read library -verily std_lib.v -append
```

If users read the verilog library into the design space, the logic function information of the verilog library will not be merged with the liberty library.

Conformal Extended Checks Reference Manual

Command Reference

- `-DFF_ASYNC_HOLD` Controls how an asynchronous conditional self-assignment is implemented for an RTL design.
- By default, this option is OFF and asynchronous pins are used to implement the gate-level design.
- In some cases, a synthesizer may use asynchronous pins for implementation instead and cause RTL and gate-level designs non-equivalent. Setting this option ON may help equivalence checking for such cases. This setting applies to the entire RTL parsing.
- `-ERROR_OUT_POSITIONAL_ASSOCIATION_ON_LIBERTY_CELL <OFF | ON>`
- OFF—*This is the default.*
- ON—When specified, the tool checks whether the design includes one or more module instances that uses port positional association of Liberty cells (rule check HRC3.8a).
- `-EXCLUDE <cell_name* ...>` Specifies the list of cell names to be excluded in the `copy_verilog_to_liberty` flow. The `-EXCLUDE` option can only be used with the `-copy_verilog_to_liberty` command option.
- `-FIX_PIN_DIRECTION <OFF | ON>`
- This changes the pin direction to the bi-directional (inout) type when an output port of a module is not driven by any logic inside the module, or when a input port of a module is driven by some logic inside the module.
- `-FULLPATH <OFF | ON>`
- Records the file name's absolute path name; when files are read in, the file source is stored in its absolute path (opposed to a relative path or a path added by the ADD SEARCH PATH command). Default is OFF.
- `-HIEREF_TO_PORT_CONN <OFF | ON>`
- When set to ON, LEC will create port(s) for hierarchical variable reference's corresponding instance module(s) and replace the hierarchical variable reference with an instance port connection plus internal net connection. Default is OFF.
- `-INCLUDE_SRC_DIR <ON | OFF>`

Conformal Extended Checks Reference Manual

Command Reference

For Verilog ``include` directives, LEC searches and reads the file in source file's relative directory before any paths added by ADD SEARCH PATH. Default is ON.

`-KEEP_OT_SV_UNION_NAME <OFF | ON>`

Determines whether to keep the sv union name in structure view when users specify the naming style with `SET NAMING STYLE DC`. For example,

```
typedef struct packed {  
    logic [3:0] amask;  
    union packed {  
        struct packed {  
            logic [1:0] [63:0] d;  
        } harv;  
        struct packed {  
            logic [15:0] [7:0] d;  
        } dude;  
    } bus;  
} mydata_t;
```

The output netlist naming style:

With option off
`DATA_reg[bus][0]`

With option on
`DATA_reg[bus][harv][d][0][0]`

OFF: Does not keep the sv union name in structure view.
This is the default.

ON: Keeps the sv union name in structure view.

`-LEGACY_DC_SCRIPT <OFF | ON>`

Controls the contents of the non-synthesizable DW script.

OFF — Includes verification priority setting. Does not include ungroup option setting. *This is the default.*

ON — Includes ungroup option setting. Does not include verification priority setting.

Conformal Extended Checks Reference Manual

Command Reference

`-LIBERTY_MAP_FILE <name>` Specifies the name and location of the file that contains the information of the original Liberty file name and the new simplified file name. If the option is not specified, no mapping file will be generated. If the option is specified without a value, the tool will report that this option is not fully specified.

For more information on this option, refer to the web interface article titled *Liberty File Optimization for Verification*.

`-MAX_FOR_LOOP_SIZE` Sets the maximum number of iterations when unfolding a loop construct. By default, the maximum is 8192 iterations.

`-MAX_PARSE_STACK <integer_value>`

Specifies the maximum parser stack size for a variable. By default, the maximum parser stack size is 100000.

For example, to increase the maximum parser stack size:

```
set hdl option -MAX_PARSE_STACK 200000
```

`-MAXMEM_address_space <threshold>`

Specifies the threshold of the maximum memory-address space. Modules whose RAM size is greater than or equal to the threshold will be blackboxed.

The default threshold value is 65536.

For example, to specify a maximum memory address space of 1024:

```
set hdl option -MAXMEM_address_space 1024
```

`-MERGE_PHYSical_cell <AUTO | OFF | USER | AUTO_USER>`

Conformal Extended Checks Reference Manual

Command Reference

Controls physical cell instance merging behavior.

AUTO—Performs instance merging on physical cells that are recognized by Conformal. *This is the default.*

OFF—Does not perform physical cell instances merging.

USER—Performs instance merging on user-specified cells only.

AUTO_USER—Performs instance merging on user-specified and tool recognized physical cells.

-MULTILINERead <OFF | ON> Enables support for multi-lined **READ DESIGN** commands. Default is **OFF**.

For example, when this option is enabled, the following is supported:

```
read design -vhdl -mapfile lib1 ent.vhdl -
noelaborate
read design -vhdl -mapfile lib1 arch.vhdl -
noelaborate
```

Note: Does not place other commands between the **READ DESIGN** commands.

-NO_NEGATIVE_INDEX_PORT <OFF | ZEROBASED | NOOVERLAPPED>

Changes the index data for negative ports.

OFF—*This is the default.* Does not change the index data.

ZEROBASED—For an array index of [-9:-1] or [-4:4], shifts the negative range to a positive range that starts at zero.

NOOVERLAPPED—For an array index of [-9:-1] (where both MSB and LSB are less than zero), shifts the negative range to a positive range that starts at zero. For an array index of [-4:4] (where MSB is less than zero, and LSB is greater than zero, or MSB is greater than zero and LSB is less than zero), changes the range by shifting the negative index to a positive index +1, and increases the positive index with size data.

-NOPARSE_FILE_CMD

Does not run any shell commands before parsing the file. *This is the default.*

Conformal Extended Checks Reference Manual

Command Reference

`-OUT_LIBERTY_DIR <path>` Specifies the directory where the simplified Liberty files will be placed. If the specified directory does not exist, it will be created. If this option is not specified, no action will be taken. If the option is specified without a value, the default output directory is the current working directory.

Default is the working directory.

For more information on this option, refer to the web interface article titled *Liberty File Optimization for Verification*.

`-OUT_LIBERTY_FILE <name>` Specifies the prefix for the filenames. Existing files will be replaced. Each file will be assigned a name as `<prefix>_number` where number starts at 00001, incremented by 1 for each new file up to 99,999. The library name is also modified and will have the same value as the new file name. If this option is not specified, the original filename will be used. If the option is specified without a value, the tool will report that this option is not fully specified.

For more information on this option, refer to the web interface article titled *Liberty File Optimization for Verification*.

`-PHYSICAL_CELL_EXTRACT <LIBERTY_ATTRIBUTE |
ATTRIBUTE_AND_STRUCTURE>`

Controls the physical cell recognitions.

LIBERTY_ATTRIBUTE—Extracts the cell defined by Liberty physical cell attributes. *This is the default.*

ATTRIBUTE_AND_STRUCTURE—Extracts the cell defined by Liberty physical cell attributes or by Conformal physical cell structure rules.

`-PORT_SIZE_limit <size>` Specifies the maximum size for a port. By default, the maximum port size is 524288.

For example, to increase the maximum port size:

```
set hdl option -PORT_SIZE_limit 1048576
```

`-PARSE_FILE_CMD <command>`

Conformal Extended Checks Reference Manual

Command Reference

Runs a shell command or executable file after printing out the "Parsing file ..." message and before Conformal parses the file during the `READ LIBRARY` or `READ DESIGN` commands.

Each current parsing file name will be stored in an environment variable `CFM_READ_FILE` before invoking the command.

`-PORTMISmatch <UNconnect | EXTend>`

When the port connection widths between the module and an instantiation do not match, this option controls how the tool models the extra bits.

`UNconnect` leaves the extra bits unconnected. *This is the default.*

`EXTend` applies zero- or sign-extending to the value, depending on the expression signedness. See Examples section.

`-PRIMITIVE_INPUT_Conversion <LSB | LOGIC>`

Specifies how the software's Verilog parser handles multiple-bit expressions.

`LSB` takes the least significant bit from the multiple-bit vector expression. *This is the default for the Verilog parser.*

`LOGIC` treats the entire multiple-bit expression as a logic true/false value. For example, for

```
wire [0:5] net0;  
or I00 (out, net0[0:5]);
```

`net0[0:5]` used in `I00` is treated as `net0[5]` by default, and it is treated as a logic true/false value when using the `LOGIC` option.

Conformal Extended Checks Reference Manual

Command Reference

`-READ_TRANSLATE_MSFF <ON | OFF>`

Models master-slave DFF of Liberty flip-flop cells that have `clock_on` and `clock_on_also` attributes.

ON translates master-slave flip-flops (MSFF) to master slave latches. With this option, two D latches are used to model the master-slave DFF. *This is the command default.*

OFF keeps the master slave flip-flops. With this option, two flip-flops are used to model the master-slave DFF.

`-REMOVE_CELL <cellnames*>`

Specifies a list of library cell instantiations to be excluded from the design. This option should be specified before the `READ DESIGN` command. Wildcards are accepted.

`-SV_2STATE_DASSIGN_CHK <OFF | ON>`

Enables rule checking on direct assignment of a 4-state variable or function to a 2-state variable (see the description of [RTL7.32f](#) rule for details).

The default is OFF.

`-SYNTHESIS_executable <syn_exe>`

Specifies the path to the Design Compiler synthesis executable. The `WRITE BLACKBOX WRAPPER - auto_substitute` command needs this to perform automatic blackbox substitution.

For example:

```
set hdl option \  
  -synthesis_executable \  
  /grid/DC_INSTALL_DIR/ \  
  latest/bin/dc_shell-t
```

`-UDP_IGN_VDD <module_name> <vdd_pin_name>`

Conformal Extended Checks Reference Manual

Command Reference

In UDP modeling, power/ground ports are not supported; therefore, having this information in your UDP state table can cause unexpected results.

Use this option to specify the module and pin names of power/ground ports that are defined in the UDP. The tool will ignore the information for these ports in the UDP state table.

-UNIQUE_IF_CONSTRAINTS When the logic 'unique-if' is valid but the final "else" is not defined, generate constraints or assertions for cases that cannot occur in 'unique if-else'. *This is the default.*

-NOUNIQUE_IF_CONSTRAINTS When the logic 'unique-if' is valid but the final "else" is not defined, generate "don't-care" assignment for the final "else". For example, if the 'unique-if' constraint fails, the LEC netlist will treat the output as 'don't care' ('x').

-UNREACHABLE_STMT_CHECK <OFF | ON>

When set to ON, the tool checks whether the design statement is unreachable. If it is, RTL checking will be skipped for that statement. Note that enabling this option will increase elaboration time.

Default is OFF.

-UNSIGNED_CONVERSION_OVERFLOW <OFF | ON>

OFF—*This is the default.*

ON—When specified, the tool will take up to 32 bits of UNSIGNED operands to signed 32-bit integer.

See example below.

-UNSIGNED_EXPRESSION_OVERFLOW <OFF | ON>

OFF—*This is the default.*

ON—When specified, unsigned operands of up to 32 bits can overflow (or wrap) to signed 32-bit integers.

See example below.

Conformal Extended Checks Reference Manual

Command Reference

- `-USE_LIBRARY_FIRST` OFF—Design modules have highest priority during synthesis. *This is the default.* At lowpower native-1801 flow, a module extracted from Liberty as a macro cell still has the highest priority even the OFF value is set.
- ON—Specifies that library cells have highest priority when being selected during synthesis.
- `-USER_DEFINED_PHYSICAL_CELL <cell_name*>`
- Specifies the list of user-defined physical cell names.
- `-VARSIZE_limit <integer_value>`
- Specifies the maximum size for a variable. By default, the maximum variable size is 4194304.
- For example, to increase the maximum variable size:
- ```
set hdl option -VARSIZE_limit 134217728
```
- `-VERILOG_INCLUDE_DIR <string>`

Specifies the order for searching paths when opening a Verilog ``include` file. The argument is a string comprised of the following keywords in quotes and separated by a colon `":"`.

- `cwd`: current working directory `"."`
- `incdir`: directories specified by `+incdir` option in the verilog command file
- `src`: directory that contains the file that specifies the ``include` directive
- `sep`: Search paths added by the `ADD SEArch Path` command
- `yd`: directories specified by `-yd` in the Verilog command file
- `y`: directories specified by `-y` in the Verilog command file

When `-INCLUDE_SRC_DIR` is `OFF`, the `src` argument will not have any effect.

Default string is `cwd:incdir:src:sep:yd:y`.

For example, if you want to search for include files only in current working directory and all directories specified by the `ADD SEARCH PATH` command, use the command as follows:

```
set hdl option -verilog_include_dir "cwd:sep"
```

Other directories added by `+incdir`, `-yd`, or the `-y` Verilog command file option are all ignored.

## Conformal Extended Checks Reference Manual

### Command Reference

---

`-VERILOG_OUTOFBOUNDRead <PARTIAL_X | ALL_X | PARTIAL_0 | PARTIAL_1>`

Controls the interpretation of Verilog bit (or part)-select of vector typed variable/signal when index is out of the defined index range.

`PARTIAL_X` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have 'x' value from the invalid (i.e. out of bound) locations of the vector. *This is the default.*

`ALL_X` specifies that when there is out-of-bound reading, the selected portion of the vector will be treated as all 'x' values.

`PARTIAL_0` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have '0' value from the invalid (i.e. out of bound) locations of the vector.

`PARTIAL_1` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have '1' value from the invalid (i.e. out of bound) locations of the vector.

`-VERILOG_OUTOFBOUNDWrite <Noeffect | X>`

Controls the interpretation of Verilog bit (or part)-select of vector typed variable/signal when index is out of the defined index range.

`Noeffect` specifies that out-of-bound writing will have no effect. *This is the default.*

`X` specifies that when there is out-of-bound writing, the related part of the variable/signal is assigned value 'x'.

`-VERILOG_XL_LIBORDER <OFF | ON>`

When set to ON, Conformal will follow the same search order as the Cadence NC-Verilog handling of the -y/-v command file options. Default is OFF, and this will keep the legacy behavior of Conformal Verilog parsing.

## Conformal Extended Checks Reference Manual

### Command Reference

---

`-VHDL_OUTOFBOUNDWrite <X | Noeffect>`

Controls the interpretation of VHDL bit (or part)-select of vector typed variable/signal when index is out of the defined index range.

`X` specifies that when there is out-of-bound writing, the related part of the variable/signal is assigned value 'x'. *This is the default.*

`Noeffect` specifies that out-of-bound writing will have no effect.

`-VERILOG_TRIMINdex <OFF | ON>`

`ON` controls to trim the index to necessary bits for the Verilog files. `OFF` is the command default.

**Note:** This option might be used to verify implementations in which indexes are intentionally trimmed.

`-VHDL_OUTOFBOUNDRead <ALL_X | PARTIAL_X | PARTIAL_0 | PARTIAL_1>`

Controls the interpretation of VHDL bit (or part)-select of vector typed variable/signal when index is out of the defined index range.

`ALL_X` specifies that when there is out-of-bound reading, the selected portion of the vector will be treated as all 'x' values. *This is the default.*

`PARTIAL_X` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have 'x' value from the invalid (i.e. out of bound) locations of the vector.

`PARTIAL_0` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have '0' value from the invalid (i.e. out of bound) locations of the vector.

`PARTIAL_1` specifies that out-of-bound reading will have values from the valid index locations of the vector, and will have '1' value from the invalid (i.e. out of bound) locations of the vector.

`-VHDL_TRIMINdex <OFF | ON>`

## Conformal Extended Checks Reference Manual

### Command Reference

---

ON controls to trim the index to necessary bits for the VHDL files. OFF is the command default.

**Note:** This option might be used to verify implementations in which indexes are intentionally trimmed.

`-WAND_GATE [FALSE | TRUE]` FALSE specifies that the direction of the WAND gate will be controlled by the `READ DESIGN - continuousassignment` command. (This option controls the direction of all WIRE/WOR/WAND continuous assignments. By default, WAND gates are bidirectional).

TRUE specifies that the WAND gate is always unidirectional, regardless of what is specified by `READ DESIGN -continuousassignment`.

Default is FALSE.

`-WOR_GATE [FALSE | TRUE]` FALSE specifies that the direction of the WOR gate will be controlled by the `READ DESIGN - continuousassignment` command. (This option controls the direction of all WIRE/WOR/WAND continuous assignments. By default, WOR gates are bidirectional).

TRUE specifies that the WOR gate is always unidirectional, regardless of what is specified by `READ DESIGN -continuousassignment`.

Default is FALSE.

`-ZERO_REPLICATE_AS_ZERO <ON | OFF>`

When set to ON, treats zero replication as 1'b0 for Verilog designs.

When set to OFF, treats zero replication as NULL in concatenation.

*OFF is the default.*



## Examples

The following examples use the Verilog language to show how to control index out of bound handling. You can use similar VHDL command options to control the interpretations for the VHDL language designs.

In the following RTL-1 example, the index range of variable `mem` is 0 to 2.

### RTL-1

```
wire a;
reg [2:0] index;
reg [2:0] mem;
always @(*) mem[index] = a;
```

If `index` is greater than 2, it is out of the index range. Some synthesis tools might intentionally interpret the RTL the same as with the following RTL-2 example:

### RTL-2

```
wire a;
reg[2:0] index;
reg[2:0] mem;
always @(*) mem[index[1:0]] = a;
```

But with simulation, RTL-1 and RTL-2 behave differently.

With the Conformal software, when running the command:

```
set hdl options -verilog_outofboundwrite x
```

then

- `index=0 : mem[0]` is assigned to the value of `a`
- `index=1 : mem[1]` is assigned to the value of `a`
- `index=2 : mem[2]` is assigned to the value of `a`
- `index=3, 4, 5, 6, 7 : mem[0], mem[1], and mem[2]` are assigned to the value of `1'bX`.

This interpretation assumes that out-of-bound writing will not happen, and consequently ignores the behavior difference when `index` is greater than 2.

## Conformal Extended Checks Reference Manual

### Command Reference

---

When running the command:

```
set hdl options -verilog_outofboundwrite noeffect
```

then

- `index=0 : mem[0]` is assigned to the value of `a`
- `index=1 : mem[1]` is assigned to the value of `a`
- `index=2 : mem[2]` is assigned to the value of `a`
- `index=3, 4, 5, 6, 7 : mem[0], mem[1], and mem[2]` will not be affected with their current value.

Based on this interpretation, RTL-1 and RTL-2 are considered functional non-equivalent, and consequently the implementation from RTL-2 will be non-equivalent to RTL-1.

Using the same RTL-1 and RTL-2 examples, when running the command:

```
set hdl options -verilog_trimindex on
```

The Conformal software will interpret RTL-1 as RTL-2 by ignoring `index[2]` in the expression `mem[index]` (RTL-1). With the `-verilog_trimindex on` option, RTL-1 and RTL-2 are considered equivalent.

**Port Mismatch Example:**

The following example illustrates when the port connection widths between the module and an instantiation do not match:

```
module top (output [3:0] o, input [2:0] in,in1);
 sub s1 (.o(o[3:0]), .in(in));
endmodule

module sub (output [2:0] o,input [2:0] in);
 assign o = in;
endmodule
```

By default, the tool leaves `top.o[3]` unconnected. If you use the `set hdl options -portmismatch extend` command, the tool applies sign extending and connects `top.o[3]` to `1'b0`.

### Unsigned Conversion Overflow Example

For example:

```
oo: out integer;
constant c1: unsigned(31 downto 0) \
:= "11000000000000000000000000000000";
oo <= conv_integer(c1);
```

When `-unsigned_conversion_overflow OFF` is specified, `oo[31]` is 1'b0. With this option ON, `oo[31]` is 1'b1.

### Unsigned Expression Overflow Example

For example,

```
in1: in natural; oo: out integer;
oo <= in1 * 2;
```

With `-unsigned_expression_overflow OFF` is specified, `oo[31]` is 1'b0. With this option ON, `oo[31]` is `in1[30]`.

### Including Source Directory Example

For example, given the following directory structure:

```
common/define.vh
rtl/define.vh
rtl/test.v
```

Where `rtl/test.v` contains:

```
`include "define.vh"
```

The `"READ DESIGN rtl/test.v"` command reads in `rtl/test.v` and `rtl/define.vh` by default.

When you use the `"SET HDL OPTIONS -include_src_dir off"` command, search paths are manually specified using the `ADD SEARCH PATH` or the `+incdir+` option in the Verilog command files. For example:

```
SETUP> SET HDL OPTIONS -INCLUDE_SRC_DIR OFF
SETUP> ADD SEARCH PATH common
SETUP> READ DESIGN rtl/test.v
```

The tool now reads in `rtl/test.v` and `common/define.vh`.

### Remove Cell Example

```
// design.v
module design(...);
 LIB_CELL1 u0 (...);
 LIB_CELL1 u1 (...);
 LIB_CELL2 u2 (...);
 LIB_CELL3 u3 (...);
endmodule

// Conformal commands
read library -liberty LIB_CELLS.lib // read in LIB_CELL1, LIB_CELL2, LIB_CELL3
set hdl options -remove_cell lib_cell1
read design design.v
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

The command `set_hdl_options -remove_cell lib_cell11` will remove the cell instances `u0` and `u1` from the module `design`, and the resulting design after removing the cells `u0` and `u1` will be equivalent to the following:

```
module design(...);
LIB_CELL2 u2 (...);
LIB_CELL3 u3 (...);
endmodule
```

### Related Command

ELABORATE DESIGN

READ DESIGN

WRITE BLACKBOX WRAPPER

## SET INSTANTIATION DEPTH

### **SET INSTANTIATION DEPTH**

<integer>  
(Setup Mode)

Use this command before the READ DESIGN, READ LIBRARY, and ELABORATE DESIGN commands to override the default instantiation depth limit. By default, the default instantiation depth is 100.

### **Tcl Command**

```
set_instantiation_depth
```

### **Parameters**

<integer>                      Specifies a positive integer for the instantiation depth limit.

### **Examples**

The following command sets the instantiation depth to 30:

```
set instantiation depth 30
```

**Note:** If you do not specify a positive integer, the tool issues an Error that says the specified number is too small. For example:

```
% set instantiation depth 0
// Error: The specified number 0 is too small.
```

## SET LOG FILE

### SET Log File

```
[<filename>
[-Replace | -Append]
[-PROGRESS | -NOPROGRESS]
[-NOBACKup]]
(Setup / Verify Mode)
```

Writes the transcript to a specified file. The log file contains the commands and all output information. As you review the file, identify commands by the keyword:

// Command:

**Note:** If the filename you specify already exists, you must use either the `-replace` or `-append` option. If you do not include an option, the Conformal software generates an error message that the file exists. If you receive this message, reenter the command with either a new filename or the appropriate option. If the filename is not writable, the software writes it to the `/tmp` directory.

If you are writing the transcript to a file, you might want to turn off the screen transcript display with the `SET SCREEN DISPLAY` command. If you do not specify otherwise, the transcript prints to the screen.

To store log files based on the software version, use the `VERIFY_VERSION` environment variable. For example:

```
set log file lec.$VERIFY_VERSION.log -replace
```

To verify the current log file setting, use the `REPORT ENVIRONMENT` command.

## Tcl Command

```
set_log_file
```

## Parameters

|            |                                                                                 |
|------------|---------------------------------------------------------------------------------|
| <filename> | Writes the transcript run to this file.                                         |
| -Replace   | If the specified filename already exists, overwrites the contents of that file. |
| -Append    | Appends the transcript run to the end of the specified filename.                |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|             |                                                                                        |
|-------------|----------------------------------------------------------------------------------------|
| -PROGRESS   | Writes the percentage completion progress to the log file. <i>This is the default.</i> |
| -NOPROGRESS | Does not write the percentage completion progress to the log file.                     |
| -NOBACKup   | Does not create a backup file.                                                         |

**Note:** If you do not specify this option, Conformal will create a backup file when you replace or append a file.

### Related Commands

SET SCREEN DISPLAY

REPORT ENVIRONMENT

## SET NAMING RULE

### SET NAMing Rule

```
<< -Hierarchical_separator <string>> |
< -Tristate <string>> |
< -REGister <string>> |
< -Inverted_pin_extension <string>> |
< -Parameter <string> [-VALUE_format <LEC | RC | DC>]> |
< -INSTANCE_Array <string>> |
< -Array_delimiter <left_string> <right_string>> |
< -Field_delimiter <left_string> <right_string>> |
< -INTERface_delimiter <left_string> <right_string>> |
< -INStance [-ALL | -VLG | -VHDL] <condgen_string> <forgen_string>
<instance_string>> |
< -VARIABLE [-ALL | -VLG | -VHDL] <condgen_string> <forgen_string>
<variable_string>> |
< < -NOMDPORTflatten | < -MDPORTflatten
[ALL | LOGIC_only | RECORD_only | UNION_only] > >
[-NOMDPORT_BITblast | -MDPORT_BITblast] > |
<-ENABLE_UNNAMED_blk_naming | -NOENABLE_UNNAMED_blk_naming> |
<-NOENABLE_PROC_name | -ENABLE_PROC_name> |
<-MAX_Para_mod_len <length_size>|
<-STRingformat <DEFAULT | HEX | STR>>
<-MOD_PARAM [USE_ANYWAY | USE_WHEN_DIFF | USE_ALL]>
(Setup Mode)
```

Specifies the naming rules for an RTL or hierarchical design. Conformal uses the renaming rule to construct the name for the design module, instance, variable, or port. Execute this command before **READ LIBRARY** and **READ DESIGN** (note that SET NAMING RULE - inverted\_pin\_extension does not have this requirement).

Each string for all of the settings must be enclosed in double quotes (" "). These double quotes can be empty.

Use the **REPORT ENVIRONMENT** command to display the settings for the naming rules.

### Tcl Command

```
set_naming_rule
```

### Parameters

```
-Hierarchical_separator <string>
```



## Conformal Extended Checks Reference Manual

### Command Reference

---

A character or string that specifies the hierarchical separator.  
*The default is "/"*.

The hierarchical separator setting has no effect on the way key points are reported (for example, when you use the `REPORT GATE` command).

`-Tristate <string>` A string specifies how tri-state names should be constructed.  
*The default is "%s\_tri". Where %s denotes the tristate variable name. The string must contain exactly one "%s".*

`-REGister <string>` A string that specifies how register names should be constructed. *The default is "%s\_reg". Where %s denotes the register name. The string must contain exactly one "%s".*

`-Inverted_pin_extension <string>`

A string that specifies the inverted pin extension.

`-Parameter <string> [-VALUE_format <LEC | RC | DC>]`

String to use in parameterized module names. When you instantiate an existing module with new parameter values, Conformal creates a new module and names it using the following renaming rule:

```
<original_module_name><<parameter_string>
<parameter value> *...>
```

Where "parameter\_string" is the string specified with this option. The default is "\_%s" (the parameter name).

[ `-VALUE_format < LEC | RC | DC >` ] is used to specify the "parameter value" format. By default the format is decided by `SET NAMING STYLE <lec | rc | genus | dc>`. You can use `-value_format` to control the format.

`-INSTANCE_Array <string>`

A string that specifies the instance array naming. The default is "%s[%d]". The string must contain at least one "%d". LEC will replace the first "%s" with the instance name and the first "%d" with the array index; all other "%s" and "%d" are kept as is.

`-Array_delimiter <left_string> <right_string>`

Two strings that specify the left and right array delimiter. *The default is "[" and "]" for the left and right string.*

`-Field_delimiter <left_string> <right_string>`

## Conformal Extended Checks Reference Manual

### Command Reference

---

Two strings that specify the left and right record (for VHDL) or struct (for SystemVerilog) field delimiters. *The default is "[" and "]" for the left and right string.*

`-INTERface_delimiter <left_string> <right_string>`

Two strings that specify the left and right SystemVerilog interface item delimiter. The default is "\_" and " " for the left and right string.

For an example, refer to Case 2 in the Examples section.

`-INSTance [-ALL | -VLG | -VHDL] <condgen_string> <for gen_string>  
<instance_string>`

Specifies how to construct instance names inside generate constructs. There are two types of generate constructs: conditional generate constructs that include if-generate and case-generate blocks, and generate constructs that include for-generate blocks.

`condgen_string` specifies how to include if-generate and case-generate block names in the instance name.

`for gen_string` specifies how to include for-generate block names in the instance name, where:

`%L` specifies the block name.

`%s` specifies the next substitute string. It could be substituted by `condgen_string`, `for gen_string`, or `instance_string` (depends on the instance location).

`%d` specifies the block index.

`instance_string` specifies how to name instance names, where:

`%s` specifies the current instance name.

*The default setting for <condgen\_string>  
<for gen\_string> <instance\_string> is:*

`"%L.%s" "%L[%d].%s" "%s"` for Verilog, Verilog2k, and SystemVerilog designs

`"%s" "%s_%d" "%s"` for VHDL designs

For an example, refer to Case 2 in the Examples section.

Note: According to the SystemVerilog LRM, unnamed generate constructs will be assigned a default generate block name. To control the auto-assigning, refer to the "set naming rule - ENABLE\_UNNAMED\_blk\_naming" command.

-ALL Specifies that this user-defined naming rule applies to Verilog/Verilog2k/SystemVerilog and VHDL designs. *This is the default.*

-VLG Specifies that this user-defined naming rule applies to only Verilog/Verilog2k/SystemVerilog designs.

-VHDL Specifies that this user-defined naming rule applies to only VHDL designs.

**Note:** The -ALL, -VLG, and -VHDL options must be used immediately after the -instance option.

```
-VARIABLE [-ALL | -VLG | -VHDL] <condgen_string> <forngen_string>
<variable_string>
```

The variable naming scheme is similar to the instance naming scheme.

condgen\_string specifies how to include if-generate and case-generate block names in the variable name.

forngen\_string specifies how to include for-generate block names in the variable name, where:

%L specifies the block name.

%s specifies the next substitute string. It could be substituted by condgen\_string, forngen\_string, or variable\_string (depends on the variable location).

%d specifies the block index.

variable\_string specifies how to specify the variable name, where:

%s specifies the current variable name.

*The default setting for <condgen\_string>  
<forngen\_string> <variable\_string> is:*

"%L.%s" "%L[%d].%s" "%s" for Verilog, Verilog2k, and SystemVerilog designs

"%s" "%s" "%s" for VHDL designs

## Conformal Extended Checks Reference Manual

### Command Reference

---

Note: According to the SystemVerilog LRM, unnamed generate constructs will be assigned a default generate block name. To control the auto-assigning, refer to the "set naming rule - ENABLE\_UNNAMED\_blk\_naming" command.

For an example, refer to Case 2 in the Examples section.

**-ALL** Specifies that this user-defined naming rule applies to Verilog/Verilog2k/SystemVerilog and VHDL designs. *This is the default.*

**-VLG** Specifies that this user-defined naming rule applies to only Verilog/Verilog2k/SystemVerilog designs.

**-VHDL** Specifies that this user-defined naming rule applies to only VHDL designs.

**Note:** The **-ALL**, **-VLG**, and **-VHDL** options must be used immediately after the **-variable** option.

**-NEGEDGE** <label\_string>

A string that specifies the domain of the negative-edge clock. *The default is %s\_neg.*

**-NOMDPORTflatten**

When synthesizing an HDL design, Conformal separates the ports of composite data types (such as structs, unions, records, or multi-dimension arrays) into multiple ports that are port-type compliant to Verilog 1995 (IEEE 1364-1995).

For example, a multi-dimension array port will be separated into multiple ports and each port will be a vector. With **-MDPORTflatten** set, ports from the same composite data port will be concatenated into a single vector port.

*This is the default.*

**-MDPORTflatten**

Flattens ports of composite data types (such as structs, unions, records, or multi-dimension arrays) into a single vector port. The vector range is `portSize-1` down to 0 where `portSize` is the bit size of the original port.

The default is **ALL**.

**ALL**—Flattens ports of all composite data types and multi-dimension arrays into a vector port.

**LOGIC\_only**—Flattens logic multi-dimension array ports into a vector port.

## Conformal Extended Checks Reference Manual

### Command Reference

---

RECORD\_only—Flattens struct and record type ports into a vector port.

UNION\_only—Flattens union type type ports into a vector port.

-NOMDPORT\_BITblast Does not bit blast ports of a module. *This is the default.*

-MDPORT\_BITblast Bit blast the ports of composite data types or multi-dimension arrays not flattened into a vector port by -MDPORTflatten into multiple ports of a single bit.

-ENABLE\_UNNAMED\_blk\_naming

By default, the tool renames unnamed blocks in accordance with Verilog-2005 LRM. For example:

```
module top #(parameter PARAM1 = 0, PARAM2 = 1) (
 input clk, input [1:0] datain,
 output [1:0] dataout
);
 wire [1:0] dataout_int;
 generate
 begin
 if (PARAM1)
 assign dataout_int = 2'b0;
 else if (PARAM1 == 0)
 begin : label1
 reg [1:0] mem_mod;
 ...
 end
 end
 endgenerate
 endmodule
```

The tool renames register `mem_mod` with `genblk1.label1.mem_mod_reg[0]` based on the Verilog-2005 LRM.

-NOENABLE\_UNNAMED\_blk\_naming

Skip the unnamed block naming. For the given example, the register will be renamed to `label1.mem_mod_reg[0]` (without `genblk1`).

-NOENABLE\_PROC\_name *This is the default.*

-ENABLE\_PROC\_name When the naming style for the RTL design is set to RC (through the command 'SET NAMING STYLE rc'), the tool prefixes variable names in the VHDL design with the process name.

-MAX\_Para\_mod\_len <length\_size>

## Conformal Extended Checks Reference Manual

### Command Reference

---

Specifies the maximum string length for parameterized module names. When you instantiate an existing module with new parameter values, the tool creates a new module and names it using the following renaming rule:

```
<orig_mod_name><<par_string><par value> *...>
```

The default length is 4096.

If the new module name is longer than the specified maximum length or longer than the default, the tool replaces it with a shorter, unique name. If a user-specified length is less than 32, the tool ignores this setting

-STRInGformat

When a parameter value is a string, use this option to control whether the string interpreted as an ASCII binary number displayed in decimal or hexadecimal notation, or displayed in string format.

The parameter value is used to compose the module name when creating a new module for an instantiation of a parameterized module (SET NAMING RULE -parameter).

Note: This option works only when the SET NAMING STYLE is set to "LEC"

DEFAULT—The string is converted into an ASCII binary number. For strings longer than 4 characters, the ASCII number is displayed in decimal base. Otherwise, the number is displayed in hexadecimal base. *This is the default.* See Example section.

HEX—The string is converted to ASCII binary number and displayed in hexadecimal base.

STR—String is kept in string format.

-MOD\_PARAM

USE\_ANYWAY—Compose parameterized module name with instance parameter data regardless whether the new value is the same as the default value or not. *This is the default setting.*

USE\_WHEN\_DIFF—Compose parameterized module name with parameter data only if the new parameter data is different from the default value.

USE\_ALL—Apply all parameters even if instance statement does not specify parameter setting or the new data is the same as default values.

## Examples

```
set naming rule -hierarchical_separator ":"
set naming rule -register "register_%s"
set naming rule -tristate "tristate_%s"
set naming rule -array "<" ">"
set naming rule -instance "%L.%s" "%L[%d].%s" "%s"
```

### Case 1: Generating Instance Names

```
generate
if (1) begin: blkA //if-generate

 for (j=0;j<=0;j=j+1) begin: forblkB //for-generate
 or n1(a,b,c);
 for (k=23;k<=23;k=k+1) begin: forblkC //for-generate
 and n2(d,e,f);
 end
 begin : blkD //if-generate
 nor n3 (g,h,i);
 end
 end
end
endgenerate
```

For this example (**Case 1**):

■ set naming rule -instance "%L.%s" "%L[%d].%s" "%s"

Renames instances n1, n2 and n3 as "\blkA.forblkB[0].n1", "\blkA.forblkB[0].forblkC[23].n2", and "\blkA.forblkB[0].blkD.n3", respectively. This is also the default setting for Verilog, Verilog2k, and SystemVerilog design.

In this example, the block hierarchy for n2 is:

```
blkA(if-generate) => forblkB (for-generate) => forblkC (for-generate)
```

The renaming process for n2 goes through the following steps:

1.) For blkA, which is a if-generate, n2 is substituted by `condgen_string "%L_%s"`. The result will be "blkA.%s".

2.) For forblkB, which is a for-generate, %s of "blkA\_%s" is substituted by `forgen_string "%L[%d].%s"`. The result will be "\blkA.forblkB[0].%s".

## Conformal Extended Checks Reference Manual

### Command Reference

---

3.) For `forblkC`, which is a `for-generate`, `%s` of `"\blkA.forblkB[0].%s"` is substituted by `forgen_string "%L[%d].%s"`. The result will be `"\blkA.forblkB[0].forblkC[23].%s"`.

4.) The `%s` of `"\blkA.forblkB[0].forblkC[23].%s"` is substituted by `instance_string "%s"`. The final result will be `"\blkA.forblkB[0].forblkC[23].n2"`

- `set naming rule -instance "%s" "%s_%d" "%s"`

Renames instances `n1`, `n2` and `n3` as `n1_0`, `n2_0_23`, and `n3_0`, respectively. This is also the default setting for VHDL design.

- `set naming rule -instance "%s" "%L[%d].%s" "%s_INS"`

Renames instances `n1`, `n2` and `n3` as `"\forblkB[0].n1_INS"`, `"\forblkB[0].forblkC[23].n2_INS"`, and `"\forblkB[0].n3_INS"`, respectively.



## Case 2: Generating SystemVerilog Interface Type Port/Variable Names

```
interface simple_bus;
 logic req, gnt;
 logic [7:0] addr, data;
endinterface: simple_bus

module memMod(simple_bus a, input bit clk);
 logic avail;
 always @(posedge clk) a.gnt <= a.req & avail;
endmodule
```

The tool generates port names `a_req`, `a_gnt`, `a_addr` and `a_data` for module `memMod`.

The following command renames ports `a_req`, `a_gnt`, `a_addr` and `a_data` as `\a.req`, `\a.gnt`, `\a_addr`, and `\a_data` respectively

```
set naming rule -INTERface_delimiter "." ""
```

## Case 3: Generating Instance Array Names

```
input [4:0] in;
output [4:0] out;
wire [4:0] out1;
sub s1 [4:0] (.in(out1), .out(out));
sub s2 [4:0] (.in(in), .out(out1));
```

For this example (Case 3):

- `set naming rule -instance_array "%s[%d]_%s_%s%d"`

The tool generates the following instance names:

```
\s1[4]_%s_%s%d
\s1[3]_%s_%s%d
\s1[2]_%s_%s%d
\s1[1]_%s_%s%d
\s1[0]_%s_%s%d
\s2[4]_%s_%s%d
\s2[3]_%s_%s%d
\s2[2]_%s_%s%d
\s2[1]_%s_%s%d
\s2[0]_%s_%s%d
```

- `set naming rule -instance_array "_%s_%s"`

The tool produces the following error because the rule does not contain a `%d`. The tool will generate the instance names using the default naming format:

```
// Error: Illegal INSTANCE_Array naming _%s_%s.
```

The tool generates the following instance names:

## Conformal Extended Checks Reference Manual

### Command Reference

---

```
\s1[4]
\s1[3]
\s1[2]
\s1[1]
\s1[0]
\s2[4]
\s2[3]
\s2[2]
\s2[1]
\s2[0]
```

- `set naming rule -instance_array "%d_%s_%s_%d"`

The tool generates the following instance names:

```
\4_s1_%s_%d
\3_s1_%s_%d
\2_s1_%s_%d
\1_s1_%s_%d
\0_s1_%s_%d
\4_s2_%s_%d
\3_s2_%s_%d
\2_s2_%s_%d
\1_s2_%s_%d
\0_s2_%s_%d
```

- `set naming rule -instance_array "_%d_%d"`

The tool generates the following instance names:

```
4%d
3%d
2%d
1%d
0%d
4%d_1
3%d_1
2%d_1
1%d_1
0%d_1
```

**Example of** `[-NOIGNORE_CASE_GEn_name | -IGNORE_CASE_GEn_name]:`

```
module test (input [3:0] d, output [7:0] q1);
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

```
parameter inv = 1;
parameter size = 8;
genvar j;

generate
for (j=0; j < 2; j ++) begin: [1]loop[/1]
 if (!j) begin: blk
 sub u1(d, q1[3:0]);
 end else if (j == inv) begin: [1]blk[/1]
 case(size)
 6 : begin : B1
 sub u1(d, q1[7:4]);
 end
 8 : begin : [1]B1[/1]
 sub u1(d, q1[7:4]);
 end
 endcase
 end
end
else begin: blk
 sub u1(d, q1[3:0]);
end
end
endgenerate
endmodule
```

By default, these are the generated module names:

```
TCL_LEC> report_modules -instance
Golden:
test
(1) \loop[0].blk.u1 (sub)
(2) [1]\loop[1].blk.B1.u1 (sub) [/1]
```

Specifying the `[1]set_naming_rule -ignore_case_gen_name[/1]` option, these are the generated module names:

```
TCL_LEC> report_modules -instance
Golden:
test
(1) \loop[0].blk.u1 (sub)
(2) [1]\loop[1].blk.u1 (sub) [/1]
```

## **Related Commands**

READ DESIGN

REPORT ENVIRONMENT

REPORT GATE

## SET NAMING STYLE

### SET NAMing Style

<LEC | RC | GENUS | DC>  
(Setup Mode)

Specifies the naming style for an RTL design. Execute this command before `READ DESIGN`.

This command handles standard naming (where the design uses only the default RC synthesis attributes); it does not handle naming caused by non-default RC synthesis attributes (which include, but are not limited to, attributes such as `inst_prefix` or `hdl_generate_index_style`). To handle those cases, you must also use the `ADD RENAMING RULE` or `SET NAMING RULE` commands.

The RTL Compiler recommended synthesis flow is detailed in the "Interfacing with Conformal Logical Equivalence Checker" chapter of the *Interfacing between RTL Compiler and Conformal User Guide*.

### Tcl Command

`set_naming_style`

### Parameters

|       |                                                                         |
|-------|-------------------------------------------------------------------------|
| LEC   | Applies the LEC naming style to the design. <i>This is the default.</i> |
| RC    | Applies the RTL Compiler naming style to the design.                    |
| GENUS | Applies the Genus naming style to the design.                           |
| DC    | Applies the Design Compiler similar naming style to the design.         |

### Examples

The following example illustrates the difference between the naming styles.

The following design file contains one arithmetic operator(\*) on line 7.

```
1 entity top is
2 port(A, B: in integer range 7 downto 0;
3 C: out integer range 64 downto 0);
4 end top;
5 architecture rtl of top is
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

```
6 begin
7 C <= A*B;
8 end rtl;
```

Using the LEC naming style, the instance name of the arithmetic operator will be `mult_7`.

Using the RC naming style, the instance name of the arithmetic operator will be `mul_7_9`.

Using the DC naming style, the instance name of the arithmetic operator will be `mult_7`.

### Related Commands

READ DESIGN

## SET PARAMETER

### SET Parameter

```
[<-MODule <moduleName>|-INTERface <interfaceName>>
[-Parameter [-INT |-STR|-ENUM] <paramName> <paramValue>]]
(Setup Mode)
```

Use this command before the READ DESIGN, READ LIBRARY, and ELABORATE DESIGN commands to override default parameter values assigned by source files. If a command has more than one parameter definition, Conformal uses the last parameter.

To specify multiple modules, interfaces, or parameters, you must use the corresponding option for each module, interface, or parameter that you want to set. For example:

```
set parameter -mod m1 -p param1 value1 -mod m2 -p param2 value2 \
-mod m2 -p param3 value3 -p int param4 value4
```

### Tcl Command

```
set_parameter
```

### Parameters

|                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -MODule <moduleName>                                   | Applies the parameter setting to the specified module.                                                                                                                                                                                                                                                                                                                                                                                                                            |
| -INTERface <interfaceName>                             | Applies this parameter setting to the specified interface.                                                                                                                                                                                                                                                                                                                                                                                                                        |
| -Parameter [-INT  -STR -ENUM] <paramName> <paramValue> | <p>Assigns module/interface parameters or replaces existing design/interface parameters.</p> <p>When using "-Parameter -INT &lt;paramName&gt; &lt;paramValue&gt;", &lt;paramValue&gt; is converted to an integer value, which can be a positive integer (1), negative integer (-1), an integer value recognized as a string ("1"/"-1"), or a Verilog style integer ("16'h0001"). When using a Verilog style integer, the value must be specified between double-quotes (" ").</p> |

When using `-Parameter -STR <paramName> <paramValue>`, the `<paramValue>` will be saved as a string.

When using `-Parameter -ENUM <paramName> <paramValue>` command, the `<paramValue>` is converted to a VHDL/SystemVerilog enumeration literal. For example, the following command sets the parameter P4 to VHDL/SystemVerilog enumeration literal GREEN:

```
set parameter -MOD m1 -P -enum P4 GREEN
```

Note: Any value that is not recognized as an unsigned decimal integer value is interpreted as a string value.

Note: If `-INT` or `-STR` is not specified, then the parameter value will be interpreted as an integer if it is not between double-quotes (" "), and as a string if it is between double-quotes. Therefore, if you want to specify a Verilog format value, it must be between double-quotes and used with the `-INT` option.

## Related Commands

ELABORATE DESIGN

READ DESIGN



## SET PREDEFINED SYNCH\_RULE

### SET PREDEFIned Synch\_rule

```
<rule_name>
<-SYNC_MODULE <module_name ...> >
| -DFF <min> [INFinity | <max>]
| [-FIRST_DFF | -FIRST_DLAT_OR_DFF]
| [-SYNC_CHAIN [BUFFer | WIRe | LOGic]
| [MULTiple_chain | SINGle_chain]]
| -MUX <min> <max> [-HOLD | -NOHOLD]
>
[-CDC_PATH [LOGic | WIRe | BUFFer]
[MULTiple_destination | SINGle_destination]]
[-Order <num>]
(Verify Mode)
```

As an alternative to adding your own synchronization rules (through the `ADD SYNCHRONIZATION RULE` command), Conformal offers predefined synchronization rules. Use this command to customize the predefined synchronization rules.

When you use the predefined synchronization rules alone—without user-defined synchronization rules—Conformal automatically enters the *Categorization* flow when you issue the `VALIDATE` command. In this flow, Conformal attempts to categorize the CDC paths in your design into one of the predefined synchronization rules. However, if you have user-defined synchronization rules, Conformal enters the normal *Validation* flow. The flow you are in affects the available options for the `REPORT VALIDATED DATA` command.

Conformal classifies the CDC paths into one of the predefined synchronization rules by validating the CDC path against the predefined synchronization rules, in linear order. In other words, Conformal categorizes a CDC path into the first predefined synchronization rule that it satisfies. Once a path satisfies a rule, Conformal ignores all subsequent rules.

A CDC path can be classified under `MIXED` if the vector has individual bits that are classified into different categories for word-level reporting. A CDC path can be classified under `OTHER`, if it does not satisfy any predefined synchronization rule. You can see the results of this categorization when you use the `REPORT VALIDATED DATA` command.

For more information, see "Using Predefined Synchronization Rules" in the Clock Domain Crossing Checks chapter of the *Conformal Extended Checks User Guide*.

## Tcl Command

```
set_predefined_synch_rule
```

## Parameters

|                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>&lt;rule_name&gt;</code>                    | Specifies one of the predefined synchronization rule names. To view the rule names, use the <code>REPORT PREDEFINED SYNCH_RULE</code> command.                                                                                                                                                                                                                                                                                  |
| <code>-SYNC_MODULE &lt;module_name ...&gt;</code> | <p>Modules are used as synchronizers.</p> <p>Specifies that Conformal Extended Checks will first check to ensure that a domain crossing path's destination D flip-flop is inside the specified module. If this condition is satisfied, Conformal Extended Checks runs a check on the <code>cdc_path</code> to ensure that logic outside the module boundary complies with all other parameters of the synchronization rule.</p> |
| <code>-DFF</code>                                 | D flip-flops are used as a synchronizer.                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>&lt;min&gt;</code>                          | The minimum number of flip-flops in the DFF synchronizer. (This <code>min</code> is an integer value: 1 or greater.)                                                                                                                                                                                                                                                                                                            |
| <code>INFINITY</code>                             | <p>The maximum number of flip-flops in the DFF synchronizer.</p> <p><i>If you do not set the max parameter, this is the default.</i></p>                                                                                                                                                                                                                                                                                        |
| <code>&lt;max&gt;</code>                          | The maximum number of flip-flops in the DFF synchronizer. (This <code>&lt;max&gt;</code> is an integer value that is greater than or equal to <code>&lt;min&gt;</code> .)                                                                                                                                                                                                                                                       |
| <code>-FIRST_DFF</code>                           | The first state element in the DFF synchronizer must be a D flip-flop (that is, a D-latch is not allowed as the first element in the DFF synchronizer). <i>This is the default.</i>                                                                                                                                                                                                                                             |
| <code>-FIRST_DLAT_OR_DFF</code>                   | The first state element in the DFF synchronizer can be either a D flip-flop or D-latch.                                                                                                                                                                                                                                                                                                                                         |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -SYNC_CHAIN    | Identifies what is allowed between the D flip-flops (or D-latch and D flip-flops) of the DFF synchronizer.<br><br>If you do not specify this option, <code>buffer</code> is the default, which means that only wire, buffers, and inverters are allowed between the state elements of the DFF synchronizer. Additionally, multiple fan-outs are allowed from within the DFF synchronizer chain.                                                                                                                   |
| BUfFer         | Allows only wire, buffers, and inverters between the state elements of the DFF synchronizer.                                                                                                                                                                                                                                                                                                                                                                                                                      |
| WIRe           | Allows only wire between the state elements of the DFF synchronizer.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| LOGic          | Allows any logic gate between the state elements of the DFF synchronizer.                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| MULtiple_chain | Allows multiple fan-out from within the DFF synchronizer chain.                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| SINgLe_chain   | Does not allow fan-out from within the DFF synchronizer chain.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| -MUX           | <b>Multiplexers are used as a synchronizer.</b><br><br><min> The minimum number of destination D flip-flops before the multiplexer.<br><br><max> The maximum number of destination D flip-flops before the multiplexer.<br><br>-HOLD Specifies that the data hold condition must exist for the MUX synchronizer. <i>This is the default.</i><br><br>The data hold condition describes a feedback loop that starts from the output of the destination register and goes back to the input of the MUX synchronizer. |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|-----------------------------------|------|-----------------------------------|--------|----------------------------------------------------------|----------------------|----------------------------------------|--------------------|-----------------------------------------------|
| -NOHOLD              | <p>Specifies that the data hold condition does not have to exist for the MUX synchronizer.</p> <p>The data hold condition describes a feedback loop that starts from the output of the destination register and goes back to the input of the MUX synchronizer.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| -CDC_PATH            | <p>Identifies what is allowed in the CDC path. The CDC path definition is different for different types of synchronizers:</p> <p>For DFF synchronizers, the CDC path is the signal path between source flip-flop and the destination flip-flop.</p> <p>For the MUX synchronization scheme, the CDC path is the signal path between the source flip-flop and destination flip-flop, excluding the MUX synchronizer element.</p> <p>For Sync module-based synchronizers, the CDC path is the signal path between source flip-flop and destination flip-flop, excluding the gates inside the specified Sync module.</p> <p>The defaults for this option are <code>logic</code> and <code>multiple_destination</code>, which means that Conformal Extended Checks supports any logic and multiple fan-outs after the source domain.</p> <table><tr><td>LOGic</td><td>Allows any logic in the CDC path.</td></tr><tr><td>WIRe</td><td>Allows only wire in the CDC path.</td></tr><tr><td>BUFfer</td><td>Allows only wire, buffer, and inverters in the CDC path.</td></tr><tr><td>MULtiple_destination</td><td>Allows multiple fan-outs from the CDC.</td></tr><tr><td>SINGle_destination</td><td>Does not allow any fan-out from the CDC path.</td></tr></table> | LOGic | Allows any logic in the CDC path. | WIRe | Allows only wire in the CDC path. | BUFfer | Allows only wire, buffer, and inverters in the CDC path. | MULtiple_destination | Allows multiple fan-outs from the CDC. | SINGle_destination | Does not allow any fan-out from the CDC path. |
| LOGic                | Allows any logic in the CDC path.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| WIRe                 | Allows only wire in the CDC path.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| BUFfer               | Allows only wire, buffer, and inverters in the CDC path.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| MULtiple_destination | Allows multiple fan-outs from the CDC.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| SINGle_destination   | Does not allow any fan-out from the CDC path.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |
| -Order <num>         | Specifies the linear order for the specified synchronization rule.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |       |                                   |      |                                   |        |                                                          |                      |                                        |                    |                                               |

## Related Commands

### ADD SYNCHRONIZATION RULE

## Conformal Extended Checks Reference Manual

### Command Reference

---

REPORT PREDEFINED SYNCH RULE

REPORT VALIDATED DATA

VALIDATE

## SET PROVE EFFORT

### SET PROVE Effort

<Low | Medium | High>  
(Setup / Verify Mode)

Specifies the amount of effort the Conformal software applies to proving a particular design. The system default is set to `low` prove effort. Therefore, if Conformal Extended Checks returns an explored depth (ED) proof status, increase the prove effort with this command.

**Note:** If you raise the effort level, you also increase the amount of time the Conformal software expends to check a particular design's static properties. Thus, you increase the total CPU time.

### Tcl Command

```
set_prove_effort
```

### Parameters

|        |                                                                                        |
|--------|----------------------------------------------------------------------------------------|
| Low    | Uses the default algorithms for proof. <i>This effort level is the system default.</i> |
| Medium | Uses more aggressive (and more time-consuming) algorithms for proof.                   |
| High   | Uses the most aggressive (and most time-consuming) algorithms for proof.               |

### Example

```
add static property -all // 10 properties
prove // 5 passed, 5 ED depth 12

set prove effort high
prove
// The 2nd proof begins where the previous proof
// ended, that is, the 5 EDs at depth 12.

// The result might now be 5 EDs at depth 20.
// Any subsequent proofs will start on the
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

```
// unproven properties from the depth already
// proven.
```

### Related Command

PROVE

## SET PROVE OPTIONS

### SET PROVE Options

```
[-FLatten [-Continue_at_top | -Stop_explored | -NOSTop_explored]
(Setup / Verify Mode)
```

Specifies whether Conformal Extended Checks runs proofs with the hierarchical (that is, bottom-up) or flat (that is, top-down) method.

### Tcl Command

set\_prove\_options

### Parameters

-FLatten                      Runs proofs with the flat (that is, top-down) method. *This is the default.*

-Continue\_at\_top  
                              If the submodule produces an Explored Depth result (ED), skips the levels in between and continue the proof at the top level. *This is the default.*

-Stop\_explored  
                              If the submodule produces an Explored Depth result (ED), stop the proof for that property.

-NOSTop\_explored  
                              If the submodule produces an Explored Depth result (ED), continues upward one hierarchy level at a time, until the top level is reached.

**Note:** This option requires more CPU time.

### Related Commands

PROVE



# Conformal Extended Checks Reference Manual

## Command Reference

---

### REPORT ENVIRONMENT

## SET SPICE OPTION

### SET SPice OPTion

```
[-BULk | -NOBULk]
[-BBOX | -NOBBox]
[-NOADDGLOBALPINs | -ADDGLOBALPINs]
[-NOKEep_InstanceX | -KEep_InstanceX]
(Setup Mode)
```

**Note:** This requires a Conformal GXL license for equivalence checking that requires circuit analysis and logic abstraction. This requires a Conformal LPGXL license for circuit analysis of designs with low power features to find leakage paths.

Specifies options for parsing a Spice or Verilog switch-level netlist.

### Tcl Command

```
set_spice_option
```

### Parameters

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -BULK            | Identifies nets connected to PMOS bulk terminals as power and nets connected to NMOS bulk terminals as ground. <i>This is the default.</i>                                                                                                                                                                                                                                                                          |
| -NOBULK          | By default, Conformal GXL identifies nets connected to PMOS bulk terminals as power and nets connected to NMOS bulk terminals as ground. However, this option removes that assumption and you will need to specify power and ground pins/nets by using *.GLOBAL <name>:P for power nets and *.GLOBAL <name>:G for ground nets or use Conformal commands to add constraints, tied signals (pins), or net attributes. |
| -BBOX            | Specifies that SUBCKT contains no transistors and will be treated as a blackbox. <i>This is the default.</i>                                                                                                                                                                                                                                                                                                        |
| -NOBBox          | Specifies that SUBCKT contains no transistors and will be removed along with all of its instantiations.                                                                                                                                                                                                                                                                                                             |
| -NOADDGLOBALPINs | Specifies that no extra ports for GLOBAL signals will be created for SUBCKT. <i>This is the default.</i>                                                                                                                                                                                                                                                                                                            |
| -ADDGLOBALPINs   | Specifies that extra ports for GLOBAL signals will be created for SUBCKT.                                                                                                                                                                                                                                                                                                                                           |

## Conformal Extended Checks Reference Manual

### Command Reference

---

- |                   |                                                                                                                  |
|-------------------|------------------------------------------------------------------------------------------------------------------|
| -NOKEep_InstanceX | Specifies that the first character 'X' of the name of instance will not be retained. <i>This is the default.</i> |
| -KEep_InstanceX   | Specifies that the first character 'X' of the name of instance will be retained.                                 |

## Related Commands

READ DESIGN

## SET ROOT MODULE

**SET ROOT Module**  
    <module\_name>  
    (*Setup Mode*)

After you read in a design, Conformal Extended Checks treats the top module as the root module by default. You can manually reset a root module with the `SET ROOT MODULE` command.

Use this command to focus on specific parts of a design for verification and debugging.

**Note:** Use the `REPORT ENVIRONMENT` command to display the settings for the design's root module.

### Tcl Command

`set_root_module`

### Parameters

|               |                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <module_name> | Specifies that this module is the root. This option lets you override the automatic assignment made when you executed the <code>READ DESIGN</code> command. |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|

### Related Commands

`READ DESIGN`

`REPORT ENVIRONMENT`

## SET RTL TYPE

### SET Rtl Type

```
<-V1995|-VERILOG2K|-SYStemverilog> <file_extension* ...>
(Setup Mode)
```

### Description

Specifies the Verilog language standard for the file extensions. When specifying `-MIXvlog` with the `READ DESIGN` command, the file extension will be used to determine the Verilog language standard to parse the Verilog file. The default file extensions for Verilog standards are as follows:

- Verilog 1995: .v95, .v95p
- Verilog 2K: .v, .vp, .v2k
- SystemVerilog: .sv, .svp

### Tcl Command

```
set_rtl_type
```

### Parameters

|                                     |                                                                                                                           |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| <code>-V1995</code>                 | Specifies a Verilog-1995 design.                                                                                          |
| <code>-VERILOG2k</code>             | Specifies a Verilog2k design.                                                                                             |
| <code>-SYStemverilog</code>         | Specifies a SystemVerilog design.                                                                                         |
| <code>&lt;file_extension&gt;</code> | Specifies the file extension for specified Verilog standard. The file extension must include "." and is case-insensitive. |

## Examples

The following example demonstrates how to specify Verilog design standard for file extensions.

```
SET RTL TYPE -V1995 .v95 .v95p .vg .gv
SET RTL TYPE -VERILOG2K .v .vp .v2k
SET RTL TYPE -SYStemverilog .sv .svp .vs
```

## Related Commands

READ DESIGN

## SET RULE HANDLING

### SET Rule Handling

```
<[<rule_name* ...> [-Warning | -Critical | -Error [-CONTinue] | -Ignore | -
Note]]
 [-EXCLude <-MODULE | -DESIGN_FILE | -LIB_FILE
 | -IFDEF_region | -OVL_instances > <name* ...>]>
[-LIMit <max>]
(Setup Mode)
```

Changes the default level of severity for rule check messages (such as HDL and low power rule check messages) or excludes specified entities from rule checking. You can also use it to limit the number of occurrences for the specified rules. The `REPORT RULE CHECK` command will include in its output any limit set on the number of occurrences of any reported rule.

Run this command before `READ LIBRARY` and `READ DESIGN`.

**Note:** The `REPORT RULE CHECK -help` command displays a list of all the predefined rules and their severity levels.

### Tcl Command

`set_rule_handling`

### Parameters

|                                     |                                                                                                                                                                                                                                                                                                                                                           |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>&lt;rule_name* ...&gt;</code> | Changes rule handling for the specified rules. This accepts wildcards.                                                                                                                                                                                                                                                                                    |
| <code>-Warning</code>               | The rule handling will be a warning message. <i>This is the default.</i>                                                                                                                                                                                                                                                                                  |
| <code>-Critical</code>              | The rule handling will be critical. Rules marked as critical are run before any non-essential, non-critical rules. The tool will stop if it encounters violations of a critical rule. Users cannot downgrade critical rules defined by Conformal LP.<br><br>To report critical rules, use the <code>REPORT RULE CHECK -overview -critical</code> command. |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|          |                                                        |                                                                                                                                                                                                                    |
|----------|--------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -EXCLude | -Error [-CONTinue]                                     | The rule handling will be an error message. The <code>-continue</code> option can be used on RTL-related rules or tasks to indicate that the program continue to run instead of erroring out.                      |
|          | -Ignore                                                | The rule handling will be ignore. See <a href="#">REPORT RULE CHECK</a> to see how this option affects reporting.                                                                                                  |
|          | -Note                                                  | The rule handling will be a note.                                                                                                                                                                                  |
|          | Excludes the specified entity from the HDL Rule check: |                                                                                                                                                                                                                    |
|          | -MODule                                                | Excludes the specified module from the HDL Rule check.                                                                                                                                                             |
|          | -DESIgn_file                                           | Excludes the specified design file from the HDL Rule check.                                                                                                                                                        |
|          | -LIB_file                                              | Excludes the specified library file from the HDL Rule check.                                                                                                                                                       |
|          | -IFDEF_region                                          | Excludes DIR6.2 rule violations on all <code>`ifdef</code> blocks starting with the specified name*.                                                                                                               |
|          |                                                        | <b>Note:</b> This option applies exclusively to DIR6.2. For example:<br><br><pre>set rule handling -exclude<br/>-ifdef_region ASSERT_* ASSUME_*</pre>                                                              |
|          |                                                        | <b>Note:</b> An <code>`ifdef</code> region spans from the beginning <code>`ifdef</code> to its corresponding <code>`endif</code> , including any <code>`else</code> or <code>`elsif</code> that exists in between. |
|          | -OVL_instances                                         | Excludes all HRC3.8 rule violations caused by OVL assertions.                                                                                                                                                      |
|          |                                                        | <b>Note:</b> This option applies exclusively to HRC3.8. For example:<br><br><pre>set rule handling -exclude<br/>-ovl_instances *</pre>                                                                             |
|          | <name* ...>                                            | The name of the excluded entity.                                                                                                                                                                                   |



## Conformal Extended Checks Reference Manual

### Command Reference

---

`-LiMit <max>`

Limits the number of occurrences of the specified rules.

`<max>` is the limit after which no more occurrences will be recorded. Changing this limit will only have effect the next time the specified rules are checked.

### Example

```
set rule handling CLK1 -error
set rule handling -exclude -module abc
```

The initialization dofile can contain a command like the following:

```
set rule handling -limit 100 *
```

### Related Commands

REPORT RULE CHECK

## SET SCREEN DISPLAY

**SET S**creen **D**isplay  
    <ON | OFf>  
    [-PROGRESS | -NOPROGRESS]  
    (*Setup / Verify Mode*)

Specifies whether the transcript output is displayed on the terminal screen. The system default is set to on. If you set the screen display to off, use the `SET LOG FILE` command to save the transcript to a file.

**Note:** The `REPORT ENVIRONMENT` command displays the current setting.

### Tcl Command

`set_screen_display`

### Parameters

|             |                                                                                                            |
|-------------|------------------------------------------------------------------------------------------------------------|
| ON          | Displays the transcript to the terminal screen.                                                            |
| OFf         | Does not display the transcript to the terminal screen.                                                    |
| -PROGRESS   | Displays the percentage of completion on the terminal screen.<br><i>This option is the system default.</i> |
| -NOPROGRESS | Does not display the percentage of completion.                                                             |

### Related Commands

`REPORT ENVIRONMENT`

`SET LOG FILE`

## SET STATETABLE

**SET STATETable**  
<ON | OFF>  
(Setup Mode)

Controls the global setting of the Synopsys Liberty state table support.

**Note:** This command must be used before `READ DESIGN` and `READ LIBRARY`.

**Note:** Using the `READ DESIGN` and `READ LIBRARY` command's `-STATETable` option supersedes these settings (it also supersedes the global setting).

### Tcl Command

`set_statetable`

### Parameters

|     |                                                                                              |
|-----|----------------------------------------------------------------------------------------------|
| ON  | Enables support for Synopsys Liberty state tables. <i>This option is the system default.</i> |
| OFF | Disables support for Synopsys Liberty state tables.                                          |

### Related Commands

`READ DESIGN -STATETable`

`READ LIBRARY -STATETable`

## SET SYNTHESIS\_OFF\_COMMAND

**SET SYNTHESIS\_OFF\_Command**  
    <string>  
    (*Setup Mode*)

Specifies the pragma that is used to indicate the beginning of non-synthesizable constructs in the source code or in the generated generic netlist.

**Note:** If you do not run the SET\_ATTR INPUT\_PRAGMA\_KEYWORD command prior to running this command, the default is `translate_off`.

**Note:** If this command is run multiple times, only the last value is used.

### Tcl Command

`set_synthesis_off_command`

### Parameters

<string>                      Specifies the action.  
*Default:* `translate_off synthesis_off`

### Examples

Sample Dofile:

```
set_attr input_pragma_keyword rtl
set_synthesis_off_command turn_off
set_synthesis_on_command turn_on
```

After running these three commands, the Conformal and VHDL parsers will recognize the pragmas in the VHDL and Verilog Source files.

In a VHDL file, the code between `-- rtl turn_off` and `-- rtl turn_on` will not be synthesized.

In a Verilog file, the code between `// rtl turn_off` and `// rtl turn_on` will not be synthesized.

## **Related Commands**

SET\_ATTR INPUT\_PRAGMA\_KEYWORD

SET\_SYNTHESIS\_ON\_COMMAND

## SET SYNTHESIS\_ON\_COMMAND

**SET SYNTHESIS\_ON\_Command**  
    <string>  
    (*Setup Mode*)

Specifies the pragma that is used to indicate the end of non synthesizable constructs in the source code or in the generated generic netlist.

**Note:** If you do not run the SET\_ATTR INPUT\_PRAGMA\_KEYWORD command prior to running this command, the default is `translate_on`.

**Note:** If this command is run multiple times, only the last value is used.

### Tcl Command

`set_synthesis_on_command`

### Parameters

<string>                      Specifies the action.  
*Default:* `translate_on synthesis_on`

### Examples

Sample Dofile:

```
set_attr input_pragma_keyword rtl
set synthesis_off_command turn_off
set synthesis_on_command turn_on
```

After running these three commands, the Conformal and VHDL parsers will recognize the pragmas in the VHDL and Verilog Source files.

In a VHDL file, the code between `-- rtl turn_off` and `-- rtl turn_on` will not be synthesized.

In a Verilog file, the code between `// rtl turn_off` and `// rtl turn_on` will not be synthesized.

## **Related Commands**

SET\_ATTR INPUT\_PRAGMA\_KEYWORD

SET\_SYNTHESIS\_OFF\_COMMAND

## SET SYSTEM MODE

### **SET System Mode**

<Setup | VErify>  
(Setup / Verify Mode)

Switches system modes between the Setup mode and the Verify mode. In Setup mode, you can read in the design and set all the necessary constraints and environment variables. In Verify mode, the Conformal Extended Checks software runs the proof and diagnosis.

**Note:** Use the `REPORT ENVIRONMENT` command to display the current system mode.

### **Tcl Command**

`set_system_mode`

### **Parameters**

|        |                                     |
|--------|-------------------------------------|
| Setup  | Switches the system mode to Setup.  |
| VErify | Switches the system mode to Verify. |

### **Related Command**

`REPORT ENVIRONMENT`



## SET UNDEFINED CELL

### SET UNDEFINED Cell

```
<Error |Black_box [-AUTO_Assign |-NOAUTO_Assign] [-ASCEND |-NOASCEND]>
(Setup Mode)
```

Specifies how Conformal Extended Checks handles undefined cells it encounters when reading a design. The default is to issue an error message if it encounters undefined cells.

Run this command before `READ LIBRARY` and `READ DESIGN`.

Use the `REPORT ENVIRONMENT` command to display the handling settings for the design's undefined cells.

*Note regarding the direction of blackbox pins:* Because blackboxes are undefined, the direction of their pins is automatically inferred from connectivities in the blackbox instance's parent module. For example, if the net associated with pin `d[31]` of the blackbox instance has a known driver (such as, the net is assigned by a continuous assignment or the net is connected to an instance's output pin), the pin `d[31]` is a blackbox input pin. If no drivers are found for the pin `d[31]`, the pin `d[31]` is a blackbox output pin.

## Tcl Command

```
set_undefined_cell
```

## Parameters

|              |                                                                                                                                                                                                                                                      |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Error        | When Conformal Extended Checks reads the designs, undefined cells trigger an error message. <i>This handling is the initial system default.</i>                                                                                                      |
| Black_box    | When Conformal Extended Checks reads the designs, undefined cells are treated as blackboxes.                                                                                                                                                         |
| -AUTO_Assign | Conformal Extended Checks automatically determines and assigns directions to all blackbox pins. If Conformal Extended Checks cannot determine the direction of a pin as "input" or "output", it assigns "I/O" direction. <i>This is the default.</i> |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|                |                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| -NOAUTO_Assign | Conformal Extended Checks assigns "I/O" directions to all blackbox pins. You must manually reassign all pin directions according to the design. |
| -ASCEND        | Arranges bits of bus pins in ascending order; that is, in1(0 to 7). <i>This is the default.</i>                                                 |
| -NOASCEND      | Arranges bits of bus pins in descending order; that is, out1(7 down to 0).                                                                      |

### Related Commands

READ DESIGN

READ LIBRARY

REPORT ENVIRONMENT

## SET UNDEFINED PORT

**SET UNDEFINED Port**  
<Error | Ignore>  
(Setup Mode)

Specifies how Conformal handles undefined ports it encounters when reading in the libraries and designs. *The system default is to report an error message if there are any undefined ports referenced by the module instance.*

Use the `REPORT ENVIRONMENT` command to display the settings for the undefined ports handling for the designs. Execute this command before `READ LIBRARY` and `READ DESIGN`.

### Tcl Command

`set_undefined_port`

### Parameters

|        |                                                |
|--------|------------------------------------------------|
| Error  | Displays an error message for undefined ports. |
| Ignore | Ignores undefined ports.                       |

### Related Commands

`READ DESIGN`

`READ LIBRARY`

`REPORT ENVIRONMENT`

## SET UNDRIVEN SIGNAL

### SET UNDRiven Signal

<Z | 0 | 1 | X >  
(Setup Mode)

Sets all undriven signals in the design to Z, 0, 1, or X. *By default, all undriven signals are set to Z.*

Run this command before READ LIBRARY and READ DESIGN.

Use the REPORT ENVIRONMENT command to display the settings for the design's undriven signals.

### Tcl Command

```
set_undriven_signal
```

### Parameters

|   |                                                                                                                        |
|---|------------------------------------------------------------------------------------------------------------------------|
| Z | Specifies undriven signals as high impedance (always driven by Z). <i>This handling is the initial system default.</i> |
| 0 | Specifies undriven signals as Logic 0.                                                                                 |
| 1 | Specifies undriven signals as Logic 1.                                                                                 |
| X | Specifies undriven signals as unknown (X).                                                                             |

### Related Commands

REPORT ENVIRONMENT

REPORT FLOATING SIGNALS

## SET X HANDLING

### SET X Handling

```
<-ALL | -BBOX | -DC>
[<ACCEPT | IGNore>]
(Setup Mode)
```

Specifies a global treatment for the design's counter-example blackbox outputs. *By default, Conformal Extended Checks treats blackbox outputs as primary inputs, accepting them as valid counter-examples.* Therefore, on finding a counterexample that disproves a property, Conformal Extended Checks discontinues its search for additional counterexamples, even though the counter-example contained blackbox outputs.



**Using this default can result in false negatives.**

With the IGNORE setting, Conformal Extended Checks does not accept blackbox outputs as valid counter-examples unless *all* counter-examples contain blackbox outputs. With this setting, Conformal Extended Checks searches the entire solution space to find a counterexample that does not contain blackbox outputs.

## Tcl Command

set\_x\_handling

## Parameters

|        |                                                                                                  |
|--------|--------------------------------------------------------------------------------------------------|
| -ALL   | Specifies how blackbox outputs and Don't Cares are treated in counterexamples.                   |
| -BBOX  | Specifies how blackbox outputs are treated in counter-examples.                                  |
| -DC    | Specifies how Don't Cares are treated in counter-examples.                                       |
| ACCEPT | Accepts blackbox outputs as normal inputs in counterexamples. <i>This option is the default.</i> |
| IGNore | Does not accept blackbox outputs as normal inputs in counterexamples.                            |

## **Related Command**

REPORT ENVIRONMENT

## SET WEB INTERFACE

```
SET WEB_interface
 [ON | OFF]
 [-DOOnly | -NODOOnly]
 [-PORT <integer>]
 [-BROWser]
 [-ANYHOST]
 [-DIRLEVEL <integer>]
 [-FILEPermission | -NOFILEPermission]
 (Setup / Verify Mode)
```

Use this command to access Conformal FAQs and beta documentation. This command starts a web server so that you can view the content in one of the following web browsers: Internet Explorer 9, Firefox 4, and Chrome 10.

**Note:** This command is available for beta testing. The features described may change prior to final release.

### Tcl Command

```
set_web_interface
```

### Parameters

|                 |                                                                                                            |
|-----------------|------------------------------------------------------------------------------------------------------------|
| ON              | Starts the web interface.                                                                                  |
| OFF             | Stops the web interface.                                                                                   |
| -DOOnly         | Launches the web interface with only the Conformal documentation viewer. <i>This is the default.</i>       |
| -NODOOnly       | Launches the full web interface viewer, which contains Conformal documentation and file directories.       |
| -PORT <integer> | Specifies a different port number.                                                                         |
| -BROWser        | Displays the web interface in a stand-alone browser.                                                       |
| -ANYHOST        | Allows other machines to access the web interface. Without this option, only the local host can access it. |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|                                        |                                                                                                                                                                                                             |
|----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-DIRLevel &lt;integer&gt;</code> | <p>Specifies how many directory levels the web interface can access. Where level 0 means unlimited. Level 1 is the current directory.</p> <p>Default is 6.</p>                                              |
| <code>-FILEPermission</code>           | <p>Specifies that the web interface can only access files with world read permissions. The read permissions of the file must be enabled for user, group, and others.</p> <p><i>This is the default.</i></p> |
| <code>-NOFILEPermission</code>         | <p>Specifies that the web interface can access files without checking world readable permissions.</p>                                                                                                       |

## Example

The following describes how to view the web interface with file directories and documentation:

### 1. Start the web server for web viewing:

```
SETUP> set web_interface ON
//Web Interface URL is http://host.xyz.com:8090 (http://1.2.3.4:8090)
// A modern browser supporting HTML5 is required.
// File browsing is enabled. To limit access to documentation only, use the -
DOOnly option.
```

### 2. The tool returns a URL of the web interface. Copy this URL into your web browser.

The following illustrates how to view the web interface with just the documentation:

```
SETUP> set web_interface ON -DOOnly
// A modern browser supporting HTML5 is required.
// Browsing is limited to documentation only.
```

**Note:** The port number is unique for each LEC run on a machine. If you try to start a server for a port that is already in use, the tool returns a message similar to the following:

```
Failed to bind to port 8090: Address already in use
// Error: Web interface server cannot be created.
```

In this case, use the `-PORT` option and specify a new port number:

```
SETUP> set web_interface ON -port 8091
// Web Interface URL is http://hostname:8091
```



## SETENV

### SETENV

<variable> <value>  
(Setup / Verify Mode)

Changes or adds the specified variable name to the environment with the specified value.

### Tcl Command

setenv

### Parameters

|            |                                                 |
|------------|-------------------------------------------------|
| <variable> | Adds the specified variable to the environment. |
| <value>    | Assigns the specified value to the variable.    |

### Related Command

PRINTENV

## SUBSTITUTE BLACKBOX WRAPPER

### **SUBStitute BLaKbox Wrapper**

```
<pattern_list>
[-Golden | -Revised]
(Setup Mode)
```

Searches for each blackbox instance in your design whose module name matches those in a specified pattern list, and replaces them with new, fully-defined modules. Use this module in conjunction with the WRITE BLACKBOX WRAPPER command.

If you enabled the automatic blackbox substitution feature using the `WRITE BLACKBOX WRAPPER -auto_substitute` command, you do not need to use this command as the models are automatically replaced.

### Tcl Command

```
substitute_blackbox_wrapper
```

### Parameters

|                                   |                                                                                                                         |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| <code>&lt;pattern_list&gt;</code> | Searches for blackbox instances whose module name matches the specified pattern(s). This option accepts the * wildcard. |
| <code>-Golden</code>              | Search and replace only in the Golden design.                                                                           |
| <code>-Revised</code>             | Search and replace only in the Revised design.                                                                          |

### Example

The following is a set of sample commands that show this and related commands in context.

#### Sample module:

```
<<< des.v>>>
module design(clk, rst, cs, wr, rd_addr, wr_addr, din, dout);
input clk, rst, cs, wr;
input [2:0] rd_addr, wr_addr;
input [4:0] din;
output[4:0] dout;

DW_ram_r_w_s_dff #(5, 8, 0) ram (.clk(clk), .rst_n(rst), .cs_n(cs),
.wr_n(wr), .rd_addr(rd_addr), .wr_addr(wr_addr), .data_out(dout), .data_in(din));
endmodule
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

1. Specify that Conformal treat undefined cells as blackboxes.

```
> set undefined cell black_box
```

2. Read in the design, which contains our sample module.

```
> read design des.v
```

3. Write a wrapper file `dir/_DW_ram_r_w_s_dff_5_8_0.v` for blackbox module `DW_ram_r_w_s_dff_5_8_0`.

```
> write blackbox wrapper DW* -directory dir
```

```
> break
```

**Note:** This command also generates synthesis script template file `dir/RUN/synth.tcl`.

4. Use the `dir/RUN/synth.tcl` script with your own synthesis tool to generate `dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v`.

5. Read the newly created `dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v` file into the design.

```
> read design -append dir/RUN/*.g.v
```

6. Substitute the old module of blackbox instance `ram` with the new module `_DW_ram_r_w_s_dff_5_8_0_DW_ram_r_w_s_dff_5_8_0_0` (`dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v:229`).

```
> substitute blackbox wrapper DW*
```

## Related Command

### WRITE BLACKBOX WRAPPER

## SYSTEM

### **sYstem**

`<string_command>`  
(*Setup / Verify Mode*)

Enables any command your UNIX operating system recognizes. When Conformal is in GUI mode, the return is printed in the transcript window.

You can substitute the exclamation mark (!) for the word "System", as shown in the "Example" section, below.

### **Tcl Command**

`system`

### **Parameters**

`<string_command>`      Any valid UNIX command.

### **Example**

```
system ls
system pwd
!pwd
```

## TCLMODE

### TCLMODE

(Setup / Verify Mode)

Switches the command entry mode from VPX to Tcl. VPX mode is the default command mode.

There are two types of Tcl mode commands: Native and Conformal.

**Note:** When issuing commands in Tcl mode mode, you must type them in lowercase. In Tcl mode, you must use a backslash to specify the following special characters: { } \$ [ ] ".

For more information about native Tcl commands, Refer to the public Tcl manual, which is widely available online. To learn more about Conformal Tcl commands, refer to the *Conformal Extended Checks User Guide*.



#### Tip

To start the Conformal software in Tcl mode without executing any initialization script, run the following command at a UNIX system prompt:

```
UNIX% lec -verify -tclmode
```



#### Tip

In the Tcl command entry mode, you can save report data to files using the redirection command. For example, the following command saves the gate report data to a file named `gate.out`:

```
TCL_SETUP> report_gate -type dff > gate.out
```

## Tcl Mode

tclmode

## Related Command

### VPXMODE

## TEST RENAMING RULE

### TEST RENaming Rule

<string>  
(Setup / Verify Mode)

Tests the renaming rules you created using the `ADD RENAMING RULE` command.

### Tcl Command

`test_renaming_rule`

### Parameters

|          |                                                                              |
|----------|------------------------------------------------------------------------------|
| <string> | Specifies the object name for which you would like apply the renaming rules. |
|----------|------------------------------------------------------------------------------|

### Example

If you created two renaming rules using these commands:

```
add renaming rule r1 "_reg_%d" "_reg[@1]"
add renaming rule r2 "/N01" "/Q"
```

You can use the `TEST RENAMING RULE` command to test these rules against the "fsm\_state\_reg\_2/N01" object name:

```
test renaming rule fsm_state_reg_2/N01
```

Conformal displays messages similar to the following:

```
Renaming rule: r1. Str: fsm_state_reg_2/N01
Result: fsm_state_reg[2]/N01
Renaming rule: r2. Str: fsm_state_reg[2]/N01
Result: fsm_state_reg[2]/Q
```

### Related Commands

[ADD RENAMING RULE](#)

[DELETE RENAMING RULE](#)

## USAGE

### USAge

```
[| -Elapse | -Delta]
[-MIN_COMMAND_SECONDS <double>]
[-NOAuto | -Auto]
(Setup / Verify Mode)
```

Displays the total CPU run time and current memory use during the current Conformal Extended Checks session.

## Tcl Command

usage

## Parameters

|                               |                                                                                                                                                                                                                                          |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -Elapse                       | Displays the elapsed time of a process.                                                                                                                                                                                                  |
| -Delta                        | Displays the difference, in seconds, between the current CPU run time and CPU run time when you last issued the <code>usage</code> command.                                                                                              |
| -MIN_COMMAND_SECONDS <double> | <p>For a single command, sets the minimum CPU run time (in seconds) required to trigger the automatic print out of usage.</p> <p>You must set the <code>-auto</code> option to use this option.</p> <p>Default value is 0.1 seconds.</p> |
| -NOAuto                       | Does not automatically run this command after running other commands. <i>This is the default.</i>                                                                                                                                        |
| -Auto                         | Automatically displays usage at the end of every command.                                                                                                                                                                                |

# VALIDATE

**VALidate**

-RTL

(Verify Mode)

## ***For extended checks:***

Verifies clock domain crossing checks.

**Note:** This command performs clock domain analysis. If there are any violations, an ACX\* rule violation is issued.

## **Tcl Command**

validate

## **Parameters**

-RTL

Selects the clocking schemes and validates crossings from register to register.

## **Related Commands**

ADD CDC CHECK

SET PREDEFINED SYNCH\_RULE

REPORT VALIDATED DATA



## VERSION

**VERsion**

*(Setup / Verify Mode)*

Displays the current Conformal version release number. You can use this command after the `SET LOG FILE` command so the version becomes a part of the transcript log. In this way, you record the Conformal version that created your results.

### Tcl Command

`version`

### Related Command

`SET LOG FILE`

## VPXMODE

### VPXMode

(Setup / Verify Mode)

Returns Conformal to the default VPX command entry mode.

**Note:** The commands in this reference manual are the VPX commands. Refer to the *Conformal Extended Checks User Guide* to learn more about Conformal Tcl commands.

### Important

When issuing this command from the Tcl command interpreter, you must type this in lowercase. For example:

```
TCL> vpxmode
```

### Tip

In VPX mode, you can save report data to files using the redirection command. For example, the following command saves the gate report data to a file named `gate.out`:

```
SETUP> report gate -type dff > gate.out
```

## Tcl Mode

vpxmode

## Related Command

TCLMODE

## WRITE BLACKBOX WRAPPER

### WRite Blackbox Wrapper

```
<pattern_list>
[-DIRectory] <dirname>
[-NDM_library <ndm_library_list>]
[-NOAUTO_substitute | -AUTO_substitute [-LATENCY <integer>]]
[-NODESIGNRule]
[-SYNTHESIS_library <STANDARD | <library_list>]
[-SYNTHESIS_Resource <filename>]
[-TARGET_library <library_list>]
[-TECH_file_library <tech_file_list>]
(Setup Mode)
```

Use this command to replace blackbox models with synthesis models.

This command generates a file that contains a wrapper for each blackbox instance in your design whose module name matches those in the specified pattern list and generates the scripts necessary for performing synthesis and generating the synthesized models.

You can also enable automatic *blackbox substitution* using the `-auto_substitute` option. This option executes the synthesis executable (specified by `SET HDL OPTIONS - synthesis_executable`) and automatically substitutes the blackbox models with the synthesized models.

**Tip:** Instead of using the ADD NOTRANSLATE MODULES command to treat particular cells as blackboxes, you can use the `set undefined cell -black_box` command prior to reading in the design and library (through the READ DESIGN and READ LIBRARY commands) to treat all undefined cells as blackboxes.

### Tcl Command

```
write_blackbox_wrapper
```

### Parameters

<pattern\_list>

Writes out module wrappers for blackbox instances whose module name matches the specified pattern(s).

This option accepts the \* wildcard.

## Conformal Extended Checks Reference Manual

### Command Reference

---

- `-DIRectory <dirname>` Specifies the working directory for this command. When this command is used, the tool creates a `RUN` directory under this working directory that will contain all the files generated by this command.
- If you do not specify this option, the tool creates a working directory called `CFM_BBOX_DIR` under your current working directory. All the files will be stored under this directory, in a subdirectory called `RUN`.
- If you specify a project directory, by default, the working directory `CFM_BBOX_DIR` is created underneath the project directory instead.
- Each time this command is used, the tool creates a new `RUN` directory under the working directory (appended by `*.<integer>`, where `integer` is the number of times the command has been issued). For example, `RUN.3`.
- `-NDM_library <ndm_library_list>` Specifies the user's ndm library files.
- `-NOAUTO_substitute` Disables automatic blackbox substitution. *This is the default.* See sample flows below.
- `-AUTO_substitute` Enables automatic blackbox substitution. With this option, the tool automatically performs synthesis using the corresponding synthesis script and substitutes the blackbox models with the synthesis models from the synthesis run.
- Note:** Before using this command, you must: specify the synthesis executable using the `SET HDL OPTION - synthesis_executable` command, read in the synthesis library for the elaborated cells using the `READ LIBRARY -append` command, and set the path to the synthesis licenses.
- `-LATENCY <integer>` Specifies the maximum number of seconds to wait before reading the wrapper netlist(s) and performing the blackbox substitution. This option is useful in cases where the wrapper netlist(s) might not be immediately available due to network delays.

## Conformal Extended Checks Reference Manual

### Command Reference

---

`-NODESIGNRule` Includes the `-no_design_rule` option into the synthesis script compile command to skip fixing design rule violations, such as `max_fanout`, `max_tran`, `max_cap`, etc.

`-SYNTHESIS_library <STANDARD | <library_list>>`

Specifies the synthesis libraries to use for synthesis. You can specify a list of space separated libraries, or you can use the `STANDARD` option to include `gtech.db` and `dw_foundation.sldb`.

If you do not specify this option, the tool uses the default set of libraries:

```
gtech.db dw01.sldb dw02.sldb dw03.sldb
dw04.sldb dw05.sldb dw06.sldb dw07.sldb
dw08.sldb standard.sldb
dw_foundation.sldb
```

These libraries are used in automatic *blackbox substitution* (enabled by the `-auto_substitute` option)

`-SYNTHESIS_Resource <filename>`

Specifies the resource file where the DW minPower component information is used. Single filename only.

`-TARGET_library <library_list>`

Specifies the user's target library.

`-TECH_file_library <tech_file_list>`

Specifies the user's technology files.

## Examples

### Sample module:

```
<<< des.v>>>
module design(clk, rst, cs, wr, rd_addr, wr_addr, din, dout);
input clk, rst, cs, wr;
input [2:0] rd_addr, wr_addr;
input [4:0] din;
output[4:0] dout;

DW_ram_r_w_s_dff #(5, 8, 0) ram (.clk(clk), .rst_n(rst), .cs_n(cs),
```

## Conformal Extended Checks Reference Manual

### Command Reference

---

```
.wr_n(wr), .rd_addr(rd_addr), .wr_addr(wr_addr), .data_out(dout), .data_in(din));
endmodule
```

### **Manual Blackbox Substitution**

The following illustrates the default flow for WRITE BLACKBOX WRAPPER (with automatic blackbox substitution disabled):

1. Specify that Conformal treat undefined cells as blackboxes.

```
> set undefined cell black_box
```

2. Read in the design, which contains our sample module.

```
> read design des.v
```

3. Write a wrapper file `dir/_DW_ram_r_w_s_dff_5_8_0.v` for blackbox module `DW_ram_r_w_s_dff_5_8_0`.

```
> write blackbox wrapper DW* -directory dir
```

```
> break
```

**Note:** This command also generates synthesis script template file `dir/RUN/synth.tcl`.

4. Use the `dir/RUN/synth.tcl` script with your own synthesis tool to generate `dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v`.

5. Read the newly created `dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v` file into the design.

```
> read design -append dir/RUN/*.g.v
```

6. Substitute the old module of blackbox instance `ram` with the new module `_DW_ram_r_w_s_dff_5_8_0_DW_ram_r_w_s_dff_5_8_0_0` (`dir/RUN/_DW_ram_r_w_s_dff_5_8_0.g.v:229`).

```
> substitute blackbox wrapper DW*
```

### **Automatic Blackbox Substitution**

The following illustrates the flow for WRITE BLACKBOX WRAPPER with automatic blackbox substitution enabled:

1. Specify that Conformal treat undefined cells as blackboxes.

```
> set undefined cell black_box
```

2. Specify the Design Compiler executable:

```
set hdl_option -synthesis_executable /grid/DC_INSTALL_DIRECTORY/latest/bin/
dc_shell-t
```

3. Read in the design, which contains our sample module.

```
> read design des.v
```

4. Enable automatic blackbox substitution for module `DW_ram_r_w_s_dff_5_8_0` and use only the `gtech.db` and `dw_foundation.sldb` libraries when running synthesis to generate the synthesis model:

```
write blackbox wrapper DW* -synthesis_library standard -auto_substitute
```

## **Related Command**

SET HDL OPTIONS

SUBSTITUTE BLACKBOX WRAPPER

## WRITE CDC CHECK

### WRite CDc Check

```
<-DIagnosis
 <-SOURCE_DATA | -DESTINATION_DATA
 | -MUX_enable | -SINGLE_bit_change
 >
 <from-instance> <to-instance>
 [| -SOURCE <clock_domain>]
 [| -DESTination <clock_domain>]
 [| -SYNC_RULE <rule_name>]
 [[-VCd] [<filename>] [-REPlace]]
| -PSL [-SINgle_vunit | -MULtiple_vunit]
 [| -SOURCE_DATA | -DESTINATION_DATA+
 | -MUX_enable | -SINGLE_bit_change
] ...
 [-SOURCE <-ALL | <clock_domain* ...> >]
 [-DESTination <-ALL | <clock_domain* ...> >]
 [-FROM <-ALL | -REG | -PI | -BBOx | <instance* ... > >]
 [-TO <-ALL | -REG | -PO | -BBOx | <instance* ... > >]
 [-ED | -PASS | -FAIL] ...
 [-FILE <filename> [-NOREPlace | -REPlace]]
>
(Verify Mode)
```

Saves the counter-example to a file in VCD format or writes out PSL assertions.

### Tcl Command

write\_cdc\_check

### Parameters

|              |                                                                                                                                                                                                                                                                                                                            |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -DIagnosis   | Saves the counter-example to a file in VCD format. Use this option after you diagnose a failed Functional CDC path (see <a href="#">DIAGNOSE CDC CHECK</a> .) You can specify a filename, or Conformal Extended Checks supplies a filename by default in the following format:<br><br>Vpx_<source ID>_<destination ID>.vcd |
| -SOURCE_DATA | Saves the results of the DIAGNOSE CDC CHECK command for the specified domain crossing path that failed the Source Data Functional CDC Check.                                                                                                                                                                               |



## Conformal Extended Checks Reference Manual

### Command Reference

---

|                             |                                                                                                                                                                                                                                                             |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -DESTINATION_DATA           | Saves the results of the DIAGNOSE CDC CHECK command for the specified domain crossing path that failed the Destination Data Functional CDC Check.                                                                                                           |
| -MUX_enable                 | Saves the results of the DIAGNOSE CDC CHECK command for the specified domain crossing path that failed the MUX Enable Functional CDC Check.                                                                                                                 |
| -SINGLE_bit_change          | Saves the results of the DIAGNOSE CDC CHECK command for the specified domain crossing path that failed the Single-bit Change Functional CDC Check.                                                                                                          |
| <from-instance>             | Saves the results of the DIAGNOSE CDC CHECK command for the clock domain crossing path that has the specified beginning.                                                                                                                                    |
| <to-instance>               | Saves the results of the DIAGNOSE CDC CHECK command for the clock domain crossing path that has the specified ending.                                                                                                                                       |
| -SOURCE <clock_domain>      | <p>Saves the results of the DIAGNOSE CDC CHECK command for the clock domain crossing path with the specified source domain.</p> <p>Use this option with multi-port latches to uniquely identify paths with the same from-instance and to-instance.</p>      |
| -DESTination <clock_domain> | <p>Saves the results of the DIAGNOSE CDC CHECK command for the clock domain crossing path with the specified destination domain.</p> <p>Use this option with multi-port latches to uniquely identify paths with the same from-instance and to-instance.</p> |

## Conformal Extended Checks Reference Manual

### Command Reference

---

|            |                  |                                                                                                                                   |
|------------|------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| -SYNC_RULE | <rule_name>      | Saves the results of the DIAGNOSE CDC CHECK command for the failing mux_enable domain crossing path with the specified rule_name. |
|            |                  | Use this option with a failed mux_enable CDC path to uniquely identify paths with the same from-instance and to-instance.         |
| -VCd       |                  | Saves the results of the DIAGNOSE CDC CHECK command to a VCD file.                                                                |
|            | <filename>       | Writes the diagnosis data to the specified file in VCD format.                                                                    |
| -REPlace   |                  | Replaces the existing file with the current diagnosis results.                                                                    |
| -PSL       |                  | Writes out PSL assertions.                                                                                                        |
|            | -SINgle_vunit    | Writes all PSL assertions in the same vunit.                                                                                      |
|            | -MULtiple_vunit  | Writes only one PSL assertion in each vunit.                                                                                      |
|            | SOURCE_DATA      | Writes PSL assertions for the Source Data Functional CDC Check results for the specified portion of the design.                   |
|            | DESTINATION_DATA | Writes PSL assertions for the Destination Data Functional CDC Check results for the specified portion of the design.              |
|            | MUX_enable       | Writes PSL assertions for the MUX Enable Functional CDC Check results for the specified portion of the design.                    |

## Conformal Extended Checks Reference Manual

### Command Reference

---

`-SOURCE <-ALL | <clock_domain* ... > >`

Writes PSL assertions for only those with the specified source clock domain.

`-ALL` uses all clock domains as the source.

`<clock_domain*>` uses the specified clock domains as the source. This accepts wildcards.

`-DESTination <-ALL | <clock_domain* ... > >`

Writes PSL assertions for only those with the specified destination clock domain.

`-ALL` uses all clock domains as the destination.

`<clock_domain*>` uses the specified clock domains as the destination. This accepts wildcards.

`-FROM <-ALL | -REG | -PI | -BBOX | <instance* ...>>`

Writes PSL assertions for only those that start from the specified key points.

`-ALL` uses all types of key points as the start of the paths. The type of key points includes registers (`-REG`), primary inputs (`-PI`), and blackbox outputs (`-BBOX`).

`instance` uses the specified instances as the start of the paths. This accepts wildcards. The instances can be registers, primary inputs, or blackbox outputs.

## Conformal Extended Checks Reference Manual

### Command Reference

---

`-TO <-ALL| -REG| -PO| -BBOx| <instance* ...>>`

Writes PSL assertions for only those that end at the specified key points.

`-ALL` uses all types of key points as the end of the paths. The type of key points includes registers (`-REG`), primary outputs (`-PO`), and blackbox inputs (`-BBOx`)

`instance` uses the specified instances as the end of the paths. This accepts wildcards. The instances can be registers, primary outputs, or blackbox inputs.

`-ED`

Writes PSL assertions for only those checks whose validation statuses are ED.

`-PASS`

Writes PSL assertions for only those checks whose validation statuses are PASS.

`-FAIL`

Writes PSL assertions for only those checks whose validation statuses are FAIL.

`-FILE <filename> [-NOREPlace | -REPlace]`

Specifies the name of the assertion file to be written.

`-NOREPlace` does not replace an existing file.

`-REPlace` specifies that if the output file exists, this replaces its contents.

## Examples

```
diagnose cdc check -source_data in2a_reg U_syn/out1_reg \
-source clkA -destination u_pll/div_by_2
write cdc check -diagnosis -source_data in2a_reg U_syn/out1_reg \
-source clkA -destination u_pll/div_by_2
write cdc check -psl source_data -source clkA -destination clkB \
-ed -fail -file psl_assert.v
```

## **Related Commands**

DIAGNOSE CDC CHECK

VALIDATE

## WRITE DESIGN

### WRite DEsign

```
<filename>
[| -ALL | -Module <module_name> | -BBOX | -Used]
[-Library]
[-GZip]
[-REPlace]
(Setup / Verify Mode)
```

Generates a file containing the current design in Verilog format to examine how Conformal translates the HDL into logical primitives.

**Tilde:** Use the tilde "~" character in the file path to replace the path to the user login home directory.

For information on how this command handles files with encrypted/protected modules, refer to the "IP Protection" section of the [READ DESIGN](#) command.

## Tcl Command

```
write_design
```

## Parameters

<filename>	Writes the design to the specified file.
-ALL	Writes out all modules that are stored in the design space.
-Module <module_name>	Writes out the specified module that is stored in the design space.
-BBOX	Writes out all of the empty module descriptions of blackboxes in the design.  By default, the command writes out the design tree of the root module, excluding modules in the library space.
-Used	Writes out the specified modules and all the referenced modules, including modules both in the design space and library spaces.
-Library	Includes the library information.

## Conformal Extended Checks Reference Manual

### Command Reference

---

-GZip	Writes out the design in gzip format.
-REPlace	If a file with the same name exists, this replaces it with the current design.

### Related Command

READ DESIGN

## WRITE DIAGNOSIS DATA

### WRite Diagnosis Data

```
[-VCd | -VErilog]
[<filename>]
[-REplace]
(Verify Mode)
```

Generates a file containing diagnosis data. You can specify a filename. Or Conformal Extended Checks supplies a filename by default in the following format:

```
Vpx_<design_root_module_name>_<Conformal_property_name>.vcd
```

Use this command after you diagnose a false property. (Refer to [DIAGNOSE STATIC PROPERTY](#).)

**Tilde:** Use the tilde "~" character in the file path to replace the path to the user login home directory.

### Tcl Command

```
write_diagnosis_data
```

### Parameters

-VCd	Writes diagnosis data to a VCD file. <i>This is the default.</i>
-VErilog	Converts counter-example vectors to a Verilog testbench.
<filename>	Writes the design to the specified file.
-REplace	Replaces an existing file with the current data.

### Related Command

[DIAGNOSE STATIC PROPERTY](#)



## WRITE LIBRARY

### WRite LIBrary

```
<filename>
[-Module <module_name>]
[-Verilog]
[-GZip]
[-REPlace]
(Setup / Verify Mode)
```

Generates a file containing the library. *The default is to generate the library as functional Verilog model descriptions.*

Use this command to examine how Conformal extracts complex UDP library models.

**Tilde:** Use the tilde "~" character in the file's path to replace the path to the user login home directory.

### Tcl Command

```
write_library
```

### Parameters

<filename>	Writes the library to the specified file.
-Module <module_name>	Writes out the specified module that is stored in the library space.
-Verilog	Generates the library in Verilog format. <i>This is the default.</i>
-GZip	Writes the library in a compressed gzip format.
-REPlace	If a file with the same name exists, this replaces it with the current library.

### Related Command

READ LIBRARY

## WRITE RULE CHECK

### WRite RuLe Check

```
<filename>
[-COMplete | -HIDden | -NOHidden]
[-FILtered_out | -NOFILtered_out | -WAIved | -NOWaived]
[-DEsign | -Llibrary]
[-NOREplace | -REPlace]
(Setup / Verify Mode)
```

Writes the rule violations to a rule file. Use this command the first time you run a session. For later runs, exclude the violations already flagged with the `read rule check -exclude <filename>` command.

Use the tilde character (~) to shorten the path of the file.



#### Tip

After waiving many occurrences, save them with the following command:

```
write rule check waived.rules -waived -replace
```

**Note:** A rule occurrence is "hidden" if it is filtered out (excluded by one or more rule filters), or if it is waived, or both

## Tcl Command

```
write_rule_check
```


## Parameters

<filename>	Specifies the name of the output file.
-NOHidden	Reports only occurrences that are not hidden. <i>This is the default.</i>
-HIDden	Reports only occurrences that are hidden.
-COMplete	Reports all occurrences regardless whether they are hidden or not.
-FILtered_out	Reports only occurrences excluded by filters (regardless whether they are waived or not).

## Conformal Extended Checks Reference Manual

### Command Reference

---

-NOFILtered_out	Reports only occurrences not excluded by filters (regardless whether they are waived or not).
-WAIved	Reports only occurrences that are waived (regardless whether they are excluded by filters or not).
-NOWaived	Reports only occurrences that are not waived (regardless whether they are excluded by filters or not).
<div> <b>Tip</b></div> <p>After waiving many occurrences, save them with the following command:</p> <pre>write rule check waived.rules -waived -replace</pre>	
-DEsign	Writes only design rule check violations. If you do not specify -design or -library, Conformal writes rule check violations from both designs and libraries.
-Llibrary	Writes only library rule check violations. If you do not specify -design or -library, Conformal writes rule check violations from both designs and libraries.
-NOReplace	Does not replace the data if the specified filename already exists. The Conformal software issues a warning if you use a filename that already exists. <i>This is the default.</i>
-REPlace	If the specified filename exists, this replaces its contents

## Examples

In the following example, the second `report rule check` will not report any rules.

```
read design des.v
write rule check rule.des -replace
read design des.v -replace
report rule check -verbose
read rule check rule.des -exclude
report rule check -verbose
```

## Related Command

[READ RULE CHECK](#)

[REPORT RULE CHECK](#)

# Conformal Extended Checks Reference Manual

## Command Reference

---

---

## Modeling Rule Checks

---

Definitions for modeling rules are explained in the following sections with simple demonstration examples. This chapter includes the following topics:

- [Introduction to Modeling Messages](#) on page 478
- [Modeling Rule Priority Levels](#) on page 479
- [Rule Categories](#) on page 481

## Introduction to Modeling Messages

Modeling messages indicate any modeling errors encountered during the analysis and modeling of the design. These errors are caused when the design is not initialized properly or when the design has problems that may lead to a mismatch between the logic behavior and the electrical behavior. Modeling Rule Checks are active as the system mode changes from Setup to Verify.

Modeling messages indicate any modeling warnings encountered during the analysis and modeling of the design.

## Violation Severity Levels

There are three levels of severity for rule violations. The severity levels are listed below from the most serious to the least serious:

- **Error:**  
Conformal Extended Checks may not allow you to begin verification until you resolve the error.
- **Warning:**  
Conformal Extended Checks allows you to begin verification; however, it warns you of potential errors in the design.
- **Note:**  
Conformal Extended Checks allows you to begin verification; however, it flags potential errors in the design.

You may view a summary or expanded report of all rule violations with one of the following commands:

- `report rule check -modeling -verbose`
- `report rule check -modeling -summary`

Additionally, you may specify the category of rules you would like to view with the following command. Use the asterisk (\*) following a prefix to list rules of the specified category.

```
REPort RULE Check
[-All |-RTL |-MODEling | rule_name*...]
[-SUMmary |-Verbose] [-HELP]
```

For example:

```
report rule check STRC* -summary
```

## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

To view information for a specific message, use the `HELP` command followed by the message name.

**HELp** [message\_name]

For example:

```
help CLK1
```

## Modeling Rule Priority Levels

The following table prioritizes Modeling Rules according to the attention they require from you and their effect on proofs.

Rules	Action Required
<b>Top Priority</b>	You must resolve these errors before proceeding to the next step of the session.
ACX2	
CLK1	
CNST1	
FSM1	
FSM2	
FSM3	
INIT2	
STRC3	

## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

Rules	Action Required
Second Priority	These violations are affecting design logic (in proving them true or false) during proofs. Design knowledge is needed in order to resolve them properly.
ACX1	
CLK2	
CLK3	
FSM4	
INIT3	
STRC1.1	
STRC1.2	
STRC2	
STRC4	
STRC5.1	
STRC5.2	
STRC5.3	
<b>Third Priority</b>	You may take these violations as warnings.
CLK4	
INIT1	
INIT4	
STRC6.1	
STRC6.2	



## Rule Categories

This chapter includes the following rule categories:

- [Structural Rules](#) on page 482
- [Clock Definition Rules](#) on page 499
- [Initialization Rules](#) on page 505
- [Finite State Machine Rules](#) on page 511
- [Asynchronous Clock Domain Crossing](#) on page 517
- [Constraints](#) on page 525

The remainder of this chapter is organized into sections by rule categories. Each section lists all the rules and warning messages for the specified category. Then, each rule message is further defined.

## Structural Rules

This category of rules checks for design structural errors. The following lists the Structural (STRC) rule checks:

- [STRC1.1](#) on page 483
- [STRC1.2](#) on page 485
- [STRC2](#) on page 486
- [STRC3](#) on page 489
- [STRC4](#) on page 492
- [STRC5.1](#) on page 493
- [STRC5.2](#) on page 494
- [STRC5.3](#) on page 495
- [STRC6.1](#) on page 496
- [STRC6.2](#) on page 497
- [STRC7](#) on page 498

## STRC1.1

### Message

Strong combinational loop detected

### Default Severity

Warning

### Description

The design includes a combinational loop that does not contain any weak devices. This could cause signals to oscillate and reflects a potential error in the design.

If this reflects design intent, you can ignore it. It will not affect verification. If this does not reflect design intent, correct the design. First, generate a report on the combinational loop using the `REPORT PATH` command:

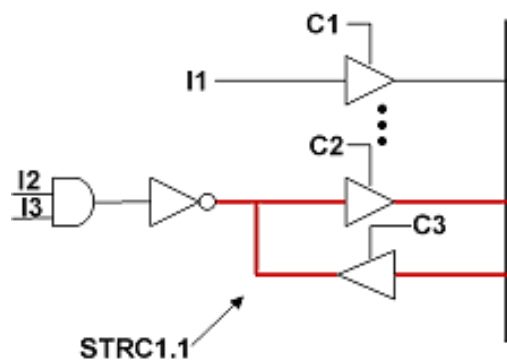
```
report path -loop
```

The presence of this violation does not automatically indicate there is a problem in the design. For example, a combinational loop may be in a false path.

**Note:** Any proofs containing a functional loop may generate inconsistent results if the loop oscillates.

## Example

In the following figure, a strong combinational feedback loop exists if C2 and C3 can be active at the same time.



## STRC1.2

### Message

Weak combinational loop detected

### Default Severity

Warning

### Description

The design includes a combinational loop that contains at least one weak device. This could cause signals to oscillate and reflects a potential error in the design.

If this reflects design intent, you may ignore it. It will not affect verification. If this does not reflect design intent, correct the design. First, generate a report on the combinational loop using the `REPORT PATH` command:

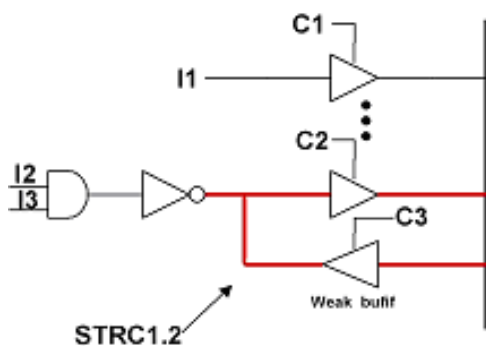
```
report path -loop
```

The presence of this violation does not automatically indicate there is a problem in the design. For example, a combinational loop may be in a false path.

**Note:** Any proofs containing a functional loop may generate inconsistent results if the loop oscillates.

### Example

In the following figure, a strong combinational feedback loop exists if C2 and C3 can be active at the same time.



## STRC2

### Message

Undriven fanin detected

### Default Severity

Warning

### Description

The design includes undriven pins. They are the subset of the floating pins of the design that are actually driving some logic.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with one of the following `REPORT RULE CHECK` commands:

- `report rule check STRC2 -summary`
- `report rule check STRC2 -verbose`

Use the "Browse Gate" (schematic view) in the GUI mode to graphically pinpoint the source of the violation and the Source Code Browser to examine the origin. Modify the design accordingly.

An undriven fan-in remains undriven even if you SET it to a value with the `SET UNDRIVEN SIGNAL` command. Thus, the STRC2 message remains. However, an undriven fan-in becomes driven when you TIE it to a value with the `ADD TIED SIGNAL` command. Thus, Conformal Extended Checks will not generate the STRC2 message.

The presence of an undriven fan-in may not indicate problems with the design; but this warning points to *potential* problem areas.



You must resolve undriven signals for assertions or clocks. Floating signals will not allow proper sequential verification. They generate a vacuous proof.

## Example

The following example shows a Verilog code with a floating signal. Note that `x4` is not being driven by anything, but it is driving the D input of a flip flop. (See line 20, in bold.) This generates an STRC2 message as well as an STRC3 message.

```
1. module test (in1, in2, in3,
2. reset_bar, clk,
3. en1, en2, en3,
4. out1);
5.
6. input in1, in2, in3;
7. input en1, en2, en3;
8. input reset_bar, clk;
9. output out1;
10.
11. wire x1, x2, x3;
12. wire dq1, dq2, dq3;
13.
14. IV U_inv1 (.Z(x1), .A(en1));
15. IV U_inv2 (.Z(x2), .A(en2));
16. IV U_inv3 (.Z(x3), .A(en3));
17.
18. FD2 U_dff1 (.Q(dq1), .D(x1), .CP(clk), CD(reset_bar));
19. FD2 U_dff2 (.Q(dq2), .D(x2), .CP(clk), .CD(reset_bar));
20. FD2 U_dff3 (.Q(dq3), .D(x4), .CP(clk), .CD(reset_bar));
21.
22. BTS4 U_trist1 (.Z(out1), .A(in1), .E(dq1));
23. BTS4 U_trist2 (.Z(out1), .A(in2), .E(dq2));
24. BTS4 U_trist3 (.Z(out1), .A(in3), .E(dq3));
```

## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

1. `module test (in1, in2, in3,`
- 25.
26. `endmodule`



## STRC3

### Message

Input to DFF/DLAT is always high-impedance (Z)

### Default Severity

Error

### Description

The design includes signals at the inputs of the flip-flop or latch that are always high-impedance (Z). These may be undriven (that is, disconnected) or connected to logic such as a constantly disabled buffer. You must resolve the situation prior to verification.

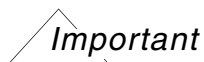
Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with one of the following `REPORT RULE CHECK` commands:

- `report rule check STRC3 -summary`
- `report rule check STRC3 -verbose`

Use the "Browse Gate" (schematic view) in the GUI mode to graphically pinpoint the source of the violation and the Source Code Browser to examine the origin. Modify the design accordingly.

This message can be caused by an undriven input to a flip-flop or latch. Thus, Conformal Extended Checks also reports STRC2. Or it can be caused by an input that is tied to Z (for example, tied to the output of a bufif1 when enable is 0.) In this case, Conformal Extended Checks does not report STRC2.

To remove this message, either physically connect the input to the DFF/DLAT to a non-Z value, or use the `ADD TIED SIGNAL` or `SET UNDRIVEN SIGNAL (non-Z)` command.



You must resolve undriven DFFs and DLATs prior to verification.

## Example

The following example shows a Verilog code with a floating signal. Note that `x4` is not being driven by anything, but it is driving the D input of a flip flop. (See line 20, in bold.) This generates an STRC2 message as well as an STRC3 message.

```
1. module test (in1, in2, in3,
2. reset_bar, clk,
3. en1, en2, en3,
4. out1);
5.
6. input in1, in2, in3;
7. input en1, en2, en3;
8. input reset_bar, clk;
9. output out1;
10.
11. wire x1, x2, x3;
12. wire dq1, dq2, dq3;
13.
14. IV U_inv1 (.Z(x1), .A(en1));
15. IV U_inv2 (.Z(x2), .A(en2));
16. IV U_inv3 (.Z(x3), .A(en3));
17.
18. FD2 U_dff1 (.Q(dq1), .D(x1), .CP(clk), CD(reset_bar));
19. FD2 U_dff2 (.Q(dq2), .D(x2), .CP(clk), .CD(reset_bar));
20. FD2 U_dff3 (.Q(dq3), .D(x4), .CP(clk), .CD(reset_bar));
21.
22. BTS4 U_trist1 (.Z(out1), .A(in1), .E(dq1));
23. BTS4 U_trist2 (.Z(out1), .A(in2), .E(dq2));
24. BTS4 U_trist3 (.Z(out1), .A(in3), .E(dq3));
```

## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

1. `module test (in1, in2, in3,`
- 25.
26. `endmodule`

## STRC4

### Message

Set/Reset port of the DFF/DLAT is always enabled

### Default Severity

Warning

### Description

The design includes floating signals at the inputs of DFFs and DLATs.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with one of the following `REPORT RULE CHECK` commands:

- `report rule check STRC4 -summary`
- `report rule check STRC4 -verbose`

## STRC5.1

### Message

Clock port of the DFF is tied

### Default Severity

Warning

### Description

The design includes a clock signal that is tied to low or high logic.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with one of the following `REPORT RULE CHECK` commands:

- `report rule check STRC5.1 -summary`
- `report rule check STRC5.1 -verbose`

## STRC5.2

### Message

Clock port of the DLAT is always disabled

### Default Severity

Warning

### Description

The design includes a clock signal of a DLAT that is tied to ground.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with the `REPORT RULE CHECK` command:

- `report rule check STRC5.2 -summary`
- `report rule check STRC5.2 -verbose`

## STRC5.3

### Message

Clock port of the DLAT is always enabled (transparent DLAT)

### Default Severity

Note

### Description

The design includes a clock signal of a DLAT that is tied to high with disabled Set and Reset.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list them with one of the following `REPORT RULE CHECK` commands:

- `report rule check STRC5.3 -summary`
- `report rule check STRC5.3 -verbose`

## STRC6.1

### Message

Data port of the DFF is tied

### Default Severity

Warning

### Description

The design includes a DFF whose data port is tied to a 1 or 0.



## STRC6.2

### Message

Data port of the DLAT is tied

### Default Severity

Warning

### Description

The design includes a DLAT whose data port is tied to a 1 or 0.

## STRC7

### Message

Set/Reset port of DFF/DLAT is driven by a multi-input logic gate

### Default Severity

Warning

### Description

The tool will trace backward from the asynchronous set or reset port of DFF or DLAT and flag the violation if it reach a multi-input logic gate that has more than one possible sensitization input.

## Clock Definition Rules

In a design with ill-defined clocks, Conformal Extended Checks detects when sequential parts of the circuit may not trigger or may trigger improperly.

- [CLK1](#) on page 500
- [CLK2](#) on page 501
- [CLK3](#) on page 503

# CLK1

## Message

Undefined clock pin

## Default Severity

Warning

## Description

The design includes undeclared clock pins. To ensure proper design verification, always specify wave forms or assign values to clock pins. If you do not take this precaution, Conformal Extended Checks assumes that at every time unit, it may assign a value to this pin.

If there is only one clock in the design, automatically declare the clock with default values. You may specify a waveform with a default waveform (0 or 1):

- Default waveform 0 is 0 1 1 2
- Default waveform 1 is 1 1 1 2

Use the `ADD CLOCK` command to declare clock pins. Or, use the GUI mode *Setup – Clock* window in the Setup system mode.

Alternatively, use the `ADD PIN CONSTRAINT` command to assign a value of "1" (to switch on) or "0" (to switch off) parts of the design during verification.

## CLK2

### Message

DFF/DLAT with gated clock

### Default Severity

Note

### Description

The design includes a clock waveform that cannot be fully determined after propagating the clock waveforms from the clock pins to the clock input.

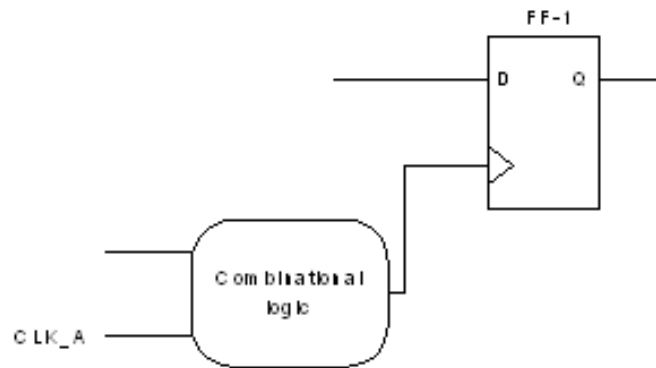
This rule check is an unspecified clock rule. It relates to a set of flip-flops and latches whose clock inputs are gated. In these cases, clock pins were either defined by the `ADD CLOCK` command or found in the fan-in cone of the clock inputs by the Undriven Fanin Rule check (STRC2) described above.

In the GUI mode, access *Tools – Clock Fanout Schematic* to graphically pinpoint the source of the violation.

To avoid this violation, you must ensure that initialization of this circuit assigns proper values to the other flip-flops of the design. This avoids propagating undefined values to the clock pin of flip-flops or latches. *Note: This explanation assumes that you do not intend to test a situation that propagates undefined values.*

### Example

The following figure shows an example of a violation of CLK2.



## CLK3

### Message

No clock waveform defined for DFF/DLAT

### Default Severity

Warning

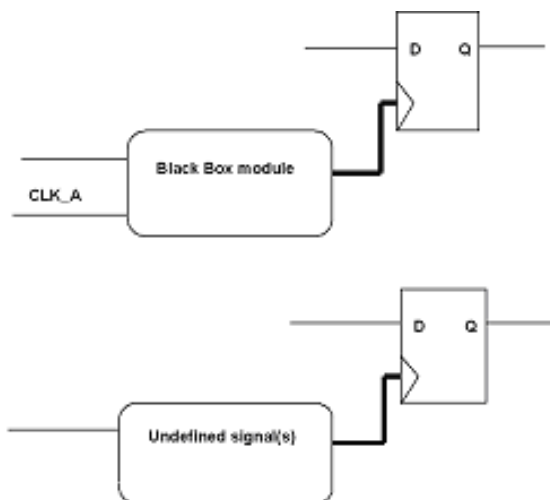
### Description

The design includes clock inputs for a set of flip-flops or latches that are not driven by any clock pin in the fan-in cone. That is, no clock pin has been defined by the `ADD CLOCK` command, nor found by the Conformal Extended Checks Undefined Clock Pin Rule (CLK1) checking in the fan-in cone of the clock inputs.

Use the `REPORT CLOCK -candidate` command to learn more about the violation, and declare additional clock pins with the `ADD CLOCK` command.

### Example

This rule check is an unspecified clock rule. In the following figure, the counter DFF chain is identified as in a single clock domain and is counted as a violation of rule CLK3.



## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

Use the following command to learn more about the violation:

```
report clock -candidate[CLK3]
```

And declare additional clock pins with the `ADD CLOCK` command.



## Initialization Rules

In order to properly initialize a design and perform a valid verification, Conformal Extended Checks uses the initialization rules to check for certain conditions in the design. The following table lists the Initialization (INIT) rule check numbers and messages:

---

<b>Rule Number</b>	<b>Message</b>
<u>INIT1</u>	<u>Inconsistent initial state setting</u>
<u>INIT2</u>	<u>Initial states are unjustifiable</u>
<u>INIT3</u>	<u>Uninitialized DFF/DLAT</u>
<u>INIT4</u>	<u>Unused initial state value from VCD</u>

---

# INIT1

## Message

Inconsistent initial state setting

## Default Severity

Warning

## Description

An INIT1 violation occurs when any part of circuit has an inconsistent initial state setting. An inconsistent initial state can occur in one of two ways:

1. Two different initialization commands try to initialize a state element to different values. In this case, the latter value overrides the previous, and Conformal Extended Checks issues the INIT1 warning.
2. After applying initialization commands, Conformal Extended Checks performs combinational propagation to determine if previously initialized values (initialized by the initialization commands) are changed by the combinational propagation. If so, the new value overrides the old, and Conformal Extended Checks issues the INIT1 warning message.

Inconsistencies in initial value settings should be resolved as they might cause the Conformal Constraint Designer to produce incorrect results. To resolve this problem, return to the Setup system mode and delete the initialization command that caused the violation.

## INIT2

### Message

Initial states are unjustifiable

### Default Severity

Error

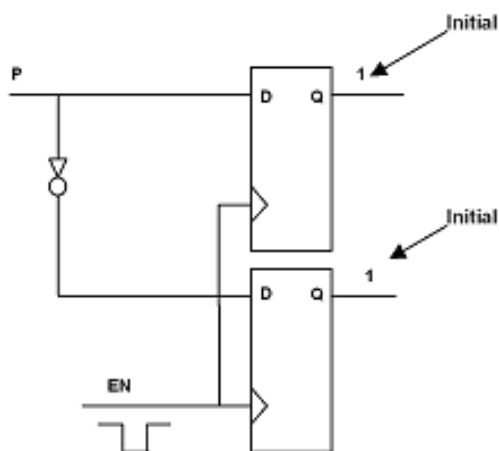
### Description

The design includes initial output values of flip-flops or latches that are not attainable based on the initial input values.

INIT2 is usually caused by a combination of conflicting initialization commands and/or constraints. Thus, during diagnosis with the GUI, the INIT2 schematic contains all the gates directly involved in the warning message. The schematic also displays the fan-ins of those gates (two to three levels deep).

### Example

The following figure shows an example of an INIT2 rule check violation.



## INIT3

### Message

Uninitialized DFF/DLAT

### Default Severity

Warning

### Description

The design includes a flip-flop or latch that is not initialized. In this case, the output port of the flip-flop or latch takes the assumed "X" value.

Use the `REPORT GATE` command to report which flip-flops and latches were not initialized correctly.

**Note:** Conformal automatically assigns ID numbers. They can differ from one version to another. Always use the full path in dofiles and any time you rerun a design with a different Conformal version.

You can use the initial state diagnosis for debugging to understand why registers did not get initialized (INIT3 violation). By default, the diagnosis information is not stored, so you must save the diagnosis information when performing design initialization by running the following command:

```
read initial state <filename> -save_diagnosis_info
```

With the diagnosis information is saved, you can diagnose in GUI mode or non-GUI mode. Using the following methods, you should be able to understand why registers did not get initialized (INIT3 violation).

In Non-GUI Mode:

1. Use the the following command to identify which registers have not been initialized:

```
report rule check init3 -ver
```

2. Then use the following command to print the value of `gate <id>` at time `<time-unit>` of the initialization:

```
report gate <id> -init <time-unit>
```

In GUI Mode:

## Conformal Extended Checks Reference Manual

### Modeling Rule Checks

---

1. Open the Modeling Rule Manager from the *Tools* menu.
2. Click on the *Initialization* tab
3. Click rule `INIT3`
4. Click on the `INIT3` rule violation that you want to diagnose
5. Right-click to open the pop-up menu and select *Open Schematics*.
6. In the Schematics window, select *Preferences – Show Initial Value*.

Here you can view the gate values at different times of the initialization by selecting the proper value in the *Time* field.

## INIT4

### Message

Unused initial state value from VCD

### Default Severity

Warning

### Description

There is no corresponding hierarchical pattern in the design matching the extracted hierarchical pattern from the VCD.

Learn more about where these violations occurred by viewing the Modeling Rule Manager in the GUI mode or list occurrences (in detail) with the `REPORT RULE CHECK` command:

```
report rule check INIT4 -verbose
```

## Finite State Machine Rules

This set of rules checks for full and valid initialization for FSMs. FSM properties can only be added to the Prove list if they do not violate the following FSM rule checks:

- [FSM1](#) on page 512
- [FSM2](#) on page 513
- [FSM3](#) on page 514
- [FSM4.1](#) on page 515
- [FSM4.2](#) on page 516

## FSM1

### Message

Initial state of the FSM is not fully specified

### Default Severity

Error

### Description

The FSM is not fully initialized. Thus, Conformal Extended Checks will not prove the FSM property that has violated this rule.



## FSM2

### Message

Initial state of the FSM is invalid

### Default Severity

Error

### Description

The FSM has an invalid initial state. Thus, Conformal Extended Checks will not prove the FSM property that has violated this rule.

## **FSM3**

### **Message**

Input to state variable/DFF of FSM is always high-impedance (Z)

### **Default Severity**

Error

### **Description**

The input to the FSM state variable is always high-impedance (Z). Thus, Conformal Extended Checks will not prove the FSM property that has violated this rule.

## FSM4.1

### Message

FSM terminal state

### Default Severity

Warning

### Description

A state in an FSM does not transition to another state.

## FSM4.2

### Message

All outgoing transitions from FSM state are reset transitions

### Default Severity

Warning

### Description

This FSM state has all the outgoing transitions that can happen only by the reset signal. The only way that this state machine can move forward is by exiting the reset.

## Asynchronous Clock Domain Crossing

The purpose of clock domain assignment is to identify the clock domain of each state element's clock port. Conformal Extended Checks flags any missing or conflicting information with a warning or error message, respectively. Since Conformal Extended Checks does not have enough information to perform CDC Checks on unknown and error clock domains, it ignores all paths involving ACX messages. Thus, it is important to resolve these messages before attempting to execute CDC Checks.

**Note:** These rule checks are performed during clock domain analysis.

To correct these ACX Modeling Rule violations, you might need to do any one or more of the following:

- Declare design constraints
- Declare or correct clock domain associations
- Modify clock domain analysis and extraction rules

The following table lists the Asynchronous Clock Domain Crossing (ACX) rule checks:

- [ACX1](#) on page 518
- [ACX2](#) on page 519
- [ACX3](#) on page 520
- [ACX4](#) on page 521
- [ACX5](#) on page 522
- [ACX6](#) on page 523
- [ACX7](#) on page 524

## ACX1

### Message

Clock port of DFF/DLAT is annotated as an unknown domain

### Default Severity

Warning

### Description

The user-declared clock domains cannot be propagated to the signal in question and, if performing automatic clock source extraction, Conformal Extended Checks cannot determine the clock domain of this signal. This rule check is performed during clock domain analysis.

Find the undefined clocks and define them to avoid missing any potential CDC problems:

```
report rule check ACX1 -verbose
report clock domain -clock_port -unknown
```

In the GUI mode, go to the Modeling Rule Manager and use the pop-up menu to access a schematic view of the instance in question.

After you identify the problem, return to the Setup mode and define the clock, which effectively declares the clock domain for the CDC check. You might also need to modify the present clock domain analysis rule so that Conformal Extended Checks properly recognizes the clock domain at the clock port of the D flip-flop or D-latch that was in question. Additionally, it could be helpful to declare the design constraints to allow optimal performance by clock domain analysis (for example, during clock domain extraction).

## ACX2

### Message

Clock port of DFF/DLAT is annotated as an error

### Default Severity

Error

### Description

The design includes a domain ID with the assignment *conflict error*, which is caused by the propagation rules and certain input domain assignment combinations. For example, if there is a logic gate that has two inputs with different assigned clock domains, then the output of this logic gate will have an assignment conflict error.

This rule check is performed during clock domain analysis.

Often, this violation coincides with an ACX1 violation. And in these cases, when you resolve the ACX1 violation, you may find that the ACX2 violation is also resolved. However, if the ACX2 violation remains, use the `SET CLOCK DOMAIN RULE` command to specify how the clock domain should be defined and extracted.

## ACX3

### Message

Clock domain of reset port of DFF/DLAT is unknown

### Default Severity

Warning

### Description

The Conformal Extended Checks cannot determine the clock domains of signals driving the asynchronous reset port of the instance in question.

This rule check is performed during clock domain analysis.

Find the undefined clocks and define them to avoid missing any potential CDC problems:

```
report rule check ACX3 -verbose
```

In the GUI mode, go to the Modeling Rule Manager and use the pop-up menu to access a schematic view of the instance in question.

After you identify the problem, return to the Setup mode and define the clock, which effectively declares the clock domain for the CDC check. You might also need to modify the present clock domain analysis rule so that Conformal Extended Checks properly recognizes the clock domains of signals driving the asynchronous reset port of the instance in question.



## ACX4

### Message

Clock domain of set port of DFF/DLAT is unknown

### Default Severity

Warning

### Description

The Conformal Extended Checks cannot determine the clock domains of signals driving the asynchronous set port of the instance in question.

This rule check is performed during clock domain analysis.

Find the undefined clocks and define them to avoid missing any potential CDC problems:

```
report rule check ACX4 -verbose
```

In the GUI mode, go to the Modeling Rule Manager and use the pop-up menu to access a schematic view of the instance in question.

After you identify the problem, return to the Setup mode and define the clock, which effectively declares the clock domain for the CDC check. You might also need to modify the present clock domain analysis rule so that Conformal Extended Checks properly recognizes the clock domains of signals driving the asynchronous set port of the instance in question.

## **ACX5**

### **Message**

Clock domain of PI port is unknown

### **Default Severity**

Warning

### **Description**

The input port is not assigned to any clock domain and hence is unknown domain. Assign a clock domain if PI-REG CDC checks are needed.

This rule check is performed during clock domain analysis.

## ACX6

### Message

Clock domain of PO port is unknown

### Default Severity

Warning

### Description

The output port is not assigned to any clock domain and hence is unknown domain. Assign a clock domain if REG-PO CDC checks are needed.

This rule check is performed during clock domain analysis.

## ACX7

### Message

Clock domain of PIO port is unknown

### Default Severity

Warning

### Description

The I/O port is not assigned to any clock domain and hence is unknown domain. Assign a clock domain if PI-REG and PO-REG CDC checks are needed.

This rule check is performed during clock domain analysis.

## Constraints

Conformal Extended Checks looks for design overconstrained errors.

The following lists the overconstrained rule check:

- CNST1 on page 526

## CNST1

### Message

System is over-constrained

### Default Severity

Error

### Description

The design includes too many constraints to a circuit.

It is possible for a designer to insert either too few or too many constraints to a circuit. In the case of the latter, overconstrained specifications occur when two conflicting assignments to the same variable are present during the same time frame, or when contradicting constraints have been assigned. In this case, the proof yields meaningless results because certain conditions would never occur due to the specified constraints.

Remove the improper pin constraint with the DELETE PIN CONSTRAINTS command:

**Note:** CNST1 violations must be resolved before you proceed to proving the circuit's properties.

---

## Tcl Command Entry Mode Support

---

Describes the describes the Encounter<sup>®</sup> Conformal<sup>®</sup> Tcl commands. The commands are presented in alphabetical order.

- [add\\_attribute](#) on page 529
- [add\\_to\\_collection](#) on page 530
- [append\\_to\\_collection](#) on page 531
- [cfm\\_is\\_gui\\_mode](#) on page 532
- [compare\\_collection](#) on page 533
- [copy\\_collection](#) on page 534
- [delete\\_attribute](#) on page 535
- [echo\\_result](#) on page 536
- [encrypt](#) on page 537
- [exit](#) on page 538
- [find](#) on page 539
- [find\\_cfm](#) on page 542
- [foreach\\_in\\_collection](#) on page 545
- [get\\_current\\_module](#) on page 546
- [get\\_fanin](#) on page 547
- [get\\_fanout](#) on page 550
- [get\\_handle\\_type](#) on page 553
- [get\\_instances](#) on page 554
- [get\\_license\\_mode](#) on page 555

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

- [get\\_module\\_definition](#) on page 556
- [get\\_names](#) on page 557
- [get\\_nets](#) on page 558
- [get\\_parent](#) on page 559
- [get\\_pins](#) on page 560
- [get\\_ports](#) on page 561
- [get\\_primitive\\_type](#) on page 562
- [get\\_property](#) on page 563
- [get\\_relative\\_path](#) on page 565
- [get\\_root\\_module](#) on page 566
- [get\\_supply\\_sets](#) on page 567
- [get\\_version\\_info](#) on page 569
- [index\\_collection](#) on page 570
- [help](#) on page 571
- [ncdecrypt](#) on page 572
- [objtype](#) on page 573
- [query\\_collection](#) on page 574
- [query\\_state](#) on page 575
- [redirect](#) on page 577
- [remove\\_from\\_collection](#) on page 578
- [report\\_attribute](#) on page 579
- [set\\_current\\_module](#) on page 580
- [sizeof\\_collection](#) on page 581
- [sort\\_collection](#) on page 582
- [tcl\\_set\\_command\\_name\\_echo](#) on page 583
- [unset\\_attribute](#) on page 584
- [usage](#) on page 585



## add\_attribute

```
add_attribute
 <name>
 <-object_type objtypes>
 [-value_type <valtypes>]
 [-multiple | -single]
 [-default <defvalue>]
 [-description <desc>]
 [-hidden]
```

Creates a user-defined attribute for the specified design objects.

### Parameters

<name>	Name of the attribute
<-object_type objtypes>	List of design objects to which the attribute will be defined
[-value_type <valtypes>]	Restricts the attribute's value to the specified object types (includes native Tcl types: int, float, bool, string, pin, and so on).
-multiple   -single	Specifies whether only a single value or multiple values are allowed. Default is multiple.
-default <defvalue>	Specifies the default value for the attribute. Default is an empty string.
-description <desc>	Specifies a short description for the attribute.
-hidden	Hides the attribute in that it is not listed when the <code>get_attribute</code> or <code>set_attribute -all/GUI</code> commands are used. It will be listed if the attribute name explicitly given.

## add\_to\_collection

```
add_to_collection <base_collection>
 <collection | object_list >
 [-unique]
```

### Description

Adds objects to a collection, resulting in a new collection.

### Parameters

<base_collection>	Specifies the base collection of objects to which objects will be added. This object will be copied, resulting in a new collection, and the specified objects will be added to the new collection.
<collection   object_list>	Specifies a collection or object list to add to the new collection.
-unique	Removes duplicate objects from the new collection.

### Example

```
TCL_LEC> set a [find_cfm -instance * -collection]
collection_0
TCL_LEC> set b [find_cfm -instance u* -collection]
collection_1
TCL_SETUP> set c [add_to_collection $a $b]
collection_2
TCL_SETUP> query_collection $c
pmu_u_gsw ul u_gsw ul
TCL_SETUP> set c [add_to_collection $a $b -unique]
collection_3
TCL_SETUP> query_collection $c
pmu_u_gsw ul
```

## append\_to\_collection

```
append_to_collection <var>
 <collection | object_list>
 [-unique]
```

### Description

Appends objects to a collection.

### Parameters

<var>	Specifies a variable name. The objects matching the specified collection or object list will be added to the collection referenced by this variable.
<collection   object_list>	Specifies a collection or list of objects to add.
-unique	Remove duplicate objects from the resulting collection. By default, duplicate objects are not removed.

### Example

```
TCL_SETUP> set b [find -instance u* -collection]
collection_1
TCL_SETUP> append_to_collection z $b
TCL_SETUP> query_collection $z
u_gsw u1
```

## **cfm\_is\_gui\_mode**

`cfm_is_gui_mode`

Returns 0 if the tool is in command-line mode; 1 if it is in GUI mode.

## compare\_collection

**compare\_collection** <collection1> <collection2>  
[-order\_dependent]

### Description

Compares two collections. Returns a value of 0 if all the objects in both collections are the same; returns 1 if the values in both collections are not the same.

### Parameters

<collection1>	Specifies the first collection
<collection2>	Specifies the second collection
-order_dependent	Specifies that the order of the objects in both collections must be the same for the collections to be considered the same.

### Example

```
TCL_LEC> set a [find_cfm -instance u* -collection]
collection_0
TCL_LEC> set a [find_cfm -instance ur* -collection]
collection_1
TCL_LEC> compare_collection $a $b -order_dependent
1
```

Notice that here it's the variable name instead of the collection name that needs to be passed as the argument.

## **copy\_collection**

**copy\_collection** <collection>

### **Description**

Returns a copy of the specified base collection. The base collection is not changed.

### **Parameters**

<collection>	Specifies the collection that will be copied
--------------	----------------------------------------------

### **Example**

```
TCL_LEC> copy_collection $a
collection_6
TCL_LEC> query_collection $a
/ud1 /u_iso1 /ur1 /ur2 /ur3
```

## delete\_attribute

```
delete_attribute
 <name>
 [-object_type objtypes]
```

Deletes a user-defined attribute.

### Parameters

<name>	Name of the attribute to delete.
<-object_type objtypes>	Deletes the attribute from the specified design object types.

## **echo\_result**

**echo\_result** [-on | -off]

Turns the command result printing on or off.



## encrypt

**encrypt** *<inputfile> <outputfile>*

Encrypts a Tcl command file.

### Parameters

<code>inputfile</code>	Tcl file that you want to encrypt. This file is left unchanged by this command.
<code>outputfile</code>	Is the output file that is the encrypted form of the input file.

## **exit**

### **exit**

Ends the existing Conformal session and returns to the operating system using the native Tcl exit command.

## find

```
find -<object_type>
 [<patterns> | <object_list> | -OF_ObJects <object_list>]
 [-Filter <condition>]
 [-COLlection]
 [-HIERarchical]
 [-HSC <hierarchical_separator>]
 [-LIMit <integer>]
 [-REGEXP]
 [-SCOPE <pathname>]
 [-SENSitive | -NOSENSitive]
```

Searches for the specified object and returns a design database object handle or list of handles for a design object or list of objects. The Tcl-based `find` and `find_cfm` commands are both useful for searching design information. They can both be used to look up design objects, such as instances, ports, pins, and nets. For gate queries, however, only the `find` command can be used.

## Parameters

-<object_type>	<p>Specifies the type of objects to be returned by this command. Where <code>object_type</code> is one of the following database object types:</p> <p>Conformal Object: <code>-conformal</code></p> <p>Design Object: <code>-design</code>, <code>-instance</code>, <code>-port</code>, <code>-pin</code>, <code>-net</code>, <code>-library</code>, <code>-libcell</code>, and <code>-libpin</code></p> <p>Power intent object: <code>-piaobj</code> (for a list of all supported power intent object options, refer to the man page of <code>REPORT POWER INTENT</code> (a Conformal Low Power Command) or to the "REPORT POWER INTENT" section of the <i>Low Power Reference Guide</i>.</p>
<patterns>	<p>Specifies the name of the object to be queried. The default is <code>*</code> (wildcard).</p>
<object_list>	<p>List of objects. When specified, the <code>find</code> command returns all the objects of type <code>object_type</code> that are members of the <code>object_list</code> and that match the optional <code>-filter</code> condition.</p>

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-OF_OBJects	<p>Queries objects from another query or list of objects.</p> <p>The following lists the supported <code>-&lt;object_types&gt;</code> and objects allowed in <code>-of_objects</code>, in alphabetical order:</p> <table><tr><td>-design</td><td>instance</td></tr><tr><td>-instance</td><td>net pin</td></tr><tr><td>-library</td><td>libcell</td></tr><tr><td>-libcell</td><td>libpin lib instance</td></tr><tr><td>-libpin</td><td>libcell pin</td></tr><tr><td>-net</td><td>pin instance port</td></tr><tr><td>-occr</td><td>sdcstmt</td></tr><tr><td>-pin</td><td>net instance</td></tr><tr><td>-port</td><td>net</td></tr></table>	-design	instance	-instance	net pin	-library	libcell	-libcell	libpin lib instance	-libpin	libcell pin	-net	pin instance port	-occr	sdcstmt	-pin	net instance	-port	net
-design	instance																		
-instance	net pin																		
-library	libcell																		
-libcell	libpin lib instance																		
-libpin	libcell pin																		
-net	pin instance port																		
-occr	sdcstmt																		
-pin	net instance																		
-port	net																		
-Filter <condition>	<p>Allows queries based on attributes. Condition must use the following format: <code>&lt;attribute_name&gt;&lt;operator&gt;&lt;value&gt;</code>. Where operator can be:</p> <p><code>==, !=, &gt;=, &gt;, &lt;=, &lt;, =~, and !~</code></p> <p>Multiple conditions can be combined with the logical operators <code>!</code>, <code>&amp;&amp;</code>, <code> </code>, and parentheses <code>()</code>.</p>																		
-COLlection	<p>Collects the results of the find command as a Conformal collection object type. This option is useful when querying for a large amount of data. The <code>-collection</code> option creates a handle to the C language that is not restricted to the Tcl variable size. See examples section for more information.</p>																		
-HIERarchical	<p>This option can be used only with the <code>pattern</code> option. This option specifies that the pattern should be matched hierarchically. For example, if the pattern matches the hierarchical name of an instance, it is included in the result.</p>																		
-HSC <hierarchical_separator>	<p>Specifies a user-defined hierarchical separator. By default, Conformal uses <code>"/</code>.</p>																		
-LIMit <int>	<p>Specifies a limit on the number of returned objects.</p>																		

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-REGEXP	Specifies a regular expression for pattern matching. Curly brackets are required. For example, <code>find -instance -regexp {my_instance_\w+}</code> .
-SCOPE <pathname>	Specifies the root of the searching tree. By default, Conformal uses the root of the design tree.
-SENSitive	Specifies that the search is case sensitive. This is the default.
-NOSENSitive	Specifies that the search is case insensitive.

## Examples

For example, to find all power domain objects:

```
find -piaobj -filter "type_name==power_domain"
```

The following generates a collection using the results of a find command:

```
TCL_SETUP> find -instance xL* -collection
collection_0
```

Where `collection_0` is the name of the collection, which is auto-generated by the tool. Conformal auto increments the name for generated collections.

To view the contents of a collection, first assign the collection to a variable and use the `query_collection` Tcl command to view the contents:

```
TCL_SETUP> set z [find -instance xL* -collection]
collection_1
TCL_SETUP> query_collection $z
xLog
```

To find the supply set power intent objects specified in the power intent, see the example below:

```
TCL_SETUP> find -supply_set
SS_PDcore SS_PD09 SS_PD1 SS_PD2
```

To find a supply set power intent objects with specific pattern in the power intent:

```
TCL_SETUP> find -supply_set *PD1*
SS_PD1
```

## find\_cfm

```
find_cfm -<object_type>
 [<patterns> | <object_list> | -OF_OBjects <object_list>]
 [-Filter <condition>]
 [-COLlection]
 [-HIERarchical]
 [-HSC <hierarchical_separator>]
 [-LIMit <int>]
 [-REGEXP]
 [-SCOPE <pathname>]
 [-SENSitive | -NOSENSitive]
```

Searches for the specified object and returns a design database object handle or list of handles for a design object or list of objects. The Tcl-based `find` and `find_cfm` commands are both useful for searching design information. They can both be used to look up design objects, such as instances, ports, pins, and nets. The `find_cfm` command, however, is the only choice for low power queries as it can also query for power intent objects, Liberty library cells, and 1801 rule filters.

### Parameters

-<object_type>	<p>Specifies the type of objects to be returned by this command. Where <code>object_type</code> is one of the following database object types: <code>-conformal</code>, <code>-design</code>, <code>-instance</code>, <code>-port</code>, <code>-pin</code>, <code>-net</code>, <code>-library</code>, <code>-libcell</code>, and <code>-libpin</code>.</p> <p>Power intent object: <code>-piaobj</code> (for a list of all supported power intent object options, refer to the man page of <code>REPORT POWER INTENT</code> (a Conformal Low Power command)).</p>
<patterns>	Specifies the name of the object to be queried. The default is <code>*</code> (wildcard)
<object_list>	List of objects. When specified, the <code>find_cfm</code> command returns all the objects of type <code>object_type</code> that are members of the <code>object_list</code> and that match the optional <code>-filter</code> condition.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-OF_OBJects	<p>Queries objects from another query or list of objects. The following lists the supported <code>-&lt;object_types&gt;</code> and objects allowed in <code>-of_objects</code>:</p> <table><tr><td>-design</td><td>instance</td></tr><tr><td>-instance</td><td>net pin</td></tr><tr><td>-library</td><td>libcell</td></tr><tr><td>-libcell</td><td>libpin lib instance</td></tr><tr><td>-libpin</td><td>libcell pin</td></tr><tr><td>-net</td><td>pin instance port</td></tr><tr><td>-pin</td><td>net instance</td></tr><tr><td>-port</td><td>net</td></tr></table>	-design	instance	-instance	net pin	-library	libcell	-libcell	libpin lib instance	-libpin	libcell pin	-net	pin instance port	-pin	net instance	-port	net
-design	instance																
-instance	net pin																
-library	libcell																
-libcell	libpin lib instance																
-libpin	libcell pin																
-net	pin instance port																
-pin	net instance																
-port	net																
-Filter <condition>	<p>Allows queries based on attributes. Condition must use the following format:</p> <p><code>&lt;attribute_name&gt;&lt;operator&gt;&lt;value&gt;</code>.</p> <p>Where operator can be: <code>==</code>, <code>!=</code>, <code>&gt;=</code>, <code>&gt;</code>, <code>&lt;=</code>, <code>&lt;</code>, <code>=~</code>, and <code>!~</code>. Multiple conditions can be combined with the logical operators <code>!</code>, <code>&amp;&amp;</code>, <code>  </code>, and parentheses <code>()</code>.</p> <p>To report all valid attributes for an object, use <code>REPORT TCL ATTRIBUTES</code>.</p>																
-COLlection	<p>Collects the results of the find command as a Conformal collection object type. This option is useful when querying for a large amount of data. The <code>-collection</code> option creates a handle to the C language that is not restricted to the Tcl variable size. See examples section for more information.</p>																
-HIERarchical	<p>This option can be used only with the pattern option. This option specifies that the pattern should be matched hierarchically. For example, if the pattern matches the hierarchical name of an instance, it is included in the result.</p>																
-HSC <hierarchical_separator>	<p>Specifies a user-defined hierarchical separator. By default, Conformal uses <code>"/"</code>.</p>																
-LIMit <int>	<p>Specifies a limit on the number of returned objects.</p>																

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-REGEXP	Specifies a regular expression for pattern matching. Curly brackets are required.
-SCOPE <pathname>	Specifies the root of the searching tree. By default, Conformal uses the root of the design tree.
-SENSitive	Specifies that the search is case sensitive. This is the default.
-NOSENSitive	Specifies that the search is case insensitive.

## Examples

For example, the following code uses a regular expression to find all the instances whose name starts with RAM under the instance of DTMF\_INST.

```
TCL_SETUP> find_cfm -instance -regexp {/DTMF_INST/RAM\w+} /DTMF_INST/
RAM_128x16_TEST_INST /DTMF_INST/RAM_256x16_TEST_INST
```

For example, to find all blackbox instances:

```
find_cfm -instance -hierarchical -filter "is_bbox==1"
```

The following generates a collection using the results of a find command:

```
TCL_SETUP> find -instance xL* -collection
collection_0
```

Where collection\_0 is the name of the collection, which is auto-generated by the tool. Conformal auto increments the name for generated collections.

To view the contents of a collection, first assign the collection to a variable and use the query\_collection Tcl command to view the contents:

```
TCL_SETUP> set z [find -instance xL* -collection]
collection_1
TCL_SETUP> query_collection $z
xLog
```



## foreach\_in\_collection

```
foreach_in_collection <var>
 <collection>
 <body>
```

### Description

Iterate through all objects in the collection and apply the commands in the script provided as <body>.

### Parameters

<var>	the variable used to iterate through the collection.
<collection>	Specifies the collection name.
<body>	Specifies the subsequent commands that are going to be applied to each item in the collection.

### Example

```
TCL_SETUP> foreach_in_collection y $z {puts $y}
u_gsw
u1
```

## **get\_current\_module**

**get\_current\_module**

Returns the `MODULE` handle for the current module.

## get\_fanin

### get\_fanin

```
<object>
<-logic | -lowpower [-stop_at_buf] [-stop_at_inv] [-stop_at_lp_cell]>
[-hierarchical]
[-filter <filter_condition>]
[-nonets_included | -nets_included]
[-start_points]
[-instance_levels <instance_depth> [-step_into_hierarchy]
 | -pin_levels <pin_depth> [-step_into_hierarchy]]
[-count_leaf | -count_hier | -count_all]
[-to <start_points>]
[-ignore_constants | -apply_constants]
```

Performs traversal to access the objects in the fanin of a target point. If the specified target point is an inout pin, traversal is run in the input direction followed by traversal in the output direction.

Traversal returns a list of objects in the fanin of the specified target point. By default, objects whose object type is port, pin, or instance are returned. In the returned list, objects are unique. The order of objects whose object type is port, pin, or instance are constructed based on the object levels.

## Parameters

<object>	Specifies the target point to be used in traversal. This object type can be port, pin, net, or instance.
-logic	Specifies logic cone traversal. Timing arcs are not used to traverse black-boxed modules. This argument is mandatory in Conformal Low Power/Verify.
-lowpower	Specifies low-power path traversal. In this traversal, paths start and end at logic leaf cells and primary ports, and traverse buffers, inverters, and low-power cells. The supply network is not traversed. This option is only available in Conformal Low Power.
-stop_at_buf	Specifies that low-power traversal should stop at buffers.
-stop_at_inv	Specifies that low-power traversal should stop at inverters.
-stop_at_lp_cell	Specifies that low-power traversal should stop at isolation and level-shifter cells.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-hierarchical	Specifies hierarchical mode for operation mode to be used in traversal.
-filter <filter_condition>	<p>Allows query based on attributes. The following operators are supported: ==, !=, &gt;=, &gt;, &lt;=, &lt;, =~, and !~</p> <p>Multiple &lt;filter_condition&gt; can be combined with &amp;&amp;,    with parenthesis ().</p>
-nonets_included	Specifies that object types of net cannot be returned. <i>This is the default.</i>
-nets_included	Specifies that object types of net are returned.
-start_points	Specifies that returned objects must be valid startpoints with respect to the paths to the specified target point.
-instance_levels <instance_depth>	<p>Stops traversal when the specified depth of search of instances is reached in object-level counting. Object-level counting is run over instances along the traversal paths.</p>
-pin_levels <pin_depth>	<p>Stops traversal when the specified depth of search of pins is reached in object-level counting. Object-level counting is run over pins along the traversal paths.</p>
-step_into_hierarchy	<p>Allows object-level counting in non-hierarchical mode to be run over leaf objects at a hierarchy level lower than that of the specified target point.</p> <p><b>Note:</b> This option must be used with -instance_levels or -pin_levels.</p>
-count_leaf	Specifies only leaf objects to be counted in object-level counting. <i>This is the default.</i>
-count_hier	Specifies only hierarchical objects to be counted in object-level counting.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

<code>-count_all</code>	Specifies both leaf and hierarchical objects to be counted in object-level counting.  <b>Note:</b> The above three <code>-count</code> options must be used with <code>-instance_levels</code> or <code>-pin_levels</code> , and <code>-hierarchical</code> or <code>-step_into_hierarchy</code> .
<code>-to &lt;start_points&gt;</code>	Restricts the fanin cone to objects in the fanout of the specified start points.
<code>-ignore_constants</code>	Specifies that the effect of constant propagations is ignored. <i>This is the default.</i>
<code>-apply_constants</code>	Specifies that the traversal stops at objects with propagated constant value.

## Examples

- The following commands run hierarchical and non-hierarchical structural fanin traversal for an output port. The fanin includes two leaf instances of buffer cells (`buf03` and `buf00`) and a hierarchical instance (`sub`), which contains two leaf instances of buffer cells (`sub/buf02` and `sub/buf01`):

```
get_fanin [find -port {q[0]}] -hierarchical \
-filter {object_type == port || object_type == pin}
```

The returned list includes

```
{ buf03/Y buf03/A {sub/sub_q[0]} sub/buf02/Y sub/buf02/A sub/buf01/Y sub/
buf01/A {sub/sub_d[0]} buf00/Y buf00/A {d[0]} }
```

For this command:

```
get_fanin [find -port {q[0]}] \
-filter {object_type == port || object_type == pin}
```

The returned list includes:

```
{ buf03/Y buf03/A {sub/sub_q[0]} {sub/sub_d[0]} buf00/Y buf00/A {d[0]} }
```

- The following command runs non-hierarchical timing fanin traversal for an output port. The fanin includes a leaf instance of a sequential cell (`reg03`):

```
get_fanin [find -port {q[1]}] \
-filter {object_type == port || object_type == pin} -timing
```

The returned list includes the following:

```
{ reg03/Q reg03/CLK }
```

## get\_fanout

### get\_fanout

```
<object>
<-logic | -lowpower [-stop_at_buf] [-stop_at_inv] [-stop_at_lp_cell]>
[-hierarchical]
[-filter <filter_condition>]
[-nonets_included | -nets_included]
[-end_points]
[-instance_levels <instance_depth> [-step_into_hierarchy]
 | -pin_levels <pin_depth> [-step_into_hierarchy]]
[-count_leaf | -count_hier | -count_all]
[-to <end_points>]
[-ignore_constants | -apply_constants]
```

Performs traversal to access the objects in the fanout of a target point. If the specified target point is an inout pin, traversal is run in the input direction followed by traversal in the output direction.

Traversal returns a list of objects in the fanout of the specified target point. By default, objects whose object type is port, pin, or instance are returned. In the returned list, objects are unique. The order of objects whose object type is port, pin, or instance are constructed based on the object levels.

## Parameters

<object>	Specifies the target point to be used in traversal. This object type can be port, pin, net, or instance.
-logic	Specifies logic cone traversal. Timing arcs are not used to traverse blackboxed modules.
-lowpower	Specifies low-power path traversal. In this traversal, paths start and end at logic leaf cells and primary ports, and traverse buffers, inverters, and low-power cells. The supply network is not traversed. This option is only available in Conformal Low Power.
-stop_at_buf	Specifies that low-power traversal should stop at buffers.
-stop_at_inv	Specifies that low-power traversal should stop at inverters.
-stop_at_lp_cell	Specifies that low-power traversal should stop at isolation and level-shifter cells.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-hierarchical	Specifies hierarchical mode for operation mode to be used in traversal.
-filter <filter_condition>	<p>Allows query based on attributes. The following operators are supported:</p> <p><code>==, !=, &gt;=, &gt;, &lt;=, &lt;, =~, and !~</code></p> <p><b>Note:</b> Multiple &lt;filter_condition&gt; can be combined with <code>&amp;&amp;</code>, <code>  </code> with parenthesis <code>()</code>.</p>
-nonets_included	Specifies that object types of net cannot be returned. <i>This is the default.</i>
-nets_included	Specifies that object types of net are returned.
-end_points	Specifies that returned objects must be valid endpoints with respect to the paths from the specified target point.
-instance_levels <instance_depth>	<p>Stops traversal when the specified depth of search of instances is reached in object-level counting. Object-level counting is run over instances along the traversal paths.</p>
-pin_levels <pin_depth>	<p>Stops traversal when the specified depth of search of pins is reached in object-level counting. Object-level counting is run over pins along the traversal paths.</p>
-step_into_hierarchy	<p>Allows object-level counting in non-hierarchical mode to be run over leaf objects at a hierarchy level lower than that of the specified target point.</p> <p><b>Note:</b> This option must be used with <code>-instance_levels</code> or <code>-pin_levels</code>.</p>
-count_leaf	Specifies only leaf objects to be counted in object-level counting. <i>This is the default.</i>
-count_hier	Specifies only hierarchical objects to be counted in object-level counting.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

<code>-count_all</code>	Specifies both leaf and hierarchical objects to be counted in object-level counting.  <b>Note:</b> The above three <code>-count</code> options must be used with <code>-instance_levels</code> or <code>-pin_levels</code> , and <code>-hierarchical</code> or <code>-step_into_hierarchy</code> .
<code>-to &lt;end_points&gt;</code>	Restricts the fanout cone to objects in the fanin of the specified end points.
<code>-ignore_constants</code>	Specifies that the effect of constant propagations is ignored. <i>This is the default.</i>
<code>-apply_constants</code>	Specifies that the traversal stops at objects with propagated constant value.

## Examples

- The following command runs hierarchical and non-hierarchical structural fanout traversal for an input port. The fanout includes two leaf instances of buffer cells (`buf00` and `buf03`) and a hierarchical instance (`sub`) which contains two leaf instances of buffer cells (`sub/buf01` and `sub/buf02`):

```
get_fanout [find -port {d[0]}] -hierarchical \
-filter {object_type == port || object_type == pin}
```

The returned list includes:

```
{ buf00/A buf00/Y {sub/sub_d[0]} sub/buf01/A sub/buf01/Y sub/buf02/A sub/
buf02/Y {sub/sub_q[0]} buf03/A buf03/Y {q[0]} }
```

For this command:

```
get_fanout [find -port {d[0]}] \
-filter {object_type == port || object_type == pin}
```

The returned list includes:

```
{ buf00/A buf00/Y {sub/sub_d[0]} {sub/sub_q[0]} buf03/A buf03/Y {q[0]} }
```

- The following runs non-hierarchical timing fanout traversal for an input port. The fanout includes a leaf instance of a sequential cell (`reg00`):

```
get_fanout [find -port {d[1]}] \
-filter {object_type == port || object_type == pin} -logic
```

The returned list includes:

```
{ reg00/D }
```



## get\_handle\_type

**get\_handle\_type** <obj\_handle>

Returns the object type of the specified object handle. The returned result is one of the following strings:

- MODULE
- MODULE\_INSTANCE
- MODULE\_PORT
- MODULE\_INSTANCE\_PIN
- MODULE\_NET
- HIERARCHY\_INSTANCE
- HIERARCHY\_PORT
- HIERARCHY\_INSTANCE\_PIN
- HIERARCHY\_NET

## get\_instances

**get\_instances** [-all\_hierarchy] <obj\_handle>

Queries for multiple objects and returns a list of instances associated with the specified object handle.

---

-all_hierarchy	Use this argument in combination with the <code>HIERARCHY_NET</code> object handle type. It returns a list of hierarchical instance object handles that are associated with the given hierarchical net object.
<obj_handle>	<p>The specified <code>obj_handle</code> as one of the following types:</p> <p><code>MODULE</code>: lists the instances in the module</p> <p><code>MODULE_INSTANCE</code>: lists the instance that identifies itself</p> <p><code>MODULE_PORT</code>: not applicable</p> <p><code>MODULE_INSTANCE_PIN</code>: lists the instances whose pins are listed in the specified object handle</p> <p><code>MODULE_NET</code>: lists the instances that connect with the net</p> <p><code>HIERARCHY_INSTANCE</code>: lists the hierarchical instance that identifies itself</p> <p><code>HIERARCHY_PORT</code>: lists the hierarchical instance of which pins include this pin</p> <p><code>HIERARCHY_INSTANCE_PIN</code>: lists the hierarchical instance of which pins include this pin</p> <p><code>HIERARCHY_NET</code>: lists the hierarchical instances that connect with the net in a hierarchical context</p>

---

## **get\_license\_mode**

**get\_license\_mode**

Returns the current license mode. For example, `custom` (also known as GXL), `ultra` (also known as XL), `asic` (also known as L), `lp`, `rcv`, `verify`, and `ccd`.

## **get\_module\_definition**

**get\_module\_definition** <obj\_handle>

Returns a module definition for the specified object handle, where <obj\_handle> is one of the following types:

- MODULE: define itself
- MODULE\_INSTANCE: define the module of the specified instance
- MODULE\_PORT: not applicable
- MODULE\_INSTANCE\_PIN: not applicable
- MODULE\_NET: not applicable
- HIERARCHY\_INSTANCE: define the module of the specified instance in a hierarchical context.
- HIERARCHY\_PORT: not applicable
- HIERARCHY\_INSTANCE\_PIN: not applicable
- HIERARCHY\_NET: not applicable

## **get\_names**

**get\_names** <obj\_handles>

Queries for multiple objects and returns a name or list of names for the specified object handles.

## get\_nets

**get\_nets** [-all\_hierarchy] <obj\_handle>

Queries for multiple objects and returns a list of nets for the specified object handle.

---

-all_hierarchy	Use this argument in combination with the <code>HIERARCHY</code> object handle type. It returns a list of hierarchical net object handles that are associated with the given hierarchical design object (for example, <code>HIERARCHY_INSTANCE</code> ).
----------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

<obj_handle>	<p>The specified <code>obj_handle</code> as one of the following types:</p> <p><code>MODULE</code>: lists the nets of the specified module</p> <p><code>MODULE_INSTANCE</code>: lists the nets that connect with the specified instance</p> <p><code>MODULE_PORT</code>: lists the nets that connect with the specified port</p> <p><code>MODULE_INSTANCE_PIN</code>: lists the nets that connect with the specified pin</p> <p><code>MODULE_NET</code>: lists the net that identifies itself</p> <p><code>HIERARCHY_INSTANCE</code>: lists the nets that connect with the specified instance in a hierarchical context</p> <p><code>HIERARCHY_PORT</code>: lists the nets that connect with the specified port in a hierarchical context</p> <p><code>HIERARCHY_INSTANCE_PIN</code>: lists the nets that connect with the specified pin in a hierarchical context</p> <p><code>HIERARCHY_NET</code>: lists the net that identifies itself</p>
--------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---

## get\_parent

**get\_parent** <obj\_handle>

Returns a handle to the parent of the specified object handle, where <obj\_handle> is one of the following types:

- MODULE: not applicable
- MODULE\_INSTANCE: return the module that contains the specified instance
- MODULE\_PORT: return the module that contains the specified port
- MODULE\_INSTANCE\_PIN: return the module that contains the specified pin
- MODULE\_NET: return the module that contains the specified net
- HIERARCHY\_INSTANCE: return the module definition of the instance
- HIERARCHY\_PORT: return the hierarchical instance that contains the specified port in a hierarchical context
- HIERARCHY\_INSTANCE\_PIN: return the hierarchical instance that contains the specified pin in a hierarchical context
- HIERARCHY\_NET: return the hierarchical instance that contains the specified net in a hierarchical context

## get\_pins

**get\_pins** [-all\_hierarchy] <obj\_handle>

Queries for multiple objects and returns a list of pins associated with the specified object handle.

---

-all_hierarchy	Use this argument in combination with the HIERARCHY_NET object handle type. It returns a list of hierarchical instance pin object handles that are associated with the given hierarchical net object.
<obj_handle>	<p>The specified <code>obj_handle</code> as one of the following types:</p> <p>MODULE: lists the pins of the specified module</p> <p>MODULE_INSTANCE: lists the pins of the specified instance</p> <p>MODULE_PORT: lists the pins that connect with the net of the specified port</p> <p>MODULE_INSTANCE_PIN: lists the pin that identifies itself</p> <p>MODULE_NET: lists the pins that connect with the specified net</p> <p>HIERARCHY_INSTANCE: lists the pins of the specified instance in a hierarchical context</p> <p>HIERARCHY_PORT: lists the pins that connect with the net of the specified port in a hierarchical context</p> <p>HIERARCHY_INSTANCE_PIN: lists the nets that connect with the specified pin in a hierarchical context</p> <p>HIERARCHY_NET: lists the pin that identifies itself</p>

---



## get\_ports

**get\_ports** [-all\_hierarchy] <obj\_handle>

Queries for multiple objects and returns a list of ports associated with the specified object handle.

---

-all_hierarchy	Use this argument in combination with the HIERARCHY_NET object handle type. It returns a list of hierarchical port object handles that are associated with the given hierarchical net object.
<obj_handle>	<p>The specified <code>obj_handle</code> as one of the following types:</p> <p>MODULE: lists the ports of the specified module</p> <p>MODULE_INSTANCE: lists the ports of the specified instance</p> <p>MODULE_PORT: lists the port that identifies itself</p> <p>MODULE_INSTANCE_PIN: returns the relative port of the specified pin</p> <p>MODULE_NET: lists the ports that connect with the specified net</p> <p>HIERARCHY_INSTANCE: lists the ports of the specified instance in a hierarchical context</p> <p>HIERARCHY_PORT: lists the port that identifies itself</p> <p>HIERARCHY_INSTANCE_PIN: returns the relative port of the specified pin in a hierarchical context</p> <p>HIERARCHY_NET: lists the ports that connect with the specified net in a hierarchical context</p>

---

## get\_primitive\_type

**get\_primitive\_type** <obj\_handle>

Returns the primitive type for the specified instance object handle, where <obj\_handle> is one of the following:

- MODULE\_INSTANCE
- HIERARCHY\_INSTANCE

## get\_property

**get\_property** <obj\_handle> <property\_type>

Returns the property of the specified object handle. The following lists the object handle one of the following types, with their property types and return values.

■ **MODULE**

Property Type: `BlackBox`; Return Value: YES, NO

Property Type: `InLib`; Return Value: YES, NO

Include the `InLib` property to test whether the module definition is defined in the library space (YES) or design space (NO)—that is, it tests whether the module definition is imported by the `READ LIBRARY` or `READ DESIGN` command.

■ **MODULE\_INSTANCE**

Property Type: `Primitive`; Return Value: YES, NO

Property Type: `CutPoint`; Return Value: YES, NO

Include the `CutPoint` property to test whether the module definition is a cut point—that is, it tests whether you have run the `ADD CUT POINT` command on this module definition.

■ **MODULE\_PORT**

Property Type: `Direction`; Return Value: INPUT, OUTPUT, BIDIR

■ **MODULE\_INSTANCE\_PIN**

Property Type: `Direction`; Return Value: INPUT, OUTPUT, BIDIR

■ **MODULE\_NET**

Property Type: `NetType`; Return Value: TIE0, TIE1, WAND, WOR, TRI, TRI0, TRI1, TRIAND, TRIOR, TRIREG, REG, GLOBAL, TIEX, TIEZ, WIRE

**Note:** GLOBAL applies to those signals that cross multiple modules (for example, assertions).

■ **HIERARCHY\_INSTANCE**

Property Type: `BlackBox`; Return Value: YES, NO

■ **HIERARCHY\_PORT**

Property Type: `Direction`; Return Value: INPUT, OUTPUT, BIDIR

■ **HIERARCHY\_INSTANCE\_PIN**

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

**Property Type:** `Direction`; **Return Value:** `INPUT`, `OUTPUT`, `BIDIR`

#### ■ HIERARCHY\_NET

**Property Type:** `NetType`; **Return Value:** `TIE0`, `TIE1`, `WAND`, `WOR`, `TRI`, `TRI0`, `TRI1`, `TRIAND`, `TRIOR`, `TRIREG`, `REG`, `GLOBAL`, `TIEX`, `TIEZ`, `WIRE`

**Note:** `GLOBAL` applies to those signals that cross multiple modules (for example, assertions).

## get\_relative\_path

**get\_relative\_path** <full\_file\_path>

Returns an equivalent file path that is relative to the current working directory.

---

full\_file\_path

Specifies the path for which to return a relative path.

For example, assuming the current working directory is /home/conformal:

The result of the command "get\_relative\_path /home/conformal/foo/bar" is the relative path "foo/bar".

The result of the command "get\_relative\_path /home/library/etc/mylib.lib" is the relative path "../library/etc/mylib.lib".

---

## **get\_root\_module**

`get_root_module`

Returns the name of the root module for the specified design.

## get\_supply\_sets

### get\_supply\_sets

```
[-<func> <supply_net_or_port>]
[-complete]
[-explicit_only | -implicit_only]
[-used_only] |
[-no_equivalence]
[-filter <expr>]
[-limit <n>]
[-collection]
```

NOTE: This is a Conformal Low Power Tcl command.

Each function can only be specified once. This command returns a list or collection of supply set objects, as specified by the following options:

-power, -ground, -nwell, -pwell, -deepnwell, -deppwell

Only supply sets that have the specified supply object associated to the given function will be returned.

<func>	Returns supply sets for the specified supply set object. <func> can be: power, ground, nwell, pwell, deepnwell, or deppwell.
-complete	Only return supply sets that are completely described by the specified functions. They do not have any additional non-null function defined in the power intent.
-explicit_only	Only return supply sets that are defined in power intent.
-implicit_only	Only return supply sets that are defined by the tool.
-used_only	Only return supply sets referenced in power intent and/or assigned as driver/receiver supply to design ports.
-no_equivalence	When matching -<func> objects to supply set functions, an exact match will be required: equivalence is ignored.
-filter	Filter the returned objects using an expression of their attributes.
-limit	Return at most the specified number of objects. The default is 0, which means that all the objects are returned.
-collection	Return a Tcl collection rather than a Tcl list.

#### Example

```
get_supply_sets -power [find -port VDD] -ground [find -port VSS]
--> supply_set_VDD_VSS supply_set_VDD_VSS_VNW
get_supply_sets -power [find -port VDD] -ground [find -port VSS] -complete
--> supply_set_VDD_VSS
```



## **get\_version\_info**

### **get\_version\_info**

Returns a TCL list of the version related info, including `version_num`, `build_date`, 32 or 64 bit, host name, and platform.

## index\_collection

**index\_collection** <collection> <index>

### Description

Returns a object that exists at the specified index of the specified object collection.

### Parameters

<collection>	Specifies the collection name.
<index>	Specifies the index number. The index starts from 0.

### Example

The following command set returns the object that exists at the index number 2 in the object collection referenced by \$insts:

```
TCL_LEC> set insts [find_cfm -instance * -collection]
collection_0
TCL_LEC> set obj [index_collection $insts 2]
/url
```

## help

**help** [command\_name]

Returns the command usage of the specified command or a list of all Conformal Tcl commands if you do not specify a command.

## ncdecrypt

### **ncencrypt**

`<input_file | -version>`

Decrypts the specified file, which was encrypted using the `ncencrypt` command. Writes the contents into the buffer for the Tcl interpreter to evaluate. This command does not write out to a file.

If the encrypted file uses a user-defined key, you must set the environment variable `NCPROTECT_KEYDB` to the directory that contains the public key needed to decrypt the file before using the `ncdecrypt` command.

`setenv NCPROTECT_KEYDB <path>`

---

<i>input_file</i>	File to decrypt.
<code>-version</code>	Report the NC library version.

---

## **objtype**

`objtype <object_or_handle>`

Returns the type of a native Tcl object, such as a string or list, or a Conformal design object handle like "vpxhandle".

## query\_collection

```
query_collection <collection>
 [-limit <integer>]
```

### Description

Returns as a TCL list all or specified number of objects from a collection of objects

### Parameters

<code>&lt;collection&gt;</code>	Specifies the collection that will be sorted
---------------------------------	----------------------------------------------

### Example

Please refer to the example of `sort_collection`.

## query\_state

### query\_state

[-boolean | -state]

(For logic pins or supply set objects)

```
<-off_on | -on_off> <obj1> <obj2> [| -power | -ground]
| <-hi_lo | -lo_hi> <obj1> <obj2> [| -power | -ground] [-tolerance <volts>]
| <-off | -on> <obj> [| -power | -ground]
```

(For supply port/net objects)

```
<-off_on | -on_off> <obj1> <obj2>
| <-hi_lo | -lo_hi> <obj1> <obj2> [-tolerance <volts>]
| <-off | -on> <obj>
```

This command checks if there is any consistent power state that satisfies the specified condition. It can return a boolean value (1 if a state is found, 0 otherwise) or the name of a witness state, if any.

The available options depend on the type of objects being queried. When a logic pin is specified, the query refers to that pin's related supply set. When a supply port/net is specified, the query refers to its own states.

NOTE: PARTIAL\_ON is treated as OFF in this command.

-boolean	This is the default. The command returns 1 if a state is found, otherwise it returns 0.
-state	The command returns a witness state name, or empty string if none is found.
-off_on	Search for a state where <obj1> is OFF when <obj2> is ON.
-on_off	Search for a state where <obj1> is ON when <obj2> is OFF.
-hi_lo	Search for a state where <obj1> is at a higher voltage than <obj2>.
-lo_hi	Search for a state where <obj1> is at a lower voltage than <obj2>.
-power	Specify that only the power of a supply set should be checked. The default is to check all the functions.
-ground	Specify that only the ground of a supply set should be checked. The default is to check all the functions.

## Conformal Extended Checks Reference Manual

### Tcl Command Entry Mode Support

---

-tolerance	Specify a tolerance for the voltage check. A power state will satisfy the query only if the objects checked have a voltage difference above the specified tolerance (in absolute value).
-off	Search for a state where <obj> is OFF.
-on	Search for a state where <obj> is ON.

---

#### Examples:

```
set driver [find -pin u1/y]
set load [find -pin u2/a]
set is_on_crossing [query_state -off_on $driver $load]
set is_overdriven [query_state -hi_lo $driver $load -tolerance 2]
set sset [find -supply_set SS1]
set ss_is_switchable [query_state -off $sset]
set some_on_state [query_state -on $sset -state]
```



## redirect

### **redirect**

`[-append] [-variable] <target> <command>`

Redirects output from a specified command to a file or Tcl variable.

<code>-append</code>	Appends the generated output to the specified Tcl variable or file.
<code>-variable</code>	Redirects output to a Tcl variable. If <code>-variable</code> is not specified, a file is created with the specified target name.
<code>&lt;target&gt;</code>	Redirects output from the specified command to this file/variable.  If the target file or variable already exists and <code>-append</code> is not specified, the command overwrites the existing variable or file.
<code>&lt;command&gt;</code>	Tcl command to redirect. Commands must be quoted. For example:  <code>redirect -variable gates "report_gate"</code>

## remove\_from\_collection

```
remove_from_collection <collection1>
 <collection2 | object_list>
 [-intersect]
```

### Description

Creates a new collection by removing the objects specified as a collection <collection2> or as a TCL list from collection <collection1>.

### Parameters

<collection1>	Specifies the first collection.
<collection2   object_list>	Specifies the second collection.
-intersect	Specifies reverse matching, the tool removes the items that are not in collection2/object_list from the base collection.

### Example

```
TCL_LEC> set b [find_cfm -instance url]
/url
TCL_LEC> set c [remove_from_collection [find_cfm -instance * -collection] $b]
collection_2
TCL_LEC> query_collection $c
/url /u_iso1 /ur2 /ur3
```

## report\_attribute

```
report_attribute
 [name]
 [-object_type objtypes]
 [-all | -user | -system]
```

Reports on a specific attribute, attributes for a particular object, a list of object types, or an instance of a design object.

### Parameters

<name>	Name of the attribute to delete.
<-object_type objtypes>	Reports attributes for the specified design object types.
-all	Reports all attributes. Default.
-user	Reports user-defined attribute types.
-system	Reports system attributes.

## set\_current\_module

**set\_current\_module** <module\_name | obj\_handle>

Changes the current module. By default, the current module is the root module.

---

<module_name>	The specified name is a module name.
<obj_handle>	This argument is a MODULE handle.

---

## sizeof\_collection

**sizeof\_collection** <collection>

### Description

Returns size of the specified object collection.

### Parameters

<collection>                      Specifies the collection.

### Example

The following example shows how to query the size of a collection:

```
TCL_LEC> set insts [find_cfm -instance * -collection]
collection_0
TCL_LEC> set obj_cnt [sizeof_collection $insts]
5
```

## sort\_collection

**sort\_collection** <collection> [-descending]

### Description

Returns a sorted collection of objects by ascending (default) or descending order

### Parameters

<collection>	Specifies the collection that will be sorted.
<-descending>	Specifies that the output will be descending order.

### Example

```
TCL_LEC> set a [find_cfm -instance u* -collection]
collection_0
TCL_LEC> sort_collection $a
collection_1
TCL_LEC> query_collection $a
/ud1 /u_isol /url /ur2 /ur3
```

## **tcl\_set\_command\_name\_echo**

**tcl\_set\_command\_name\_echo** <OFF | ON>

Turns Tcl command set (including Tcl source and nested Tcl commands) echoing on or off. Default is OFF.

## unset\_attribute

```
unset_attribute
 <name>
 <objects>
```

Resets the value of a user-defined attribute back to its default value.

### Parameters

<name>	Name of the attribute to reset.
<objects>	Specifies the object(s) whose attributes need to be reset.



## **usage**

### **usage**

Displays usage values in Tcl scripts for total CPU run time and peak memory use.